國 立 交 通 大 學

電信工程研究所

博 士 論 文

無線區域網路中控式電源管理技術之研究

Centralized Power Management Techniques for Wireless Local Area Networks

研 究 生：謝景融

指導教授：李程輝 博士

中 華 民 國 九十九 年 九 月

無線區域網路中控式電源管理技術之研究

Centralized Power Management Techniques for Wireless Local Area Networks

研究生：謝景融　　Student: Jing-Rong Hsieh

指導教授：李程輝 博士　　Advisor: Dr. Tsern-Huei Lee

國立交通大學

電信工程研究所

博士論文

A Dissertation
Submitted to Institute of Communication Engineering
College of Electrical and Computer Engineering
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in
Communication Engineering

September 2010

Hsinchu, Taiwan

中華民國九十九年九月

無線區域網路中控式電源管理技術之研究

研究生：謝景融　　指導教授：李程輝 博士

國立交通大學電信工程研究所

摘要

近年來 IEEE 802.11 無線區域網路已成為寬頻無線存取的主要途徑。其使用者在享受各種新穎服務的同時，也期待可攜無線設備的使用時間能符合需求。為減少能量浪費，電源管理是透過適時調降無線設備電源狀態的節能技術。對於共享傳輸媒介且多使用者的無線網路而言，除了依訊務的就緒與否來調節電源狀態，如何滿足個別需求且錯開使用者的服務時間以避免耗費電能的無意聆聽(overhearing)，也是很重要的議題。因此我們聚焦於電源管理排程演算法，考慮變動位元率訊務與錯誤重傳，減少服務期間重疊情形以達節能之目的。

在本論文中，為了支援 802.11e 標準中的節能機制—排程式自動節能遞送(Scheduled Automatic Power Save Delivery)，我們首先提出一個易實現的排程演算法。此排程準則為最大化新加入訊務與現存已排程事件的距離。利用服務排程間距之週期性，能避免相關研究提出的暴力搜尋法(brute-force search)造成的冗餘檢查。此外，基於現實中的訊務種類有限的特點，我們提出事先計算並儲存不同類訊務間的相對關係的演算法，進一步降低實現複雜度。由於前述排程準則為基於系統原有排程找尋新訊務的合適安插位置，因而排程結果與訊務加入順序相關，所得亦為漸進式的最佳化結果。所以我們延伸提出重整系統現存訊務排程的整體最佳化問題，目標為最大化整體系統排程的最小距離。我們設計的重整演算法妥善利用平分排程的低複雜度且保持最小排程間距的上限，可達到良好的節能效果。

為了提高無線頻寬使用效率，多重輪詢(multi-polling)經常被用來降低通訊協定中的虛耗(overhead)。然而，為了維持排序競爭式多重輪詢之正常運作，需要無線站台耗費許多時間在無意聆聽上，導致電能浪費而降低電池可用時間。在本論文中，我們提出一個節能多重輪詢(Energy-Efficient Multi-Polling)機制，結合低虛耗通訊協定與電源管理技術。在此機制中，我們推出一個給定頻寬使用率下的最佳甦醒時間排程(wake-up time schedule)，相較於原先的排序式多重輪詢機制，分析與模擬結果均展示良好的節能表現。在給定損失 5%的頻寬使用率且由 20 個無線站台組成之環境下，可達到 80%的節能改進。此顯著效果來自妥適的甦醒時間排程，大幅避免無意聆聽的情形。最後，我們亦探討無線網路中因錯誤回復(error recovery)造成的節能問題。對於類分時多工的多重輪詢機制，提出有效的甦醒時間排程以及相應於重傳致使延遲的排程更新演算法，增加站台進入低電源狀態的機會。模擬結果顯示採用我們的演算法可顯著改進錯誤重傳發生時的節能效果。

Centralized Power Management Techniques for Wireless Local Area Networks

Student: Jing-Rong Hsieh Advisor: Dr. Tsern-Huei Lee

Institute of Communication Engineering
National Chiao Tung University

Abstract

In the past decade, IEEE 802.11 wireless LANs has gained large popularity in broadband wireless access. Users are demanding high performances while keeping respectable operation time for the mobile devices. Power management (PM) is an essential technique for energy saving by putting devices into low power state during appropriate interval. For a multi-user and shared medium wireless network, in addition to managing power state according to readiness of traffic, it is important to separate the usage time of different users to prevent energy-consuming overhearing. Hence, in this dissertation, considering variable-bit-rate traffic and unpredictable error recovery, we focus on the scheduling algorithms to reduce the chance of service period overlapping.

To support standardized power saving mechanism in IEEE 802.11e, we propose a feasible scheduling algorithm for the Scheduled Automatic Power Save Delivery. The goal is to maximize the minimum distance between the scheduled instants of new joining traffic stream (TS) and existing scheduled events (SEs). By the proven periodicity of service schedules, the redundant check in the previous brute-force method can be avoided. Moreover, considering limited number of classes for TSs, we can pre-calculate and store necessary information to further reduce the implementation complexity. Extending the idea of finding the optimal service start time for new joining TS incrementally, we also study the rearrangement of existing SEs to further maximize the system minimum distance. We prove the upper bound of the system minimum distance and design efficient rearranging algorithms to achieve satisfactory energy saving.

In order to achieve higher system bandwidth utilization (BU), multi-polling mechanisms are often employed to reduce protocol overhead. However, they may require wireless stations (STAs) to spend much time in overhearing. We propose an energy-efficient multi-polling mechanism which combines PM strategy with a low overhead Medium Access Control protocol. Given a desirable guarantee of BU, an energy optimized wake-up time schedule (WTS) is devised. Significant saving of energy can be obtained with only small loss of BU as trade-off. It is the consequence of alleviating the overhearing problem by well scheduled WTSs for STAs. In the end, we also study the energy saving issue induced from error recovery. A WTS and a renewal algorithm in correspondence with the delay caused by retransmissions are proposed for the TDMA-like multi-polling mechanism. Simulation results show that, compared with the original setting, significant improvement can be obtained by the proposed algorithms.

Acknowledgements

I am very fortunate to have Professor Dr. Tsern-Huei Lee as my supervisor during my graduate studies. For these years, his enthusiasm for research and dedication to my professional developments are truly inspiring. I am deeply grateful to Prof. Lee for his continuous encouragement, support, valuable suggestions, and discussions to help me complete this work. I would also appreciate each committee member of my dissertation defence. Their valuable comments give this dissertation considerable polish.

I would like to show my gratitude to my dear friends in Network Technology Laboratory and in Department of Communication Engineering for their stimulating discussion, warm care, and assistance throughout my study.

Special thanks are given to National Science Council for the financial supports which made this work possible. I am thankful to Deutsch Akademischer Austausch Dienst (DAAD, German Academic Exchange Service) which offered me a great opportunity and scholarship to experience the research environment in Germany. I would also like to thank National Chiao Tung University for providing such a wonderful environment for studying and doing researches.

In the end, I would like to dedicate this dissertation to my family for their love, encouragement, and long expectation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

The IEEE 802.11 [1] wireless local area network (wireless LAN) has been widely spread due to its low cost and easy installation. One can easily find wireless LAN hot spots in most places of a modern city such as office, campus, cafe, or even on the street. Therefore, including popular laptop computers and Netbooks, more and more mobile devices integrate wireless LAN functionality inside for Internet accessing. As the hardware performance of the mobile devices is greatly improved and many useful features such as location-based service and the notion of cloud computing are introduced, it is more likely for people to access the Internet anytime and anywhere through their mobile devices.

Over the past decade, the popularity of wireless LAN has also driven many research works and enhancements to the original standard, such as throughput, quality of service (QoS), and security. Due to high protocol overhead, the throughput performance of 802.11 wireless LAN is much worse than the underlying physical layer (PHY) transmission rate. It has been proved in [14] that there is theoretical upper limit of achievable throughput for IEEE 802.11 protocol. The upper limit depends on payload length and is about 75 Mbps for 802.11a with 1500-byte payload length. To increase system throughput, the IEEE 802.11n task group is initiated to achieve a maximum throughput of at least 100 Mbps, as measured at the MAC data service access point, by both PHY and medium access control (MAC) enhancements

and the specification [3] was finalized in the end of 2009. Despite the goal to provide QoS in wireless LAN, IEEE 802.11e [2] defines some features which also improves the system bandwidth utilization. The introduction of transmission opportunity (TXOP), Block ACK (*acknowledgement*), and direct link protocol can be utilized to reduce some MAC overhead. For centralized channel coordination, to further improve bandwidth utilization, multi-polling schemes [15]– [17] and [18] were proposed to reduce overhead caused by possibly constant polling frames.

While emerging researches and standards are pursuing high performance and high capacity wireless links, the required power consumptions of mobile devices are also increased quickly. The resulted short battery lifetime and cost for cooling the power hungry system could become disturbances to mobile device users [9]. It had been noted that the battery technology has no significant advances for decades [29]. Comparing with the increase of processor power consumption of 200 % every four year, a modest 25 % increase in the battery energy density is observed for the same interval [7]. It is also revealed in [10] that wireless network interface could consume upwards of 30 % of scarce portable system energy. Since most wireless devices are battery powered, many techniques have been proposed to reduce the power consumption of wireless network interface, such as transmission power control [12], [13], and power management. Regarding the power management method, powering down the transceiver can lead to great power savings even during active sessions since the consumed power of the Awake state and the Doze state differ much. For example, the Lucent IEEE 802.11 WaveLAN card [22] consumes 1.65 W, 1.4 W and 1.15 W in the transmit, receive and idle modes, respectively, in the Awake state. In the Doze state, it consumes 0.045 W which is much less than any mode of the Awake state. Most previous works [21]- [25] consider ad hoc scenarios because devices in an infrastructure system are usually connected to a AC power supply or equipped with long life battery. The situation has been changed for current multi-mode mobile devices because their small size and weight severely limits the battery size and capacity. Therefore, it is worth to study the techniques to save the energy

of wireless devices even in the infrastructure mode.

In [5], the major sources of energy waste of shared medium wireless networks were listed:

1. ***collision***: Transmission collision obviously is a source of energy waste because frames involved in a collision are destroyed. Moreover, the subsequent retransmissions also require additional energy consumption.

2. ***overhearing***: Overhearing means that a wireless station (STA) receives and decodes packets that are not destined to it. This situation does happen in a multi-user and shared medium scenario, especially that the default state is the receive state for wireless LANs despite whether a STA is actually receiving a packet [20].

3. ***control packet overhead***: Control (packet) overhead such as ACK frame and inter-frame space (IFS) are necessary to maintain MAC operations normally. Unfortunately, they increase the active time and energy consumption of STAs when transmitting, receiving control frames or experiencing some backoff deferral to facilitate distributed coordination.

4. ***idle listening***: Idle listening represents useless energy consumption of an STA waiting for possible frames destined to it. This is especially true in ad hoc or sensor network applications.

It is clear that the battery lifetime can be extended if these factors of energy waste can be eliminated.

Since the number of devices with wireless LAN functionality is expected to keep growing, a large network consists of multiple users should be investigated. To increase energy conservation, the focus should be reducing the number of STAs in Awake state at a given time and separate their communicating durations. There are several previous works [26] - [28] studied the problem of adaptively tracking packet inter-arrival time to avoid unnecessary idle listening while satisfying the QoS requirement. However, in a large network, even the periodic

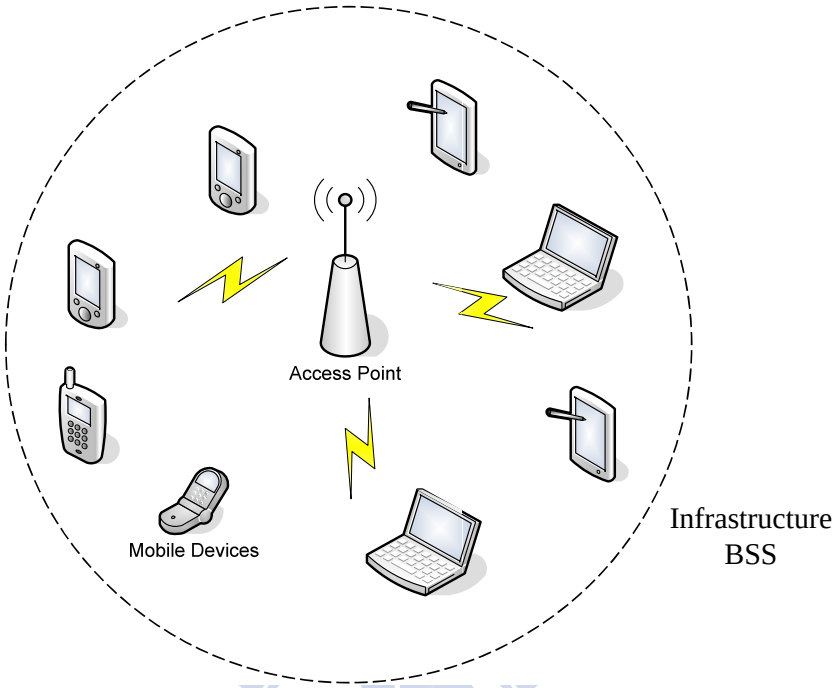

Figure 1.1: The infrastructure Wireless LANs.

service schedule has been determined or the traffic are available for transmission in the access point (AP) or STA, the efficient coordinations for channel access and awake duration for the STAs are still challenging due to the uncertainties of actual required time. Hence, in this dissertation, we specifically focus on the centralized power management techniques to avoid above mentioned sources of energy wastage and dedicate to the elimination of ***multi-user overhearing problem***. The applying scenario is infrastructure wireless LANs, as shown in Fig. 1.1, where the AP can take the responsibility of scheduling the activities and power states of STAs and it is assumed that STAs can hear each other within the BSS.

1.2 Acronyms

The acronyms used in this dissertation are listed in the following. [1]

ACK acknowledgement

AP access point

[1]In this dissertation, the terms AP and STA refer to what in the 802.11e standard is denoted as QAP and non-AP QSTA respectively.

APSD automatic power save delivery

BSS basic service set

DCF Distributed Coordination Function

EDCA Enhanced Distributed Channel Access

HCCA HCF Controlled Channel Access

HCF Hybrid Coordination Function

IFS inter-frame space

MAC medium access control

MIMO multiple-input multiple-output

OFDM orthogonal frequency division multiplexing

PCF Point Coordination Function

QoS quality of service

SI service interval

SE scheduled event

SP service period

STA wireless station

TDMA time division multiple access

TS traffic stream

TXOP transmission oppourtunity

1.3 Contributions and Dissertation Organization

The main parts of this dissertation are placed in Chapter 3 and Chapter 5. For clarity, the contributions of Chapter 3 and Chapter 5 are firstly briefed as follows.

As other specifications, the specific scheduling algorithm for the Scheduled Automatic Power Save Delivery (S-APSD) mechanism of IEEE 802.11e is left open to allow vendor dif-

ferentiations. In Chapter 3, to overcome the high implementation complexity of the previous work [38], we develop two low-complexity scheduling algorithms for the S-APSD mechanism.

1. A low complexity non-contiguous scheduling algorithm which exploits the periodicity of the service schedules is presented in Section 3.2.

2. The idea is further enhanced in Section 3.3 by precalculating and storing required information for the limited number of Classes of traffic streams to allow fast scheduling and a suggested implementation method is presented.

This work was submitted and under reviewed in *IEEE Transactions on Mobile Computing* [46] and was presented in parts at the 2010 *IEEE* 71^{st} *Vehicular Technology Conference: VTC2010-Spring* [44].

In Chapter 5, to achieve the goal of maximizing energy saving while maintaining the merit of high bandwidth utilization multi-polling scheme, a multi-polling mechanism considering energy efficiency is proposed and studied.

1. An Energy-efficient Multi-Polling mechanism is presented with detailed operations of AP and wireless stations in Section 5.2.

2. To optimize the performance of energy saving subject to a pre-defined tolerable loss of bandwidth utilization, a derivation of wake-up time schedule is presented in Section 5.3.

This work was appeared in *IEEE Transactions on Wireless Communication* in 2009 [45] and was presented in parts at the 2007 *IEEE* 65^{th} *Vehicular Technology Conference: VTC2007-Spring* [32].

Besides Chapter 3 and 5, the remainder of this dissertation is organized as follows. For ease of readability, we describes the backgrounds and literature review of channel access schemes and energy-saving in Chapter 2. Including basic channel access schemes provided in the standards, the low overhead multi-polling schemes [15]– [17] and [18] are also reviewed in

this chapter. In Chapter 4, we investigate the problem of rearranging the existing scheduled events to attain better energy saving performance. We prove the upper bound of the system minimum distance and design efficient rearranging algorithms to achieve satisfactory energy saving. This work was accepted and to be presented in parts at the *IEEE TENCON2010* [48]. To increase the performance on energy saving considering the error recovery issue for the TDMA-like multi-polling mechanism [18], Chapter 6 develops wake-up time schedules and an update algorithm in correspondence to the delay caused by retransmissions. This work was presented at the 2008 *IEEE* 67^{th} *Vehicular Technology Conference: VTC2008-Spring* [44]. In the end, we conclude our results, and propose some future work in Chapter 7.

Chapter 2

Backgrounds

For a better understanding, some background knowledge about medium access control and power management of wireless LANs is provided in this chapter. In short, the medium access control protocols of IEEE 802.11 [1] and 802.11e [2] and some variations are briefly introduced in Section 2.1. What follows is Section 2.2 that gives a description of the power management schemes covered in the standards and a related work [38] for the scheduling algorithm used in S-APSD.

2.1 Medium Access Control of Wireless LANs

The basic access scheme of wireless LANs is named as Distributed Coordination Function (DCF) which is known to be a contention-based mechanism. Considering a multiple STAs scenario, every STA uses the carrier-sense multiple access with collision avoidance (CSMA/CA) before its transmissions. A sensing duration of DCF IFS (DIFS) is preceding the access attempt of non-AP STAs. If the medium is sensed to be busy, a STA holds its access attempt and randomly generates a backoff time in units of time slots from a region called contention window by itself. Once the medium becomes idle for DIFS, a STA starts to decrease its backoff counter one by one for every additional idle time slot. Then a STA freezes the counter and sets its network allocation vector (NAV) with the value carried in the duration field of the overheard frame for virtual carrier sensing. When the backoff counter

reaches zero, a STA has the right to initiate its transmission for a duration of a single data frame.

On the other hand, there is also a centralized channel coordination scheme named as Point Coordination Function (PCF) defined in 802.11. In PCF, the AP maintains a polling list and a polled STA has the right to transmit one data frame during the contention-free period. Such policy may induce bandwidth hogging by low rate STAs, i.e., a single STA may possess the medium for a long time. Therefore, in addition to the priority differentiations defined in Enhanced Distributed Channel Access (EDCA) and HCF Controlled Channel Access (HCCA), the succeeding 802.11e introduces the notion of TXOP to limit the transmission time of a STA who may need to fragment a large frame into smaller ones to fit into the interval. The TXOP is a bounded interval during which the holder can initiate an uninterrupted frame exchange sequence. Moreover, if a TXOP holder does not have enough data to use up the allocated time, it can return the remaining time to the AP to do resource reallocation. (The join/leaving handshaking for the polling list of a STA can utilize the ADDTS (Add Traffic Stream)/DELTS (Delete Traffic Stream) frame defined in [2].) Nevertheless, the requirement of polling frames is kind of overhead lowering system BU. To reduce protocol overhead, multi-polling schemes are often employed.

Ordered-Contention Multi-polling Mechanisms

The concept of ordered contention was first proposed in CP-Multipoll (Contention-Period Multi-Polling) [15] and slightly modified in [16] and [17]. The basic idea is that the AP announces the channel access order of STAs in the polling list via backoff value assignments. In other words, the CSMA/CA access scheme is incorporated into the polling scheme except that the backoff value is controlled by the AP. After receiving the notification, all STAs set their backoff counters and start to count down after the medium is sensed to be idle for a period called Short IFS (SIFS), which is shorter than DCF IFS (DIFS) used by legacy STAs to prevent interruption. ($DIFS = SIFS + 2 \times SlotTime$ [1]) If the medium is busy, a STA

freezes the counter and sets its network allocation vector (NAV) with the value carried in the duration field of the overheard frame for virtual carrier sensing. When the backoff counter reaches zero, a STA has the right to initiate its transmission for a duration as long as the allocated TXOP. The ordered-contention scheme adapts well to the *variable bit rate* (VBR) traffic.

Since the operations of the above multi-polling mechanisms require the scheduled STAs to monitor channel activity constantly, STAs which have not yet finished their transmission cannot enter the Doze state because they need to update their NAV values during others' transmission and decrease the backoff counters when the medium is idle. The overhearing problem exists in such scenario and would be more serious when there are many STAs to be polled. STAs of later orders tend to run out of their energy quickly since they spend more time to wait before accessing the medium. To the best of our knowledge, there is no power management scheme, except our previous work [32], which addresses the overhearing problem associated with the ordered-contention multi-polling mechanisms. In [32], the wake-up time schedule is computed based on a heuristic algorithm and achieves satisfactory energy saving with slight sacrifice of bandwidth utilization. However, the proposed scheme does not maximize the energy saving under the constraint of a pre-determined loss of bandwidth utilization. Hence, in Chapter 4, we present an enhancement to [32] and design a wake-up time schedule to fulfill the mentioned requirement.

Two-Step Multi-polling Mechanism

The TSMP (Two-Step Multi-Polling) scheme [18] provides TXOP allocation with two multi-polling frames. The idea is to poll in the first step the STAs which are likely to have pending data to transmit with the Status-Request Multipoll (SRMP) frame to initiate the Status Collection Period (SCP). The polled STAs reply Status-Response frames one by one with their buffer statuses and the selected downlink rates from the channel estimation based on the received SRMP frame. After collecting the responses, the AP estimates the uplink rate

for STAs. In the second step, the determined rates and the buffer statuses of STAs are used to derive an exact schedule which is then announced in the Data Transmission Multipoll frame. Since the time allocation for each STA in the Data Transmission Period (DTP) can be exactly derived, it seems very helpful for energy saving. However, the status collection process could bring significant time overhead since the handshaking frames should be sent with the base PHY rate. Moreover, the TDMA-alike channel access fashion could make it more vulnerable to the error recovery issue induced from the imperfect wireless channel. Therefore, we also studied the energy-saving issue caused by the retransmissions of previous accessing STAs [43].

2.2 Power Management of Wireless LANs

The IEEE 802.11 standard document defines two power management (PM) modes - Active mode and Power Save (PS) mode. In the Active mode, STAs stay in the Awake state and are fully powered to receive frames. On the other hand, STAs in the PS mode shall be in the Doze state and awake to listen to selected beacons. a STA in the Doze state consumes much less power than in the Awake state by turning off the RF circuitry. The switchover in between the states takes about 250μs [30]. The Lucent IEEE 802.11 WaveLAN card [22], for example, consumes 1.65 W, 1.4 W and 1.15 W in the transmit, receive and idle modes, respectively, in the Awake state. In the Doze state, it consumes 0.045 W which is much less than any mode of the Awake state.

A STA in the PS mode uses the power-save polling (PS-Poll) scheme to retrieve data buffered in AP to improve energy efficiency. However, the performance of downlink packet latency and bandwidth utilization may not be satisfactory when this scheme is implemented because of the dependency on beacon interval, access contention, and additional signaling load by PS-Poll frames. The 802.11e also includes some optional extension of power save functionality defined as Automatic Power Save Delivery (APSD). The unscheduled APSD

(U-APSD) is a distributed mechanism for STAs to decide when to awake to receive buffered frames at the AP and use triggering data frames to retrieve corresponding buffered data. Scheduled APSD (S-APSD) is a centralized mechanism for the AP to determine some fixed intervals that STAs should periodically awake to receive the frames. These frameworks can reduce the signaling load (PS-Poll frames) and are, as stated, mainly designed for downlink usage.

There are some other centralized PM schemes [19], [20] which achieve energy saving by adopting the *shortest-job-first* policy to schedule transmissions. The shortest-job-first policy arranges the access orders of STAs according to their aggregate required time and STAs which finish their transmissions can switch to the Doze state. It can minimize the average waiting time after the announcement of schedule for STAs; however, the STAs of later orders tend to overhear more.

Scheduled Automatic Power-Save Delivery

To support QoS while dealing with power saving issue, U-APSD and S-APSD architectures are defined in IEEE 802.11e [2], [38]. The AP should use EDCA access method when selecting the U-APSD scheme. On the other hand, the S-APSD lays the burden of channel coordination on the AP to calculate the schedule and announce it to STAs. When S-APSD is used, depending on whether the usage is for an access category or for a TS, EDCA or HCCA is chosen as access policy, respectively. For the S-APSD, the STA firstly communicates with the AP via ADDTS request frame setting both APSD and Schedule subfields in the TS info field before getting admitted. If the requested service can be satisfied, the QAP will notify the STA of the schedule including the Service Start Time (SST) and the negotiated Service Interval (SI). In APSD, the contiguous time that a STA stays awake to receive the buffered frames from QAP is defined as Service Period (SP). STAs using S-APSD should automatically switch to the Awake state at the scheduled starting time of each SP defined

by

$$SST + m \times SI, \text{where } m \geq 0. \tag{2.1}$$

Then they fall back to sleep till receiving the frames with the EOSP (End Of Service Period) flag being set. The service schedule can be updated after negotiation between the QAP and STA finishes. To maintain QoS guarantee, the new SST should fall into the region between the minimum SI and maximum SI after the beginning of the previous SP. Compared with the 802.11 PSM, besides QoS support, the S-APSD can also reduce signaling loads such as PS-Poll. Moreover, the number of collisions can be decreased as well.

Overlapping Aware S-APSD

Although IEEE 802.11e defines the architecture of S-APSD, its specific implementation is left as an open issue. To reduce the chance of SP overlapping, as described in [38], there could be two scheduling approaches to schedule the starting time of SPs. One is contiguous scheduling, which means that the scheduled SPs should be placed one after another. It has the advantage of simplifying the process to determine the SST for the schedule of a new traffic streams (TS). However, it often requires the SIs to be altered to satisfy certain constraint so that contiguous scheduling is possible. A necessary and sufficient condition for a group of periodic tasks defined by service intervals (SIs) and Transmission Opportunities (TXOPs) to be scheduled contiguously by an Equal-spacing-based Rate Monotonic algorithm was derived in [36]. Altering SIs may shorten the sleeping time of STAs and, as a result, cause more energy consumption to retrieve the same amount of data buffered at QAP. Besides, the unused time of scheduled SP might be wasted because it could be too short for other STAs using EDCA to transmit their frames. On the other hand, the non-contiguous scheduling does not put any restrictions on the setting of SIs. Compared with contiguous scheduling, non-contiguous scheduling allows other EDCA STAs to have higher chance to transmit their frames for the unused time of scheduled SP [38]. However, the larger degree of freedom for scheduling also increases its complexity for finding the suitable SST of a new TS.

In [38], the authors proposed a non-contiguous scheduling algorithm called Overlapping Aware S-APSD (OAS-APSD). The OAS-APSD algorithm aims at finding the SST of a new TS which achieves the least probability of SP overlapping. The pseudo-code of the OAS-APSD algorithm is shown below. To be concise, we use scheduled instants to represent the scheduled starting time of SPs. The Scheduled Events (SEs) in the OAS-APSD algorithm refer to the scheduled instants known to the QAP, for example, Beacons with period BI and already scheduled SPs with period SIs for TSs. In this algorithm, SI_{new} represents the SI of the new TS to be scheduled.

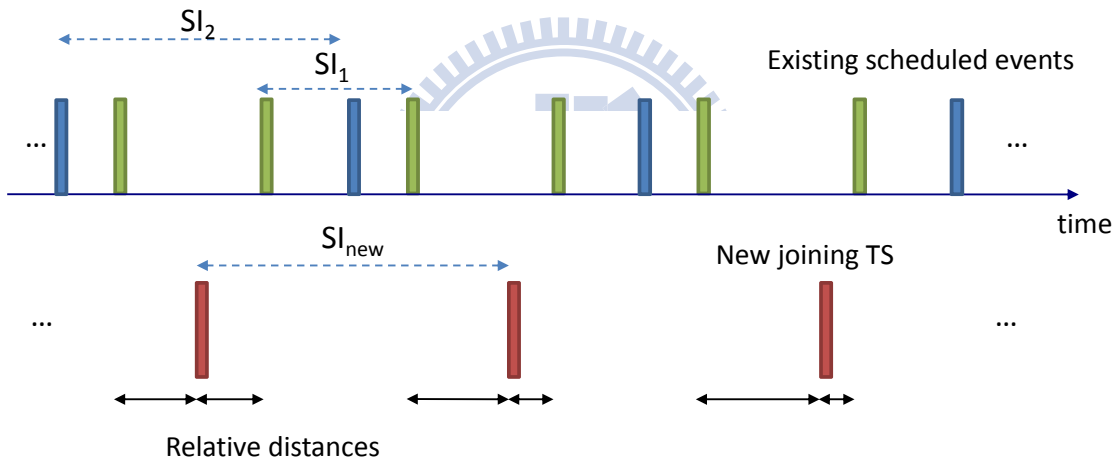

Figure 2.1: Illustration of relative distance between scheduled events.

The OAS-APSD algorithm [38].

SST to be determined given a specific SI_{new}

N, SST, SST_{temp}, $dist_{avg}$, $temp\_dist_{avg}$, $max\_distmin \leftarrow 0$

$temp\_dist_{min} \leftarrow BI$

Create empty list of $SEs \rightarrow ListSE$

Compute LCM **considering** $All\ SIs$ **plus** $BI \rightarrow LCM$

for $\forall\ SEs \in [t_{current}, t_{current} + LCM]$ **do**

Insertion in $ListSE$ **of** SEs

end for

while $SST_{temp} < SI_{new}$ **do**

while $SST_{temp} + SI_{new} \times \ N < LCM$ **do**

Find $prev\_SE$ and $next\_SE$ **in** $ListSE$

$dist_{next\_SE} \leftarrow next\_SE - (SST_{temp} + SI_{new} \times N)$

$dist_{prev\_SE} \leftarrow SST_{temp} + SI_{new} \times \ N - prev\_SE$

Insertion in $distances\_SST_{temp}$ **of** $distnext\_SE$ **and** $dist_{prev\_SE}$

$N \rightarrow N + 1$

end while

$temp\_dist_{min} \leftarrow$ **Minimum of** $distances\_SST_{temp}$

$temp\_dist_{avg} \leftarrow$ **Average of** $distances\_SST_{temp}$

if $temp\_dist_{min} > max\_dist_{min}$ **then**

$max\_dist_{min} \leftarrow temp\_dist_{min}$

$dist_{avg} \leftarrow temp\_dist_{avg}$

$SST \leftarrow SST_{temp}$

else if $temp\_dist_{min} = max\_dist_{min}$ **then**

if $temp\_dist_{avg} > dist_{avg}$ **then**

$dist_{avg} \leftarrow temp\_dist_{avg}$

$SST \leftarrow SST_{temp}$

else if $temp\_dist_{avg} = dist_{avg}$ **then**

$SST \leftarrow random(SST, SST_{temp})$

end if

end if

$SST_{temp} \leftarrow SST_{temp} + precision$

end while

The basic idea of the OAS-APSD algorithm is to find the optimal SST of the new TS in an interval $[0, SI_{new}]$ which achieves the maximum among minimum relative distances between SPs of the new TS and existing scheduled events. Here, the relative distances between SPs of the new TS and existing scheduled events are defined as the distances from every SP of the new TS to its closest previous and next existing scheduled events, as illustrated in Fig. 2.1. Note that there are only a finite number of possibilities for the SST because there is a maximum precision used by the 802.11e specification. To determine the optimal SST, relative distances are calculated for an interval of duration LCM, the least common multiple of the SIs, including SI_{new}. If there is a tie, then the one with maximum average relative distance is selected. In case there is still a tie based on average relative distance, it is broken arbitrarily. Clearly, the computational complexity of the OAS-APSD algorithm is large for large values of LCM. This could make the algorithm infeasible for real systems and the complexity issue is listed as one of the major future works in [38]. Therefore, it motivates our work to design low complexity scheduling algorithms for S-APSD [44] [46] which will be presented in the following chapters.

Chapter 3

Scheduling Algorithm for Scheduled Automatic Power Save Delivery

The purpose of this chapter is to present a low complexity scheduling algorithm for S-APSD which achieves QoS support and power saving simultaneously. In S-APSD, the most important goal for the scheduler is to reduce the chance of service period (SP) overlapping which would results in energy-consuming overhearing. Here we adopt the non-contiguous scheduling approach as in [38][1], while the scheduling criterion is slightly different from its. To reduce online computational complexity, we prove the periodicity of distances between two periodic sequences, classify TS based on their SIs, and store necessary information to allow fast scheduling.

The rest of this chapter is organized as follows. In Section 3.1, we explain the system model used for the design of our scheduling algorithm. Section 3.2 describes the idea of our proposed scheduling algorithm. The idea is further extended to a system with multiple classes of TSs in Section 3.3. In Section 3.4, we compare our proposed algorithms with the existing work [38] in terms of complexity and affordable transmissions in the available time, i.e. the number of frames that can be transmitted by EDCA STAs during the time excluding scheduled transmissions. Finally, we draw the summary in Section 3.5.

[1]Please refer to Section 2.2 for the definition of non-contiguous scheduling and the explanations of the OAS-APSD algorithm in [38].

3.1 System Model

The system for the scheduling is actually time-slotted with slot size equal to the duration of precision used in IEEE 802.11e.[2] Without loss of generality, we shall normalize the duration of precision to one. As a result, the scheduled instants for a specific TS can be represented as a sequence, say,

$$\boldsymbol{X} = \{x_m\}_{m=-\infty}^{\infty}, \text{ such that } x_m = x_{m-1} + p, \tag{3.1}$$

where p, called period, is the SI of the TS. The origin can be chosen arbitrarily. Consequently, the SST of the TS corresponds to some x_i and the STA to which the TS is attached wakes up periodically at time instants x_j for all $j \geq i$.

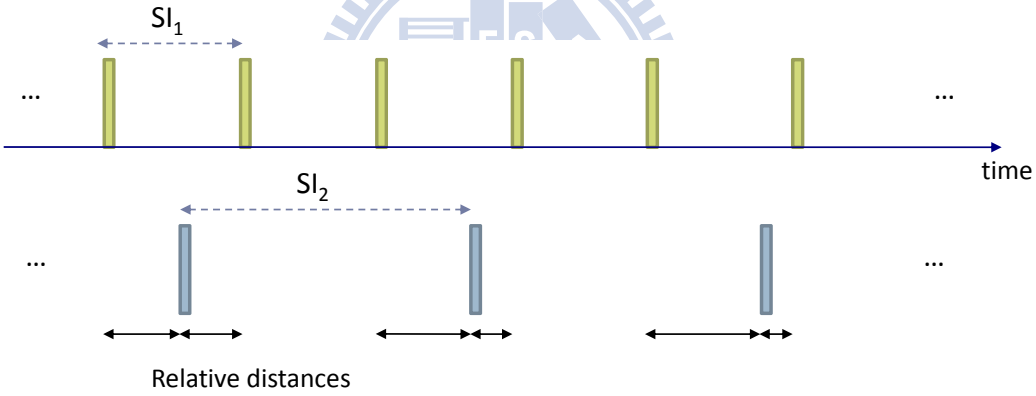

Figure 3.1: Illustration of relative distance between two periodic sequences.

Let $\boldsymbol{Y} = \{y_m\}_{m=-\infty}^{\infty}$ be another periodic sequence of period such that $y_m = q \times m$ for all m. Further, let

$$\boldsymbol{Y} + k = \{y_m + k\}_{m=-\infty}^{\infty}. \tag{3.2}$$

be a shifted version of $\boldsymbol{Y}$. We call k the offset of $\boldsymbol{Y} + k$ with respect to $\boldsymbol{Y}$. It is clear that $\boldsymbol{Y} + k$ is periodic with period q when the offset is k. Define the distance between sequences $\boldsymbol{X}$ and $\boldsymbol{Y} + k$ as

$$d(\boldsymbol{X}, \boldsymbol{Y} + k) = \min_{-\infty \leq l, m \leq \infty} |y_l + k - x_m|. \tag{3.3}$$

[2]Note that the meaning of slot is different from the *SlotTime* mentioned in Chapter 2.

It is not hard to see that

$$d(\boldsymbol{X}, \boldsymbol{Y}+k) = d(\boldsymbol{Y}+k, \boldsymbol{X}) = d(\boldsymbol{X}-k, \boldsymbol{Y}) \tag{3.4}$$

, and

$$d(\boldsymbol{X}, \boldsymbol{Y}+0) = d(\boldsymbol{X}, \boldsymbol{Y}) = 0. \tag{3.5}$$

Define the largest possible distance between sequences $\boldsymbol{X}$ and $\boldsymbol{Y}$ as

$$D(\boldsymbol{X}, \boldsymbol{Y}) = \max_k \{d(\boldsymbol{X}, \boldsymbol{Y}+k)\}. \tag{3.6}$$

3.2 Low Complexity Scheduling Algorithm

3.2.1 Basic Idea

In this sub-section, we present the basic idea of determining the optimal SST for a new TS, i.e., the SST that causes the largest minimum distance between the new TS and existing TSs. Consider the simplest case of scheduling the SPs for the first TS with period p. Let $\boldsymbol{X}$ be the scheduled instants for the first TS. It is clear that any sequence of period p is optimal. For simplicity, we choose $x_m = p \times m$ for all m.

Consider now the case where an existing TS with period had been scheduled and a new TS is to be scheduled. Assume that the period of the new TS to be scheduled is q and the $\boldsymbol{Y}+k$ is a periodic sequence with period q and offset k.

Therefore, **our goal is to find the offset** k^*, the optimal value of k which satisfies

$$k^* = \arg\max_k d(\boldsymbol{X}, \boldsymbol{Y}+k). \tag{3.7}$$

Once k^* is obtained, the scheduled instants of the new TS is $\boldsymbol{Y}+k^*$ and the SST can be determined based on current time. Let $G = gcd(p, q)$ and $L = lcm(p, q)$ be, respectively, the greatest common divisor and the least common multiple of p and q. We prove in Theorem 3.1 a property of $d(\boldsymbol{X}, \boldsymbol{Y}+k)$.

Theorem 3.1. *$d(\boldsymbol{X}, \boldsymbol{Y}+k)$ is periodic in k with period G.*

Proof. It is clear that $d(\boldsymbol{X}, \boldsymbol{Y}+k)$ is periodic in k because $d(\boldsymbol{X}, \boldsymbol{Y}+q+k) = d(\boldsymbol{X}, \boldsymbol{Y}+k)$. Let n be its period. We shall prove that $n|G$ (i.e., n divides G) and $G|n$. According to the Euclidean algorithm [40], there exist integers a and b such that

$$G = a \times \ p + b \times \ q \text{ or } G + (-b) \times \ q = a \times p \tag{3.8}$$

which implies one of the scheduled instants of $\boldsymbol{Y}+G$ coincides with some scheduled instant of X. Consequently, we have $d(\boldsymbol{X}, \boldsymbol{Y}+G+k) = d(\boldsymbol{X}, \boldsymbol{Y}+k)$, which implies $n|G$.

Conversely, since n is the period of $d(\boldsymbol{X}, \boldsymbol{Y}+k)$, we have $d(\boldsymbol{X}, \boldsymbol{Y}+0) = d(\boldsymbol{X}, \boldsymbol{Y}+n)$, which implies there must exist integers s and t such that

$$n + s \times q = t \times p \text{ or } n = (-s) \times q + t \times p. \tag{3.9}$$

As a result, it holds that $G|n$ because $G|p$ and $G|q$. This completes the proof of Theorem 3.1. □

A consequence of Theorem 3.1 is that k^* can be chosen to satisfy $0 \leq k^* \leq G-1$. Note that to compute $d(\boldsymbol{X}, \boldsymbol{Y}+k)$, $0 \leq k^* \leq G-1$, we need only consider finite partial sequences of $\boldsymbol{X}$ and $\boldsymbol{Y}+k$ because the same situation repeats every L slots.

Two cases are analyzed separately below.

Case 1. $q \geq p$

For $q \geq p$, we need only consider $\{x_m\}_{m=0}^{L/p-1}$ and $\{y_m + k\}_{m=0}^{L/q-1}$. Let

$$f_m(k) = \Big\lfloor \frac{q \times m + k}{p} \Big\rfloor, \tag{3.10}$$

where $\lfloor x \rfloor$ represents the largest integer smaller than or equal to x . Define

$$a_m(k) = min\Big\{ q \times m + k - p \times f_m(k), p \times [f_m(k)+1] - (q \times m + k) \Big\} \tag{3.11}$$

as the shorter distance of $y_m + k$ to the two closest neighboring x_m's. We have

$$d(\boldsymbol{X}, \boldsymbol{Y}+k) = \min_{0 \leq m \leq L/q-1} \{a_m(k)\}. \tag{3.12}$$

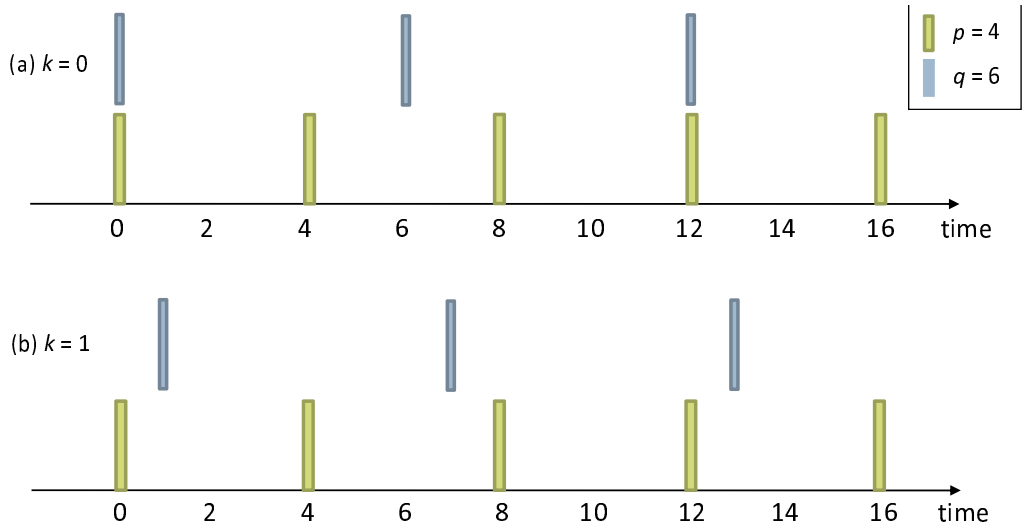

Figure 3.2: S-APSD scheduling example.

Figures 3.2(a)-3.2(b) show an example for $p = 4$ and $q = 6$. For this example, we have $G = 2$ and, therefore, we need only compute $d(\boldsymbol{X}, \boldsymbol{Y} + 0)$ and $d(\boldsymbol{X}, \boldsymbol{Y} + 1)$. One can easily verify that $a_0(0) = min\{0, 4\} = 0$, $a_1(0) = min\{2, 2\} = 2$, $d(\boldsymbol{X}, \boldsymbol{Y} + 0) = min\{0, 2\} = 0$; and $a_0(1) = min\{1, 3\} = 1$, $a_1(1) = min\{3, 1\} = 1$, $d(\boldsymbol{X}, \boldsymbol{Y} + 1) = min\{1, 1\} = 1$. As a consequence, we have $D(\boldsymbol{X}, \boldsymbol{Y}) = max\{d(\boldsymbol{X}, \boldsymbol{Y} + 0), d(\boldsymbol{X}, \boldsymbol{Y} + 1)\} = 1$ and $k^* = 1$.

Case 2. $q < p$

For $q < p$, one can compute $a_m(k)$ for $0 \leq m \leq L/q - 1$ and then determine $d(\boldsymbol{X}, \boldsymbol{Y} + k) = \min_{0 \leq m \leq L/q-1} \{a_m(k)\}$ Alternatively, one can change the roles of p and q and apply the procedure performed for *Case* 1. Note that $d(\boldsymbol{X}, \boldsymbol{Y} + k) = d(\boldsymbol{X} + k, \boldsymbol{Y}) = d(\boldsymbol{X} + G + k, \boldsymbol{Y})$ implies the desired results can be obtained by interchanging the roles of p and q. By doing so, the complexity of computing $d(\boldsymbol{X}, \boldsymbol{Y} + k)$ is reduced because fewer $a_m(k)$'s (L/p *vs.* L/q) are calculated.

To summarize, in order to determine $d(\boldsymbol{X}, \boldsymbol{Y} + k)$, we need to compute $L/max(p, q)$ $a_m(k)$'s and then pick the smallest one. Since there are G different values for variable k, the complexity of determining $d(\boldsymbol{X}, \boldsymbol{Y} + k)$, $0 \leq k \leq G - 1$, is $O(G \cdot L/max(p, q)) = O(min(p, q))$ multiplications and divisions. The following algorithm eliminates all multiplications and di-

visions. The algorithm requires roughly $4 \times \min(p, q)$ comparisons.

Algorithm *for computing* $D(X, Y)$ *and* k^* *assuming that* $q \geq p$.

$R = q \bmod p$

$D,\ k^* \leftarrow 0$ /* D stores the value of $D(\boldsymbol{X}, \boldsymbol{Y})$.*/

$k > 1$

while $k < G$

$m \leftarrow 0$

$S \leftarrow k$ /* S stores the value of $q \times m + k - p \times f_m(k)$. */

$relative\_dist \leftarrow \min(S, p - S)$

$min\_relative\_dist \leftarrow relative\_dist$

while $m \leq L/q - 1$

$S \leftarrow S + R$

if $S > p$

$S \leftarrow S - p$

end if

$relative\_dist \leftarrow \min\{S, p - S\}$

$min\_relative\_dist \leftarrow \min\{relative\_dist, min\_relative\_dist\}$

$m = m + 1$

end while

if $D < min\_relative\_dist$

$D \leftarrow min\_relative\_dist$

$k^* \leftarrow k$

else if $D = min\_relative\_dist$

$k^* = random(k^*, k)$

end if

$k \leftarrow k + 1$

end while

3.2.2 Generalization to K Existing Traffic Streams

Let us now extend the results to $K \geq 2$ existing TSs when a new TS is to be scheduled. The notations and their definitions used in this section can also be referred in Table 3.1. Assume that the period of the i^{th} existing TS is p_i and the period of the TS to be scheduled is q . Let $\boldsymbol{X}_i = \{x_{i,m}\}_{m=-\infty}^{\infty}$, $1 \leq i \leq K$, be a sequence of period p_i such that $x_{i,m} = p_i \times m$ for all m. Further, let

$$\boldsymbol{X}_i^{'} = \boldsymbol{X}_i + O_i \tag{3.13}$$

be the scheduled instants of the i^{th} existing TS. We shall use $\boldsymbol{X}_1$ as reference and, therefore, assign $O_1 = 0$. Consequently, we have $\boldsymbol{X}_1^{'} = \boldsymbol{X}_1$ and

$$O_i = x_{i,0} - x_{1,0} \tag{3.14}$$

represents the offset of $\boldsymbol{X}_i^{'}$ with respect to $\boldsymbol{X}_1^{'}$. According to the results obtained in subsection 3.2.1, it holds that $0 \leq O_2 \leq p_2 - 1$. We shall prove that O_i satisfies $0 \leq O_i \leq p_i - 1$ for all i . Let $G_i = gcd(p_i, q)$, $1 \leq i \leq K$, and

$$GL = lcm\{G_1, G_2, ..., G_K\} \tag{3.15}$$

the least common multiple of G_1, G_2, ..., and G_K. Also, let $L_i = lcm\{p, q\}$, $1 \leq i \leq K$.

Again, let $\boldsymbol{Y} = \{y_m\}_{m=-\infty}^{\infty}$ be a periodic sequence of period q with $y_m = q \times m$ for all m and $\boldsymbol{Y} + k = \{y_m + k\}_{m=-\infty}^{\infty}$ be a shifted version of $\boldsymbol{Y}$. Let

$$\boldsymbol{X} = \bigcup_{1 \leq i \leq K} \boldsymbol{X}_i^{'} \tag{3.16}$$

such that x is an element of $\boldsymbol{X}$ if and only if it is an element of $\boldsymbol{X}_i^{'}$ for some i, $1 \leq i \leq K$. Clearly, $\boldsymbol{X}$ is a periodic sequence with period $lcm\{p_1, p_2, ..., p_K\}$. Define

$$d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + k) = \min_{-\infty \leq l,m \leq \infty} |y_l + k - x_{i,m} - O_i| \tag{3.17}$$

and

$$d(\boldsymbol{X}, \boldsymbol{Y} + k) = \min_{1 \leq i \leq K} \{d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + k)\}. \tag{3.18}$$

The optimal value of k is again given by (3.7). It can be easily shown that $d(\boldsymbol{X}, \boldsymbol{Y} + k)$ is periodic with period GL because $d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + k)$ is periodic with period G_i, $1 \leq i \leq K$. As a result, k^* can be chosen to satisfy $0 \leq k^* \leq GL - 1$. The fact that $G_i | q$, $1 \leq i \leq K$, implies $GL | q$. In other words, GL is a factor of q and, therefore, is upper bounded by q. If the new TS is considered as the $(K+1)^{th}$ TS when the $(K+2)^{th}$ TS is to be scheduled, then we have

$$O_{K+1} = k^* \leq q - 1 = p_{K+1} - 1. \tag{3.19}$$

This proves the property that O_i satisfies $0 \leq O_i \leq p_i - 1$.

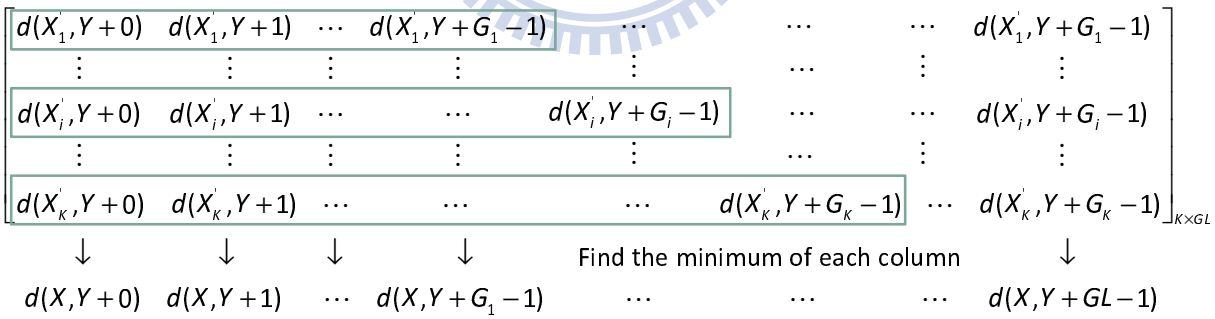

Figure 3.3: The scheduling matrix.

To compute $d(\boldsymbol{X}, \boldsymbol{Y} + k)$, we need $d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + k)$ for all i, $1 \leq i \leq K$. To determine k^* , a scheduling matrix of size $K \times GL$ is constructed, as shown in Fig. 3.3. The i^{th} row of the scheduling matrix is [$d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + 0)$ $d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + 1)$ ... $d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + G_i - 1)$] repeated for GL/G_i times. Given the scheduling matrix, $d(\boldsymbol{X}, \boldsymbol{Y} + k)$ can be obtained as the minimum element of the k^{th} column. Finally, the optimal value of k is given by the index of the column with the maximum $d(\boldsymbol{X}, \boldsymbol{Y} + k)$.

As derived previously, the complexity of computing $d(\boldsymbol{X}_i^{'}, \boldsymbol{Y}+k)$, $0 \leq k \leq G_i - 1$, requires $4 \times \min\{p_i, q\}$ comparisons. The overall complexity to generate the scheduling matrix for K existing TSs is, therefore, $\sum_{i=1}^{K} 4 \times \min\{p_i, q\}$, which is upper bounded by $4 \times q \times K$ comparisons. To find k^* , we need $K \times GL$ comparisons to obtain $d(\boldsymbol{X}, \boldsymbol{Y}+k)$, $0 \leq k \leq GL-1$, and GL comparisons to determine k^* .

Note that our algorithm is unable to break a tie based on average distance. In case there are multiple choices for k^* , we select the one with the maximum column sum. As a result, it requires $2(K-1)$ additions and one comparison for each tie-breaking. If there is still a tie, it is broken arbitrarily.

Example 3.1. Consider an example for $K = 2$, $p_1 = 12$, $p_2 = 15$, and $q = 18$ Assume that the TS with period p_1 was scheduled earlier than the TS with period p_2 . Since $\boldsymbol{X}_1$ is used as reference, we assign $\boldsymbol{X}_1^{'} = \boldsymbol{X}_1 = \{12 \cdot m\}_{m=-\infty}^{\infty}$. O_2, the offset of $\boldsymbol{X}_2^{'}$ with respect to $\boldsymbol{X}_1^{'}$ has to be determined based on the scheduling algorithm. Since $gcd(12, 15) = 3$, there are only three possible values for O_2. After some calculations, we get $O_2 = 1$ or 2. Assume that we choose $O_2 = 2$. To schedule the third TS with period $q = 18$, we need to compute $d(\boldsymbol{X}_1^{'}, \boldsymbol{Y}+k)$, $0 \leq k \leq 5$, and $d(\boldsymbol{X}_2^{'}, \boldsymbol{Y}+k)$, $0 \leq k \leq 2$, because $G_1 = gcd(12, 18) = 6$ and $G_2 = gcd(15, 18) = 3$. The results are [$d(\boldsymbol{X}_1^{'}, \boldsymbol{Y}+0)$ $d(\boldsymbol{X}_1^{'}, \boldsymbol{Y}+1)$ ... $d(\boldsymbol{X}_1^{'}, \boldsymbol{Y}+5)$] = [0 1 2 3 2 1] and [$d(\boldsymbol{X}_2^{'}, \boldsymbol{Y}+0)$ $d(\boldsymbol{X}_2^{'}, \boldsymbol{Y}+1)$ $d(\boldsymbol{X}_2^{'}, \boldsymbol{Y}+2)$] = [1 1 0]. Since $GL = lcm\{6, 3\} = 6$, the scheduling matrix is of size 2×6 and is given by

$$\begin{bmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

. Note that the second row is [1 1 0] repeated for two times. Given the scheduling matrix, we have [$d(\boldsymbol{X}, \boldsymbol{Y}+0)$ $d(\boldsymbol{X}, \boldsymbol{Y}+1)$ ... $d(\boldsymbol{X}, \boldsymbol{Y}+5)$] = [0 1 0 1 1 0]. As a result, the value of k^* can be selected as 1, 3, or 4. The column sums are 2, 4, and 3 for columns 1, 3, and 4, respectively. Therefore, k^* is selected as 3.

3.3 Class-Based Scheduling Algorithm

In real applications, it is likely that there are only a few possible periods to schedule TSs. Therefore, one can partition TSs into classes such that two TSs are in the same class if and only if they have identical periods of schedule. Assume that there are C classes, called Class 1, Class 2, ..., and Class C . Let p_i represent the period of Class i and n_i the number of TSs in Class i. If $n_i = 0$, then Class i is considered not exist. For ease of description, we assume that $n_i > 0$ for all i, $1 \leq i \leq C$. The notations and their definitions used in this section can also be referred in Table 3.2.

Consider Class i and let e_1, e_2, ..., and e_{n_i} be the TSs in the class. A TS, say, e_1, is selected as the representative of Class i. Let $\boldsymbol{X}_i = \{x_{i,m}\}_{m=-\infty}^{\infty}$ be a sequence of period p_i such that $x_{i,m} = p_i \times m$ for all m. We shall use the representative TS as reference within the class and, therefore, assign $\boldsymbol{X}_i$ as the scheduled instants of TS e_1. Let $o_{i,s}$ be the intra-class offset of TS e_s with respect to TS e_1. As a result, the scheduled instants for TS e_s is $\boldsymbol{X}_i + o_{i,s}$. Let

$$\boldsymbol{X}_{i,s} = \boldsymbol{X}_i + o_{i,s} \tag{3.20}$$

and

$$\overline{\boldsymbol{X}}_i = \bigcup_{1 \leq s \leq n_i} \boldsymbol{X}_{i,s}. \tag{3.21}$$

Assume that a new TS in Class j is to be scheduled. The impact of $\overline{\boldsymbol{X}}_i$ to the new TS can be analyzed as follows.

Let $\boldsymbol{Y} = \{y_m\}_{m=-\infty}^{\infty}$ be a periodic sequence of period p_j with $y_m = p_j \times \ m$ for all m. According to the results presented in the previous section, we need to construct a scheduling matrix of size $n_i \times G_{i,j}$, where $G_{i,j} = gcd(p_i, p_j)$. The s^{th} row of the scheduling matrix is [$d(\boldsymbol{X}_i, \boldsymbol{Y} + 0)$ $d(\boldsymbol{X}_i, \boldsymbol{Y} + 1)$ ... $d(\boldsymbol{X}_i, \boldsymbol{Y} + G_{i,j} - 1)$] circularly shifted to the right by $o_{i,s}$ positions. By taking the minimum element in each column, we obtain

$$R(\overline{\boldsymbol{X}}_i, \boldsymbol{Y}) = [\ d(\overline{\boldsymbol{X}}_i, \boldsymbol{Y} + 0)\ d(\overline{\boldsymbol{X}}_i, \boldsymbol{Y} + 1) \ ... \ d(\overline{\boldsymbol{X}}_i, \boldsymbol{Y} + G_{i,j} - 1)\ ]. \tag{3.22}$$

Note that the optimal SST of the new TS cannot be determined solely by $R(\overline{\boldsymbol{X}}_i, \boldsymbol{Y})$ because there are still TSs in other classes.

Assume that $R(\overline{\boldsymbol{X}}_i, \boldsymbol{Y})$, $1 \leq i \leq C$, are obtained. We shall use the representative TS of Class 1 as reference of the overall system. Let

$$\overline{\boldsymbol{X}} = \underset{1 \leq i \leq C}{\cup} \overline{\boldsymbol{X}}_i. \tag{3.23}$$

Further, let O_i, $1 \leq i \leq C$, be the inter-class offset of the Class i representative TS with respect to the Class 1 representative TS. To determine the optimal SST of the new TS, we construct a global scheduling matrix M of size $C \times \ GL_j$,where

$$GL_j = lcm\{G_{1,j}, G_{2,j}, ..., G_{C,j}\}. \tag{3.24}$$

The i^{th} row of M is $R(\overline{\boldsymbol{X}}_i, \boldsymbol{Y})$ repeated for $GL_j/G_{i,j}$ times and then circularly shifted to the right by O_i positions. By taking the minimum element of each column, we obtain

$$R(\overline{\boldsymbol{X}}, \boldsymbol{Y}) = [\ d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + 0)\ d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + 1)\ ...\ d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + GL_j - 1)\ ]. \tag{3.25}$$

Finally, the optimal scheduled instants of the new TS is given by $\boldsymbol{Y} + k^*$, where k^* satisfies

$$k^* = \arg \max_{0 \leq k \leq GL_j - 1} d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + k). \tag{3.26}$$

If the new TS is considered as the $(n+1)^{th}$ TS of Class j, then we update the intra-class offset

$$o_{j,n_j+1} = k^* - O_j. \tag{3.27}$$

3.3.1 Suggested Implementation Method

To reduce online scheduling complexity, we allocate C pairs of arrays for each class. Again, consider Class i . Denote the k^{th} element of the j^{th} pair of arrays by , $A_{i,j}[k]$ and , $B_{i,j}[k]$, $0 \leq k \leq G_{i,j} - 1$. The array $A_{i,j}$ stores $[\ d(\boldsymbol{X}_i, \boldsymbol{Y} + 0)\ d(\boldsymbol{X}_i, \boldsymbol{Y} + 1)\ ...\ d(\boldsymbol{X}_i, \boldsymbol{Y} + G_{i,j} - 1)\ ]$ for $\boldsymbol{Y} = \{y_m\}_{m=-\infty}^{\infty}$ with $y_m = p_j \times\ m$ for all m. Note that $A_{i,j}$ represents the impact of the

representative TS e_1 to a new TS of Class j. The array $B_{i,j}$ stores $R(\overline{\boldsymbol{X}_i}, \boldsymbol{Y})$ and represents the impact of all TSs in Class i to a new TS of Class j. When a new TS in Class j is to be scheduled, the arrays $B_{i,j}$, $1 \leq i \leq C$, are used to construct the global scheduling matrix M as illustrated in Fig. 3.4. All we need to do is taking the minimum of each column and then find the maximum among the minima. The complexity is only $C \times GL_j$ comparisons. After the new TS is scheduled, we need to update $B_{j,l}[k]$, $1 \leq l \leq C$ and, $0 \leq k \leq G_{j,l} - 1$, as

$$B_{j,l}[k] = \min\{B_{j,l}[k], A_{j,l}[k - o_{j,n_j+1}]\} \tag{3.28}$$

because the impact of TSs in Class j to a new TS of every class is changed. Here, the index $k - o_{j,n_j+1}$ is performed modulo $G_{i,j}$. The complexity of the update process is $\sum_{l=1}^{C} G_{j,l}$ comparisons, which is upper bounded by $C \times p_j$ comparisons.

When a TS in Class i finishes, we need to update $B_{i,j}[k]$, $1 \leq j \leq C$ and , $0 \leq k \leq G_{i,j}-1$, as follows. Remove the finished TS so that the updated n_i, denoted by $n_i^{'}$, becomes $n_i - 1$. Construct a scheduling matrix of size $n_i^{'} \times G_{i,j}$ such that the m^{th} row is $A_{i,j}$ circularly shifted to the right by $o_{i,m}$ positions. The content of $B_{i,j}[k]$ is updated as the minimum of the k^{th} column. Of course, a new representative TS is selected before the update process if the finished one was originally the representative TS of Class i. The complexity of the update process is $n_i^{'} \times \sum_{j=1}^{C} G_{i,j}$ comparisons.

Again, we break a tie based on column sum of the global scheduling matrix which requires $2(C-1)$ additions and one comparison. Note that one can pre-compute $A_{i,j}[k]$, $1 \leq i, j \leq C$ and , $0 \leq k \leq G_{i,j} - 1$. The initial content of $B_{i,j}[k]$ is set to a sufficiently large value for all j and k if there is no Class i TS, i.e., $n_i = 0$.

Once the STA has negotiated with the AP a power saving delivery service according to the S-APSD mechanism, it should automatically switch to the Awake state at designated scheduled time every SI unless any further notification. Therefore, the signaling load is saved effectively when compared with legacy power save mode. The start of its established service, i.e., the SST, can be easily found by the AP according to the repetition pattern of

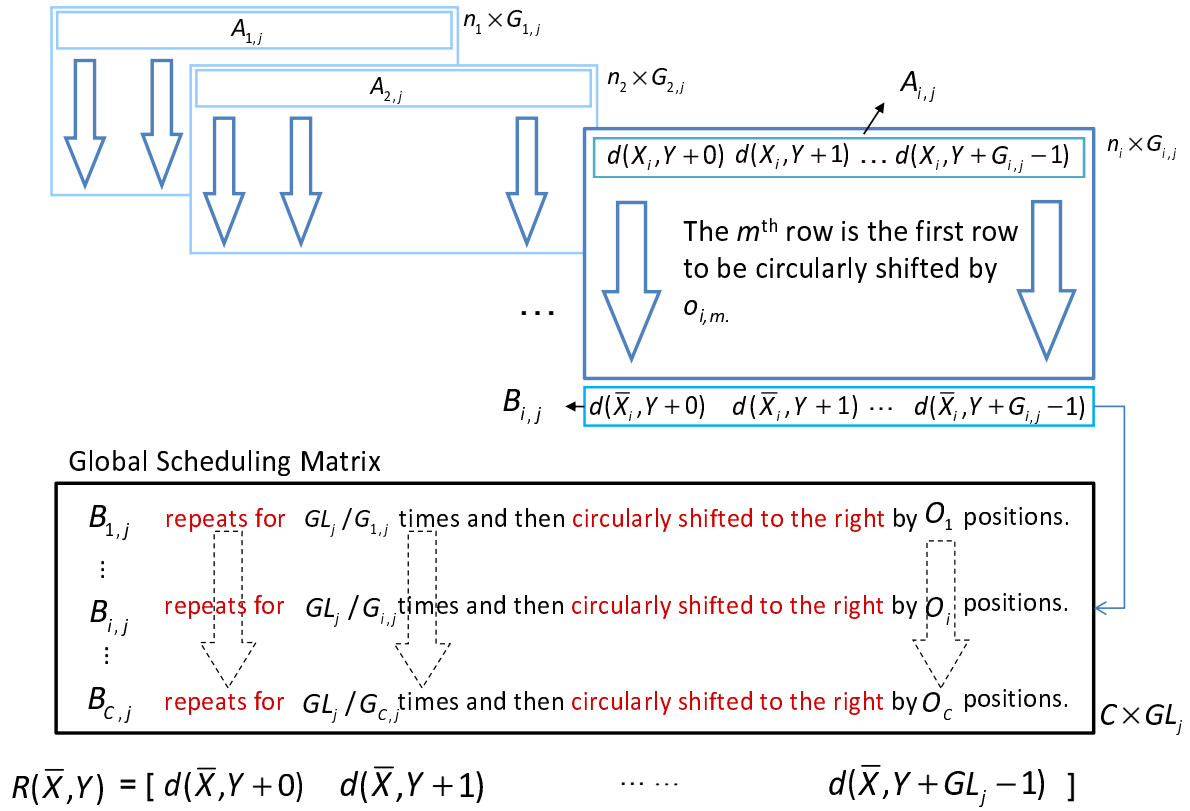

Figure 3.4: The illustration of constructing a global scheduling matrix.

the established periodic schedules and the current time since the notion of sequence can be extended to both directions of the time line as shown in our basic idea. Since the service delivered every SI, the delay for the starting of the established service is upper bounded by the value of its SI, rather than the period of the total repetition pattern of different periodic service schedules.

Example 3.2. Assume that $C = 2$, $n_1 = 2$, $n_2 = 1$, $p_1 = 6$, $p_2 = 9$, and a new TS in Class 2 is to be scheduled. Based on the assumptions, we have $G_{1,1} = 6$, $G_{1,2} = 3$, $G_{2,1} = 3$, and $G_{2,2} = 9$. Besides, the contents of $A_{i,j}[k]$ are given by $A_{1,1} = [0\ 1\ 2\ 3\ 2\ 1]$, $A_{1,2} = [0\ 1\ 1]$; and $A_{2,1} = [0\ 1\ 1]$, $A_{2,2} = [0\ 1\ 2\ 3\ 4\ 4\ 3\ 2\ 1]$. Let e_1 and e_2 represent the TSs in Class 1 with e_1 being the representative. Also, let f_1 be the representative TS in Class 2. Assume that TS e_1 was scheduled first, followed by TS f_1, and then TS e_2. As a result, when the new TS f_2 is to be scheduled, we have $O_2 = 1$ (which is randomly selected from 1 and 2) and $o_{1,2} = 3$

(which is randomly selected among 2, 3, and 5). At this moment, the contents of $B_{i,j}[k]$ are given by $B_{1,1}$ = [0 1 1 0 1 1], $B_{1,2}$ = [0 1 1]; and $B_{2,1}$ = [0 1 1], $B_{2,2}$ = [0 1 2 3 4 4 3 2 1]. The global scheduling matrix

$$M = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 2 & 3 & 4 & 4 & 3 & 2 \end{bmatrix}.$$

Therefore, the value of k^* can be chosen as 2, 4, 5, 7, or 8. We select $k^* = 5$ because it has maximum column sum. Then we compute $o_{2,2} = k^* - O_2 = 4$ and update $B_{2,1}$ = [0 0 1], $B_{2,2}$ = [0 1 2 1 0 1 2 2 1].

3.4 Performance Evaluations

The considered scenario is composed of periodic beacons and 5 classes of traffic. The 5 classes of traffic in the system are bi-directional real-time voice, realtime video, streaming audio, streaming video, and bi-directional real-time gaming. The traffic characteristics, listed in Table 3.3, are obtained from [36], [38], and [41]. The video traces are available online [49]. It is assumed that there are K STAs, each is configured with a scheduled TS belonging to one of the 5 classes. The number of STAs is increased in multiples of 5 STAs to keep the ratio of existing traffic classes. The system parameters are listed in Table 3.4 and conform with the IEEE 802.11a OFDM wireless LANs. The calculation for frame transmission time can be found in [14]. The repetition period L equals $lcm\{100, 40, 60, 150, 300\} = 600$ (ms) when all traffic classes, including the beacons, exist in the system. The precision used in our simulations is set to 100 μs.

3.4.1 Comparison of Computational Complexity

For the OAS-APSD algorithm shown in the end of Chapter 2, firstly it needs to insert the already scheduled events within L into the $List_{SE}$ and thus a sorting of the elements is required and takes $\log_2 K \times \sum_{i=1}^{K} L/p_i$ comparisons if the Merge Sort [40] is used. To find the two closest scheduled events for all the scheduled instants given a candidate of SST, by the

idea of the Insertion Sort [40], requires $\sum_{i=1}^{K} L/p_i$ comparisons. Since there are q candidates, the overall complexity $q \times \sum_{i=1}^{K} L/p_i$ comparisons. Computation of relative distances for all the q candidates takes $q \times 2 \times L/q = 2L$ subtractions. To find the minimum relative distances for all the q candidates requires $q \times 2 \times L/q = 2L$ comparisons. Finally, it takes q comparisons to determine the optimal SST. Note that the tie-breaking can be realized by using the sum of relative distances, rather than the average distance. It takes $2L - q$ additions to obtain those sum of relative distances and each tie-breaking needs one comparison. The comparisons of the online scheduling complexity are also listed in Table 3.5.

Except the analysis of the order of scheduling complexity, we also provide some numerical result here. In the numerical evaluation, we increase the number of existing TS in each class of application, and check the average online complexity of the OAS-APSD algorithm and that of the proposed algorithms. Given the number of existing TSs, the average complexity for finding the SST is derived by averaging the number of necessary online operations when a new TS belonging to each class of application joins. Since the $List_{SE}$ of OAS-APSD could be reused after it is established, the complexity of sorting is ignored here. The LCS-APSD (Low Complexity S-APSD)algorithm refers to the idea described in Section 3.2.1; however, to reduce complexity, the contents of $d(\boldsymbol{X}_i, \boldsymbol{Y} + k)$'s are reused for the TSs of the same class. Therefore, the bounded complexity in preparing the scheduling matrix for K existing TSs is reduced from $4 \times q \times K$ to $4 \times q \times C$ comparisons. Our proposed algorithm using the suggested implementation method presented in Section 3.3 is referred to as Class-based LCS-APSD (CLCS-APSD) algorithm. The required number of operations for the three algorithms and complexity reduction ratios when compared with the OAS-APSD algorithm are shown in Fig. 3.5. Here, the complexity reduction ratio is defined as

$$(N_{OAS\_APSD} - N_i)/N_{OAS\_APSD}, \tag{3.29}$$

where N_{OAS_APSD} is the number of operations (comparisons and subtractions) required by the OAS-APSD algorithm and N_i, $i = 1, 2$, is that required by our proposed Algorithm i. As

can be seen in Fig. 3.5, the average reduction ratio is as high as about 82% for LCS-APSD and 98% for CLCS-APSD when there are 50 TSs in the system.

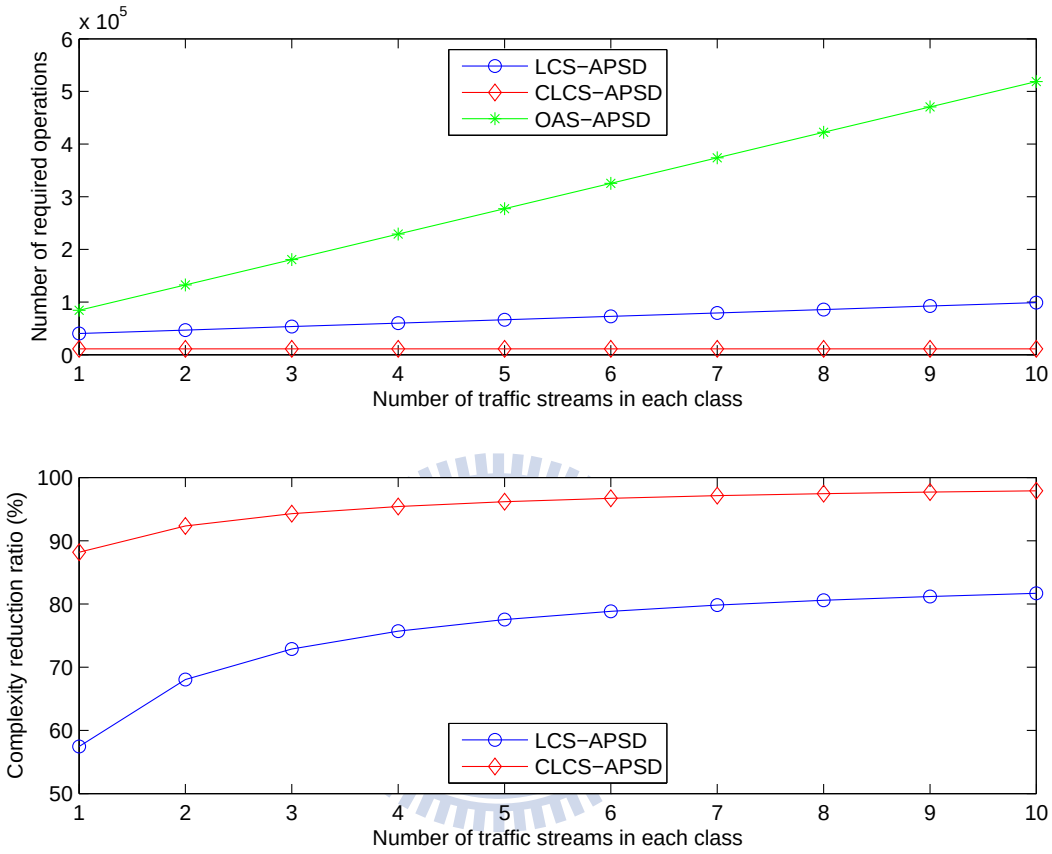

Figure 3.5: The performance of scheduling complexity.

3.4.2 Comparison of Power Consumptions

In this evaluation, we fix the number of existing TSs at 50 (10 for each class) and use the OAS-APSD and our proposed algorithms to schedule those TSs. We assume that the $(5 \times \ i + j)^{th}$ TS belongs to Class j for $0 \leq i \leq 9$ and $1 \leq j \leq 5$. The simulation is performed to model 600 seconds of the real time. During the simulations, when the SPs of the scheduled transmissions are overlapped, we give higher priorities to the TSs/STAs with earlier adding order and thus the STAs with lower priorities would tend to spend more energy in waiting. As revealed in Fig. 3.6, the power consumptions relate to the amount of traffic and the assigned priorities. In general, the resulted power consumptions for the TSs of the

OAS-APSD and our proposed algorithms are quite close. Although the proposed algorithm spends a bit more of power than that of the OAS-ASPD algorithm on average, the online scheduling complexity reduces significantly. Furthermore, compared with the case that the STA operates in Active mode during the simulations and it takes 1.4W, the simulation result justifies the necessity of using S-APSD to save energy even during active sessions.

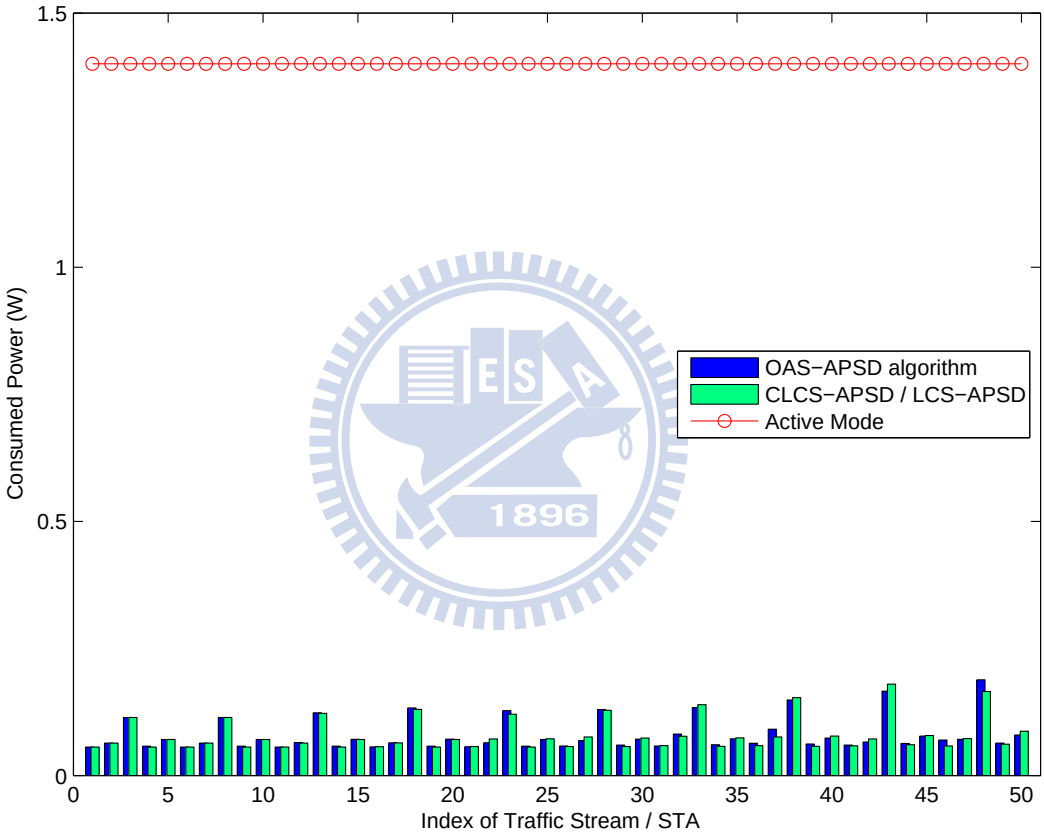

Figure 3.6: The performance of power consumptions.

3.4.3 Comparison of Affordable Transmissions

Here we remain the same setting as previous evaluation. Since the medium is not saturated, there are still empty time slots available for EDCA transmissions between scheduled SPs. Therefore, we check the average numbers of affordable transmissions in the available time within the repetition period L for frame transmission time ranging from 100 μs to 1000 μs. We plot in Fig. 3.7 the comparison of our proposed algorithms with the OAS-APSD. Note

that the affordable transmissions for our proposed LCS-APSD algorithm and CLCS-APSD algorithm are similar to each other. As one can see in Fig. 3.7, the average numbers of affordable transmissions are very close for our proposed algorithms and the OAS-APSD, which means that the distributions of the scheduled SPs for both scheduling algorithms are similar, even though the tie breaking criteria are different.

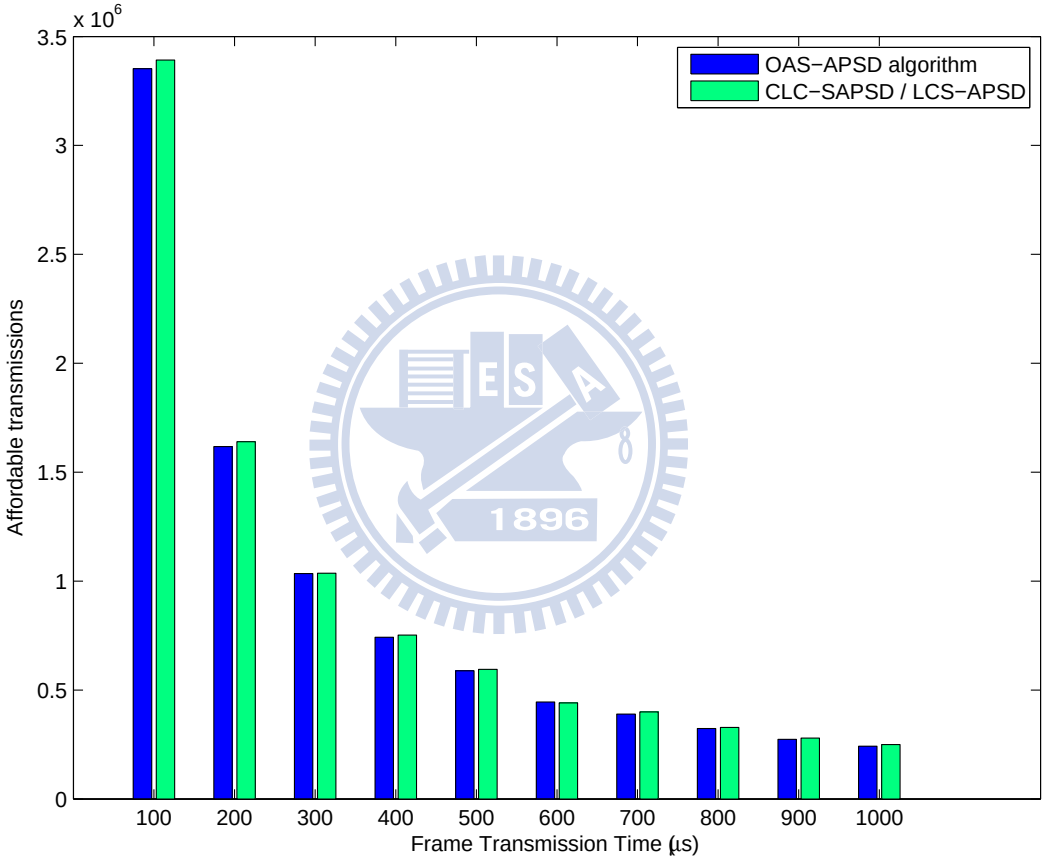

Figure 3.7: Comparison of affordable transmissions.

3.5 Summary

Compared with PSM, the S-APSD scheme defined in IEEE 802.11e provides a better mechanism to increase power saving performance when delivering QoS-sensitive traffic. In this chapter we focus on designing a feasible non-contiguous scheduling algorithm to be used for S-APSD. Our design takes advantage of the periodicity property of schedule to largely re-

duce online computational complexity. We also present an efficient implementation method for class-based systems. As demonstrated in performance comparison, the resulted power consumptions and the distributions of scheduled instants of TSs are similar for the proposed CLCS-APSD/LCS-APSD algorithms and the OAS-APSD algorithm, while the online computational complexity of our proposed algorithms is much smaller than that of previous related work.

Table 3.1: Notations used in Section 3.2

Notations	Descriptions
$\boldsymbol{X}_i$	A periodic sequence $\boldsymbol{X}_i = \{x_{i,m}\}_{m=-\infty}^{\infty}$, such that $x_{i,m} = x_{i,m-1} + p_i$ and i is the index of a TS, $1 \leq i \leq K$.
$\boldsymbol{X}_i^{'}$	The shifted version of $\boldsymbol{X}_i$ such that $\boldsymbol{X}_i^{'} = \boldsymbol{X}_i + O_i$. $\boldsymbol{X}_i^{'}$ stands for the scheduled instants of the i^{th} existing TS.
$\boldsymbol{X}$	The aggregation of scheduled instants of $\boldsymbol{X}_i^{'}$: $\boldsymbol{X} = \bigcup_{1 \leq i \leq K} \boldsymbol{X}_i^{'}$.
$\boldsymbol{Y}$	The periodic sequence used to represent the new joining TS: $\boldsymbol{Y} = \{y_m\}_{m=-\infty}^{\infty}$; $\boldsymbol{Y} + k = \{y_m + k\}_{m=-\infty}^{\infty}$.
O_i	The offset of the i^{th} TS: $O_1 = 0$ and $O_i = x_{i,0} - x_{1,0}$. $0 \leq O_i \leq p_i - 1$.
L_i	The lowest common multiple of of p_i and q, i.e., $L_i = lcm\{p_i, q\}$.
G_i	The greatest common divisor of of p_i and q, i.e. $G_i = \gcd(p_i, q)$.
GL	The lowest common multiple of of $G_1, G_2, ..., G_K$, i.e., $GL = lcm\{G_1, G_2, ..., G_K\}$.
$d(\boldsymbol{X}_i^{'}, \boldsymbol{Y} + k)$	The distance between the i^{th} TS and the new joining TS with offset k.
$d(\boldsymbol{X}, \boldsymbol{Y} + k)$	The minimum distance between all TSs and the new joining TS with offset k.

Table 3.2: Notations used in Section 3.3

Notations	Descriptions
$\boldsymbol{X}_i$	A periodic sequence $\boldsymbol{X}_i = \{x_{i,m}\}_{m=-\infty}^{\infty} = \{p_i \times m\}_{m=-\infty}^{\infty}$ represents the representative TS e_1 of Class i, $1 \leq i \leq C$.
$\boldsymbol{X}_{i,s}$	The scheduled instants for TS e_s of Class i, such that $\boldsymbol{X}_{i,s} = \boldsymbol{X}_i + o_{i,s}$, $1 \leq s \leq n_i$.
$\overline{\boldsymbol{X}}_{\boldsymbol{i}}$	The aggregation of scheduled instants of all Class i TSs, i.e., $\overline{\boldsymbol{X}}_{\boldsymbol{i}} = \bigcup_{1 \leq s \leq n_i} \boldsymbol{X}_{i,s}$.
$\overline{\boldsymbol{X}}$	The aggregation of scheduled instants of all Classes of TSs, i.e., $\overline{\boldsymbol{X}} = \bigcup_{1 \leq i \leq C} \overline{\boldsymbol{X}}_i$.
$\boldsymbol{Y}$	A periodic sequence to represent the new joining TS belonging to Class j: $\boldsymbol{Y} = \{y_m\}_{m=-\infty}^{\infty} = \{p_j \times m\}_{m=-\infty}^{\infty}$.
O_i	The inter-class offset of the Class i representative TS with respect to the representative TS of Class 1 .
$o_{i,s}$	The intra-class offset of TS e_s with respect to the representative TS e_1 within Class i.
$G_{i,j}$	The greatest common divisor of of p_i and p_j, i.e., $G_{i,j} = \gcd(p_i, p_j)$.
GL_j	The lowest common multiple of of $G_{1,j}, G_{2,j}, ..., G_{C,j}$, i.e., $GL_j = lcm\{G_{1,j}, G_{2,j}, ..., G_{C,j}\}$.
$d(\boldsymbol{X}_i, \boldsymbol{Y} + k)$	The distance between the Class i representative TS and the new joining TS with offset k.
$d(\overline{\boldsymbol{X}}_i, \boldsymbol{Y} + k)$	The distance between all Class i TSs and the new joining TS with offset k.
$R(\overline{\boldsymbol{X}}_i, \boldsymbol{Y})$	A vector stores the distances between all Class i TSs and the new joining TS. The k^{th} element of this vector is the distance when the offset of the new joining TS is k.
$R(\overline{\boldsymbol{X}}, \boldsymbol{Y})$	A vector stores the distances between all Classes of TSs and the new joining TS. The k^{th} element of this vector is the distance when the offset of the new joining TS is k.
$A_{i,j}$	Store [$d(\boldsymbol{X}_i, \boldsymbol{Y}+0)$ $d(\boldsymbol{X}_i, \boldsymbol{Y}+1)$ ... $d(\boldsymbol{X}_i, \boldsymbol{Y}+G_{i,j}-1)$] which represents the impact of the representative TS e_1 of Class i to a new TS of Class j.
$B_{i,j}$	Store the value of $R(\overline{\boldsymbol{X}}_i, \boldsymbol{Y})$ which represents the impact of all TSs in Class i to a new TS of Class j.

Table 3.3: Traffic Characteristics.

Class	Application	Mean Data Rate	Service Interval
1	Real-time Gaming	20 Kbps	100 ms
2	Real-time Voice (G.711)	64 Kbps	40 ms
3	Real-time Video (MPEG 4 Trace: Lecture Room Cam)	58 Kbps	60 ms
4	Streaming Audio	128 Kbps	150 ms
5	Streaming Video (MPEG 4 Trace: First Contact)	330 Kbps	300 ms

Table 3.4: System parameters

Parameter	**Value**
PHY Data Rate	54 Mbps
PHY Control Rate	6 Mbps
Transmission time for PHY header and preambles	20 μs
Transmission time for an OFDM symbol	4 μs
SIFS	16 μs
Slot Time	9 μs
MAC frame header	30 bytes
ACK frame	14 bytes
IP header	20 bytes
UDP header	8 bytes
RTP header	12 bytes
Power Consumption in Awake state	1.4 W
Power Consumption in Doze state	0.045 W
Hardware delay of switchover	250 μs

Table 3.5: Online Complexity Comparisons. (In Comparison/Subtraction/Addition)

Algorithm	Distance preparation	Find minimum distance	Find max of min's	Tie-breaking
OAS-APSD	$q \times \sum_{i=1}^{K} L/p_i + 2L$	$q \times 2 \times L/q$	q	$2L - q$
LCS-APSD	$4 \times q \times C$	$K \times GL$	GL	$2(K-1)$
CLCS-APSD	offline	$C \times GL_j$	GL_j	$2(C-1)$

Chapter 4

Rearrangement Algorithms for the IEEE 802.11e S-APSD

As introduced in Chapters 2 and 3, Overlapping-Aware S-APSD (OAS-APSD) [38] and Low Complexity S-APSD (LCS-APSD) [44] are non-contiguous scheduling algorithms for the S-APSD mechanism. Given the layouts of existing scheduled events (SEs), in order to reduce overlapping probability of SPs, the OAS-APSD and the LCS-APSD scheduling algorithms determine the SST of a new TS so that the minimum distance between the scheduled instants of the new TS and the SEs is maximized. These two algorithms are greedy in the sense that, at each stage, with the hope of finding the global optimum, it always makes the choice that looks the best for now [40]. Obviously, their performances depend on the order that TSs join the system and the resulted layouts of SEs. In many cases, the results differ much from each other and could be unsatisfactory. Therefore, to improve the power saving performance by reducing the chance of SP overlapping, the purpose of this work is to investigate the problem of rearranging the offsets of SEs to achieve the global optimum, i.e. to obtain the largest possible minimum distance between the SEs for the considered system.

The rest of this chapter is organized as follows. The system model is described in Section 4.1. Section 4.2 presents the brute-force searching method and provides a naive equal-spacing algorithm with the intention to reduce the implementation complexity. In Section 4.3, we prove an upper bound for the system minimum distance and design *coprime-avoiding*

scheduling algorithms, named as G_{min}-Maintaining Decomposition greedy scheduling and Simply Sorted greedy scheduling, which can avoid the worst case residing in the naive equal-spacing algorithm. The performance evaluation is presented in Section 4.4. Finally, we draw the summary in Section 4.5.

4.1 System Model

As described in Chapter 3, the system for the scheduling is actually time-slotted with slot size equal to the duration of precision used in IEEE 802.11e. Without loss of generality, we shall normalize the duration of precision to one. As a result, the scheduled instants for a specific TS with SI p can be represented as a periodic sequence. Let

$$\boldsymbol{X} = \{x_r\}_{r=-\infty}^{\infty}, \text{ such that } x_r = p \times r \text{ for all } r. \tag{4.1}$$

Let $\boldsymbol{Y} = \{y_l\}_{l=-\infty}^{\infty}$, such that $y_l = q \times l$ for all l, be another periodic sequence with period q and $\boldsymbol{Y} + k = \{y_l + k\}_{l=-\infty}^{\infty}$ be its shifted version. We call k the offset of $\boldsymbol{Y} + k$ with respect to $\boldsymbol{Y}$. Define the distance between x_r and $y_l + k$ as

$$d(x_r, y_l + k) = |y_l + k - x_r| \tag{4.2}$$

and define the distance between $\boldsymbol{X}$ and $\boldsymbol{Y} + k$ as

$$d(\boldsymbol{X}, \boldsymbol{Y} + k) = \min_{-\infty \le l,r \le \infty} d(x_r, y_l + k). \tag{4.3}$$

It is clear that $d(\boldsymbol{X}, \boldsymbol{Y}+k) = d(\boldsymbol{Y}+k, \boldsymbol{X}) = d(\boldsymbol{X}-k, \boldsymbol{Y})$ and $d(\boldsymbol{X}, \boldsymbol{Y}+0) = d(\boldsymbol{X}, \boldsymbol{Y}) = 0$. Let $G = gcd(p, q)$ be the greatest common divisor of p and q. In Chapter 3, the periodicity of $d(\boldsymbol{X}, \boldsymbol{Y} + k)$ is proved and is referred to as Lemma 4.1 in this chapter.

Lemma 4.1. *$d(\boldsymbol{X}, \boldsymbol{Y}+k)$ is periodic in k with period G, i.e, $d(\boldsymbol{X}, \boldsymbol{Y}+k) = d(\boldsymbol{X}, \boldsymbol{Y}+G+k)$.*

Assume now that there are K already scheduled TSs in the system when a new TS with period q is to be scheduled. The period of the m^{th} existing TS is p_m, $1 \le m \le$

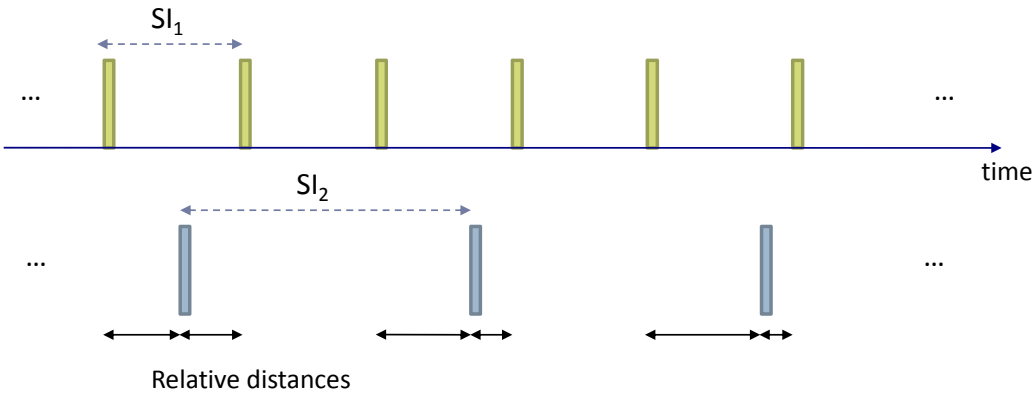

Figure 4.1: Illustration of relative distances between two periodic sequences.

K. Let $\boldsymbol{X}_m = \{x_{m,r}\}_{r=-\infty}^{\infty} = \{p_m \times r\}_{r=-\infty}^{\infty}$ be a periodic sequence with period p_m. Let $\overline{\boldsymbol{X}}_m = \{\overline{x}_{m,r}\}_{r=-\infty}^{\infty} = \boldsymbol{X}_m + O_m = \{p_m \times r + O_m\}_{r=-\infty}^{\infty}$ be the scheduled instants of the m^{th} existing TS. We shall use $\boldsymbol{X}_1$ as reference and, without loss of generality, assign $O_1 = 0$. Consequently, we have $\overline{\boldsymbol{X}}_1 = \boldsymbol{X}_1$ and $O_m = \overline{x}_{m,0} - \overline{x}_{1,0}$ represents the offset of $\overline{\boldsymbol{X}}_m$ with respect to $\overline{\boldsymbol{X}}_1$. Let $\overline{\boldsymbol{X}} = \bigcup_{1 \le m \le K} \overline{\boldsymbol{X}}_m$ such that x is an element of $\overline{\boldsymbol{X}}$ if and only if (iff) it is an element $\overline{\boldsymbol{X}}_m$ of for some m. Clearly, $\overline{\boldsymbol{X}}$ is periodic with period $lcmp_1, p_2, ..., p_K$, the least common multiple of p_1, p_2 and p_K. Define $d(\overline{\boldsymbol{X}}_m, \boldsymbol{Y} + k) = \min_{-\infty \le l, r \le \infty} |y_l + k - x_{m,r} - O_m|$ and $d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + k) = \min_{1 \le m \le K} d(\overline{\boldsymbol{X}}_m, \boldsymbol{Y} + k)$. Let $G_m = gcd\{p_m, q\}$ and $L_{K+1} = lcm\{G_1, G_2, ..., G_K\}$. $d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + k)$ is periodic with period L_{K+1} because $d(\overline{\boldsymbol{X}}_m, \boldsymbol{Y} + k)$ is periodic with period G_m according to Lemma 4.1. The goal of scheduling in LCS-APSD is to find k^*, the optimal value of which satisfies

$$k^* = \arg\max_k d(\overline{\boldsymbol{X}}, \boldsymbol{Y} + k), 0 \le k^* \le L_{K+1}. \tag{4.4}$$

In a real system, it is likely that TSs can be classified into a finite number of classes according to their periods. Assume that there are C classes of TSs such that two TSs are in the same class if and only if they have the same period and the K existing TSs are classified. Let p_i and p_j represent the periods of Class i and j, $1 \le i, j \le C$, respectively. The impact of the TS of Class i to a Class j TS can be pre-calculated and stored in advance. Let $d(\boldsymbol{X}_i, \boldsymbol{X}_j + k)$ be the distance between a Class i sequence $\boldsymbol{X}_i = \{p_i \times r\}_{r=-\infty}^{\infty}$ and another

Class j sequence $\boldsymbol{X}_j + k = \{p_j \times l + k\}_{l=-\infty}^{\infty}$. The impact can be represented by an array $A_{i,j}$, $1 \leq i, j \leq C$, which stores

$$[\ d(\boldsymbol{X}_i, \boldsymbol{X}_j + 0)\ d(\boldsymbol{X}_i, \boldsymbol{X}_j + 1)\ ...\ d(\boldsymbol{X}_i, \boldsymbol{X}_j + G_{i,j} - 1)\ ], \tag{4.5}$$

where $G_{i,j} = gcd(p_i, p_j)$ is the greatest common divisor of the period of Class i and Class j. For LCS-APSD, the impact of all existing TSs to the new Class j TS can be obtained by constructing a *scheduling matrix*[1] of size $K \times L_{K+1}$ according to corresponding $A_{i,j}$'s and offsets.[2]

4.2 The Rearrangement of Scheduled Events

Rearrangement is a process to simultaneously schedule all existing TSs irrespective of the order they join the system. Let n_i represent the number of TSs in Class i and assume that there are K TSs in the system, $K = \sum_{i=1}^{C} n_i$. Denote $\{O_1, O_2, ..., O_K\}$ the offsets of the K TSs. Given offsets O_m and O_n, the distance between TS m and TS n, $1 \leq m, n \leq K$, is defined as $d(\overline{\boldsymbol{X}}_m, \overline{\boldsymbol{X}}_n) = d(\boldsymbol{X}_m + O_m, \boldsymbol{X}_n + O_n) = \min_{-\infty \leq l,r \leq \infty} |x_{n,l} + O_n - x_{m,r} - O_m|$. Moreover, given $\{O_1, O_2, ..., O_K\}$, the minimum distance of the system, denoted as $min\_dist(O_1, O_2, ..., O_K)$, is defined as

$$min\_dist(O_1, O_2, ..., O_K) = \min_{1 \leq m,n \leq K} d(\overline{\boldsymbol{X}}_m, \overline{\boldsymbol{X}}_n) \tag{4.6}$$

which is defined as the minimum distance of the system. The goal of rearrangement is to determine the optimal offsets $\{O_1^*, O_2^*, ..., O_K^*\}$ which satisfies

$$\{O_1^*, O_2^*, ..., O_K^*\} = \arg \max_{\{O_1, O_2, ..., O_K\}} min\_dist(O_1, O_2, ..., O_K). \tag{4.7}$$

[1] Please refer to Chapter 3 for the description of scheduling matrix.

[2] Since the impacts have been pre-calculated, the online complexity in determining the SST is much less than that of OAS-APSD.

4.2.1 The Brute-Force Searching Method

The brute-force searching method is to enumerate all possible combinations of the offset and derive the optimal solution $\{O_1^*, O_2^*, ..., O_K^*\}$. Before the exhaustive search, the process to prepare $d(\overline{\boldsymbol{X}}_m, \overline{\boldsymbol{X}}_n)$ is described below.

Without loss of generality, assume that TS 1 belongs to Class i with period p_i and is assigned offset $O_1 = 0$. Assume further that TS 2 belongs to Class j and is assigned offset O_2. As described in the system model, we can pre-calculate an array $A_{i,j}$ which stores

$$[\ d(\boldsymbol{X}_i, \boldsymbol{X}_j + 0)\ d(\boldsymbol{X}_i, \boldsymbol{X}_j + 1)\ ...\ d(\boldsymbol{X}_i, \boldsymbol{X}_j + G_{i,j} - 1)\ ], \tag{4.8}$$

where the k^{th} element, $0 \leq k \leq G_{i,j} - 1$, represents the distance for a sequence with period p_i to a sequence with period p_j when their relative offset modulo $G_{i,j}$ is equal to k. Therefore, the distance between TS 1 and TS 2 can be obtained by

$$d(\overline{\boldsymbol{X}}_1, \overline{\boldsymbol{X}}_2) = A_{i,j}[(O_2 - O_1) \mod G_{i,j}]. \tag{4.9}$$

The reason why modulo operation is necessary is described as follows. By the periodicity of the distance, $d(\boldsymbol{X}_1 + O_1, \boldsymbol{X}_2 + O_2) = d(\boldsymbol{X}_1 + O_1, \boldsymbol{X}_2 + O_2 + G_{i,j})$, it seems that the offset O_2 should be chosen from the range in between 0 and $G_{i,j} - 1$ since the same value repeats every shift of $G_{i,j}$; however, since the other $K - 2$ TSs may belong to classes other than i or j, the periodicity between Class j and other classes should also be considered in determining the offset. Therefore, the value that O_2 can take is from 0 to $GL_2 - 1$, where $GL_2 = lcm\{G_{1,j}, G_{2,j}, ..., G_{C,j}\}$, the least common multiple of the GCDs between the period of TS 2 (belonging to Class j) and those of other existing classes. Similarly, if O_3 has been assigned and TS 3 belongs to Class k, we can obtain the distances between TS 3, TS 2, and TS 1, by further checking the value of $d(\overline{\boldsymbol{X}}_1, \overline{\boldsymbol{X}}_3)$ and $d(\overline{\boldsymbol{X}}_2, \overline{\boldsymbol{X}}_3)$ by looking up $A_{i,k}$ and $A_{j,k}$, as shown in Eq. (4.9). Therefore, for a given set of $\{O_1, O_2, ..., O_K\}$, we can have the distances between any pair of TSs, $d(\overline{\boldsymbol{X}}_m, \overline{\boldsymbol{X}}_n)$, $1 \leq m, n \leq K$, by looking up the corresponding $A_{i,j}$'s, $1 \leq i, j \leq C$.

With the above setting, the brute-force searching is to examine all possible combinations of $\{O_1, O_2, \ldots, O_K\}$and compare the derived $min\_dist(O_1, O_2, ..., O_K)$'s. The complexity for this exhaustive search takes $\binom{K}{2} \times \prod_{i=2}^{K} GL_i$ of comparisons to find all $min\_dist(O_{1,2}, ..., O_K)$, which is bounded by $\binom{K}{2} \times \prod_{i=2}^{K} p_i$. The optimal set $\{O_1^*, O_2^*, ..., O_K^*\}$ which brings the maximum of system minimum distance can be determined after $\prod_{i=2}^{K} p_i$ comparisons. Clearly, this method becomes infeasible for real system since the complexity increases dramatically with the number of TSs. Therefore, alternate low-complexity solutions would be necessary.

4.2.2 The Naive Equal-Spacing Method

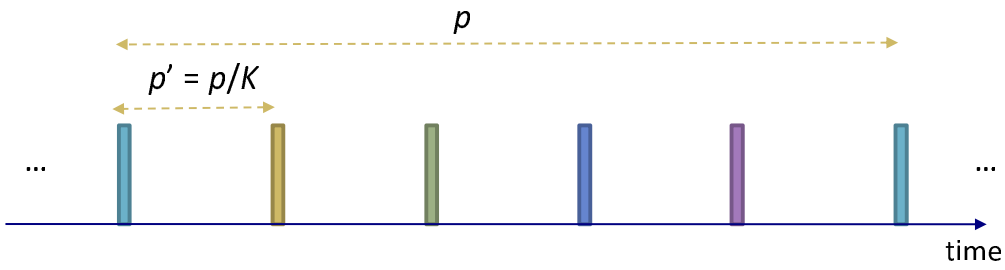

Figure 4.2: An illustration of strict-sense equal spacing scheduling ($K = 5,\ K|p$).

Assume that there are K TSs served by the same period p in the system. It is clear that p can be expressed as below:

$$p = K \times \left\lfloor \frac{p}{K} \right\rfloor + r, \quad 0 \leq r < K. \tag{4.10}$$

A scheduling algorithm is *wide-sense* equal-spacing if it generates the minimum distance of $\lfloor p/K \rfloor$ for these K TSs. Note that, in general, there could be multiple wide-sense equal-spacing scheduling algorithms if $r > 0$. If $r = 0$, then wide-sense equal-spacing becomes *strict-sense* equal-spacing because the relative distance between any two scheduled instants is equal to p/K. In this case, the offset for the m^{th} TS relative to the first TS can be easily selected as $O_m = (m-1) \times p, 1 \leq m \leq K$. We prove here a property of scheduling K TSs of the same period p.

Lemma 4.2. *When there are K TSs to be served by the same period p, the maximum achievable minimum distance can only be obtained by performing wide-sense equal spacing scheduling for these K TSs.*

Proof. Assume that there exists a non-wide-sense equal-spacing scheduling algorithm, say, Algorithm A, which provides a minimum distance $d > \lfloor p/K \rfloor$. Let $O_1 = 0$ and $O_m, 2 \leq m \leq K$, represent the offset of the m^{th} TS relative to the first TS, using scheduling algorithm A. Without loss of generality, assume that $O_{m+1} > O_m$. Since it has been shown [3] that $0 \leq O_m \leq p-1$, the offsets $\{O_1, O_2, ..., O_K\}$ shall produce K spacing between 0 and p. The m^{th} spacing resulted by Algorithm A is left-bounded by O_m and right-bounded by O_{m+1} and its length is denoted as $d_m = O_{m+1} - O_m, 1 \leq m < K$. In addition, we have $d_K = p - O_K$.

Since $d_m \geq d > \lfloor p/K \rfloor$ for all m, it holds that

$$\sum_{m=1}^{K} d_m \;\geq\; K \times d \geq K \times \Big(\lfloor p/K \rfloor + 1 \Big) > K \times \lfloor p/K \rfloor + r = p,$$

However, the result contradicts the fact that the sum of spacings should constitute the period p. This completes the proof of Lemma 4.2. □

One naive scheduling algorithm is to schedule TSs with identical periods based on the wide-sense equal-spacing concept, treat the results as the scheduled instants of an aggregated TS, and find the optimal offsets for the aggregated TSs. Such an approach reduces complexity but often results in unsatisfactory results. To see this, let us consider the special case that $n_i | p_i, 1 \leq i \leq C$, so that strict-sense equal-spacing is adopted to obtain the aggregated TSs. Under this scenario, there are C aggregated TS and the one obtained for Class i TSs is periodic with period $p_i^{'} = p_i/n_i$. The intra-class offsets can be easily determined as $o_{i,k} = (k-1) \times p_i^{'}, 1 \leq k \leq n_i$.

Given the aggregated TSs, one can schedule them using greedy method or brute force search. Since the brute-force search requires high complexity, the greedy method is more

[3] Shown by Eq. (3.19) in Section 3.2.2.

preferable. However, unlike the greedy methods in [38] and [44] which schedule the TSs according to the order of joining, here we introduce the idea of scheduling period-revised TSs one by one according to ascending order of revised periods. The reason of adopting the scheduling by ascending order is stated as follows. The smaller period implies that the periodic scheduled instants would repeat more frequently than that of the larger ones. Therefore, in a given repetition interval, there are more scheduling instants for the TS with shorter period to be placed into. Also, the choices for the SST are fewer for the TS with shorter period since it had been proved in Chapter 3 that the offset O_i satisfies $0 \leq O_i \leq p_i - 1$. Consequently, the shorter the period is, the higher the scheduling priority should be. We set the TS with shortest revised-period as the reference, and then, by the ascending order of size of the revised period, we add the remaining TSs one after another by gradually constructing the scheduling matrices as described in Chapter 3 and determining the inter-class offsets. For convenience, this scheduling approach is referred to as the *sorted greedy* method in this work.

With the above setting, the naive equal-spacing method may, however, produce unsatisfactory result. Let $p_i^{'}$ and $p_j^{'}$ be the revised periods of aggregated TSs $\boldsymbol{X}^{'}$ and $\boldsymbol{Y}^{'}$, respectively. If $gcd(p_i^{'}, p_j^{'}) = 1$, then, according to Lemma 4.1, we have $d(\boldsymbol{X}^{'}, \boldsymbol{Y}^{'} + k) = 0$ for all k. This observation motivates us to develop designs to avoid any pair of the revised periods being coprime.

4.3 The Coprime-Avoiding Scheduling Algorithms

Consider two sequences $\boldsymbol{X} = \{x_r\}_{r=-\infty}^{\infty} = \{p \times r\}_{r=-\infty}^{\infty}$ and $\boldsymbol{Y} = \{y_l\}_{l=-\infty}^{\infty} = \{q \times l\}_{l=-\infty}^{\infty}$. $G = gcd(p, q)$. It is clear that if $y_l \in \boldsymbol{Y}$, then $y_l + k \in \boldsymbol{Y} + k$. To develop our idea, we prove in Theorem 4.1 a property of $d(\boldsymbol{X}, \boldsymbol{Y})$. We firstly present Lemma 4.3 before Theorem 4.1 to facilitate the proof process.

Lemma 4.3. *It holds that*

$$d(x_r, y_l + k + 1) = \begin{cases} d(x_r, y_l + k) + 1, & \text{if } x_r \leq y_l + k \\ d(x_r, y_l + k) - 1, & \text{if } x_r > y_l + k \end{cases} . \tag{4.11}$$

Proof. By the definition of distance, if $x_r \leq y_l + k$, we have $d(x_r, y_l + k + 1) = |y_l + k + 1 - x_r| = |y_l + k - x_r| + 1 = d(x_r, y_l + k) + 1$. Similarly, if $x_r > y_l + k$, we have $d(x_r, y_l + k + 1) = |y_l + k + 1 - x_r| = |y_l + k - x_r| - 1 = d(x_r, y_l + k) - 1$. This completes the proof. □

Theorem 4.1. *It holds that*

$$d(\boldsymbol{X}, \boldsymbol{Y} + k) = \begin{cases} k, & 0 \leq k \leq \lfloor G/2 \rfloor \\ G - k, & \lceil G/2 \rceil \leq k \leq G - 1 \end{cases} . \tag{4.12}$$

Proof. Assume that G is an even number. We claim that $d(\boldsymbol{X}, \boldsymbol{Y}+k) = k$ for $0 \leq k \leq (G/2)$. It is clear that $d(\boldsymbol{X}, \boldsymbol{Y}) = 0$. Since $d(x_0, y_0 + k) = k$, we know that $d(\boldsymbol{X}, \boldsymbol{Y} + k) \leq k$. Assume that there exist some x_r and y_l such that $d(x_r, y_l + k) = d' < k$. If $x_r \leq y_l + k$, then, according to Lemma 4.3, we have $d(x_r, y_l + k - d') = 0$, which contradicts Lemma 4.1 because $0 < k - d' < G$. Assume that $x_r > y_l + k$. According to Lemma 4.3, we have $d(x_r, y_l + k + d') = 0$ which implies $d(\boldsymbol{X}, \boldsymbol{Y} + k + d') = 0$. This contradicts Lemma 4.1 again because $k + d' < 2k \leq G$. Therefore, we conclude that $d(\boldsymbol{X}, \boldsymbol{Y} + k) = k$ for $0 \leq k \leq (G/2)$.

Consider now that $(G/2) + 1 \leq k \leq G - 1$. Since $d(\boldsymbol{X}, \boldsymbol{Y} + k) = d(\boldsymbol{X} - k, \boldsymbol{Y}) = d(\boldsymbol{X}+G-k, \boldsymbol{Y})$ by Lemma 4.1, we have $d(\boldsymbol{X}, \boldsymbol{Y}+k) \leq G-k$ because $d(x_0+G-k, y_0) = G-k$. Assume that there exist some x_r and y_l such that $d(x_r, y_l + k) = d' < G - k$. If $x_r \leq y_l + k$, then Lemma 4.3 implies $d(x_r, y_l+k-d') = 0$ and thus $d(\boldsymbol{X}, \boldsymbol{Y}+k-d') = 0$, which contradicts Lemma 4.1 because $k-d' > 2k-G > 0$ and $k-d' \leq (G-1)-d' < G$. Assume that $x_r > y_l+k$. From Lemma 4.3 we have have $d(x_r, y_l + k + d') = 0$ which implies $d(\boldsymbol{X}, \boldsymbol{Y} + k + d') = 0$. This contradicts Lemma 4.1 again because $0 < k + d' < G$. Therefore, Theorem 4.1 is also true for $(G/2) + 1 \leq k \leq G - 1$. This completes the proof of Theorem 4.1 for G being even. The proof for odd values of G is similar except that the largest achievable distance becomes $\lfloor G/2 \rfloor$. □

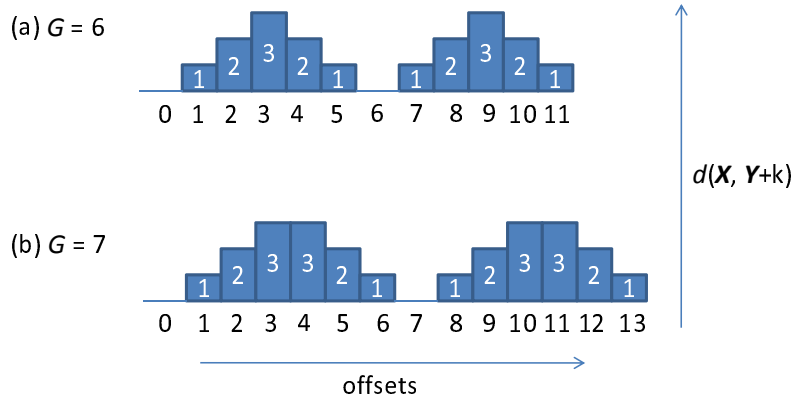

Figure 4.3: The relationship between the offset, distance, and GCD's.

As a consequence of Theorem 4.1, $\lfloor G/2 \rfloor$ is the largest possible value for $d(\boldsymbol{X}, \boldsymbol{Y} + k)$ since $G/2 \geq \lfloor G/2 \rfloor = G - \lceil G/2 \rceil$. Therefore, we find that the largest possible minimum distance between two periodic sequences is actually strictly upper bounded by the value of G rather than p or q. For C classes, define the minimum GCD among the pairwise GCD's as G_{min}, i.e.,

$$G_{min} = \min_{1 \leq i,j \leq C.\ i \neq j} G_{i,j}. \tag{4.13}$$

Then the maximum achievable minimum distance of system is upper bounded by $\lfloor G_{min}/2 \rfloor$. Therefore, to reserve the value of minimum distance for the system when rearranging the existing TSs, the value of G_{min}'s should be carefully treated.

As an example, consider a scenario with $C = 5$. The five traffic classes are real-time voice, video, streaming audio, video, and real-time gaming whose SIs are $40, 60, 150, 300$, and $100ms$, respectively. Assume that the used precision is $100\mu s$. As a result, the G_{min} is 100 and $G_{min}|G_{i,j}$, $1 \leq i, j \leq C$. It holds that $G_{min}|p_i$ and we can express pi as

$$p_i = G_{min} \times f_i. \tag{4.14}$$

From Eq. (4.14), it is clear that G_{min} remains intact if p_i is divided by any factor of f_i. Therefore, the equal spacing can be employed without altering the G_{min} as long as n_i, the number of TSs in Class i, is a factor of f_i. The following sub-section presents a decomposition algorithm to handle the case when n_i is not a factor of f_i.

4.3.1 The G_{min}-Maintaining Decomposition Greedy Scheduling

In this sub-section we propose an algorithm, named as the G_{min}-Maintaining Decomposition greedy scheduling (GMD greedy scheduling) with the intention to increase the performance of rearrangement while keeping the complexity to be feasible. The idea of equal spacing is still exploited in this algorithm to reduce scheduling complexity.

Assume that $p_i = G_{min} \times f_i$, we factorize f_i and let

$$F_i = \{f_{i,1}, f_{i,2}, ..., f_{i,u_i}\} = \{f_i, ..., 1\} \tag{4.15}$$

be the set of complete factors of f_i, where $f_{i,m} > f_{i,m+1}$, and u_i is the cardinality of F_i. For a group with $f_{i,k}$ members, the revised period after performing equal spacing scheduling becomes $p_i/f_{i,k} = p_{i,k}$.

Given G_{min}, the decomposition of n_i is to find the integers $q_{i,k}$'s ≥ 0 such that $n_i = \sum_{k=1}^{ui} q_{i,k} \times f_{i,k}$. The decomposition is optimal iff it maintains G_{min} and $\sum_{k=1}^{ui} q_{i,k}$ is minimum. Below we provide an algorithm for the decomposition and the scheduling of TSs.

The G_{min}-Maintaining Decomposition algorithm.

$n \leftarrow n_i$ /* $1 \leq i \leq C$ */

$t \leftarrow 1$

while $n > 0$ **do**

  Choose the largest element $f_{i,k}$ **from** F_i **such that** $f_{i,k} < n$.

  $f(t) \leftarrow f_{i,k}$

  $Q(t) \leftarrow \lfloor n/f(t) \rfloor$

  $n \leftarrow n - Q(t) \times f(t)$

  $t \leftarrow t + 1$

end while

Based on the algorithm, we can decompose n_i as below.

$$n_i = Q_i(1) \times f_i(1) + Q_i(2) \times f_i(2) + ... = \sum_t Q_i(t) \times f_i(t). \tag{4.16}$$

Here $Q_i(t)$ represents the number of groups with $f_i(t)$ members and $f_i(t)$ is the largest element from F_i, such that $f_i(t) \leq (n_i - \sum_{k=0}^{t-1} Q_i(k) \times f_i(k))$ and $f_i(t) > f_i(t+1)$. We assume that the number of TSs of Class i after the grouping becomes R_i. Therefore, $R_i = \sum_t Q_{i,t} \leq n_i$ since $f_i(t) \geq 1$.

After the decomposition, the Class i TSs are divided into R_i groups. Then equal spacing scheduling are performed within each group. For groups with $f_i(t)$ members, the inter-group offset for the m^{th} TS to the leading TS is set to $(m-1) \times p/f_i(t), 1 < m \leq f_i(t)$. Clearly, the equal-spacing scheduling can still be utilized in this way to decrease complexity without altering the G_{min}. Finally, for C classes, we have $R = \sum_{i=1}^{C} R_i$ aggregated TSs to be scheduled. By the size of the revised periods for the these TSs, the sorted greedy scheduling is adopted to reduce computational load and to determine the inter-group offset. The final offset for each TS would be the sum of the inter-group offset and the intra-group offset.

Example 4.1. Assume that $K = 30$, $C = 3$, $n_1 = n_2 = n_3 = 10$, $p_1 = 40$, $p_2 = 60$, $p_3 = 150$. Based on naive equal spacing method, we have $p_1^{'} = 40/10 = 4$, $p_2^{'} = 60/10 = 6$, $p_3^{'} = 150/10 = 15$, and $p_1^{'}$ and $p_3^{'}$ are pairwise coprime since $G_{1,3} = 1$. This would lead to unsatisfactory result of system minimum distance.

As for GMD greedy method, we have $G_{min} = min\{20, 10, 30\} = 10$, and thus $p_1 = G_{min} \times f_1 = 10 \times\ 4$, $p_2 = G_{min} \times\ f_2 = 10\ \times 6$, and $p_3 = G_{min} \times f_3 = 10 \times\ 15$. We factorized f_i. The contents of F_i are given by $F_1 = \{4, 2, 1\} = \{f_{1,1}, f_{1,2}, f_{1,3}\}$, $F_2 = \{6, 3, 2, 1\} = \{f_{2,1}, f_{2,2}, f_{2,3}, f_{2,4}\}$, and $F_3 = \{15, 5, 3, 1\} = \{f_{3,1}, f_{3,2}, f_{3,3}, f_{3,4}\}$. Then n_i can be expressed as a linear combination of the elements of F_i. Therefore, we have $n_1 = 10 = 2 \times f_{1,1} + f_{1,2} = 2 \times\ f_1(1) + f_1(2)$, the resulted $R_1 = 3$, and the revised periods are $p_{1,1} = 40/4 = 10$ and $p_{1,2} = 40/2 = 20$, respectively; $n_2 = 10 = f_{2,1} + f_{2,2} + f_{2,4} = f_2(1) + f_2(2) + f_2(3)$, $R_2 = 3$, and $p_{2,1} = 60/6 = 10$, $p_{2,2} = 60/3 = 20$, and $p_{2,4} = 60/1 = 60$; $n_3 = 10 = 2 \times f_{3,2} = 2 \times f_3(1)$,

$R_3 = 2$, and $p_{3,2} = 150/5 = 30$.

As a result, the resulted $p_{i,k}$'s are not pairwise coprime and the value of G_{min} is reserved. The rearrangement for the $R = \sum_{i=1}^{3} R_i = 8$ aggregated TSs can then be done according to the ascending order of the revised period one by one.

4.3.2 The Simply Sorted Greedy Method

Here we propose another rearranging algorithm named Simply Sorted greedy method. In this algorithm, TSs are scheduled one by one according to ascending order of their original periods. Clearly, this algorithm also maintains G_{min}. However, its computational load is higher than that of the GMD greedy method since we do not perform the equal-spacing scheduling for each class.

4.3.3 Implementation Issues

Before really applying the determined rearrangement, the AP can constantly perform the rearranging algorithms in the background and compare the derived result with that of the current configurations. If the current configuration results in a much worse condition of the distances between existing scheduled events when compared with the background result, e.g., the minimum distance of the system is less than half of the background one, the application of the rearrangement should be triggered.

Since the periodic beacon is important for issues such as timing synchronization, it is better to leave its scheduled transmission times unchanged. Therefore, the target beacon transmission times can be a reference for the new schedules to take effect. Similar to what described in Section 3.4.1, the installation of the rearranging results, i.e., the revised schedules, can be easily done by the AP according to the repetition pattern of the established periodic schedules and the current time since the notion of sequence can be extended to both directions of the time line as shown in the system model. According to this idea, the

longest delay experienced by certain TS for the starting of the rearranged service is upper bounded by the value of its SI, rather than the period of the total repetition pattern of different periodic service schedules. Moreover, the longest possible interval for certain TS between the original schedule and the renewed one is upper bounded by $2 \times SI$. Although it could violate the delay constraint for some TSs when the new schedule takes effect, it is only transient condition.

There could be two possible ways to notify the change of schedule to STAs. In addition to the established S-APSD service, STAs should regularly listen to the beacons for administrating issues of the associated basic service set. Therefore, the renewed schedule can be attached in the beacons and broadcast to the STAs. However, if some STAs intentionally lengthen the listening interval of the beacons, the application of the new schedule cannot be read immediately. Moreo-ver, the STAs using this method should be able to understand the new arrangement and update its wake-up time according to the current time. An alternative method is to individually notify the STAs during the unicast communications. A more energy efficient approach is to inform the STA before the new schedule taking effect. Therefore, given the target schedule renewal time, the AP should trace the *last* scheduled instants of every STA before the application of the new arrangement. Denote the time to apply the new arrangement as T_{new} which corresponds to the l^{th} scheduled instants of the reference $\overline{\boldsymbol{X}}_r$ with period p_r and offset O_r with respect to $\overline{\boldsymbol{X}}_1$ within the total repetition interval. Therefore, the time to notify the STA attaching the m^{th} TS with relative offset O_m to the reference can be found as follows. Assume that

$$O_m + s_m \times p_m < O_r + l \times p_r < O_m + (s_m + 1) \times p_m. \tag{4.17}$$

Then s_m can be obtained as

$$s_m = \left\lfloor \frac{O_r + l \times p_r - O_m}{p_m} \right\rfloor. \tag{4.18}$$

Therefore, the time to notify the i^{th} TS is $T_{new} - (O_r + j \times p_r - O_m - s_m \times p_m)$. Then STAs can sleep till the revised time which could be either earlier or later than the original wake-up

time, depending on the repetition pattern of the new schedule and the current time. The sleeping interval for the transition is bounded between 0 and $2 \times SI$. After that, it returns to SI.

4.4 Performance Evaluations

The considered scenario is composed of K scheduled TSs which belong to 5 classes. The 5 classes of traffic, listed in Table 4.2, in the system are real-time voice, real time video, streaming audio, streaming video, and real-time gaming, and they are referred to Class 1 to 5 in the following discussion. The traffic characteristics are obtained from [36], [38], and [41]. The video traces are available online [49]. It is assumed that there are K STAs, each is configured with a scheduled TS belonging to one of the 5 classes. The number of STAs is increased in multiples of 5 STAs to keep the ratio of existing traffic classes. The system parameters are listed in Table 4.3 and conform to the IEEE 802.11a OFDM wireless LANs. The calculation for frame transmission time can be found in [14]. The precision used in our simulations is set to $100\mu s$. We consider the scenario of increasing the number of TSs in each class, n_i, from 1 to 10.

4.4.1 Comparison of Computational Complexity

In this numerical evaluation, we increase the number of existing TS in each class of application, and check the average online complexity of LCS-APSD and those of the proposed GMD greedy method and the Simply Sorted greedy method. Given the number of existing TSs, the average complexity for finding the SST is derived by averaging the number of necessary online operations when a new TS belonging to any class of application joins.

For LCS-APSD and K existing TSs belonging to C classes, we add a new TS as the $(K+1)^{th}$ TS of Class j. It is assumed that the contents of $A_{i,j}$ have had been calculated and stored in advance. To construct the scheduling matrix, we need $K \times GL_{K+1}$ comparisons to

obtain $d(\boldsymbol{X}, \boldsymbol{Y}+k), 0 \leq k \leq GL_{K+1}-1$. Then it takes GL_{K+1} comparisons to determine k^*. As for the suggested implementation method in Section 3.3.1 for the Class-based system, i.e., given that the contents of $B_{i,j}$ have had been renewed to the up to date situation, the complexity is reduced to $C \times GL_{K+1}$ comparisons after the scheduling matrix is constructed. Finally, GL_{K+1} comparisons to determine k^* is necessary.

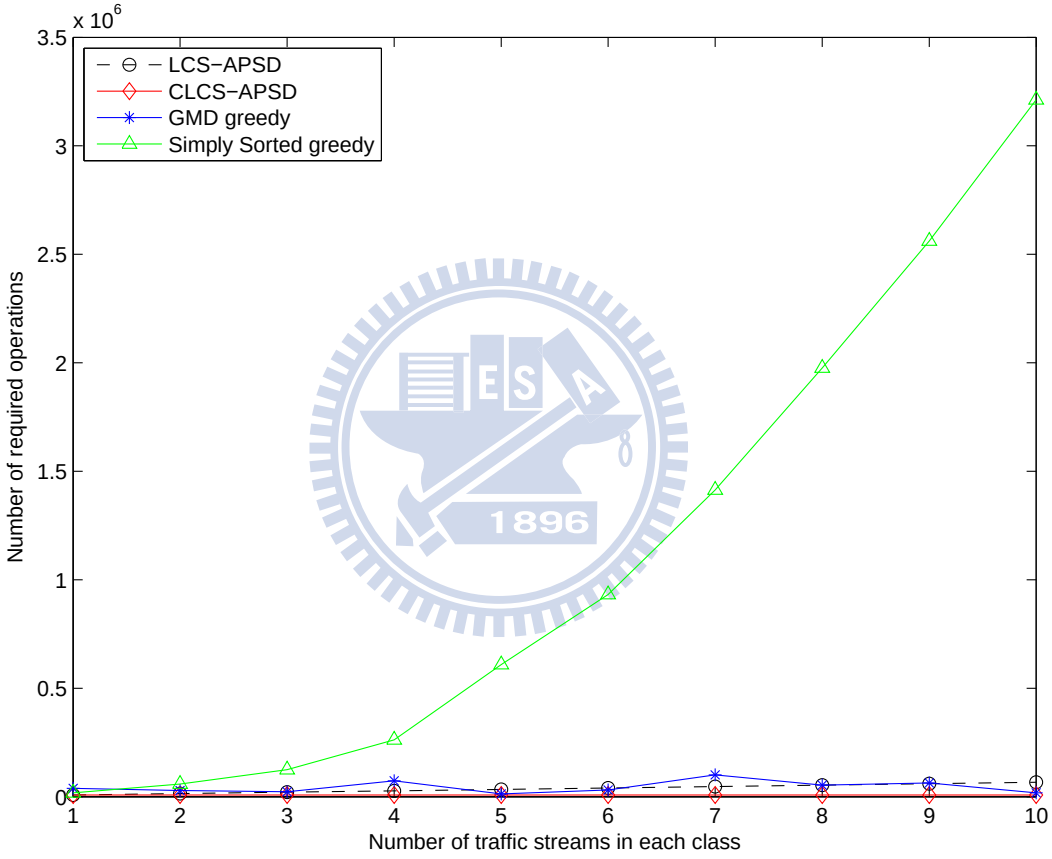

Figure 4.4: Comparison of implementation complexity.

Regarding the GMD greedy method, in addition to constructing necessary scheduling matrices to determine the offsets, firstly it needs to divide TSs into groups and then performs equal spacing within each group. For Class i, n_i is represented by

$$n_i = Q_i(1) \times f_i(1) + Q_i(2) \times f_i(2) + ... = \sum_t Q_i(t) \times f_i(t).$$

We assume that the number of TSs of Class i after the grouping becomes R_i. Therefore, $R_i = \sum_t Q_i(t)$. In summary, we need R_i subtractions to obtain these R_i groups. To determine

$f_i(t)$ from F_i, the number of comparisons is upper bounded by $Q_i(t) \times u_i$. Therefore, to find out $f_i(t)$'s from F_i for all t, it is bounded by $\sum_t Q_i(t) \times \ u_i$ operations. The overall complexities for grouping C classes are $\sum_{i=1}^{C} R_i$ subtractions and $\sum_{i=1}^{C} R_i \times u_i$ comparisons. When the grouping is done, there are $R = \sum_{i=1}^{C} R_i \leq (K+1)$ period-revised TSs belonging to C' classes, where C' is the number of distinct revised periods. Then we sort the TSs by the size of the revised periods. If the Merge Sort [40] is used, a sorting of the elements requires $(R \log_2 R)$ comparisons. Subsequently, we add the period-revised TSs one after another by gradually constructing the scheduling matrices. From Theorem 4.2, we can easily determine the relative distances by $G'_{i,j}$ operations, where $G'_{i,j}, 1 \leq i, j \leq C'$, is the GCD of two revised periods. The complexity in preparing all necessary relative distances is thus $\sum_{j=2}^{C'} \sum_{i=1}^{j-1} G'_{i,j}$ operations. The complexity for R period-revised TSs to generate all necessary scheduling matrices to obtain the impacts from scheduled TSs is $\sum_{m=2}^{R}(m-1) \times GL'_m$ comparisons where we have the period-revised TS m belongs to Class i and $GL'_m = lcm\{G'_{1,i}, G'_{2,i}, ..., G'_{j,i}, ...\}$. To determine the offset for each TS, it needs $\sum_{m=2}^{R} GL'_m$ comparisons.

For the Simply Sorted greedy method, because it does not divide any TS into the groups, there are $K+1$ TSs to be scheduled. Firstly, it needs to sort the TSs according to the length of SIs and requires $(K+1)\log_2(K+1)$ comparisons if the Merge Sort is adopted. The complexity for these $K+1$ TSs to generate all necessary scheduling matrices to obtain the impacts from scheduled TSs requires $\sum_{m=2}^{K+1}(m-1) \times GL_m$ comparisons. To determine the offset for each TS, it needs $\sum_{m=2}^{K+1} GL_m$ comparisons.

The comparisons of computational complexity are shown in Fig.4.4. As can be seen, the complexity of the GMD greedy method is close to LCS-APSD. The complexity varies because that the decomposition does not necessarily generate more period-revised TSs with the increasing number of TSs . Moreover, due to the fact that TSs are divided into a few groups in the GMD greedy method, the complexity in constructing scheduling matrices and comparisons for determining the offsets is not that much. For the Simply Sorted greedy method, the TSs are not divided into groups. Therefore, the number and the size of scheduling matrices

would tend to grow with the increasing number of TSs to be scheduled.

4.4.2 Comparison of the System Minimum Distance

Here we check the obtained minimum distances of the system for the scheduling algorithms by increasing the number of existing TSs in each class. For the LCS-APSD, the scheduling result depends on the joining order of TSs. If there are K TSs belonging to C classes, there are $(K!/\prod_{i=1}^{C} n_i!)$ possible combinations of the joining order, where the operator "!" stands for factorial. For simplicity, we only consider several cases for the LCS-APSD as specified below:

1. The joining order is according to Class $[i, i+1, i+2, i+3, i+4]$ (mod 5) for $1 \leq i \leq 5$, and repeats till K is reached.
2. The joining order is according to Class $[i, i-1, i-2, i-3, i-4]$ (mod 5) for $1 \leq i \leq 5$, and repeats till K is reached.
3. The joining order is by the ascending order of periods
4. The joining order is by the descending order of periods

In Fig. 4.5, we record the worst, average, and best performance for the LCS-APSD scheme for the studied cases. As shown in Theorem 4.1, without producing the coprime situation, the system minimum distance is bounded by $G_{min}/2 = 50$ for all algorithms. For the studied cases, the performance of the GMD greedy algorithm is better than the LCS-APSD on average especially when the number of TSs is large. This may because that the TSs forming aggregated TS had been distributed uniformly away from each other before the inter-group scheduling. Also, we schedule the aggregated TSs based on their levels of scheduling difficulty. As for the Simply Sorted greedy algorithm, it also performs well since the TSs are scheduled in ascending order according to the size of the period. Due to the fact that we do not divide any TS into the groups, the periods of the TSs are larger than using

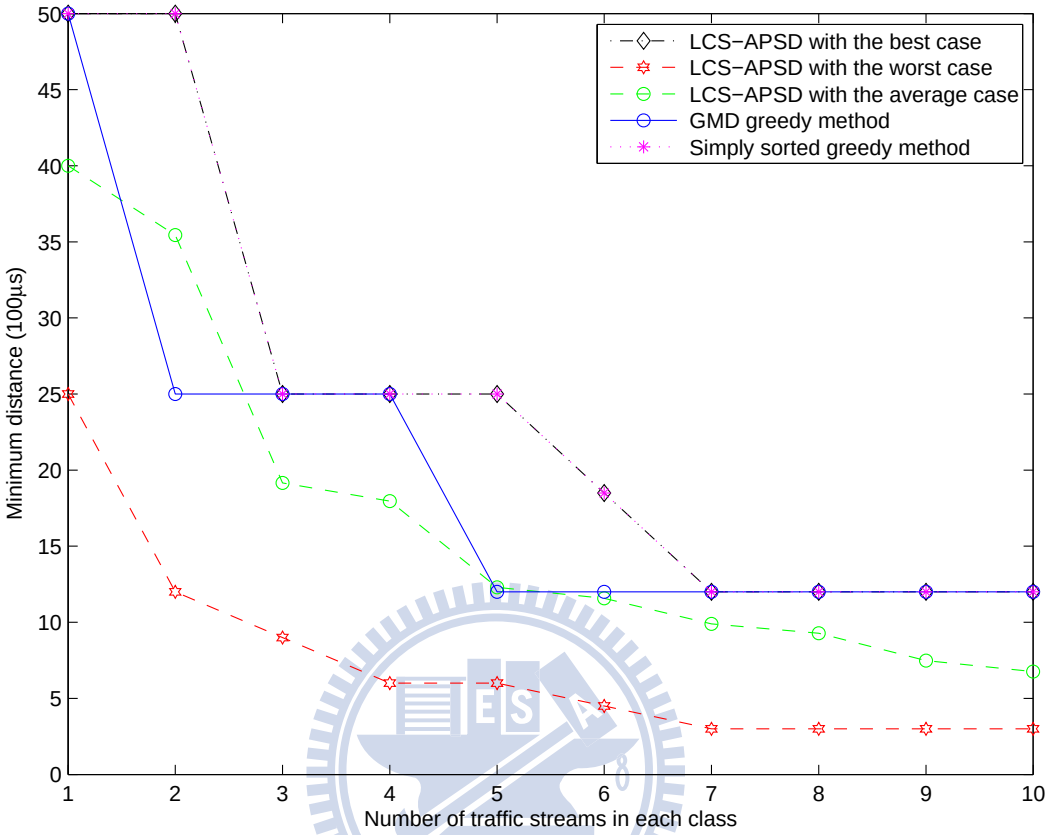

Figure 4.5: Comparison of the obtained maximum of minimum distance.

the GMD method and more positions can be chosen to maximize the minimum distance. Thus intuitively it has better chance to find the offsets to maximize the minimum distance of the system. However, the Simply Sorted greedy algorithm requires much higher complexity than that of the other algorithms as shown in previous evaluation.

4.4.3 Comparison of Power Consumptions

In this evaluation, we fix the number of existing TSs at 50 (10 for each class) and use the LCS-APSD and OAS-APSD algorithms to schedule those TSs by the orders specified in pre-vious sub-section. The simulation is performed to model 300 seconds of the real time. The power consumptions of Active state and Doze state are listed in Table 4.3. According to the usage of ap-plication and the distribution of scheduled instants resulted by individual algorithms, the average consumed power per TS is measured as the normalized

energy consumption. The derived power consumptions depend on the specified joining order and differ much for the LCS-APSD and OAS-APSD algorithm. As shown in Fig. 4.6, the proposed GMD greedy algorithm and the Simply Sorted greedy algorithm outperform the LCS-APSD algorithm with the specified cases. The largest improvement obtained is 6% for the considered scenarios. We find that although both the GMD greedy algorithm and the Simply Sorted greedy algorithm achieve the same system mini-mum distance, the obtained average power consumptions are not equal. This may be because that the determined offsets for the GMD greedy algorithm, due to the equal spacing grouping, are more uniform when compared with other algorithms. Also, the overlapping of SPs may depend on the durations of SPs. From the simulation results, it is also shown that the average consumed power for each STA is much smaller than that of the Active state which spends 1.4W. Therefore, it is worth to utilize the power saving scheduling even during active sessions.

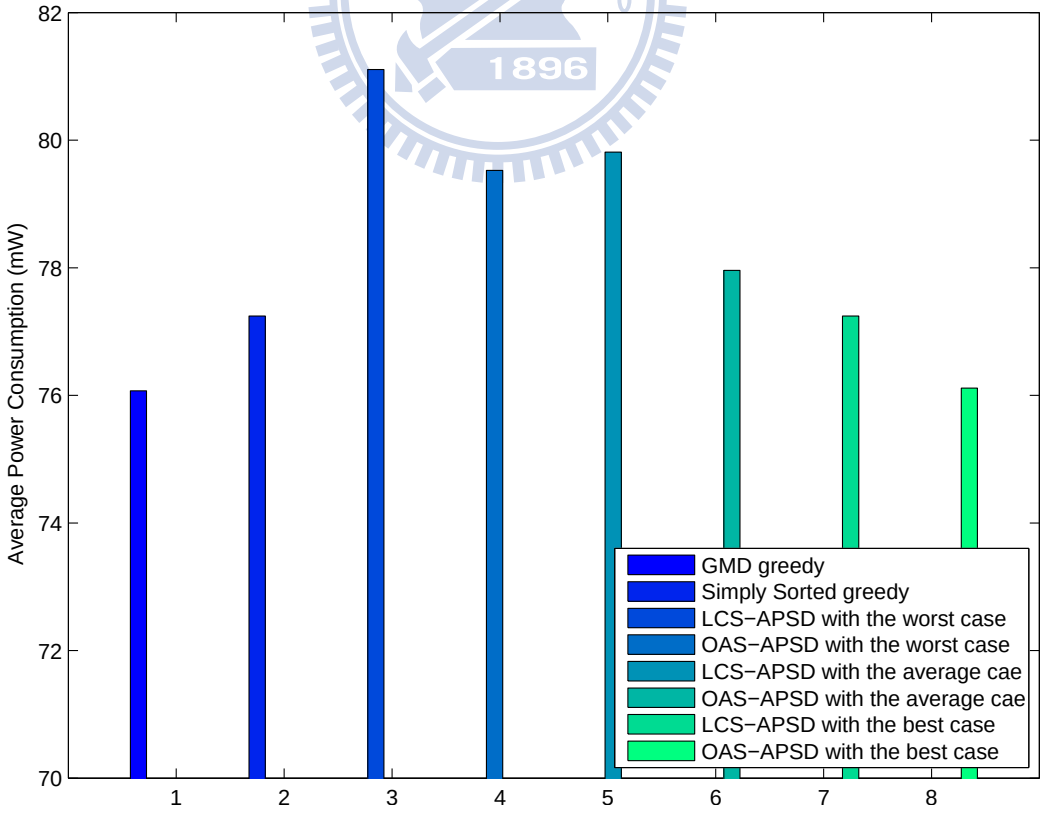

Figure 4.6: Comparison of the Average Consumed Power.

4.5 Summary

In this chapter, our contributions are summarized as follows. We first present the brute-force searching method to obtain the global optimum of rearranging scheduled instants. It is, however, infeasible because of the huge complexity when the number of TSs is large. To reduce complexity, we develop several solutions to fulfill the goal of rearrangement. We show that the maximum of minimum distance between two periodic scheduled events is closely related to the greatest common divisor of their periods. Taking advantage of this observation, we devise the GMD greedy scheduling to increase the tractability while improving the energy saving performance. From simulation results, both the proposed GMD greedy method and the Simply Sorted greedy method are able to obtain larger system minimum distance than that of the previous greedy-based scheduling method generally. Consequently, better power saving performance is attained as well. An interesting and challenging further research topic is to determine the optimal schedule when the durations of SPs are taken into account.

Table 4.1: Notations used in Chapter 4

Notations	Descriptions
C	The number of classes of applications.
K	The number of existing TSs in the system.
i, j	The indices to represent the classes, $1 \leq i, j \leq C$.
m, n	The indices to represent the TSs, $1 \leq m, n \leq K$.
$\boldsymbol{X}$	A periodic sequence $\boldsymbol{X} = \{x_r\}_{r=-\infty}^{\infty} = \{p \times r\}_{r=-\infty}^{\infty}$.
$\boldsymbol{X}_m$	A periodic sequence with period p_m. $\boldsymbol{X}_m = \{x_{m,r}\}_{r=-\infty}^{\infty} = \{p_m \times r\}_{r=-\infty}^{\infty}$.
$\overline{\boldsymbol{X}}_{\boldsymbol{m}}$	$\boldsymbol{X}_m = \{\overline{x}_{m,r}\}_{r=-\infty}^{\infty} = \{p_m \times r + O_m\}_{r=-\infty}^{\infty}$ represents the scheduled instants of the m^{th} TS.
$\overline{\boldsymbol{X}}$	The aggregation of existing scheduled events: $\overline{\boldsymbol{X}} = \bigcup_{1 \leq m \leq K} \overline{\boldsymbol{X}}_m$.
$\boldsymbol{X}_i$	A Class i sequence with period p_i and $\boldsymbol{X}_i = \{x_{i,r}\}_{r=-\infty}^{\infty} = \{p_i \times r\}_{r=-\infty}^{\infty}$.
$\boldsymbol{Y}$	A periodic sequence to represent the new joining TS belonging to Class j: $\boldsymbol{Y} = \{y_l\}_{l=-\infty}^{\infty} = \{p_j \times l\}_{l=-\infty}^{\infty} = \{q \times l\}_{l=-\infty}^{\infty}$. $q = p_j$.
O_m	The offset of the m^{th} TS. $O_1 = 0$.
$G_{i,j}$	The greatest common divisor of of p_i and p_j, i.e., $G_{i,j} = \gcd(p_i, p_j)$.
GL_m	If TS m belongs to Class i, $GL_m = lcm\{G_{1,i}, G_{2,i}, ..., G_{C,i}\}$ is the least common multiple of the period of p_i and those of other existing classes.
$d(\boldsymbol{X}, \boldsymbol{Y} + k)$	The distance between the sequences $\boldsymbol{X}$ and $\boldsymbol{Y} + k$.
$A_{i,j}$	Store [$d(\boldsymbol{X}_i, \boldsymbol{X}_j + 0)$ $d(\boldsymbol{X}_i, \boldsymbol{X}_j + 1)$ ... $d(\boldsymbol{X}_i, \boldsymbol{X}_j + G_{i,j} - 1)$] which represents the impact of a Class i sequence to a sequence of Class j with offset k.

Table 4.2: Traffic Characteristics.

Class	Application	Mean Data Rate	Service Interval
1	Real-time Voice (G.711)	64 Kbps	40 ms
2	Real-time Video (MPEG 4 Trace: Lecture Room Cam)	58 Kbps	60 ms
3	Streaming Audio	128 Kbps	150 ms
4	Streaming Video (MPEG 4 Trace: First Contact)	330 Kbps	300 ms
5	Real-time Gaming	20 Kbps	100 ms

Table 4.3: System parameters

Parameter	**Value**
PHY Data Rate	54 Mbps
PHY Control Rate	6 Mbps
Transmission time for PHY header and preambles	20 μs
Transmission time for an OFDM symbol	4 μs
SIFS	16 μs
Slot Time	9 μs
MAC frame header	30 bytes
ACK frame	14 bytes
IP header	20 bytes
UDP header	8 bytes
RTP header	12 bytes
Power Consumption in Awake state	1.4 W
Power Consumption in Doze state	0.045 W
Hardware delay of switchover	250 μs

Chapter 5

Energy-Efficient Multi-Polling Mechanism

In previous chapters, we have considered scheduling algorithms which try to separate the service periods of downlink applications with different service intervals with the intention to minimize the chance of overhearing situations. When the *uplink* from STAs are also taken into account and it is known by the AP that there are multiple STAs ready to transmit their frames, the centralized polling method is often adopted to reduce contention and collisions for these STAs. Since the constantly (single) polling frames could be a source of overhead, multi-polling schemes were proposed to replace their necessities. However, previous multi-polling schemes do not study the energy conservation issues and the energy consuming overhearing problem exists due to the required operations. This motivates our works for the subsequent chapters to alleviate the overhearing problem of previous multi-polling schemes and make a nice combination of both energy efficiency and low-overhead MAC.

In this chapter, we propose an MAC scheme called EE-Multipoll (Energy-Efficient Multi-polling) which generally retains the merits of high bandwidth utilization (BU) as the ordered-contention multi-polling scheme while effectively lowering energy consumption. Given traffic characteristics, a *wake-up time schedule* (WTS) is derived to statistically guarantee the BU. Since the overhearing problem is largely mitigated, the energy conservation for an STA of later access order is effectively improved. It is good for energy saving for STAs not involved in

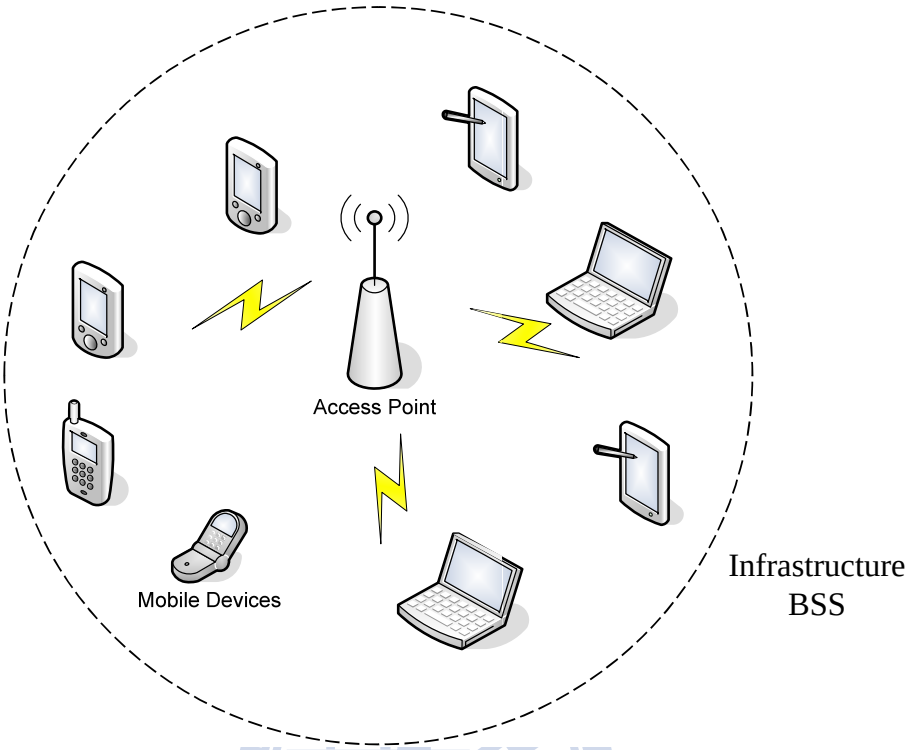

Figure 5.1: The considered network scenario of the EE-Multipoll mechanism.

the schedule as well since they can check the announcement at the beginning of the scheduled period when compared with the legacy polling scheme. Through analysis and simulation, when compared with previous ordered-contention multi-polling schemes, the proposed EE-Multipoll mechanism achieves significant energy saving with only slight degradation in BU.

The remainder of this chapter is organized as follows. Following the system model, Section 5.2 develops the framework and details of our proposed energy-efficient multi-polling protocol. In Section 5.3, we formally define the problem of optimal WTS subject to a pre-defined loss of BU and present a near-optimal feasible solution. Analyses of the wake-up times and energy saving are also presented in this section. Performance evaluation is provided in Section 5.5. Finally, we draw summary in Section 5.5.

5.1 System Model

An infrastructure *basic service set* (BSS) composed of m STAs which can hear each other within the BSS and an AP taking the responsibility of scheduling and having no energy concern as shown in Fig. 5.1 is considered. Assume that there are only a finite number of different applications and the traffic characteristics of all applications are known. Traffic arrival processes are assumed to be stationary, i.e., the probability distribution does not change when shifted in time. Note that the density function of required transmission time for traffic arrivals can be estimated offline via nonparametric density estimation methods such as histograms or the kernel estimator [34], or parametric methods to use a parametric probability density function (pdf) as an approximation. As a result, the AP can have pre-calculated traffic arrival distribution for each STA as long as it knows the applications that are admitted.

5.2 Energy-Efficient Multi-Polling Mechanism

5.2.1 Mechanism Design

AP Operation

AP can learn the type and the number of traffic flows every STA intends to transmit/receive during the *contention-free periods* (CFP) after the join/leaving handshaking. It manages the polling list, computes the appropriate WTS from the built-in database for each STA, and periodically announces EE-Multipoll frame after beacon or at scheduled time in every scheduled *service interval* (SI). The SI represents the interval between two successive multi-poll frames. To meet the QoS requirement of real-time traffic streams, the SI can be chosen such that every packet is served before its deadline. The guideline for SI selection can be found in [2] and [35].

The suggested frame format of EE-Multipoll is shown in Fig. 5.2. Each STA in the

Octets: 2	2	6	1	6 × RecordCount				4
Frame Control	Duration /ID	BSSID	Record Count (0-255)	Poll Record (6 octets)				FCS
				AID (2 octets)	Backoff (1 octets)	Wake-up time (2 octets)	TXOP Limit (1 octets)	

Figure 5.2: Suggested frame format of EE-Multipoll.

polling list has a corresponding poll record for its possible uplink traffic and another record for its downlink traffic, if AP has data buffered for the STA. The number of poll records is indicated in the Record Count field. The Poll Record field contains the information of the *association identifier* (AID) in the BSS, the assigned backoff value, the wake-up time (in units of slot time) relative to the receiving time of this EE-Multipoll frame, and the maximal usable duration of an aggregate TXOP for a specified STA (in units of 32μs [2]). The backoff filed is filled with the all-1s value for downlink usage. As for the uplink phase, the backoff value assignment should make sure that no two STAs have the same backoff value at any time to avoid collisions. To reduce overhead, the backoff value (in units of slot time) bt_j of STA with access order j in the uplink phase follows the rule: $bt_1 = 0$ and $bt_j = bt_{j-1} + 1$ for $j \geq 2$.

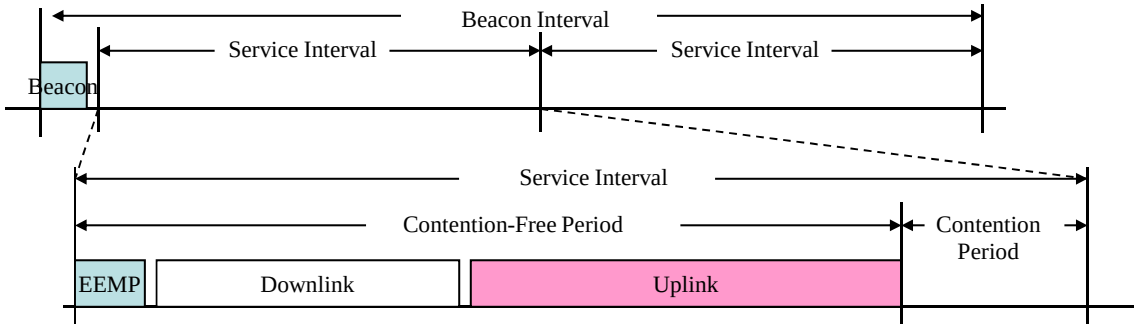

Figure 5.3: The framework of the proposed EE-Multipoll mechanism.

Fig. 5.3 depicts the framework of the proposed mechanism. Multipoll-capable STAs should remain awake to check the multipoll frame and update their NAVs. The AP should first serve the downlink traffic which can be exactly scheduled. Then it replies the uplink traffic with ACKs during the uplink phase. Since a polled STA can perform physical carrier

sensing to determine the channel condition after it wakes up and collision can be avoided by the initial backoff assignment, the backoff value should be reduced more effectively for the busy case if the order of the ongoing transmitting STA can be learned. Hence, when AP replies ACKs to STAs, it should contain the order of the ongoing transmitting STA in the QoS Control field of the MAC header. After CFP, the remaining time of an SI will become contention period.

Scheduled STA Operation

Scheduled STAs should periodically awake to check the EE-Multipoll frame and achieve synchronization by listening to the beacon frames. They should keep awake to listen to the notification of EE-Multipoll frame and cannot fall back to sleep until any explicit information about their new wake-up time are successfully decoded. The announced information such as SI, backoff value, wake-up time and TXOP limit should be obeyed. To save energy, STAs not listed in the EE-Multipoll frame should enter the Doze state.

For an STA with downlink traffic, it should awake at the notified wake-up time to receive the buffered data and then back to sleep after its TXOP. As for the STA with uplink traffic, the assigned backoff value implies its access order during the uplink phase. When an STA wakes up at the assigned wake-up time, its backoff value count down process begins after the channel is sensed idle for an SIFS period. When the channel is busy, it sets the NAV and checks the access order of the ongoing STA from the overheard information. The obtained ongoing STA's order can be utilized to adjust the possessed backoff value according to the difference between the STA's access order and the obtained one. This adjustment keeps the possessed backoff values of awake STAs being unique and reduces the overhead. In case an STA cannot overhear a whole frame to identify the order of the ongoing STA or the medium is idle when the STA wakes up, it just keeps the original backoff value. Finally, when backoff value reaches zero, the STA transmits an uplink data frame to initiate its TXOP. When the frame exchanges are finished, the STAs can enter the Doze state till the transmission time of

the next multipoll frame. Note that it is possible that there is no uplink frame for a polled STA in some polling round and it will remain in the Doze state till the next multipoll frame.

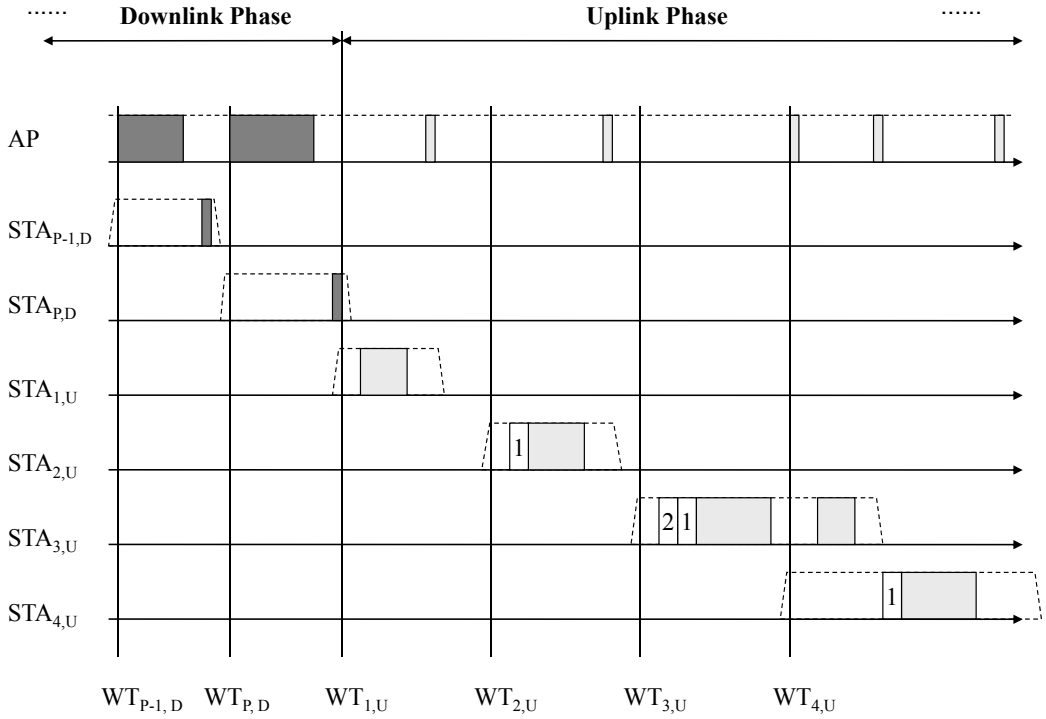

Figure 5.4: Illustration of EEMP operations.

Error Recovery Issue

In a real environment, there could be transmission error and thus error recovery should be taken into account. For our scheme, the EE-Multipoll frame should be delivered at base rate to reduce error probability. Moreover, the *first STA* listed in the multipoll frame should take the responsibility of confirming the successful delivery of the multipoll frame. It should reply an ACK frame (or piggyback the confirmation on the uplink frame) when it receives the multipoll frame. In case the multipoll frame contains error and the first STA does not response as expected, the AP should retransmit the multipoll frame. To avoid failure of the first STA, the AP can change the order to let each STA take turn to be the first one. As for data frames, the immediate retransmission policy, i.e., retransmitting a data frame which is not correctly acknowledged after SIFS, can be adopted for error recovery.

5.3 Energy-Efficient Wake-up Time Schedule

Since the number of buffered downlink frames is completely known by AP, it can calculate the required time to derive an exact WTS. On the other hand, the latest buffer statuses of uplink traffic before polling are generally unknown to the AP. Therefore, it is challenging to estimate WTS for the uplink flows due to dynamic characteristics of VBR traffic.

In this section, we present a calculation of WTS for the proposed EE-Multipoll mechanism. Here we only address the uplink phase since the downlink WTS can be easily computed by the AP. In Section 5.3.1, we first formally define the problem of optimal WTS subject to a predetermined BU loss and then provide a near-optimal feasible solution. The performance analysis is then provided in Section 5.3.2. Finally, we provide the analysis about the impact of discrepancy between the real density function and the estimated one on BU performance in Section 5.3.3.

5.3.1 An Energy-Efficient Scheduling Model

Assume that there are n STAs with uplink traffic and the required transmission time for traffic arrivals to the STAs in one SI are independent, identically distributed (*i.i.d.*). The derivation is not limited to traffic of identical distributions while *i.i.d.* assumption yields more concise results.

Problem Formulation

Without knowledge of exact buffer statuses of STAs, the AP can only estimate the transmission time required by each STA. As a result, the system BU might have to be sacrificed if STAs are put to sleep for energy saving because it is possible that the designated wake-up time of an STA is larger than the actual total transmission time of STAs assigned to transmit before it in some SI. As such, we define the scheduling problem of EE-Multipoll as follows. The notations and their definitions used in this section are listed in Table 5.1. In the

following description, CP-Multipoll represents the ordered-contention multi-polling scheme.

Determine $\mathbf{WT_1 \leq WT_2 \leq ... \leq WT_n}$ ***to maximize energy saving subject to at most*** x***% degradation of bandwidth utilization compared with the CP-Multipoll scheme.***

Note that the original CP-Multipoll did not take PM into account. The access scheme, however, can function normally without modifications if the shortest job first policy is adopted. Therefore, to compare the performance on energy saving, we assume that SJF policy is applied and thus an STA keeps awake till the end of its transmission and then enters the Doze state. As a result, the CP-Multipoll corresponds to $WT_i = 0$ for all i, $1 \leq i \leq n$, and $x = 0$. It seems difficult to solve the above optimization problem. In the following, we present a feasible solution with an additional constraint of degrading exactly x% BU for the first i STAs for all i. Since the first STA does not have the overhearing problem, it has $WT_i = 0$ and can start to transmit after *SIFS* of the EE-Multipoll frame. The i^{th} STA is assigned backoff value of $(i-1)$ slots.

Consider first the BU of the CP-Multipoll scheme. To simplify analysis, we assume that all traffic arrivals in one SI are served in the following polling period. To compute the BU for the first i STAs, we assume that there are i scheduled STAs and the AP is considered as STA $(i+1)$ with an assigned backoff value i. The BU for the first i STAs, which is defined as the average time used for transmission by the first i STAs over the access start time of STA $(i+1)$, can be obtained as

$$\frac{i(1-p)\overline{L}}{t_{MP}(i) + i(1-p)\overline{L} + i \cdot Slot + (i(1-p)+1)SIFS} \ . \tag{5.1}$$

Let us now evaluate the BU of our proposed feasible solution. Since the required transmission time (with pdf $h(t) = p \cdot \delta(t) + (1-p)l(t)$) is independent of the transmission start time (with pdf $s_i(t)$), it is clear that $u_i(t) = p \cdot u_{i-1}(t) + (1-p)(l(t) \otimes s_i(t))$, where $\otimes$ represents the convolution operation and $u_0(t) = \delta(t)$. The BU for the first i STAs is also

defined by $\frac{i(1-p)\overline{L}}{t_{MP}(i)+\overline{S_{i+1}}}$ to agree with that of CP-Multipoll.

Solution for the Defined Problem

The process to calculate WT_i can be depicted in two steps.

Step 1. Find the target $\overline{S_i}$ satisfying the requirement of BU.

Step 2. Solve for WT_i making the $\overline{S_i}$ as we expected by their relationship.

The derivation, which is shown below, is done iteratively from $i = 1$ to $i = n$.

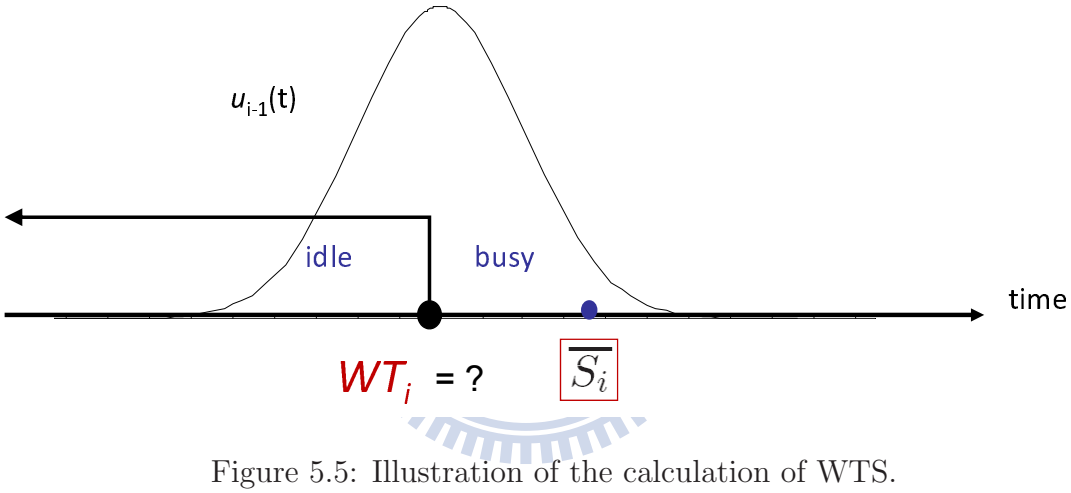

Figure 5.5: Illustration of the calculation of WTS.

Consider the first STA with $WT_1 = 0$. We have $s_1(t) = \delta(t - SIFS)$, $u_1(t) = p \cdot \delta(t) + (1 - p)l(t - SIFS)$, and $\overline{S_1} = SIFS$ is the same as that for the CP-Multipoll scheme. Now consider the second STA. The BU for the first STA is given by $\frac{(1-p)\overline{L}}{t_{MP}(1)+\overline{S_2}}$. Since the BU is allowed to degrade by $x\%$ as compared with the CP-Multipoll scheme, we have

$$\frac{t_{MP}(1) + (1-p)\overline{L} + Slot + (2-p)SIFS}{t_{MP}(1) + \overline{S_2}} = (100 - x)\%. \tag{5.2}$$

Therefore, the target average transmission start time for STA 2, i.e. $\overline{S_2}$, can be derived. The next step is to find a suitable WT_2 resulting in the target $\overline{S_2}$.

Since STA 2 wakes up at time WT_2, we consider the duration spent by STA 1 as WT_2 if its actual finish time is shorter than WT_2. Therefore, the average transmission start time

for STA 2 is given by

$$\begin{aligned}\overline{S_2} &= U_1(WT_2)(WT_2 + SIFS + Slot) \\ &\quad + \int_{WT_2}^{\infty} (t + SIFS + Slot)u_1(t)dt. \end{aligned} \tag{5.3}$$

We can solve for WT_2 in (5.3) by using the obtained $\overline{S_2}$ from (5.2).

With the obtained WT_2 and $u_1(t)$, the pdf of the transmission start time for STA 2 if it has data to transmit, $s_2(t)$, is then given by

$$s_2(t) = \begin{cases} 0, & \text{if } t < WT_2 + SIFS + Slot. \\ U_1(WT_2)\delta(t - (WT_2 + SIFS + Slot)), & \\ & \text{if } t = WT_2 + SIFS + Slot. \\ u_1(t - (SIFS + Slot)), & \\ & \text{if } t > WT_2 + SIFS + Slot. \end{cases} \tag{5.4}$$

Note that STA 2 spends $SIFS + Slot$ to check the channel status before its transmission. Therefore, we have $s_2(t) = 0$ for $t < WT_2 + SIFS + Slot$ because STA 2 cannot start its transmission before waking up and sensing the channel being idle for a duration of $SIFS + Slot$. For $t = WT_2 + SIFS + Slot$, we have $s_2(t) = U_1(WT_2)\delta(t - (WT_2 + SIFS + Slot))$ where $U_1(WT_2)$ represents the probability that the time used by STA 1 is less than or equal to WT_2. Finally, for $t > WT_2 + SIFS + Slot$, it holds that $s_2(t) = u_1(t - (SIFS + Slot))$ because STA 2 starts to transmit $SIFS + Slot$ after STA 1 finishes its transmission. After $s_2(t)$ and $\overline{S_2}$ are obtained, the pdf of average total duration for STAs 1 and 2 to finish their transmissions can be derived as $u_2(t) = p \cdot u_1(t) + (1 - p)(l(t) \otimes s_2(t))$. The acquired $u_2(t)$ can be used in the calculation of WT_3.

Consider now the derivation of WT_3. Since the BU for STAs 1 and 2 is given by $\frac{2(1-p)\overline{L}}{t_{MP}(2)+\overline{S_3}}$, the target $\overline{S_3}$ satisfies

$$\frac{t_{MP}(2) + 2(1 - p)\overline{L} + 2Slot + (3 - 2p)SIFS}{t_{MP}(2) + \overline{S_3}} = (100 - x)\%. \tag{5.5}$$

Similar to the derivation of WT_2, we can obtain the average transmission start time for STA

3 as

$$\begin{aligned}\overline{S_3} &= U_2(WT_3)(WT_3 + SIFS + 2Slot) \\ &\quad + \int_{WT_3}^{\infty} (t + SIFS + Slot)u_2(t)dt. \end{aligned} \tag{5.6}$$

Similarly, we can solve for WT_3 in (5.6) by using the obtained $\overline{S_3}$ from (5.5).

Likewise, after the WT_3 is derived, the $s_3(t)$ is given by

$$s_3(t) = \begin{cases} 0, & \text{if } t \leq WT_3 + SIFS + Slot. \\ u_2(t - (SIFS + Slot)), & \text{if } t \in (WT_3 + SIFS + Slot, \\ & \quad WT_3 + SIFS + 2Slot). \\ U_2(WT_3)\delta(t - (WT_3 + SIFS + 2Slot)), & \text{if } t = WT_3 + SIFS + 2Slot. \\ u_2(t - (SIFS + Slot)), & \text{if } t > WT_3 + SIFS + 2Slot. \end{cases} \tag{5.7}$$

For the first region, i.e., $t \leq WT_3 + SIFS + Slot$, we have $s_3(t) = 0$ because STA 3 cannot start to transmit before waking up and sensing the channel for $SIFS + Slot$. The second region, i.e., $WT_3 + SIFS + Slot < t < WT_3 + SIFS + 2Slot$, represents the situation that the total duration spent by STAs 1 and 2 is in between WT_3 and $WT_3 + Slot$. For the case that the medium is still busy after STA 3 wakes up, the sensing time for STA 3 actually depends on whether or not STA 2 has data to transmit. If it does not, then the ongoing one is STA 1 and STA 3 has to sense the channel for $SIFS + 2Slot$, the assigned backoff value plus the duration of one $SIFS$. On the other hand, if STA 2 has data to transmit, then STA 3 only needs to sense the channel for $SIFS + Slot$ before transmission. For simplicity, we assume that STA 2 has data to transmit in our analysis. For $t = WT_3 + SIFS + 2Slot$, we have $s_3(t) = U_2(WT_3)\delta(t - (WT_3 + SIFS + 2Slot))$ where $U_2(WT_3)$ denotes the probability that the total duration spent by STAs 1 and 2 is less than or equal to WT_3. Finally, for $t > WT_3 + SIFS + 2Slot$, we have $s_3(t) = u_2(t - (SIFS + Slot))$ because STA 3 starts its transmission $SIFS + Slot$ after STAs 1 and 2 finish their transmissions. Again, here we assume

that STA 2 has data to transmit to simplify the analysis. We make similar assumption in deriving $s_k(t)$ for $k \geq 4$. In other words, we assume that STA $(k-1)$ has data to transmit if the medium is busy when STA k wakes up. In a real system, the sensing time of STA k, if medium is busy when it wakes up, is equal to $SIFS + (k-j)Slot$ if STA j is the last one to transmit before STA k does.

In general, to derive WT_k, we should first compute the target average transmission start time $\overline{S_k}$ for STA k by solving

$$\frac{t_{MP}(k-1)+(k-1)(1-p)\overline{L}+(k-1)Slot+(k-(k-1)p)SIFS}{t_{MP}(k-1)+\overline{S_k}} = (100-x)\%. \tag{5.8}$$

The obtained value of $\overline{S_k}$ is then used to derive WT_k by solving

$$\begin{aligned} \overline{S_k} &= U_{k-1}(WT_k)(WT_k + SIFS + (k-1)Slot)) \\ &\quad + \int_{WT_k}^{\infty} (t + SIFS + Slot)u_{k-1}(t)dt. \end{aligned} \tag{5.9}$$

Note that $u_{k-1}(t)$ and the obtained WT_k are needed to compute $s_k(t)$ as

$$s_k(t) = \begin{cases} 0, & \text{if } t \leq WT_k + SIFS + Slot. \\ u_{k-1}(t-(SIFS+Slot)), & \\ \qquad \text{if } t \in (WT_k + SIFS + Slot, & \\ \qquad\quad WT_k + SIFS + (k-1)Slot). & \\ U_{k-1}(WT_k)\delta(t-(WT_k + SIFS + (k-1)Slot)), & \\ \qquad \text{if } t = WT_k + SIFS + (k-1)Slot. & \\ u_{k-1}(t-(SIFS+Slot)), & \\ \qquad \text{if } t > WT_k + SIFS + (k-1)Slot. & \end{cases} \tag{5.10}$$

The acquired $s_k(t)$ and $u_{k-1}(t)$ can be used to compute $u_k(t)$ as $u_k(t) = p \cdot u_{k-1}(t) + (1-p)(l(t) \otimes s_k(t))$ which is required in the calculation of WT_{k+1}.

Implementation Issues

The bisection method can be adopted to solve for WT_i. The initial values for the lower bound and the upper bound can be selected as WT_{i-1} and $\overline{S_i} - SIFS - Slot$, respectively.

In real implementation, the WT_i can only be represented by finite precisions. Therefore, the solving process of WT_i is stopped when both the precision has been met and the resulted degradation of BU is smaller than the constraint of our defined optimization problem.

Retransmission caused by channel error can be easily incorporated in the computation of WTS. Let A be the random variable for the transmission time of an STA without channel error. Assume that frame error probability is q. All we need to do is to compute the WTS with the modified random variable $B = (1+q)A$. Moreover, since the hardware delay should be taken into account in a real system, WT_i will be compared with the *hardware delay* D once it is derived:

$$WT_i = \begin{cases} WT_i, & \text{if } WT_i > D. \\ 0, & \text{if } WT_i \leq D. \end{cases} \tag{5.11}$$

5.3.2 Analysis of Energy Efficiency

Let us now evaluate the energy saved by our proposed EE-Multipoll scheme with the WTS for the uplink phase. Let P_A and P_D denote the power consumption in the Awake state and the Doze state, respectively. Here we consider the energy consumption during each SI.

The CP-Multipoll with SJF

For the CP-Multipoll scheme corresponding to $WT_i = 0$ for all i, $1 \leq i \leq n$, STA i will stay in the Awake state for an average duration of

$$A_{CPMP}(i) = \begin{cases} (1-p)(SIFS + \overline{L}), & \text{if } i = 1. \\ (1-p)\Big[\overline{L} + p^{i-1}(SIFS + (i-1)Slot) \\ \quad + \int_{0^+}^{\infty}(t + SIFS + Slot)u_{i-1}(t)dt\Big], & \text{if } i > 1. \end{cases} \tag{5.12}$$

For the case that $i > 1$, the term $p^{i-1}(SIFS+(i-1)Slot)$ represents the time spent on waiting if all STAs before STA i do not have data to transmit, while the term $\int_{0^+}^{\infty}(t + SIFS + Slot)u_i(t)dt$ denotes the waiting time (with some simplification on sensing time) for the situation that

at least one of the STAs has data to transmit. Therefore, the energy consumed by STA i, denoted by E_i, is given by $E_i = [t_{MP}(n) + A_{CPMP}(i)]P_A + [SI - t_{MP}(n) - A_{CPMP}(i)]P_D$. Consequently, the average total energy consumed by n STAs, denoted by E_{CPMP}, can be obtained as

$$\begin{aligned} E_{CPMP} &= \sum_{i=1}^{n} E_i \\ &= \Big[\sum_{i=1}^{n}(A_{CPMP}(i) + t_{MP}(n))\Big]P_A + \\ &\quad \Big[n \cdot SI - \sum_{i=1}^{n}(A_{CPMP}(i) + t_{MP}(n))\Big]P_D. \end{aligned} \tag{5.13}$$

The proposed EE-Multipoll with wake-up time schedule

Now consider the i^{th} STA in our proposed EE-Multipoll scheme. The average time staying in the Awake state equals

$$A_{EEMP}(i) = \begin{cases} (1-p)(SIFS + \overline{L}), & \text{if } i = 1. \\ (1-p)(\overline{L} + sensing_i + overhearing_i \\ \qquad + hw\_delay_i), & \text{if } i > 1. \end{cases} \tag{5.14}$$

The term $overhearing_i = \int_{WT_i^+}^{\infty}(t - WT_i)u_{i-1}(t)dt$ represents the average time for STA i to wait for STAs 1, 2, ..., and $(i-1)$ to finish their transmissions after it wakes up. The term $sensing_i = U_{i-1}(WT_i)[SIFS + (i-1)Slot] + \int_{WT_i^+}^{\infty}(SIFS + Slot)u_{i-1}(t)dt$ represents the average time STA i spent on sensing the channel. Additionally, the term $hw\_delay_i = D - (D - WT_i)^+$ stands for the required time of hardware delay for STA i which consumes roughly the same power as in the Awake state during this period [31]. The $(y)^+$ equals y if $y > 0$ or 0 if $y < 0$.

Let E_{EEMP} be the average total energy consumed by n STAs in one SI for our proposed

feasible solution. Similar to the analysis of CP-Multipoll, we have

$$\begin{aligned} E_{EEMP} &= \Big[\sum_{i=1}^{n}(A_{EEMP}(i)+t_{MP}(n))\Big]P_A + \\ &\quad \Big[n\cdot SI-\sum_{i=1}^{n}(A_{EEMP}(i)+t_{MP}(n))\Big]P_D. \end{aligned} \tag{5.15}$$

Finally, compared with the CP-Multipoll scheme with SJF, the percentage of energy saved by our proposed feasible solution is equal to $100(E_{CPMP}-E_{EEMP})/E_{CPMP}$.

5.3.3 Impact of Estimation Discrepancy

Since there could be estimation error for the pdf ($h(t)$ and the corresponding $u_{i-1}(t)$), the actual degradation of BU is likely to deviate from the desired value. The impact of estimation discrepancy is analyzed below. Assume that we have derived the wake-up time WT_i for STA i from the expected $x\%$ loss of BU, the expected transmission start time $\overline{S_i}$, and the approximated pdf $u_{i-1}(t)$. Let $\overline{S_{r,i}}$ denote the actual expected transmission start time for STA i given WT_i and the actual traffic with pdf $u_{r,i-1}(t)$. Also, let the actual loss of BU be $y\%$.

Denote the estimation error of pdf by $e_{i-1}(t) := u_{r,i-1}(t)-u_{i-1}(t)$. Then the discrepancy of expected transmission start time can be expressed as:

$$\begin{aligned} ER_i &:= \overline{S_{r,i}}-\overline{S_i} \\ &= \int_0^{WT_i}(WT_i+SIFS+(i-1)Slot)e_{i-1}(t)dt \\ &\quad +\int_{WT_i}^{\infty}(t+SIFS+Slot)e_{i-1}(t)dt. \end{aligned} \tag{5.16}$$

We can find the relationship between the expected BU loss and the actual one by ER_i.

$$\begin{aligned} y &= \frac{100(\overline{S_{r,i}} - S_{CPMP,i})}{t_{MP}(i-1) + \overline{S_{r,i}}} \\ &= \frac{100(ER_i + \overline{S_i} - S_{CPMP,i})}{t_{MP}(i-1) + ER_i + \overline{S_i}} \\ &= \frac{x(t_{MP}(i-1) + S_{CPMP,i}) + (100-x)ER_i}{(t_{MP}(i-1) + S_{CPMP,i}) + \frac{100-x}{100}ER_i}, \end{aligned} \tag{5.17}$$

where $S_{CPMP,i} = (i-1)(1-p)\overline{L} + (i-1)Slot + (i-(i-1)p)SIFS$. Clearly, if $x(t_{MP}(i-1) + S_{CPMP,i}) \gg ER_i$, then y is close to x. The condition tends to be true when i is large and/or every STA transmits a large number of frames.

5.4 Performance Evaluation

Three examples are studied for the proposed WTS strategy. System parameters are shown in Table 5.2. The PHY parameters which conform to the IEEE 802.11a standard are available in [4]. More available non-overlapping channels in the 5 GHz band provide better flexibility to avoid interference. The power consumption in either state is according to [22]. Since the time spent on sensing is simplified in analysis, we conduct computer simulations using Matlab [33] as a comparison to reflect the real situation. The TSMP scheme is also studied in our simulations. An STA can enter the Doze state in both SCP and DTP, if it is worthy of energy saving. The considered scenario is composed of n STAs, $1 \leq n \leq 20$, with *i.i.d.* traffic in an infrastructure BSS with an AP. We allow 5% degradation of BU when deriving the proposed WTS for the examples.

5.4.1 Example 5.1

Assume that the pdf $h(t) = p \cdot \delta(t) + (1-p)\delta(t-\overline{L})$, where $\overline{L}$ represents the time for exchanging a constant-length Data frame. The case of $p = 0.6$ is considered because it is a typical model for an on-off voice connection [18]. The frame exchange time, denoted by

$\overline{L}$, is chosen to be $200\mu s$ which is equivalent to the transmission time of a 200-byte VoIP frame. The mean and standard deviation of the traffic model are $\mu = 0.4\overline{L}$ and $\sigma = \sqrt{0.24\overline{L}}$, respectively. Fig. 5.6(a) shows the BU loss obtained from simulations for the first n STAs. As revealed in this figure, the analytical results are quite accurate since the maximum error is under 2.5%. The first 4 STAs have no loss of BU since the original wake-up times are smaller than the time of switchover and thus have wake-up time zero. The simulation result is slightly larger than the planned 5% for STAs after STA 5. The reason is that the sensing time of an STA is slightly underestimated for the busy channel case and thus resulting in more optimistic wake-up time estimation. Fig. 5.6(b) shows the energy-saving performance of the proposed WTS compared with the CP-Multipoll. The energy saved increases as the number of STAs increases since the overhearing problem is getting severer for more STAs. As for TSMP, there is no trade-off between energy saving and BU because the transmission time of each STA is exactly known. However, the SCP represents a significant overhead to affect BU, especially when each STA only demands short transmission time as shown in this example. Note that the loss of BU decreases as the number of STAs increases because the effect of the fixed fields in the multipoll frame diminishes. The energy saving performance shown in Fig. 5.6(b) decreases initially since STAs need to spend more energy in the SCP. However, it is gradually improved when there are more than 9 STAs. The reason is that it becomes worthy for some STAs to switch states to save energy during the SCP.

5.4.2 Example 5.2

In the second example, we assume that pdf $h(t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(t-\mu)^2}{2\sigma^2}}$ with $\mu = 1000\mu s$ and $\sigma = 200\mu s$ to model the aggregated traffic of multiple connections from an STA. Since the required transmission time cannot be negative, a new value is generated if a negative transmission time is obtained. We also study the impact of channel error in this example. The immediate retransmission scheme is adopted as the error recovery strategy. The frame error rate q is set to 0.1. Fig. 5.7(a) shows the BU degradation obtained with simulations.

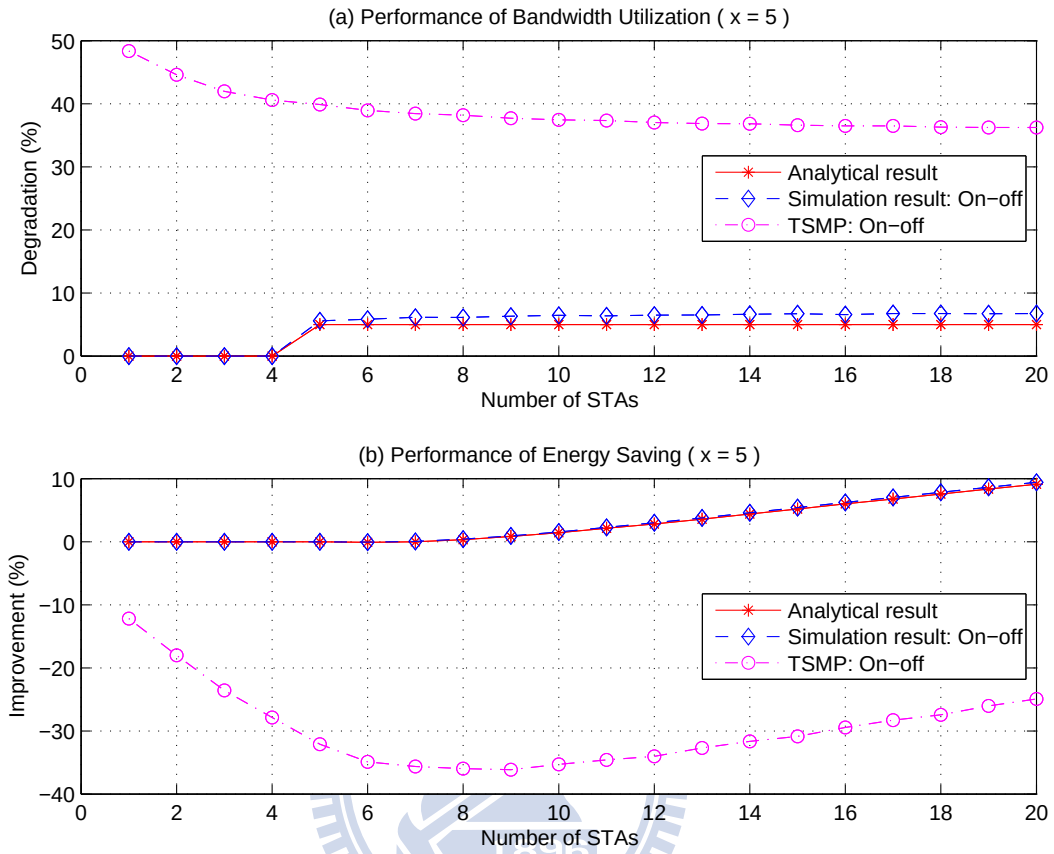

Figure 5.6: Performances of the derived wake-up time schedule for Example 1.

Note that the curves for analytical and simulation results cannot be distinguished clearly because they are almost identical. The situation of underestimated sensing time, as shown in Example 1, does not appear in this example. The reason is that the probability for an STA to have zero required transmission time is zero for continuous transmission time case. Fig. 5.7(b) illustrates the energy- saving performance of the proposed WTS, compared with the CP-Multipoll. The saved energy is about 80% for 20 STAs that is much higher than 10% obtained for Example 1. The reason is that the coefficient of variation (σ/μ) of the traffic model and the impact of overhead such as sensing and switchover time of this example are smaller than those for Example 1. The TSMP scheme performs much better in this example since the impacts of SCP on BU and energy saving are relatively smaller due to larger transmission time for each STA in this example. However, the proposed WTS with the EE-Multipoll scheme still outperforms TSMP for both comparisons due to less overhead.

In Fig. 5.7(b), the energy saving performance of TSMP declines by 6.1% from $q = 0$ to 0.1 because retransmissions delay the original transmission schedule and become a new source of overhearing which cannot be exactly predicted. For the EE-Multipoll mechanism with WTS considering retransmission time in advance, the improvement in energy saving is slightly increased since the overhearing problem of CP-Multipoll is aggravated in erroneous situation.

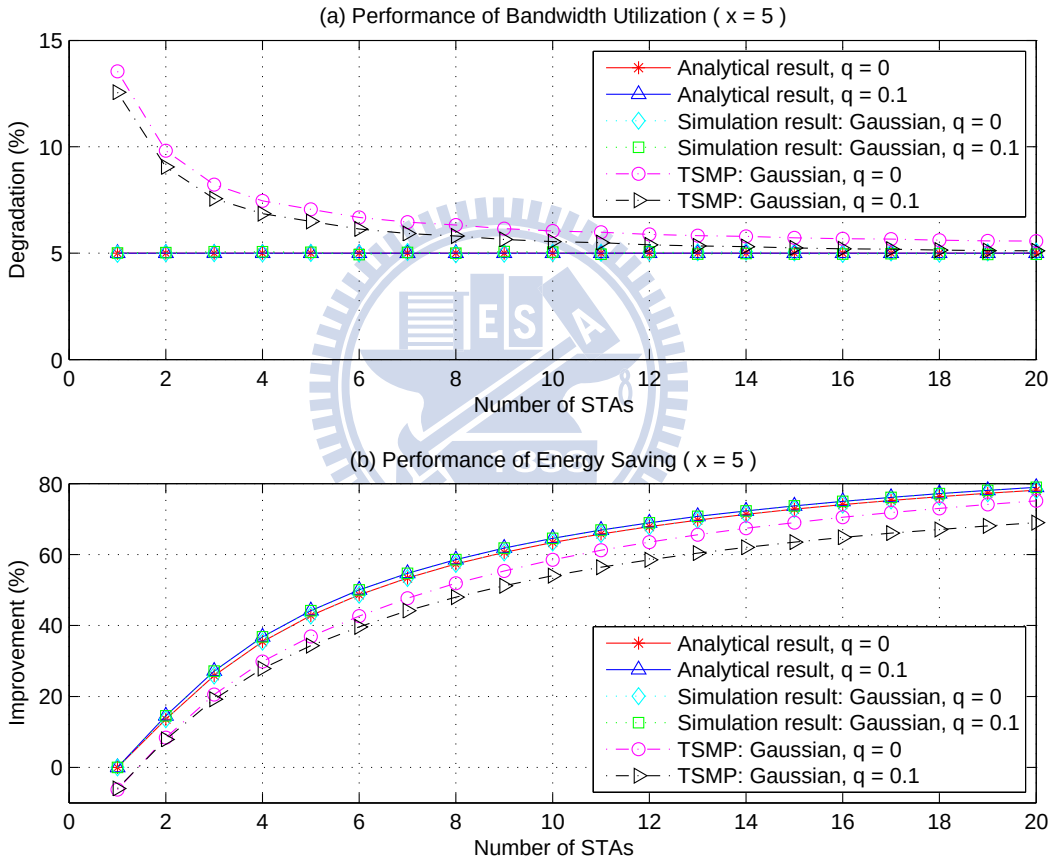

Figure 5.7: Performances of the derived wake-up time schedule with different frame error rates for Example 2.

We also investigated the performance of the proposed WTS for this traffic model with different standard deviations. Table 5.3 shows the wake-up times of STAs, the corresponding target average access start times, and the saved energy for a network consisting of 8 STAs. To meet the same constraint of BU loss, STAs have to wake up earlier if the standard deviation is larger. This is intuitively true since higher standard deviation results in more conservative

estimates. The saved energy decreases by 5.21% as the standard deviation increases from 100 to 300.

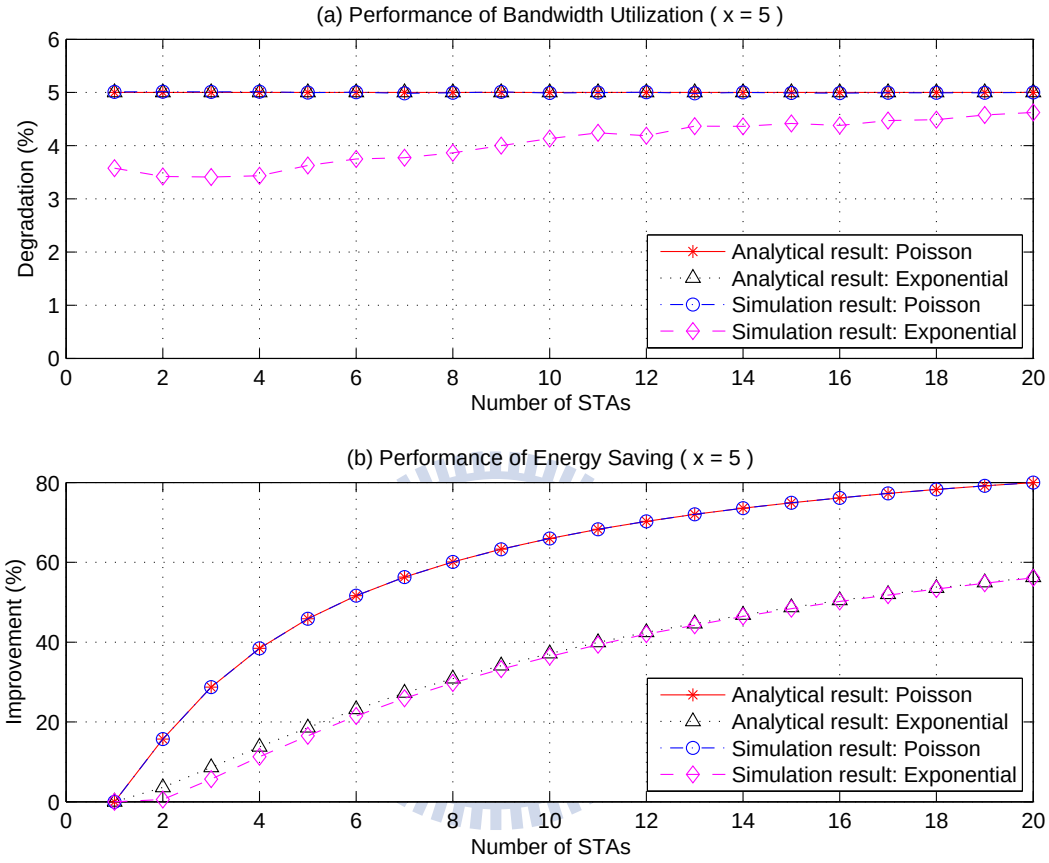

Figure 5.8: The deviation of using Normal distribution as traffic model.

5.4.3 Example 5.3

In this example, we study the performance of using Normal distribution as an approximation of the density function of true traffic arrivals. The traffic arrivals are assumed to follow the Poisson and the Exponential distributions with identical means selected to be $1000\mu s$. As shown in Fig. 5.8, using Normal distribution as an approximate traffic model yields satisfactory results. As the number of STAs increases, the losses of BU for both traffic models approach the desired value, i.e., 5%. It conforms with the analysis we provided in the impact of estimation discrepancy. Based on the observations, we suggest using Normal

distribution as traffic model to reduce the complexity when accurate estimation of pdf is difficult. For Normal distribution, only estimates of mean and variance are needed.

5.5 Summary

Current ordered-contention multi-polling schemes significantly improve BU by reducing control overhead. However, they suffer from useless energy consumption caused by overhearing which may largely decrease battery operating time. To solve the overhearing problem, we provide a PM method which aims to achieve maximum energy saving subject to a pre-defined degradation on BU. The derivations of WTS and saved energy are verified with computer simulations. According to numerical results, the saved energy is significant as compared with the original ordered-contention multi-polling schemes, especially when there are a large number of STAs. Moreover, when compared with the TSMP scheme, the simulation results reveal that the TDMA-like access method may not guarantee to bring better energy-saving performance since the prior handshaking can cause significant overhead to STAs. An interesting and challenging further research topic is to efficiently incorporate QoS guarantee into the proposed EEMP mechanism.

Table 5.1: Notations used in the analysis

Notations	Descriptions
WT_i	The wake-up time of STA i, $1 \leq i \leq n$, relative to the end of the multi-poll frame.
$l(t)$	The probability density function (pdf) of required transmission time conditioning on there are traffic arrivals.
$\overline{L}$	The mean transmission time for the traffic arrival to an STA in one SI conditioning on there is something to transmit ($\overline{L} = \int_0^\infty tl(t)dt$).
$h(t)$	The pdf of the required transmission time for traffic arrivals to an STA ($h(t) = p \cdot \delta(t) + (1-p)l(t)$, where p is the probability of no traffic arrival in one SI and $\delta(t)$ represents the Dirac delta function).
$s_i(t)$	The pdf of the transmission start time for STA i if it has data to transmit.
$\overline{S_i}$	The mean transmission start time for STA i, relative to the end of the multi-poll frame, if it has data to transmit ($\overline{S_i} = \int_0^\infty ts_i(t)dt$).
$u_i(t)$	The pdf of the total duration for STAs 1, 2, ..., and i to finish their transmissions.
$U_i(t)$	The cumulative distribution function (CDF) of $u_i(t)$ ($U_i(t) = \int_0^t u_i(\tau)d\tau$).
$t_{MP}(i)$	The time for receiving and verifying a multi-poll frame which contains i STAs.

Table 5.2: System parameters

Parameter	**Value**
PHY Data Rate	54 Mbps
PHY Control Rate	6 Mbps
Transmission time for PHY header and preambles	20 μs
Transmission time for an OFDM symbol	4 μs
SIFS	16 μs
Slot Time	9 μs
MAC frame header	30 bytes
ACK frame	14 bytes
IP header	20 bytes
UDP header	8 bytes
RTP header	12 bytes
Service Interval	25 ms
Beacon Interval	100 ms
Power Consumption in Awake state	1.4 W
Power Consumption in Doze state	0.045 W
Hardware delay of switchover	250 μs

Table 5.3: The WTS for different standard deviations, the target transmission start time, and the corresponding energy saving performances.

STA i	2	3	4	5	6	7	8
$\overline{S_i}$ (μs)	1099	2179	3258	4337	5417	6496	7576
WT_i (std $= 100$ μs)	1051	2112	3180	4245	5318	6388	7465
Energy saved for the first i STAs (%)	15.22	28.08	37.76	45.16	50.98	55.65	59.49
WT_i (std $= 200$ μs)	969	1998	3045	4100	5166	6225	7305
Energy saved for the first i STAs (%)	13.57	25.89	35.41	42.80	48.66	53.40	57.34
WT_i (std $= 300$ μs)	866	1851	2871	3900	4955	5981	7030
Energy saved for the first i STAs (%)	11.43	23.04	32.32	39.63	45.54	50.31	54.28

Chapter 6

Dealing with Energy-Saving Issue Induced from Error-Recovery

Continued from the idea of combining power saving and low overhead MAC in the preceding chapter, here we study the power saving issue in the TDMA-like scheme such as Two-Step Multi-Polling (TSMP) [18] in this chapter. It is different from the ordered-contention-based we proposed in Chapter 5 in determining the exact channel usage time of each notified STA. The TDMA-like manner seems to bring the advantage in easily determining the wake-up times. However, over the wireless fading channel, even the source of collision is eliminated and an SNR-based rate adaptation is adopted, frame transmissions still subject to a target error rate and cannot be predicted exactly. Therefore, consideration of frame retransmission is necessary and the power saving issue induced from error recovery is investigated in this work. We incorporate wake-up time schedules and a distributed wake-up time updating algorithm into TSMP to increase the chance of eliminating unnecessary overhearing induced from retransmissions.

The rest of this chapter is organized as follows. Section 6.1 reviews the methods of transmission error recovery and introduces the mechanism of TSMP scheme. In Section 6.2, we first present a wake-up time schedule without considering transmission errors and then another schedule which considers the impact of transmission errors. The distributed wake-up time updating algorithm is also presented in this section. After simulation results shown in

Section 6.3, this chapter is concluded in Section 6.4.

6.1 System Model

6.1.1 Error recovery

Over a time-varying wireless fading channel, frame transmissions are experiencing some level of error. Thus frame retransmission is necessary and it will occupy another period of time. Based on the timing of retransmission without recontending for the channel access, there are two kinds of error recovery policies depicted in [30]. The first is immediate retransmission which means that STA will initiate another transmission immediately when an ACK-timeout is expired or an NAK is received. The other is delayed retransmission which requires the AP to reschedule the failed transmission in some other period of time.

Immediate retransmission is good for system throughput because the sender retransmits immediately after failure and thus does not create channel idle time. Moreover, the transmission order can be maintained and no additional handshaking is necessary for retransmissions. However, STAs of later orders may be delayed for their accesses and overhear the overtime transmissions. On the other hand, the delayed retransmission is good for not changing the announced schedule thus keeps the STAs access the channel on time. Consequently, it keeps the access timings of STAs and the advantage of energy saving. However, STAs which wish to recover frame errors need to wait for another schedule time. Their frame delays are thus increased. Besides, the time originally allocated for transmitting the packets following the failed one may become idle period if the STA stops transmission once error is detected. The idle time can be eliminated but the packets may be transmitted out of order.

6.1.2 Two-Step Multi-Polling

TSMP scheme was proposed in [18] to provide exact TXOP allocations. The idea is to poll the STAs which are likely to have pending data to transmit in the first multi-polling frame

called status-request multipoll (SRMP). The polled STAs should reply status-response (SR) frames with their buffer statuses and the selected downlink rates from the channel estimation based on the received SRMP frame. After collecting the responses from STAs, the AP should first estimate the uplink rate for STAs with the SR frames. Then the determined rate and the known buffered statuses can help the AP to calculate an exact schedule which will be announced in the data transmission multipoll (DTMP) frame. Therefore, it not only reduces the overhead from polling frames to each STA but also the overhead of polling an STA with nothing to send. Since the time allocation for each STA is clearly specified, it is very helpful from the perspective of saving energy. It seems to match both the goals of high throughput and low energy consumption.

Regarding the adopted rate adaptation scheme in TSMP, the modulation method is selected according to the perceived SNR to meet a target frame error rate. The resulted TDMA-like channel allocation seems to provide great opportunity for energy saving. However, the TXOP calculation provided in [18] does not take the time spent in retransmission into account. For the reason of reducing penalty on throughput, we assume that immediate retransmission policy is adopted. Nevertheless, the STAs of later access orders will suffer from the delay caused by the accumulated time spent on error recovery. The resulted overhearing elimination issue is therefore worthy to be investigated.

6.2 The Proposed Power Saving Scheme for TSMP

The original TSMP scheme does not consider the situation of error recovery and the time allocated to each STA is dedicated. To cope with transmission errors, we assume that STAs with transmission errors will retransmit the failed frames immediately and, therefore, the original schedule has to be shifted if error does happen. For an STA obeying the announced wake-up time schedule, it may probably wake up to sense that the wireless channel is still occupied and its access is delayed. To save energy, if the *learned delay* is long enough, it

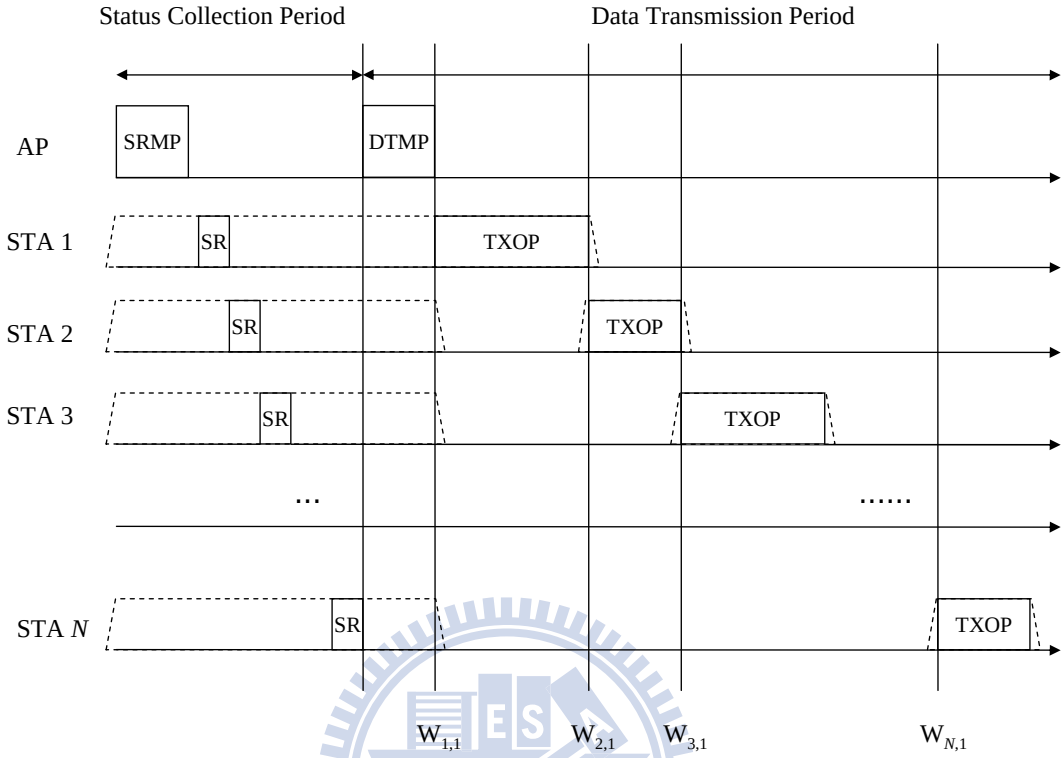

Figure 6.1: The TSMP scheme and the proposed wake-up time schedule for error-free condition.

should fall back to the Doze state until the delay expires. Since there could be error recovery during the new sleep period, it can determine whether or not to sleep again after it wakes up at the renewed wake-up time. Let $W_{i,k}$ denote the k^{th} wake-up time of STA i in a single polling round. Note that the time value mentioned hereafter represents the time relative to the receiving time of the DTMP frame. In the following, the derivation of wake-up time schedules and the distributed renewal algorithm are presented.

6.2.1 Wake-up Time Schedule without Pre-Delay

If transmission error is not considered, the wake-up time schedule can be easily derived from the obtained buffer status and PHY rate. Assume that there are N STAs and, without loss of generality, the STA scheduled to access the medium in the i^{th} order is called STA i. Let T_i denote the required TXOP for STA i, $1 \leq i \leq N$. To avoid overhearing, the initial wake-up

time for STA i is set as

$$W_{i,1} = \begin{cases} \sum_{m=1}^{i-1} T_m, & \text{if } \sum_{m=1}^{i-1} T_m > D \\ 0, & \text{if } \sum_{m=1}^{i-1} T_m \leq D \end{cases}, \text{ for } i > 1 \text{ and } W_{1,1} = 0. \tag{6.1}$$

D denotes the total hardware delay of switchovers from the Awake state to the Doze state and back to the Awake state. We assume that a STA can access the channel immediately after it wakes up if the channel is sensed idle. This situation happens if there is no transmission error before the STA wakes up. After the frame exchange process of a STA is finished, it can fall back to the Doze state till the time the next SRMP frame is scheduled to be transmitted.

Under the given schedule, STA i, $i > 1$, may overhear others' transmissions when it wakes up at time $W_{i,1}$ and there are frame retransmissions before $W_{i,1}$. Obviously, the expected duration of overhearing for STA j is longer than that for STA i if $j > i$. The energy consumed in overhearing could be significant if STAs are heavily loaded. Therefore, it is necessary for a STA to fall back to sleep because energy can be saved by doing so. To help STAs go back to the Doze state if the channel is still occupied when they wake up, we let the AP keep recording and announcing the ***accumulated delay*** caused by unsuccessful transmissions. It is denoted by $A(t)$ at time t and will be reset to zero when the next DTMP frame is transmitted. This information is piggybacked on all downlink frames including ACKs which are sure to be read by awake STAs in the Basic Service Set. Clearly, $A(t)$ is a *nondecreasing* function of t within a polling round. Besides, $A(t_1) = A(t_2)$ if the medium is busy and there is no transmission error during $[t_1, t_2]$.

As for STAs, they should keep separate records of ***cumulative consumed delay*** caused by error recovery. The consumed delay of STA i is the time STA i spent on waiting since $\sum_{m=1}^{i-1} T_m$, i.e., its access start time under error-free condition. Let $C_i(t)$ denote the consumed delay of STA i at time t. The initial value of $C_i(t)$ in each polling round is set as

$$C_i(W_{i,1}) = \left(W_{i,1} - \sum_{m=1}^{i-1} T_m\right)^+. \tag{6.2}$$

Each time when a new DTMP frame is received, $C_i(t)$ is reset to its initial value. Therefore,

the frame format of the DTMP frame should be incorporated with the wake-up time field for each polled STA. The suggested format for the DTMP frame is shown in Figure 6.2.

Octets: 2	2	6	1	7 × RecordCount				4
Frame Control	Duration /ID	BSSID	Record Count (0-255)	Polling Control				FCS
				AID (2 octets)	Rate (1 octet)	TXOP (2 octets)	Wake-up Time (2 octets)	

Figure 6.2: The suggested format of DTMP.

To maintain the access order and avoid collision, STA i keeps a variable E_i which represents its expected access start time. The initial value of E_i is set as

$$E_i = \max\Big(W_{i,1}, \sum_{m=1}^{i-1} T_m\Big) \tag{6.3}$$

and is recalculated each time a valid $A(t)$ is received. The updated E_i can be derived from the announced $A(t)$ and the kept $C_i(t)$. The details are shown in the following algorithm.

Renewal Algorithm for Wake-up Time and Expected Access Start Time

Assume that the AP assigns the initial wake-up time schedule $W_{1,1} \leq W_{2,1} \leq W_{3,1} \leq ... \leq W_{N,1} < \infty$ to STAs 1 to N with the DTMP frame and each STA wakes up at its designated time. According to the setting of initial value for E_i given above, we have $E_1 < E_2 < E_3 < ... < E_N$.

The idea is as follows. Consider STA i and assume that it wakes up for the k^{th} time at $W_{i,k}$. Let $R_{i,k}$, a random variable, represent the time duration for STA i to receive the nearest valid accumulated delay after $W_{i,k}$. The basic idea of updating the wake-up time and the expected access start time is as follows. If the channel is sensed idle at $W_{i,k}$, then STA i starts to transmit right away. On the other hand, if the channel is sensed busy at $W_{i,k}$, then a new wake-up time is updated at time $W_{i,k} + R_{i,k}$. In case the updated wake-up time is smaller than $W_{i,k} + R_{i,k} + D$, then E_i is set to the updated wake-up time, which is then set to ∞ to indicate the situation for STA i to remain awake till its frame exchange

process is finished. If the updated wake-up time is greater than $W_{i,k} + R_{i,k} + D$, then STA i falls back to sleep and wakes up again at the updated wake-up time. In this case, E_i is set to the updated wake-up time and the consumed delay is updated accordingly.

Initialization:

STA i maintains the following variables:

1. $C_i(t)$: consumed delay (It is set to $C_i(W_{i,1})$ given in equation (6.2) after the DTMP frame is received.)
2. E_i : expected access start time (It is set as equation (6.3) after the DTMP frame is received.)
3. W_i : wake-up time (It is set to $W_{i,1}$ announced in the DTMP frame.)

Updating:

while $W_i < \infty$ **do**

/* When the i^{th} STA wakes up for the k^{th} time at time $W_i = W_{i,k} < \infty$, its consumed delay is $C_i(W_{i,k})$. Assume that the medium is still occupied and the nearest valid accumulated delay is received at time $W_{i,k} + R_{i,k}$ with value $A(W_{i,k} + R_{i,k})$.*/

$temp = W_i$; /* a temporary variable */

$C_i(temp + R_{i,k}) = C_i(temp) + R_{i,k}$;

$W_i = temp + R_{i,k} + A(temp + R_{i,k}) - C_i(temp + R_{i,k})$;

$E_i = W_i$;

if $W_i < temp + R_{i,k} + D$

$W_i = \infty$;

/* Do NOT update the W_i in this polling round and stay in the Awake state. */

else

$C_i(W_i) = C_i(temp + R_{i,k}) + [A(temp + R_{i,k} - C_i(temp + R_{i,k}))]$

$= A(temp + R_{i,k})$;

/* Enter the Doze state and wake up again at time $W_i = W_{i,k+1}$.*/

end if

end while

The condition for STA i to start its transmission in case it stays in the Awake state since $W_{i,k}$ (stored in the *temp* variable) is described below.

---

while E_i is not reached **do**

Upon receiving a valid $A(\tau)$ at time τ:

$C_i(\tau) = C_i(temp) + \tau - temp$;

$E_i = \tau + A(\tau) - C_i(\tau)$;

if $\tau = E_i$

Start to transmit ;

else

Do nothing ;

/* Stay in the Awake state. */

end if

end while

Note that, according to the above updating algorithm, the condition for STA i to wake up at $W_{i,k}$ and sense the channel idle is the same as the condition that the wake-up time $W_{i,k}$ is equal to the expected access start time E_i. Consequently, STA i starts to transmit if and only if the current time is equal to E_i . Moreover, even with the above updating algorithm, STAs still access the medium in the order scheduled by AP. We sketch the proof in the following.

Proof:

Assume that we already have $W_{i,k} < W_{i+1,j} < W_{i,k+1} < \infty$. All we need to show is that either

1. $W_{i,k+1} < W_{i+1,j+1} < \infty$, i.e., there exists the $(j+1)^{th}$ wake-up time for STA $i+1$ which is still larger than the $(k+1)^{th}$ wake-up time of STA i, or

2. $W_{i+1,j+1} = \infty$ and $E_i < E_{i+1}$, i.e., STA $i+1$ would remain awake but its expected access start time is still later than that of STA i.

According to the definitions of $A(t)$, $C_i(t)$, and the fact that $W_{i,k} < W_{i+1,j} < W_{i,k+1} < \infty$, for the value of E_i updated by STA i at time $W_{i,k} + R_{i,k}$, we have

$$\begin{aligned} E_i &= W_{i,k+1} \\ &= W_{i,k} + R_{i,k} + A(W_{i,k} + R_{i,k}) - C_i(W_{i,k} + R_{i,k}) \\ &= W_{i,k} + R_{i,k} + A(W_{i,k} + R_{i,k}) - C_i(W_{i,k}) - R_{i,k} \\ &= W_{i,k-1} + A(W_{i,k} + R_{i,k}) - C_i(W_{i,k-1}) \\ &= \ldots = W_{i,1} + A(W_{i,k} + R_{i,k}) - C_i(W_{i,1}) \\ &= \sum_{m=1}^{i-1} T_m + A(W_{i,k} + R_{i,k}). \end{aligned} \tag{6.4}$$

The above equation can be derived iteratively using the following properties:

1. $W_{i,k} = W_{i,k-1} + A(W_{i,k-1} + R_{i,k-1}) - C_i(W_{i,k-1})$,

2. $C_i(W_{i,k}) = A(W_{i,k-1} + R_{i,k-1})$,

3. $C_i(W_{i,1}) = W_{i,1} - \sum_{m=1}^{i-1} T_m$.

Similarly, for the value of E_{i+1} updated by STA $i+1$ at time $W_{i+1,j}+R_{i+1,j}$, it holds that

$$\begin{aligned} E_{i+1} &= W_{i+1,j}+R_{i+1,j}+A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,j}+R_{i+1,j}) \\ &= W_{i+1,j}+R_{i+1,j}+A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,j})-R_{i+1,j} \\ &= W_{i+1,j}+A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,j}) \\ &= W_{i+1,j-1}+A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,j-1}) \\ &= ... = W_{i+1,1}+A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,1}) \\ &= \sum_{m=1}^{i} T_m + A(W_{i+1,j}+R_{i+1,j}). \end{aligned} \tag{6.5}$$

Since $A(t)$ is a non-decreasing function of t and the fact that $W_{i,k}+R_{i,k} \leq W_{i+1,j}+R_{i+1,j}$, we have $A(W_{i,k}+R_{i,k}) \leq A(W_{i+1,j}+R_{i+1,j})$. Consequently, it is true that $E_{i+1} \geq \sum_{m=1}^{i} T_m + A(W_{i,k}+R_{i,k}) > \sum_{m=1}^{i-1} T_m + A(W_{i,k}+R_{i,k}) = W_{i,k+1} = E_i$.

If $A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,j}+R_{i+1,j}) > D$, we get $W_{i,k+1} < W_{i+1,j+1} = E_{i+1} < \infty$, i.e., STA $i+1$ would wake up at $W_{i+1,j+1}$ which is later than $W_{i,k+1}$. Therefore, the access order is maintained in this case.

Otherwise, if $A(W_{i+1,j}+R_{i+1,j})-C_{i+1}(W_{i+1,j}+R_{i+1,j}) \leq D$, we have $W_{i+1,j+1} = \infty > E_{i+1}$ and STA $i+1$ keeps awake since time $W_{i+1,j}$. Based on the above result that $E_i = W_{i,k+1} < E_{i+1}$ where E_{i+1} is calculated by STA $i+1$ at time $W_{i+1,j}+R_{i+1,j}$, what we need to prove is the following: If STA i cannot access the medium immediately when it wakes up at $W_{i,k+1}$ and both STA i and STA $i+1$ remain awake and receive $A(W_{i,k+1}+R_{i,k+1})$ at time $W_{i,k+1}+R_{i,k+1}$, the updated E_i and E_{i+1} keep the relationship that $E_i < E_{i+1}$.

Again, at time $W_{i,k+1} + R_{i,k+1}$, for the value of E_i updated by STA i, we have

$$\begin{aligned} E_i &= W_{i,k+1} + R_{i,k+1} + A(W_{i,k+1} + R_{i,k+1}) - C_i(W_{i,k+1} + R_{i,k+1}) \\ &= W_{i,k+1} + A(W_{i,k+1} + R_{i,k+1}) - C_i(W_{i,k+1}) \\ &= W_{i,k} + A(W_{i,k+1} + R_{i,k+1}) - C_i(W_{i,k}) \\ &= \ldots = W_{i,1} + A(W_{i,k+1} + R_{i,k+1}) - C_i(W_{i,1}) \\ &= \sum_{m=1}^{i-1} T_m + A(W_{i,k+1} + R_{i,k+1}). \end{aligned} \tag{6.6}$$

Since $E_{i+1} = \tau + A(\tau) - C_{i+1}(\tau)$ and $C_{i+1}(\tau) = C_{i+1}(temp) + \tau - temp$, for the value of E_{i+1} updated by STA $i+1$, we have

$$\begin{aligned} E_{i+1} &= W_{i,k+1} + R_{i,k+1} + A(W_{i,k+1} + R_{i,k+1}) - C_{i+1}(W_{i,k+1} + R_{i,k+1}) \\ &= W_{i,k+1} + R_{i,k+1} + A(W_{i,k+1} + R_{i,k+1}) - \Big[C_{i+1}(W_{i+1,j}) + (W_{i,k+1} + R_{i,k+1}) - W_{i+1,j}\Big] \\ &= W_{i+1,j} + A(W_{i,k+1} + R_{i,k+1}) - C_{i+1}(W_{i+1,j}) \\ &= W_{i+1,j-1} + A(W_{i,k+1} + R_{i,k+1}) - C_{i+1}(W_{i+1,j-1}) \\ &= \ldots = W_{i+1,1} + A(W_{i,k+1} + R_{i,k+1}) - C_{i+1}(W_{i+1,1}) \\ &= \sum_{m=1}^{i} T_m + A(W_{i,k+1} + R_{i,k+1}). \end{aligned} \tag{6.7}$$

Therefore, it is true that $E_i < E_{i+1}$ because $\sum_{m=1}^{i} T_m > \sum_{m=1}^{i-1} T_m$. Consequently, the access order is still maintained for the case when $W_{i+1,j+1} = \infty$. This completes the sketched proof. □

6.2.2 Wake-up Time Schedule with Pre-Delay

The initial wake-up time schedule presented in the previous sub-section assumes no transmission error and thus is good for system throughput because it does not create any idle period. However, the number of switchovers may be considerable in an error-prone environment. Since switchovers consume energy, we present in this sub-section an alternative wake-up time schedule which takes the error recovery time into account by using the pre-delay technique.

Assume that the average frame error rate, say $(1-p)$, is known to the AP and errors occur independently for different transmission attempts. Let L_i and n_i represent the time for a frame exchange and the number of buffered frames in STA i, respectively. As a result, the average transmission time required for STA i, including retransmission, can be obtained as $\mu_i = n_i L_i/p$ while the variance is $\sigma_i^2 = n_i(1-p)L_i^2/p^2$. Therefore, we modify the initial wake-up time schedule as:

$$W_{i,1} = \begin{cases} \max\left(\sum_{m=1}^{i-1} T_m, \sum_{m=1}^{i-1} \mu_m + \alpha\sqrt{\sum_{m=1}^{i-1} \sigma_m^2}\right), \\ \qquad \text{if } \max\left(\sum_{m=1}^{i-1} T_m, \sum_{m=1}^{i-1} \mu_m + \alpha\sqrt{\sum_{m=1}^{i-1} \sigma_m^2}\right) > D \\ 0, \\ \qquad \text{if } \max\left(\sum_{m=1}^{i-1} T_m, \sum_{m=1}^{i-1} \mu_m + \alpha\sqrt{\sum_{m=1}^{i-1} \sigma_m^2}\right) \leq D. \end{cases} \tag{6.8}$$

Note that α in equation (6.8) is an adjustable parameter. It is clear that the value of α affects the performance of energy saving. In general, the goal of saving energy conflicts with the goal of high system throughput. A possible tradeoff is to select a pre-defined percentage of bandwidth utilization loss, say 5%, and maximize energy saving. Also note that the wake-up time of a STA should not be earlier than its error-free access start time because such a schedule will help neither the system throughput nor the energy saving.

The updating algorithm proposed in the previous subsection is still applicable to this pre-delayed wake-up time schedule. However, the content of $A(t)$ should be modified to make the algorithm work properly. In addition to keeping the time spent on unsuccessful transmission in $A(t)$, the *channel idle time* caused by conservative wake-up time schedule should also be taken into account.

6.3 Performance Evaluation

6.3.1 Simulation Scenario

The scenario we considered in our simulation analysis is as follows. Assume that there are 20 STAs each with 10 frames ready for transmission during each service interval of $100ms$.

Each frame takes $150\mu s$ for the whole exchange process. The setting is equivalent to a high quality MPEG-4 video traffic model with mean bit rate of 770 Kbps and an 800-byte payload length over a link of speed 216 Mbps for data frames and 24 Mbps for ACK frames. The required attempt for a frame to be successfully delivered is geometrically distributed with probability p. The required time for switchover is set to 250 μs according to [30].

6.3.2 Simulation Results

Performance of wake-up time schedules without applying updating

We first investigate both the performance of energy saving and system throughput for the wake-up time schedules without applying the distributed updating algorithm. The successful transmission probability is set to 0.9. For convenience, schedule without pre-delay and schedule with pre-delay are denoted by Schedule 1 and Schedule 2, respectively. The performance of energy saving is measured by both the overhearing time and the reduced proportion of overhearing time when compared with that of Schedule 1.

As shown in Figures 6.3 (a) and (b), one can see that simply applying Schedule 1 to the TSMP scheme will make STAs suffer from overhearing under the given erroneous condition. On the other hand, the STAs of later orders can benefit from the schedules with pre-delay considerations to eliminate larger proportions of unnecessary overhearing.

Note that STA 2 has the largest percentage of improvement when $\alpha = 1$. For our studied simulation scenario, we have $W_{i,1} = (i-1)\mu + \alpha\sqrt{i-1}\sigma$ for STA i ($i > 1$). Consequently, $W_{i,1} - W_{i-1,1} = \mu + \alpha(\sqrt{i-1} - \sqrt{i-2})\sigma$ decreases as i increases and larger value of raises the decreasing rate. Hence, when $\alpha = 1$, what is special for STA 2 is that the preceding STA 1 is not delayed by the schedule while $W_{2,1} - W_{1,1} = \mu + \sigma$ is larger than that of succeeding STAs. It therefore benefits from the setting of schedule to have better improvement. For such a conservative schedule for STA 2, STA 3 wakes up relatively earlier when compared with the situation of STA 2 and thus has larger overhearing time and less improvement

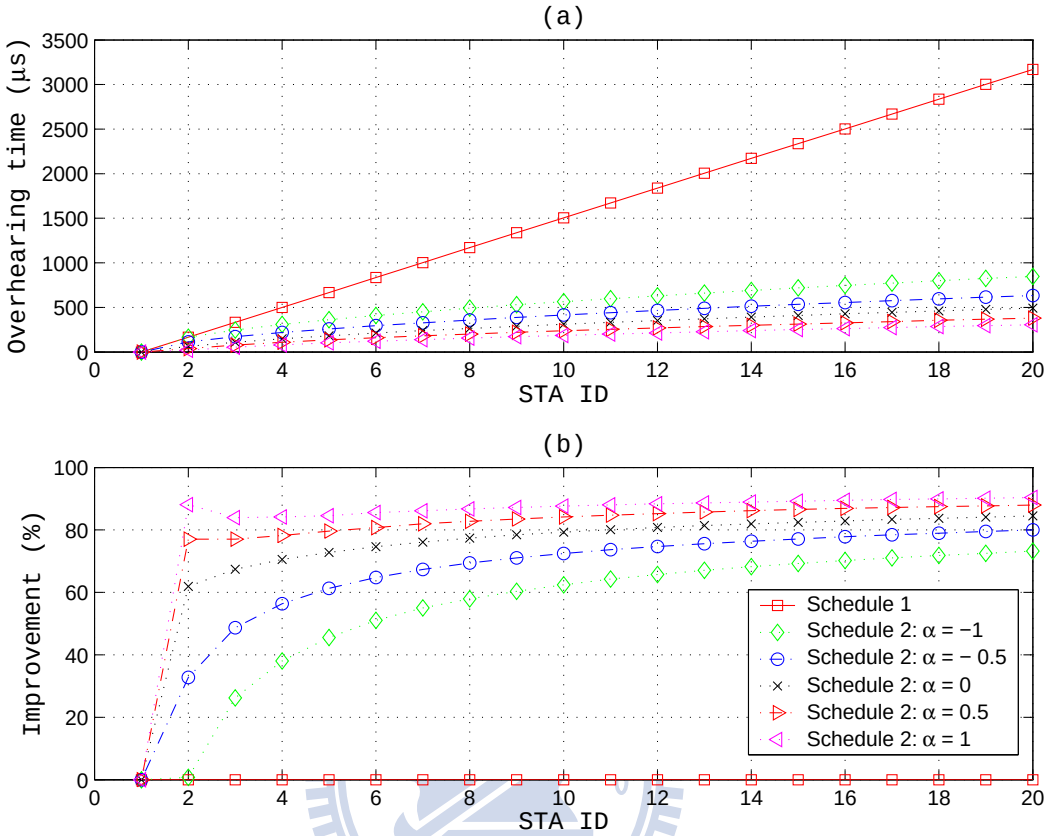

Figure 6.3: The improvement on energy saving for both schedules without the updating algorithm.

percentage than STA 2 does. Nevertheless, since $C_i(W_{i,1}) = (i-1)(1-p)\mu + \alpha\sqrt{i-1}\sigma$ increases with i, more eliminated overhearing time for STAs with increasing order can still be expected. Therefore, the percentage of improvement for STA i increases as i increases.

The performance of system throughput is shown in Figures 6.4 (a) and (b). The throughput is defined as the required time for data transmission over total transmission finish time for the first i STAs. The degradation in throughput of Schedule 2 is measured by the difference of throughput compared with that of Schedule 1. Note that the system throughput of Schedule 1 is the same as that of the original TSMP scheme. It can also be explained by $W_{i,1} - W_{i-1,1} = \mu + \alpha(\sqrt{i-1} - \sqrt{i-2})\sigma$ that the degradation mitigates with increasing order since STAs wake up and finish their transmission earlier. As can be seen from Figures 6.3 and 6.4, for a system tolerating 3% degradation of system throughput for 20 STAs, it

should choose $\alpha = 1$ to save more energy of STAs.

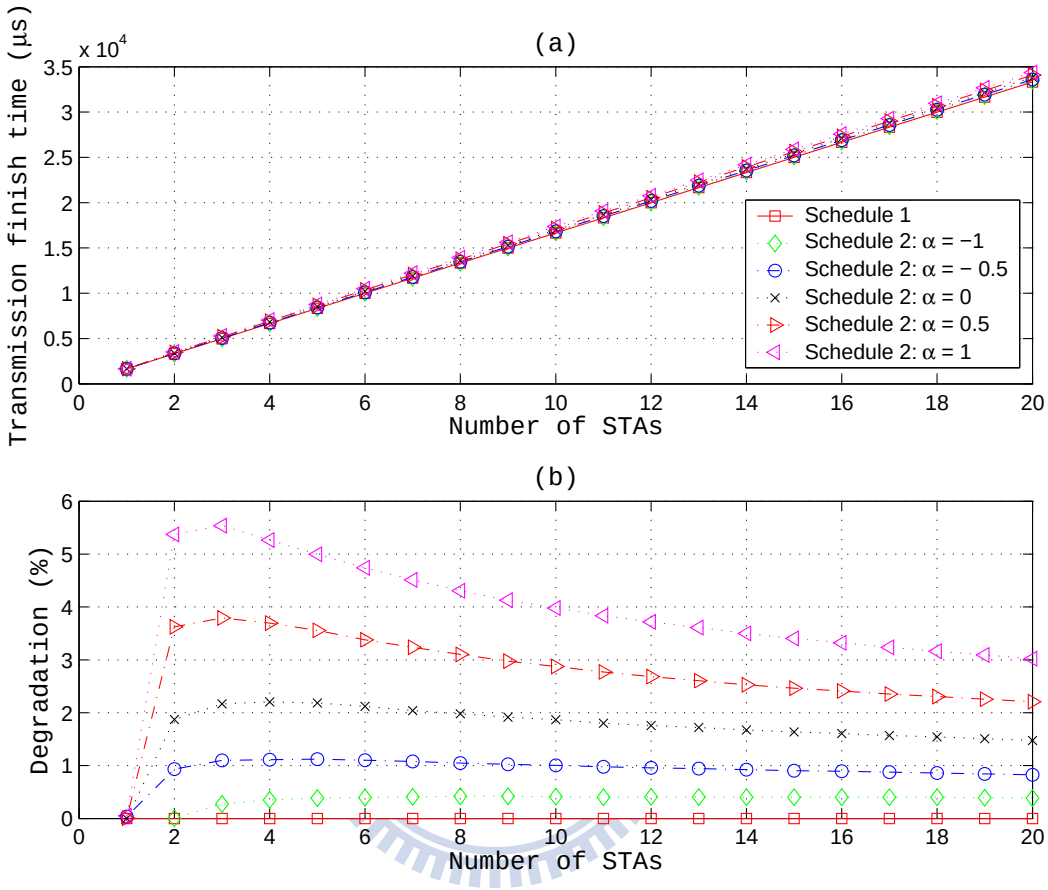

Figure 6.4: System throughput performance of our proposed schedules.

Performance of wake-up time schedules applying updating

The performances of energy saving when applying the distributed wake-up time updating algorithm are shown in Figures 6.5 (a) and (b). With the updating algorithm, the improvement in energy saving is significant for Schedule 1. However, there is only slight improvement for Schedule 2. This is because Schedule 2 has eliminated large proportion of overhearing by pre-delaying the initial access attempts. Since the proposed updating algorithm does not alter the actual access start time for both Schedule 1 and Schedule 2, the system throughputs are the same as those shown in Figure 6.4 and thus are not shown here again.

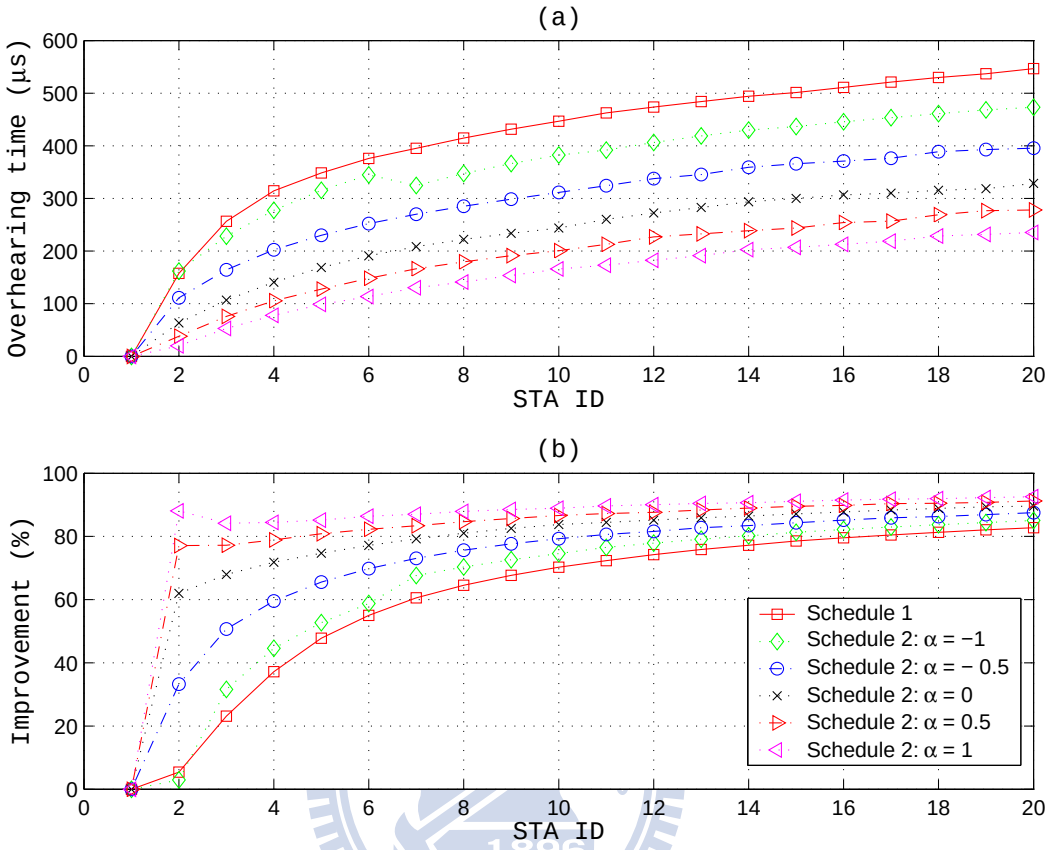

Figure 6.5: The improvement on energy saving for both schedules with the updating algorithm.

6.4 Summary

In this chapter, we proposed wake-up time schedules and an update algorithm for the TDMA-like TSMP scheme to improve its performance on energy saving when the immediate retransmission is adopted for error recovery. According to the simulation results, the proposed schemes can effectively alleviate the problem of overhearing under the erroneous wireless environment to prolong the usage time of wireless devices. The impact of incorporating the wake-up time schedule into the TSMP scheme on system throughput is also studied. For our proposed schedule with pre-delay, the wake-up time schedule can be adjusted by selecting a suitable parameter value to achieve better energy saving and meet a predetermined loss of system throughput. Since the proposed wake-up time schedules and the updating algorithm are simple, we believe they are feasible to be implemented in real systems.

Chapter 7

Conclusions

In this dissertation, we study the important energy saving issues regarding the shared medium, infrastructure wireless LAN. Under the multi-user scenario, the *overhearing problem* is highlighted and thoroughly studied for different medium access mechanisms.

We propose an algorithm to schedule the service start time of TSs belonging to different applications for the IEEE 802.11e S-APSD mechanism. Simulation results show that the proposed scheduling algorithm can achieve similar performance as [38] in preventing the overlapping of SPs while the implementation complexity is largely reduced by utilizing the proven periodicity of service schedules and the limited number of traffic classes. Since the scheduling optimality of Chapter 3 and [38] depend on the joining order of TSs, they could be unsatisfactory in some cases. To improve the energy saving performance, we also studied the problem of rearranging existing scheduled events to enlarge the minimum distance of the system. We show that the maximum of minimum distance between two periodic scheduled events is closely related to the greatest common divisor of their periods. Taking advantage of this observation and scheduling TSs according to the ascending order of period, we devise the GMD greedy scheduling to increase the tractability while improving the energy saving performance.

Regarding the low overhead multi-polling mechanism, especially for serving uplink VBR traffic, we propose several ideas for the goal of energy saving and a wake-up time schedule

is presented to achieve the best energy saving with a tolerable sacrifice of system bandwidth utilization. Both analytical and simulation results show that the overhearing problem can be effectively mitigated with the proposed EE-Multipoll mechanism especially when there are a lot of STAs and STAs are heavily loaded. Since the wireless environment could cause transmission error more likely than wired scenario, the energy saving issue induced from error recovery is also studied in this dissertation. Without altering the transmission order while achieving the goal of energy saving, the settings of wake-up time schedule and the renewal algorithm are studied for the TDMA-like multi-polling mechanism.

The most significant conclusion of this dissertation is that, under a tolerable performance tradeoff, the focus of the energy conservation should try to put as more STAs into low power state as possible and as long as possible. The proposed scheduling algorithms and wake-up time schedules can direct STAs not involved in a transmission going into Doze state and reduce the coordination effort for them when they are ready to accessing the channel. Moreover, the information contained in the overheard transmission can also aid the STAs going to the Doze state if appropriate. The performance of energy conservation is expected to be more significant for a larger network since a greater percentage can use low power state by the coordination and scheduling.

Extensions of our design can be done in several directions. An interesting topic is to consider the duration and the randomness of the service period when considering the scheduling in S-APSD mechanism. Another interesting topic is the scheduling algorithm for the Power Save Multiple Poll mechanism (PSMP) in IEEE 802.11n [3]. This mechanism is different from our proposed EE-Multipoll in channel accessing and handling of the traffic burst. The specific scheduling algorithm for PSMP and the control of power states for Multiple-Input Multiple-Output (MIMO) transceiver are still open issues.

Bibliography

[1] IEEE 802.11 WG: IEEE Standard 802.11-1999, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, 1999.

[2] 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 Enhancements.

[3] Std 802.11n-2009, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 5: Enhancements for Higher Throughput.

[4] M. S. Gast, *802.11 Wireless Networks - The Definition Guide*. O'Reilly, 2002.

[5] W. Ye, J. Heidemann and D. Estrin, "Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Trans. Networking*, vol. 12, no. 3, pp. 493- 506, June 2004.

[6] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114V117, Apr. 1965.

[7] K. Lahiri, A. Raghunathan, S. Dey, and D. Panigrahi, "Battery-driven system design: A new frontier in low power design," in *Proc. Intl. Conf. on VLSI Design*, pp. 261V267, Jan. 2002.

[8] R. Mangharam, R. Rajkumar, S. Pollin, F. Catthoor, B. Bougard, L. Van der Perre, and I. Moeman, "Optimal fixed and scalable energy management for wireless networks," in *Proc. IEEE INFOCOM*, Vol. 1, pp. 114V125, Mar. 2005.

[9] S. H. Gunther, F. Binns, D. M. Carmean, and J. C. Hall, "Managing the Impact of Increasing Microprocessor Power Consumption," Intel Technology Jounal, First Quarter, 2001.

[10] Atheros White Paper, "Power Consumption and Energy Efftciency Comparisons of WLAN Products". 2003.

[11] Dell, "Dell TrueMobile 1400 WLAN Card." [Online].
Available: http:// support.ap.dell.com/docs/network/p44970/en/specs.htm

[12] N.A. Pantazis and D.D. Vergados, "A survey on power control issues in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, Vol. 9 , no. 4, pp. 86 - 107, 2007.

[13] D. Qiao, S. Choi,and K.G. Shin, "Interference Analysis and Transmit Power Control in IEEE 802.11a/h Wireless LANs," *IEEE Trans. Networking*, Vol. 15 , no. 5, pp. 1007 - 1020, 2007.

[14] Y. Xiao and J. Rosdahl, "Throughput and Delay Limits of IEEE 802.11," *IEEE Commun. Lett.*, vol. 6, no.8, pp. 355- 357, Aug. 2002.

[15] S.-C. Lo, G. Lee, and W.-T. Chen, "An Efficient Multipolling Mechanism for IEEE 802.11 Wireless LANs," *IEEE Trans. Comput.*, vol. 52, no. 6, pp. 764- 778, June 2003.

[16] S.-C. Lo and W.-T. Chen, "An Efficient Scheduling Mechanism for IEEE 802.11E MAC Enhance-ments," in *Proc. IEEE WCNC*, pp. 777- 782, vol. 2, March 2004.

[17] S. Kim, Y. Kim, S. Choi, K. Jang and J. Chang, "A High-Throughput MAC Strategy for Next-Generation WLANs," in *Proc. IEEE WoWMoM*, pp. 278- 285, June 2005.

[18] B.-S. Kim, S.W. Kim, Y. Fang, and T.F. Wong, "Two-Step Multipolling MAC Protocol for Wireless LANs," *IEEE J. Select. Areas Commun.*, vol. 23, no. 6, pp. 1276-1286, June 2005.

[19] Z.-T. Chou, C.-C. Hsu, and S.-N. Hsu, "UPCF: A New Point Coordination Function With QoS and Power Management for Multimedia Over Wireless LANs," *IEEE/ACM Trans. Networking*, vol. 14, no. 4, pp. 807-820, Aug. 2006.

[20] J. A. Stine and G. D.Veciana, "Improving Energy Efficiency of Centrally Controlled Wireless Data Networks," *ACM/Kluwer Wireless Networks*, vol.8, pp. 681-700, Nov. 2002.

[21] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks," in Proc. *IEEE INFOCOM*, Vol. 1, pp. 200-209, June 2002.

[22] E.-S. Jung and N. H. Vaidya, "An Efficient MAC Protocol for Wireless LANs," in *Proc. IEEE INFOCOM*, vol. 3, pp. 1756-1764, June 2002.

[23] S. Singh, and C. S. Raghavendra, "PAMAS - Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks," in *ACM SIGCOMM*, Vol. 28 , no. 3, pp. 5 - 26, July 1998.

[24] H. Woesner, J. P. Ebert, M. Schlager, and A. Wolisz, "Power-Saving Mechanisms in Emerging Standards for Wireless LANs: The MAC Level Perspective," *IEEE Persinal Communications*, pp. 40-48, June 1998.

[25] S. Jayashree, and C. Siva Ram Murthy, "A Taxonomy of Energy Management Protocols for Ad Hoc Wireless Networks," *IEEE Commun. Magazine*, Vol. 45 , no. 4, pp. 104 - 110, 2007.

[26] R. Krashinsky and H. Balakrishnan, "Minimizing Energy for Wireless Web Access with Bounded Slowdown," in Proc. *ACM MOBICOM*, Sep. 2002.

[27] D. Qiao and K. G. Shin, "Smart power-saving mode for IEEE 802.11 wireless LANs," in Proc. *IEEE INFOCOM*, Mar. 2005.

[28] X. Pérez-Costa and D.Camps-Mur, "APSM: Bounding the Downlink Delay for 802.11 Power Save Mode," in Proc. *IEEE ICC*, May 2005.

[29] C. E. Jones, K. Sivalingam, P. Agrawal, and J.-C. Chen, "A Survey of Energy Efficient Network Protocols for Wireless Networks," *ACM/Kluwer Wireless Networks*, vol. 7, pp. 343-358, Nov. 2001.

[30] A. Kamerman and L. Monteban, "WaveLAN-II: A High-Performance Wireless LAN for the Unli-censed Band," *Bell Labs Technical Journal*, vol. 2, no. 3, 1997.

[31] Y. Jiao, A. R. Hurson, and B. A. Shirazi, "Online Adaptive Application-Driven WLAN Power Management," in *Proc. IEEE Globecom*, pp. 2663-2668, Nov. 2005.

[32] J.-R. Hsieh, T.-H. Lee, and Y.-W. Kuo, "Power Efficient Multipolling Mechanism for Next Generation Wireless LANs," in *Proc. IEEE VTC2007-Spring*, pp. 2971-2975, Apr. 2007.

[33] MATLAB and Simulink for Technical Computing, http://www.mathworks.com/

[34] B.W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[35] A. Grilo, M. Macedo, and M. Nunes, "A Scheduling Algorithm for QoS Support in IEEE 802.11e Networks," *IEEE Wireless Commun. Mag.*, vol. 10, no. 3, pp.36-43, June 2003.

[36] Q. Zhao and D. H.K. Tsang, "An Equal-Spacing-Based Design for QoS Guarantee in IEEE 802.11e HCCA Wireless Networks," *IEEE Trans. Mobile Computing*, Vol. 7, no. 12, Dec. 2008.

[37] S. Mangold, S. Choi, G. R. Hiertz, O. Klein, B. Walke, "Analysis of IEEE 802.11e for QoS Support in Wireless LANs", *IEEE Wireless Comm. Mag.*, Vol. 10, no. 6, Dec, 2003.

[38] X. Pérez-Costa, D. Camps-Mur, J. Palau, D. Rebolleda and S. Akbarzadeh, "Overlapping Aware Scheduled Automatic Power Save Delivery Algorithm," in *Proc. of European Wireless Conference (EW)*, Paris, France, Apr. 2007.

[39] X. Pérez-Costa, D.Camps-Mur and T.Sashihara. "Analysis of the Integration of IEEE 802.11e Capabilities in Battery Limited Mobile Devices," *IEEE Wireless Comm. Mag., special issue on Internetworking Wireless LAN and Cellular Networks*, Vol. 12, no. 6, December 2005.

[40] T. H. Cormen, C. R. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, Second ed., MIT Press and McGraw-Hill. ISBN 0262032937.

[41] F. Fitzek , A. Koepsel , A. Wolisz , M. Krishnam , and M. Reisslein, "Providing Application-Level QoS In 3G/4G Wireless Systems: A Comprehensive Framework Based On Multi-Rate CDMA," *IEEE Wireless Comm. Mag.*, Vol. 9, no. 2, Apr., 2002.

[42] T. Simunic, H. Vikalo, P. Glynn, and G. D. Micheli, "Energy Efficient Design of Portable Wireless Systems," in *Proc. of the International Symposium on Low Power Electronics and Design*, pp. 49-54, 2000.

[43] J.-R. Hsieh and T.-H. Lee, "An Energy Saving Scheme with Error Recovery for Multi-Polling in Wireless LANs," in *Proc. IEEE VTC2008-Spring*, pp. 2121-2125, May 2008.

[44] T.-H. Lee and J.-R. Hsieh, "An Efficient Scheduling Algorithm for Scheduled Automatic Power Save Delivery for Wireless LANs," in *Proc. IEEE VTC2010-Spring*, May 2010.

[45] J.-R. Hsieh and T.-H. Lee, "Energy-Efficient Multi-polling Scheme for Wireless LANs," *IEEE Trans. Wireless Commu.*, Vol. 8, No. 3, pp. 1532-1541, Mar. 2009.

[46] T.-H. Lee and J.-R. Hsieh, "Low Complexity Class-based Scheduling Algorithm for Scheduled Automatic Power-Save Delivery for Wireless LANs," submitted to *IEEE Trans. Mobile Computing*. (under review)

[47] T.-H. Lee, J.-R. Hsieh, M.-C. Huang and Y.-W. Huang, "A Bandwidth Efficient Pairing Strategy for the MIMO-OFDM Based WLANs," in *Proc. of IEEE VTC2009-Spring*, Apr. 2009.

[48] T.-H. Lee, J.-R. Hsieh, and H.-W. Chen, "A Rearranegement Algorithm for Scheduled Automatic Power Save Delivery of Wireless LANs," accepted and to appear in *IEEE Tencon2010*, Nov. 2010.

[49] F.H.P Fitzek and M.Reisslein, "MPEG-4 and H.263 Video Traces for Network Performance Evaluation," *IEEE Network*, Vol. 15, No.6, pp. 40-54, Dec. 2001.
http://www-tkn.ee.tu-berlin.de/research/trace/trace.html