國立交通大學

資訊科學與工程研究所

碩 士 論 文

在 IEEE 802.11s 網格狀網路中使用較少的天線運用多通道

Utilizing Multi-Channel with Less Radios in the IEEE 802.11s Mesh Networks

研 究 生：古佳育

指導教授：林盈達 教授

中 華 民 國 九 十 六 年 六 月

在 IEEE 802.11s 網格狀網路中使用較少的天線運用多通道

Utilizing Multi-Channel with Less Radios in the IEEE 802.11s Mesh Networks

研 究 生：古佳育 Student: Chia-Yu Ku

指導教授：林盈達 Advisor: Dr. Ying-Dar Lin

國立交通大學

資訊科學與工程研究所

碩士論文

A Thesis
Submitted to Institutes of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
In

Computer Science and Engineering

June 2007

HsinChu, Taiwan, Republic of China

中華民國九十六年六月

在 IEEE 802.11s 網格狀網路中使用較少的天線運用多通道

學生：古佳育　　　　　　　　　　　　　　　指導教授：林盈達

國立交通大學資訊科學與工程研究所

摘要

在 IEEE 802.11s 網格狀網路中，基本服務組合中的流量可能會因為同一通道中的載波偵測造成網格狀網路中的流量無法被處理。使用多通道來解決這個問題是個直覺的想法，但是這需要多個天線。使用少於通道數的天線來降低成本時，在通道之間做切換會造成封包丟失與連線中斷。因此需要一個機制來避免在沒注意到的通道中發生封包丟失以及一個演算法去處理天線的切換。這篇論文提出了一個時間比例考量切換監督者來解決上述的問題。當切換點切換到別的通道時，IEEE 802.11s 中的省電機制與混合控制通道訪問被用來通知鄰居幫切換點暫存資料。此外還有一個為多天線設計的演算法被用來處理通道切換。時間比例考量切換監督者實作在 RTL8186，瑞昱交大聯合研發中心的 IEEE 802.11s 網格狀網路開發計畫所採用的平台。實作評估結果驗證了時間比例考量切換監管者能將封包丟失率減少到百分之一以下，並且將浪費的通道時間減到最小。此外模擬結果展示出在多通道環境中生產率與多天線的關係。時間比例考量切換監管者提供了 RTL8186 一個優良的多通道傳輸方案，以及評估結果給與設計網格狀網路設備的硬體一個參考。

關鍵字：網格狀網路，多通道，通道切換，時間比例考量

Utilizing Multi-Channel with Less Radios in the IEEE 802.11s Mesh Networks

Student: Chia-Yu Ku **Advisor: Dr. Ying-Dar Lin**

Department of Computer Science and Engineering

National Chiao Tung University

Abstract

In the IEEE 802.11s mesh networks, basic service set traffic might starve mesh forwarding traffic due to carrier sensing in a single channel. Using the multi-channel transmission is an intuitive solution but multiple radios are required. When radios are less than channels for lower cost concern, switching among channels results in packet loss and link disassociation. Therefore, a mechanism to avoid packet loss in the unattended channel and an algorithm to handle channel switching are necessary. This work proposes Time Ratio Aware Switching Superintendent (TRASS) to tackle the above challenges. The power-saving mechanism in IEEE 802.11s and Hybrid coordination function Controlled Channel Access (HCCA) are used to notify neighbors to buffer for the switch node. Moreover, an algorithm designed for multiple radios is proposed to handle channel switching. TRASS is implemented on RTL8186, the adopted platform by the mesh networking project of Realtek-NCTU Joint Research Center. The benchmark results of implementation demonstrate that TRASS reduce packet loss to the value less than 1% and minimize wasted channel time. Furthermore, the simulation results show the relationship between throughput and multi-radio in the multi-channel environment. TRASS provides an excellent solution for multi-channel transmission over RTL8186 and the evaluation results give a reference to design the hardware of the mesh networking device.

Keywords: mesh networks, multi-channel, channel switching, time ratio aware

Contents

List of Figures

List of Tables

Chapter 1 Introduction

Wireless networks offer several advantages such as easy and fast deployment over conventional wired networks. To provide large cover range wireless networks such as MAN, there are three kinds of technologies. The first one is enabling the delivery of last mile wireless broadband access like IEEE 802.16. The second one is deploying a high density of wireless local area networks such as IEEE 802.11 [1]. The last one is a wireless mesh network which intends to support a broad range of deployment scenarios and data delivery over self-configuring multi-hop topologies. IEEE 802.11s [2], an extension for mesh networks of IEEE 802.11, specifies a framework using the IEEE 802.11 MAC/PHY layers to support broadcast, multicast and unicast data delivery by data forwarding and path selection.

In the infrastructure basic service set (infrastructure BSS) of IEEE 802.11, a station (STA) accesses Internet through an access point (AP) which forwards the data to the destination. In each infrastructure BSS, only a single channel is needed for the STAs to communicate with the AP. However, in IEEE 802.11s, there are two kinds of traffic, BSS traffic which is forwarded by the AP to the STAs and mesh forwarding traffic which is forwarded by the intermediate mesh points (MPs) on the path. Thus, a mesh access point (MAP) must deal with these two kinds of traffic simultaneously. In a single-channel scenario, BSS traffic occupying the channel potentially starves the neighbor MPs and results in long packet delay or serious packet loss. Therefore, separating different traffic into different channels is an intuitive thought. In a multi-channel scenario, it can not only separate BSS traffic and mesh forwarding traffic into two channels but also separate mesh forwarding traffic into multiple channels to improve network throughput.

In the IEEE 802.11 wireless networks, each channel applies for one radio. Representing that, N channels need N radios on each wireless device for the best performance. Nevertheless, the more radios, the more hardware cost and power consumption, especially for mobile stations. Moreover, a lot of devices were designed for a fixed number of radios and they are not easy to modify to equip with more radios. When radios are less than channels, switching among channels results in packet loss and link disassociation in unattended channels. Such a device whose radios switch among channels is called a switch node like a MAP or a MP in the multi-channel mesh networks. To take a MAP for an example, link disassociation makes its neighbor MPs rediscover the routing path and forces the STAs associate with another AP. While the radio switches back to the channel, the switch node rejoins the topology of the mesh network and the STAs might re-associate with it because of a better link quality. The above may exhaust the network resources if it happens frequently.

Several researchers have studied the multi-channel protocols. They discussed that each node requires one radio [3] [4], or two radios [5] [6] [7], or N radios [8] [9] for a N-channel environment. All the methods require that each node supports their specific protocols which might not be compatible with the standard. In addition, using a single radio in a multi-channel mesh network requires one unrealistic constraint, the global timer synchronization, which is very difficult to achieve in the mesh networks because the delay in a large area cannot be correctly estimated. Another work dealing with channel switching [10] provides a method for a STA to connect to multiple networks with a single radio, but it is insufficient for the mesh networks because there are more scenarios like channel switching for a MAP and equipping with multiple radios.

Although much research has been devoted to the multi-channel wireless networks, little information is available on the IEEE 802.11s mesh networks. This paper investigates how to use less radios to switch among channels in the IEEE 802.11s mesh networks. The proposed method, Time Ratio Aware Switching Superintendent (TRASS), could run on all devices in a mesh network and handle channel switching. This method includes two parts: (1) TRASS mechanisms existing in IEEE 802.11 and IEEE 802.11s ensures frame delivery to the switch node. For the infrastructure BSS, a switch node uses the Hybrid coordination function Controlled Channel Access mechanism (HCCA) [11] or the CTS-to-self mechanism [12] to notify the STAs to wait for it. For the mesh networks, a switch node enters its power-saving mode [2] to notify its neighbor MPs to buffer the data; (2) a TRASS algorithm considering the general case that is a switch node has only N radios in a M-channel environment while M is greater than N. This algorithm selects which channel to stay in according to the traffic load ratio for the switch node and adaptively allocates time for the channel by its total traffic load ratio. This paper carries out simulation-based and implementation-based studies. It examines packet loss ratio, channel utilization, throughput and average latency. Additionally, the relationship between these factors and the number of radios in the multi-channel mesh networks is also investigated.

The rest of this paper is organized as follows. Chapter 2 introduces the IEEE 802.11 MAC, the architecture of the IEEE 802.11s mesh networks and the impact of channel switching with less radios in a multi-channel environment. Chapter 3 presents TRASS and illustrates its detailed operations with some examples. Chapter 4 shows the system architecture of implementation and chapter 5 investigates the evaluation results of simulation and implementation. Finally, we conclude this work with some future direction in chapter 6.

Chapter 2 Background & Problem Statement

First, this chapter describes the limitations and the basic behaviors of IEEE 802.11. IEEE 802.11 uses the carrier sense multiple access with collision avoidance mechanism (CSMA/CA) to detect whether the medium is available to avoid collisions and provides distributed and centralized coordination functions to achieve medium access control. Second, the architecture of the IEEE 802.11s mesh networks is introduced. Because of the diversity of architecture between IEEE 802.11 and IEEE 802.11s, using multiple channels is necessary. Finally, the problems caused by channel switching in the IEEE 802.11s mesh networks and the issues for solving the problems are described.

2.1 Background

2.1.1 IEEE 802.11

IEEE 802.11 uses the CSMA/CA mechanism to access medium. Each node checks whether the medium is available by carrier sensing before transmitting. Virtual carrier sensing, one of the carrier sensing functions in IEEE 802.11, is provided by the network allocation vector (NAV) which indicates the amount of time that the wireless medium will be reserved. Each node computes the expected amount of time to complete its operation sequence and sets this value to the NAV and then the other nodes count down from the NAV to zero. The NAV being nonzero implies that the virtual carrier sensing function deems that the medium is occupied. On the contrary, the virtual carrier sensing function indicates that the medium is available when the NAV is zero. It protects the operation sequences from interruption by the NAV.

The IEEE 802.11 MAC layer provides two medium access coordination functions: the distributed coordination function (DCF) [1] and the point coordination function (PCF) [1]. The DCF is a contention-based mechanism. Each node checks whether the medium is available before attempting to transmit. When the medium is available, each node accesses the medium after a random back-off time generated from its contention windows. On the other hand, The PCF is a contention-free mechanism. A point coordinator which resides in an AP uses a centralized access control method. When the PCF is working, time is divided into the contention free period (CFP) and the contention period (CP). In the CP, the DCF works and each node contends for medium access. In the CFP, each node transmits frames only when it is polled by the point coordinator. In IEEE 802.11e, the PCF has been extended to the HCCA. The basic behavior of the HCCA is similar to the PCF but the HCCA includes more mechanisms for the quality of services.

2.1.2 IEEE 802.11s mesh networks

IEEE 802.11s defines the mesh networking using the IEEE 802.11 MAC/PHY layers that support layer-2 path selection protocols and data forwarding over multi-hop topologies. Moreover, IEEE 802.11s also defines the multi-channel mesh networking to separate traffic into different channels to improve network throughput. Figure 1 illustrates the architecture of the mesh networks. Each node which joins a mesh network is called a mesh point (MP). A MP which also plays the role of an AP is called a mesh access point (MAP). A MP which bridges wired networks is called a mesh point portal (MPP). Mostly, an user is a MP or a STA. For the MP case, an user transmits data through its neighbor MPs which forward these data to the destination. For the STA case, the mesh networks play the role of a wireless distribution system which is extended from wired networks. An user transmits data through the MAP and then the MAP forwards these data to the mesh networks. If BSS traffic and mesh forwarding traffic use the same channel, they starve each other because the channel can only be occupied by one side. As a result, they are usually separated into different channels.

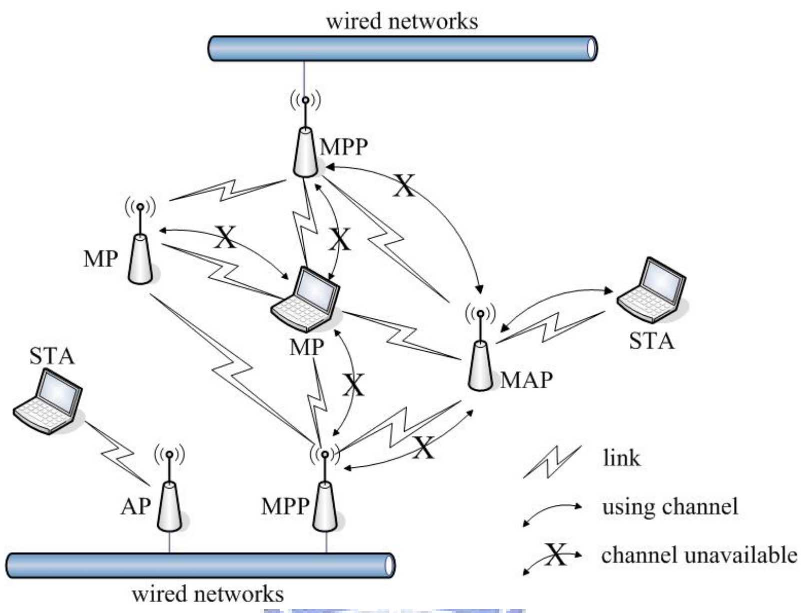

Figure 1 Architecture of the IEEE 802.11s mesh networks

2.2 How to switch N radios among M channels? ($M > N$)

In a multi-channel scenario, the radios must switch among channels if there are less radios. However, some problems caused by channel switching are given as examples shown in Figure 2.

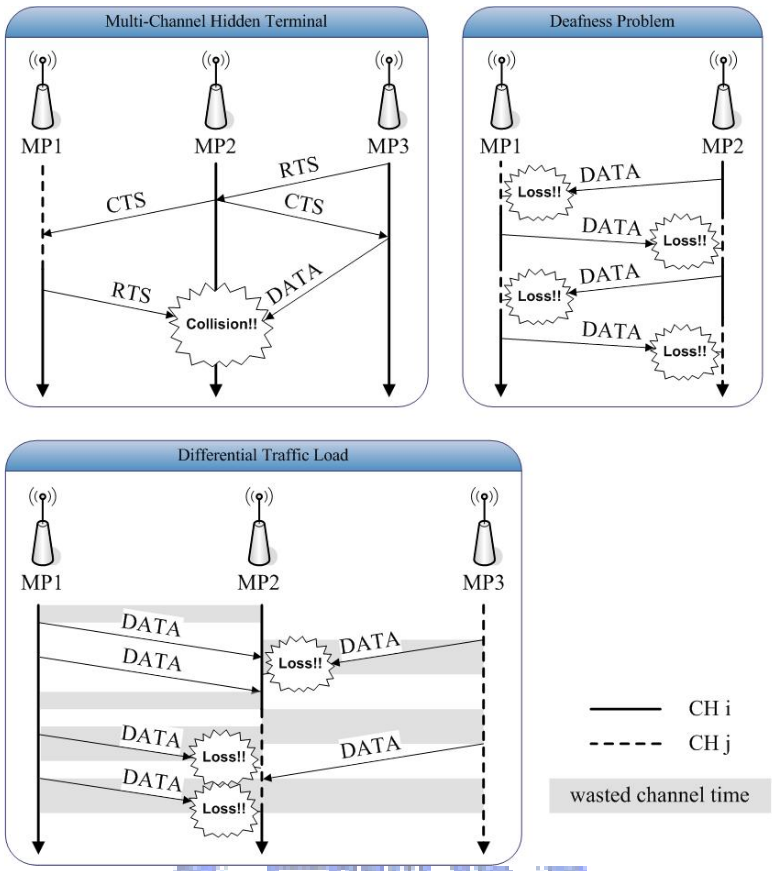

Figure 2 Examples of the problem at channel switching

Switching among channels causes packet loss in the unattended channel because no radio stays in. This is one kind of the deafness problems [13] [14] [15]. Furthermore, when the switch node returns to a channel and intends to transmit data, a collision might happen due to the channel is occupied by the other transmission and the switch node does not update its NAV. This is one kind of the multi-channel hidden terminal problems [16]. Furthermore, packet loss results in link disassociation and more overhead in the infrastructure BSS and the mesh networks. In the mesh networks, dropping mesh forwarding traffic renders all consumed resource as being wasted, especially for a long forwarding path. Moreover, link disassociation might cause that the network topology is destroyed and rebuilt. In a infrastructure BSS, a STA might be isolated or associate with another AP because of link disassociation. When the switch node returns, STAs might re-associate with the switch node due to a better link quality. The radio switches frequently might exhaust the resource of the networks. Therefore, to minimize the above overhead, it is necessary to notify STAs and neighbor MPs before leaving the channel to avoid packet loss.

Using shorter channel switching interval is another intuitive idea to reduce the damage of the deafness problem. However, high switching frequency causes that channel switching overhead becomes quite heavier and the multi-channel hidden terminal problem becomes more serious. On the other hand, using longer channel switching interval could decrease switching frequency. Nevertheless, allocating a long period for a low traffic load channel is inefficient in channel utilization. A suitable switching interval depends on channel switching overhead and traffic load. Therefore, in the multi-channel environments, it is an important issue that how to switch radios among channels sufficiently in the IEEE 802.11s mesh networks.

Chapter 3 Time Ratio Aware Switching Superintendent (TRASS)

The proposed time ratio aware switching superintendent (TRASS) not only controls radios switching among channels but also notifies neighbor nodes to avoid packet loss before the switch node leaves the channel. It includes two functions, the TRASS mechanisms compatible with the standards and the TRASS algorithm. The TRASS mechanisms announce that the switch node will leave some channel and notify its neighbors to wait and buffer for the switch node. Moreover, the TRASS algorithm selects which channels to stay in and allocates time for these channels.

3.1 Concepts of TRASS

TRASS deals with two important issues of channel switching in a multi-channel mesh network, that is, how to notify the neighbors of the switch node and how to switch radios among channels sufficiently. First, to avoid the deafness problem without needing timer synchronization and be compatible with IEEE 802.11, we propose the mechanisms existing in IEEE 802.11 to ensure frame delivery to the switch node. If the switch node is a MP, it uses the power-saving mechanism to notify its neighbors before leaving the channel. However, a MAP also plays the role of an AP which cannot enter its power-saving mode (PSM) [2]. The HCCA mechanism [11] is used for the infrastructure BSS if the switch node is a MAP. However, when the switch node adopts the HCCA or the power-saving, all nodes which coordinate with the switch node must support this mechanism. Therefore, the CTS-to-self mechanism [12] is used when there is any node which does not support the HCCA or the power-saving. Nevertheless, the leak of using the CTS-to-self is that the other nodes in the neighborhood cannot access the medium because of carrier sensing. Moreover, the occupied duration by the NAV is limited and might not be long enough.

The second issue can be divided into two parts, that is, how to select the next channel to stay in and how to allocate time for the channel. An intuitive idea is to select the next channel and to allocate time for the channel according to traffic load for the switch node. However, low traffic load for the switch node might result from a short duration in a congested channel. Moreover, the deafness problem still exists if there are more than two switch nodes in the neighborhood. A lot of buffered data might remain in the switch node because there is no chance to transmit or failed transmission due to either a bad link quality or the deafness problem. Therefore, traffic load for the switch node is treated as both the processed data and the buffered data Moreover, it adopts traffic load ratio for the switch node to correspond to the urgency of each channel. In view of the above, TRASS selects the channel with the highest traffic load ratio for the switch node to stay in. Moreover, allocating suitable duration for the selected channel can minimize waste of channel utilization and reduce channel switching frequency. Therefore, it not only estimates traffic load for the switch node and for the other nodes respectively but also adjusts total traffic load ratio to a fixed appropriate ratio.

3.2 TRASS mechanisms compatible with IEEE 802.11

3.2.1 Overview of the TRASS mechanisms in IEEE 802.11

TRASS uses three mechanisms existing in IEEE 802.11 and its extensions. The first one is the power-saving mechanism. Because active radios consume a large amount of energy, a node could turn off radios for a while to save energy. It enters its PSM and retrieves the data after it wakes up. The second one is the HCCA mechanism, which is a contention-free medium access mechanism for the quality of service. Its concept is similar to the PCF. The HCCA interleaves the CFP and the CP to control medium access. In the CP, each node contends for the medium before transmitting. On the other hand, in the CFP, each node must wait for the AP to polling itself. The last one is the CTS-to-self mechanism. It is a protection mechanism for IEEE 802.11b [17] and IEEE 802.11g [16]. Because some modulations in IEEE 802.11g are not in IEEE 802.11b, IEEE 802.11b chipsets cannot receive and decode some IEEE 802.11g signals. Therefore, IEEE 802.11g nodes issues a CTS-to-self frame to update the NAV before transmitting. This avoids the interference of IEEE 802.11b nodes by occupying the channel.

3.2.2 A state transition diagram of a switch node in a channel

Without loss of generality, let us consider a switch node which has N radios is in M channels environments. The channel number i is denoted by Ch_i. $Time_i$ is the duration allocated for Ch_i. The state machine running at a switch node for each channel is shown in Figure 3 and the state transitions are described below.

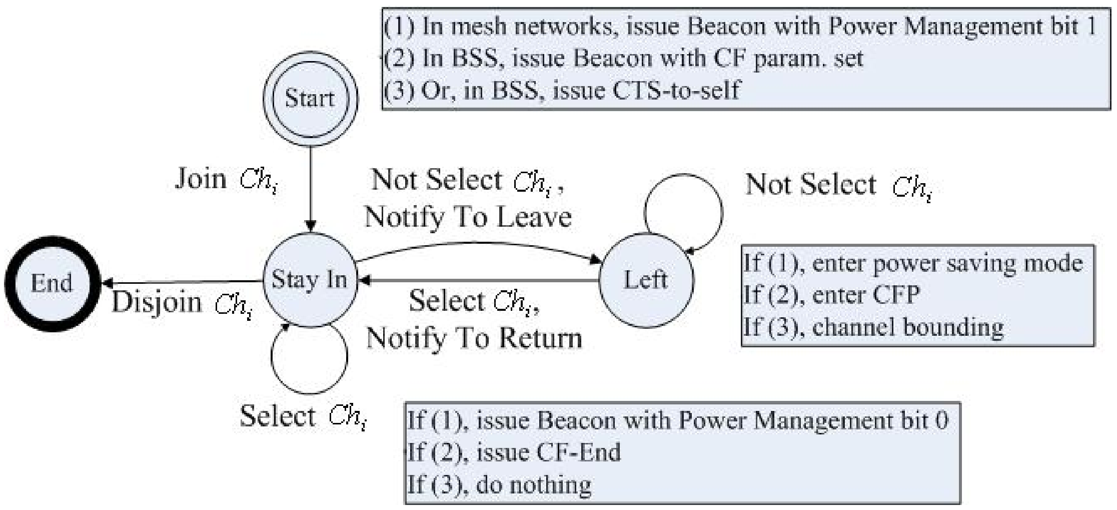
(1) In mesh networks, issue Beacon with Power Management bit 1
(2) In BSS, issue Beacon with CF param. set
(3) Or, in BSS, issue CTS-to-self
Start
Join Ch_i
Not Select Ch_i, Notify To Leave
Not Select Ch_i
End
Stay In
Left
Disjoin Ch_i
Select Ch_i, Notify To Return
If (1), enter power saving mode
If (2), enter CFP
If (3), channel bounding
Select Ch_i
If (1), issue Beacon with Power Management bit 0
If (2), issue CF-End
If (3), do nothing

Figure 3 State transition diagrams of a switch node in the Ch_i

<u>STATES</u>

- *Start*: The switch node has not enough radios to join Ch_i or it does not plan to join it.
- *Stay In*: The switch node has joined Ch_i. When $Time_i$ is over, the algorithm selects which channel this radio will stay in. If Ch_i is selected again, the switch node does nothing. Else the switch node notifies its neighbors of its leaving and then it enters state *Left*.
- *Left*: The switch node has left Ch_i. While $Time_j$ is over and j is not equal to i, the channel switching algorithm selects which channel that radio will stay in. If Ch_i is not selected, the switch node does nothing. Else the switch node returns to state *Stay In* and then it notifies its neighbors of its return.
- *End*: The switch node has disjoined Ch_i.

At first, a switch node is in state *Start*. After joining Ch_i, it transits to state *Stay In*. In state *Stay In*, the channel switch algorithm selects which channel this radio will stay in when $Time_i$ is over. If the selected channel is Ch_i, it remains in state *Stay In*. Else the switch node is going to leave and notifies its neighbors to avoid packet loss in Ch_i. In the mesh networks, it uses the power-saving to achieve this behavior. The switch node announces that it enters its PSM by sending its Beacon frame with the Power Management bit set [1] [2]. While the switch node is in its PSM, its neighbor MPs buffer the frames sent to it. In the infrastructure BSS, because an AP cannot enter its PSM, it uses the HCCA. The switch node announces that the BSS enters the CFP by sending its Beacon frame with the CF Parameter Set [1] [11]. All the STAs in this BSS must wait for the AP polling them. If the above two mechanisms are not available, the switch node issues a CTS-to-self frame and updates the NAV to occupy Ch_i to protect from the other nodes in Ch_i transmitting to the switch node. Then the switch node could safely leave Ch_i and it transits to state *Left*. In state *Left*, the channel switch algorithm is triggered to select which channel to stay in when $Time_j$ is over and j is not equal to i. If the selected channel is not Ch_i, it remains in state *Left*. Else the switch node returns to Ch_i and notifies its neighbors of its return. In the mesh networks, the switch node announces that it has woken up by sending its Beacon frame with the Power Management bit cleared. The switch node could retrieve the data buffered for it by sending a PS-Poll frame to its neighbors. In the infrastructure BSS, it announces that the BSS returns to the CP by sending a CF-End frame. For the CTS-to-self case, it resets the NAV to release Ch_i. Finally, the switch node transits to state *Stay In*.

3.2.3 Examples for the TRASS mechanisms

The concrete behaviors of the mechanisms are described by means of the following examples. Figure 4 illustrates the time series of three scenarios: two channels share one radio, three channels share one radio and three channels share two radios.

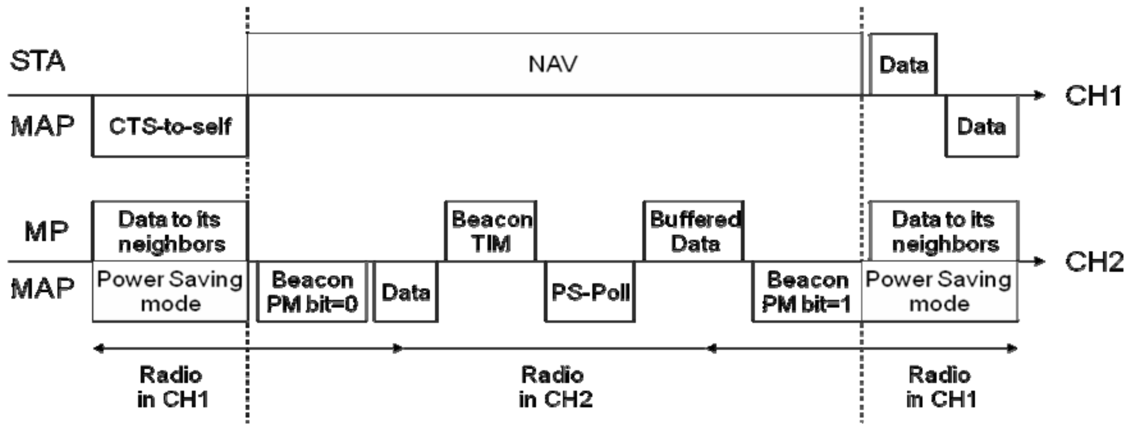

Figure 4(a) Two channels sharing one radio

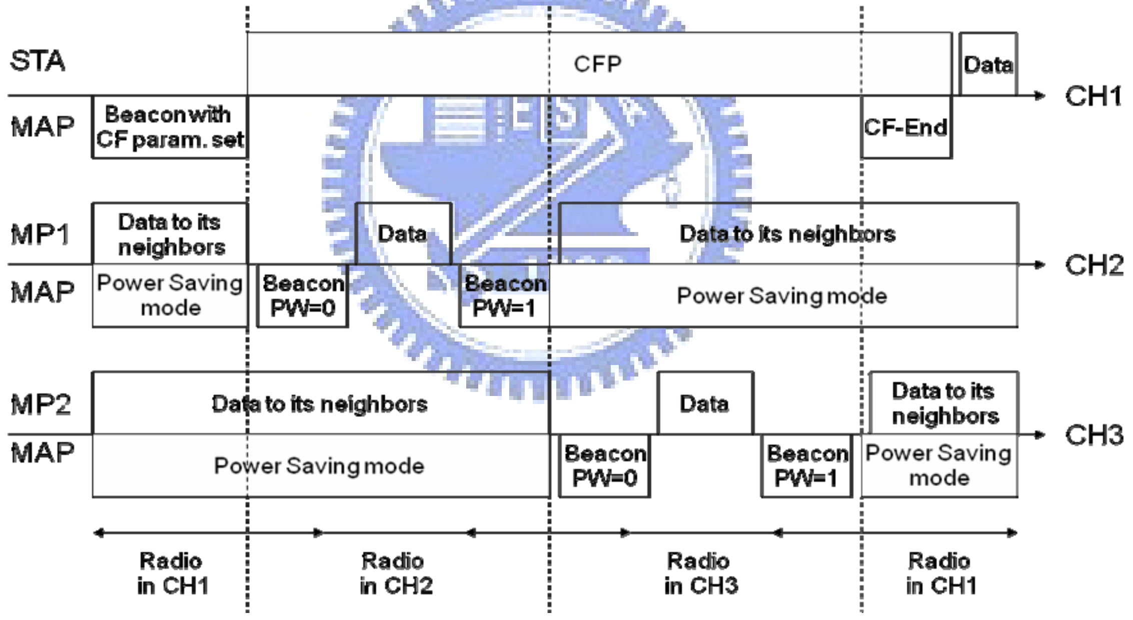

Figure 4(b) Three channels sharing one radio

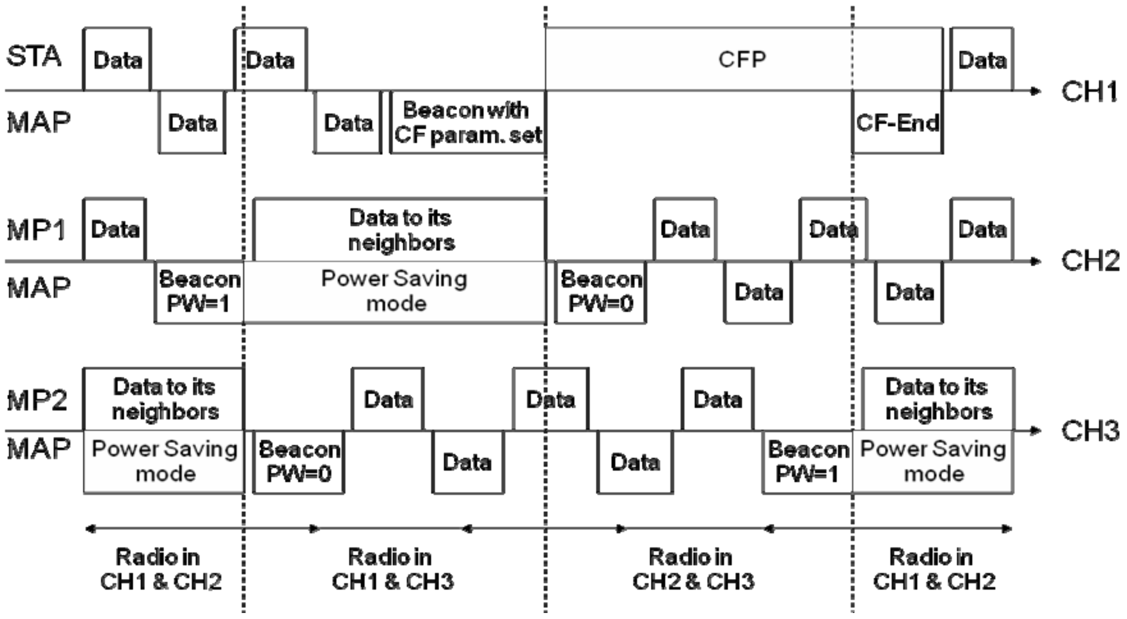

Figure 4(c) Three channels sharing two radios

To take Figure 4(a) for an example, the MAP uses the CTS-to-self in the infrastructure BSS. At first, the MAP stays in $CH1$. When the radio is ready to leave $CH1$, the MAP issues a CTS-to-self frame with the NAV set to the maximum duration. Then the MAP switches to $CH2$ and announces its return by sending its Beacon frame. If the MP has buffered data for it, the MP announces that by their Beacon frames with the TIM element. When the MAP receives such a Beacon frame, it sends a PS-Poll frame to get the data. When there is no traffic in $CH2$ or time allocated for $CH2$ is over, the radio switches to $CH1$ and resets the NAV. Then the STA can send data to the MAP. In Figure 4(b) and Figure 4(c), it uses the HCCA in the infrastructure BSS. The different between the first example and the latter two examples is that the MAP announces that the BSS enters the CFP by sending a Beacon frame with the CF Parameter set and then it switches to $CH2$. When the MAP returns, it announces that the BSS enters the CP by sending a CF-End frame.

3.3 TRASS algorithm

3.3.1 Concepts of the TRASS algorithm

TRASS not only assigns each radio to stay in the scheduled channels but also allocates time for the channel. First, the algorithm selects which channel to stay in according to traffic load ratio for the switch node. Traffic load for the switch node is defined as the amount of time which the switch node occupies the channel. The amount of time for the buffered data which is estimated by the processed data is also accumulated to traffic load to deal with the deafness problem. Furthermore, Traffic load ratio for the switch node is defined as the ratio of the active period to traffic load. Then the algorithm selects the channels with the highest traffic load ratios of for the switch node. Moreover, an aging mechanism is adopted to avoid starvation. Second, the algorithm allocates time for the selected channel according to total traffic load ratio of the channel which is defined as the ratio of the active period to total traffic load. It is assumed that traffic load for the other nodes retain a fixed ratio and traffic load for the switch node is estimated according to the duration in state $Left$. Finally, the algorithm eliminates wasted time by allocating the suitable duration to retain an appropriate total traffic load ratio.

3.3.2 Selecting the channels with the highest traffic load ratio for the switch node

Traffic load is often defined as the number of packets or the amount of data. However, occupied time of a channel affects channel utilization immediately in IEEE 802.11. Therefore, traffic load is defined as the amount of time to process all the data which includes the buffered data, the processed data. Nevertheless, the higher traffic load might not represent the higher urgency. In a congested channel, low traffic load might result from a short duration in state $Stay\ In$. As a result, the algorithm selects the channel which has the highest traffic load ratio for the switch node. A scenario that the switch node leaves some channels for Ch_i is taken for an example to explain the details of the algorithm. At first, the variables used by this algorithm are defined in the following.

$\forall Ch_i$,

$T_{Stay\ In}(i,j)$ is the j-th duration in state $Stay\ In$. It assigns $T_{Stay\ In}(i,j)$ to the duration to complete one basic operation if $T_{Stay\ In}(i,j)$ is less.

$T_{Left}(i,j)$ is the duration in state $Left$ before the j-th in state $Stay\ In$.

$T_{SN}(i,j)$ is occupied time of Ch_i for the switch node during $T_{Stay\ In}(i,j)$

$T_{Other}(i,j)$ is occupied time of Ch_i for the other nodes during $T_{Stay\ In}(i,j)$

$D_{Buffered}(i,j)$ is the amount of buffered data for Ch_i after $T_{Stay\ In}(i,j)$ is over

$D_{Done}(i,j)$ is the amount of processed data for Ch_i during the j-th period in state $Stay\ In$. $D_{Done}(i,j)$ is assigned to the minimum frame size if $D_{Done}(i,j)$ is equal to zero.

When $T_{Stay\ In}(i,j\text{-}1)$ has passed, the algorithm selects the next channel with the greatest $Weight_i$ which is defined as

$$Weight_i = \frac{T_{SN}(i,j\text{-}1)}{T_{Stay\ In}(i,j\text{-}1)} \times \left(1 + \frac{D_{Buffered}(i,j\text{-}1)}{D_{Done}(i,j\text{-}1)}\right) + T_{Left}(i,j) \times \alpha_i \tag{1}$$

where α_i is the aging rate of Ch_i. $Weight_i$ reflects two viewpoints, that is, how congested Ch_i is and how long Ch_i stays in state $Left$. $\frac{T_{SN}(i,j\text{-}1)}{T_{Stay\ In}(i,j\text{-}1)}$ is the traffic load ratio for $D_{Done}(i,j\text{-}1)$ and $\frac{T_{SN}(i,j\text{-}1)}{T_{Stay\ In}(i,j\text{-}1)} \times \frac{D_{Buffered}(i,j\text{-}1)}{D_{Done}(i,j\text{-}1)}$ is the traffic load ratio for

$D_{Buffered}(i,j)$ estimated by the ratio of $D_{Done}(i,j\text{-}1)$ over $D_{Buffered}(i,j\text{-}1)$. α_i is a factor to control the aging rate of Ch_i for different conditions. For example, if the TRASS mechanism for Ch_i can only tolerates a short duration in state $Left$, α_i shouldl be assigned to a great value.

3.3.3 Estimating the amount of time by the total traffic load ratio

After selecting Ch_i, the algorithm estimates $T_{Stay\ In}(i,j)$ according to its total traffic load ratio. We assume that the longer period the switch node leaves Ch_i, the more data its neighbor nodes buffer for it in Ch_i and the other nodes occupy an approximate ratio of channel utilization. That is, $T_{SN}(i,j)$ is in direct proportion to $T_{Left}(i,j)$ and $T_{Other}(i,j)$ is in direct proportion to $T_{Stay\ In}(i,j)$. The equations of $T_{SN}(i,j)$ and $T_{Other}(i,j)$ are

$$T_{SN}(i,j) = \frac{T_{SN}(i,j\text{-}1) \times T_{Left}(i,j)}{T_{Left}(i,j\text{-}1)} \times \left(1 + \frac{D_{Buffered}(i,j\text{-}1)}{D_{Done}(i,j\text{-}1)}\right) \tag{2}$$

and

$$T_{Other}(i,j) = \frac{T_{Other}(i,j\text{-}1) \times T_{Stay\ In}(i,j)}{T_{Stay\ In}(i,j\text{-}1)}.$$

$T_{SN}(i,j\text{-}1)$ is the traffic load for $D_{Done}(i,j\text{-}1)$ and $T_{SN}(i,j\text{-}1) \times \frac{D_{Buffered}(i,j\text{-}1)}{D_{Done}(i,j\text{-}1)}$ is the traffic load for $D_{Buffered}(i,j)$ estimated by the ratio of $D_{Done}(i,j\text{-}1)$ over $D_{Buffered}(i,j\text{-}1)$. The algorithm retains the total traffic load ratio to an appropriate ratio by allocating the suitable duration. The equation is written as follows

$$\frac{T_{SN}(i,j) + T_{Other}(i,j)}{T_{Stay\ In}(i,j)} = \beta_i,$$

where $\frac{T_{SN}(i,j)+T_{Other}(i,j)}{T_{Stay\ In}(i,j)}$ is the total traffic load ratio, β_i which is a pre-determined channel utilization ratio for Ch_i and $T_{Stay\ In}(i,j)$ is the j-th duariotn in state $Stay\ In$ for Ch_i which is the only unknown variable. After solving the above equations, we get

$$T_{Stay\ In}(i,j)=\frac{T_{SN}(i,j)}{\beta_i-\frac{T_{Other}(i,j\text{-}1)}{T_{Stay\ In}(i,j\text{-}1)}}. \tag{3}$$

There are two special cases for $T_{Stay\ In}(i,j)$. The first one is $\frac{T_{Other}(i,j\text{-}1)}{T_{Stay\ In}(i,j\text{-}1)}\geq\beta_i$. That means Ch_i is too congested to achieve the appropriate ratio for such a case. Therefore, it assigns $T_{Stay\ In}(i,j)$ to $T_{Stay\ In}(i,j\text{-}1)$. Another case is that total traffic load ratio is always greater than β_i and it might result from either $T_{Other}(i,j)$ or $T_{SN}(i,j)$ is too large. For the $T_{Other}(i,j)$ case, $T_{Stay\ In}(i,j)$ is bounded by the first special case. For the $T_{SN}(i,j)$ case, $T_{Stay\ In}(i,j)$ might grow up unrestrictedly. If there are enough radios to stay in each congested channel, $T_{Stay\ In}(i,j)$ becomes shorter than $T_{Stay\ In}(i,j\text{-}1)$ because Ch_i is selected earlier and $T_{Left}(i,j)$ is shorter. By the Pigeon-Hole principle, there exists at least one unattended channel if there are more than N such channels. Time allocated for the unattended channel will be more because its congestion and waiting time. However, it results in allocating more time for the next unattended channel because of the longer waiting time. The vicious spiral causes $T_{Stay\ In}(i,j)$ to grow up unrestrictedly. To avoid the problem, the algorithm assigns $T_{Stay\ In}(i,j)$ to $T_{Stay\ In}(i,j\text{-}1)$ if $\frac{T_{SN}(i,j\text{-}2)+T_{Other}(i,j\text{-}2)}{T_{Stay\ In}(i,j\text{-}2)}\geq\beta_i$ and $\frac{T_{SN}(i,j\text{-}1)+T_{Other}(i,j\text{-}1)}{T_{Stay\ In}(i,j\text{-}1)}\geq\beta_i$.

3.3.4 An example of the TRASS algorithm

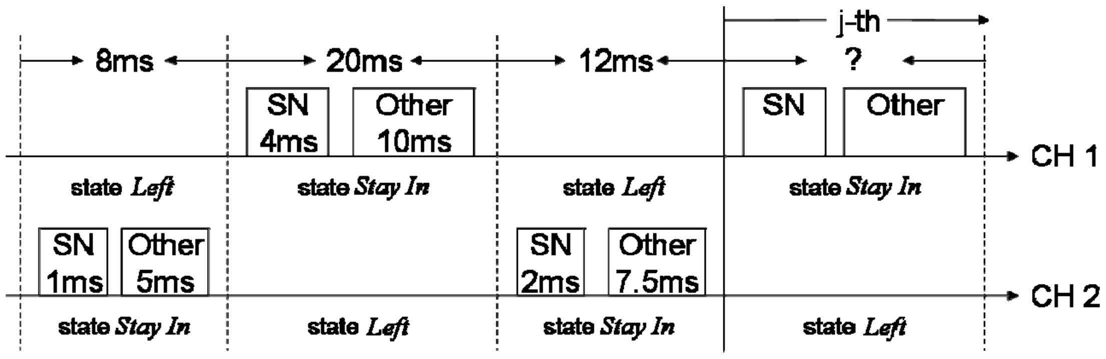

Figure 5 Time series of an example of the algorithm

A scenario shown in Figure 5 is taken for an example to explain the details. There are two channels sharing with one radio. It is assumed that $\alpha_1 = \alpha_2 = \frac{1}{100}$, $\beta_1 = \beta_2 = 0.9$ and no buffered data for each of the channels. TRASS runs at the time which is marked by the solid line. First, it calculates $Weight_1$ and $Weight_2$ by equation (1). The process of calculation is in the following:

$$T_{SN}(1,j\text{-}1) = 4\,,\ T_{Stay\ In}(1,j\text{-}1) = 20\,,\ T_{Left}(1,j) = 12$$

$$\Rightarrow Weight_1 = \frac{4}{20} + 12 \times \frac{1}{100} = 0.32$$

$$T_{SN}(2,j\text{-}1) = 2\,,\ T_{Stay\ In}(2,j\text{-}1) = 12\,,\ T_{Left}(2,j) = 0$$

$$\Rightarrow Weight_2 = \frac{2}{12} + 0 \times \frac{1}{100} = 0.167$$

Because $Weight_1 > Weight_2$, the selected channel is $CH1$. Second, it estimates $T_{SN}(1,j)$ by equation (2) and $T_{Stay\ In}(1,j)$ by equation (3). The process of calculation is in the following:

$$T_{SN}(1,j) = \frac{4 \times 12}{8} = 6ms\,,\ T_{Other}(1,j\text{-}1) = 10ms\,,\ T_{Stay\ In}(1,j\text{-}1) = 20ms$$

$$\Rightarrow T_{Stay\ In}(1,j) = \frac{6}{0.9 - \frac{10}{20}} = 15ms$$

Chapter 4 Implementation

TRASS is implemented on the Linux driver registers multiple virtual interfaces to upper layers for the infrastructure BSS and the mesh networks to the Linux kernel. Moreover, the multi-channel transmission is completed above all and then the behaviors of the power-saving and the HCCA are implemented at the WLAN driver. Finally, the TRASS algorithm is implemented in a switching timer which controls radio switching and the whole behaviors.

4.1 System architecture of implementations

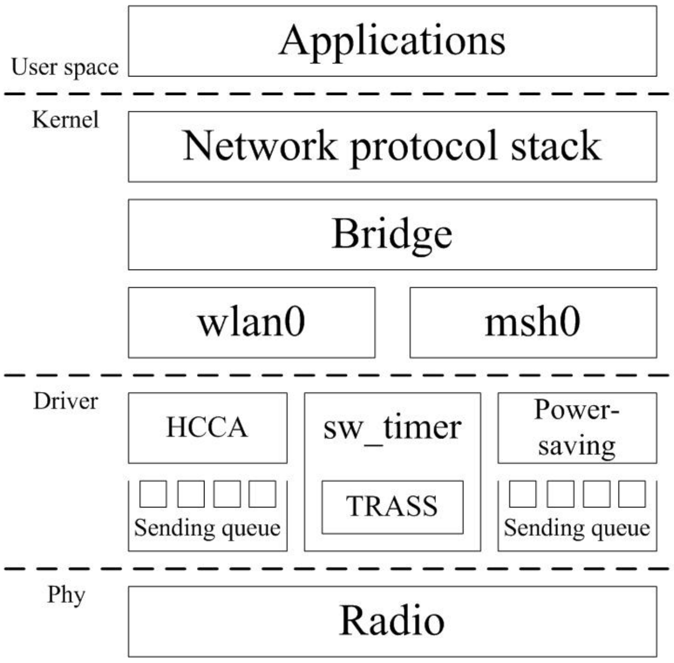

Figure 6 System architecture of implementation on RTL8186

To verify the feasibility of TRASS, the implementation was realized on the RTL8186 AP system of the company Realtek. RTL8186 is a commercial product which is a highly integrated System-on-a-Chip with embedded Linux supporting IEEE 802.11a/b/g and RTL8186 equips with a single radio. Moreover, the project of Realtek-NCTU Joint Research Center is developed the mesh networking on RTL8186 and follows the IEEE 802.11s draft revision 1.0. Therefore, RTL8186 can play the role of an STA, an AP, an MP and an MAP. The architecture of our implementation on RTL8186 is shown in Figure 6. We purely implemented TRASS at the driver level because channel switching is a monotonous, recursive and frequent behavior.

From abstract view, the mesh network interface and the infrastructure BSS interface of a MAP should be separated. because individual interface provided to the upper layers and the applications is more modular and flexible. Afterwards, a Linux timer is registered to control radios switching among channels. The handler of the switching timer being the core of TRASS manages the behaviors of the TRASS mechanisms and embeds the TRASS algorithm in itself. Thereafter, we have to complete the multi-channel transmission by modifying the data path of the original IEEE 802.11 driver. In a single-channel scenario, the driver immediately transmits or forwards data to current channel if the medium is available. On the other hand, in a multi-channel scenario, it needs to know which channel the destination stays in and it has to switch to the channel before transmitting. However, the multicast data and the broadcast data results in more channel switching overhead because it needs to switch to multiple channels for transmitting each frame. Therefore, we supply a sending queue for each channel and dispatch a frame into the sending queue of the corresponding channel if the current channel does not correspond. Furthermore, the multicast data and the broadcast data are cloned as multiple frames and are queued up into the corresponding sending queues of these channels. When the switch node stays in some channels, it processes the frames in the sending queue of the channel and then transmits them if the medium is available.

4.2 TRASS mechanisms compatibles with IEEE 802.11

Because the HCCA and the power-saving were not implemented on our platform, our works include implementation of the TRASS mechanisms on the platform.

However, TRASS only requires their behaviors to notify its neighbors and buffer for the switch node. Therefore, the necessary frames for notification and the handler for receiving these frames are implemented but the actual internal processing for the power-saving and the HCCA such as power consumption and the quality of service are not implemented.

First, the notification frames and their handlers are implemented. However, most platforms issue the Beacon frame and the control frames by the hardware. We cannot set our information elements into those frames easily. Thus, the driver is modified to provide a software Beacon frame to carry the necessary information elements for the HCCA and the power-saving and to trigger the hardware to send a CF-End frame. Moreover, a flag to indicate whether this node is in the CFP or the power-saving mode is added and the corresponding handler sets or clears the flag while a STA or a MP has received the frames which is described in chapter 3. Second, STAs should buffer the data in the CFP until they are polled by the AP or return to the CP. Thus, each STA need a sending queue for buffering data in the CFP. On the other hand, MPs should buffer the data for all neighbor MPs which are in their power-saving mode. Thus, each MP needs the number of sending queues as many as the number of its neighbor MPs. We implement the sending queue to record the pointer of each incomplete frame and modify the transmission data path to queue up frames. The buffered frames will be processed while the flag is modified by the corresponding handler of the Beacon frame with the power management bit set and the CF-End frame.

4.3 TRASS algorithm

The time ratio aware channel switching algorithm is implemented in the switching timer handler which is registered by the driver. It decides which channel to stay in and computes the length of the period to stay in. First, we have to log the necessary parameters for our algorithm. Because the WLAN driver can capture all the frames in current $channel_i$, we record the duration filed and data length of the frames in $channel_i$ to get $T_{SN}(i,j)$, $T_{Other}(i,j)$ and $D_{Done}(i,j)$. On the receiving side, the switch node filters the frames according to the address 1 field and BSSID. If the frames need to be handled by the switch node, we accumulate the duration filed to $T_{SN}(i,j)$ and accumulate the packet length to $D_{Done}(i,j)$. Otherwise, we accumulate the duration filed to $T_{Other}(i,j)$. On the transmitting side, we check whether the destination node of the frame stays in current channel before transmitting. If the destination node is in the other channel, the frame is queued up and accumulates the packet length to $D_{Not\ Yet}(i,j)$. Otherwise, the frame is transmitted and accumulate the duration filed to $T_{SN}(i,j)$. In addition, α_i, β_i, and $MinTime$ are the experience values and given at compile-time. We observe the maximum channel utilization of current channel to adjust α_i and β_i to reasonable values. $MinTime$ is adjusted with the channel switching overhead and the Linux timer precision on the embedded platform. Afterwards, the algorithm computes $Weight$ for each channel with the equation (1) to select the next channel to stay in while the switching timer is triggered. After determining the next channel which is labeled as $channel_k$, the algorithm computes $T_{Stay\ In}(k,j)$ with the equation (3) and accumulate $T_{Stay\ In}(k,j)$ to $T_{Left}(i,j)$ where $i \neq k$ and modifies the timer with $T_{Stay\ In}(k,j)$. Finally, the radio switches to the selected channel and the driver processes the frames in the sending queue for the channel.

Chapter 5 Evaluation Results

The implemented-based evaluation results show the improvements of the two-channel one-radio scenario. In addition, the multi-channel multi-radio scenarios are simulated with the same configurations in practice. The implementation is used to measure the improvements of TRASS and the effects of the TRASS mechanisms and the TRASS algorithm are separately discussed. On the contrary, simulation-based study is used to observe the throughput variation for each pair (M, N) where M channels sharing N radios.

5.1 Benchmark result of implementation

The experiment environment is that the switch node plays the role of a MAP and another two devices play the role of a STA and a MP respectively. The STA and the MP stay in different channels and generate differential traffic to the MAP by the Linux socket programs with the client-server model. The client programs which can generate the TCP traffic and the UDP traffic are implemented on the STA and the MP. At the application layer, the TCP traffic is controlled by the select function to guarantee that the transmission is successful. On the other hand, the UDP traffic adopts the transmit-and-pray strategy. The client application can generate a random packet size within a configurable range and control the packet generation rate with a configurable random back-off window. Furthermore, the server programs runs on the MAP and it plays the role of a echo server for the TCP traffic but only receives the UDP traffic without responding to the client.

We have implement three channel switching algorithms, which is the fixed time interval algorithm (FTI), the packet ratio aware algorithm (PRA) [10] and the proposed time ratio aware algorithm (TRA). We only implement is algorithm of PRA which is proposed by [10] but the parameters of all algorithms are configured according to our practical system. All of the three algorithms use the TRASS mechanisms. FTI without buffers is also implemented to observe the effects of the TRASS mechanisms because the behavior of FTI is the simplest. For FTI, the switch node switches among channels in a fixed time interval in sequence. For PRA, the switch node switches to each channel in sequence during the switching cycle and divides the switching cycle into M parts where M is the number of channels. PRA divides the amount of time for each channel according to the packets per second in the channel during the last period. For TRA, it switches among channels in two phases which are selecting to channel with $Weight$ and allocating time for the selected channel according to the time-based traffic load ratio. Since α_i, β_i, and $MinTime$ need to be pre-determined at compile time, they are determined as follows. We observe the average channel utilization to determine β_i. Moreover, we determine α_i and $MinTime$ according to the precision of the Linux timer and the channel switching overhead. The FTP application is used to probe the maximum channel utilization and the Omnipeek application is used to monitor the wireless media to measure the average channel utilization. .

Table 1 Benchmark parameters

Algorithm	Parameter	Value
Fixed time interval	Switching interval	150 (TUs)
	Buffer size	8 (packets)
Packet ratio aware	Switching cycle	300 (TUs)
	$MinTime$	10 (TUs)

	Buffer size	8 (packets)
Time ratio aware	α	0.0001
	β	1 / 32
	$MinTime$	10 (TUs)
	Buffer size	8 (packets)
Traffic	Back-off window size	1/8、1/16、1/32、1/64、1/128 (s)
	Packet size range	1500 (Bytes)

Finally, the observed average channel utilization is about 5% and then β_i is determined as 3.125%. Furthermore, the average channel switching overhead is in the scope of 4~8 ms and the minimum unit is 10 ms in the Linux timer. To reduce the effects of the system restrictions, we enlarge the scales of time unit (TU) to 10ms. Thus, $MinTime$ is determined as 100 ms and α_i is determined as 0.0001. The parameters used in the benchmark are shown in table 1.

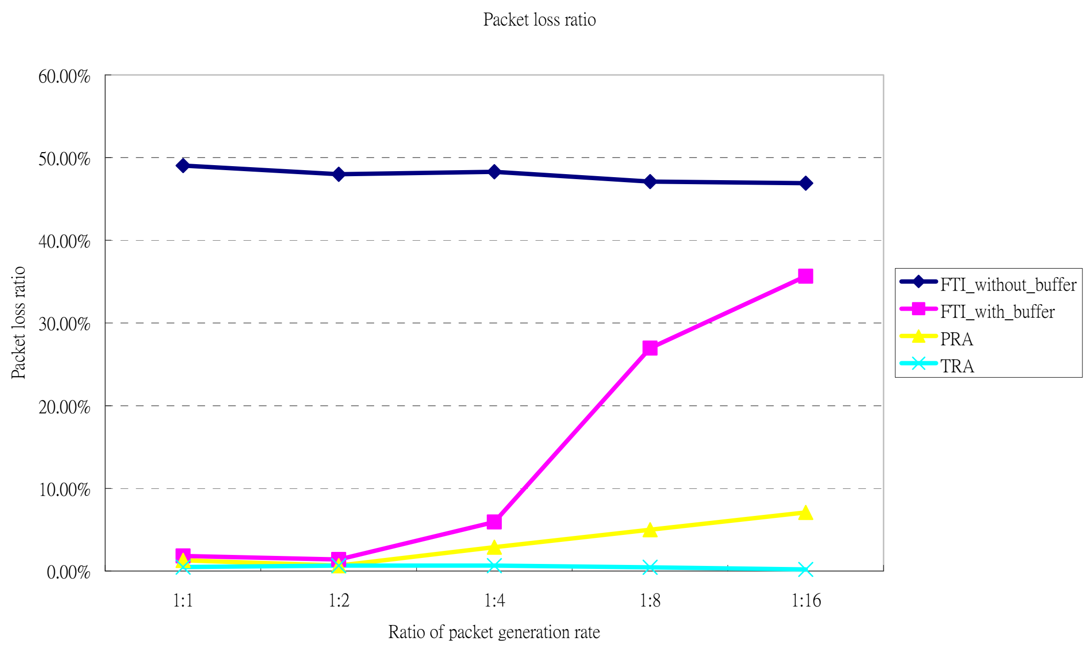

Figure 7 Packet loss ratio of benchmark results

The original problem which is caused by channel switching is the deafness problem which is described in chapter 2. It results in packet loss and then affects other deterioration. Therefore, we observer the packet loss ratio of each algorithms in differential traffic load which is adjusted with the different range of their random back-off window. The ratio of packet generation rate is the ratio of the random back-off window of the MP to the one of the STA. We fix the random back-off window of the STA and reduce the random back-off window of the MP to a half every time. Furthermore, the client generates the UDP traffic and records the amount of packets it has transmitted. The server also records the amount of packets it has received. Thus, we could get the packet loss ratio and the result is shown as figure 7. The packet loss ratio of FTI without buffer is close to 50% because the MAP switches back-and-forth with the same time interval. That is, about a half of packets are transmitted while the MAP leaves. On the other hand, the packet loss ratios of the other algorithms with buffers are close to 0% while the ratio of packet generation rate is lower. The reason why the packet loss ratio is not 0% is that some packets have been passed to the hardware and the hardware has not transmitted while the MAP switches to another channel. Since we cannot to control the hardware buffer, those packets cannot be buffered by either the HCCA or the power-saving mode and are still transmitted by the hardware. Furthermore, the packet loss ratios of the algorithms with buffers are increasing while the ratio of packet generation rate becomes higher. It results from buffer overflow because the channel switching algorithm is not corresponding to traffic load enough. Unlike the other algorithm, The packet loss ratio of the TRA is not increasing since it selects the next channel with $Weight$. That is, the congested channel might be selected twice if its congestion has not been eased. If the switch cycle of PRA is too long, it results in switching to the low traffic channel for a while and this might compresses the amount of time in the congested channel. Based on the results, the most improvement of the packet loss is contributed by the TRASS mechanisms and the channel switching algorithms contribute to avoid buffer overflow if the buffer size is too small or traffic load is too heavy.

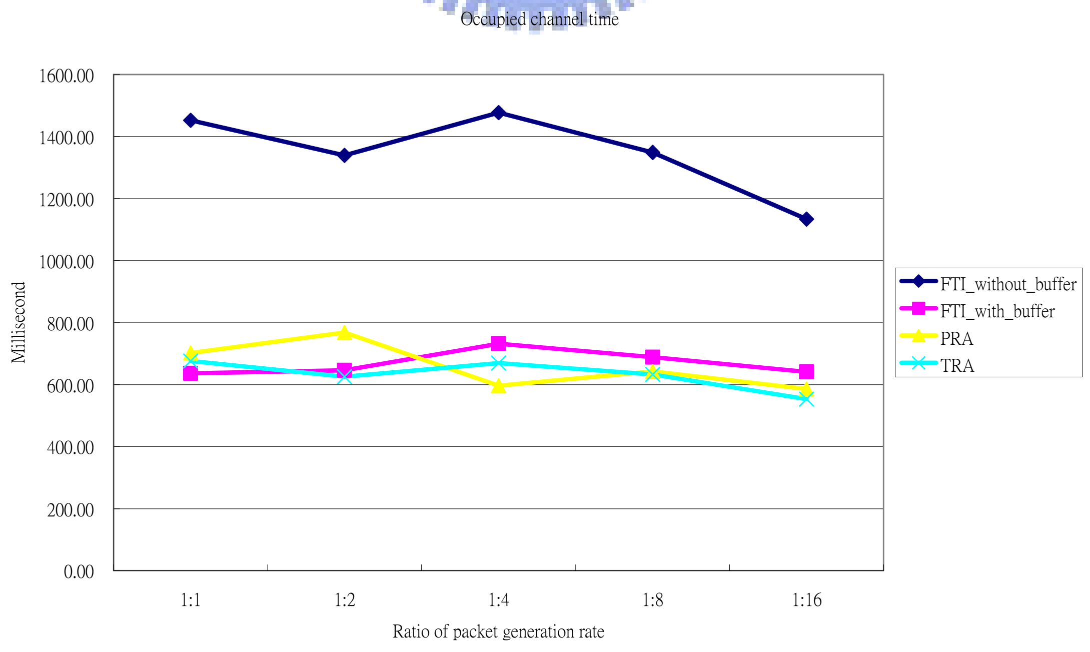

Figure 8 Occupied channel time of benchmark results

While TRASS could ease the packet loss which is the original problem resulting from channel switching, it reduces the overhead of the redundancy processing of transmission. Afterwards, we observer what the improvement contributes to the channel utilization which is an important issue of the mesh networks. Traffic load in the mesh networks might be heavy because data forwarding occupied a multiple amount of channel time per frame. Therefore, we measure the occupied channel time by means of that the client transmits a fixed amount of the TCP traffic to the server and we accumulate the occupied channel time for completing these transmission. The less retransmission, the less channel time it wastes. We use another device to capture all frames which is transmitted to or from the MP and it accumulates the duration filed of the frames. Thus, we can measure the occupied channel time for completing the transmission and the result is shown as figure 8. The occupied channel time of FTI without buffers is much greater than the other methods which with buffers. The reason is that the driver often retransmits because the MAP stay in another channel and then it drops the frames. Dropping frames results in wait the TCP retransmission for a long time and the retransmitted frames occupy the channel again. On the other hand, the difference of occupied channel time between the other algorithms with buffers is small because each frame is buffered until the MAP return to this channel. This minimizes the retransmission overhead. Nevertheless, buffering overflow might happen if packet generation rate is too high to buffer all transmitted frames but the result does not represent the appearance. It is due to TCP congestion control and the socket buffer is full. The frames buffered by the driver are treated as incomplete packet by the TCP and then the TCP will reduce the bandwidth it allocates. Moreover, the client application holds by the selection function and stops to transmit data until it can transmit data through the socket again.

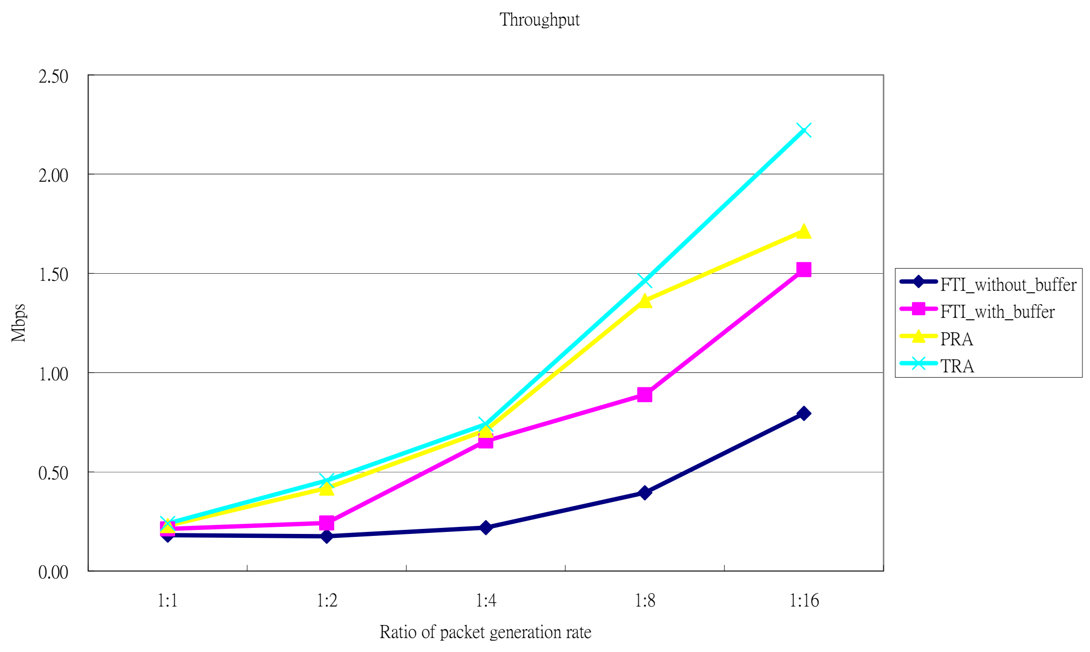

Figure 9 Throughput of benchmark results

However, the result of occupied channel time cannot discover the improvement of the different channel switching algorithm. Thus, we measure the throughput to compare with these methods and the result is shown as figure 9. The higher throughput, the less amount of time elapses for completing the transaction. While packet generation rate becomes higher, there is not enough time to transmit the data and need to buffer the data. As a result, the congested channel needs more time and the channel switching algorithm has to consider about both the buffered data and the transmitted or received data. However, PRA only records the amount of packets without considering the packet size. Moreover, PRA does not consider the buffered packets and the transmitted or received packets at the same time. These might affect the forecast of traffic load. Therefore, the TRA shows a better throughput than the other methods. Based on the analyses above, the most improvement of occupied channel time is contributed by the TRASS mechanisms and the throughput is contributed by both the TRASS mechanisms and the channel switching algorithms. However, the improvement of throughput is not obvious because the TCP congestion control decreasing the required bandwidth while either the packet is loss or buffer by the driver.

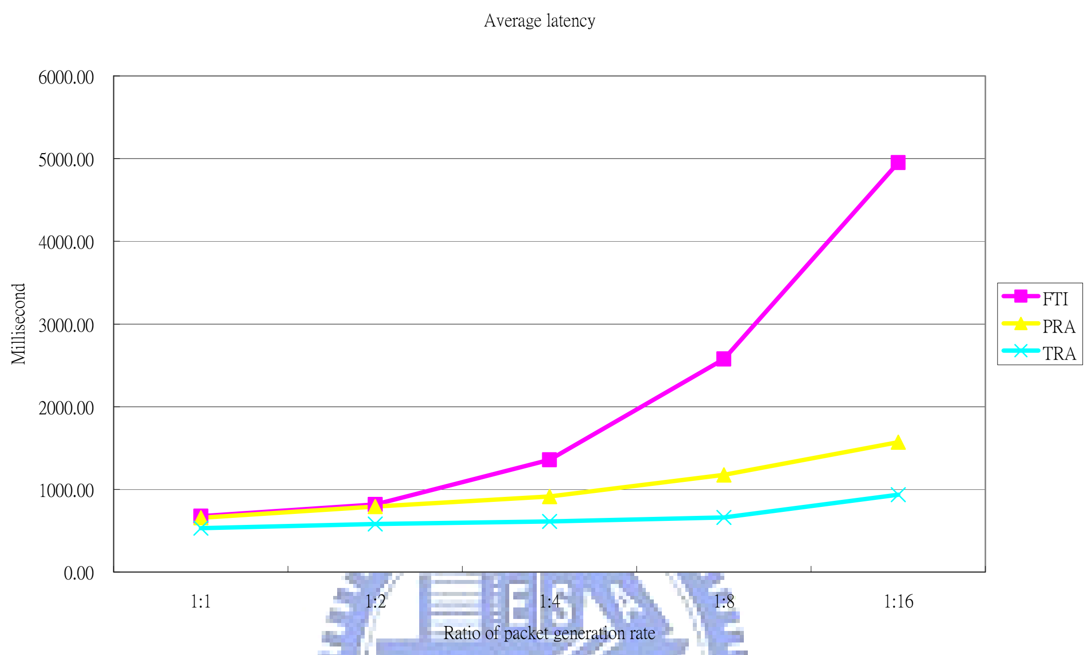

Figure 10 Latency of benchmark results

Finally, we measure the latency of the algorithms with buffers except FTI without buffers because the scale of the latency without buffers is too large due to link disassociation happening and the frequent TCP retransmission. While packet generation rate is low, the TRA is better than the others as a result of the amount of time is not bounded. On the other hand, the amount of time allocated by FTI is fixed and the amount of time allocated by PRA is bounded by the switching cycle. They might allocate redundancy time for some channel and increase the latency of the other channels. Furthermore, the latency of all the algorithms is increasing while packet generation rate becomes higher. It results from the TCP congestion control reduces the required bandwidth and then the client application stops to transmit data due to the socket buffer is full. The application keeps generating packets and the packets are blocked at the application layer. Thus, the latency of the packets is increasing quickly. Based on the results, the improvement of the latency is related to whether the amount of forecasted time is corresponding to traffic load. Furthermore, another observation is that the TRASS mechanisms are not helpful to improve the latency for the TCP traffic or its application because the TCP congestion control decreases the required bandwidth and then the socket buffer might be full. Thus, it could not avoid the latency increasing.

5.2 Simulation result

Table 2 Simulation parameters

Parameter	Value
SIFS	9 (us)
DIFS	27 (us)
Maximum contention window size	1024
Minimum contention window size	32
Contention window time slot	10 (us)
Supported transmission rate	1、2、5.5、11、12、18、24、36、48、54 (Mbps)
Retransmission algorithm	Auto Rate Fallback
Maximum frame size	2367 (Bytes)
Channel switching overhead	4~8 (ms)
Buffer size	64 (Packets)
Maximum channel utilization	25%
Packet loss probability of channel	0.04

Since there is only a single radio equipped on RTL8186, the performance with multiple radios cannot be measured out. Thus, we use simulation to observe the relationship of the throughput of each channel and the pair (M, N) where M channels sharing N radios. We reference the system parameters of RTL8186 in its data sheet [18] and then simulate channel switching in the multi-channel environment. Those parameters of RTL8186 are configured in the simulation are shown in table 2. Only Maximum channel utilization and packet loss probability of channel are the parameters of the channel needing to be pre-determined and we measure the two parameters in practice. To measure the maximum throughput of each channel, it tries to reach the maximum channel utilization. Each channel follows the DCF and generates traffic randomly. TRASS runs on the switch node to manage switching the radios among the channels. We measure the maximum throughput per channel which can be handled by the switch node and our simulation covers all the pair (M , N).

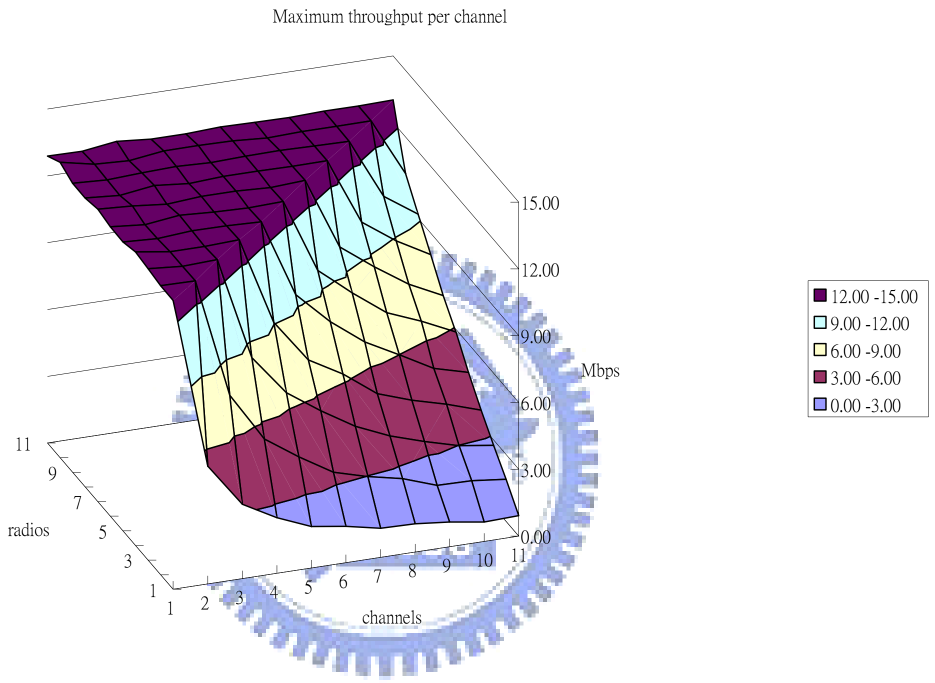

Figure 11(a) Maximum throughput per channel with (M , N)

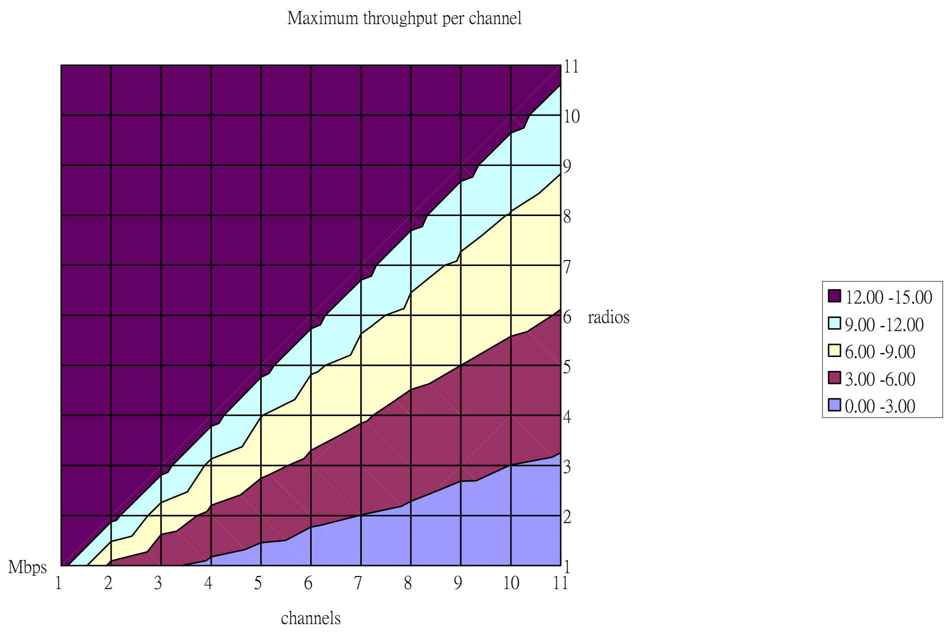

Figure 11(b) Vertical view of Figure 11

The results are shown as figure 11 and figure 12. In figure 11, the throughput decreasing violently while the number of radios is less than the number of channels, especially the number of radios is small. Because channel switching overhead is serious when the ratio of the number of channels to the number of radios is small, it results in waste of channel time and then the throughput is decreasing. Thus, channel switching is appropriate to be used in a heavy traffic channel but there is light traffic the switch node in the channel. For example, a MAP locates near the root of the mesh networks but it only associated with a few neighbors. The MAP might only forward data for the infrastructure BSS to the mesh networks. It could avoid the effects of carrier sensing and reserve more available time for both the root and the infrastructure BSS if the MAP use channel switching and multi-channel transmission. Furthermore, figure 12 is the vertical view of figure 11 and it shows the relationship between the pair (M , N) and throughput. For example, a product is designed for 5 channels and its throughput requirement is around 3~6Mbps. The vender could design a device which equips three radios to reduce the cost and to reserve power. This might be a reference for venders to design future products and to deploy networks

Chapter 6 Conclusions and Future Works

This paper aims at utilizing multiple channels with less radios in the IEEE 802.11s mesh networks and proposes TRASS to manage channel switching. TRASS minimizes the problems which result from channel switching. The TRASS mechanisms which have existed in IEEE 802.11 and its extensions are used to notify neighbors to buffering for the switch node and avoid packet loss in the unattended channels. This reduces packet loss ratio to the value less than 1% and minimizes waste channel time which results from retransmission. Moreover, the TRASS algorithm manage channel switching according to time-based traffic load ratio which considers both buffered data, transmitted data and received data. TRASS flexibly selects the channels according to *Weight* and allocating time for the selected channels resiliently. This makes TRASS more suitable for the varied traffic load in the mesh network. In addition, TRASS is realized on the commercial embedded system RTL8186 and the improvements are measured in practice. TRASS provides stable multi-channel transmission and it has been investigated into the IEEE 802.11s mesh network project of Realtek-NCTU Joint Research Center. Furthermore, we simulate PRActical environment and the parameters of RTL8186 are configured into the simulation to observe the throughput in a multi-channel multi-radio environment. The simulation results provide a reference for designing future product design and deploying in the multi-channel multi-radio environments.

This work has completed synchronization of multiple switch nodes compatible with the standard and has not discussed more applications with channel switching in the multi-channel mesh networks. For example, network topologies and channel assignments are importance issues for avoid carrier sensing and improve performance. Combining with channel switching, it can reduce the hardware restriction and is more flexible to deploy in the multi-channel mesh networks. Moreover, the multi-radio implementation has not been completed in practice. In the mesh networks, different scenarios such as a root and a MP located on the edge of the mesh networks might require different hardware resources. For the $N > M$ case, there are more varied purposes for multiple radios such as IEEE 802.11n. Finally, mathematical modeling and analyses are also important in our future researches.

The IEEE 802.11s project of Realtek-NCTU Joint Research Center will combine with IEEE 802.11n and TRASS will be integrated into the system. Therefore, the future works would be focused on implementation of multi-radio practical system and improvement of the proposed methods. Furthermore, mathematical modeling should be investigated to analyse the performance of channel switching.

References

[1] IEEE 802.11 Working Group, “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” Sep. 1999.

[2] IEEE 802.11 Working Group, “Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Amendment: ESS Mesh Networking,” Nov. 2006.

[3] Jungmin So and Nitin H. Vaidya, “Multi-channel mac for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver,” in Mobihoc, 2004.

[4] Paramvir Bahl, Ranveer Chandra, and John Dunagan, “Ssch: Slotted seeded

channel hopping for capacity improvement in ieee 802.11 ad-hoc wireless networks," in ACM Mobicom, 2004.

[5] Shih-Lin Wu, Chih-Yu Lin, Yu-Chee Tseng, and Jang-Ping Sheu, "A new multi-channel mac protocol with on-demand channel assignment for multi-hop mobile ad hoc networks," in International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN), 2000.

[6] S.-L.Wu, Y.-C. Tseng, C.-Y. Lin and J.-P. Sheu, "A Multi-Channel MAC Protocol with Power Control for Multi-Hop Mobile Ad Hoc Networks," The Computer Journal, vol. 45, pp. 101–110, 2002.

[7] N. Jain, S. Das, and A. Nasipuri, "A multichannel csma mac protocol with receiver-based channel selection for multihop wireless networks," in IEEE International Conference on Computer Communications and Networks (IC3N), October 2001.

[8] A. Nasipuri, J. Zhuang, and S.R. Das, "A multichannel csma mac protocol for multihop wireless networks," in WCNC, September 1999.

[9] A. Nasipuri and S.R. Das, "Multichannel csma with signal power-based channel selection for multihop wireless networks," in VTC, September 2000.

[10] R. Chandra, V. Bahl, and P. Bahl, "MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card," In INFOCOM, 2004.

[11] IEEE 802.11 Working Group, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements," Sep. 1999.

[12] IEEE 802.11 Working Group, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band," Jun. 2003.

[13] Jungmin So and Nitin H. Vaidya, "A Routing Protocol for Utilizing Multiple

Channels in Multi-Hop Wireless Networks with a Single Transceiver," Tech. Rep., University of Illinois at Urbana-Champaign, October 2004.

[14] Y. Li and A. M. Safwat. Efficient deafness avoidance in wireless ad hoc and sensor networks with directional antennas. In ACM PE-WASUN'05.

[15] R. Choudhury and Nitin Vaidya, "Deafness: A mac problem in ad hoc networks when using directional antennas," in IEEE ICNP, Oct 2004.

[16] J. So, N. Vaidya, Multi-channel MAC for ad hoc networks: handling multi-channel hidden terminals using a single transceiver, in: ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), May 2004, pp. 222–233.

[17] IEEE 802.11 Working Group, "Wireless LAN Medium Access Control (MAC) And Physical Layer (PHY) Specifications: Higher-speed Physical Layer Extension In The 2.4 GHz Band," Sep. 1999