OUKA
The University of Osaka
Institutional Knowledge Archive

Title	Transport-Layer Solutions for Achieving Fairness and Energy Efficiency in Wireless LANs
Author(s)	Hashimoto, Masafumi
Citation	大阪大学, 2013, 博士論文
Version Type	VoR
URL	https://hdl.handle.net/11094/27474
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

Transport-Layer Solutions for Achieving Fairness and Energy Efficiency in Wireless LANs

January 2013

Masafumi HASHIMOTO

Transport-Layer Solutions for Achieving Fairness and Energy Efficiency in Wireless LANs

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2013

Masafumi HASHIMOTO

List of Publications

Journal Papers

1. M. Hashimoto, G. Hasegawa, and M. Murata, "A transport-layer solution for alleviating TCP unfairness in a wireless LAN environment," *IEICE Transactions on Communications*, vol. E94-B, no. 3, pp. 765–776, Mar. 2011.

Refereed Conference Papers

1. M. Hashimoto, G. Hasegawa, and M. Murata, "Trade-off evaluation between fairness and throughput for TCP congestion control mechanisms in a wireless LAN environment," in *Proceedings of SPECTS 2010*, July 2010.

2. M. Hashimoto, G. Hasegawa, and M. Murata, "Modeling and analysis of power consumption in TCP data transmission over a wireless LAN environment," in *Proceedings of ICC Workshop (GreenComm 2011)*, pp. 1–6, June 2011.

3. M. Hashimoto, G. Hasegawa, and M. Murata, "Energy efficiency analysis of TCP with burst transmission over a wireless LAN," in *Proceedings of ISCIT 2011*, pp. 292–297, Oct. 2011.

4. M. Hashimoto, G. Hasegawa, and M. Murata, "Exploiting SCTP multistreaming to reduce energy consumption of multiple TCP flows over a WLAN," in *Proceedings of GreeNETs 2012*, Oct. 2012.

Non-Refereed Conference Papers

1. M. Hashimoto, G. Hasegawa, and M. Murata, "Fairness improvement of hybrid TCP congestion control mechanisms in wireless LAN environment," *Technical Report of IEICE (6th QoS Workshop of Technical Committee on Communication Quality)*, Dec. 2008. (in Japanese).

2. M. Hashimoto, G. Hasegawa, and M. Murata, "Experimental evaluation of TCP congestion control mechanisms in wireless LAN environment with a novel performance metric," *Technical Report of IEICE* (NS2009-126), vol. 109, no. 326, pp. 33–38, Dec. 2009. (in Japanese).

3. M. Hashimoto, G. Hasegawa, and M. Murata, "Analysis of power consumption in TCP data transmission over a wireless LAN environment," *Technical Report of IEICE* (NS2010-105), vol. 110, no. 339, pp. 1–6, Dec. 2010. (in Japanese).

4. M. Hashimoto, G. Hasegawa, and M. Murata, "Analysis of power consumption of a single TCP flow with burst transmission in a wireless LAN environment," in *Proceedings of the IEICE General Conference*, pp. S-107–S-108, Mar. 2011. (in Japanese).

5. M. Hashimoto, G. Hasegawa, and M. Murata, "Energy efficiency analysis of TCP with delayed ACK in a wireless LAN environment," *Technical Report of IEICE* (NS2011-50), vol. 111, no. 144, pp. 1–6, July 2011.

6. M. Hashimoto, G. Hasegawa, and M. Murata, "Proposal of SCTP tunneling for energy-efficient TCP data transfer over a wireless LAN environment," *Technical Report of IEICE* (IN2012-26), vol. 112, no. 88, pp. 13–18, June 2012. (in Japanese).

7. M. Hashimoto, G. Hasegawa, and M. Murata, "Study on exploiting SCTP to save energy of TCP data transfer over a wireless LAN," in *Proceedings of the IEICE Society Conference*, pp. S-20–S-21, Sept. 2012. (in Japanese).

Preface

With recent developments in network technologies, wireless networks have been an essential infrastructure for our everyday lives. To satisfy various user demands, wireless network technologies continue to develop and are put to practical use. In particular, IEEE 802.11-based WLANs have grown in popularity because of their high-speed access bandwidth and ease of installation of WLAN access points (APs) without any license. As a result, WLANs have been installed widely and many mobile devices have been able to access the Internet everywhere everytime through WLANs. Furthermore, a wide variety of applications on mobile devices have been running over a WLAN, which have different quality of service (QoS) requirements such as large bandwidth for file synchronization applications, low delay for voice over IP (VoIP) applications, and both for video chat applications. However, as the popularity of WLANs grows, many drawbacks of WLANs become obvious.

In WLANs, all mobile devices, or wireless clients, and an AP share the wireless channel fairly with carrier sense multiple access with collision avoidance (CSMA/CA), and thereby the WLAN ensures fairness among each at the media access control (MAC)-layer level. However, fairness at the MAC-layer level does not mean the fairness at upper layers. For instance, when we use transmission control protocol (TCP) as a transport-layer protocol in a WLAN, serious unfairness are caused among TCP flows; only few flows occupy a wireless channel and the others are starved. On the other hand, energy efficiency of wireless clients is an important issue to enhance availability because wireless clients are generally battery-driven. To save energy effectively, it is effective that a wireless network interface (WNI) of a client sleep during idle duration in which data are not being sent or received. However, the timings of packet transmission and reception depend on the behavior of transport-layer protocols and that of upper-layer applications on a wireless client. Therefore, we

should consider those behavior to improve energy efficiency.

In this thesis, in order to resolve these issues on transport-layer protocols and WLANs, we study on transport-layer solutions to achieve fairness and energy efficiency in WLAN environments. A major reason why we focus on transport-layer protocols is that their behavior is closely related to the causes of these issues. In addition, transport-layer protocols are easily modified since they are implemented as software.

This thesis begins by proposing, designing, and implementing a transport-layer solution to alleviate unfairness among TCP flows in WLANs. Two unfairness among TCP flows in a WLAN are focused, which are unfairness among upstream TCP flows and, unfairness between upstream and downstream TCP flows. To alleviate such unfairness, we present a transport-layer solution that TCP congestion control mechanisms are activated against congestion at an AP. It requires a small modification to TCP congestion control mechanisms at only wireless clients. In addition, for the appropriate evaluation of the proposed method, this thesis introduces a new performance evaluation metric that considers the trade-off relationships between per-flow fairness and bandwidth utilization at a network bottleneck. Through simulations, the proposed method successfully achieve both fairness among upstream TCP flows and fairness between upstream and downstream flows. We also conduct experiments using real WLAN environments with WLAN products from several vendors to confirm applicability and product-dependent characteristics of the proposed method. The experiments show the proposed method can achieve better a trade-off between fairness and bandwidth utilization, regardless of vendor implementations of APs and WNIs.

The second part of this thesis deals with energy efficiency of wireless clients in WLANs. Energy consumption models are constructed for TCP data transfer over a WLAN in order to assess impacts of the TCP behavior on energy efficiency. Because of importance of the TCP behavior, the proposed models consist of the combination of two layer models: a MAC-level model and a TCP-level model. Energy consumption models are derived for a WNI with continuously active mode (CAM) and with *ideal sleep* mode to reveal the sleep efficiency while considering detailed TCP behavior. Ideal sleep mode implies that a WNI knows the schedules of both the transmission and reception of TCP packets such that it can sleep and wake up with exact timing. Furthermore, in order to improve the sleep efficiency, TCP-level *burst transmission* behavior is proposed, which reduces the number of

state transitions between active and sleep modes by transmitting multiple data packets in groups. By comparing the energy consumption between WNIs with CAM and with ideal sleep mode, numerical examples reveal the upper bound of energy savings under these sleep strategies. In particular, TCP-level burst transmission successfully reduces energy consumption with increasing moderate delay.

To realize the energy saving by the above work, we need to alleviate uncoordinated behavior by multiple applications running on a single client, which produces the difficulty in control of sleep timing. For that purpose, as the last part of this thesis, stream control transmission protocol (SCTP) tunneling is proposed and designed. SCTP tunneling has two features: *flow aggregation* and *burst transmission*. It aggregates multiple TCP flows into a single SCTP association between a wireless client and an AP to control packet transmission and reception timings. Furthermore, to enhance the sleep efficiency, SCTP tunneling reduces the number of state transitions by handling multiple packets in a bursty fashion. We extend the above energy consumption model and constructs a mathematical model of the energy consumed by SCTP tunneling to assess its energy efficiency. Numerical examples based on the model show that the proposed method can reduce energy consumption by up to 69%. Implementation issues of SCTP tunneling are also discussed.

As stated above, we propose, design, and implement transport-layer solutions to achieve per-flow fairness among TCP flows and energy efficiency for multiple TCP flows in a WLAN. Through examining the issues on both fairness and energy efficiency, this thesis shows that these solutions are feasible approaches and can be applied to real WLAN environments.

Acknowledgments

First and foremost, I would like to express my heartfelt gratitude to Professor Hirotaka Nakano of Cybermedia Center, Osaka University, for his creative suggestions and precise guidance. His deeply insightful advice and inspiration have provided an invaluable experience for my research activity.

I also would like to express my sincere appreciation to Professor Masayuki Murata of Graduate School of Information Science and Technology, Osaka University, for his encouragement, insightful comments, and precise advice. His insightful directions inspired me to aim for higher goals.

I am heartily grateful to the members of my thesis committee, Professor Teruo Higashino and Professor Koso Murakami of Graduate School of Information Science and Technology, Osaka University, for reviewing my thesis and providing many valuable comments.

I am also deeply grateful to Associate Professor Go Hasegawa of Cybermedia Center, Osaka University, for his beneficial advice and precise guidance. My study would not have been accomplished without his constant support and persistent encouragement.

I would like to thank Assistant Professor Yoshiaki Taniguchi of Cybermedia Center, Osaka University, for his useful suggestions and fruitful discussion. His comments have been always informative and helpful.

I am thankful to all the members of Ubiquitous Network Laboratory and Advanced Network Architecture Laboratory in Graduate School of Information Science and Technology, Osaka University, for their inciting discussions and fellowship.

Last but not least, I thank my parents for their considerate support and continuous encouragement during my doctoral course.

Contents

List of Publications i

Preface iii

Acknowledgments vii

1 Introduction 1

1.1 Research Background 1

1.1.1 Fairness Issues among TCP Flows in WLANs 3

1.1.2 Energy Consumption Models in WLANs 4

1.1.3 Energy Saving Mechanisms in WLANs 5

1.2 Outline of Thesis 6

2 A Transport-Layer Solution to Improve Fairness among TCP Flows in a WLAN 11

2.1 Introduction 11

2.2 Fairness among TCP Flows in WLAN Environments 13

2.3 TCP Congestion Control for ACK Packet Losses 16

2.4 Performance Metric for Evaluating Trade-off between Fairness and Utilization . . . 18

2.4.1 Jain's Fairness Index 18

2.4.2 Proposed Metric 20

2.4.3 Comparison with Jain's Fairness Index 21

2.5 Evaluation with Simulation Experiments 22

2.5.1 Simulation Settings and Methods 22
2.5.2 Evaluation Results and Discussion 23
2.6 Experimental Evaluation 26
2.6.1 Experimental Settings and Methods 26
2.6.2 Evaluation Metric 30
2.6.3 Evaluation Results and Discussion 30
2.7 Summary 35

3 Energy Consumption Models of a Single TCP Flow in a WLAN **37**
3.1 Introduction 37
3.2 Network Model and Assumptions 40
3.3 Energy Consumption Models 41
3.3.1 Energy Consumption during Frame Exchanges for IEEE 802.11 MAC . . . 42
3.3.2 Energy Consumption of TCP Data Transfer 44
3.3.2.1 Number of Rounds during TCP Data Transfer 46
3.3.2.2 Energy Consumption of Packet Transmission and Reception . . . 48
3.3.2.3 Energy Consumption during Idle Duration 49
3.3.2.4 Energy Consumption during Sleep Duration 50
3.3.2.5 Energy Consumption due to State Transitions 53
3.3.2.6 Increase in Data Transfer Latency by Burst Transmission 54
3.4 Model Validation 55
3.4.1 Parameter Settings of Simulation and Analysis 55
3.4.2 Validation Results 56
3.5 Discussions with Analysis Results 58
3.5.1 Parameter Settings and Evaluation Metrics 58
3.5.2 Numerical Results 59
3.5.2.1 Fundamental Characteristics of Sleeping 59
3.5.2.2 Energy Efficiency of Burst Transmission 61

3.5.2.3 Trade-off Relationships between Energy Efficiency and Data Transfer Latency . . . 63
3.5.2.4 Summary . . . 65
3.6 WNI Factors for Energy Saving . . . 65
3.7 Summary . . . 68
3.A Derivation of Number of Rounds during TCP Data Transfer . . . 69

4 SCTP Tunneling: A Transport-Layer Solution to Reduce Energy Consumption in a WLAN 71

4.1 Introduction . . . 71
4.2 SCTP Tunneling . . . 73
4.2.1 Flow Aggregation . . . 74
4.2.2 Burst Transmission . . . 75
4.3 Power Consumption Model . . . 76
4.3.1 Assumptions . . . 76
4.3.2 Power Consumption of SCTP Tunneling . . . 77
4.3.3 Expected Size of SCTP Packets . . . 81
4.3.4 Increase in Transmission Latency due to Buffering Delay . . . 82
4.4 Discussion with Numerical Results . . . 82
4.4.1 Parameter Settings and Evaluation Metrics . . . 82
4.4.2 Numerical Results . . . 83
4.4.2.1 Energy Efficiency . . . 83
4.4.2.2 Trade-off Relationship between Energy Efficiency and Buffering Delay . . . 86
4.5 Implementation Issues of SCTP Tunneling . . . 87
4.5.1 Implementation Overview . . . 88
4.5.2 Trigger Frame Transmission . . . 90
4.6 Summary . . . 92

5 Conclusion **95**

Bibliography **99**

List of Figures

1.1 Typical WLAN environment 3

2.1 Congestion at the AP 14
2.2 Behavior of TCP Reno with and without the proposed method 16
2.3 Simulation environment 23
2.4 Average throughput of each flow when ten upstream flows exist 24
2.5 Average throughput of each flow when five upstream flows and five downstream flows coexist 25
2.6 Average throughput of each flow when ten upstream flows exist and wireless clients are relocated 26
2.7 Average RTT of each flow 27
2.8 Experimental environments 27
2.9 Effect of the number of upstream and downstream flows 31
2.10 Proposed index with SWM when ten upstream flows exist 33
2.11 Proposed index with SWM when five upstream flows and five downstream flows coexist 34
2.12 Effect of the delayed ACK option when using Buffalo-AP with a 50 [ms] one-way delay 35
2.13 Proposed index with SWM when using Buffalo-AP with a 50 [ms] one-way delay . 36

3.1 WLAN environment 40

3.2 Frame exchange in IEEE 802.11 MAC 42
3.3 An example of evolution of congestion window size 46
3.4 Packet sequence of transmission and reception and state transitions of WNI where $w_k = 5$ 50
3.5 WLAN environment for ns-2 simulation 56
3.6 Distribution of interarrival time of ACK packets where the rate of cross traffic is 20 Mbps 57
3.7 Energy consumption for the complete data transfer 60
3.8 Energy consumption ratio for varying q when RTT=100 [ms] 61
3.9 Energy efficiency as functions of both RTT and probability of packet loss events . . 62
3.10 Energy efficiency of burst transmission when RTT=100 [ms] 63
3.11 Energy efficiency as a function of m when $p = 0.01$ 64
3.12 Transfer latency as a function of m when $p = 0.01$ 64
3.13 Energy reductions for changing WNI parameters 67

4.1 SCTP tunneling 74
4.2 Packet sequences at a client's WNI 75
4.3 Evolution of SCTP congestion window size 78
4.4 SCTP packet sequences at WNI of wireless client 80
4.5 Power consumption as a function of aggregate throughput of upstream TCP flows . 84
4.6 Energy reduction ratio for various m values and $q = 0.1$ 86
4.7 Average buffering delay for various m values and $q = 0.1$ 87
4.8 Energy reduction ratio achieved subject to $D < D_{th}$ and $q = 0.1$ 88
4.9 Intercommunication between network protocol stacks of SCTP tunneling 89
4.10 Frame exchanges between client and AP focusing on trigger frame transmissions . 91

List of Tables

2.1 Comparison between Jain's fairness index and the proposed index (C= 30 [Mbps]) 22

2.2 WLAN products 28

2.3 Estimated buffer size of each AP 29

3.1 Average interarrival time of ACK packets for simulation and analysis at a variety of the rate of cross traffic 58

3.2 WLAN parameters 59

3.3 Power consumption of Atheros AR5004 [1] 59

4.1 WLAN parameters for numerical analyses 82

4.2 Power consumption of Atheros AR5004 [1] and parameters of state transitions [2, 3] 83

Chapter 1

Introduction

1.1 Research Background

With recent developments in network technologies, wireless networks have been an essential infrastructure for our everyday lives. To satisfy various user demands, wireless network technologies continue to develop and are put to practical use, such as Bluetooth [4], ZigBee [5], wireless LANs (WLANs) [6], 3G cellular networks, worldwide interoperability for microwave access (WiMAX) [7], and long term evolution (LTE) [8]. In particular, IEEE 802.11-based WLANs have grown in popularity because of their high-speed access bandwidth and ease of installation of WLAN access points (APs) without any license. As a result, WLANs have been installed widely in many areas such as airports, railway stations, hotels, offices, cafes, and homes. We can easily access the Internet through WLANs with mobile devices such as laptops, smartphones, portable game devices, and e-book readers.

Furthermore, a wide variety of applications on mobile devices have been running over WLANs, which have different quality of service (QoS) requirements such as large bandwidth for file synchronization applications, low delay for voice over IP (VoIP) applications, and both for video chat applications. Mobile applications that have different QoS requirements often run concurrently on a single device. For example, some people call their friends with VoIP while sharing documents, photos, and videos over a WLAN, and other people surf websites while watching videos with real-time

video streaming systems. Such a variety of applications and user usage brings the growth of the importance of fairness among users in WLANs. From a different aspect, they increase the amount of time spent on wireless communication, which is related to energy consumption.

WLANs ensure fairness among users because many users share a wireless channel. In detail, all mobile devices, or wireless clients, and an AP share the wireless channel fairly with carrier sense multiple access with collision avoidance (CSMA/CA), and thereby the WLANs ensure fairness among each at the media access control (MAC)-layer level. However, fairness at the MAC-layer level does not mean the fairness at upper layers [9–11]. For instance, when we use transmission control protocol (TCP) as a transport-layer protocol in a WLAN, serious unfairness are caused among TCP flows; only few flows occupy a wireless channel and the others are starved. On the other hand, energy efficiency of wireless clients is an important issue to enhance availability because wireless clients are generally battery-driven. However, accessing WLANs consumes large energy compared with the other wireless technologies [12]. Thus, to save energy effectively, it is effective that a wireless network interface (WNI) of a client sleeps during idle duration in which data are not being sent or received. However, packet transmission and reception depend on the behavior of transport-layer protocols and that of upper-layer applications on a wireless client. Therefore, we should consider those behavior to improve energy efficiency.

In order to deal with the above issues, the former part of this thesis addresses the fairness issue among TCP flows in WLANs. A transport-layer approach to alleviate unfairness among TCP flows is presented. For appropriate evaluation of the fairness issue, we also propose a performance metric considering both per-flow fairness and bandwidth utilization. In the latter part of this thesis, we address the energy issue of a wireless client in WLANs while considering the detailed TCP behavior. To this end, we first construct energy consumption models for a single TCP flow over a WLAN. Based on findings of numerical examples, we then develop a transport-layer approach to achieve energy saving of a wireless client in a WLAN.

In what follows, issues regarding fairness among TCP flows, energy consumption models, and energy saving mechanisms in WLANs are described in turn.

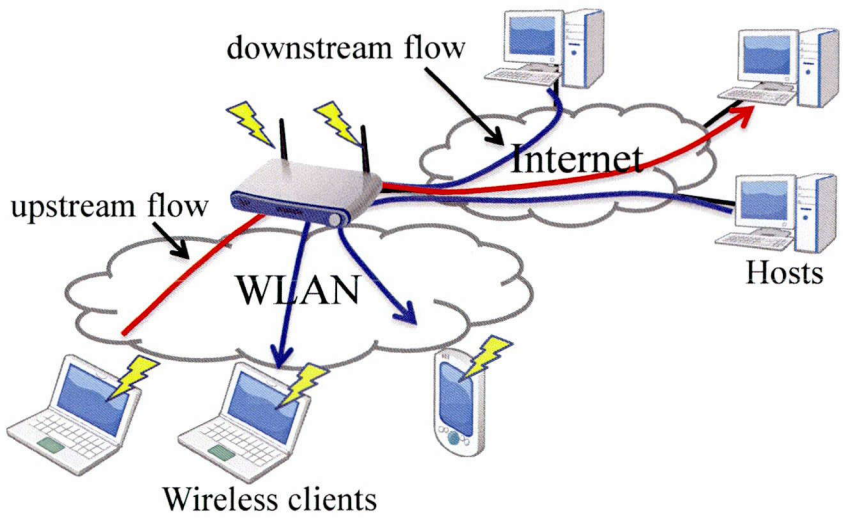


Figure 1.1: Typical WLAN environment

1.1.1 Fairness Issues among TCP Flows in WLANs

Figure 1.1 depicts a typical WLAN environment where three wireless clients associate with a single AP. In this thesis, flows from wireless clients toward an AP and vice versa are referred to as *upstream flows* and *downstream flows*, respectively. In a typical WLAN environment, all wireless clients and an AP use the same parameters of CSMA/CA. Thus, the contending clients and the AP obtain fair access opportunities to wireless channel in the long term. This also means that upstream flows and downstream flows share the same wireless channel. Because all traffic generated from wireless clients are forwarded through the AP, the AP is likely to become a congestion point, which results in the occurrence of unfairness between upstream and downstream traffic. Furthermore, when TCP is used as a transport-layer protocol, serious per-flow unfairness are caused not only between upstream and downstream flows but also among upstream flows. Reason for this is that TCP congestion control mechanisms does not contribute to alleviate the congestion at the AP.

Many solutions have been proposed for alleviating the above TCP unfairness [9–11, 13–16]. Pilosof *et al.* [9] proposed a solution to improve fairness among upstream and downstream flows by rewriting the advertised receiver window size at the AP. In [10], TCP unfairness among upstream flows is diminished by filtering ACK packets in an AP. Ha *et al.* [11] alleviate unfairness between

upstream and downstream flows by dividing the buffer in an AP into separate buffers for data packets and for ACK packets. In [13, 14], the authors proposed shortening the carrier sense duration of the WLAN APs. A solution proposed in [15] improves TCP unfairness among upstream and downstream flows by controlling the rate of upstream flows such that total throughput should be divided equally between upstream and downstream flows. Abeysekera *et al.* [16] achieved fairness between upstream and downstream flows by dynamically controlling the minimum contention window size at an AP. However, the cost of changing existing hardware devices using these methods is high because the MAC protocols or queue management mechanisms must be modified.

Therefore, in this thesis, we deal with the fairness issues among TCP flows from the viewpoint of transport-layer protocols, because the unfairness is mainly caused by the behavior of TCP congestion control mechanisms. In addition, transport-layer protocols are modified easier than MAC-layer protocols since they are generally implemented by software.

1.1.2 Energy Consumption Models in WLANs

Energy efficiency is one of the most important issues for wireless clients. Wireless communication via a WLAN is reported to account for up to 50% of the client's total energy consumption [12, 17–19]. Therefore, in order to investigate the factors to affect energy consumption of wireless clients in WLANs, many researchers have constructed energy consumption models for WLAN clients [20–24]. Anastasi *et al.* [20] modeled a single wireless client in power saving mode (PSM) of IEEE 802.11 downloading a file from a server in the presence of multiple wireless clients. Ergen and Varaiya [21] presented the results of an analysis of energy consumption during different MAC operations for a wireless client with multiple clients in a WLAN; they found that 80% of the total energy in saturated situations is wasted. Kuo [22] modeled the energy consumption of wireless clients based on Markov chain model in a saturated WLAN. Rantala *et al.* [23] modeled power consumption of a WNI, which consists of the power consumption of application-specific integrated circuit (ASIC) and that of the power amplifier (PA). Agrawal *et al.* [24] created a discrete-time Markov chain model of the energy consumption for TCP transfers in continuously active mode

(CAM) and in PSM in the presence of TCP background traffic. However, most of the above researches mainly focus on the behavior at only the MAC level and do not consider detailed TCP behavior. Because TCP congestion control mechanisms primarily determine the timing of packet transmission and reception, analysis of the detailed TCP behavior is necessary to assess sleep efficiency at packet interarrivals.

Other researchers modeled energy consumption based on the behavior of the TCP congestion mechanisms under the existence of channel error [25, 26]. Zorzi and Rao [25] modeled energy consumption for different TCP versions while considering a two-state Markov model for frame losses. Vacirca *et al.* [26] constructed an energy consumption model that consists of combination of a TCP behavior model and a MAC model under a Rayleigh fading channel. These two models consider energy consumption for a device activated in CAM. For improving energy efficiency, the behavior of sleep mechanisms and that of TCP congestion mechanisms should be considered.

In this thesis, we mainly focus on the impact of the TCP behavior for sleep efficiency when the WNI sleeps at packet interarrivals. By doing so, we investigate the maximum potential of energy saving from the transport-layer level while preventing degradation of performance such as throughput and delay.

1.1.3 Energy Saving Mechanisms in WLANs

For energy saving in MAC layer protocols of WLANs, the IEEE 802.11 standard defines PSM [6], as opposed to the mode under normal operation, which is referred to as the CAM. In IEEE 802.11 PSM, wireless clients only wake up at the beacon interval, which is typically 100 [ms]. Due to this, PSM can achieve high-energy efficiency while degrading network performance such as throughput and latency. To overcome this, IEEE 802.11e defines new power saving mechanisms, which is called automatic power save delivery (APSD) [27]. The main idea of unscheduled APSD (U-APSD), which is one of power saving modes of APSD, is that a data frame sent from wireless clients, which is referred to as trigger frame, is used as an indication of request for transmitting the buffered data frame at the AP. It requires determining the trigger generation algorithm for effective

usage of U-APSD. Mor *et al.* [28] overcome this issue by dynamically configuring trigger generation intervals as a function of the average intervals of arrival packets at the wireless client. Other researchers have proposed energy-efficient methods in WLANs by modifying MAC protocols or WNI hardware [2, 29–32]. The solutions in the references [2, 29–32] achieve energy efficiency on WLAN clients, which requires modifying the MAC protocols and the WNI hardware.

Other solutions for energy saving of wireless clients focus on the behavior of the upper-layer protocols [33–35]. Namboodiri and Gao [33] proposed GreenCall algorithm for VoIP applications, which derives sleep and wake-up schedules for the wireless client to save energy during VoIP calls. Dogar *et al.* [34] developed the Catnap proxy for data-oriented applications such as web browsing and file transfer. Yan *et al.* [35] presented a client-centered method in TCP over WLANs, with burst transmission realized by manipulating the TCP receiver's window size. In terms of sleep granularity, the Catnap proxy allows wireless clients to stay sleep mode at the whole data transmission scale, whereas the client-centered method allows them to stay sleep mode at the round trip time (RTT) scale. The recent development in radio frequency (RF) circuit design has resulted in sleep mode of shorter transition time [32], which implies that sleeping can be activated at an interarrival time of packets within one RTT.

Therefore, in this thesis, we focus on a transport-layer approach to save energy without requiring modifications of MAC-layer and physical-layer protocols. By changing timing of packet transmission and reception at the transport-layer level while collaborating with MAC-layer and physical-layer solutions, we aim to realize high energy efficiency of a client's WNI.

1.2 Outline of Thesis

This thesis begins by proposing a transport-layer solution to alleviate unfairness among TCP flows in a WLAN in Chapter 2. In Chapter 3, we investigate factors for effective energy saving through discussions based on numerical models for energy consumptions of a single TCP flow in a WLAN. Based on the findings to save energy in Chapter 3, Chapter 4 presents a transport-layer approach to reduce energy consumption in the presence of multiple TCP flows in a WLAN. Finally, this thesis is concluded in Chapter 5.

A Transport-Layer Solution to Improve Fairness among TCP flows in a WLAN [36–39]

In Chapter 2, we propose, design, and implement a transport-layer solution to alleviate unfairness among TCP flows in WLANs. When TCP detects packet losses as an indication of network congestion, its congestion control mechanisms are activated to alleviate the network congestion between two end hosts. However, its behavior cannot alleviate congestion at the AP with which multiple clients associate, which causes two unfairness among TCP flows in a WLAN: unfairness among upstream TCP flows and unfairness between upstream and downstream TCP flows. Therefore, to alleviate such unfairness, TCP congestion control should be activated not only against data packet losses but also against acknowledge (ACK) packet losses. To this end, we present a transport-layer solution that exploits ACK packet losses as an indication of congestion at an AP. It requires a small modification to TCP congestion control mechanisms at only wireless clients.

On the other hand, for such fairness issues among flows, Jain's fairness index [40] has been generally used as an evaluation metric. However, because improvement of fairness is sometimes achieved at the expense of bandwidth utilization, this index considering only fairness among flows cannot evaluate accurately a situation that fairness has a trade-off relationship with bandwidth utilization. Therefore, for the appropriate evaluation of the proposed method, this thesis introduces the proposed metric that considers the trade-off relationships between per-flow fairness and bandwidth utilization at a network bottleneck.

Through simulations, the proposed method successfully achieves both fairness among upstream TCP flows and fairness between upstream and downstream flows. We also conduct experiments using real WLAN environments with WLAN products from several vendors to confirm applicability and product-dependent characteristics of the proposed method. The experiments show the proposed method can achieve better a trade-off between fairness and bandwidth utilization, regardless of vendor implementations of APs and WNIs.

Energy Consumption Models of a Single TCP Flow in a WLAN [41–46]

Chapter 3 addresses energy efficiency of wireless clients in WLANs. Behavior of TCP congestion control mechanisms primarily determines the timing of packet transmission and reception, which has a large impact on the energy efficiency. Thus, energy consumption models are constructed for TCP data transfer over a WLAN in order to assess impacts of the TCP behavior on energy efficiency. Because of importance of the behavior of TCP congestion control mechanisms, the proposed models consist of the combination of two layer models: a MAC-level model and a TCP-level model. In the MAC-level model, the amounts of energy consumed in sending and receiving one data frame are calculated based on frame exchanges in CSMA/CA mechanisms. In the TCP-level model, which is based on the behavior of TCP congestion control mechanisms, the energy consumption in TCP data transfer is determined from packet transmission and reception timing, depending on the evolution of the TCP congestion window size. Energy consumption models are derived for a WNI with CAM and with *ideal sleep* mode to reveal the sleep efficiency while considering detailed TCP behavior. Ideal sleep mode implies that a WNI knows the schedules of both the transmission and reception of TCP packets such that it can sleep and wake up with exact timing. Furthermore, in order to improve the sleep efficiency, TCP-level *burst transmission* behavior is proposed, which reduces the number of state transitions between active and sleep modes by transmitting multiple data packets in groups.

By comparing the energy consumption between WNIs with CAM and with ideal sleep mode, numerical examples reveal the upper bound of energy savings under these sleep strategies. In particular, TCP-level burst transmission successfully reduces energy consumption with increasing moderate delay.

SCTP Tunneling: A Transport-Layer Solution to Reduce Energy Consumptions in a WLAN [47–50]

Chapter 4, we realize the burst transmission feature at transport-layer level in a WLAN to effective energy saving. In a typical WLAN, multiple network applications are running concurrently on a single wireless client. In such a situation, packets of each application are sent and received independently, which are multiplexed at MAC-level. This uncoordinated behavior produces the

difficulty in control of sleep timing. In order to overcome the issue, stream control Transmission Protocol (SCTP) tunneling is proposed and designed, which is a transport-layer approach to save energy for TCP data transfer over a WLAN. SCTP tunneling has two features: *flow aggregation* and *burst transmission.* It aggregates multiple TCP flows into a single SCTP association between a wireless client and an AP to control packet transmission and reception timing. Furthermore, to enhance the sleep efficiency, SCTP tunneling reduces the number of state transitions by handling multiple packets in a bursty fashion. By extending the above energy consumption models, we construct a mathematical model of the energy consumed by SCTP tunneling to assess its energy efficiency.

Numerical examples based on the model shows that the proposed method can reduce energy consumption by up to 69%. Implementation issues of SCTP tunneling are also discussed.

Chapter 2

A Transport-Layer Solution to Improve Fairness among TCP Flows in a WLAN

2.1 Introduction

With recent developments in wireless networking technologies, accessing the Internet through a wireless LAN (WLAN) is becoming common, and WLAN-based Internet access environments are often available at public areas, such as railway stations and airports. As such, it is important to consider fairness among coexisting users.

IEEE 802.11 families [6] are standardized as the WLAN environment. The current IEEE 802.11 implementations primarily use the distributed coordination function (DCF) as the medium access control protocol. In the DCF, a medium access mechanism is based on CSMA/CA, which enables wireless clients and an access point (AP) to fairly access a wireless channel [6]. Therefore, fairness among clients in the WLAN is ensured at the MAC layer. However, the fairness at the MAC layer does not mean the fairness at upper layers. When TCP is used as a transport-layer protocol, two types of unfairness among TCP flows (TCP unfairness) are caused [9–11]: unfairness between upstream and downstream TCP flows and unfairness among upstream TCP flows. Here, the traffic in a WLAN consists of the traffic flowing from clients to wired networks via an AP (called upstream traffic) and vice versa (called downstream traffic). Downstream traffic is transmitted only from an

AP, whereas upstream traffic is generated from multiple clients. Therefore, upstream traffic obtains more access opportunities to the wireless channel than downstream traffic, resulting in unfairness between upstream and downstream traffics. On the other hand, the unfairness among upstream flows is caused by their congestion control mechanisms, i.e., the TCP congestion control is activated against data packet losses, but not against ACK packet losses [51].

Many solutions have been proposed to alleviate unfairness in WLANs [9–11, 13–16]. These solutions diminish TCP throughput unfairness by modifying the MAC protocol parameters or queue management mechanisms in APs. However, the MAC protocols of APs are generally implemented at the hardware level, so changing these protocols is costly. Furthermore, some methods need to estimate the number of flows and the throughput of each flow at the AP. In addition, although certain solutions can alleviate the unfairness by changing the MAC layer, they may also cause other unfairness problems in the transport layer. For example, a priority-based solution at the AP [13] can improve fairness among TCP flows, but this solution may cause unfairness among UDP flows by assigning higher priority to traffic from the AP.

As the first contribution of this chapter, we propose a transport-layer approach to alleviate the above unfairness. The reasons for using a transport-layer solution are as follows. First, MAC-layer solutions may cause other unfairness issues at the transport layer, as described above. Second, the unfairness is mainly caused by the behavior of the transport-layer protocols. Third, since transport-layer protocols are generally implemented by software, modifying transport-layer protocols is easier than modifying MAC-layer protocols. The proposed method alleviates unfairness among TCP flows by detecting TCP ACK packet losses as an indication of congestion at an AP. It requires a small modification of the TCP congestion control mechanisms only on WLAN clients.

Generally, Jain's fairness index [40] has been used as an evaluation metric of fairness among users, flows, etc. Since this index depends only on the variation of allocated values, the index values intuitively indicate whether the allocated values are fair. Here, assume that we have two solutions, both of which are identically effective for alleviating unfairness, but one of them degrades the total amount of allocated values, whereas the other does not decrease the total value. Although the latter should be judged to be superior to the former, both solutions are identical from the viewpoint

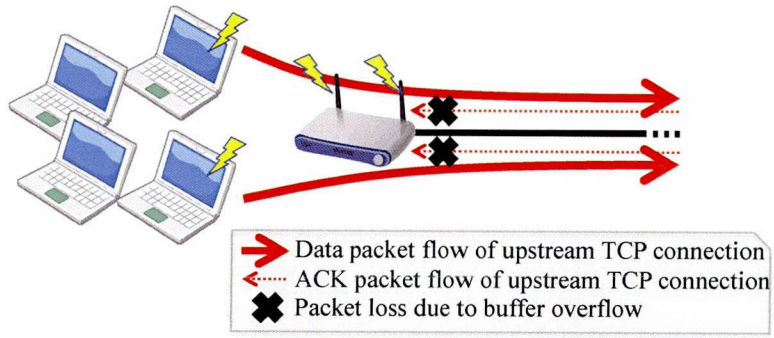
of Jain's fairness index. In other words, Jain's fairness index cannot evaluate such situations accurately. Therefore, as the second contribution of this chapter, we propose a novel performance metric that considers the trade-off relationships between per-flow fairness and bandwidth utilization at a network bottleneck. The proposed metric can simultaneously evaluate both fairness and utilization with a single metric value.

We conducted ns-2 [52] simulation experiments in order to confirm the fundamental characteristics of the proposed method. Through the simulation results, we show that the proposed method is effective not only for fairness among upstream flows but also for fairness between upstream and downstream flows. We then confirm the applicability and product-dependent characteristics of the proposed method through experiments using real environments with WLAN products from several vendors.

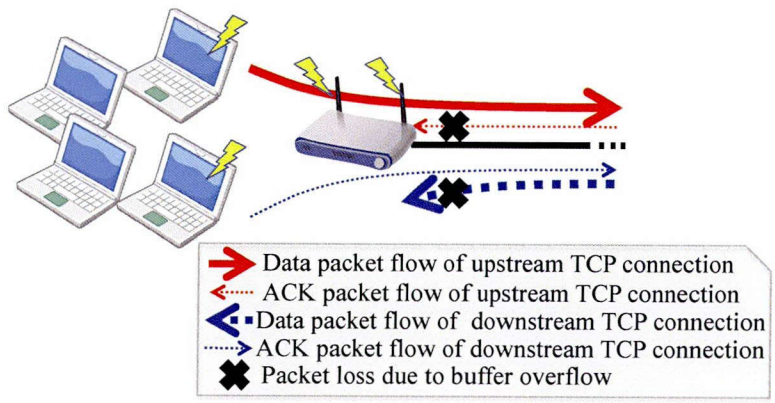
The remainder of Chapter 2 is organized as follows. In Sect. 2.2, we discuss unfairness problems among TCP flows in WLANs. Section 2.3 describes a solution for alleviating TCP unfairness in WLANs. In Sect. 2.4, we introduce a performance metric considering both per-flow fairness and bandwidth utilization. In Sect. 2.5, we confirm the fundamental characteristics of the solution proposed in Sect. 2.3 through simulation experiments. Section 2.6 presents experimental results. Finally, conclusions and a discussion of future research are presented in Sect. 2.7.

2.2 Fairness among TCP Flows in WLAN Environments

In a typical WLAN environment, all wireless clients and an AP use the same parameters of CSMA/CA. Thus, the contending clients obtain fair access opportunities to wireless channel in the long term. This also means that upstream flows transmitted from the multiple clients and downstream flows transmitted from the AP share the same wireless channel. Therefore, when n clients share a single AP in the WLAN, the access opportunities to the wireless channel of upstream traffic is $n/(n+1)$, whereas that of downstream traffic is $1/(n+1)$. Hence, when the amount of the upstream and downstream traffic is roughly the same, the AP is likely to become a congestion point, and unfairness occurs between upstream and downstream traffic. It means that the fairness at the MAC layer protocol does not always contribute to the fairness at upper layer protocols such as UDP and



(a) Multiple upstream TCP flows exist



(b) Upstream and downstream TCP flows coexist

Figure 2.1: Congestion at the AP

TCP. Since the many kinds of applications such as P2P file sharing and audio/video conference applications generate both upstream and downstream flows, this issue should be resolved.

Furthermore, when TCP is used as a transport-layer protocol, serious per-flow unfairness is caused not only between upstream and downstream flows but also among upstream flows. In what follows, we specifically explain the two kinds of the unfairness. Figure 2.1 depicts the situations in which the unfairness occurs when the wireless network bandwidth is fully utilized.

Suppose that each client station has an upstream TCP flow, as depicted in Fig. 2.1(a). In this situation, ACK packets of the upstream flows are discarded at the AP buffer. Note that TCP congestion control is activated against data packet losses, but not against ACK packet losses. Therefore, the congestion window size of upstream flows continues to grow until retransmission timeout (RTO)

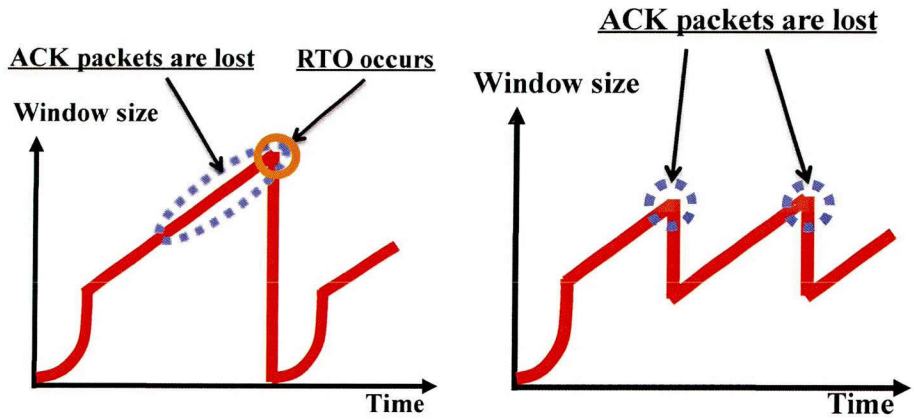
occurs and *all* ACK packets in a window are lost. When all ACK packets of a certain upstream flow are discarded, the congestion window size is set to one packet. At this time, the buffer at the AP is still fully utilized because the congestion is not resolved. As a result, once a certain flow experiences an RTO, the flow cannot increase its congestion window size for some time, which causes throughput unfairness among upstream TCP flows.

On the other hand, when upstream and downstream TCP flows coexist as shown in Fig. 2.1(b), ACK packets of upstream flows and data packets of downstream flows are discarded at the AP buffer. In such a situation, the upstream TCP flows continue to grow the congestion window size, whereas downstream TCP flows decrease the congestion window size, since TCP activates the congestion control mechanism only against data packet losses. Consequently, the downstream TCP flows maintain the low transmission rate, whereas the upstream TCP flows maintain the high transmission rate. Therefore, serious throughput unfairness is caused between upstream and downstream TCP flows.

Such two kinds of TCP unfairness are caused when using TCP variants that use data packets losses as an indication of the occurrence of network congestion, such as TCP Reno, TCP NewReno, and TCP SACK. In what follows, we assume the use of TCP Reno as an example for TCP unfairness issues.

Many solutions have been proposed for alleviating the above TCP unfairness [9–11, 13–16]. A solution proposed in [9] improves fairness among upstream and downstream flows by rewriting the advertised receiver window size at the AP. A solution proposed in [11] alleviates unfairness between upstream and downstream flows by dividing the buffer in an AP into a buffer for data packets and a buffer for ACK packets. In [10], TCP unfairness among upstream flows is diminished by filtering ACK packets in an AP. In [13], the author proposed shortening the carrier sense duration of the WLAN APs. A solution proposed in [15] improves TCP unfairness among upstream and downstream flows by controlling the rate of upstream flows such that total throughput should be divided equally between upstream and downstream flows. However, the cost of changing existing hardware devices using these methods is high because the MAC protocols or queue management mechanisms must be modified.

On the other hand, since transport-layer protocols are generally implemented through software,



(a) Without modification (b) Proposed method

Figure 2.2: Behavior of TCP Reno with and without the proposed method

changing transport-layer protocols are easier than changing MAC-layer protocols. In addition, the above unfairness is mainly caused by the behavior of TCP. Therefore, in this chapter, we propose a transport-layer solution for alleviating TCP unfairness.

2.3 TCP Congestion Control for ACK Packet Losses

Figure 2.2(a) depicts the behavior of TCP Reno when ACK packets are discarded at the AP buffer. The main reason for unfairness among TCP flows is that the TCP continues to increase the congestion window size even when the AP is highly congested and numerous ACK packets are discarded, as shown in Fig. 2.2(a). Therefore, we propose a simple modification to TCP congestion control mechanisms to alleviate these unfairness. The proposed method is based on a simple concept: TCP congestion control should be activated when ACK packet losses are detected, whereas the traditional TCP activates congestion control only when data packet losses are detected. Figure 2.2(b) shows the behavior of the proposed method on TCP Reno, corresponded to Fig. 2.2(a). The congestion control with the proposed method is activated when detecting ACK packet losses, as shown in Fig. 2.2(b).

More specifically, in the proposed method, congestion control is activated when the number

of ACK packet losses in a window exceeds a pre-determined threshold (*thresh_ack_losses*). The TCP sender detects ACK packet losses by monitoring the sequence number of the received ACK packets. When the TCP sender observes abnormal jumps in the ACK sequence numbers, these jumps are regarded as ACK packet losses in the network and are calculated as follows:

$$ack_loss = \max\left(\left\lfloor \frac{ack - prev_ack}{MSS} \right\rfloor - 1,\ 0\right) \tag{2.1}$$

where ack_loss is the number of ACK packet losses, ack is the sequence number [bytes] of the current received ACK packet, $prev_ack$ is the sequence number [bytes] of the previous received ACK packet, and MSS is maximum segment size [bytes]. When the number of ACK packet losses in a round trip time (RTT) exceeds *thresh_ack_losses*, a TCP sender halves the congestion window ($cwnd$) and slow-start threshold ($ssthresh$) is set to the halved congestion window size. Note that before halving the window size, the TCP sender waits for another RTT to avoid false detection caused by the disorder of data packet reception at the TCP receiver. Additionally, when detecting data packet losses, a TCP sender stops checking the ACK packet losses for one RTT.

When using Eq. (2.1) to detect ACK packet losses, the effect of the delayed ACK option [53] should be considered. Note that the delayed ACK option has been implemented in both Windows and Linux [54, 55]. Let b be the number of data packets acknowledged by an received ACK packet. Some TCP receiver implementations with the delayed ACK send one ACK packet for two consecutive received packets, so b is two typically. When the delayed ACK option is enabled, Eq. (2.1) cannot correctly estimate the number of ACK packet losses. Therefore, TCP senders must determine the value of b.

There are two possible methods by which determine the value of b. In the first method, a TCP receiver explicitly informs a TCP sender of the value of b. The second method is that a TCP sender estimates the value of b without any explicit information from the TCP receiver. In the first method, the TCP sender can obtain an accurate value of b, however this method requires TCP receiver-side modifications. For this reason, we use the second method, which does not require TCP receiver-side modifications.

In the proposed method, the value of b is estimated as follows:

$$b_{est} = \left\lfloor sb_i + \frac{1}{2} \right\rfloor \tag{2.2}$$

$$sb_i = (1 - \beta)\, sb_{i-1} + \beta \left(\frac{ack - prev_ack}{MSS} \right) \tag{2.3}$$

where b_{est} is the estimated value of b, and sb_i is the i th smoothed value for b with smoothing factor β. Note that sb_i in Eq. (2.3) is a continuous value, but b_{est} in Eq. (2.2) should be a discrete value because b should be a discrete value. Thus, b_{est} is calculated by half-adjust rounding. Using Eq. (2.2), Eq. (2.1) is transformed into:

$$ack_loss = \max \left(\left\lfloor \frac{ack - prev_ack}{b_{est} \times MSS} \right\rfloor - 1,\ 0 \right). \tag{2.4}$$

Algorithm 1 presents the pseudo-code of the proposed method. Note that the proposed method can be utilized with arbitrary TCP modifications because the proposed method can be applied to some TCP variants without any ill-effect.

In addition, because the proposed method is one of TCP modifications which detects early congestion of networks such as TCP Reno, the proposed method is ineffective when the proposed method and that TCP modifications coexist. Thus, the proposed method has the same issues of TCP modifications which have been reported in several researches [56, 57].

2.4 Performance Metric for Evaluating Trade-off between Fairness and Utilization

2.4.1 Jain's Fairness Index

The definition of fairness is important when we discuss fairness among flows, because the improvement of fairness is sometimes achieved at the expense of total bandwidth utilization. In previous researches [9–11, 13, 15], fairness is defined as all flows that contend on a wireless channel in a

Algorithm 1 TCP congestion control for ACK packet losses

Initialization:
$prev_ack \leftarrow$ the smallest sequence number of the unacknowledged packets (snd_una)
$sb \leftarrow MSS \times 2$, $b \leftarrow MSS \times 2$
$cnt_ack_loss \leftarrow 0$
$data_loss \leftarrow 0$, $wait_state \leftarrow 0$
On each ACK:
$sb \leftarrow (1-\beta)\, sb + \beta \left(\frac{ack - prev_ack}{MSS}\right)$
$b_{est} \leftarrow \lfloor sb + \frac{1}{2} \rfloor$
$cnt_ack_loss \leftarrow \max\left(\left\lfloor \frac{ack - prev_ack}{b_{est} \times MSS} \right\rfloor - 1 + cnt_ack_loss,\ 0\right)$
$prev_ack \leftarrow ack$
On each RTT:
if $wait_state$ **then**
  if $cnt_ack_loss \geq thresh_ack_losses$ and not $data_loss$ **then**
    $cwnd \leftarrow \max\left(\frac{cwnd}{2},\ 1\right)$
    $ssthresh \leftarrow cwnd$
  end if
  $data_loss \leftarrow 0$
  $cnt_ack_loss \leftarrow 0$
  $wait_state \leftarrow 0$
else
  if $cnt_ack_loss \geq thresh_ack_losses$ or $data_loss$ **then**
    $wait_state \leftarrow 1$
  end if
end if
Packet loss:
$data_loss \leftarrow 1$
Packet disorder:
$cnt_ack_loss \leftarrow \max(cnt_ack_loss - 1,\ 0)$

WLAN achieve the same throughput, and its impact on the total network throughput is not considered.

Jain's fairness index, which defined as follows, has been generally used to assess the fairness:

$$F_j(X) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} x_i^2} \tag{2.5}$$

where n is the number of contending users, $X = \{x_1, x_2, \cdots, x_n\}$ is a set of allocations for n users

such that x_i is an allocation for user i. The index value approaches one as the variation of allocations decreases, and the index value approaches $1/n$ as the variation of allocations increases. Note that Jain's index is independent of the scale of allocations. For example, consider fairness when allocating 10, 30, and 40 dollars, respectively, to three persons, and fairness when allocating 100, 300, and 400 dollars, respectively, to three persons. Both cases are equivalent from the viewpoint of Jain's index (0.82).

However, the total amounts of allocated values are different. That is, Jain's index is not suitable for comparing fairness while considering the total amount of allocations. The total amount of allocated values corresponds to the network bandwidth utilization in the context of network bandwidth sharing. Therefore, when we have a solution for alleviating unfairness while slightly degrading the total throughput, Jain's index cannot accurately evaluate such a performance trade-off.

2.4.2 Proposed Metric

Given a throughput set $X = \{x_1, x_2, \cdots, x_n\}$, where x_i is the throughput of the i th flow, and the network bandwidth at the bottleneck, C, where $\sum_{i=1}^{n} x_i \leq C$, we define *fair and fully-utilized throughput* $x_f = \frac{C}{n}$, where all flows achieve the same throughput and the network bandwidth is fully utilized. Using the relationship between Jain's fairness index $F_j(X)$ in Eq. (2.5) and total throughput $\sum_{i=1}^{n} x_i$, we define the desired properties for proposed fairness index $F(X, C)$ as follows:

1. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \leq C$ and $F_j(X) < F_j(Y)$, then $F(X, C) < F(Y, C)$.
2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \leq C$ and $F_j(X) = F_j(Y)$, then $F(X, C) = F(Y, C)$.
3. If $\sum_{i=1}^{n} x_i = C$, then $F_j(X) = F(X, C)$.

where $Y = \{y_1, y_2, \cdots, y_n\}$.

We start from the index $f(X, C)$, which represents the average squared distance between the each flow's throughput and the fair and fully-utilized throughput (x_f):

$$f(X, C) = \frac{1}{n} \sum_{i=1}^{n} (x_i - x_f)^2. \tag{2.6}$$

We then normalize $f(X,C)$ by x_f and obtain $g(X,C)$ as follows:

$$g(X,C) = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - x_f)^2}}{x_f}. \tag{2.7}$$

According to [40], Jain's fairness index can be transformed into

$$F_j(X) = \frac{1}{1+COV^2} \tag{2.8}$$

$$COV = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}}{\bar{x}} \tag{2.9}$$

where COV is the coefficient of variance of allocations to the users, and $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the average of the allocations. Finally, by comparing Eqs. (2.7) and (2.9), we obtain the following definition of the proposed index:

$$\begin{aligned} F(X,C) &= \frac{1}{1+g(X,C)^2} \\ &= \frac{C^2}{n\sum_{i=1}^{n} x_i^2 - 2C\sum_{i=1}^{n} x_i + 2C^2}. \end{aligned} \tag{2.10}$$

Note that the index satisfies the above-mentioned properties. The index approaches one when the bandwidth utilization of the network bottleneck approaches 100% and the throughput variance of each flow is small. In contrast, the index approaches $1/n$ when the throughput variance is large.

2.4.3 Comparison with Jain's Fairness Index

In Table 2.1, Jain's fairness index and the proposed index are compared using simple examples in which the network capacity bandwidth is 30 [Mbps]. Cases 1, 2, and 3 in Table 2.1 have fair throughput distributions, but different total throughputs. The proposed index can differentiate these cases, whereas Jain's index cannot distinguish among them. As in Cases 1, 2, and 3, Cases 4, 5, and 6 have different total throughputs, however the throughput distributions have the same variations. The proposed index becomes small for Cases 4, 5, and 6 since the variance of throughput distribution of which are 2.25, 4.0, and 9.0, respectively. Thus, the values of the proposed index become

Table 2.1: Comparison between Jain's fairness index and the proposed index (C= 30 [Mbps])

Case	Throughput distribution [Mbps]	Total [Mbps]	Jain's index	Proposed index
1	{3, 3, 3, 3, 3, 3, 3, 3, 3, 3}	30	1.00	1.00
2	{2, 2, 2, 2, 2, 2, 2, 2, 2, 2}	20	1.00	0.90
3	{1, 1, 1, 1, 1, 1, 1, 1, 1, 1}	10	1.00	0.69
4	{1, 1, 1, 1, 1, 1, 1, 1, 1, 6}	15	0.50	0.67
5	{1, 1, 1, 1, 1, 1, 1, 1, 6, 6}	20	0.50	0.64
6	{2, 2, 2, 2, 2, 2, 2, 2, 2, 12}	30	0.50	0.50
7	{1, 1, 1, 1, 1, 1, 2, 3, 3, 6}	20	0.62	0.73
8	{1, 1, 1, 3, 3, 3, 3, 4, 5, 6}	30	0.78	0.78

small when a small number of flows obtain much bandwidth, even if the total throughput is high. In Cases 2, 5, and 7, the total throughputs are identical, but the throughput distributions have different variations. These cases corresponded to property 1 in Subsect. 2.4.2. In this situation, the order of the three cases in Jain's index and that in the proposed index are identical. Moreover, when the total throughput is equal to the network bandwidth capacity, as in Cases 1, 6, and 8, Jain's index and the proposed index are identical. This situation is related to property 3 in Subsect. 2.4.2.

2.5 Evaluation with Simulation Experiments

In this section, we show simulation results in order to confirm the basic characteristics of the proposed method.

2.5.1 Simulation Settings and Methods

Figure 2.3 shows the simulation environment using IEEE 802.11a WLAN with the ns-2 simulator. In the environment, multiple wireless clients share a single AP that is connected to a wired node through a wired link with 100 [Mbps] capacity and one-way propagation delay of 100 [ms]. In the simulation evaluation, except as otherwise noted, all wireless clients were located at four [meters] from the AP and the buffer size of the AP was set to 100 [packets]. The buffer size of the wired link, the sender buffer size of each client, and the advertised receiver window size were set to large

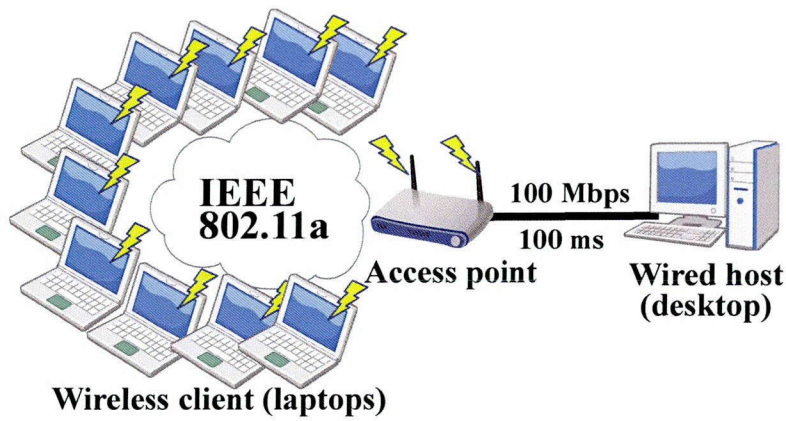


Figure 2.3: Simulation environment

enough not to limit the TCP performance. As the queue management mechanisms, the Drop Tail principle was used in the AP and the wired link. TCP connections used either the proposed method on TCP Reno or conventional TCP Reno. In order to use these TCPs in the ns-2 simulator, we used the ns-2 modules that are converted from the Linux implementation codes by NS-2 TCP-Linux [58]. The *thresh_ack_losses* parameter in the proposed method was set to one. Note that we confirmed that we can obtain the best performance of the proposed method when the *thresh_ack_losses* parameter is one. In addition, for comparative purposes, the prioritised AP scheme proposed in [13] was also evaluated. In order to focus on fundamental characteristics, the delayed ACK option me was disabled. The effects of the delayed ACK option are investigated in Sect. 2.6.

In the simulation experiments, only one flow was generated for each client, meaning that we increased the number of clients for increasing the number of concurrent flows in the network. The simulation time was set to 200 [seconds] and the data transmission of each flow started at random time which was uniformly distributed on [0 s, 10 s]. The simulations ran ten times in order to average the results.

2.5.2 Evaluation Results and Discussion

Figures 2.4 and 2.5 show the snapshot results for the average throughput when ten upstream flows exist and when five upstream flows and five downstream flows coexist, respectively. Note that the

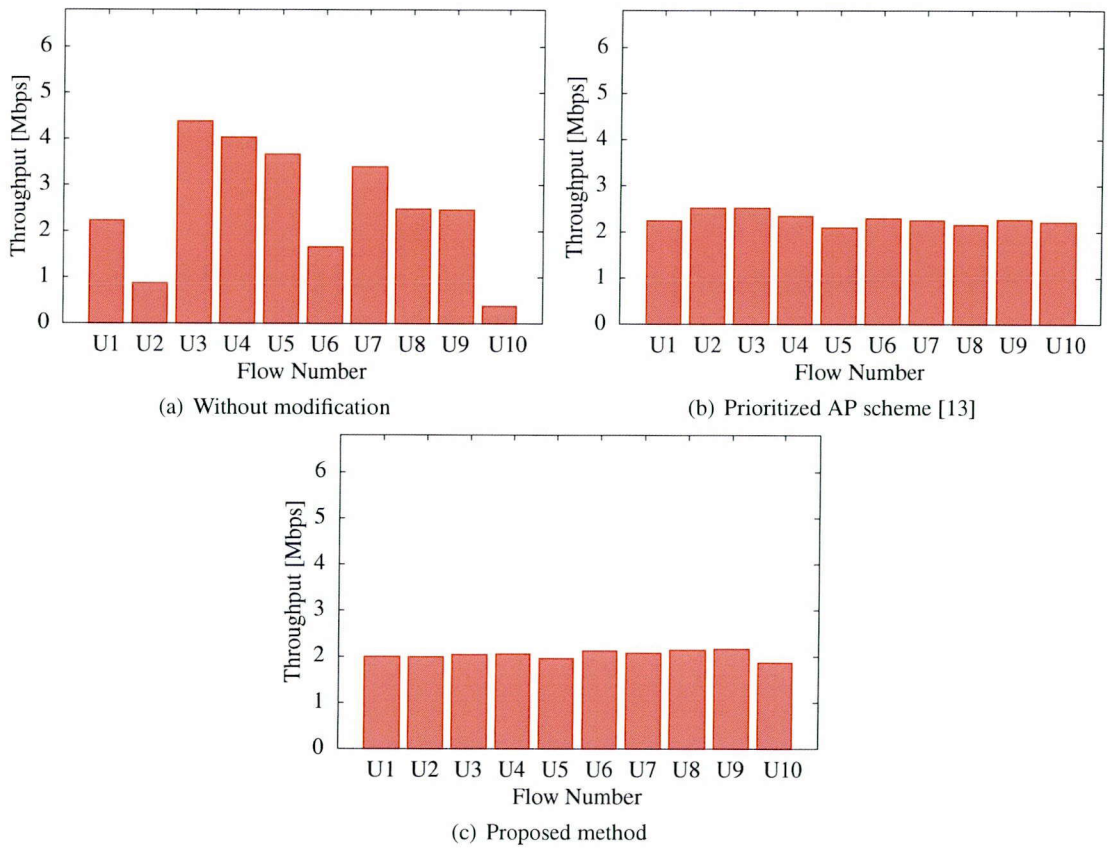


Figure 2.4: Average throughput of each flow when ten upstream flows exist

other results of ten times trials are also almost the same as Figs. 2.4 and 2.5. In these figures, Ui and Dj in x-axis denote the i th upstream flow and the j th downstream flow, respectively. The average throughput of each flow is calculated using the amount of data sent in 50-200 [seconds] in the simulation. Figures 2.4(a) and 2.5(a) shows the serious TCP unfairness is caused among upstream flows and between upstream and downstream flows without modification, whereas the prioritized AP scheme and the proposed method successfully alleviates TCP unfairness both among upstream flows and between upstream and downstream flows as shown in Figs. 2.4(b), 2.4(c), 2.5(b), and 2.5(c).

Figure 2.6 represents the results when there are ten upstream flows and the five clients are located at one [meter] and the others are located ten [meters] from the AP. Note that we obtained the

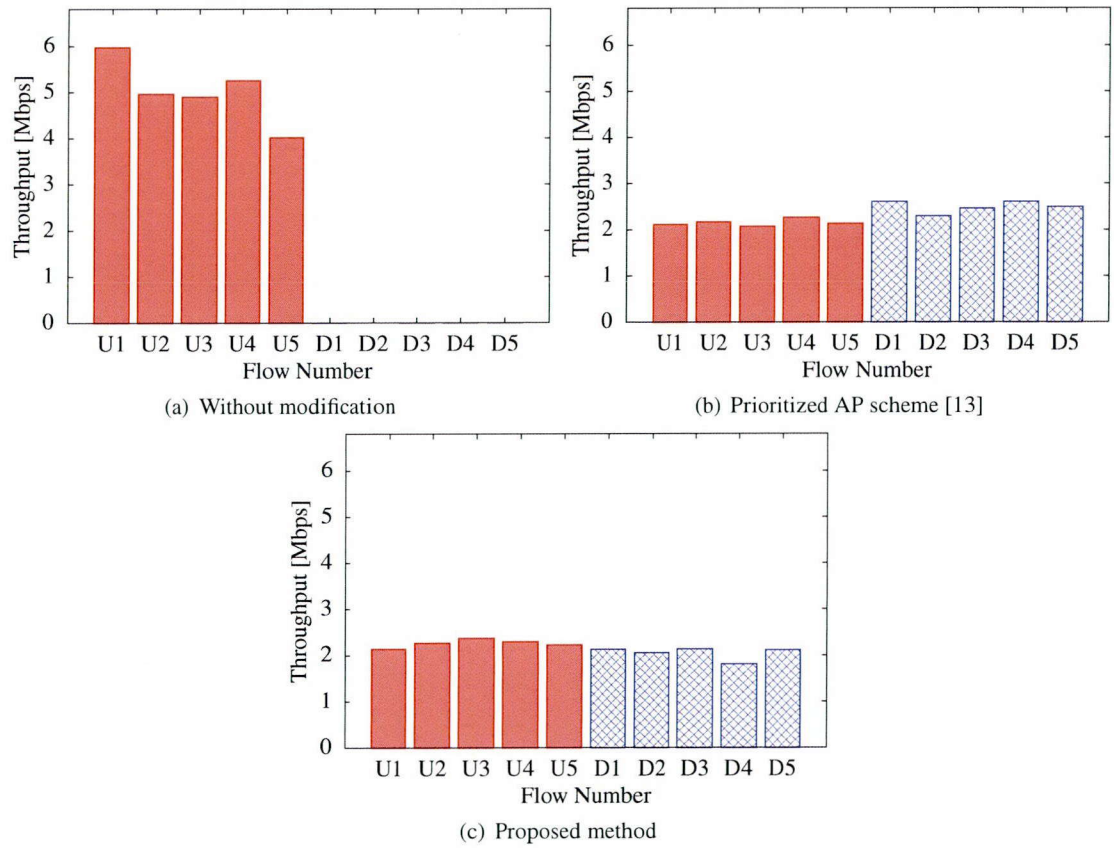
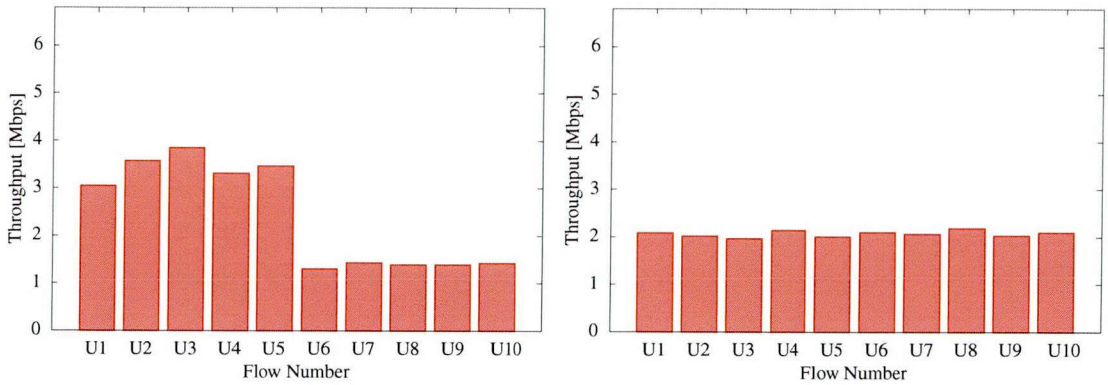


Figure 2.5: Average throughput of each flow when five upstream flows and five downstream flows coexist

similar results when upstream and downstream flows coexist. The prioritized AP scheme degrades the effectiveness for alleviating unfairness among flows dependently on distance of the clients from AP, as shown in Fig. 2.6(a). This is because that the prioritized AP scheme is sensitive to the wireless channel environment since it is based on MAC parameter tuning regarding the channel access. On the other hand, the performance of the proposed method is independent on the wireless channel since the proposed method is based on a transport-layer approach.

Figure 2.7 plots the average RTT of each flow corresponded to the results in Figs. 2.4 and 2.5. In Fig. 2.7, RTTs of each flow with the proposed method are smaller than without modification or with the prioritized AP scheme. The reason for this is as follows. Without modification, the



(a) Prioritized AP scheme [13]

(b) Proposed method

Figure 2.6: Average throughput of each flow when ten upstream flows exist and wireless clients are relocated

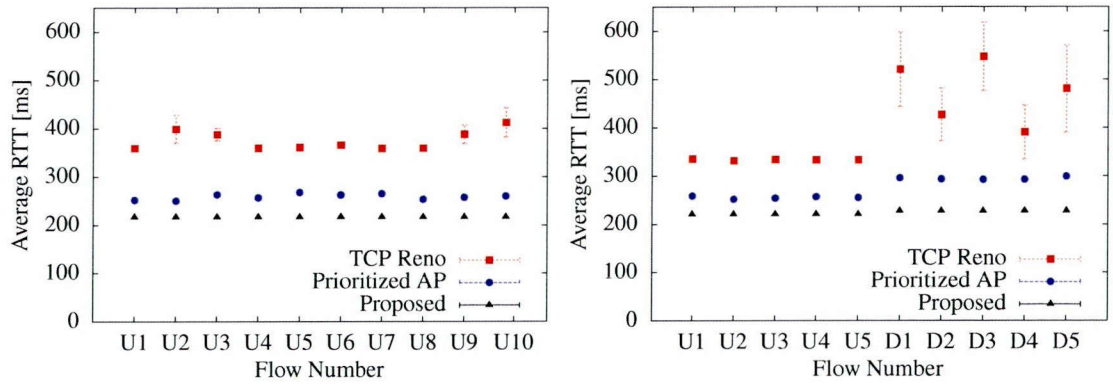
congestion window size of upstream flows continues to grow and the packets exceeded the data rate of wireless channel are buffered at the sender buffer of each client. This results in that RTTs of each flow become large. Likewise, with the prioritized AP scheme, RTTs of each flow also become large because the prioritized AP scheme can improve fairness among flows, but the congestion window size of upstream flows remain to grow when the wireless channel is fully utilized. In contrast, the proposed method can also alleviate the congestion at the wireless channel because the congestion control with the proposed method is activated against upstream flows. As a result, the proposed method can maintain low RTT.

2.6 Experimental Evaluation

In this section, we present experimental results using real environments with WLAN products from several vendors in order to confirm the applicability and product-dependent characteristics of the proposed method in real environments.

2.6.1 Experimental Settings and Methods

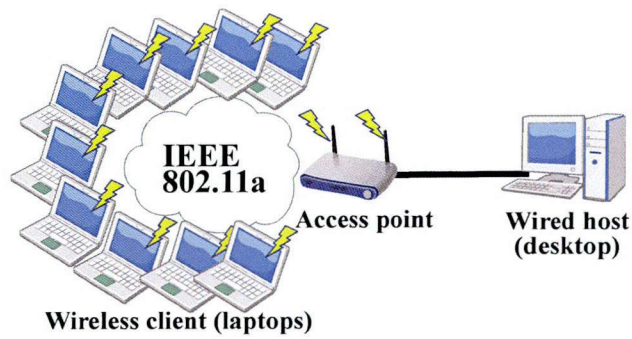
Two experimental environments are shown in Fig. 2.8. In both environments, ten wireless clients



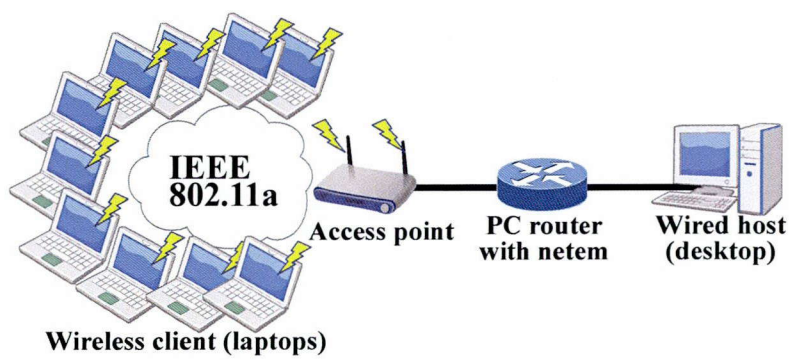
(a) Ten upstream flows

(b) Five upstream flows and five downstream flows

Figure 2.7: Average RTT of each flow



(a) Simple experimental environment



(b) Experimental environment with a PC router

Figure 2.8: Experimental environments

Table 2.2: WLAN products

(a) Wireless Interface Cards

Vendor	Product name
Buffalo	WLI-CB-AGHP
NEC	Aterm WL54AG

(b) Access Points

Vendor	Product name
Buffalo	WAPS-HP-AM54G54
NEC	Aterm WR8500N
Corega	CG-WLR300NNH

share a single AP. All clients were located within 50 [cm] of the AP in order to avoid packet losses due to wireless link error. In Fig. 2.8(a), a wired node is directly connected to the AP through a wired link. On the other hand, in the experimental environment in Fig. 2.8(b), we introduced a personal computer (PC) router between the AP and the wired node for the purpose of evaluation in long delay environments. DELL Latitude E5500 laptops and a DELL Precision 390 desktop were used as the client and the wired node, respectively. All nodes, including the wired node, used Ubuntu 8.10 (Linux kernel 2.6.28) as the operating system (OS). As shown in Fig. 2.8(b), another DELL Precision 390 desktop was used as the PC router with netem [59] for generating a 50 [ms] delay to the wired link between the AP and the wired node. We used Web100 [60] patch to collect the TCP connection information, such as the congestion window size and the RTT from the Linux kernel. We used TCP Reno and implemented the proposed method on the Linux code of TCP Reno.

The wireless devices listed in Table 2.2 were used as wireless interface cards for clients and the AP. Note that all wireless clients used the same type of wireless interface card in each experiment. We show only results obtained using Buffalo's wireless interface cards because the tendencies of results are similar regardless of the type of wireless interface cards. In the following, APs are abbreviated as *[vendor name]-AP*, respectively, e.g., Buffalo-AP.

The buffer size of an AP significantly affects on flow's throughput and end-to-end delay [61]. However, AP vendors do not make publish the buffer size in detail. Due to this, we estimated the buffer size through simple experiments with a single TCP connection, with an estimation equation as follows:

$$B_{est} = \left(cwnd_{overflow} - \frac{T \cdot minRTT}{8} \right) / MSS \tag{2.11}$$

Table 2.3: Estimated buffer size of each AP

Vendor	$minRTT$ [ms]	T [Mbps]	$cwnd_{overflow}$ [bytes]	B_{est} [packets]
Buffalo	0.975	21.8	118003	76.9
NEC	0.563	19.6	76719	50.2
Corega	0.674	20.2	366826	243.4

where B_{est} is the estimated buffer size [packets] of an AP, $cwnd_{overflow}$ is the congestion window size [bytes] of a single TCP connection when TCP detects data packet losses, regarding it as buffer overflow occurs at the AP, T is throughput [bps] just before TCP detects data packet losses, and $minRTT$ is a minimum RTT [seconds] during the experiments. Table 2.3 presents the results of estimated buffer size of each AP. Note that the experiments were conducted ten times for averaging the results. In Table 2.3, the buffer size of Corega-AP is the largest of all.

The experiments using the environments in Fig. 2.8 were conducted as follows. Only one TCP flow was generated for each client station using Iperf [62], assuming bulk data transfer. We kept the number of concurrent TCP flows at ten and changed the ratio of upstream and downstream TCP flows from (0, 10) to (10, 0). For the purposes of comparison, TCP connections used either the proposed method on TCP Reno or conventional TCP Reno. As in Sect. 2.5, the *thresh_ack_losses* parameter in the proposed method was set to one. The experiment time was set to 180 [seconds], and each TCP connection was generated almost simultaneously when the experiment started. More specifically, a computer separated from the experimental network in Fig 2.8, which are connected to each wireless client and the wired node through a separate wired network, simultaneously executed Iperf programs through the separate wired network in order to generate a TCP connection at each client and the wired node. We disabled vendor-specific functions implemented at APs, e.g., *Frame Burst* feature in Buffalo-AP. For each experimental setting, the experiments were conducted ten times in order to average the results.

We first conducted experiments using the environment shown in Fig. 2.8(a) in which the delayed ACK option was disabled at TCP receivers. In the experiments, the proposed method used the equation in Eq. (2.1) in order to estimate the number of ACK packet losses. On the other hand,

in order to investigate the effects using the equation in Eq. (2.4), we then conducted experiments using the environment in Fig. 2.8(b) with and without the delayed ACK option. In the situation, the smoothing factor β in Eq. (2.3) was set to $1/32$.

2.6.2 Evaluation Metric

The effectiveness of the proposed method is evaluated from the viewpoint of fairness and bandwidth utilization by assessing the throughput of each flow, and the index proposed in Sect. 2.4.

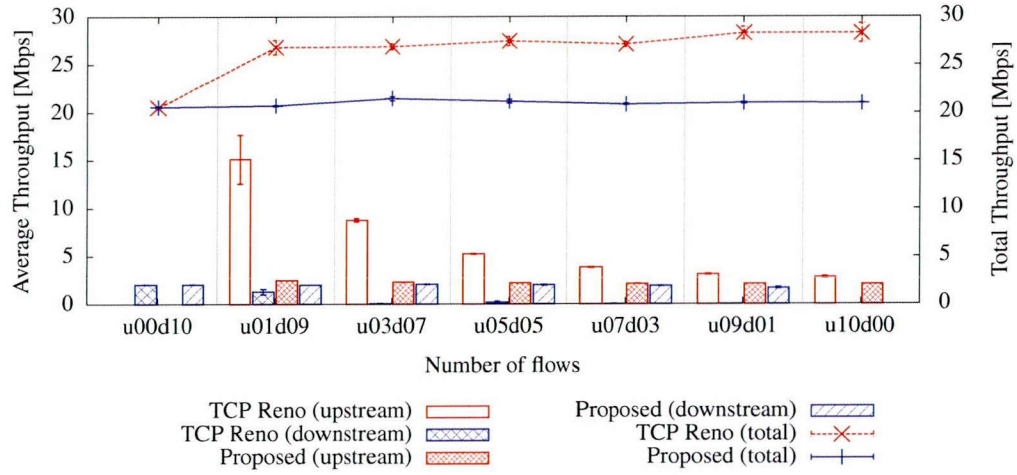
In order to evaluate fairness and trade-off relationships between fairness and bandwidth utilization, sliding window method (SWM) function [63] is applied to the proposed index. The SWM can give a quantitative measure of an arbitrary metric over a wide range of time scales. Especially, when using the SWM to measure fairness, it has an advantage of measuring *short-term fairness* and *long-term fairness* at the same time. Intuitively, short-term fairness of a data transmission flow refers to its ability to provide equitable access to resources to all the contending flows over short time scales. In contrast, long-term fairness measures the average amount of resources assigned over a longer time. The SWM function applied to Eq. (2.10) is calculated as

$$SWM(w) = \frac{C^2}{n\sum_{i=1}^{n} x_i(w)^2 - 2C\sum_{i=1}^{n} x_i(w) + 2C^2} \tag{2.12}$$

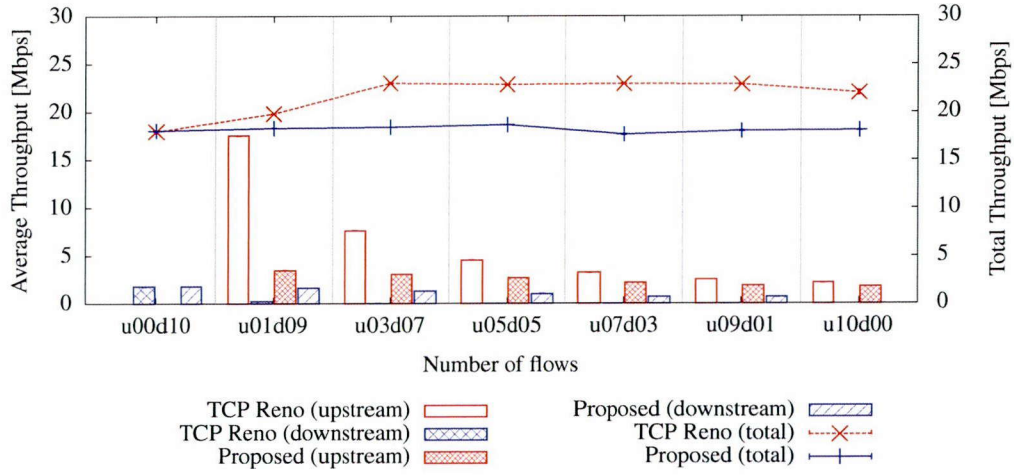
where w is a time-window size for evaluating the fairness, and $x_i(w)$ is an average throughput of flow i in a time-window w. In this chapter, parameter C in Eq. (2.12) is set to 29.60 [Mbps] according to the theoretical maximum throughput of IEEE 802.11a WLAN with 1460 [bytes] maximum transmission unit (MTU) [64].

2.6.3 Evaluation Results and Discussion

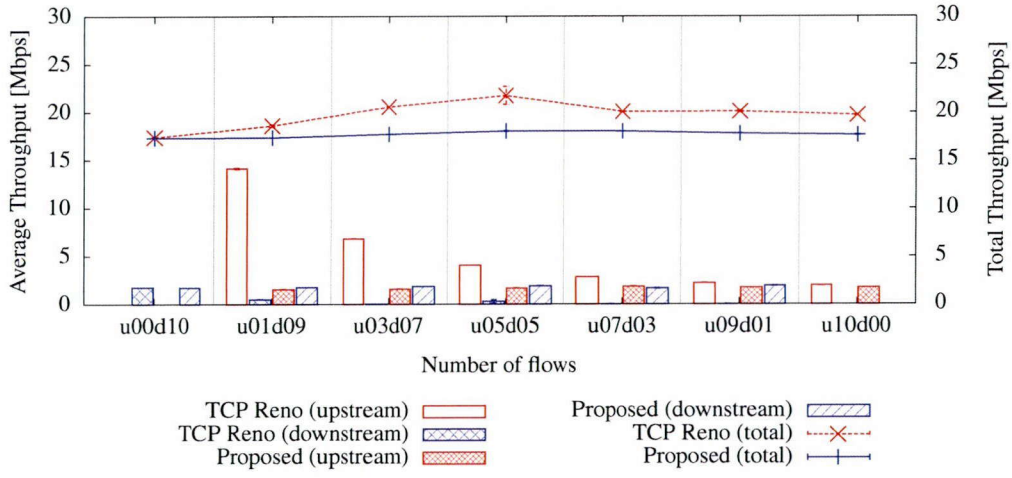
Figure 2.9 presents the average throughput of upstream and downstream flows, and the total throughput for various ratios of upstream and downstream flows and three types of APs. In this figure, uxdy in x-axis denotes that the number of upstream and downstream TCP flows are x and y, respectively. The lines in these figures represent total throughput, and the bars represent average throughput of upstream and downstream flows, respectively. When the bars are the same height, it means the



(a) Buffalo-AP



(b) NEC-AP



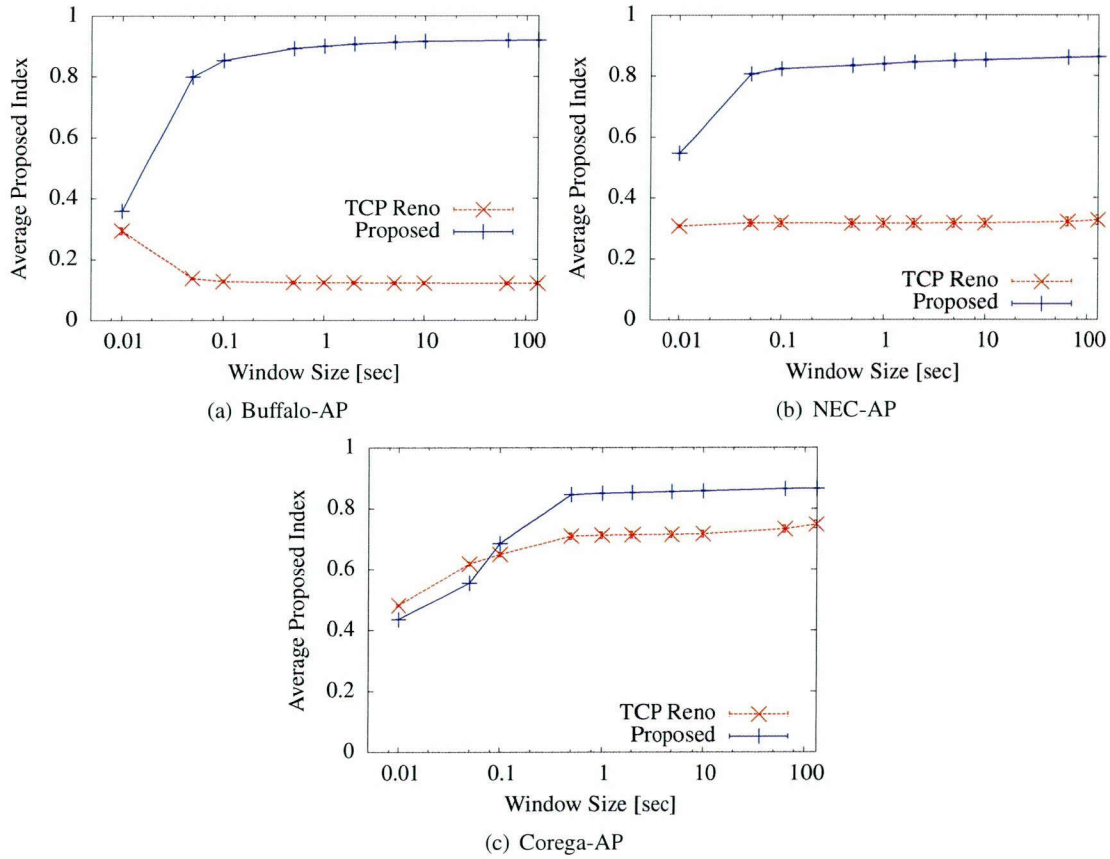
(c) Corega-AP

Figure 2.9: Effect of the number of upstream and downstream flows

perfect fairness between upstream and downstream flows. The average throughput of each flow is calculated using the amount of data transmitted during 50-180 [seconds] in the experiment. Figure 2.9 shows that, when at least one upstream flow with TCP Reno exists in the network, the upstream flows occupy almost all of the network bandwidth, and the downstream flows are starved. On the other hand, the proposed method can significantly improve the throughput fairness between upstream and downstream flows, and no flow is starved. Figure 2.9 reveals that the degree of fairness improvement of the proposed method is small when using NEC-AP. The reason for this is as follows. Flows with the proposed method do not experience RTO when using Buffalo-AP or Corega-AP. In contrast, flows with the proposed method experience RTO when using NEC-AP because the buffer size of NEC-AP is too small for ten flows to flow. However, even when using NEC-AP, the proposed method can avoid starvation of flows.

In terms of total throughput, the total throughput of ten downstream flows with TCP Reno and with the proposed method are equivalent, regardless of type of AP. This is because ACK packets are not discarded at the APs and the behavior of TCP with and without the proposed method is identical. However, comparing the total throughput of the proposed method and TCP Reno when there exist one or more upstream flows, the former is smaller than the latter. The reason for this is as follows. When using TCP Reno, packet transmissions are stopped at TCP-level in the starved flows. Thus, only a few clients that have a non-starved flow can send packets at the MAC-level, which produces the low probability of frame collisions. On the other hand, when the proposed method is used, all ten wireless clients can transmit packets at MAC-level, which results in the high probability of frame collisions. As a result, the total throughput decreases when using the proposed method. Therefore, we should consider a trade-off relationship between fairness and bandwidth utilization.

Figures 2.10 and 2.11 show the evaluation results obtained using the proposed metric with SWM when ten upstream flows exist and when five upstream flows and five downstream flows coexist, respectively, under the same conditions as Fig. 2.9. When we use Buffalo-AP or NEC-AP, the index values of the proposed method are significantly better than that of TCP Reno in terms of not only long-term fairness but also short-term fairness when one or more upstream flows exist in the network. On the other hand, the index value of TCP Reno when using Corega-AP is better



(a) Buffalo-AP

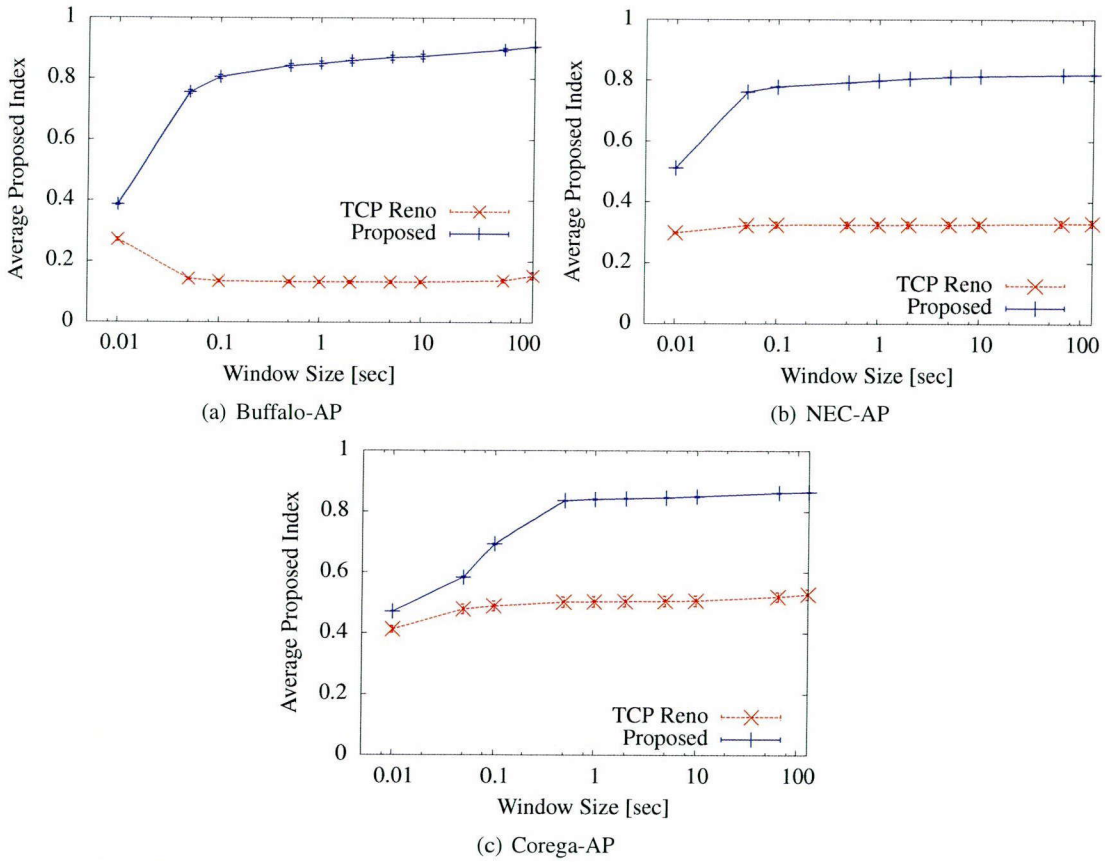
(b) NEC-AP

(c) Corega-AP

Figure 2.10: Proposed index with SWM when ten upstream flows exist

than that when using the other APs. This is due to the large buffer size of Corega-AP as shown in Table 2.3. More specifically, as more the buffer size of an AP becomes large, the more packets of each flow are maintained at the AP. Because this leads the number of RTO occurred to reduce, the degree of unfairness is slightly alleviated.

In order to investigate the effect of using the delayed ACK option and using the equation in Eq. (2.4), we conducted experiments with and without the delayed ACK option under the environment shown in Fig. 2.8(b). Figure 2.12 shows the average throughput of upstream and downstream flows and the total throughput for various ratios of upstream and downstream flows in the experimental environment of Fig. 2.8(b) using Buffalo-AP with a 50 [ms] delay. In Fig. 2.12, the results



(a) Buffalo-AP

(b) NEC-AP

(c) Corega-AP

Figure 2.11: Proposed index with SWM when five upstream flows and five downstream flows coexist

obtained using and without using the delayed ACK option are labeled as *delack* and *nodelack*, respectively. From the viewpoint of fairness, TCP Reno experiences serious unfairness among upstream and downstream flows, regardless of the use of the delayed ACK option, whereas the proposed method significantly improves fairness with or without the delayed ACK option. Furthermore, both the total throughput of TCP Reno and the proposed method increase when using the delayed ACK option. This is because the delayed ACK option decreases the number of ACK packets in the WLAN and that consequently increases the number of data packets injected into the WLAN. Therefore, when using the delayed ACK option, the proposed method can enhance the total throughput without degrading the improvement in fairness.

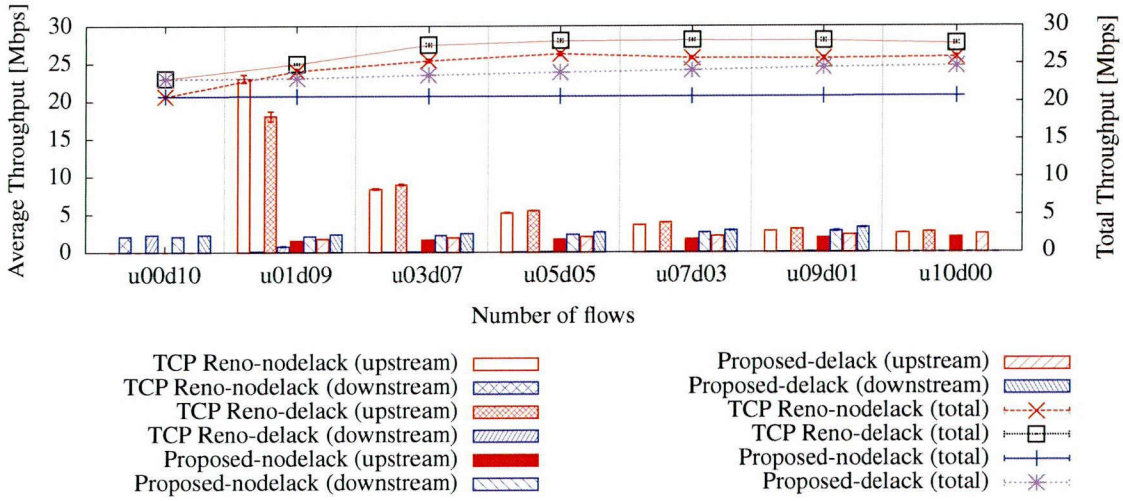


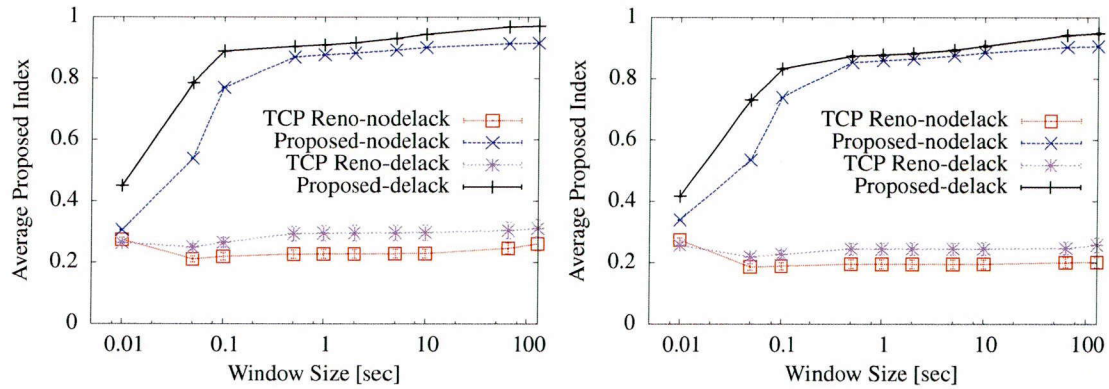
Figure 2.12: Effect of the delayed ACK option when using Buffalo-AP with a 50 [ms] one-way delay

Figure 2.13 shows the evaluation results obtained using the proposed index under the same conditions as Fig. 2.12. Comparing the index values of the proposed method with and without using the delayed ACK option, the index value with the delayed ACK option is better than that without the delayed ACK option. This means that, in the proposed method, the improvements in bandwidth utilization leads to the improvement in the index values.

2.7 Summary

In this chapter, we first proposed a transport-layer solution for alleviating TCP unfairness in a WLAN environment. We then proposed a novel performance metric for evaluating the trade-off relationship between fairness and bandwidth utilization at a network bottleneck. The proposed index is based on the variations in throughput of concurrent flows and the ideal throughput distribution in which all flows achieve the same throughput and the network bandwidth is fully utilized.

In order to confirm the basic characteristics of the proposed method, we conducted simulation experiments. Based on the results of the simulation experiments, we confirmed that the proposed method can alleviate TCP unfairness among upstream flows and between upstream and downstream



(a) Ten upstream flows

(b) Five upstream flows and five downstream flows

Figure 2.13: Proposed index with SWM when using Buffalo-AP with a 50 [ms] one-way delay

flows while maintaining low RTT. Through extensive experiments using real WLAN environments with the products from several vendors, we then confirmed that the proposed method alleviates TCP unfairness regardless of the vendor of the APs and wireless interface cards. Moreover, the proposed method with the delayed ACK option enhances the total throughput without degrading the effectiveness of fairness improvement. Through trade-off evaluations using the proposed metric, we also demonstrated that the proposed method can achieve a markedly better trade-off relationship between fairness and bandwidth utilization.

Finally, we make mention of directions for the widespread use of the proposed method. To alleviate TCP unfairness with the proposed method, we need to implement the proposed method to TCP of all wireless clients that have upstream TCP flows. When TCP flows with and without the proposed method coexist in a WLAN, TCP unfairness is not alleviated and the proposed method may decrease its TCP throughput due to the behavior of its congestion control mechanisms. However, such a decrease in TCP throughput can be avoided by adjusting thresh_ack_losses, which is a parameter of the proposed method. Therefore, one of feasible scenarios to introduce the proposed method is that the proposed method gradually accommodates to TCP variants on wireless clients with software updates.

In the future, we intend to evaluate the proposed method in environments that include wired networks with several traffic scenarios.

Chapter 3

Energy Consumption Models of a Single TCP Flow in a WLAN

3.1 Introduction

With recent developments in wireless network technologies, the Internet is increasingly accessed via IEEE 802.11-based wireless LANs (WLANs) by using mobile devices, such as mobile phones, smartphones, laptops, and tablet PCs. Wireless communication through a mobile device can account up to 50% of the device's total energy consumption [17–19]. Therefore, there is a great deal of interest in reducing the energy consumed through wireless communication, particularly because most mobile devices are battery-driven.

For energy saving in media access control (MAC) layer protocols, the IEEE 802.11 standard defines a power saving mode (PSM) [6], as opposed to the mode under normal operation, which is referred to as the continuously active mode (CAM). Although PSM can significantly reduce energy consumption, it can also degrade network performance characteristics, such as throughput and latency [2]. Many researchers have proposed energy-efficient methods in WLANs [2, 27–35]. Some of them achieve high-energy efficiency by mainly modifying MAC protocols. The others are energy-efficient solutions for the specific applications. In contrast, in this chapter we aim to derive general-purpose transport-layer solution for energy saving without requiring any modifications for

wireless network interface (WNI) hardware.

In order to investigate the factors to affect energy consumption of wireless clients in WLANs, many researchers have constructed energy consumption models for WLAN clients [20–24]. Anastasi *et al.* [20] modeled a single wireless client in PSM downloading a file from a server in the presence of multiple wireless clients. Ergen and Varaiya [21] presented the results of an analysis of energy consumption during different MAC operations for a wireless client with multiple clients in a WLAN; they found that 80 % of the total energy in saturated situations is wasted. Kuo [22] modeled the energy consumption of wireless clients based on Markov chain model in a saturated WLAN. Rantala *et al.* [23] modeled power consumption of a WNI, which consists of the power consumption of application-specific integrated circuit (ASIC) and that of the power amplifier (PA). Agrawal *et al.* [24] created a discrete-time Markov chain model of the energy consumption for TCP transfers in CAM and in PSM in the presence of TCP background traffic. However, most of the above researches mainly focus on the behavior at only the MAC level and do not consider detailed TCP behavior. Because TCP congestion control mechanisms primarily determine the timing of packet transmission and reception, analysis of the detailed TCP behavior is necessary to assess sleep efficiency at packet interarrivals.

Other researchers modeled energy consumption based on the behavior of the TCP congestion mechanisms under the existence of channel error [25, 26]. Zorzi and Rao [25] modeled energy consumption for different TCP versions while considering a two-state Markov model for frame losses. Vacirca *et al.* [26] constructed an energy consumption model that consists of combination of a TCP behavior model and a MAC model under a Rayleigh fading channel. These two models consider energy consumption for a device activated in CAM. In this chapter, we mainly focus on the impact of the TCP behavior for sleep efficiency when the WNI sleeps at packet interarrivals.

Therefore, to assess the impact of TCP on sleep efficiency, in this chapter we construct a new energy consumption model for upstream flow in TCP data transfer over a WLAN. The proposed model consists of the combination of two layer models: a MAC-level model and a TCP-level model. In the MAC-level model, the amounts of energy consumed in sending and receiving one data frame are calculated based on frame exchanges in CSMA/CA mechanisms. In the TCP-level model, which is based on the behavior of TCP congestion control mechanisms, the energy consumption in TCP

data transfer is determined from packet transmission and reception timing, depending on the growth of the TCP congestion window size. We derive energy consumption models for a device with CAM and with *ideal sleep* mode to reveal the sleep efficiency while considering detailed TCP behavior. Ideal sleep mode implies that a WNI knows the schedules of both the transmission and reception of TCP packets such that it can sleep and wake up with exact timing. Furthermore, in order to improve the sleep efficiency, we propose TCP-level *burst transmission* behavior, which reduces the number of state transitions between active and sleep modes by transmitting multiple data packets in groups. Note that we assume that the available bandwidth of the WLAN is larger than that of the wired network, which is a necessary condition for sleeping at packet interarrivals in the WLAN. In this situation, frame losses due to collision or channel error are moderate in the WLAN to obtain the sufficient available bandwidth in the WLAN. Therefore, we consider a simple frame error model in the WLAN against the above models [25, 26].

By comparing the energy consumption between devices with CAM and with ideal sleep mode, we reveal the upper bound of energy savings under these sleep strategies. We then demonstrate the energy efficiency of TCP-level burst transmission and discuss the trade-off between energy efficiency and TCP latency. Finally, we discuss the factors contributing to efficient energy saving in wireless network devices. Based on this discussion, design guidelines for energy-efficient WNIs are presented.

The remainder of Chapter 3 is organized as follows. First, we describe the network model and our assumptions in Sect. 3.2. Then, in Sect. 3.3, our models of energy consumption during TCP data transfer over a WLAN are introduced. Section 3.4 describes the simulation-based validation for our models. Section 3.5 shows numerical results of the analysis based on our models, and we discuss on factors for energy efficiency in Sect. 3.6. Finally, conclusions and future research directions are presented in Sect. 3.7.

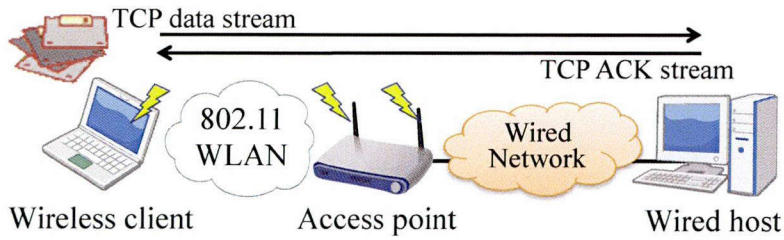


Figure 3.1: WLAN environment

3.2 Network Model and Assumptions

Suppose we have a WLAN environment in which a single wireless client associates with an access point (AP) connected to a host via a wired network (Fig. 3.1)[1]. In the WLAN, the wireless client sends a file of S_d bytes to the wired host by TCP; that is, we consider upstream TCP data transfer. Note that our model can be easily adjusted to deal with downstream TCP data transfer. In Sect. 3.3, energy consumption is taken to be from the transmission of the first packet of the file until the reception of the ACK packet for the last packet of the file. We assume that at the MAC level, RTS/CTS mechanisms are used by the wireless client when transmitting data frames to the AP, whereas the AP does not utilize RTS/CTS when transmitting data frames to the wireless client.

Suppose that at the hardware level the WNI has four communication modes — *transmit*, *receive*, *idle* or *listen*, and *sleep* — and each of these modes has a different power consumption denoted by P^t, P^r, P^l, and P^s, respectively [17]. Furthermore, the WNI consumes power when transiting between active and sleep modes, and we define P^{as} and P^{sa} as the power consumption when changing from and to active mode, respectively. The duration of these power consumptions is then denoted by T^{as} and T^{sa}, respectively. In order to focus on the impact on sleep efficiency by TCP, we assume ideal sleep mode, which means that a WNI knows the schedules of both the transmission and reception of TCP packets such that it can sleep and wake up with exact timing.

We assume that data frames are lost randomly at MAC-level of the WLAN due to channel error and collisions. In addition, TCP data packets are lost in the wired network due to network

[1] In this chapter, we do not consider situations in which multiple clients share a single AP. One of the purposes of this chapter is to reveal how TCP behavior affects sleep efficiency. Therefore, issues that should be solved at MAC-level, e.g., collisions and overhearing in the existence of multiple clients, are out of scope of this chapter.

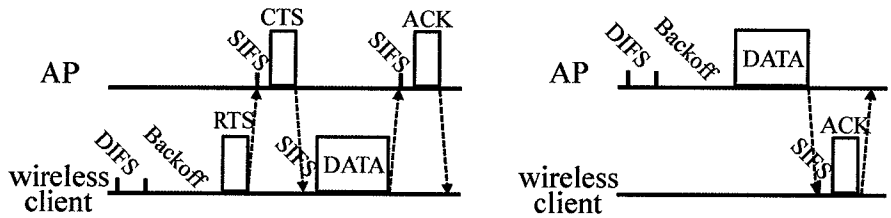
congestion. For simplicity, the probability of frame transmission failures at MAC-level, q, and the probability of TCP-level packet loss events in the wired network, p_l, are given. Let p_w denotes the probability of TCP-level packet loss events in the WLAN, which is calculated as $p_w = q^{N+1}$ where N is the maximum number of frame retransmissions at MAC-level. Therefore, the TCP at the wireless client experiences the packet loss events at a probability of p, which is defined as $p = 1 - (1 - p_w)(1 - p_l)$.

Other assumptions are as follows.

- No TCP ACK packets are lost in the wired network.
- Energy consumption due to retransmission of data packets is not considered because it has a small impact on sleep efficiency.
- TCP-level burst transmission is achieved by using TCP delayed ACK [53]. When the delayed ACK is used, growth of the TCP congestion window has been reported to be inhibited [65]. In this chapter, we assume that this problem has been resolved.
- Unless otherwise noted, we follow the assumptions of Padhye *et al.* [66] and Cardwell *et al.* [67] for TCP congestion control behavior.

3.3 Energy Consumption Models

In this section, we formulate our models which describe expected energy consumed at the WNI of a wireless client during TCP transfer of S_d bytes data. The behavior of MAC-layer protocols determines the energy consumption when a single data frame is transmitted and received, whereas the number of data packets sent and received per RTT is dependent on the behavior of TCP congestion control mechanisms. Thus, using the MAC-level model developed in Subsect. 3.3.1, we derive an energy consumption model in TCP data transfer in Subsect. 3.3.2.



(a) From client to AP with RTS/CTS mechanisms

(b) From AP to client without RTS/CTS mechanisms

Figure 3.2: Frame exchange in IEEE 802.11 MAC

3.3.1 Energy Consumption during Frame Exchanges for IEEE 802.11 MAC

We first present a model of energy consumption at the MAC level when a wireless client transmits or receives one data frame[2]. Figure 3.2 shows details of the frame exchanges between a wireless client and an AP. To send one data frame to the AP by using RTS/CTS mechanisms (Fig. 3.2(a)), the wireless client first exchanges RTS/CTS frames after a random backoff time. After that, it sends the data frame and receives the corresponding ACK frame. In contrast, when the AP sends one data frame to the client using a non-RTS/CTS mechanism (Fig. 3.2(b)), it sends the data frame immediately after a random backoff time.

In both of the above sequences, the expected backoff time of ith transmission after $(i-1)$ consecutive transmission failures is determined by the following equations:

$$T_{BO}(i) = CW(i)T_{slot}/2 \tag{3.1}$$

where T_{slot} is the slot time and $CW(i)$ is the contention window size of ith transmission after $(i-1)$ consecutive transmission failures. $CW(i)$ is given by

$$CW(i) = \min\left((CW_{min}+1)2^{i-1}-1,\ CW_{max}\right) \tag{3.2}$$

where CW_{min} and CW_{max} are the minimum and maximum values of the contention window size,

[2]Since we consider upstream TCP data transfer, TCP data and TCP ACK packets are contained in the data frame sent from and received at the client, respectively.

respectively.

From Fig. 3.2(a) and Eq. (3.1), the average time for the wireless client to send one data frame for the ith transmission, $t^t(i)$, is calculated as follows:

$$t^t(i) = 3T_{SIFS} + T_{DIFS} + T_{BO}(i) + 4\tau + T_{RTS} + T_{DATA}^{client} + T_{CTS} + T_{ACK} \tag{3.3}$$

where T_{SIFS} is the short interframe space (SIFS), T_{DIFS} is the distributed interframe space (DIFS). T_{RTS} and T_{CTS} are the transmission duration of the RTS and CTS frame, respectively. T_{DATA}^{client} is the transmission duration of a data frame, T_{ACK} is the reception duration of an ACK frame, and τ is the radio propagation delay between the wireless client and the AP.

Similarly, the average time by the wireless client to receive one data frame for the ith transmission, $t^r(i)$, is

$$t^r(i) = T_{SIFS} + T_{DIFS} + T_{BO}(i) + 2\tau + T_{DATA}^{AP} + T_{ACK} \tag{3.4}$$

where T_{DATA}^{AP} is the reception duration of a data frame sent from the AP.

From Eqs. (3.3) and (3.4), we derive the expected values of $t^t(i)$ and $t^r(i)$, denoted by $E[t^t]$ and $E[t^r]$, respectively. Using q and N, the probability that a data frame is transmitted i times can be calculated as follows.

$$Q(i) = \begin{cases} q^{i-1}(1-q) & \text{if } i \leq N \\ q^N & \text{if } i = N+1 \end{cases} \tag{3.5}$$

Since the time in which the ith transmission becomes successful after $(i-1)$ failures is given by $\sum_{j=1}^{i} t^t(j)$, from Eqs. (3.3), (3.4), and (3.5), $E[t^t]$ and $E[t^r]$ are given by

$$E[t^t] = \sum_{i=1}^{N+1} \sum_{j=1}^{i} t^t(j)Q(i), \; E[t^r] = \sum_{i=1}^{N+1} \sum_{j=1}^{i} t^r(j)Q(i). \tag{3.6}$$

Using $E[t^t]$ and $E[t^r]$, we calculate the maximum number of packets injected into the network per RTT at the TCP-level, which is denoted by W_{bdp}. Now, let m denote the number of data

packets sent in a bursty fashion in TCP-level burst transmission. When the delayed ACK is used, the TCP receiver sends an ACK packet every received m data packets. Therefore, when the burst transmission is used, the number of packets sent per unit time at the TCP-level is calculated as $1/(E[t^t] + E[t^r]/m)$. Then, W_{bdp} is calculated as

$$W_{bdp} = RTT/(E[t^t] + E[t^r]/m) \tag{3.7}$$

where RTT is average RTT.

Next, we determine the expected energy consumed during transmission ($E[e^t]$) and reception ($E[e^r]$) of one data frame. From Fig. 3.2(a) and Eq. (3.1), the energy consumption for the ith data frame transmission after $(i-1)$ failures is obtained by

$$\begin{aligned} e^t(i) =& P^l(3T_{SIFS} + T_{DIFS} + T_{BO}(i) + 4\tau) \\ &+ P^t(T_{RTS} + T_{DATA}^{client}) + P^r(T_{CTS} + T_{ACK}). \end{aligned} \tag{3.8}$$

In a similar way, the energy consumption for the ith data frame reception after $(i-1)$ failures is

$$e^r(i) = P^l(T_{SIFS} + T_{DIFS} + T_{BO}(i) + 2\tau) + P^t T_{ACK} + P^r T_{DATA}^{AP}. \tag{3.9}$$

Using Eqs. (3.5), (3.8), and (3.9), $E[e^t]$ and $E[e^r]$ are then calculated as follows:

$$E[e^t] = \sum_{i=1}^{N+1} \sum_{j=1}^{i} e^t(j)Q(i), \; E[e^r] = \sum_{i=1}^{N+1} \sum_{j=1}^{i} e^r(j)Q(i). \tag{3.10}$$

3.3.2 Energy Consumption of TCP Data Transfer

We now construct a TCP-level model of energy consumption during TCP data transfer. Expected energy consumed during the whole data transfer is calculated by the sum of energy consumption of each state of a WNI. Thus, the expected energy consumption with CAM during the whole TCP data

transfer, $E[J_{cam}]$, is obtained as

$$E[J_{cam}] = E[J^t] + E[J^r] + E[J^l_{cam}] \tag{3.11}$$

where $E[J^t]$ and $E[J^r]$ are the expected energy consumed in transmission and reception of TCP packets, respectively, and $E[J^l_{cam}]$ is the expected energy consumption during the duration in which no packet is sent nor received, which is referred to as idle duration. On the other hand, the expected energy consumption with sleeping during the whole TCP data transfer, $E[J_{sleep}]$, is given by

$$E[J_{sleep}] = E[J^t]+E[J^r]+E[J^l_{sleep}]+E[J^s]+E[J^{st}] \tag{3.12}$$

where $E[J^l_{sleep}]$ is the expected energy consumption during the idle duration, $E[J^s]$ is the expected energy consumption during sleep mode, and $E[J^{st}]$ is the expected energy consumed due to state transitions between active and sleep modes.

In what follows, we derive each term in Eqs. (3.11) and (3.12) by utilizing TCP analysis models [66, 67]. Padhye *et al.* [66] formulated the average TCP throughput analyzed based on the detailed behavior of TCP congestion control mechanisms. By extending [66], Cardwell *et al.* [67] derive the expected TCP latency of finite-size data transfer. The derivations of both models are based on a *round* that starts when the first packet of a window is transmitted and ends when the corresponding ACK packet is received. Note that a round makes a RTT. We determine the terms in Eqs. (3.11) and (3.12) by using the expected number of rounds during the data transfer of S_d bytes which is derived in [66, 67].

In the following analysis, in Subsect. 3.3.2.1 we first formulate the number of rounds required for S_d bytes data transfer. We then derive $E[J^t]$ and $E[J^r]$ in Subsect. 3.3.2.2. In addition, $E[J^l_{cam}]$ and $E[J^l_{sleep}]$ are formulated in Subsect. 3.3.2.3. Subsections 3.3.2.4 and 3.3.2.5 show the derivations of $E[J^s]$ and $E[J^{st}]$, respectively. Finally, we formulate the increase in data transfer latency when bust transmission is used in Subsect. 3.3.2.6.

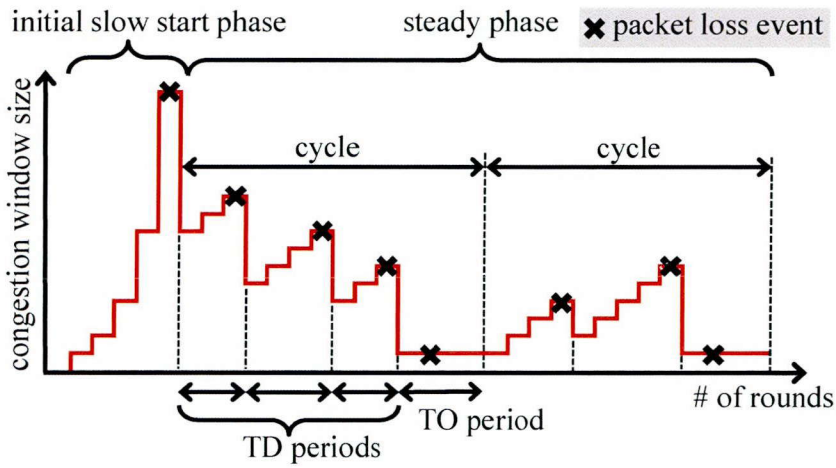


Figure 3.3: An example of evolution of congestion window size

3.3.2.1 Number of Rounds during TCP Data Transfer

In TCP data transfer, the number of packets sent per round is determined by the growth of the TCP congestion window size. Figure 3.3 depicts the typical evolution of the congestion window size of a TCP connection from the beginning of the transmission. The data transfer starts with the *initial slow start phase* that ends due to the occurrence of a packet loss event. After that, data packets are sent in the *steady phase* until the data transfer ends. TCP continues increasing its congestion window size until a packet loss event is detected and decreases it after detecting the packet loss event. The packet loss events are detected either by triple duplicate ACK packets or by retransmission timeouts (RTOs). Here, we define a triple duplicate (TD) period as the duration between two consecutive packet loss events detected by triple duplicate ACK packets. Further, the duration of a sequence of RTO is referred to as a timeout (TO) period. In the steady phase, we can observe one TO period appears after multiple TD periods, and this sequence appears repeatedly [66]. Here, we define a *cycle* as the duration between two consecutive packet loss events detected by RTOs. Then, the steady phase is constructed by multiple cycles.

We mainly follow the TCP latency formulations derived in [66, 67] to calculate the expected number of rounds required for the whole data transfer. Here, let $E[r_\zeta]$ and $E[r_\eta]$ denote the expected numbers of rounds during the initial slow start phase and during the steady phase, respectively.

Then, the expected number of rounds during the whole data transfer is obtained as $(E[r_\zeta] + E[r_\eta])$. Refer to 3.A for the derivations of $E[r_\zeta]$ and $E[r_\eta]$. However, [66,67] assume situations in which the packets sent per round is constrained by the maximum window size, whereas we assume situations in which the packets sent per round is constrained by the wireless network bandwidth. Due to this, we modify some equations in [66, 67].

The modification of the constraint affects the evolution of the congestion window size. When the number of packets sent per round is constrained by the maximum window size, the evolution of the congestion window size stops when the congestion window size reaches the maximum window size. However, when the number of packets sent per round is constrained by the wireless network bandwidth, the congestion window continues increasing while the number of packets injected into the network is limited by the bandwidth. In the following, considering the above situations, we derive the expected numbers of rounds during the data transfer of the initial slow start phase ($E[r_{ss}]$) and during the TD period ($E[r_{td}]$).

We first derive $E[r_{ss}]$. $E[r_{ss}]$ can be obtained simply by replacing the term of maximum window size of the equation derived in [67] with W_{bdp}, $E[r_{ss}]$ is calculated as

$$E[r_{ss}] = \begin{cases} \log_2\left(\frac{E[d_{ss}]}{w_1} + 1\right) & \text{if } E[W_{ss}] \leq W_{bdp} \\ \log_2\left(\frac{W_{bdp}}{w_1}\right) + 1 + \frac{1}{W_{bdp}}\left(E[d_{ss}] - 2W_{bdp} + w_1\right) & \text{otherwise} \end{cases} \tag{3.13}$$

where $E[d_{ss}]$ is the expected number of packets sent during the initial slow start phase and w_1 is the initial window size.

We then calculate $E[r_{td}]$. Let $E[W]$ be the expected window size when a packet loss event occurs in the TD period. Then, the number of packets sent per round is dependent on the relation between $E[W]$ and W_{bdp}, thereby changing $E[r_{td}]$. Therefore, we obtain $E[r_{td}]$ by dividing into three cases: $E[W] \leq W_{bdp}$, $W_{bdp} < E[W] \leq 2W_{bdp}$, and $2W_{bdp} < E[W]$. When $E[W] \leq W_{bdp}$, $E[r_{td}] = (E[W]/2+1)$ [66]. When $W_{bdp} < E[W] \leq 2W_{bdp}$, the expected number of packets sent per round increases from $E[W]/2$ up to W_{bdp}, and after that, it is fixed to W_{bdp} until the TD period ends. On the other hand, when $2W_{bdp} < E[W]$, it is constantly W_{bdp} for all rounds. Following

calculation processes [66], $E[r_{td}]$ is calculated as

$$E[r_{td}] = \begin{cases} \frac{E[W]}{2} + 1 & \text{if } E[W] \leq W_{bdp} \\ \frac{1-p}{pW_{bdp}} + \frac{E[W](E[W]+8)}{8W_{bdp}} + \frac{W_{bdp}-E[W]}{2} + \frac{3}{4} & \text{if } W_{bdp} < E[W] \leq 2W_{bdp} \\ \frac{1-p}{pW_{bdp}} + \frac{E[W]}{W_{bdp}} + \frac{3}{4} & \text{otherwise.} \end{cases} \quad (3.14)$$

3.3.2.2 Energy Consumption of Packet Transmission and Reception

Let $E[n^t]$ and $E[n^r]$ denote the expected number of packets sent and received during the data transfer of S_d bytes, respectively. Then, $E[J^t]$ and $E[J^r]$ are obtained as

$$E[J^t] = E[n^t]E[e^t], \quad E[J^r] = E[n^r]E[e^r]. \quad (3.15)$$

For simplicity, the number of retransmitted packets is not counted. Then, $E[n^t]$ is given by S_d/S_p where S_p is the TCP data packet size. On the other hand, the number of received packets equals that of received ACK packets, which depends on timing of the occurrence of packet loss events. Therefore, $E[n^r]$ is calculated as follows.

Let $E[n^r_\zeta]$ and $E[n^r_\eta]$ denote the expected numbers of ACK packets received during the initial slow start phase and during the steady phase, respectively. Then, $E[n^r] = E[n^r_\zeta] + E[n^r_\eta]$.

We next determine $E[n^r_\zeta]$. In the initial slow start phase, $E[d_{ss}]$ data packets are transmitted. Let $E[W_{ss}]$ be the expected window size when a packet loss event occurs in the initial slow start phase. When packet loss events are detected by triple duplicate ACK packets, we assume that $E[W_{ss}]/2$ of $E[d_{ss}]$ data packets are lost averagely[3]. In contrast, when packet loss events are detected by RTOs, $E[W_{ss}]$ data packets are lost. When m data packets are sent in a bursty fashion at a time by using the delayed ACK, the TCP sender receives an ACK packet every m data packets sent from it. From the above discussion, in a similar way to the calculation process for Eq. (3.50), $E[n^r_\zeta]$ is obtained as

$$E[n^r_\zeta] = \left(E[d_{ss}] - l_{ss}(1 + Q(E[W_{ss}], p)) \frac{E[W_{ss}]}{2} \right) / m \quad (3.16)$$

[3]This assumption follows [66].

where $Q(w,p)$ is the probability that a packet loss is detected by a RTO as functions of window size w and probability of packet loss events p and l_{ss} is the probability that at least one packet is lost in the initial slow start phase.

Similarly, $E[n_\eta^r]$ is obtained as follows. $E[W]/2$ packets of $\left(\frac{1-p}{p} + E[W]\right)$ packets sent during the TD period are lost averagely. Thus, $\left(\frac{1-p}{p} + E[W]/2\right)$ ACK packets are received in the TD period. According to the calculation process for Eq. (3.60), $E[n_\eta^r]$ is calculated as

$$E[n_\eta^r] = E[n_{cycle}]E[n]\left(\frac{1-p}{p} + \frac{E[W]}{2}\right)\frac{1}{m} \tag{3.17}$$

where $E[n_{cycle}]$ is the expected number of cycles in the steady phase and $E[n]$ is the expected number of TD periods in a cycle.

3.3.2.3 Energy Consumption during Idle Duration

$E[J_{cam}^l]$ and $E[J_{sleep}^l]$ are calculated by multiplying P^l by the expected length of idle duration during the whole data transfer which is obtained by subtracting the expected duration of the other states from the expected duration of the whole data transfer, $E[T^{all}]$. As a result, $E[J_{cam}^l]$ and $E[J_{sleep}^l]$ are obtained as

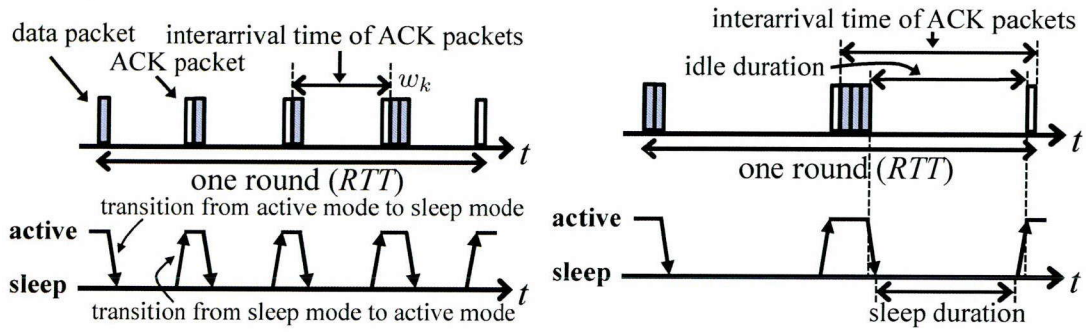
$$E[J_{cam}^l] = P^l\left(E[T^{all}] - E[T^t] - E[T^r]\right), \tag{3.18}$$

$$E[J_{sleep}^l] = P^l\left(E[T^{all}] - E[T^t] - E[T^r] - E[T^s] - E[T^{st}]\right) \tag{3.19}$$

where $E[T^t]$, $E[T^r]$, $E[T^s]$, and $E[T^{st}]$ are the expected duration of packet transmission, of packet reception, of sleeping, and of state transitions during the whole data transfer, respectively. $E[T^s]$ and $E[T^{st}]$ are derived in Subsects. 3.3.2.4 and 3.3.2.5, respectively.

$E[T^{all}]$, $E[T^t]$ and $E[T^r]$ are obtained as follows. $E[T^{all}]$ is derived as

$$E[T^{all}] = RTT \cdot \{E[r_\zeta] + E[r_\eta]\}. \tag{3.20}$$



(a) $m = 1$ (b) $m = 2$

Figure 3.4: Packet sequence of transmission and reception and state transitions of WNI where $w_k = 5$

In a similar form to Eq. (3.15), $E[T^t]$ and $E[T^r]$ are written by

$$E[T^t] = E[n^t]E[t^t], \quad E[T^r] = E[n^r]E[t^r]. \tag{3.21}$$

3.3.2.4 Energy Consumption during Sleep Duration

In this subsection, we derive $E[J^s]$ based on a bottom-up approach. Figure 3.4 represents the packet sequence of transmission and reception, and state transitions of WNI in one round when $w_k = 5$ where w_k is the congestion window size of the kth round in either the initial slow start phase or the TD period of the steady phase. Note that sleeping with $m = 1$ means sleeping without burst transmission. In an interarrival time of ACK packets that arrive at the WNI of the client, m data packets are sent and a single ACK packet is received. Then, the idle duration is obtained by subtracting packet transmission and reception times from the interarrival time. Furthermore, subtracting the time for state transitions between active and sleep modes, we can obtain sleep duration in the interarrival time of ACK packets.

We first consider an interarrival time of ACK packets. With the delayed ACK, the number of interarrivals of ACK packets in the kth round is $\lceil w_{k-1}/m \rceil$. Assuming ACK packets in a window are arrived at the client at equal intervals, an interarrival time of ACK packets, $t^a(w_k)$, is obtained

as

$$t^a(w_k) = RTT/\lceil w_{k-1}/m \rceil . \tag{3.22}$$

Here, w_k for the data transfer of the initial slow start phase and for the TD period are denoted by w_k^{ss} and w_k^{ca}, respectively, which are given by

$$w_k^{ss} = \begin{cases} w_1 & \text{if } k = 0 \\ 2^{k-1} w_1 & \text{otherwise,} \end{cases} \tag{3.23}$$

$$w_k^{ca} = \begin{cases} E[W]/2 & \text{if } k = 0 \\ E[W]/2 + k - 1 & \text{otherwise.} \end{cases} \tag{3.24}$$

We now determine the expected sleep duration at one interarrival time of ACK packets. Note that we consider that the WNI can sleep only when the time for state transitions between active and sleep modes is longer than the idle duration at an interarrival time of ACK packets. As shown in Fig. 3.4, m data packets are sent and an ACK packet is received at an interarrival time of ACK packets except for the last interarrival time of ACK packets. For simplicity, we regard the number of packets sent in the last interarrival time of ACK packets in one round as the same as that in the other interarrival times of ACK packets. As a result, the expected duration that the WNI sleeps at an interarrival time of ACK packets, $E[t^s(w_k)]$, is calculated as

$$E[t^s(w_k)] = \left[t^a(w_k) - mE[t^t] - E[t^r] - t^{st} \right]^+ \tag{3.25}$$

where $x^+ = \max(0, x)$ and $t^{st} = T^{as} + T^{sa}$.

To derive the expected sleep duration of each phase, we determine the expected sleep duration of data transfer and that of packet loss events. Using Eq. (3.25), we derive the expected sleep duration from the first round to the round in which a packet loss event occurs. Since the number of interarrivals of ACK packets in the kth round is $\lceil w_{k-1}/m \rceil$, the expected sleep duration from the

first to the rth rounds, $E[t_d^s(r, w_k)]$, is obtained as

$$E[t_d^s(r, w_k)] = \sum_{k=1}^{r} \left\lceil \frac{w_{k-1}}{m} \right\rceil E[t^s(w_k)] \tag{3.26}$$

where $r = E[r_{ss}]$ for the initial slow start phase and $r = E[r_{td}]$ for the TD period.

In the round in which a packet loss event is detected, the number of ACK packets received at the client is dependent on which packet in the window is actually lost. When a packet loss event is detected by triple duplicate ACK packets, in average, the idle duration of $RTT/2$ occurs after $\lceil w_r/m \rceil /2$ ACK packets are received. Therefore, the expected sleep duration in the round in which a packet loss event is detected by triple duplicate ACK packets, $E[t_{td}^s(r, w_k)]$, is calculated as

$$E[t_{td}^s(r, w_k)] = \frac{\lceil w_r/m \rceil}{2} E[t^s(w_{r+1})] + \left[RTT/2 - t^{st}\right]^+. \tag{3.27}$$

On the other hand, during the duration of a sequence of RTOs, $E[R]$ data packets are sent whereas no ACK packet is received. We here assume that WNI can always sleep at idle duration of the round because the idle duration is larger enough than t^{st}. Then, the expected sleep duration of a sequence of RTOs, $E[t_{to}^s]$, is given by

$$E[t_{to}^s] = RTT \cdot E[r_{to}] - E[R](E[t^t] + t^{st}). \tag{3.28}$$

where $E[r_{to}]$ is the expected number of rounds during a sequence of RTOs.

Using Eqs. (3.26)–(3.28), we derive the expected sleep duration during the initial slow start phase and during the steady phase, which are denoted by $E[t_\zeta]$ and $E[t_\eta]$, respectively. In a similar way to the calculation in Eq. (3.50), $E[t_\zeta]$ is obtained as

$$\begin{aligned} E[t_\zeta] = E[t_d^s(r_{ss}, w_k^{ss})] + l_{ss}((1 - Q(E[W_{ss}], p))E[t_{td}^s(r_{ss}, w_k^{ss})] \\ + Q(E[W_{ss}], p)E[t_{to}^s]). \end{aligned} \tag{3.29}$$

Similarly, in the same manner as the calculation in Eq. (3.60), $E[t_\eta]$ is calculated as

$$E[t_\eta] = E[n_{cycle}] \times \{E[n]\,(E[t_d^s(r_{td}, w_k^{ca})] + E[t_{td}^s(r_{td}, w_k^{ca})]) + E[t_{to}^s]\}\,. \tag{3.30}$$

Consequently, $E[J^s]$ and $E[T^s]$ are obtained as

$$E[J^s] = P^s \cdot E[T^s], \;\; E[T^s] = E[t_\zeta] + E[t_\eta]. \tag{3.31}$$

3.3.2.5 Energy Consumption due to State Transitions

In a similar way to Subsect. 3.3.2.4, we determine $E[J^{st}]$ based on a bottom-up approach. From the condition the WNI sleeps during the idle duration discussed in Subsect. 3.3.2.4, the expected number of state transitions between active and sleep modes at an interarrival time of ACK packets, $E[n^{st}(w_k)]$, is given by

$$E[n^{st}(w_k)] = \begin{cases} 1 & \text{if } t^a(w_k) > mE[t^t] + E[t^r] + t^{st} \\ 0 & \text{otherwise} \end{cases} \tag{3.32}$$

Therefore, the expected number of state transitions from the first to the rth rounds, $E[n_d^{st}(r, w_k)]$, is calculated as

$$E[n_d^{st}(r, w_k)] = \sum_{k=1}^{r} \left\lceil \frac{w_{k-1}}{m} \right\rceil E[n^{st}(w_k)] \tag{3.33}$$

On the other hand, the expected number of state transitions during the round in which a packet loss event is detected by triple duplicate ACK packets, $E[n_{td}^{st}(r, w_k)]$, is obtained as

$$E[n_{td}^{st}(r, w_k)] = \frac{\lceil w_r/m \rceil}{2} E[n^{st}(w_r)] + n_2^{st} \tag{3.34}$$

where n_2^{st} represents the number of state transitions at the last interarrival time of ACK packets, which is given by

$$n_2^{st} = \begin{cases} 1 & \text{if } RTT/2 > t^{st} \\ 0 & \text{otherwise.} \end{cases} \tag{3.35}$$

In contrast, because the state transitions occur $E[R]$ times during the duration of a sequence of RTOs, the expected number of state transitions during the duration of a sequence of RTOs, $E[n_{to}^{st}]$, is calculated as

$$E[n_{to}^{st}] = E[R] \tag{3.36}$$

Let $E[n_\zeta^{st}]$ and $E[n_\eta^{st}]$ denote the expected numbers of state transitions during the initial slow start phase and during the steady phase, respectively. Following the calculation for Eqs. (3.50) and (3.60), using Eqs. (3.33)–(3.36), $E[n_\zeta^{st}]$ and $E[n_\eta^{st}]$ are derived as follows:

$$\begin{aligned} E[n_\zeta^{st}] =& E[n_d^{st}(r_{ss}, w_k^{ss})] + l_{ss}((1 - Q(E[W_{ss}], p))E[n_{td}^{st}(r_{ss}, w_k^{ss})] \\ &+ Q(E[W_{ss}], p)E[n_{to}^{st}]), \end{aligned} \tag{3.37}$$

$$E[n_\eta^{st}] = E[n_{cycle}] \times \left\{E[n]\left(E[n_d^{st}(r_{td}, w_k^{ca})] + E[n_{td}^{st}(r_{td}, w_k^{ca})]\right) + E[n_{to}^{st}]\right\}. \tag{3.38}$$

Using Eqs. (3.37) and (3.38), $E[J^{st}]$ and $E[T^{st}]$ are calculated as

$$E[J^{st}] = (E[n_\zeta^{st}] + E[n_\eta^{st}])(P^{as}T^{as} + P^{sa}T^{sa}), \tag{3.39}$$

$$E[T^{st}] = (E[n_\zeta^{st}] + E[n_\eta^{st}])t^{st}. \tag{3.40}$$

3.3.2.6 Increase in Data Transfer Latency by Burst Transmission

In this subsection, we consider a disadvantage of burst transmission—the increase in the data transfer latency. For simplicity, we assume that the expected window size is less than W_{bdp} and that the duration of a sequence of RTOs is ignored because they have little influence on the increase in latency with burst transmission. With delayed ACK, the TCP receiver does not send an ACK packet

until m data packets are received or the delayed ACK timer expires. Due to this waiting period, burst transmission achieved by the delayed ACK increases the latency in the whole data transfer.

Assuming that data packets sent in a bursty fashion are received at the TCP receiver at equal intervals due to the background traffic in the wired network, the RTT observed at the TCP sender is increased averagely by $(m-1)RTT/(\frac{1}{r}\sum_{k=1}^{r} w_k)$. Consequently, $E[T^{all}]$ with delayed ACK, $E[\hat{T}^{all}](m)$, is obtained as

$$E[\hat{T}^{all}](m) = \left(RTT + (m-1)\frac{RTT}{E[r_{ss}]w_1}\left(2 - \frac{1}{2^{E[r_{ss}]-1}}\right)\right)E[r_\zeta] + \left(RTT + (m-1)\frac{4RTT}{3E[W]}\right)E[r_\eta]. \tag{3.41}$$

3.4 Model Validation

In this section, to confirm the accuracy of our model, we compare results of analysis with that of simulation experiments. The energy efficiency of the proposed method is dependent on the total sleep duration and the number of state transitions between active and sleep modes. This implies that we can confirm the accuracy of our model by comparing interarrival time of ACK packets of our model and simulations. Therefore, we focus on the distribution of interarrival time of ACK packets at TCP of the client in our analysis model and simulation results.

3.4.1 Parameter Settings of Simulation and Analysis

We conduct simulations with ns-2.35 [52]. We assume a simple WLAN environment in Fig. 3.5 in which two end-hosts are added to the network in Fig. 3.1 to control the background traffic in the wired network. In the WLAN, a wireless client, which associates the AP, sends data to the wired host by TCP. The TCP sender keeps sending data during the simulation. The bottleneck link between routers A and B has 30 [Mbps] of physical bandwidth and 50 [ms] of one-way delay. To control the congestion level of the bottleneck link at the wired network, the UDP sender sends UDP constant bit rate (CBR) traffic to the UDP receiver as cross traffic. One-way delays of all links in the wired network except for the bottleneck link are set to small enough. We set the probability (q)

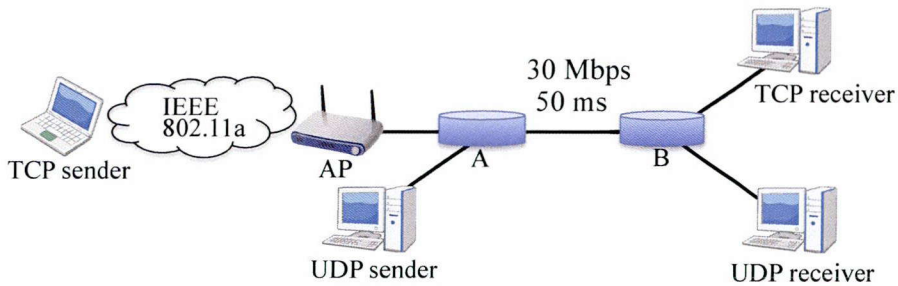


Figure 3.5: WLAN environment for ns-2 simulation

of transmission failures at MAC level to 0.3, which is intended for a moderate loss rate.

Because our model requires both average RTT and the probability (p) of packet loss events, we need to determine them. Since the one-way delay of the bottleneck link is 50 [ms], we set RTT to 100 [ms] for our model. p is calculated by solving Eq. (3.57) for p. Now, giving the expected TCP throughput $E[R_{tcp}]$ [Mbps], $E[W]$ can be calculated

$$E[W] = \frac{E[R_{tcp}] \cdot RTT \times 1024^2}{8 \times 1500}. \tag{3.42}$$

Note that $E[R_{tcp}]$ is calculated by subtracting the rate of cross traffic from the bandwidth of bottleneck link. For example, for 20 Mbps cross traffic, $E[R_{tcp}]$ is 10 Mbps.

3.4.2 Validation Results

Figure 3.6 shows the cumulative distribution function (CDF) of interarrival time of ACK packets for the various values of m, which is the number of packets sent in a bursty fashion, when the rate of cross traffic is 20 [Mbps]. Note that regardless of the rate of cross traffic, the overall tendencies are almost identical.

We can see in Fig. 3.6 that, regardless of the values of m, the distribution of interarrival time of ACK packets obtained by the analysis is reasonably well matched at that by the simulation. However, our model slightly over-estimates interarrival time of ACK packets. The reason for this is as follows. In our model, we assume that ACK packets arrives at the client with equal intervals in a RTT, whereas in the simulation the packet intervals are apt to be compressed after the congestion

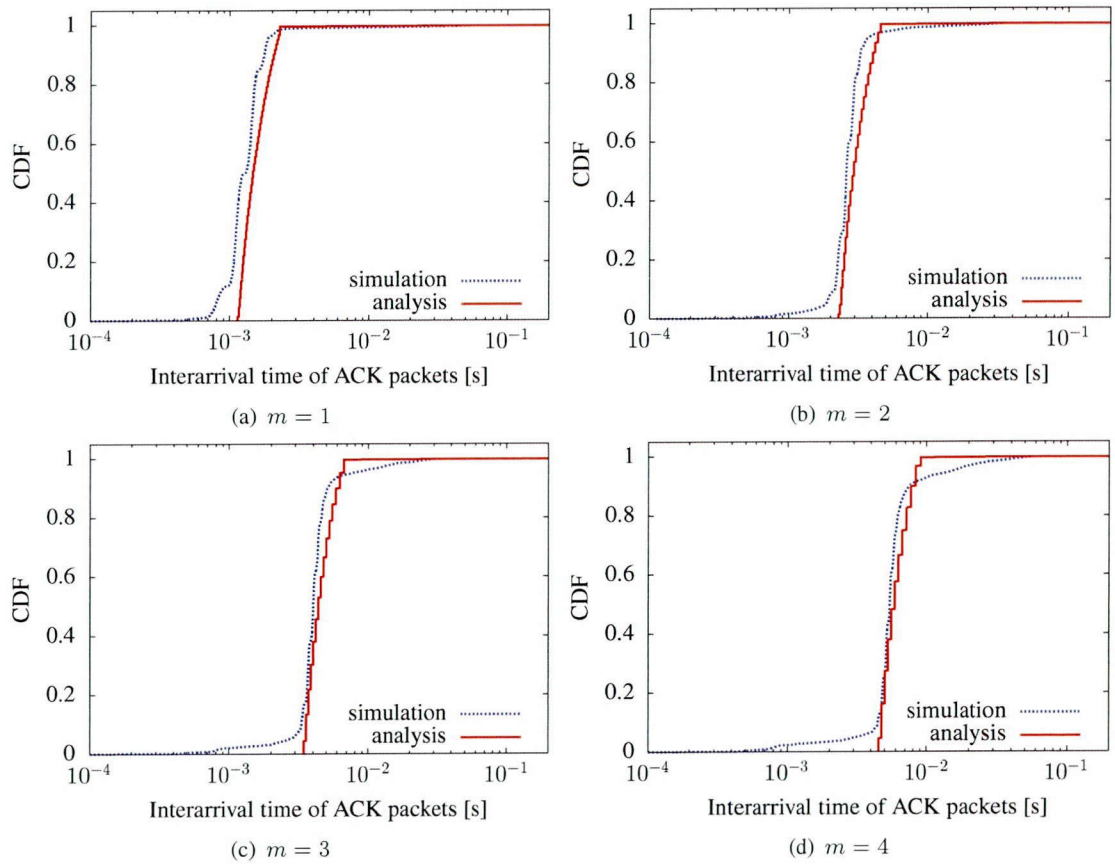


Figure 3.6: Distribution of interarrival time of ACK packets where the rate of cross traffic is 20 Mbps

window size decreases. Thus, the interarrival time of ACK packets in the simulation are distributed to the smaller than that in the analysis.

In order to assess the effects of the rate of cross traffic, we present the average interarrival time of ACK packets, which are calculated from the distribution of interarrival time of ACK packets, for a variety of the rate of cross traffic in Table 3.1, when $m = 1$. Note that the tendencies of results are almost identical regardless of the values of m. When the rate of cross traffic is 25 [Mbps] or less, the average interarrival times of analysis are close to simulation results. However, the average interarrival times of analysis become under-estimated as the rate of cross traffic increases. This is because our analysis model assumes that packet loss events occur independently on each other,

Table 3.1: Average interarrival time of ACK packets for simulation and analysis at a variety of the rate of cross traffic

Rate of cross traffic [Mbps]	Avg. interarrival time of ACKs [ms]		Relative error [%]
	Simulation	Analysis	
12	0.898	0.954	6.23
20	1.57	1.66	5.26
22	2.03	1.97	3.03
25	3.50	3.19	8.92
28	11.35	7.59	33.1
29	54.7	21.6	60.6

whereas in the simulation packet loss events are apt to occur repeatedly. This difference affects the frequency of RTOs especially when the cross traffic is large.

From the above discussion we conclude that our model is reasonably validated when the congestion level is moderate in the wired network.

3.5 Discussions with Analysis Results

In this section, we assess the energy efficiency of TCP data transfer with and without burst transmission by means of the energy consumption model described in Sect. 3.3.

3.5.1 Parameter Settings and Evaluation Metrics

We consider TCP data transfer of a 10 [MB] file from the wireless client to the wired host in Fig. 3.1, by using an IEEE 802.11a WLAN. The WLAN parameters of IEEE 802.11a are summarized in Table 3.2. To calculate radio propagation delay in Eqs. (3.8) and (3.9), we assume that the wireless client is located four meters from the AP. From a data sheet for a WNI implemented by the Atheros AR5004 chip [1], we set P^t, P^r, P^l, and P^s to the values shown in Table 3.3. Following Krashinsky and Balakrishnan [2], who measured power consumption in a specific WNI to determine consumption during transition from active to sleep modes and vice versa, we set $P^{as} = P^l$ and $P^{sa} = P^t$. Moreover, T^{as} and T^{sa} are set equal to 1 [μs] and 1 [ms], respectively, in accordance with Andren *et al.* [3]. The TCP data and TCP ACK packet sizes are set equal to 1500 [bytes] and 40 [bytes],

Table 3.2: WLAN parameters

Name	Value
Data rate	54 Mbps
Slot time	9 μs
SIFS	16 μs
DIFS	34 μs

Name	Value
PLCP preamble	16 μs
MAC header	24 bytes
LLC header	8 bytes
CW_{min}	15
CW_{max}	1023

Table 3.3: Power consumption of Atheros AR5004 [1]

P^t	P^r	P^l	P^s
1.4 W	0.9 W	0.8 W	0.016 W

respectively. Unless otherwise noted, we set q to 0.3, which means we assume a moderate frame loss rate.

In Subsect. 3.5.2, we evaluate energy efficiency and trade-off relationships between energy efficiency and TCP latency in TCP burst transmission. To assess the energy efficiency, we use two metrics: *energy consumptions*, which are given by Eqs. (3.11) and (3.12), and *energy consumption ratio*, which is defined as

$$R_{energy} = E[J_{sleep}]/E[J_{cam}]. \tag{3.43}$$

On the other hand, to evaluate the trade-off relationships, we use *TCP transfer latency*, which is obtained by Eq. (3.41), in addition to energy consumption ratio.

3.5.2 Numerical Results

3.5.2.1 Fundamental Characteristics of Sleeping

To assess the maximum potential energy saving is attained by sleeping, we first show energy consumption with CAM and with sleeping without burst transmission ($m = 1$) in Fig. 3.7. The x-axis in this figure denotes the probability (p) of the packet loss events. We plot the results with various

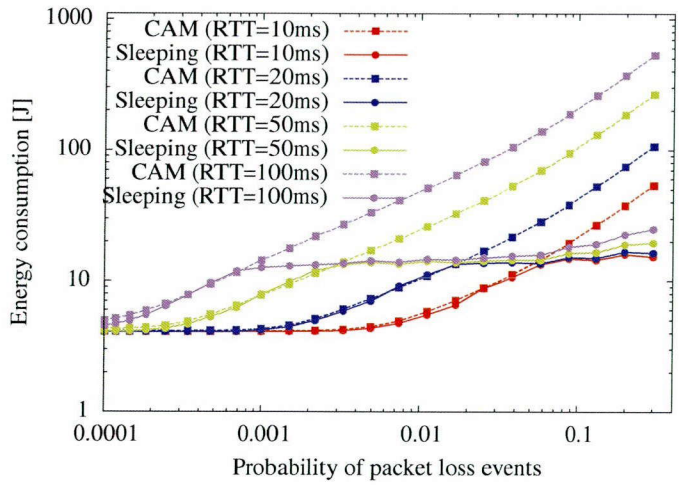


Figure 3.7: Energy consumption for the complete data transfer

values of RTT.

We see that energy consumption decreases as RTT decreases regardless of whether or not sleeping is utilized. This is due to the transfer latency for the whole data becoming small as RTT decreases. Comparing the results for CAM and for sleeping, when p is small, the energy consumptions with CAM and with sleeping are almost identical. For small p, the average congestion window size becomes large, thereby shortening the idle duration at packet interarrivals. Consequently, the WNI cannot sleep at the idle duration. In contrast, for large p, energy consumption with sleeping becomes significantly smaller than that with CAM because the WNI can sleep at all idle duration that occupies a large portion of the transfer latency.

In order to assess the impact on energy consumption due to packet losses at the WLAN, we compare the energy consumption for varying the probability (q) of frame transmission failures at MAC-level. To this end, we depict the energy consumption ratios as a function of p for varying q from 0 to 0.7 in Fig. 3.8, when RTT=100 [ms]. Note that $q = 0$ means the situation in which packet loss events occur only in the wired network, and that we change p_l to control the value of p. We see that energy consumption ratio with a certain value of p increases slightly as q increase. As q increases, the expected time in which one data frame is sent or received becomes longer due to increased frame retransmissions, resulting in the shorter sleep duration at packet interarrivals.

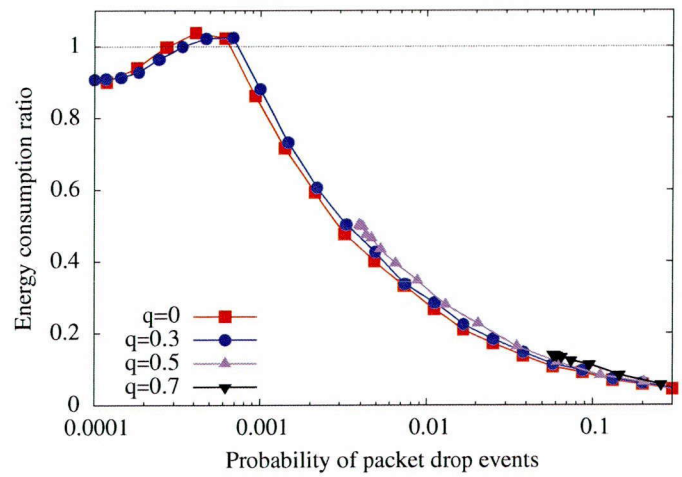


Figure 3.8: Energy consumption ratio for varying q when RTT=100 [ms]

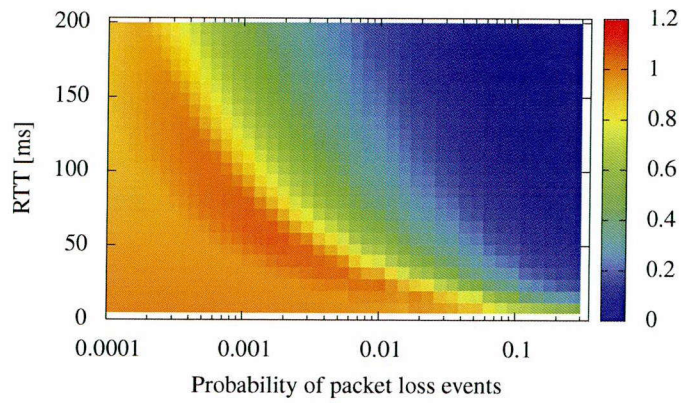
However, the increase in energy consumption due to the shortened sleep duration has a small portion of the total energy consumption. Therefore, the increase in q has a small impact on the sleep efficiency.

From the above results, we conclude that energy consumption is largely affected by RTT and p, which are dominantly determined by wired part of the network. This implies the importance of the analysis of both of wired and wireless parts of the network path to understand the energy efficiency.

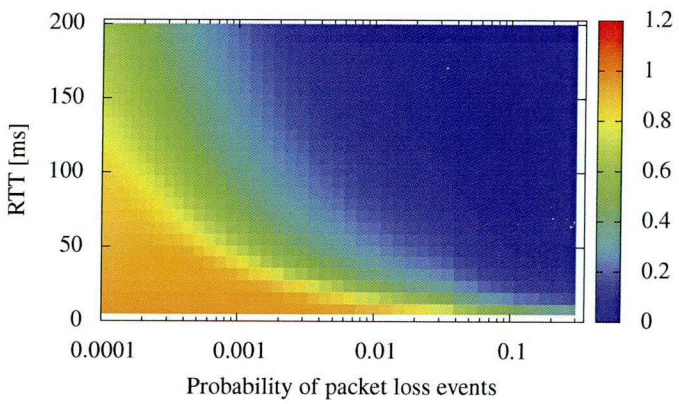
3.5.2.2 Energy Efficiency of Burst Transmission

Figure 3.9 shows the distribution of the energy consumption ratio as functions of RTT and p with $m = 1$ (Fig. 3.9(a)) and $m = 5$ (Fig. 3.9(b)). Comparing Figs. 3.9(a) and 3.9(b), the energy consumption ratio is reduced for the most part with the introduction of burst transmission. In particular, energy efficiency improves significantly with burst transmission when RTT or p is small. When RTT or p is small, energy consumption due to state transitions account a relatively large portion of the total energy consumption. The burst transmission can reduce the number of state transitions, thereby reducing the energy consumed due to state transitions.

Figure 3.10 shows the energy consumption ratio as a function of p for various values of m, when RTT=100 [ms]. When $p = 0.001$, sleeping with $m = 1$ can reduce only about 10 % of



(a) $m = 1$



(b) $m = 5$

Figure 3.9: Energy efficiency as functions of both RTT and probability of packet loss events

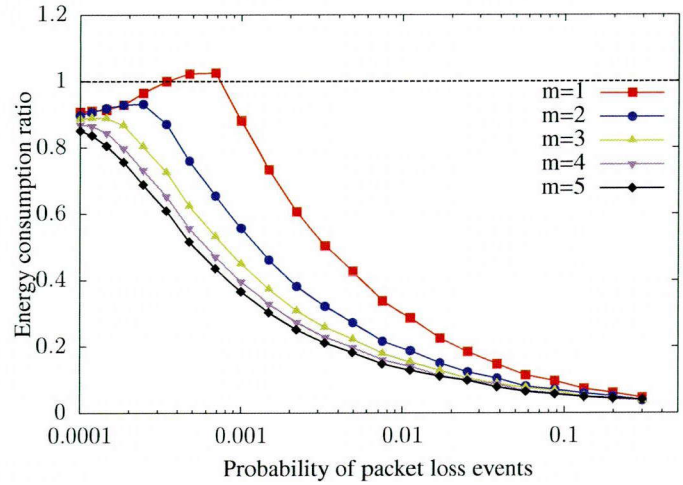


Figure 3.10: Energy efficiency of burst transmission when RTT=100 [ms]

energy consumption with CAM, whereas sleeping with $m = 5$ can reduce further 50 %. Although energy efficiency increases as m increases, the additional energy saving diminishes, implying that good energy efficiency can be obtained with a small value of m.

3.5.2.3 Trade-off Relationships between Energy Efficiency and Data Transfer Latency

To evaluate trade-off relationships between energy efficiency and data transfer latency, we present the energy efficiency and transfer latency as a function of m in Figs. 3.11 and 3.12, respectively.

We can observe that the latency increases linearly while the energy efficiency converges to a constant. When RTT is large, energy efficiency is high even without burst transmission and it increases slightly as m increases, whereas the transfer latency increases largely compared to that without burst transmission ($m = 1$). Conversely, when RTT is small, energy efficiency improves significantly as m increases, whereas the transfer latency increase slightly from that without burst transmission. This result means that burst transmission is suitable for energy saving when RTT is small.

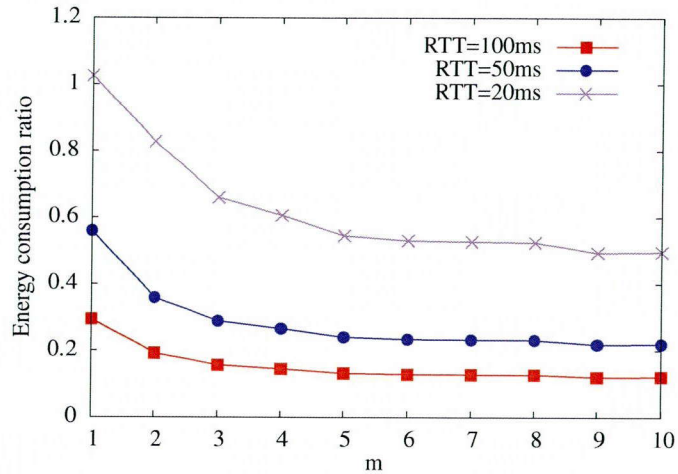


Figure 3.11: Energy efficiency as a function of m when $p = 0.01$

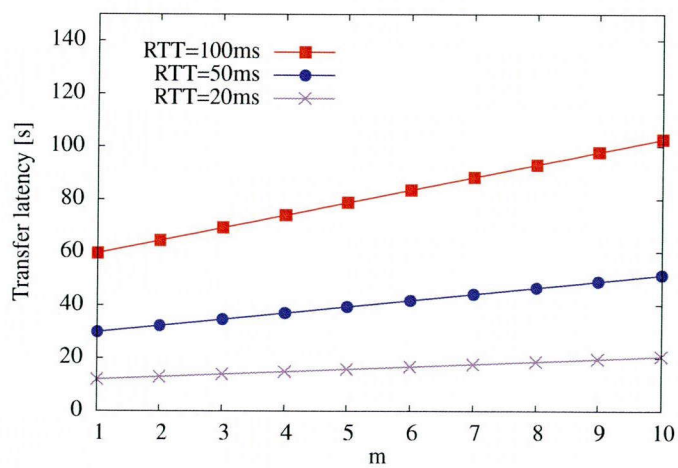


Figure 3.12: Transfer latency as a function of m when $p = 0.01$

3.5.2.4 Summary

From the above discussion, we draw the following conclusions. When the probability of packet loss events is large, sleeping without burst transmission is effective for achieving good energy efficiency. As the probability of packet loss events becomes smaller, burst transmission is more effective for reducing energy consumption. However, for large RTT, further reduction of energy consumption becomes small while the latency increases largely. In contrast, for small RTT, further energy reduction becomes large with increasing moderate delay.

3.6 WNI Factors for Energy Saving

In this section, we discuss the effects of WNI parameters on energy saving characteristics. For this purpose, we compare the energy reductions achieved by varying WNI parameter values.

We first derive the energy reduction by sleeping. From the equations in Subsect. 3.3.2, the expected energy consumptions with CAM and with sleeping during one RTT, denoted by $E[J^{rtt}_{cam}]$ and $E[J^{rtt}_{sleep}]$, respectively, are derived as

$$E[J^{rtt}_{cam}] = w_k E[e^t] + N^{st} E[e^r] + P^l(RTT - w_k E[t^t] - N^{st} E[t^r]), \tag{3.44}$$

$$\begin{aligned} E[J^{rtt}_{sleep}] = {} & w_k E[e^t] + N^{st} E[e^r] + P^s N^{st} E[t^s(w_k)] \\ & + P^l(RTT - w_k E[t^t] - N^{st}(E[t^r] + E[t^s(w_k)]T^{as} + T^{sa})) \\ & + (P^{as}T^{as} + P^{sa}T^{sa}) N^{st} E[n^{st}(w_k)] \end{aligned} \tag{3.45}$$

where $N^{st} = \lceil w_{k-1}/m \rceil$. For simplicity, we assume all idle duration in which the WNI can sleep. By subtracting $E[J^{rtt}_{sleep}]$ from $E[J^{rtt}_{cam}]$, we then obtain the energy reduction achieved by sleeping during one RTT:

$$\begin{aligned} E[J^{rtt}_{cam}] - E[J^{rtt}_{sleep}] = {} & (P^l - P^s) N^{st} E[t^s(w_k)] \\ & - \left\{ (P^{as} - P^l)T^{as} + (P^{sa} - P^l)T^{sa} \right\} N^{st}. \end{aligned} \tag{3.46}$$

We now analyze the impact of WNI parameters on energy consumption by varying each in turn

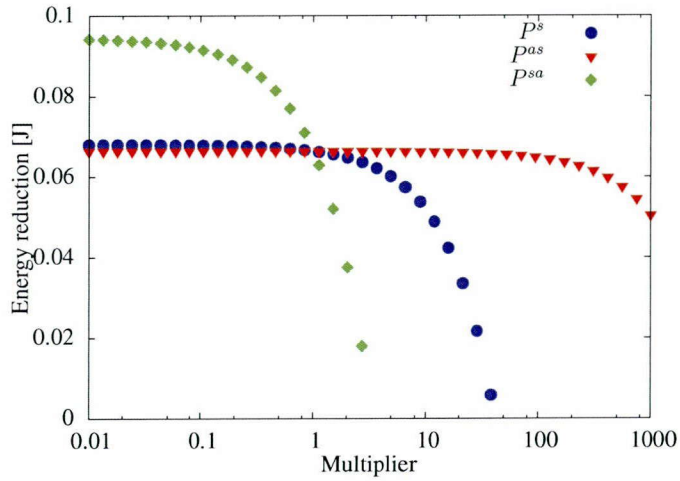
while keeping the others fixed. The WNI parameters, apart from that being varied, are set equal to the values shown in Table 3.3.

Figures 3.13(a) and 3.13(b) show the energy reductions achieved for changing power consumption and transition time, respectively. The x-axes in these figures represent the multiplier applied to the original WNI parameter value. Thus, when the multiplier is one, the energy reduction is achieved for the values in Table 3.3. The multiplier that is smaller than one means that power consumption and transition time of each mode improve by the development in RF circuit design. We also plot results of the multiplier that is larger than one to find the maximum multiplier such that the value of each mode is small enough to reduce further energy consumption. Note that in these figures, $E[t^s(w_k)] = 100$ [ms] and $N^{st} = 20$.

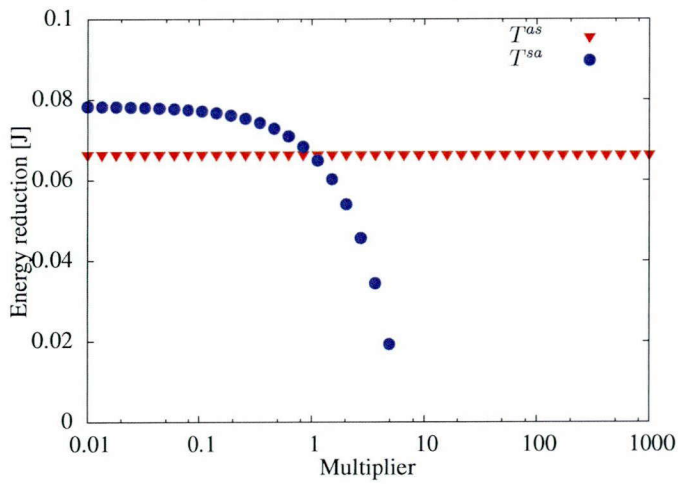
From Fig. 3.13(a), energy reduction is almost unchanged by the reduction of P^s and P^{as}, implying that the values of these parameters are already sufficiently small. In contrast, as P^{sa} decreases, energy reduction increases until the multiplier is approximately 0.05. Likewise, from Fig. 3.13(b), energy reduction is constant with decreasing T^{as}, whereas an increase is found when T^{sa} decreases. However, no further reduction in energy reduction is found when the multiplier of T^{sa} becomes less than 0.05, in common with P^{sa}. This agreement in the multiplier is because the power consumption and transition time of state transitions affect the energy reduction to the same degree.

Thus, we see that there are *critical* multipliers for each WNI parameter, below which no further energy reduction is found. From Figs. 3.13(a) and 3.13(b), we can see that the critical multipliers for P^s, P^{as}, P^{sa}, and T^{sa} are around 1, 100, 0.05, and 0.05, respectively. The critical multiplier for T^{as} can be selected any values since $P^{as} = P^l$, so we set the critical multiplier for T^{as} to 100 as same as that for P^{as}.

We can use these critical multipliers to obtain conditions on WNI parameter values for achieving satisfactory energy reduction. For P^s, according to Eq. (3.46), the degree of energy reduction by sleeping is determined dominantly by the difference between P^l and P^s. By using the ratio of the



(a) Changing power consumption



(b) Changing transition time

Figure 3.13: Energy reductions for changing WNI parameters

original values of P^l and P^s, the power consumption in sleep mode is sufficiently small if

$$P^s \leq \frac{0.016}{0.8} P^l = \frac{1}{50} P^l \tag{3.47}$$

Conversely, because energy reduction is affected to the same degree by the power consumption caused by state transitions and by transition time, we derive conditions for the product of power consumption and transition time. From the ratios of the original values and the critical values, we find the following conditions:

$$P^{as}T^{as} \leq \frac{0.8}{0.8} P^l \times 10^{-6} \times 100 \approx P^l \times 10^{-4}, \tag{3.48}$$

$$P^{sa}T^{sa} \leq \frac{1.4}{0.8} P^l \times 10^{-3} \times 0.05 \approx P^l \times 10^{-4}. \tag{3.49}$$

We can thus design energy-efficient WNIs for TCP data transfer by using WNI parameter values that are sufficiently small compared with the power consumption in listen mode. Specifically, we can develop more efficient WNIs by reducing WNI parameters in descending order of the difference between the current value of the parameters and the values given by the conditions in Eqs. (3.47), (3.48), and (3.49).

3.7 Summary

In this chapter, we have proposed new models for describing the energy consumption of TCP data transfer in a WLAN environment for a wireless client in CAM and in ideal sleep mode when combined with burst transmission. Burst transmission is realized by using TCP delayed ACK. Numerical analyses based on our model showed that, when the probability of packet loss events is large, sleeping without burst transmission is an effective mechanism for energy saving. However, when the probability of packet loss events is small, burst transmission becomes increasingly important. Furthermore, when RTT or the probability of packet loss events are small, burst transmission can reduce energy consumption with increasing moderate delay. Conditions for each WNI parameter are also determined to design energy-efficient WNIs.

Finally, we make mention of the remained issues of this chapter. In this chapter, we derived the energy consumption models based on some strong assumptions. For example, we assumed that a wireless client knows timing of packet transmissions and receptions, and it can sleep at idle duration with exact timing. These assumptions are not feasible in a typical WLAN. Note that we believe that knowledge obtained in this chapter is applied to real WLAN environments if we can control timing of packet receptions at a wireless client. The above issues are addressed in Chapter 4.

Appendix 3.A Derivation of Number of Rounds during TCP Data Transfer

In what follows, we formulate $E[r_\zeta]$ and $E[r_\eta]$. We first determine $E[r_\zeta]$. Following [67], $E[r_\zeta]$ is calculated as

$$E[r_\zeta] = E[r_{ss}] + l_{ss}((1 - Q(E[W_{ss}], p)) + Q(E[W_{ss}], p)E[r_{to}]). \tag{3.50}$$

l_{ss}, $Q(w,p)$, and $E[W_{ss}]$ are derived as

$$l_{ss} = 1 - (1-p)^{S_d/S_p}, \tag{3.51}$$

$$Q(w,p) = \min\left(\frac{(1+(1-p)^3)(1-(1-p)^{w-3})}{(1-(1-p)^w)/(1-(1-p)^3)}, 1\right), \tag{3.52}$$

$$E[W_{ss}] = (E[d_{ss}] + w_1)/2. \tag{3.53}$$

$E[d_{ss}]$ is given by

$$E[d_{ss}] = \min\left\{\frac{(1-(1-p)^{S_d/S_p})(1-p)}{p} + 1, \frac{S_d}{S_p}\right\}. \tag{3.54}$$

Eqs. (3.51)–(3.54) can be found in [67]. Using results in [67], $E[r_{to}]$ is obtained by

$$E[r_{to}] = \frac{G(p)}{1-p}\frac{T_0}{RTT} \tag{3.55}$$

where T_0 is the length of a RTO and $G(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6$.

We can determine $E[r_\eta]$ by utilizing the analysis results in [66]. From Fig. 3.3, in one cycle of the steady phase, multiple TD periods occur before a TO period occurs. Therefore, the expected number of data packets sent during a cycle is given by $(E[n]E[Y] + E[R])$ where $E[Y]$ and $E[R]$ is the expected number of data packets sent during a TD period and during a TO period, respectively. $E[Y]$ and $E[R]$ are obtained as

$$E[Y] = \frac{1-p}{p} + E[W], \quad E[R] = \frac{1}{1-p}. \tag{3.56}$$

$E[W]$ and $E[n]$ are given by

$$E[W] = 1 + \sqrt{8(1-p)/(3p) + 1}, \tag{3.57}$$

$$E[n] = 1/Q(E[W], p). \tag{3.58}$$

Eqs. (3.56) and (3.58) are found in [66].

Since the expected number of packets sent during the steady phase is $(S_d/S_p - E[d_{ss}])$, the number of cycles during the steady phase is calculated as

$$E[n_{cycle}] = (S_d/S_p - E[d_{ss}])/(E[n]E[Y] + E[R]). \tag{3.59}$$

Consequently, we can obtain the expected number of rounds during a cycle as $(E[n]E[r_{td}] + E[r_{to}])$. Using the above notations, $E[r_\eta]$ is calculated as

$$E[r_\eta] = E[n_{cycle}]\,(E[n]E[r_{td}] + E[r_{to}])\,. \tag{3.60}$$

Chapter 4

SCTP Tunneling: A Transport-Layer Solution to Reduce Energy Consumption in a WLAN

4.1 Introduction

Owing to recent developments in wireless network technologies, the Internet is increasingly accessed by using mobile devices such as smartphones, laptops, and tablet PCs. Wireless communication accounts for a large portion of energy consumption in mobile devices. For example, wireless communication via a IEEE 802.11 standard wireless LAN (WLAN) is reported to account for up to 50% of the device's total energy consumption [12, 17–19]. Therefore, there is a great deal of interest in reducing the energy consumed through wireless communication, particularly because most mobile devices are battery-driven. In this thesis, we focus on a WLAN environment since it has grown in popularity and consumes large energy compared with the other wireless technologies [12].

For energy saving in media access control (MAC) layer protocols, the IEEE 802.11 standard defines a power saving mode (PSM) [6], as opposed to the mode under normal operation, which is referred to as the continuously active mode (CAM). In CAM, the radio devices of a wireless client are constantly activated. Thus, while network performance is high, the energy efficiency is

low. In contrast, a wireless client in PSM sleeps when data is not being transmitted or received and periodically wakes up to receive a beacon transmitted from an access point (AP). Although PSM can considerably reduce energy consumption, it can also degrade network performance such as throughput and latency [2].

In order to overcome this issue and to improve further energy efficiency, many researchers have proposed energy-efficient solutions for WLANs [2, 27–35]. Some of these methods [2, 27–32] achieve high energy efficiency by mainly modifying MAC protocols, whereas the others [33–35] are energy-efficient solutions for specific applications. In contrast, we aim to derive a generalized transport-layer solution for energy saving without modifying the applications or MAC protocols.

To maximize energy saving without degrading network performance characteristics, it is important to control a wireless network interface (WNI) of a wireless client to sleep and wake up at appropriate timing. To achieve this, the prediction of timing of packet transmissions and receptions is needed. However, it is difficult due to uncoordinated behavior of applications running concurrently on a single wireless client. In a typical environment where mobile devices are used, multiple TCP connections are established for such applications, resulting in the unpredictable packet transmissions and receptions at MAC-level. Another issue is that the WNI consumes extra energy when transiting active and sleep modes. Therefore, frequent state transitions caused by multiple TCP connections reduce sleep efficiency.

In this chapter, to overcome these issues, we propose SCTP tunneling, which is a transport-layer approach to save energy for TCP data transfer over a WLAN. The proposed method has two key features: *flow aggregation* and *burst transmission* at transport-layer level. In the proposed method, multiple TCP flows are aggregated into a single aggregate flow in order to control the sleep timing. Furthermore, packets from the aggregate flow are sent and received in a bursty fashion to save energy by reducing the number of state transitions. To this end, the proposed method exploits stream control transport protocol (SCTP) [68]. An SCTP association is established between a wireless client and an AP, and all packets of TCP flows at the wireless client are aggregated into the association by means of SCTP multistreaming.

The SCTP multistreaming feature, which is utilized by our proposed method, is mainly used for improving network application performance [69, 70]. Ladha and Amer [69] showed that SCTP

multistreaming improves FTP performance compared with TCP. Natarajan *et al.* [70] demonstrated that HTTP over SCTP has an advantage in terms of transfer latency under lossy networks compared with HTTP over TCP. In contrast, the proposed method utilizes the SCTP multistreaming feature to enhance the sleep efficiency in the presence of multiple TCP flows at a wireless client.

We derive a power consumption model for SCTP tunneling to assess its energy-saving potential. This model is based on our energy consumption models in Chapter 3 for a single TCP flow in a WLAN, which focus on both the frame exchanges of a IEEE 802.11 MAC and the detailed behavior of TCP congestion control mechanisms. From the numerical results of the current model, we demonstrate the energy efficiency of SCTP tunneling as functions of aggregate TCP throughput, network environment, and the degree of burst transmission. We also evaluate an increase in transmission delay by using SCTP tunneling.

In the model mentioned above, some strong assumptions are still left in common with Chapter 3 although the model considers a situation in which multiple TCP flows are established on a single wireless client. For example, we assume that a wireless client knows timing of packet transmissions and receptions, and it can sleep at idle duration with exact timing. This should be resolved to apply SCTP tunneling to real WLAN environments. To this end, we discuss implementation issues of SCTP tunneling.

The remainder of this chapter is organized as follows. First, we describe SCTP tunneling in Sect. 4.2. In Sect. 4.3, the proposed power consumption model for SCTP tunneling is introduced. Section 4.4 shows numerical analysis results for our model. Section 4.5 discusses implementation issues of SCTP tunneling. Finally, conclusions and future research directions are presented in Sect. 4.6.

4.2 SCTP Tunneling

This subsection describes two key features of SCTP tunneling: flow aggregation and burst transmission.

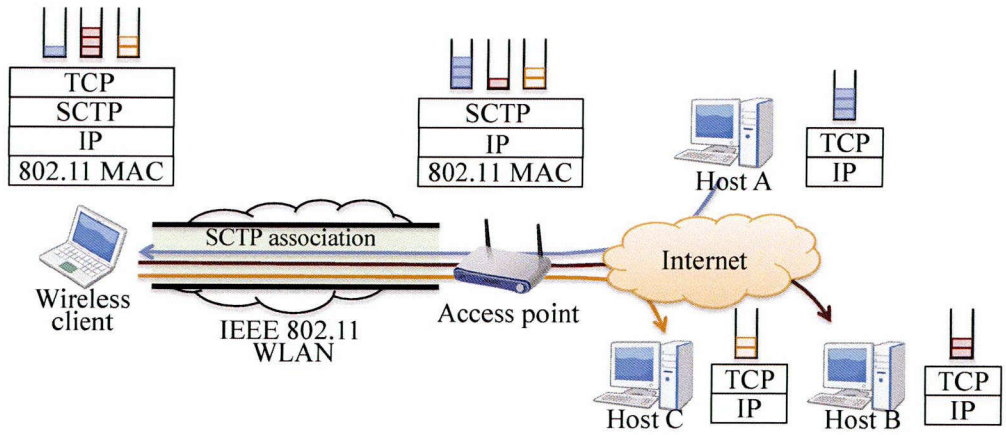
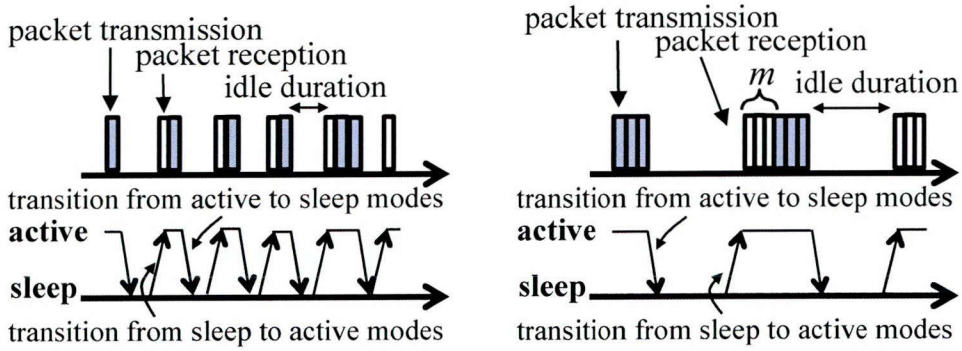


Figure 4.1: SCTP tunneling

4.2.1 Flow Aggregation

SCTP is a connection-oriented transport protocol providing a service similar to TCP and has some advanced features to support increased application requirements. In contrast to the stream-oriented nature of TCP, SCTP is message-oriented. It means that user messages are added to DATA chunks in SCTP, and multiple SCTP-DATA chunks are used to construct an SCTP packet. Another feature of SCTP is multistreaming, which enables streams of user messages from multiple upper-layer applications to be multiplexed into a single SCTP association. Additionally, SCTP has the same congestion control mechanisms as TCP except that the use of selective acknowledgments (SACKs) in SCTP is mandatory.

In SCTP tunneling, an SCTP association is established between a wireless client and an AP, as shown in Fig. 4.1. All packets of multiple TCP flows (e.g., the three flows in Fig. 4.1) are sent by SCTP tunneling, and each TCP flow is distinguished as a single stream in the SCTP association through multistreaming. Note that SCTP tunneling can also be applicable to UDP flows. For simplicity, we assume that only TCP flows are established on a client in this chapter. A TCP packet generated in a wireless client is encapsulated in an SCTP-DATA chunk and enqueued in a transmission queue of the SCTP association. When a new SCTP packet can be transmitted, an SCTP-DATA chunk is dequeued from the transmission queue and is placed in a single SCTP packet.



(a) Without burst transmission

(b) With burst transmission

Figure 4.2: Packet sequences at a client's WNI

Transmission of the SCTP packet then obeys SCTP congestion control mechanisms. Once the SCTP packet is received by an AP, the packet is decapsulated and the original TCP packet is forwarded to its destination. At this time, SCTP at the AP generates an SCTP-SACK chunk to acknowledge receipt of the SCTP-DATA chunk. This SCTP-SACK chunk may be piggybacked with other SCTP-DATA chunks to the client. Data transmission from the AP to the client is conducted in a similar way to the above sequence.

By this method, multiple TCP flows are aggregated on a single SCTP association, and thus the transmission and reception timing of multiple TCP packets can be controlled.

4.2.2 Burst Transmission

To reduce energy consumed due to state transitions, SCTP tunneling employs burst transmission of SCTP packets. Figure 4.2 shows packet sequences of a client's WNI during SCTP tunneling with and without burst transmission. When m packets are sent in burst transmission, once an SCTP has received m SCTP packets, in which the last packet contains an SCTP-SACK chunk, new m packets can be sent simultaneously. Note that an SCTP-SACK chunk piggybacks with an SCTP-DATA chunk in the mth SCTP packet.

SCTP tunneling enables a wireless client to save energy by sleeping during the idle duration lengthened by burst transmission. In practical cases, SCTP tunneling is used by combining it with

existing sleep mechanisms at MAC-level such as PSM and automatic power save delivery (APSD) of IEEE 802.11e [27].

4.3 Power Consumption Model

In this section, we construct a power consumption model for SCTP tunneling to confirm its energy efficiency, by extending the energy consumption models described in Chapter 3. The assumptions for deriving the model are first described in Subsect. 4.3.1, after which the submodel is outlined in 4.3.2. In Subsect. 4.3.3, the expected size of SCTP packets is determined, which is needed in the MAC-level submodel. Finally, we formulate the increase in transmission latency due to buffering delay of SCTP tunneling in Subsect. 4.3.4.

4.3.1 Assumptions

We assume a WLAN environment in which a single SCTP association is established between a wireless client and an AP. Multiple TCP upstream and downstream flows are established in the wireless client by upper-layer applications. We assume that the average throughputs of TCP flows are given, which are usually dependent on network conditions, e.g., network congestion and physical bandwidth between two end hosts of each flow. We also assume that data frames are lost randomly at MAC-level of the WLAN due to channel error and collisions. The probability of transmission failures at MAC-level is given.

Suppose that at the hardware level the WNI has four communication modes — *transmit*, *receive*, *idle* or *listen*, and *sleep* modes [17]. Each of these modes has a different power consumption denoted by P^t, P^r, P^l, and P^s, respectively. Furthermore, the WNI consumes power when transiting between active and sleep modes, and we define P^{as} and P^{sa} as the power consumption when changing from and to active mode, respectively. The duration of these power consumptions is then denoted by T^{as} and T^{sa}, respectively.

We assume that SCTP tunneling uses *ideal sleeping*, instead of PSM and APSD, to assess the maximum energy-saving potential by SCTP tunneling. Ideal sleeping implies that a WNI knows the schedules of both the transmission and reception of TCP packets such that it can sleep and wake

up with exact timing. We also assume that RTS/CTS mechanisms are used by the wireless client when transmitting a frame to an AP, whereas and AP does not utilize RTS/CTS when transmitting a frame to the wireless client.

4.3.2 Power Consumption of SCTP Tunneling

The congestion control mechanisms in SCTP tunneling are the same as those in TCP. Therefore, congestion control mechanisms at the client and AP are activated independently. However, because the client and AP are under the same wireless conditions, we assume that their average behavior is identical. In addition, SCTP congestion control is applied to the entire association, and not to individual streams. Therefore, we can regard the behavior of SCTP congestion control mechanisms for multiple streams as being that for a single TCP flow. As a result, the power consumption model for SCTP tunneling is formulated based on the energy consumption model for a single TCP flow in Chapter 3. Specifically, we determine the power consumption for a WNI of the wireless client.

In what follows, we first explain the behavior of the congestion control mechanisms, which is utilized for calculation of the power consumption for SCTP tunneling. After that, we describe duration of data transfer to determine the power consumption, which differs from that in Chapter 3.

In SCTP data transfer, the number of packets sent per RTT depends on the evolution of the SCTP congestion window size. Figure 4.3 depicts the typical evolution of the congestion window size of an SCTP association from the beginning of the transmission. The data transfer starts with the *initial slow start phase* that ends due to the occurrence of a packet loss event. After that, SCTP packets are sent in the *steady phase* until the data transfer ends. For simplicity, the effects of initial slow start phase are not considered here because we focus on power consumption during the steady-state situation. Here, we define a triple duplicate (TD) period as the duration between two consecutive packet loss events detected by triple duplicate SCTP-SACKs. Further, the duration of a sequence of retransmission timeout (RTO) is referred to as a Timeout (TO) period. In steady phase, we can observe one TO period appears after multiple TD periods, and this sequence appears repeatedly. Here, we define a *cycle* as the duration between two consecutive packet loss events detected by RTOs. Then, the steady phase is constructed by multiple cycles. In Chapter 3, the length of cycles

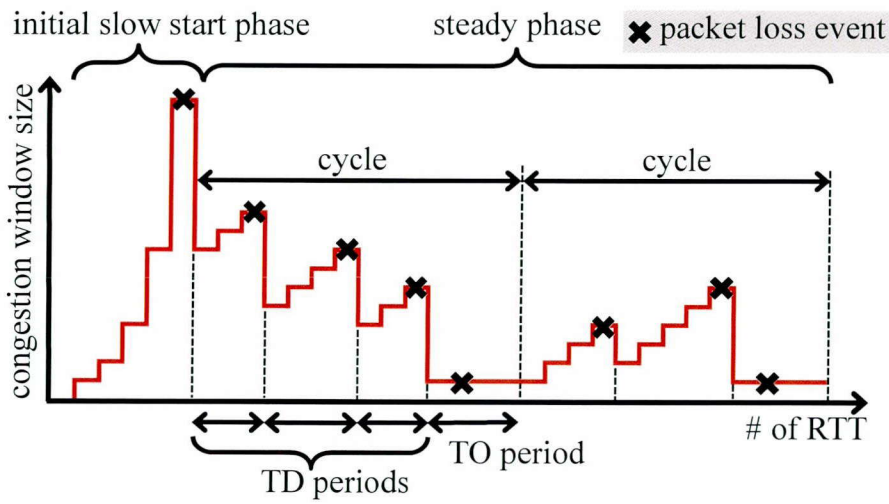


Figure 4.3: Evolution of SCTP congestion window size

is calculated based on a *round* that when the first packet of a window is sent and ends when the corresponding ACK packet is received.

According to our models in Chapter 3, the power consumptions for a WNI of the wireless client with CAM and with sleeping, which are defined as P_{cam} and P_{sleep}, are given by

$$P_{cam} = \frac{E[J^{\eta}_{cam}]}{E[T^{all}_{\eta}]}, \tag{4.1}$$

$$P_{sleep} = \frac{E[J^{\eta}_{sleep}]}{E[T^{all}_{\eta}]} \tag{4.2}$$

where $E[T^{all}_{\eta}]$ is the expected total duration during the steady phase, and $E[J^{\eta}_{cam}]$ and $E[J^{\eta}_{cam}]$ are the expected energy consumptions during the steady phase when a WNI is activated in CAM and in sleep mode, respectively. $E[r_{\eta}]$ is derived in Eq. (3.17), while $E[J^{\eta}_{cam}]$ and $E[J^{\eta}_{sleep}]$ are calculated

as

$$E[J^{\eta}_{cam}] = E[n_{cycle}]E[Y]E[e^t] + E[n^r_{\eta}]E[e^r] + P^l\left(E[T^{all}_{\eta}] - E[n_{cycle}]E[Y]E[t^t] - E[n^r_{\eta}]E[t^r]\right), \tag{4.3}$$

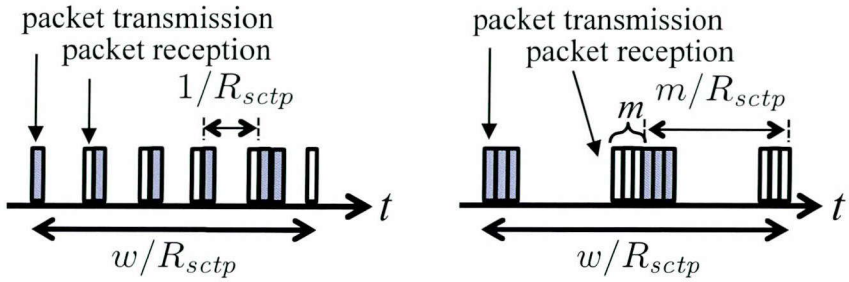
$$E[J^{\eta}_{sleep}] = E[n_{cycle}]E[Y]E[e^t] + E[n^r_{\eta}]E[e^r] + E[t_{\eta}]P^s + E[n^{st}_{\eta}](P^{as}T^{as} + P^{sa}T^{sa}) + P^l\left(E[T^{all}_{\eta}] - E[n_{cycle}]E[Y]E[t^t] - E[n^r_{\eta}]E[t^r] - E[t_{\eta}] - E[n^{st}_{\eta}](T^{as} + T^{sa})\right). \tag{4.4}$$

Here, $E[n_{cycle}]$ is the expected number of cycles, $E[Y]$ is the expected number of SCTP packets sent during a TD period, $E[n^r_{\eta}]$ is the expected number of packets received during the steady phase, and $E[e^t]$ and $E[e^r]$ are the expected energy consumed during transmission and reception of one data frame, respectively. $E[t^t]$ and $E[t^r]$ are the corresponding duration of $E[e^t]$ and $E[e^r]$, respectively. $E[t_{\eta}]$ and $E[n^{st}_{\eta}]$ represent the expected sleep duration and the expected number of state transitions during the steady phase, respectively. The above notations are derived in Chapter 3.

However, in Chapter 3, $E[T^{all}_{\eta}]$ is calculated under the assumption that congestion control behavior is dependent on the average RTT of a TCP connection, whereas in SCTP tunneling this behavior is determined by the behavior of aggregate TCP flows. In this work, it depends on the arrival rate of TCP packets since we assume that average throughput of each TCP flow is given.

Let n^u and n^d be the numbers of upstream and downstream TCP flows, respectively, and r^u_i and r^d_j be the average throughputs [byte/s] of the ith upstream and the jth downstream TCP flows. Assuming that delayed ACK is not utilized in TCP[1], the numbers of TCP-DATA and TCP-ACK packets in a single TCP connection are identical. Here, let R^u [packet/s] and R^d [packet/s] are the number of SCTP packets sent from the wireless client to the AP per unit time and that sent from the AP to the wireless client per unit time, respectively. Since an SCTP packet contains a single TCP

[1]This assumption can be relaxed easily. When delayed ACK is used, the number of TCP-ACK packets sent from a TCP receiver decreases.



(a) $m = 1$ (b) $m = 3$

Figure 4.4: SCTP packet sequences at WNI of wireless client

packet in our SCTP tunneling, R^u and R^d are given as

$$R^u = R^d = \sum_{i=1}^{n^u} \frac{r_i^u}{s_{data}^{tcp}} + \sum_{j=1}^{n^d} \frac{r_j^d}{s_{data}^{tcp}} \tag{4.5}$$

where s_{data}^{tcp} is the size [bytes] of a TCP-DATA packet. In what follows, R^u and R^d are simply denoted by R.

Next, we consider the SCTP packet sequence in the WNI of the wireless client, as shown in Fig. 4.4. In the figure, w is the SCTP congestion window size and R_{sctp} is the average throughput of SCTP tunneling given by

$$R_{sctp} = \min\left(R,\ R_{sctp}^{max}\right). \tag{4.6}$$

Here, R_{sctp}^{max} is the maximum throughput achieved by SCTP tunneling, which is given by $1/\left(E[T^t] + E[T^r]\right)$. Note that duration of sent packets in a window is identical with and without burst transmission.

Finally, from Eq. (4.6) and our previous model in Chapter 3, $E[T_\eta^{all}]$ is calculated as

$$E[T_\eta^{all}] = E[r_\eta]\frac{1}{R_{sctp}} \tag{4.7}$$

where $E[r_\eta]$ is the expected number of rounds during the steady phase.

4.3.3 Expected Size of SCTP Packets

In SCTP tunneling, the expected values of T_{DATA}^{client} and T_{DATA}^{AP} in Eqs. (3.3) and (3.4), respectively, are as functions of the expected sizes of SCTP packets sent and received. Therefore, the expected sizes of SCTP packets sent and received by the wireless client must be computed. To this end, we first calculate the ratios between the numbers of TCP-DATA and TCP-ACK packets in SCTP packets sent from the wireless client and the total number of SCTP packets sent from the client, which are denoted by N_{data}^{u} and N_{ack}^{u}, respectively. By using the ratio between the aggregate throughputs of upstream TCP flows and downstream TCP flows, $\mu = \sum_{j=1}^{n^d} r_j^d / \sum_{i=1}^{n^u} r_i^u$, N_{data}^{u} and N_{ack}^{u} are calculated as follows:

$$N_{data}^{u} = \frac{1}{1+\mu}, \quad N_{ack}^{u} = \frac{\mu}{1+\mu}. \tag{4.8}$$

Similarly, N_{data}^{d} and N_{ack}^{d}, which are the ratios between the numbers of TCP-DATA and TCP-ACK packets in SCTP packets sent from an AP and the total SCTP sent from the AP, respectively, are given by

$$N_{data}^{d} = \frac{\mu}{1+\mu}, \quad N_{ack}^{d} = \frac{1}{1+\mu}. \tag{4.9}$$

Disregarding the length of an SCTP-SACK chunk, the expected sizes of SCTP packets sent from the wireless client and from the AP, which are defined as $\hat{s}_{sctp}^{u}$ [bytes] and $\hat{s}_{sctp}^{d}$ [bytes], are

$$\hat{s}_{sctp}^{u} = \frac{1}{1+\mu} s^{sctp}(s_{data}^{tcp}) + \frac{\mu}{1+\mu} s^{sctp}(s_{ack}^{tcp}), \tag{4.10}$$

$$\hat{s}_{sctp}^{d} = \frac{\mu}{1+\mu} s^{sctp}(s_{data}^{tcp}) + \frac{1}{1+\mu} s^{sctp}(s_{ack}^{tcp}). \tag{4.11}$$

Here, s_{ack}^{tcp} is the size of a TCP-ACK packet and $s^{sctp}(s)$ is the size of a SCTP packet as a function of payload size s [bytes], and

$$s^{sctp}(s) = 12 + 20 + s + padding \tag{4.12}$$

Table 4.1: WLAN parameters for numerical analyses

Name	Value
Data rate	54 [Mbps]
Slot time	9 [μs]
SIFS	16 [μs]
DIFS	34 [μs]

Name	Value
PLCP preamble	16 [μs]
MAC header	24 [bytes]
LLC header	8 [bytes]
CW_{min}	15
CW_{max}	1023

where *padding* is a padding byte to ensure that the length of $s^{sctp}(s)$ is a multiple of four bytes.

4.3.4 Increase in Transmission Latency due to Buffering Delay

Burst transmission, which is utilized by SCTP tunneling, causes transmission latency to each packet. Specifically, SCTP tunneling buffers SCTP packets at the tunnel inlet until m TCP packets arrive, which results in an additional delay for each TCP packet. Here, by using R, which is obtained by Eq. (4.5) , the average buffering delay at the tunnel inlet is calculated as

$$D = \frac{m-1}{2}\frac{1}{R}. \tag{4.13}$$

4.4 Discussion with Numerical Results

In this section, we assess the energy efficiency of SCTP tunneling by means of the power consumption model described in Sect. 4.3.

4.4.1 Parameter Settings and Evaluation Metrics

We consider an IEEE 802.11a WLAN in which multiple upstream and downstream TCP flows are established between a wireless client and wired hosts (Fig. 4.1). The WLAN parameters of IEEE 802.11a are summarized in Table 4.1. To calculate τ, which is the radio propagation between the wireless client and the AP, we assume that the wireless client is located four [meters] from the AP. From a data sheet for a WNI implemented by using the Atheros AR5004 chip [1] and measurement

Table 4.2: Power consumption of Atheros AR5004 [1] and parameters of state transitions [2, 3]

P^t	P^r	P^l	P^s	P^{as}	P^{sa}	T^{as}	T^{sa}
1.4 [W]	0.9 [W]	0.8 [W]	0.016 [W]	0.8 [W]	1.4 [W]	1 [μs]	1 [ms]

studies [2, 3], we set parameters of power consumption to the values listed in Table 4.2. The TCP-DATA and TCP-ACK packet sizes are set to 1500 [bytes] and 40 [bytes], respectively. The SCTP packet sizes for both directions are calculated by Eqs. (4.10) and (4.11). The maximum number of frame retransmissions, N, is set to seven.

In Subsect. 4.4.2, we evaluate energy efficiency and trade-off relationships between energy efficiency and average buffering delay of SCTP tunneling. In order to assess the energy efficiency, we use power consumptions obtained by Eqs. (4.1) and (4.2). In contrast, to evaluate the trade-off relationships, we use *average buffering delay*, which is obtained by Eq. (4.13), in addition to *energy reduction ratio*, which is defined as

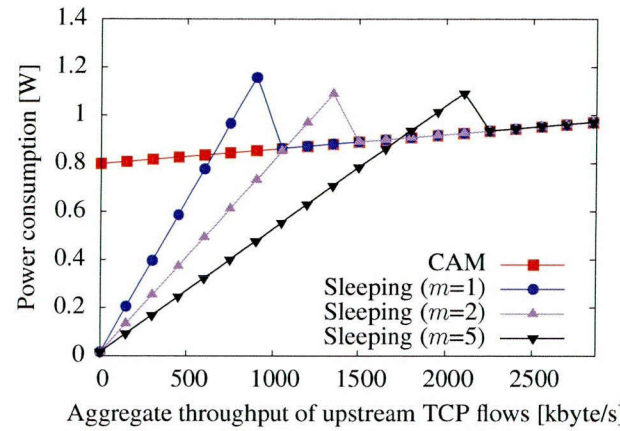
$$P_{ratio} = \left(P_{cam} - P_{sleep}\right) / P_{cam}. \tag{4.14}$$

Energy efficiency is high when the ratio is large.

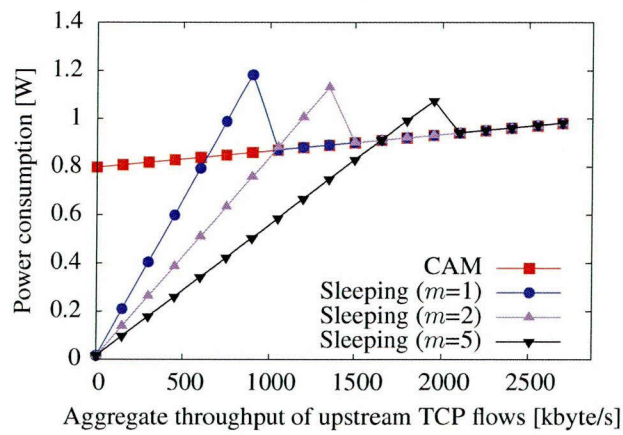
4.4.2 Numerical Results

4.4.2.1 Energy Efficiency

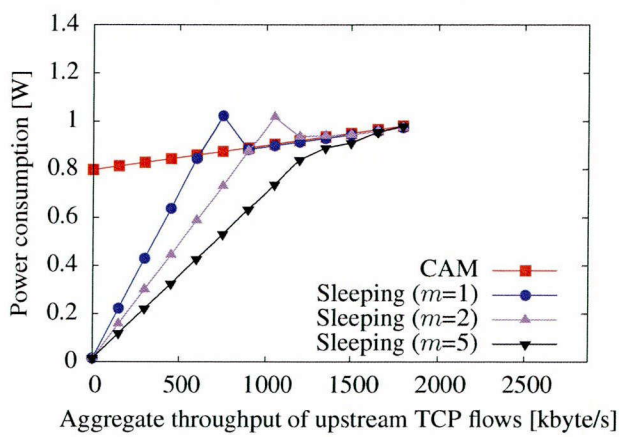
Figure 4.5 shows the power consumption results in the case that only upstream TCP flows exist when $q = 0.1$, 0.2, and 0.5. Here, we evaluate the performance of CAM and sleeping with burst transmission for m = 1, 2, and 5. Note that $m = 1$ signifies sleeping without burst transmission, while $m > 1$ is sleeping with burst transmission. In this figure, the x-axis represents the aggregate throughput of upstream TCP flows, where the average throughput of each TCP flow is 150 [kbyte/s]. Note that the maximum value of the aggregate throughput of upstream TCP flows is limited by the maximum throughput achieved by SCTP tunneling, R_{sctp}^{max}, which is dependent on the value of q. We also note that the results when upstream and downstream TCP flows coexist show a similar



(a) $q = 0.1$ ($p = 1.00 \times 10^{-8}$)



(b) $q = 0.2$ ($p = 2.56 \times 10^{-6}$)



(c) $q = 0.5$ ($p \approx 3.91 \times 10^{-3}$)

Figure 4.5: Power consumption as a function of aggregate throughput of upstream TCP flows

trend to Fig. 4.5. Although the difference between the number of upstream and downstream TCP flows affects the expected sizes of SCTP packets obtained by Eqs. (4.10) and (4.11), the packet size have only a small influence on the power consumption compared with protocol overheads, e.g., backoff and control frame exchanges.

From Fig. 4.5, we observe that the power consumption when utilizing CAM is increased by an increase in the aggregate throughput of upstream TCP flows. As the aggregate throughput grows, the duration of packet transmission and reception increases while idle duration decreases, which increases the power consumption. When sleeping is employed, the power consumption is considerably reduced regardless of the value of m. The power consumption increases for large aggregate throughput, whereas the increase rate of power consumption is low at large m values. For instance, when the aggregate throughput of upstream TCP flows is 450 [kbyte/s] in Fig 4.5(a), sleeping without burst transmission reduces power consumption by 27% compared with CAM. In contrast, the reduction is around 69% for sleeping with $m = 5$. These results mean that the smaller number of state transitions resulting from burst transmission has a large impact on energy reduction.

However, when aggregate throughput further increases, the power consumption with sleeping exceeds the power consumption of CAM. The power consumption required for state transitions exceeds the reduction realized by sleeping since the idle duration is short. Note that such a situation can be avoided by staying in active mode when the idle duration is insufficient. After that, aggregate throughput further increases, the power consumption with sleeping eventually becomes the same value of that with CAM. At this time, there is no idle duration to sleep.

Comparing Figs. 4.5(a), 4.5(b), and 4.5(c), we can observe that changes in q have little effect on the power consumption when CAM is employed. The power consumption during idle duration, whose length is affected by protocol overheads at MAC-level, provides a large contribution to the total power consumption compared with that during data frame retransmission even though the number of frame retransmissions increases for large q. In contrast, when sleeping is employed, the value of q has a greater impact on the power consumption with large m than that with small m. An increase in q decreases idle duration to sleep, which results in the decrease in the energy reduction for large m. Note that it has a smaller impact on the power consumption compared with the impact of aggregate throughput.

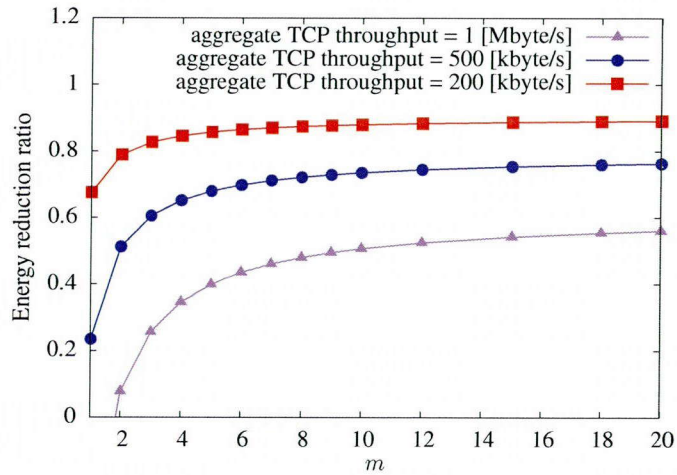


Figure 4.6: Energy reduction ratio for various m values and $q = 0.1$

From the above results, we conclude that the power consumption of SCTP tunneling is hence predominantly determined by the aggregate throughput of TCP flows, while data frame retransmission provides only a small contribution to power consumption.

4.4.2.2 Trade-off Relationship between Energy Efficiency and Buffering Delay

Figure. 4.6 shows the energy reduction ratio for various m values and $q = 0.1$. The corresponding average buffering delay is presented in Fig. 4.7.

As m increases, the energy reduction ratio converges to a constant value, whereas the average buffering delay increases linearly. On the other hand, as the aggregate throughput of TCP flows increases, the improvement rate of energy reduction ratio becomes large, whereas the increase rate of the delay becomes low. The power consumption of state transitions, which is reduced by burst transmission, provides a large portion of the total power consumption compared with the power consumption of state transitions when the aggregate throughput is low.

To further understand the trade-off relationships, we consider the situation in which acceptable buffering delay is given by a user or application. To this end, we introduce D_{th}, which denotes an upper limit of an acceptable average buffering delay. Figure 4.8 depicts the energy reduction ratio achieved subject to $D \leq D_{th}$. In the figure, the average throughput of each TCP flow is 50

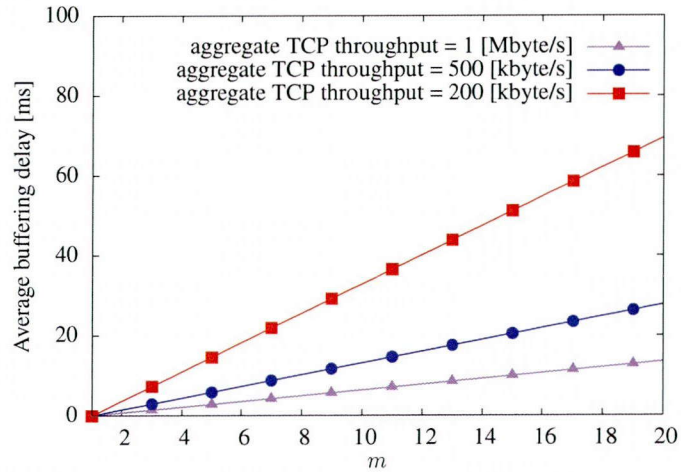


Figure 4.7: Average buffering delay for various m values and $q = 0.1$

[kbyte/s]. We observe that, as the aggregate TCP throughput increases, the energy reduction ratio when $D_{th} = 0$ decreases linearly and finally reaches zero, whereas the higher energy reduction is obtained for larger D_{th}. For example, when only 3 [ms] of additional delay is acceptable, we obtain roughly 0.3 and 0.4 reductions from the ratio for $D_{th} = 0$ when the aggregate TCP throughputs are 500 [kbyte/s] and 800 [kbyte/s], respectively. When 10 [ms] of additional delay is acceptable, we can obtain roughly 0.5 and 0.6 reductions from the ratio for $D_{th} = 0$ when the aggregate TCP throughputs are 500 [kbyte/s] and 800 [kbyte/s], respectively.

From above results, we conclude that, because SCTP tunneling causes additional delay for each TCP packet, a value of m for sleeping with burst transmission must be selected such that the trade-off between energy efficiency and the delay is at a level acceptable for users or applications.

4.5 Implementation Issues of SCTP Tunneling

In this subsection, we discuss implementation issues of SCTP tunneling.

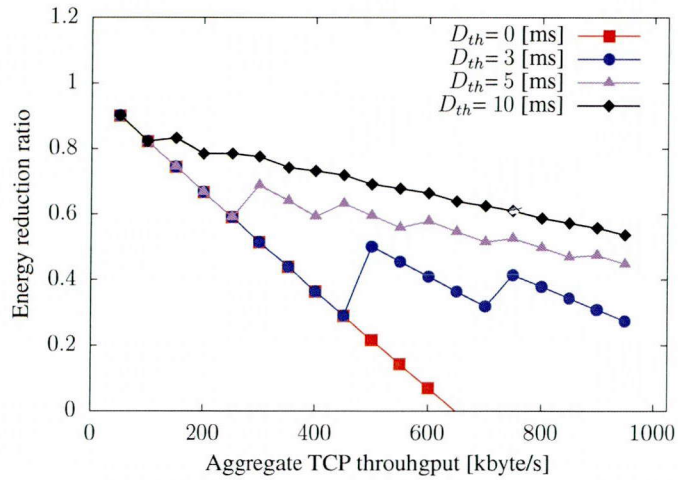


Figure 4.8: Energy reduction ratio achieved subject to $D < D_{th}$ and $q = 0.1$

4.5.1 Implementation Overview

We assume that SCTP tunneling is implemented in a Linux-based application since Linux is an open source software and it can be modified freely.

In order to implement SCTP tunneling, we need to resolve the following problems. First, we should construct the method to forward packets from multiple applications to SCTP tunneling. Second, we need to employ a MAC-level sleep mechanism as a sleep mechanism of SCTP tunneling. In addition, we need to resolve problems regarding unknown timing of packet receptions at wireless clients. Finally, the specific method is required to realize burst transmission.

In what follows, each issue is discussed in turn. One of feasible methods to aggregate packets from multiple applications is to use a TUN device driver [71]. The TUN driver is a virtual network device driver, which allows IP packets to be delivered to user-space application that attaches itself to the TUN device. Thus, SCTP tunneling with the TUN device can easily obtain packets from all applications running on a wireless client. Figure 4.9 depicts the intercommunication between network protocol stacks of SCTP tunneling with a TUN device. When TCP packets are sent from applications running on a wireless client, they are delivered to SCTP tunneling via the TUN device. After that, the client-side SCTP endpoint sends SCTP packets that contain TCP packets as chunks

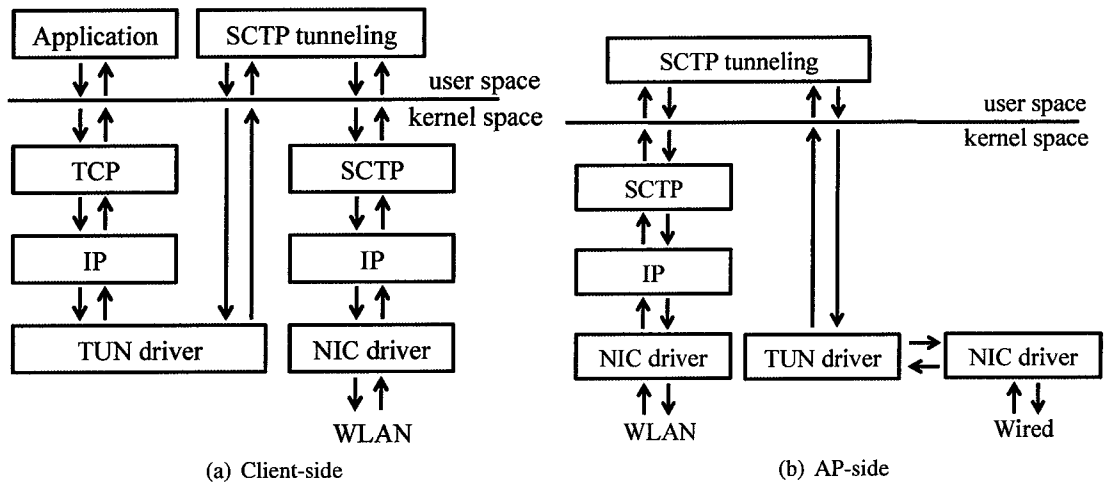


Figure 4.9: Intercommunication between network protocol stacks of SCTP tunneling

to AP-side SCTP endpoint through a WLAN. When receiving TCP packets via SCTP, AP-side SCTP tunneling forwards them to the TUN device, and then they are forwarded to their original destinations. Data transmission from the AP to the client is conducted in a similar way to the above sequence.

Next, we determine a sleep mechanism for SCTP tunneling. IEEE 802.11 has two sleep mechanisms: PSM and APSD, which buffer data frames destined for wireless clients in sleeping. We here employ unscheduled APSD (U-APSD), which is one of power saving modes of APSD, as a sleep mechanism of SCTP tunneling. This is because PSM has a drawback that wireless clients only receive frames from the AP at beacon intervals of typically 100 [ms]. U-APSD overcomes this issue by using a data frame sent from wireless clients, which is referred to as a *trigger frame*, as an indication of request for sending the buffered data frame at the AP. This mechanism can determine timing of packet receptions at the wireless clients. Note that, in order to receive data frames from the AP at intended interval, SCTP tunneling needs to send trigger frames at appropriate timing, which is determined in Subsect. 4.5.2.

Finally, we consider a method to realize burst transmission. As a burst transmission mechanism, we employ a simple method that buffers packets on SCTP tunneling until the number of buffered packets reaches m or a certain period passes from the first packet is buffered. Here, when trigger

frames are sent to the AP when the buffer at the AP stores m packets, they are sent to the wireless client. Such simple mechanism can realize burst transmission at the AP. Note that we need to implement additional mechanisms for burst transmission only at a wireless client.

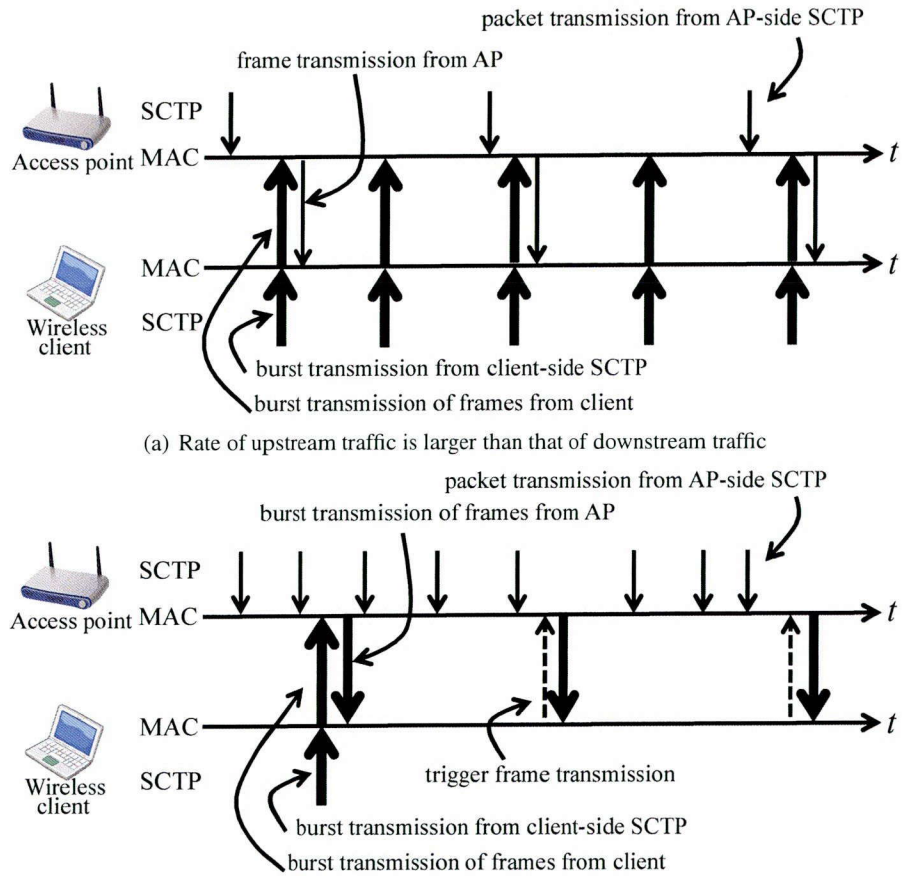
From the above discussions, we conclude that one of feasible implementations of SCTP tunneling is a Linux application with TUN devices with U-APSD as a sleep mechanism. However, we still leave the issue of timing of trigger frame transmissions, which are discussed in Subsect 4.5.2.

4.5.2 Trigger Frame Transmission

We here discuss an algorithm for determining timing of trigger frame transmission. Figure 4.10 depicts frame exchanges between a client and an AP when focusing on trigger frame transmissions. As shown in Fig. 4.10(a), when the rate of upstream traffic is larger than the rate of downstream traffic, there is no need to send a trigger frame actively from a wireless client because data frames sent from the wireless client are used as trigger frames in U-APSD. In contrast, intentional trigger frames are needed to be transmitted in situations where the rate of downstream traffic is larger than that of upstream traffic as depicted in Fig. 4.10(b). This is because it is likely that the AP buffers data frames destined for a wireless client with sleeping until receiving a trigger frame, which causes additional delays for the data frames. Ideally, it is preferable that trigger frames arrive at the AP just after m data frames are buffered at the AP. However, in practical cases, the wireless client cannot obtain such information without any assists from the AP. Therefore, we need to estimate such timing at the wireless client.

From the above discussions, we employ a simple method for determining timing of trigger transmissions as follows. In the method, a trigger frame is sent when passing a certain period in which a wireless client do not send any data frames. This period is determined based on estimations of time taken to buffer m data frames at the AP buffer.

In what follows, we estimate the time taken to buffer m data frames at the AP buffer. Since SCTP tunneling employs burst transmission, it is difficult to know the packet interarrivals at the AP's buffer from that observed at the wireless client. Therefore, the average interarrival of packets



(a) Rate of upstream traffic is larger than that of downstream traffic

(b) Rate of downstream traffic is larger than that of downstream traffic

Figure 4.10: Frame exchanges between client and AP focusing on trigger frame transmissions

is calculated by counting the number of packets received on the wireless client during the predetermined time slot, H, which is regarded as the estimate value of packet interarrivals at the AP buffer.

We first determine the moving average for the number of packets received on a wireless client, which is denoted by n_{i+1}^r. Then, n_{i+1}^r is calculated as

$$n_{i+1}^r = \alpha \hat{n}_i^r + (1 - \alpha) n_i^r \tag{4.15}$$

where α is the smoothing factor and $\hat{n}_i^r$ is the number of received packets observed on the wireless

client during the ith time slot. Let t^a_{i+1} denote the average interarrivals of packets during the $(i+1)$th time slot. Then, from Eq. (4.15), t^a_{i+1} is given by

$$t^a_{i+1} = H/n^r_{i+1}. \tag{4.16}$$

Thus, in the $(i+1)$th time slot, we can estimate the time for buffering new m data frames at the AP buffer as $m \cdot t^a_{i+1}$. Here, when the transmission interval of trigger frames is shorter than the state transition time between active and sleep modes, this causes unnecessary load for the wireless client. In contrast, when the transmission interval of trigger frames is larger than the beacon interval, the transmission of trigger frames makes no sense since the wireless client can receive frames buffered at the AP with the beacon. To avoid such situations, we define the maximum and minimum values for the transmission interval of trigger frames, which are denoted by $t^{trigger}_{max}$ and $t^{trigger}_{min}$, respectively. Consequently, we obtain the transmission interval of trigger frames as

$$t^{trigger}_{i+1} = \max\left(\min\left(m \cdot t^a_{i+1},\, t^{trigger}_{max}\right),\, t^{trigger}_{min}\right). \tag{4.17}$$

4.6 Summary

We have proposed a transport-layer approach to reduce energy consumption for TCP data transfer over a WLAN, termed SCTP tunneling. SCTP tunneling has two key features: flow aggregation and burst transmission at transport-layer level. To assess the energy efficiency gained by SCTP tunneling, we formulated a power consumption model of SCTP tunneling based on energy efficiency analysis of a single TCP flow in a WLAN. Numerical results of the model show that the power consumption of SCTP tunneling is predominantly determined by the aggregate throughput of TCP flows, while burst transmission can considerably reduce power consumption with increasing moderate delay. We also discussed implementation issues of SCTP tunneling.

Finally, we make mention of directions for the widespread use of SCTP tunneling. SCTP tunneling needs to be implemented at both wireless clients and APs. One of feasible scenarios to introduce SCTP tunneling is that WLAN vendors provide APs implemented SCTP tunneling and

applications of client-side SCTP tunneling. Note that AP-side SCTP tunneling is not necessarily implemented on the AP. Thus, the other scenario is that a network device that is implemented SCTP tunneling is installed near APs. For example, SCTP tunneling is implemented to WLAN controllers in public WLANs or office WLANs, which allows SCTP tunneling to be used in these WLANs.

In the future, we plan to implement the SCTP tunneling on commercial WLAN APs and wireless clients, and to evaluate the performance of SCTP tunneling in real WLAN environments.

Chapter 5

Conclusion

As the IEEE 802.11-based WLANs grow in popularity, applications used in WLANs and user usage of WLAN environments become diverse. Such changes bring the growth of the importance of fair resource usage among users in WLANs. Furthermore, as they increase the amount of time spent on wireless communication, energy efficiency of WLAN-enabled devices becomes more and more important. Therefore, in this thesis, we discussed both per-flow fairness and energy efficiency of client's WNIs in WLANs.

In Chapter 2, we addressed fairness issue among TCP flows in a WLAN. Behavior of TCP congestion mechanisms causes two unfairness among TCP flows in a WLAN. In order to alleviate such unfairness, we proposed, designed, and implemented a transport-layer solution that exploits ACK packet losses as an indication of congestion at an AP. For appropriate evaluation for such fairness issues, we then presented a performance metric that considers both per-flow fairness and bandwidth utilization at a network bandwidth. The proposed metric is based on the variations in throughput of concurrent flows and the ideal throughput distribution in which all flows achieve the same throughput and the network bandwidth is fully utilized. Through the simulation experiments and the experiments with WLAN products from several vendors, we obtained the following findings. The proposed method could improve two kinds of TCP unfairness in a WLAN while maintaining low RTT by alleviating congestion at the AP buffer, regardless of type of APs. It also could avoid starvation of flows even when the buffer size at the AP is too small to accommodate many flows. In

addition, the proposed method was confirmed that its performance was independent on conditions of the wireless channel. From the above findings, we conclude that the transport-layer approach with simple modifications suffices for improving TCP unfairness in a WLAN.

In Chapters 3 and 4, we dealt with the energy efficiency issue of a client's WNI in a WLAN. In Chapter 3, in order to assess the impact of TCP behavior on sleep efficiency, we constructed energy consumption models for a single TCP flow in a WLAN. Furthermore, in order to improve the sleep efficiency, TCP-level burst transmission behavior was also proposed. The accuracy of our model was confirmed by comparing results of numerical analysis with that of simulation experiments. The validation results presented that our model is reasonably validated when the congestion level is moderate in the wired network. From the numerical analyses based on our model, we obtained the following results. The sleep efficiency is largely affected by RTT and the probability of packet losses at transport-layer level, which are dominantly determined by wired part of the network. Sleeping with burst transmission is suitable for situations in which RTT or the probability of packet losses at transport-layer level are small, i.e., TCP transmission rate is high. In such a situation, sleeping with burst transmission can significantly reduce energy consumption with increasing moderate delay. In contrast, when RTT or the probability of packet losses at transport-layer level are small, sleeping with burst transmission incurs large delays for TCP packets although it achieves energy saving. From the above knowledge, we conclude that sleeping with burst transmission at transport-layer level is an effective energy saving mechanism during data transfer over a WLAN.

Based on the above knowledge, in Chapter 4, we proposed and designed SCTP tunneling for achieving energy efficiency of a wireless client in the presence of multiple TCP flows over a WLAN. SCTP tunneling has two features: flow aggregation and burst transmission. We extended the above energy consumption model and constructed a mathematical model of the energy consumed by SCTP tunneling to assess its energy efficiency. Through numerical examples based on the model, it was shown that the power consumption of SCTP tunneling was predominantly determined by the aggregate throughput of TCP flows, while burst transmission could considerably reduce power consumption with increasing moderate delay. In particular, for 800 [kbyte/s] of the aggregate TCP throughput, when 10 [ms] of additional delay is acceptable, we obtained 60% of power reduction

from power consumption when any delay is unacceptable. Implementation issues of SCTP tunneling are also discussed. From the above findings, we conclude that SCTP tunneling is a feasible solution to accomplish effective energy saving in the presence of multiple TCP flows on a wireless client in a WLAN environment.

As one of future directions of our work, we plan to combine the proposed method in Chapter 2 with other TCP modifications for wireless networks such as TCP Westwood [72], TCP Veno [73], and TCP-Jersey [74]. Another future direction is to apply the proposed method in Chapter 4 to other wireless networks such as cellar network and WiMAX. We believe that those discussions and results in this thesis contribute to the design and development of future wireless networks and transport-layer protocols.

Bibliography

[1] Wistron NeWeb Corp., "CM9: WLAN 802.11 a/b/g mini-PCI Module." available at `http://site.microcom.us/CM9.pdf`. [Accessed on Dec. 2012].

[2] R. Krashinsky and H. Balakrishnan, "Minimizing energy for wireless web access with bounded slowdown," *Wireless Networks*, vol. 11, no. 1-2, pp. 135–148, Jan. 2005.

[3] C. Andren, T. Bozych, B. Road, and D. Schultz, "PRISM power management modes: Application note AN9665," Feb. 1997.

[4] IEEE 802.15.1-2005, *Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*. IEEE, June 2005.

[5] IEEE 802.15.4-2003, *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)*. IEEE, 2003.

[6] IEEE 802.11-2007, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*. IEEE, June 2007.

[7] IEEE 802.16-2009, *Part 16: Air Interface for Broadband Wireless Access Systems*. IEEE, May 2009.

[8] Recommendation ITU-R M.1457-10, *Detailed Specifications of the Radio Interfaces of International Mobile Telecommunications-2000 (IMT-2000)*, June 2011.

[9] S. Pilosof, D. Raz, and Y. Shavitt, "Understanding TCP fairness over wireless LAN," in *Proceedings of INFOCOM 2003*, vol. 2, pp. 863–872, Apr. 2003.

[10] F. Keceli, I. Inan, and E. Ayanoglu, "TCP ACK congestion control and filtering for fairness provision in the uplink of IEEE 802.11 infrastructure basic service set," in *Proceedings of ICC 2007*, pp. 4512–4517, June 2007.

[11] J. Ha, E.-C. Park, K.-J. Park, and C.-H. Choi, "A cross-layer dual queue approach for improving TCP fairness in infrastructure WLANs," *Wireless Personal Communications*, vol. 51, no. 3, pp. 499–516, June 2009.

[12] T. Jin, G. Noubir, and B. Sheng, "WiZi-Cloud: Application-transparent dual ZigBee-WiFi radios for low power internet access," in *Proceedings of INFOCOM 2011*, pp. 1593–1601, Apr. 2011.

[13] Y. Fukuda and Y. Oie, "Unfair and inefficient share of wireless LAN resource among uplink and downlink data traffic and its solution," *IEICE Transactions on Communications*, vol. E88-B, no. 4, pp. 1577–1585, Apr. 2005.

[14] S. Kim, B. Kim, and Y. Fang, "Downlink and uplink resource allocation in IEEE 802.11 wireless LANs," *IEEE Transactions on Vehicular Technology*, vol. 54, no. 1, pp. 320–327, Jan. 2005.

[15] N. Blefari-Melazzi, A. Detti, I. Habib, A. Ordine, and S. Salsano, "TCP fairness issues in IEEE 802.11 networks: problem analysis and solutions based on rate control," *IEEE Transactions on Wireless Communications*, vol. 6, no. 4, pp. 1346–1355, Apr. 2007.

[16] B. Hirantha Sithira Abeysekera, T. Matsuda, and T. Takine, "Dynamic contention window control mechanism to achieve fairness between uplink and downlink flows in IEEE 802.11 wireless LANs," *IEEE Transactions on Wireless Communications*, vol. 7, no. 9, pp. 3517–3525, Sept. 2008.

[17] Atheros Communications, "Power consumption and energy efficiency comparisons of wlan products." In Atheros White Papers, May 2003.

[18] V. Raghunathan, T. Pering, R. Want, A. Nguyen, and P. Jensen, "Experience with a low power wireless mobile computing platform," in *Proceedings of ISLPED 2004*, pp. 363–368, 2004.

[19] Y. Agarwal, C. Schurgers, and R. Gupta, "Dynamic power management using on demand paging for networked embedded systems," in *Proceedings of ASP-DAC 2005*, pp. 755–759, Jan. 2005.

[20] G. Anastasi, M. Conti, E. Gregori, and A. Passarella, "Saving energy in Wi-Fi hotspots through 802.11 PSM : an analytical model," in *Proceedings of WiOpt 2004*, Mar. 2004.

[21] M. Ergen and P. Varaiya, "Decomposition of energy consumption in IEEE 802.11," in *Proceedings of ICC 2007*, pp. 403–408, June 2007.

[22] W.-K. Kuo, "Energy efficiency modeling for IEEE 802.11 DCF system without retry limits," *Computer Communications*, vol. 30, no. 4, pp. 856–862, Feb. 2007.

[23] E. Rantala, A. Karppanen, S. Granlund, and P. Sarolahti, "Modeling energy efficiency in wireless internet communication," in *Proceedings of MobiHeld 2009*, pp. 67–68, Aug. 2009.

[24] P. Agrawal, A. Kumar, J. Kuri, M. K. Panda, V. Navda, R. Ramjee, and V. N. Padmanabhani, "Analytical models for energy consumption in infrastructure WLAN STAs carrying TCP traffic," in *Proceedings of COMSNETS 2010*, pp. 1–10, Jan. 2010.

[25] M. Zorzi and R. Rao, "Energy efficiency of TCP in a local wireless environment," *Mobile Networks and Applications*, vol. 6, no. 3, pp. 265–278, June 2001.

[26] F. Vacirca, A. De Vebductis, and A. Baiocchi, "Optimal design of hybrid FEC/ARQ schemes for TCP over wireless links with Rayleigh fading," *IEEE Transactions on Mobile Computing*, vol. 5, no. 4, pp. 289–302, Apr. 2006.

[27] IEEE 802.11e, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Control (MAC) Quality of Service Enhancements.* IEEE, Oct. 2005.

[28] D. Camps Mur, X. Pérez-Costa, and S. Sallent Ribes, "An adaptive solution for wireless LAN distributed power saving modes," *Computer Networks*, vol. 53, no. 18, pp. 3011–3030, Dec. 2009.

[29] J. Lee, S. Kwon, and D. Cho, "Adaptive beacon listening protocol for a TCP connection in slow-start phase in WLAN," *IEEE Communications Letters*, vol. 9, no. 9, pp. 853–855, Sept. 2005.

[30] Y. He and R. Yuan, "A novel scheduled power saving mechanism for 802.11 wireless LANs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 10, pp. 1368–1383, Oct. 2009.

[31] K.-C. Ting, H.-C. Lee, H.-H. Lee, and F. Lai, "An idle listening-aware energy efficient scheme for the DCF of 802.11n," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 447–454, May 2009.

[32] J. Liu and L. Zhong, "Micro power management of active 802.11 interfaces," in *Proceedings of MobiSys 2008*, pp. 146–159, 2008.

[33] V. Namboodiri and L. Gao, "Energy-efficient VoIP over wireless LANs," *IEEE Transactions on Mobile Computing*, vol. 9, no. 4, pp. 566–581, Apr. 2010.

[34] F. R. Dogar, P. Steenkiste, and K. Papagiannaki, "Catnap: Exploiting high bandwidth wireless interfaces to save energy for mobile devices," in *Proceedings of MobiSys 2010*, pp. 107–122, June 2010.

[35] H. Yan, S. A. Watterson, D. K. Lowenthal, K. Li, R. Krishnan, and L. L. Peterson, "Client-centered, energy-efficient wireless communication on IEEE 802.11b networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 11, pp. 1575–1590, Nov. 2006.

[36] M. Hashimoto, G. Hasegawa, and M. Murata, "Fairness improvement of hybrid TCP congestion control mechanisms in wireless LAN environment," *Technical Report of IEICE (6th QoS Workshop of Technical Committee on Communication Quality)*, Dec. 2008. (in Japanese).

[37] M. Hashimoto, G. Hasegawa, and M. Murata, "Experimental evaluation of TCP congestion control mechanisms in wireless LAN environment with a novel performance metric," *Technical Report of IEICE* (NS2009-126), vol. 109, no. 326, pp. 33–38, Dec. 2009. (in Japanese).

[38] M. Hashimoto, G. Hasegawa, and M. Murata, "Trade-off evaluation between fairness and throughput for TCP congestion control mechanisms in a wireless LAN environment," in *Proceedings of SPECTS 2010*, July 2010.

[39] M. Hashimoto, G. Hasegawa, and M. Murata, "A transport-layer solution for alleviating TCP unfairness in a wireless LAN environment," *IEICE Transactions on Communications*, vol. E94-B, no. 3, pp. 765–776, Mar. 2011.

[40] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1–14, June 1989.

[41] M. Hashimoto, G. Hasegawa, and M. Murata, "Analysis of power consumption in TCP data transmission over a wireless LAN environment," *Technical Report of IEICE* (NS2010-105), vol. 110, no. 339, pp. 1–6, Dec. 2010. (in Japanese).

[42] M. Hashimoto, G. Hasegawa, and M. Murata, "Analysis of power consumption of a single TCP flow with burst transmission in a wireless LAN environment," in *Proceedings of the IEICE General Conference*, pp. S–107–S–108, Mar. 2011. (in Japanese).

[43] M. Hashimoto, G. Hasegawa, and M. Murata, "Energy efficiency analysis of TCP with delayed ACK in a wireless LAN environment," *Technical Report of IEICE* (NS2011-50), vol. 111, no. 144, pp. 1–6, July 2011.

[44] M. Hashimoto, G. Hasegawa, and M. Murata, "Modeling and analysis of power consumption in TCP data transmission over a wireless LAN environment," in *Proceedings of ICC Workshop (GreenComm 2011)*, pp. 1–6, June 2011.

[45] M. Hashimoto, G. Hasegawa, and M. Murata, "Energy efficiency analysis of TCP with burst transmission over a wireless LAN," in *Proceedings of ISCIT 2011*, pp. 292–297, Oct. 2011.

[46] M. Hashimoto, G. Hasegawa, and M. Murata, "An analysis of energy consumption for TCP data transfer with burst transmission over a wireless LAN," submitted to *International Journal of Communication Systems*, Dec. 2012.

[47] M. Hashimoto, G. Hasegawa, and M. Murata, "Proposal of SCTP tunneling for energy-efficient TCP data transfer over a wireless LAN environment," *Technical Report of IEICE* (IN2012-26), vol. 112, no. 88, pp. 13–18, June 2012. (in Japanese).

[48] M. Hashimoto, G. Hasegawa, and M. Murata, "Study on exploiting SCTP to save energy of TCP data transfer over a wireless LAN," in *Proceedings of the IEICE Society Conference*, pp. S–20–S–21, Sept. 2012. (in Japanese).

[49] M. Hashimoto, G. Hasegawa, and M. Murata, "Exploiting SCTP multistreaming to reduce energy consumption of multiple TCP flows over a WLAN," in *Proceedings of GreeNETs 2012*, Oct. 2012.

[50] M. Hashimoto, G. Hasegawa, and M. Murata, "SCTP tunneling: Flow aggregation and burst transmission to save energy for multiple TCP flows over a WLAN," submitted to *IEICE Transactions on Communications*, Oct. 2012.

[51] S. Floyd, "The NewReno modification to TCP's fast recovery algorithm," *Request for Comments 3782*, Apr. 2004.

[52] The Network Simulator ns-2. available at `http://www.isi.edu/nsnam/ns/`.

[53] R. Braden, "Requirements for internet hosts – communication layers," *Request for comments 1122*, Oct. 1989.

[54] Microsoft Corporation, *Microsoft Windows Server 2003 TCP/IP Implementation Details*, June 2003.

[55] P. Sarolahti and A. Kuznetsov, "Congestion control in linux TCP," in *Proceedings of 2002 USENIX Annual Technical Conference*, pp. 49–62, June 2002.

[56] J. Mo, R. J. La, V. Anantharam, and J. Walrand, "Analysis and comparison of TCP Reno and Vegas," in *Proceedings of INFOCOM 1999*, vol. 3, pp. 1556–1563, Mar. 1999.

[57] G. Hasegawa, K. Kurata, and M. Murata, "Analysis and improvement of fairness between TCP Reno and Vegas for deployment of TCP Vegas to the Internet," in *Proceedings of ICNP 2000*, pp. 177–186, 2000.

[58] A Linux TCP implementation for NS2. available at `http://netlab.caltech.edu/projects/ns2tcplinux/ns2linux/`.

[59] netem. available at `http://www.linuxfoundation.org/en/Net:Netem`.

[60] M. Mathis, J. Heffner, and R. Reddy, "Web100: Extended TCP instrumentation for research, education and diagnosis," *ACM Computer Communications Review*, vol. 33, no. 3, pp. 69–79, July 2003.

[61] F. Li, M. Li, H. Wu, M. Claypool, and R. Kinicki, "Measuring queue capacities of IEEE 802.11 wireless access points," in *Proceedings of BROADNETS 2007*, pp. 846–853, Sept. 2007.

[62] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "Iperf-the TCP/UDP bandwidth measurement tool." available at `http://dast.nlanr.net/Projects/Iperf/`.

[63] C. E. Koksal, H. Kassab, and H. Balakrishnan, "An analysis of short-term fairness in wireless media access protocols," in *Proceedings of SIGMETRICS 2000*, pp. 118–119, June 2000.

[64] J. Jun, P. Peddabachagari, and M. Sichitiu, "Theoretical maximum throughput of IEEE 802.11 and its applications," in *Proceedings of NCA 2003*, pp. 249–256, Apr. 2003.

[65] M. Allman, "On the generation and use of TCP acknowledgments," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 5, pp. 4–21, Oct. 1998.

[66] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, pp. 303–314, Oct. 1998.

[67] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *Proceedings of INFOCOM 2000*, vol. 3, pp. 1742–1751, Mar. 2000.

[68] R. Stewart, "Stream control transmission protocol," *Request for Comments 4960*, Sept. 2007.

[69] S. Ladha and P. D. Amer, "Improving file transfers using SCTP multistreaming," in *Proceedings of IPCCC 2004*, pp. 513–522, Apr. 2004.

[70] P. Natarajan, P. D. Amer, and R. Stewart, "Multistreamed web transport for developing regions," in *Proceedings of NSDR 2008*, pp. 43–48, Aug. 2008.

[71] Universal TUN/TAP driver. available at `http://vtun.sourceforge.net/tun/`.

[72] C. Casetti, M. Gerla, S. Mascolo, M. Sanadidi, and R. Wang, "TCP Westwood: End-to-end congestion control for wired/wireless networks," *Wireless Networks*, vol. 21, no. 2, pp. 216–228, Feb. 2002.

[73] C. P. Fu and S. C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 2, pp. 216–228, Feb. 2003.

[74] K. Xu, Y. Tian, and N. Ansari, "TCP-Jersey for wireless IP communications," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 4, pp. 747–756, May 2004.