2013

Utilizing ZigBee Technology for More Resource-efficient Wireless Networking

Hua Qin
Iowa State University

Utilizing ZigBee Technology for More Resource-efficient Wireless Networking

by

Hua Qin

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Program of Study Committee:
Wensheng Zhang, Major Professor
Yong Guan
Robyn R. Lutz
Daji Qiao
Johnny S. Wong

Iowa State University

Ames, Iowa

2013

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Wireless networks have been an essential part of communication in our daily life. Targeted at different applications, a variety of wireless networks have emerged. Due to constrained resources for wireless communications, challenges arise but are not fully addressed. Featured by low cost and low power, ZigBee technology has been developed for years. As the ZigBee technology becomes more and more mature, low-cost embedded ZigBee interfaces have been available off the shelf and their sizes are becoming smaller and smaller. It will not be surprising to see the ZigBee interface commonly embedded in mobile devices in the near future. Motivated by this trend, we propose to leverage the ZigBee technology to improve existing wireless networks. In this dissertation, we classify wireless networks into three categories (i.e., infrastructure-based, infrastructure-less and hybrid networks), and investigate each with a representative network. Practical schemes are designed with the major objective of improving resource efficiency for wireless networking through utilizing ZigBee technology. Extensive simulation and experiment results have demonstrated that network performance can be improved significantly in terms of energy efficiency, throughput, packet delivery delay, etc., by adopting our proposed schemes.

CHAPTER 1. INTRODUCTION

Wireless networks have been an essential part of communication in our daily life. Targeted at different applications, a variety of wireless networks have emerged. According to network architecture, they can be divided into three categories: infrastructure-based, infrastructure-less and hybrid wireless networks, as shown in Fig. 1.1.

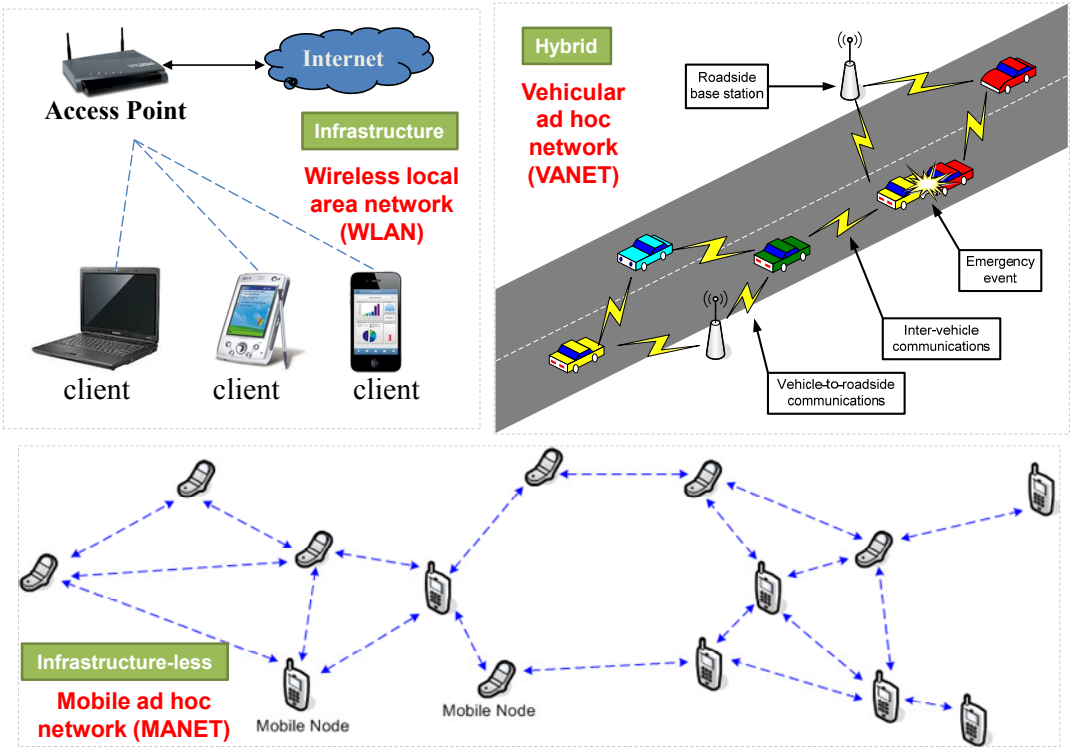


Figure 1.1: Types of wireless networks

A Wireless Local Area Network (WLAN) links wireless devices (e.g., laptop, smartphone, PAD, etc.) together, and usually provides a connection to the Internet through a fixed infrastructure called access point (AP). Thus, WLAN is a typical *infrastructure-based* wireless network. Besides, wireless

devices can also form a Mobile Ad hoc NETwork (MANET) to enable device-to-device communication without relying on any infrastructure, achieving the advantages such as self-organizing and rapid deployment. Such network can be used to enable responsive communications for rescue operations in emergency situations such as earthquakes and hurricanes, where infrastructures may be destroyed. Hence, MANET is an example of *infrastructure-less* wireless network. With the major objective of improving driving safety and comfort, Vehicular Ad hoc NETwork (VANET) has emerged recently. In VANET, vehicles are equipped with wireless devices, through which they form an infrastructure-less MANET to communicate with each others; meanwhile, they can also connect to roadside infrastructures to access the Internet or obtain current traffic conditions. Therefore, VANET is an instance of *hybrid* wireless network.

1.1 Challenges in Wireless Network

Several challenges have been widely recognized in designing and deploying wireless networks.

- **Energy Efficiency:** To provide high data rate, a wireless device typically needs to consume energy also at a high rate. Unfortunately, energy is often very constrained on wireless devices as they are often powered by batteries with limited capacity. To improve energy efficiency, many protocols have been proposed. However, their performance may vary greatly in different scenarios because of dynamic network conditions (e.g., unpredictable traffic, high mobility, unreliable communication, etc.).

- **Quality of Service (QoS):** Since wireless channel is a shared resource, there are potentially a large number of wireless devices co-located in a small area. If many devices transmit at the same time, the contention among them would be intensive, which could incur long latency, low throughput and high packet loss rate. Thus, ensuring a desire level of QoS is important and challenging.

- **Dynamics:** As wireless devices are usually mobile, network topology and traffic may change frequently and unpredictably, which makes network optimization difficult.

- **Heterogeneity:** Different wireless devices may play different roles or have different resources (e.g., energy capacity, data rate, communication range, etc.). For example, mobile devices are commonly equipped with long-range WiFi interfaces [1] for accessing the Internet and short-range Bluetooth [2] for exchanging data between them. Hence, heterogeneity should be taken into account when designing wireless networking protocols.

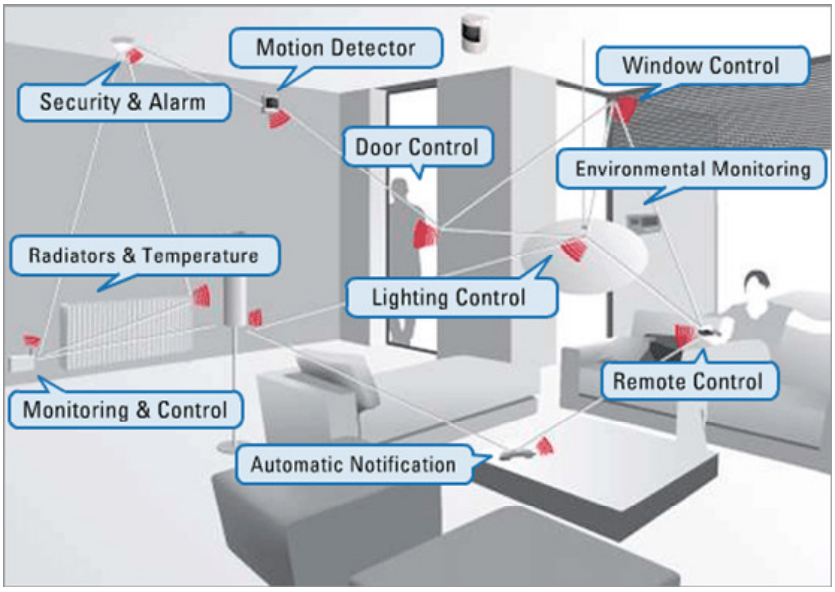


Figure 1.2: An example of ZigBee-based application [3]

1.2 ZigBee Technology

Based on the IEEE 802.15.4 standard, ZigBee technology [4] has been developed and deployed for years. It is a wireless technology developed as an open global standard to address the unique needs of low-cost, low-power wireless sensor and control networks. ZigBee has a defined data rate of 250 Kbps, best suited for periodic or intermittent data transmission. Applications include wireless light switches, electrical meters with in-home-displays, traffic management systems, and other industrial equipments that require short-range wireless transfer of data at relatively low rate. The ZigBee technology is intended to be simpler and less expensive than other wireless technologies, such as Bluetooth or WiFi [5].

	ZigBee	Bluetooth	WiFi
Standard	802.15.4	802.15.1	802.11
Data Rate	250 Kbps	1 Mbps	1∼54 Mbps
Range	10∼80 m	10 m	50∼90 m
Operating Frequency	868 MHz; 916 MHz; 2.4 GHz	2.4 GHz	2.4 GHz
Complexity	Low	Normal	High
Cost	Low	Low	High
Ease of Use	Easy	Normal	Hard
Battery Lifetime	Years	Days	Hours

Table 1.1: Comparison of different wireless standards

Due to low power consumption, ZigBee devices can sustain themselves on inexpensive batteries for months, or even years. This makes them ideal for a host of monitoring and control applications (Fig. 1.2) like home/building automation, lighting controls, smart energy monitoring, mobile payment, and automatic meter reading, where ZigBee technology is making significant advancements. Particularly, one typical application of ZigBee in wireless network is *sensor network*, where ZigBee is used for the communications among sensors with the purpose of prolonging network lifetime.

ZigBee modules are also easy to embed, which allows people to integrate them with some other devices for higher level of efficiency and convenience. As the ZigBee technology becomes more and more mature, embedded ZigBee interfaces have emerged and been equipped in wireless devices. Particularly, the first Android phone with ZigBee, TazTag TPH-One [6], has emerged recently. Thus, the ZigBee interface is expected to be more and more commonly embedded in wireless devices together with other interfaces such as WiFi and Bluetooth in near future.

With all these unique features, we envision that utilizing the ZigBee technology can greatly improve resource efficiency for wireless networks. In this dissertation, we investigate the aforementioned wireless networks as follows.

- **WLAN:** In WLAN, the high-power WiFi interface is often used for wireless communications. Although it provides a good combination of throughput and range, it accounts for a large proportion of wireless devices' total energy consumption. With the assistance of low-power ZigBee, high-power WiFi interface can safely go to sleep when there is no data to transmit, and wake up immediately when there is some incoming data to receive. This presents an ideal *energy-efficient*

data transmission pattern.

- **MANET:** In MANET, since there is no infrastructure to provide coordination among mobile devices, a considerable amount of bandwidth and energy may be wasted for contention or idle listening. To address these issues, ZigBee can be used to coordinate mobile devices in a certain manner to minimize the probability of contention and/or the duration of idle listening. In this way, wireless communications in MANET could become more *bandwidth-efficient* and *energy-efficient*.

- **VANET:** In VANET, due to high deployment costs, roadside infrastructures are sparsely deployed. So, the safety-related information collected by vehicles may get lost due to unavailable network connectivity. Since ZigBee-enabled sensors are much cheaper than conventional roadside infrastructures, we can use inexpensive ZigBee-enabled sensors to replace expensive roadside infrastructures in order to achieve *investment-efficient* monitoring of road conditions and *communication-efficient* interactions among vehicles.

1.3 Proposed Research

The major objective of our research is to *utilize ZigBee technology to build more resource-efficient wireless networking systems*. We target at identifying the critical issues that affect network performance, and creating *practical* solutions that can be implemented on wireless devices and offer greater performance in terms of system energy efficiency, delay, throughput, etc.. The following research topics are included in this dissertation:

- *Utilizing ZigBee for more energy-efficient WLAN*

  We first identify the limitations of the standard power saving management widely used in WLAN. Then, to overcome these limitations, we propose a ZigBee-assisted Power Saving Management (ZPSM) scheme, leveraging low-power ZigBee interface to wake up sleeping high-power WiFi interface on demand of current traffic. The goal is to improve clients' energy efficiency without violating their delay requirements. A prototype of the proposed system is implemented on wireless devices to verify its performance in real-world environments.

- *Utilizing ZigBee for more resource-efficient MANET*

  We study the issues in current research on MANET and design two ZigBee-based systems with different objectives and applications. One targets at improving energy efficiency of MANET in emergency situations. The other aims to improve bandwidth efficiency of MANET in high traffic density scenarios.

  - *Utilizing ZigBee for more energy-efficient MANET:* We first identify the challenges for deploying a MANET in emergency situations. Then, we conduct a theoretical analysis to study energy consumption and delay of multi-hop communications in MANET, based on which a practical system is proposed to prolong network lifetime with bounded end-to-end packet delivery delay. Finally, we implement a prototype system to evaluate our design in various scenarios.
  - *Utilizing ZigBee for more bandwidth-efficient MANET:* We first study the impact of WiFi contentions on network throughput via experiments. Motivated by the experiment results, we design a protocol atop MAC layer with the basic idea of leveraging the ZigBee interface to coordinate transmission activities of WiFi interfaces for reduced contention and thereby improved network throughput. To evaluate feasibility and performance of our designed system, we implement and test a prototype system in a realistic testbed.

- *Utilizing ZigBee for more resource-efficient VANET*

  To overcome the limitations of conventional pure VANET, we propose to integrate the VANET with the inexpensive ZigBee-enabled wireless sensor network (WSN). Along with the concept of VANET-WSN integration, new challenges arise. We identify and investigate these challenges and propose schemes for effective and efficient vehicle-sensor and sensor-sensor communications. The designed networking system is much more efficient than traditional VANET in terms of investment and communication.

The rest of the dissertation is organized as follows. In Chapter 2, we discuss recent research efforts for improving performance of wireless networks. Chapter 3 proposes our research on utilizing ZigBee for more energy-efficient WLAN. Then, we present our research on utilizing ZigBee for more

energy-efficient and bandwidth-efficient MANET in Chapter 4 and Chapter 5, respectively. After that, Chapter 6 reports our research on utilizing ZigBee for more resource-efficient VANET. Finally, in Chapter 7, we conclude this dissertation with a summary of the major contributions and a discussion about possible future research directions.

CHAPTER 2. RELATED WORK

Numerous efforts have been made to improve resource efficiency for traditional wireless networks and huge advances have been witnessed in the last decade. However, due to the difficulty of the fundamental challenges in wireless networks, as mentioned in Chapter 1.1, their accomplishments are still limited.

WLAN: A considerable amount of research has been conducted to improve energy efficiency of WiFi-enabled mobile devices in WLAN, especially for web browsing applications. Particularly, power saving management (PSM) has been standardized and widely used for many years. Based on PSM, many protocols have been proposed to further improve energy efficiency in WLAN. For example, authors in [7, 8] proposed to minimize energy consumption with bounded slowdown. To reduce the congestion at the AP and thus improve the performance of the standard PSM, an opportunistic PSM is proposed in [9] to allow only one client download at any time. One common shortcoming of these schemes lies in that, their savings largely depend on the accuracy in predicting client's network usage patterns, because they are not able to wake up asleep clients at will without the assistance of additional interfaces. Thus, their performance may be limited. Besides, GreenCall [10] has been proposed to save energy while preserving quality of VoIP application in WLANs by leveraging PSM. Cell2Notify [11] takes advantage of cellular network to improve energy efficiency of VoIP over WiFi-enabled smartphones. Although these schemes can achieve good performance for real-time applications, they may not be applicable to delay tolerant applications, such as short file transfers, web browsing, video streaming, etc.

MANET: Power saving management (PSM) has been also proposed for distributed coordination function (DCF), which is IEEE 802.11's standard for ad hoc networks. Although PSM can reduce en-

ergy consumption significantly, it could incur long packet delivery delay, especially in multi-hop ad hoc networks. To reduce delay, a number of protocols have been designed on top of PSM. Basically, they can be categorized into two classes: *proactive* and *reactive*. Proactive protocols (such as SPAN [12] and Odds [13]) typically form a backbone consisting of a set of nodes operating in constantly awake mode (CAM) while other nodes stay in PSM to save energy. Their major drawback is that backbone nodes have to stay awake even if there is no traffic in the network. Reactive schemes (such as ODPM [14] and TITAN [15]) switch nodes between CAM and PSM on demand of current traffic. One of their common shortcomings is that, the energy saving depends on the accuracy in predicting data traffic, which is difficult in practice. Thus, the performance may be poor in irregular traffic scenarios [16]. Moreover, energy-aware routing protocols have been also proposed to either improve energy efficiency [17, 18] or extend network lifetime [19, 20]. Besides, various MAC protocols have been proposed to improve the throughput of IEEE 802.11 ad hoc network. Overlay MAC [21] proposes a multi-hop TDMA-typed MAC protocol by employing a distributed algorithm to allocate time slots among nodes and also implements a precise control over time slots. Koutsonikolas et al. [22] design and implement a TDM MAC protocol for multi-hop wireless mesh networks using a programmable wireless platform. Inspired by the IEEE 802.4 Token Ring protocol, the token-passing protocols, such as [23] and [24], have been designed to the performance of IEEE 802.11 network. However, these protocols rely on underlying MAC layer and thereby may introduce lots of control overheads, which could degrade their performance.

VANET: Several ongoing projects [25–27] are researching and developing the network infrastructure to facilitate vehicular communications and applications. Especially, with the advance of ZigBee and semiconductor technologies, the sensor becomes a good candidate to facilitate the communication in VANET. A static sensor-assisted adaptive routing protocol [28] has been proposed, where static sensors are deployed at intersections to facilitate the routing by temporarily storing messages. A prototype of hybrid sensor-vehicular networks has been proposed in [29]. Besides, many related applications have been investigated. Bohil et al. [30] propose a secure WSN-based roadside architecture to support accident prevention and post-accident investigation. But very little detail is given on the underlying network design and implementation. MobEyes (Smart Mobs for Urban Monitoring) has been developed in [31]. It exploits vehicle mobility to opportunistically diffuse summaries of sensed

data. An in-vehicle sensor network has been proposed for remote vehicle diagnosis and management in [32], where sensors inside and around the vehicle form a hierarchical network. However, most of existing works directly apply the technologies in the conventional WSN to the VANET but do not consider much the unique characteristics of the VANET (e.g., high mobility, pre-defined layout, potential interference, dynamic traffic, etc.). The collaboration between sensors and vehicles has not been exploited adequately. Therefore, a more seamless and profitable integration of the WSN and the VANET is still demanded.

To overcome the above-mentioned limitations of traditional wireless networks and further advance wireless networking efficiency, lots of recent research has been conducted to exploit the potential of co-located interfaces for more efficient wireless communications. One of the first works that brought forth the idea is CoolSpots [33], which proposed some basic policies to enable a mobile device to automatically switch between multiple radio interfaces such as WiFi and Bluetooth, in order to increase battery lifetime. As the first concrete implementation of such system, Blue-Fi [34] uses the co-located Bluetooth to predict the availability of the WiFi connectivity, which allows the device to intelligently turn the WiFi interface on only when there is WiFi connectivity available. Another Bluetooth-assisted protocol has been proposed in [35] to reduce power consumption of WLAN by using Bluetooth to form a cluster, called Bluetooth Personal Area Network (PAN). The cluster head acts as a gateway between the PAN and the WLAN, enabling clients to access the AP via low-power Bluetooth.

By leveraging the unique features of long communication range and low power consumption, ZigBee technologies have been used recently to improve efficiency for WiFi networks in mobile environments. A number of novel ZigBee-WiFi systems have been developed. WiZi-Cloud [36] was proposed to use WiFi and ZigBee radios on mobile phones and APs to achieve ubiquitous connectivity, high energy efficiency, and real time intra-device/inter-AP handover. Besides, protocols have been designed to allow a ZigBee radio to capture WiFi signals. ZiFi [37] was proposed to utilize ZigBee radios to identify the existence of WiFi networks through unique interference signatures generated by WiFi beacons. HoWiES [38] is a WiFi-ZigBee message delivery system that enables WiFi radios to convey different messages to ZigBee radios in mobile devices for saving WiFi energy in scanning, standby and wakeup respectively. Similar to these work, our research also focuses on the study of ZigBee-WiFi systems,

where the WiFi interface is mainly responsible for data transmission while the ZigBee interface is used to coordinate or assist WiFi communications in order to deliver higher level of resource efficiency. However, different from these works which target at certain specific applications or networks, our work aims to leverage ZigBee technologies to achieve resource efficiency for more generic wireless networking, which could potentially benefit a wide range of applications in various networks. Besides, because of using different methodologies, the accomplishment is also different.

CHAPTER 3. UTILIZING ZIGBEE FOR MORE ENERGY-EFFICIENT WLAN

3.1 Overview

In WLAN, the AP and clients are equipped with WiFi interfaces for data transmission. Although the WiFi interface can provide good combination of throughout and range, its energy efficiency is very low. To save energy, power saving management (PSM) has been widely used in WiFi devices. With the standard PSM, a client periodically wakes up to receive beacon frames from its AP. If there is an incoming packet buffered at the AP, the client retrieves the packet; otherwise, it goes to sleep. Although PSM can reduce energy consumption, the energy efficiency varies greatly under different traffic patterns. For example, suppose a client wakes up every 0.1s (default value) to receive beacons and its packet arrival rate is 1pkt/s on average (e.g., web browsing). Then, the achieved energy-efficiency, defined as the ratio of *useful wakeups* (i.e., the number of wakeups during which the client has packets retrieved from the AP) to the total number of wakeups, is only 10%. This is because clients do not know the arrival time of incoming packets at the AP and thereby need to wake up periodically, resulting in an inefficient utilization of energy. To improve the energy efficiency, adaptive PSM schemes [7] have been proposed and widely applied. The basic idea is to let the WiFi radio switch between the PSM and CAM (constantly awake mode) modes according to some heuristics. However, high energy efficiency may still not be achieved when the traffic pattern drastically changes.

To deal with these issues, we propose a ZigBee-assisted power saving management (ZPSM) for WiFi devices, aiming to deliver energy efficiency with bounded packet delivery delay. The key idea is to use the low-power ZigBee radio to dynamically wake up sleeping high-power WiFi radio for packet transmission between the AP and clients. Unlike the standard PSM, ZPSM system presents a wakeup strategy which is adapted to both *packet arrival rate* and *delay requirements* in order to maximize energy efficiency. Moreover, ZPSM is built atop the standard PSM, and thereby, requires no change to

the WiFi standard.

A prototype of the designed system has been developed, and experiments based on the the prototype have been conducted to evaluate the feasibility and performance of the design. In addition, to evaluate the performance of ZPSM in a large-scale network, a detailed ns2-based simulator is built and extensive simulations have been conducted. The results show that our proposed system can significantly reduce power consumption in a wide range of scenarios, compared to the standard PSM, and achieve a level of energy efficiency approximating an optimal value derived from our theoretical analysis.

In the rest of this chapter, Section 3.2 presents preliminaries, followed by theoretical analysis in Section 3.3. Section 3.4 elaborates the proposed design of ZPSM. The results of comprehensive simulation and prototype implementation are reported in Section 3.5 and 3.6, respectively. Section 3.7 summarizes this chapter.

3.2 Preliminaries

3.2.1 Power Management for WiFi Devices

WiFi devices usually support two power management modes: power saving mode (PSM) in which the radio periodically wakes up to receive data packets so as to reduce idle listening and thereby energy consumption, and constantly awake mode (CAM) in which the radio keeps awake and therefore data packets can be received promptly at the cost of high power consumption.

In PSM, the AP broadcasts beacon frames every beacon interval (BI); each client wakes up every certain number of BIs, called *listening interval* (LI), to check whether it has data packets buffered at the AP. The AP indicates the presence of buffered packets by setting the Traffic Indication Map (TIM) fields in the beacon frame. If a client finds its corresponding TIM field set, it sends a Power Save Polling (PS-POLL) frame to retrieve buffered packets. The AP also uses MORE bit in a data packet to indicate if more packets are buffered, helping the client to decide whether it can go to sleep after receiving the packet. By default, BI is fixed to 100ms. Parameter LI is configurable, and its setting influences the performance.

With PSM, clients are allowed to retrieve packets only after receiving a beacon frame. In the worst case, a packet may have a delay up to *two* BIs, i.e., one BI delay for the AP to wait for client to

wake up and one BI delay for the client to be served by the AP as there may exist other clients to be served simultaneously. Hence, PSM may not be able to guarantee packet delivery delay lower than two BIs. For example, a TCP data packet typically requires to be ACKed within 200ms; otherwise, a TCP retransmission is triggered. Thus, PSM may have to switch to CAM sometimes (as in widely-used adaptive PSM [7]). However, this may waste lots of energy on idle listening, especially when the traffic rate is low.

Based on the desired delivery delay bound, we classify data packets into two categories: *short delay (SD)* data packets (with a delay shorter than two BIs) and *long delay (LD)* data packets (with a delay bound longer than two BIs). Applications that can tolerate relative long delay may include text messaging with which users typically spend seconds or longer for typing or thinking before sending out a message and email client which checks the server for newly arriving emails only every several minutes. In these scenarios, a message delivery delay of a few seconds could be acceptable. Other examples include UDP-based file transfers during web browsing where in between two requests there is a relatively long period of inactivity or think time, video streaming in which data are intermittently buffered in advance for users, and so on. In this chapter, we study the delay-bounded delivery of both SD and LD data packets.

3.2.2 A Realistic Concern: Interference

To utilize the co-existence of WiFi and ZigBee interfaces, a practical concern is how severely they can interfere with each other, as both ZigBee and WiFi interfaces work on the 2.4 GHz frequency band. Experiments [39,40] have shown that WiFi communication can interfere ZigBee communication severely if their working channels overlap. However, if their channels do not overlap, the interference becomes insignificant. To verify the validity of the above observation when WiFi and ZigBee interfaces are co-located in the same mobile station, we further conducted experiments to measure the impacts of interference on packet delivery ratio (PDR) of ZigBee interfaces in this scenario. Both indoor (e.g, library) and outdoor (e.g., parking lot) experiments have been conducted, using two laptops. Each laptop has a ZigBee interface (i.e., attached Crossbow telosB mote) and a WiFi interface. Two WiFi interfaces uniformly generate data traffic in both directions and run the IEEE 802.11g protocol to keep

network throughput saturated, while one ZigBee interface transmits packets at a constant rate to the other. The channel of ZigBee interfaces is fixed to Channel 26, while WiFi is tuned to Channel 6 (default channel) or Channel 11 (closest to Channel 26).

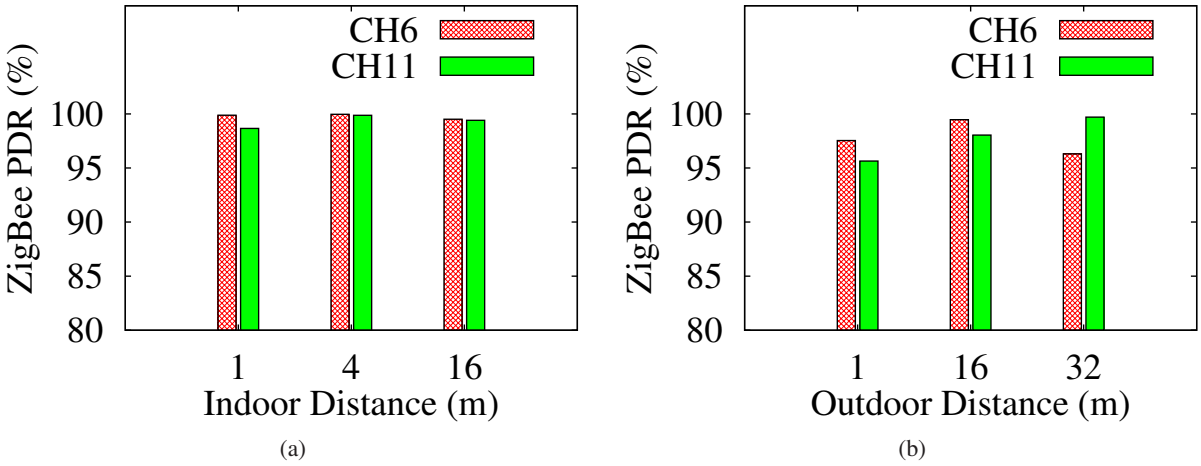


Figure 3.1: Packet delivery ratio of ZigBee

The experimental results, as shown in Fig. 3.1, indicate that, when ZigBee and WiFi interfaces use non-overlapping channels, the packet delivery ratio of ZigBee communication is high ($> 95\%$) and the WiFi communication is nearly not affected, which not only motivates but also supports our leveraging the co-existed ZigBee interface to facilitate the WiFi power management.

3.2.3 System Model

In our proposed system, each of the AP and clients has a ZigBee and a WiFi interface. The WiFi interface is for data transmission while the ZigBee interface is for power management. The WiFi and ZigBee interfaces of the AP are always awake, but the interfaces of clients are awake intermittently for energy conservation. In addition, each client can run either the standard PSM (SPSM) or the ZigBee-assisted PSM (ZPSM). Particularly, when a client is out of the ZigBee range (but still in the WiFi range) of the AP, it defaults to SPSM.

Each client i has a desired *delay bound* for downlink packet transmission. Specifically, the percentage of packets received with a delay lower than the desired *delay bound* d_i among all incoming packets

should be at least δ_i (called *delay-meet ratio*), where $0 < \delta_i < 1$. This is called *delay requirement*. Here, the delay is defined as the time elapsed from the arrival of a packet at the AP to the receipt of the packet at the destination client. Besides, client i is called *short delay (SD) client* if d_i is smaller than two BIs; otherwise, it is called *long delay (LD) client*.

As with the SPSM, we assume all clients are time synchronized with the AP. In addition, due to the unreliable link quality of ZigBee channel, ZigBee transmission may fail; also, as the ZigBee interface at a client may be used for other purposes, packets transmitted by the ZigBee interface at the AP may fail to reach the client occasionally. We use the link quality p_i to represent the probability that a packet sent by the AP through its ZigBee interface arrives at client i successfully. Note that the value of p_i may vary over time.

The AP is static while the clients can be mobile. We assume that the mobility of clients is relatively low. For example, the mobile WiFi devices may be carried by people who stay in conference rooms, libraries, cafe shops, stadiums, etc., where it is typical that a client is static, or moves for a while and then pauses for a while and so on and so forth, following the well-known *random waypoint model*.

3.2.4 Design Objectives

- *Energy Efficiency*: Through minimizing unnecessary wakeup and idle listening, our design should decrease the overall power consumption (including both WiFi and ZigBee power consumption) of clients in various scenarios.
- *Bounded Delay*: Our system should be able to satisfy the delay requirements for each client.
- *Compatibility*: Due to the popularity and diversity of the WiFi devices, our system should not demand changes to the IEEE 802.11 standards. The system should be built atop the standard PSM and transparent to the underlying standard PSM.

3.2.5 Wakeup Strategies

To successfully transmit a data packet from the AP to a client, the WiFi interface of the client should be turned on. This can be achieved by using the following two wakeup strategies.

- *Regular wakeup*: The client's WiFi interface wakes up every one LI to retrieve buffered data packets from the AP. This *proactive* wakeup strategy has been supported by the SPSM.

- *On-demand wakeup*: The client's WiFi interface is woken up on demand by the AP through the ZigBee communication. This is a *reactive* (on-demand) wakeup strategy. With this strategy, the ZigBee interface of the AP is always awake and broadcasts a *wakeup frame* every certain time interval, called *wakeup interval* (WI), through its ZigBee interface. A wakeup frame contains information about when each client should wake up. Once a client receives a wakeup frame indicating a wakeup request, it turns on its WiFi interface to retrieve buffered data packets at the *specified* time. On-demand wakeup works differently for SD and LD clients, as elaborated below.

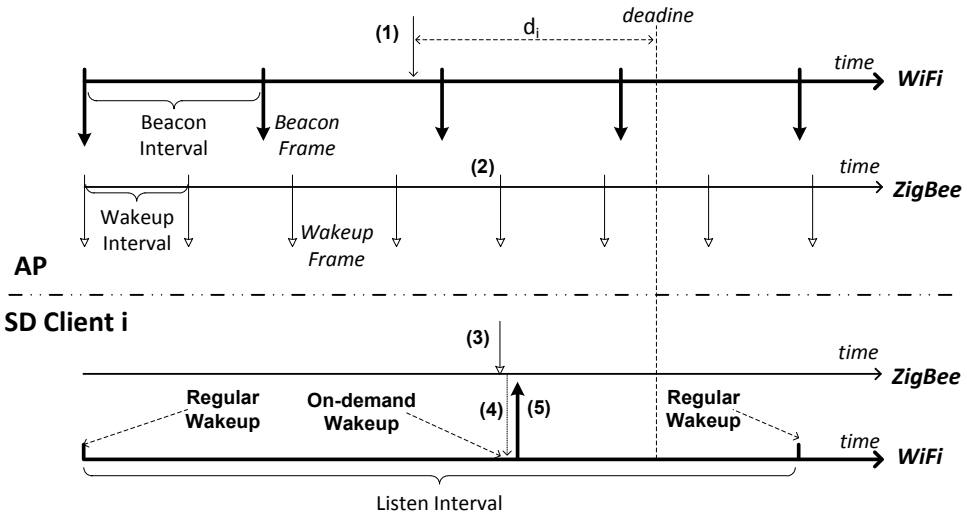


Figure 3.2: The wakeup behaviors of the WiFi and ZigBee interfaces at AP and SD client i (LI = 4). (1) A packet with a delay shorter than two BIs arrives at the AP. (2) The AP broadcasts a wakeup frame through its ZigBee interface. (3) Client i receives the wakeup frame. (4) Client i turns on its WiFi interface immediately. (5) Client i sends a PS-POLL and retrieves the packet from the AP.

  - *SD client*: The AP indicates an on-demand wakeup request in wakeup frame once there is an incoming SD data packet. Thus, once the target SD client receives the wakeup frame, it *immediately* turns on its WiFi interface and sends a PS-POLL to the AP to retrieve the packet, as the example shown in Fig 3.2. With on-demand wakeup strategy, SD client

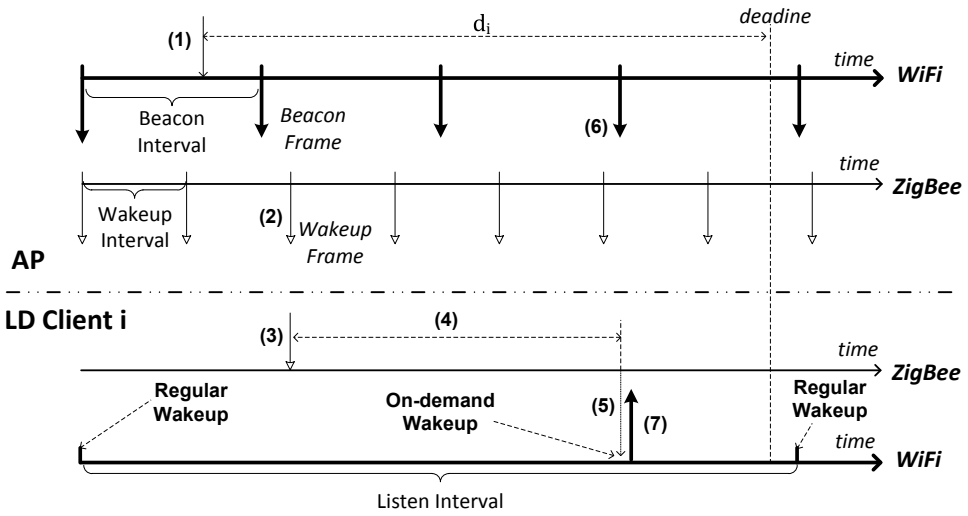


Figure 3.3: The wakeup behaviors of the WiFi and ZigBee interfaces at AP and LD client i (LI = 4). (1) A packet with a delay longer than two BIs arrives at the AP. (2) The AP broadcasts a wakeup frame through its ZigBee interface. (3) Client i receives the wakeup frame. (4) Client i waits for a while in order for the AP to buffer more possible incoming packets. (5) Client i turns on its WiFi interface at a later BI. (6) The AP broadcasts a beacon frame. (7) Upon receiving the beacon frame, client i sends a PS-POLL and retrieves all buffered packets from the AP.

spends less time staying in CAM than if it adopts adaptive PSM in which it should stay in CAM all the time.

– *LD client*: Since LD client can tolerate a relatively long delay, it does not need to wake up immediately once there is an incoming packet. Instead, it can let the AP wait for longer time as long as the corresponding delay requirements can be satisfied. In this way, more packets may be buffered and retrieved at once, which can reduce unnecessary wakeups and thereby improve energy efficiency. Besides, for compatibility, LD client always wakes up at the beginning of specified BI to retrieve data packets following SPSM. An example is shown in Fig 3.3.

Apparently, using on-demand wakeup only can maximize energy efficiency as clients wake up their WiFi interfaces only when necessary. However, due to the unreliable link quality of ZigBee channel and occasional unavailability of ZigBee interface at a client, the delay requirements may not always be satisfied. On the other hand, the regular wakeup strategy can guarantee certain delay bound by setting

appropriate LI, but this strategy may spend much energy on unnecessary wakeups. Hence, our proposed design aims to employ these two approaches in a combined manner to strike a balance between energy consumption and delay.

As the AP needs to dynamically determine the best time to wake up LD clients while SD clients simply are woken up upon packet arrival, optimizing wakeup behaviors of LD clients is more complex than that of SD clients. Thus, in the following, we focus our study on LD clients. However, our practical design is extended to support SD clients as well. Unless specified otherwise, we use *LD client* and *client* interchangeably.

3.3 Theoretical Study

To provide a theoretical foundation, this section develops an optimization problem step by step to formulate the design challenge in our proposed system. The analysis results are used as a guide to design a practical scheme and as a reference to evaluate the practical design.

3.3.1 Problem Definition

The problem to be solved in our proposed system is *how to schedule the regular and on-demand wakeups for each client so as to minimize the overall energy consumption (including both WiFi and ZigBee) of all clients while satisfying their delay requirements*.

For simplicity, we only consider the behaviors of ZPSM clients when the system is in *steady state*. The notations used are listed in the following table.

3.3.2 Assumptions

The following assumptions are made to simplify the analysis, though *the practical design to be presented in the following section is not restrained by these assumptions*.

- Uplink data traffic (i.e., data traffic from clients to the AP) and the data traffic to/from CAM clients are not considered.
- Downlink data packets for each client arrive at the AP following the Poisson process [35,41,42].

Z	the set of ZPSM clients
B	the length of a BI (the default value is 100 ms)
W	the length of a wakeup slot
m	the number of wakeup slots contained in a WI
x_i	the number of on-demand wakeups for client i within a LI
y_i	the number of BIs contained in a LI of client i
p_i	the ZigBee link quality between the AP and client i
d_i	the desired delay bound for packets targeted at client i
δ_i	the required delay-meet ratio for client i
λ_i	the average arrival rate of packets for client i
θ_i	the success probability of any on-demand wakeup of client i
τ_i	the expected interval between two on-demand wakeups of client i

- Ideal WiFi channel conditions, meaning no packet loss, are assumed. The packet delay due to contention can be either negligible or constant.
- The size of all data packets is the same.
- The system is not saturated and no packet is dropped due to overflow of the queue. Thus, the buffered packets for clients will be eventually sent.

3.3.3 Delay Analysis

We first analyze the relation among m (i.e., the length of a WI), x_i (i.e., the number of successful on-demand wakeups of client i during one LI) and y_i (i.e., the number of BIs contained in one LI of client i) that should be satisfied in order to meet the delay requirements.

Consider the packet transmission between the AP and a ZPSM client i during a LI. Without loss of generality, the period is assumed to be from time instant 0 to y_iB. Let λ_i denote the average arrival rate of packets targeted at client i. Then, the number of packets whose delay bound can be guaranteed through regular wakeup is $(d_i - B)\lambda_i$, because all packets arriving between time $y_iB - (d_i - B)$ and y_iB can be transmitted during the BI following the next regular wakeup (i.e., between time y_iB and $y_iB + B$) with a delay less than d_i. Thus, the number of packets that need on-demand wakeup during a BI is $y_iB\lambda_i - (d_i - B)\lambda_i = (y_iB - d_i + B)\lambda_i$.

As the ZigBee transmission may fail, on-demand wakeup cannot be always guaranteed. To deal with this, we assume that, once the AP sets the corresponding bit in wakeup frame, it keeps that bit set

until the AP receives a PS-POLL (indicating the client wakes up and retrieves packets) from that client, which can be modeled as Geometric distribution. Besides, instead of staying awake all the time, the ZigBee interface of each client wakes up every *wakeup slot* (denoted by W) to save energy. By default, W is set to 40ms (i.e., the typical time to exchange one message between the AP and a client via their ZigBee interfaces). The AP broadcasts wakeup frame every certain number of wakeup slots, denoted by m. Thus, wakeup interval can be computed as mW. Then, for a client i with link quality p_i, the success probability of any on-demand wakeup, denoted by θ_i, can be computed as

$$\theta_i = 1 - (1 - p_i)^{\frac{d_i - B}{mW}}. \tag{3.1}$$

This is because any packet arriving $d_i - B$ time before an on-demand wakeup can be transmitted during the BI following that wakeup with a delay less than d_i.

Therefore, the delay requirements of client i can be defined as

$$\delta_i \leq \frac{(y_i B - d_i + B)\theta_i + (d_i - B)}{y_i B} \leq 1. \tag{3.2}$$

From Eq. (3.2), we can solve y_i and get

$$y_i \geq \frac{d_i - B}{B} \tag{3.3}$$

and

$$(\delta_i - \theta_i) y_i \leq \frac{(1 - \theta_i)(d_i - B)}{B}. \tag{3.4}$$

For Eq. (3.4), there exist the following two cases.

- **Case I:** if $\delta_i \leq \theta_i$, the inequality always holds regardless the value of y_i. This indicates that the delay requirements can be satisfied through only on-demand wakeup. Since the IEEE 802.11 standard only specifies 2 bytes to represent the LI parameter, $y_i \leq Y_{max} = 2^{16} - 1$, where Y_{max}

denotes the maximum LI. Combining it with Eq. (3.3), we have

$$\frac{d_i - B}{B} \leq y_i \leq Y_{max}. \tag{3.5}$$

- **Case II:** if $\delta_i > \theta_i$, then

$$\frac{d_i - B}{B} \leq y_i \leq \frac{(1-\theta_i)(d_i - B)}{B(\delta_i - \theta_i)}, \tag{3.6}$$

which indicates that on-demand wakeup alone cannot satisfy the delay requirements without using regular wakeup.

In addition, the expected time interval between two on-demand wakeups of client i, denoted by τ_i, consists of two parts. One is the expected time of the first packet arrival after one on-demand wakeup, which is $1/\lambda_i$. The other is the expected time between the first packet arrival and the next on-demand wakeup of client, which can be computed as follows. If the client is woken up before the deadline (with the probability of θ_i), the client has to wait for at most d_i before its wakeup; otherwise (with the probability of $1 - \theta_i$), based on the memoryless property of Geometric distribution, the waiting time can be computed as $d_i + mW/p_i$, where $1/p_i$ is the expected number of attempts to wake up the client after deadline. Hence,

$$\tau_i = 1/\lambda_i + \left[\theta_i d_i + (1-\theta_i)(d_i + mW/p_i)\right], \tag{3.7}$$

For any LI, it holds that

$$y_i B - (d_i - B) \leq \tau_i x_i \leq y_i B - B. \tag{3.8}$$

3.3.4 Energy Consumption Analysis

3.3.4.1 WiFi Energy Consumption

For a given BI j, suppose there are n_j clients woken up to retrieve packets. Then, n_j can be computed as $\sum_{\forall i \in Z} \frac{x_i+1}{y_i}$. The overall energy consumption of all clients consists of three parts as follows.

- $n_j \cdot E_0$: the total energy consumed by all clients for receiving beacon frames and sending PS-POLLs, where E_0 is the energy for receiving a beacon frame and sending a PS-POLL. Specif-

ically, let T_B and T_{POLL} denote the durations for receiving a beacon frame and sending a PS-POLL, respectively. Then, $E_0 = T_B \cdot P_{rx} + T_{POLL} \cdot P_{tx} + SIFS \cdot P_{idle}$, where P_{tx}, P_{rx} and P_{idle} are the rates of energy consumption (in Watt) for WiFi interface to transmit, receive and stay idle listening, respectively.

- $l_j \cdot E_D$: the total energy consumed by all clients for receiving data packets from the AP, where E_D is the energy consumption for receiving a data packet and l_j is the total number of packets to be retrieved during BI j. Specifically, let T_D and T_{ACK} denote the durations for receiving a data packet and sending an ACK, respectively. Then, $E_D = T_D \cdot P_{rx} + T_{ACK} \cdot P_{tx} + (DIFS + SIFS)P_{idle}$.

- Let E_{idle} denote the energy consumed by a client in idle listening while other client is retrieving *one* data packet. Then, $E_{idle} = (T_D + T_{ACK} + DIFS + SIFS)P_{idle}$. Then, the overall energy consumed for idle listening is computed as follows.

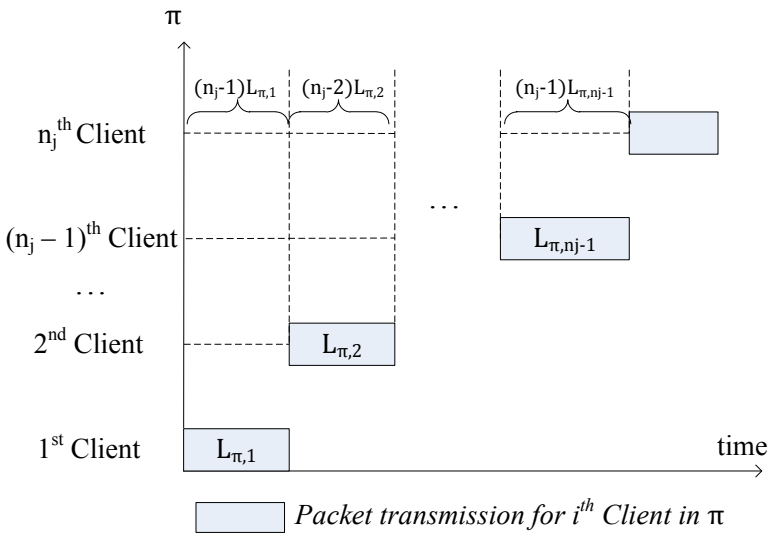


Figure 3.4: Idle listening caused by contentions among clients

Upon receiving beacon frame that indicates data packets buffered at the AP, each client sends a PS-POLL to compete for the channel. The total time duration for all clients to send PS-POLL is $n_j E_{POLL}$, where $E_{POLL} = T_{POLL} P_{idle}$. After that, the client that wins the competition starts to retrieve all its data packets from the AP, while others are idly listening to the channel during the transmission.

This process continues until all clients finish their transmission. Generally, there are $n_j!$ possible transmission sequences, as illustrated in Fig. 3.4. Let Π denote such set of sequences, where each sequence $\pi \in \Pi$ occurs with the probability of $1/n_j!$. Then, the total energy consumption of all clients for idly listening data packet transmission is denoted by X, whose expectation can be computed as

$$\begin{aligned} E[X] &= \sum_{\pi \in \Pi} \frac{1}{n_j!}(E[L_\pi]E_{idle}) \\ &= \frac{E_{idle}}{n_j!} \sum_{\pi \in \Pi} [(n_j - 1)L_{\pi,1} + (n_j - 2)L_{\pi,2} + \cdots + L_{\pi,n_j-1}]. \end{aligned}$$

Since $\sum_{\pi \in \Pi} L_{\pi,1} = \cdots = \sum_{\pi \in \Pi} L_{\pi,n_j-1} = l_j(n_j - 1)!$

$$\begin{aligned} E[X] &= \frac{E_{idle}}{n_j!}[l_j(n_j - 1)(n_j - 1)! + l_j(n_j - 2)(n_j - 1)! + \cdots + l_j(n_j - 1)!] \\ &= \frac{(n_j - 1)!}{n_j!} \cdot l_j \cdot \frac{n_j(n_j - 1)}{2} \cdot E_{idle} \\ &= \frac{(n_j - 1)l_j}{2} \cdot E_{idle}. \end{aligned}$$

Here, $L_{\pi,i}$ is the total time that the i^{th} client in π spends in retrieving data packets from the AP, and L_π is the total time that all clients spend in idle listening by following sequence π.

Hence, the overall WiFi energy consumption, denoted by E_{wifi}, for BI j, is given by

$$\begin{aligned} E_{wifi} &= n_j \cdot E_0 + l_j \cdot E_D + (n_j E_{POLL} + E[X]) \\ &= (E_0 + E_{POLL} + \frac{E_{idle} l_j}{2})n_j + (E_D - \frac{E_{idle}}{2})l_j, \end{aligned} \tag{3.9}$$

where $l_j = B \cdot \sum_{\forall i \in Z} \lambda_i$, which is constant.

3.3.4.2 ZigBee Energy Consumption

In addition to WiFi energy consumption, each client also consumes energy for receiving ZigBee wakeup frames. To save energy, once a client receives a wakeup frame, it can completely turn off its ZigBee radio until it has retrieved the data packets from its AP. After the completion of packet transmission, it wakes up for a short duration at the beginning of each wakeup slot to sense the channel. If the channel is idle, indicating no wakeup frame is to be received, then the ZigBee radio goes to sleep. Note that, as to be presented in the following section, m may change over time as the packet rate and ZigBee link quality may vary over time. Thus, clients do not have exact knowledge of m used by the AP for broadcasting wakeup frame and thereby need to sense the channel for possible incoming wakeup frames.

For any given LI, the expected number of wakeup frames received by client i within a BI is $\frac{x_i/p_i}{y_i}$ and thereby the corresponding number of channel sensing is $m \cdot \frac{x_i/p_i}{y_i}$. Hence, the overall ZigBee energy consumption in a BI, denoted by E_{zigbee}, can be computed as

$$\begin{aligned} E_{zigbee} &= E_{wakeup} \cdot \sum_{\forall i \in Z} \frac{x_i}{p_i y_i} + E_{sense} \cdot \sum_{\forall i \in Z} \frac{m x_i}{p_i y_i} \\ &= (E_{wakeup} + m \cdot E_{sense}) \cdot \sum_{\forall i \in Z} \frac{x_i}{p_i y_i}, \end{aligned} \tag{3.10}$$

where E_{wakeup} and E_{sense} denote the energy consumed for receiving and sensing a ZigBee wakeup frame, respectively.

3.3.4.3 Summary

Minimizing the overall energy consumption (i.e., $E_{wifi} + E_{zigbee}$) is equivalent to minimizing

$$\begin{aligned} E_{overall} &= \left(E_0 + E_{POLL} + \frac{E_{idle} B \sum_{i=1}^{n} \lambda_i}{2} \right) \cdot \sum_{\forall i \in Z} \frac{x_i + 1}{y_i} \\ &+ (E_{wakeup} + m \cdot E_{sense}) \cdot \sum_{\forall i \in Z} \frac{x_i}{p_i y_i} \end{aligned} \tag{3.11}$$

where $x_i, y_i \ \forall i \in Z$ and m are unknown.

3.3.5 Optimization Problem

Based on the above analysis, we can get the following optimization problem. Here, $\Omega = \{m\} \cup \{(x_i, y_i) \mid \forall i \in Z\}$ represents wakeup schedule for the system.

Objective: Find Ω to minimize $E_{overall}$

s.t.,

$$\text{if } \delta_i > \theta_i, \ \ y_i \leq \frac{(1-\theta_i)(d_i - B)}{B(\delta_i - \theta_i)}, \ \forall i \in Z \tag{3.12}$$

$$y_i B - (d_i - B) \leq \tau_i x_i \leq y_i B - B, \ \forall i \in Z \tag{3.13}$$

$$0 \leq x_i \leq y_i - 1, \ \forall i \in Z \tag{3.14}$$

$$\frac{d_i - B}{B} \leq y_i \leq Y_{max}, \ \forall i \in Z \tag{3.15}$$

$$1 \leq m \leq \max \left\{ \frac{d_i - B}{W} \right\}_{\forall i \in Z} \tag{3.16}$$

Obviously, this problem is non-linear and hard to be solved. However, a numerical method can be adopted to solve the problem after we transform the problem. Specifically, we first transform the above problem to an *equivalent* problem by letting $x'_i = x_i / y_i$ and $y'_i = 1/y_i$. Then, based on the observation

that m can be any integer within the range of $\{1, 2, \cdots, \max\left\{\lfloor\frac{d_i - B}{W}\rfloor\right\}_{\forall i \in Z}\}$, which is a small set in practice, we can solve the transformed problem for all possible m efficiently and get the solution that yields the smallest objective value as the *optimal* solution to the original problem.

3.4 Design

From the above analysis, we can see that appropriate scheduling of regular and on-demand wake-ups is the key to minimize energy consumption while satisfy delay requirements. To achieve this, we present a practical scheme that dynamically adjusts the regular and on-demand wakeups of WiFi interfaces.

3.4.1 Overview

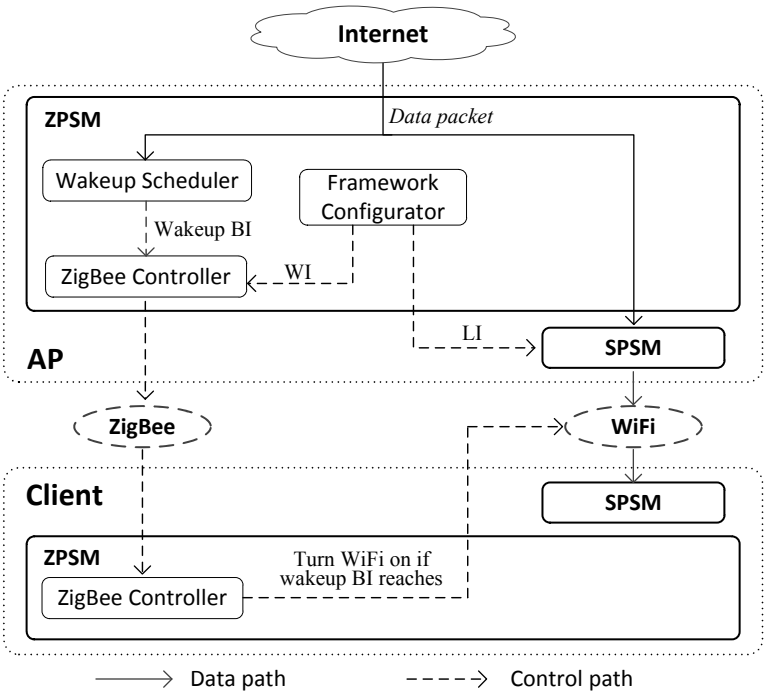


Figure 3.5: ZPSM architecture

Fig. 3.5 shows the architecture of the ZPSM system, which consists of three components. The *framework configurator* component periodically configures LI and WI for each client and the AP (called

wakeup framework), respectively. Under the configured wakeup framework, the delay requirements can be satisfied by adopting our proposed wakeup strategies. Based on this framework, the *wakeup scheduler* component dynamically schedules on-demand wakeup (called *wakeup dynamics*) for minimizing energy consumption, if a client cannot meet the delay bound of its incoming data packets through regular wakeup. Finally, the *ZigBee controller* component, implemented on both AP and client sides, is responsible for exchanging control messages and waking up client at scheduled BIs.

In this section, we first focus on LD clients and present how to set up the wakeup framework to meet their delay requirements, and then propose a scheduling algorithm to deal with wakeup dynamics to minimize energy consumption. After that, we extend our design to support SD clients. Finally, we briefly describe the protocol for membership management in the ZPSM network.

3.4.2 Wakeup Framework

An optimal solution to the scheduling problem defined in Section 3.3.5 consists of three parts: m for the AP, y_i and x_i for each client i. m and y_i explicitly determine the *periodic* wakeup behaviors of the AP's ZigBee interface and the client's WiFi interfaces, respectively. Thus, they together constitute the *wakeup framework* of ZPSM. Besides, x_i characterizes the behavior of on-demand wakeup for client i, which can be changed frequently to adapt to dynamic traffic and link quality. This is called *wakeup dynamics* presented in the next subsection.

In the proposed system, the AP periodically solves the optimization problem with inputs λ_i and p_i, which are measured online. The resulting optimal solution is used to determines the values of m and y_i for each client i. To reduce computational overhead, the wakeup framework dose not change over frequently and the updated interval is pre-determined (e.g., 10 seconds). Moreover, since our system directly adopts the configurations (i.e., m and y_i) from optimal solution and the delay requirements given by Eq. (3.2) only depends on m and y_i, the delay requirements can be satisfied under our proposed wakeup framework.

3.4.3 Wakeup Dynamics

The wakeup framework only defines the time points at which the AP is allowed to wake up clients on demand. In this subsection, we present the scheme to determine when each client should wake up to minimize energy consumption.

3.4.3.1 Insights

Consider any k consecutive BIs in our ZPSM system. The total number of incoming packets during k BIs, denoted by L, can be computed as $kB \cdot \sum_{\forall i \in Z} \lambda_i$. Let n be a random variable representing the total number of wakeups of all clients during these k BIs, n_j be the total number of awake clients and l_j be the total number of data packets transmitted at the j^{th} BI ($j = 1, \cdots, k$). Then, by Eq. (3.9), the total WiFi energy consumption E of these BIs can be computed as

$$E = \sum_{j=1}^{k} \left[(E_0 + E_{POLL} + \frac{E_{idle} l_j}{2}) n_j + (E_D - \frac{E_{idle}}{2}) l_j \right]. \tag{3.17}$$

Ignoring the constant term, i.e., $(E_D - \frac{E_{idle}}{2}) \sum_{j=1}^{k} l_j = (E_D - \frac{E_{idle}}{2}) L$, we can get the following scheduling problem as follows.

Objective: minimize energy consumption

$$E = (E_0 + E_{POLL}) n + \frac{E_{idle}}{2} \sum_{j=1}^{k} n_j l_j \tag{3.18}$$

s.t.,

$$\sum_{j=1}^{k} n_j = n \quad \text{and} \quad \sum_{j=1}^{k} l_j = L$$

Treating n as a constant and applying Lagrange Multiplier, we can get that E is minimized if $n_1 l_1 = n_2 l_2 = \cdots = n_k l_k = \frac{n}{k} \cdot \frac{L}{k} = \frac{nL}{k^2}$. Thus, $E = n(E_0 + E_{POLL} + \frac{L \cdot E_{idle}}{2k})$, which is a function of n and it is minimized if the following two conditions are satisfied.

- **Condition I**: the *wakeup frequency* is minimized.
- **Condition II**: the product of the number of awake clients and transmitted packets (i.e., $n_j l_j$, called *transmission workload*) is balanced among different BIs.

Note: Although the above analysis only deals with minimizing WiFi energy consumption, the goal of minimizing ZigBee energy consumption can also be accomplished. This is because, given m and y_i, minimizing ZigBee energy consumption is equivalent to minimizing x_i by Eq. (3.10), namely, minimizing wakeup frequency (Condition I).

3.4.3.2 Representation of Wakeup Schedule

The scheduling algorithm is run by the AP at the beginning of each WI. Let $a_{first,i}$ denote the first packet arrival for client i since its last regular wakeup. If the next regular wakeup time of client i, say time t_{wakeup}, is one BI before the packet's delay bound, all packets of client i that arrive between $a_{first,i}$ and t_{wakeup} can be delivered before their deadline as well. Otherwise, client i needs to be woken up on demand, and the corresponding bits in wakeup frame are set immediately to ensure the delay requirements. Besides, to minimize energy consumption, the scheduling algorithm is used to determine a BI at which a client should be woken up, which is also indicated in wakeup frames. The set of candidate BIs for client i includes all *whole* BIs contained between $a_{first,i}$ and t_{wakeup}. Following time order, the j^{th} BI in the set is denoted by $b_{i,j}$ for $j = 1, \cdots, B_{max,i}$. Fig. 3.6 shows an example, the set of candidate BIs are $b_{i,1}$, $b_{i,2}$ and $b_{i,3}$, and $B_{max,i} = 3$. Note that wakeup frame may be received by client i after the scheduled BI, in which case client i wakes up at the next immediate BI to retrieve packets from the AP.

To wake up a client, the AP should send a wakeup frame through its ZigBee interface. In the wakeup frame, the AP allocates a certain number of bits to represent the scheduled BI at which a client should wake up. Due to limited bandwidth of the ZigBee channel, the size of wakeup frame should not be too large. In our research, we use default ZigBee data payload of 29 bytes to contain a wakeup frame. Particularly, if the maximum possible BI index ($B_{max,i}$) for client i exceeds the largest number K that can be represented by the allocated bits, then each number between 1 and K represents $\lceil B_{max,i}/K \rceil$ consecutive BIs.

Moreover, to locate the corresponding bits in a wakeup frame without consuming extra bits, each client is assigned a unique index i, where $i = 1, \cdots, n$ and n is the total number of ZPSM clients, when it joins the ZPSM network. The index serves as an offset into wakeup frame to locate the corresponding

bits for a client. The details on the index maintenance in the ZPSM network will be presented in Section 3.4.5.

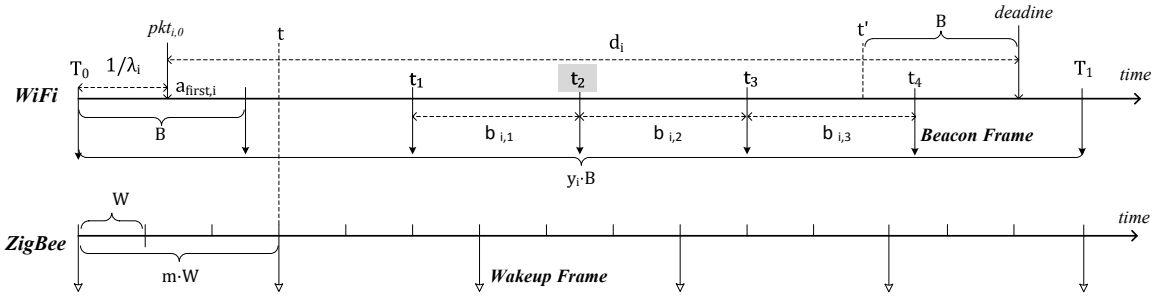


Figure 3.6: Example of AP's behaviors for waking up client i on demand. In this example, data packet $pkt_{i,0}$ for client i arrives the AP at $a_{earliest,i}$ and it is the earliest packet arrival for client i since its last regular wakeup at T_0. Here, the LI of client i is y_iB, where $y_i = 6$, and the WI of the AP is mW, where $m = 3$. As $a_{earliest,i} + d_i - B = t' < T_1$, where T_1 is the next regular wakeup time, client i needs to be woken up on-demand in order to retrieve $pkt_{i,0}$ before deadline. Note that if client i is woken up at t_4, it cannot be guaranteed that the packet can be retrieved by the client before the deadline, as packet transmission may occur at any time during the BI between t_4 and T_1. Only the packet transmissions that happen during b_1, b_2 and b_3 can ensure the bounded delay. Thus, the number of the chances to meet the delay bound is 3, i.e., $B_{max,i} = 3$.

3.4.3.3 Scheduling Algorithm

Intuitively, to minimize average wakeup frequency (Condition I), each client should be woken up as late as possible, i.e., $b_{i,B_{max,i}}$. However, simply adopting this wakeup strategy for every client may result in long idle listening if multiple clients are woken up at the same BI and all have many packets to retrieve. To address this problem, the idea of transmission workload balancing (Condition II) can be used to stagger the on-demand wakeups of different clients so as to improve energy efficiency. Based on these observations, we propose a scheduling algorithm executed by the AP at the beginning of each WI, say time t. The key idea is to balance transmission workload among different clients by incurring the minimal number of wakeups.

The algorithm considers the wakeup BIs of clients one by one. Suppose client i is considered. Let $\Delta_{i,j}$ and $P_{i,j}$ denote the increment of transmission workload and the probability if client i is woken up at $b_{i,j}$, respectively. Specifically, $\Delta_{i,j}$ can be derived by synthesizing current traffic (λ_i), the expected number of on-demand wakeups in a LI (x_i obtained from the optimal solution) and

the already determined wakeup schedules (y_i and on-demand wakeup BIs of scheduled clients); $P_{i,j}$ can be computed as $P_{i,j} = 1 - (1 - p_i)^{\frac{(j-1)B+(t_1-t)}{W}}$, where t_1 is the time that the first candidate BI (i.e., $b_{i,1}$) begins. As mentioned before, if client i fails to receive the wakeup frame before $b_{i,j}$, it should wake up at next immediate BI $b_{i,k}$, where $k = j+1, \cdots, B_{max,i}$, upon receiving the wakeup frame. Thus, the probability $Q_{i,k}$ that client i is woken up at $b_{i,k}$ can be computed as $Q_{i,k} = (1-p_i)^{\frac{(k-2)B+(t_1-t)}{W}} \cdot [1-(1-p_i)^{\frac{B}{W}}]$, which is the product of the probability that the transmission of wakeup frame fails before $b_{i,k}$ and the probability that the transmission succeeds during $b_{i,k}$. For example, as illustrated in Fig. 3.6, where $B_{max,i} = 3$. If client i is scheduled to b_2 (starting from t_2), then we have N-TWIN$(i,2) = (P_{i,2} \cdot \Delta_{i,2})/2 + (Q_{i,3} \cdot \Delta_{i,3})/3$.

Alg. 1 describes our proposed scheduling algorithm. Imitating the greedy approach for solving *partition problem*, it goes through the clients in descending order with respect to their resulting **N**ormalized **T**ransmission **W**orkload **IN**crement (N-TWIN) and schedules each of them to whichever BI that has the smallest transmission workload. Particularly, if a BI has been just assigned to a client that has lots of packets to retrieve, then the likelihood that the BI will be allocated to another client becomes low.

Formally, the N-TWIN of client i if it is scheduled to wake up at $b_{i,j}$, denoted by N-TWIN(i,j), is defined as

$$\text{N-TWIN}(i,j) = \frac{P_{i,j} \cdot \Delta_{i,j}}{j} + \sum_{k=j+1}^{B_{max,i}} \frac{Q_{i,k} \cdot \Delta_{i,k}}{k}, \tag{3.19}$$

which is the expected sum of all possible transmission workload increments (from $b_{i,j}$ till $b_{i,B_{max,i}}$) normalized to their corresponding BI index.

Algorithm 1 Scheduling for on-demand wakeup

1: Let Z_{sched} be the set of clients to be scheduled
2: **while** $Z_{sched} \neq \emptyset$ **do**
3: Find for each client $i \in Z_{sched}$ a b_{i,k_i}, where $k_i = 1, \cdots, B_{max,i}$, s.t., waking up at b_{i,k_i} results in the **smallest** N-TWIN, denoted by N-TWIN$_{min,i}$
4: Schedule client z with N-TWIN$_{min,z}$, where N-TWIN$_{min,z}$ is the **largest** N-TWIN$_{min,i}$ among all $i \in Z_{sched}$, at b_{z,k_z}
5: $Z_{sched} = Z_{sched} \setminus \{z\}$
6: **end while**

The rationales behind the using of N-TWIN as the greedy metric are as follows: (1) Due to varying

link quality of the ZigBee channel, on-demand wakeup may fail and the client may not be able to wake up at its scheduled BI. Hence, both the success and failure cases should be taken into account. (2) By normalizing transmission workload to BI index, it is fair to combine the transmission workload increments of different BIs together, because a client may have more data packets to retrieve and thereby incur more transmission workload if it is scheduled to wake up at a later BI (with a larger index) than an earlier BI (with a smaller index). (3) As N-TWIN(i, j) is the sum of all possible transmission workload increments starting from $b_{i,j}$ till $b_{i,B_{max,i}}$, the candidate BI with a large index is more likely to result in a small N-TWIN and thereby to be chosen, which complies with the goal of reducing wakeup frequency.

Briefly, the algorithm first finds for each client a BI that has the smallest N-TWIN at each round in line 3, with the purpose of minimizing wakeup frequency. Then, it schedules clients in the descending order with respect to their smallest N-TWIN round by round in line 4, with the purpose of balancing transmission workload. In addition, the computation of N-TWIN at the current round is based on the transmission workload of the clients scheduled in previous rounds. Therefore, it is less likely to schedule a client to most recently scheduled BIs, since this may result in a larger increment on transmission workload due to the non-linear increase of transmission workload according to the definition.

3.4.3.4 Estimation of Link Quality and Traffic Rate

As mentioned in Section 3.4.2, the global parameter WI (or m) is updated every a certain long period of time, called *update interval* (UI). The AP notifies each client of the change by piggybacking current WI in wakeup frame. With the knowledge of current WI, each client is able to know when it should receive wakeup frames. Therefore, it can estimate the ZigBee link quality between the AP and itself as follow. Once a client knows the updated WI after the beginning of UI i, it records the expected number of receptions of wakeup frames r_i and the actual number of receptions of wakeup frames r'_i until the end of UI i. Then, r'_i/r_i gives one estimation of the current link quality between the AP and that client.

In our system, each client always maintains N (e.g., 50) most recently estimates (i.e., $\hat{p_i}$ for $i = 0, \cdots, N-1$), which are synthesized through *exponential moving average* to derive actual link quality p_i. Particularly, let P_i be the estimated link quality by synthesizing the first i estimates. P_i is computed

as

$$P_i = \alpha_i \cdot \hat{p}_i + (1 - \alpha_i) \cdot P_{i-1}, \tag{3.20}$$

where $P_0 = \hat{p_0}$ and α_i is the smoothing factor between 0 and 1, defined as

$$\alpha_i = 1 - \exp\left(-\frac{r_i}{\sum_{i=0}^{N-1} r_i}\right). \tag{3.21}$$

With moving averaging, the impact of time elapsing and sample size are considered by giving a bigger weight to more recent estimates with larger sample size while a smaller weight to older ones with smaller sample size. Finally, the estimated link quality p_i is quantified as $p_i = P_{N-1}$, which is reported to the AP through ZigBee communication.

Similarly, traffic rate (i.e., packet arrival interval $1/\lambda_i$) of each client i can be also estimated through moving averaging certain number of most recently packet arrivals.

3.4.4 Extension for SD Clients

To improve energy efficiency of SD clients, we propose an adaptive ZPSM to dynamically switch SD clients between ZPSM and CAM toward minimizing proportion of time that they stay in CAM. Specifically, the AP broadcasts wakeup frame *every* WI. During the wakeup slots that are not scheduled for wakeup frame transmission for LD clients according to the above design, the AP delays wakeup frame transmission for a while (i.e,. the time to sense a wakeup frame). Hence, LD clients go to sleep and avoid receiving unnecessary wakeup frames to save energy consumed by ZigBee. In these wakeup frames dedicated to waking up SD clients, only one bit is used to indicate whether there is a buffered packet for a SD client. If the client successfully receives a wakeup frame and the corresponding bit is set, the client turns on its WiFi interface and sends a PS-POLL to retrieve the buffered packet from the AP immediately.

Since ZigBee link quality may change over time, using only on-demand wakeup may not satisfy the delay requirements. Thus, regular wakeup should also be used. Formally, in a LI (i.e., $y_i B$) of SD client i, if a packet arrives at the AP no earlier than d_i time before the end of the LI, then the deadline can be met through the follow-up regular wakeup (assume the channel is in good condition and not

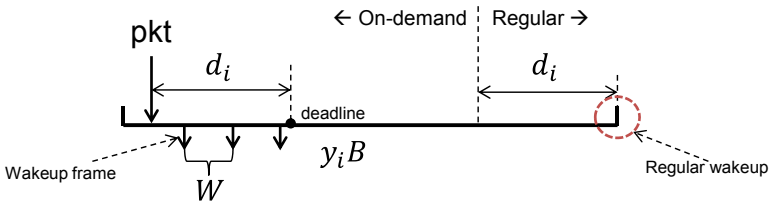


Figure 3.7: Analysis of SD client i's delay requirements

busy). Otherwise, on-demand wakeup is performed to deliver the packet. The probability that this case occurs is $1-\frac{d_i}{y_iB}$. As mentioned before, on-demand wakeup is conducted through wakeup frame which is broadcast the AP repeatedly through ZigBee interface before the deadline. If client i fails to receive all these wakeup frames, the on-demand wakeup fails to meet the deadline. The probability that this occurs is $(1-p_i)^{\lfloor\frac{d_i}{W}\rfloor}$. Sine the probability that a packet fails to meet its deadline should be smaller than $1-\delta_i$, we can get

$$\left(1-\frac{d_i}{y_iB}\right)(1-p_i)^{\lfloor\frac{d_i}{W}\rfloor} \leq 1-\delta_i. \tag{3.22}$$

According to currently estimated ZigBee link quality, SD client i computes and uses the maximum value of y_i using Eq. (3.22). Since there may exist multiple clients in the network, client i may need to wait for a while before it is served by the AP, which may incur some additional delay for packet transmission. Thus, the derived LI may not be able to satisfy delay requirements. To deal with this problem, each SD client records most recently received R (e.g., 50 in our simulation and experiments) data packets, based on which current delay-meet ratio $\hat{\delta}_i$ can be estimated. If the estimated delay-meet ratio is lower than δ_i, y_i is decreased to $max\{1, \lfloor y_i/2 \rfloor\}$. When $y_i = 1$ and the resulting delay-meet ratio is still below δ_i, client i switches to CAM. After a while, if ZigBee link quality becomes better such that the resulting LI is no less than 2 according to Eq. (3.22) (to avoid switching between ZPSM and CAM frequently), client i switches back to ZPSM.

3.4.5 Membership Management

Due to mobility, each client needs to broadcast beacon message through its ZigBee interface so as to announce its presence; a ZPSM client may move out the ZigBee range of its AP and becomes

a normal SPSM client; a SPSM client may discover an AP in its ZigBee range and join the ZPSM network maintained by that AP.

3.4.5.1 Beacon Exchanging

To announce its presence and report other scheduling related information to the AP, each client periodically (e.g., 1 s in our simulation and experiments) broadcasts beacon messages through it ZigBee interface. The beacon message contains: BSSID of the network that the client belonging to, estimated link quality and unique index assigned by the AP for locating scheduling information in wakeup frame. With client's index periodically reported, the AP can ensure the consistency of indices among all clients. For a client, the wakeup frames broadcast by the AP serve as beacon messages from the AP. Thus, both the AP and clients can have enough knowledge about the ZigBee network topology, which facilitates clients' joining and leaving.

3.4.5.2 Client Joining

When a SPSM client overhears a wakeup frame from the AP, it sends an *association request* packet, containing the identity of itself. Upon receiving the association request packet, the AP first checks if the client is already associated with the WiFi network (i.e., wireless LAN) managed by itself. If so, the AP accepts the association request and sends an *association response* packet containing the unique ID of the network (i.e., BSSID) as well as a unique index (typically the smallest unused index in the network) assigned to the client. Once the index packet is successfully received by the client, it switches to ZPSM mode. The AP can be informed about this from the beacon messages (containing BSSID) sent by that client. Note that since there may exist multiple APs nearby, a client needs to first join a WiFi network following the standard 802.11 procedure before it is allowed to join the ZigBee network managed by the same AP.

3.4.5.3 Client Leaving

When a ZPSM client with index i finds it has moved out of the ZigBee range of its AP, i.e., failing to receive wakeup frames for a certain period of time (e.g., a whole UI), it can default to either SPSM

or CAM, depending on the delay requirements. On the other hand, if the AP cannot receive the beacon message from client i for a certain period of time, it automatically de-associates with client i and assigns the index i to the client with the largest index in the network by sending a *re-association* packet. The goal is to keep continuity of indices in the network, which can ensure the efficiency of utilizing the bits of wakeup frame. Meanwhile, since it is likely that the AP cannot receive beacon messages from the client but the client can receive wakeup frames from the AP, the AP sends a *ZPSM de-association* packet to the client through its WiFi interface so as to make sure that the client is aware of this and switches to SPSM or CAM accordingly.

In addition, to avoid interference with wakeup frames, all ZigBee control packets (i.e., association request, association response and re-association packets) and beacon messages are always sent during the gap between any two consecutive wakeup frames.

3.5 Simulation

To evaluate the proposed system, we conduct extensive simulation based on ns2. In the simulation, we measure two performance metrics:

- *Per-packet energy consumption* (mJ/pkt) is defined as the total energy consumption (including both WiFi and ZigBee) divided by the total number of data packets transmitted. The energy consumed by the WiFi and ZigBee interfaces is measured according to the specified power consumption rate of SX-SDWAG 802.11g wireless module [43] and CC2530 RF transceiver [44], respectively, as shown in Table 3.1.

- *Actual delay-meet ratio* is defined as the total number of data packets that meet their delay requirements (at the client side) divided by the total number of data packets transmitted. The packet arrival is modeled as Poisson process [35,41,42]. WiFi transmission may fail due to interference or collision.

In the simulation, we adopt the random waypoint mobility model, where the pause time is fixed to 10s and the moving speed is randomly chosen from 0 to 3m/s. The standard IEEE 802.11g is used for WiFi transmission. The detailed settings for our simulation are shown in Table. 3.1. To facilitate

Network size	20 clients	Packets collected	200,000
WiFi range	120 m	ZigBee range	100 m
Wakeup slot (W)	40 ms	Update interval (UI)	10 s
MAC header	34 bytes	WiFi channel bit rate	54 Mbps
PHY header	17 bytes	WiFi basic bit rate	1 Mbps
Beacon packet	28 bytes	SIFS	16 μs
PS-POLL	20 bytes	DIFS	34 μs
ACK	14 bytes	Beacon interval (B)	100 ms
Data packet	2312 bytes	ZigBee channel bit rate	250 Kbps
WiFi transmission	1.152 Watt	ZigBee transmission	0.087 Watt
WiFi reception	0.561 Watt	ZigBee reception	0.072 Watt
WiFi idle listening	0.462 Watt	ZigBee idle listening	0.019 Watt

Table 3.1: Simulation settings

our evaluation, network-wide average packet arrival rate λ, average delay bound d, average link quality p and average required delay-meet ratio δ are specified. Each client is randomly configured with a packet arrival rate, a required delay bound, a ZigBee link quality and a required delay-meet ratio within $(0.5\lambda, 1.5\lambda)$, $(0.5d, 1.5d)$, $(p - 0.1, p + 0.1)$ and $(\delta - 0.05, \delta + 0.05)$, respectively. Besides downlink traffic, uplink traffic is also generated at each client and the average rate is 0.1λ. In the following, we first study the performance of our proposed ZPSM system; then, we compare it with other schemes.

3.5.1 Performance Study of ZPSM

3.5.1.1 Overall Performance in Different Scenarios

Fig. 3.8a shows that the energy consumption decreases as the required delay bound increases. This is because, as the delay bound increases, clients wake up less frequently, which reduces the wakeup overheads for turning on/off WiFi, receiving beacon and sending/idly listening PS-POLL (i.e., $E_0 + E_{POLL}$). Moreover, due to balanced transmission workload, the contention among clients does not degrade energy efficiency, even when clients have many buffered packets to retrieve at each wakeup. We can also observe that, as packet rate goes up, the energy consumption drops. The reasons are as follows: the wakeup overheads (or wakeup frequency) remain almost the same since it mainly depends on delay bound; when packet rate increases, the per-packet energy consumption naturally becomes smaller as the wakeup overheads are averaged over more packets. Particularly, when the average delay

bound drops below 0.4s, some clients are SD clients requiring a delay bound shorter than two BIs. Consequently, the overall energy consumption increases rapidly since SD clients may enter CAM from time to time.

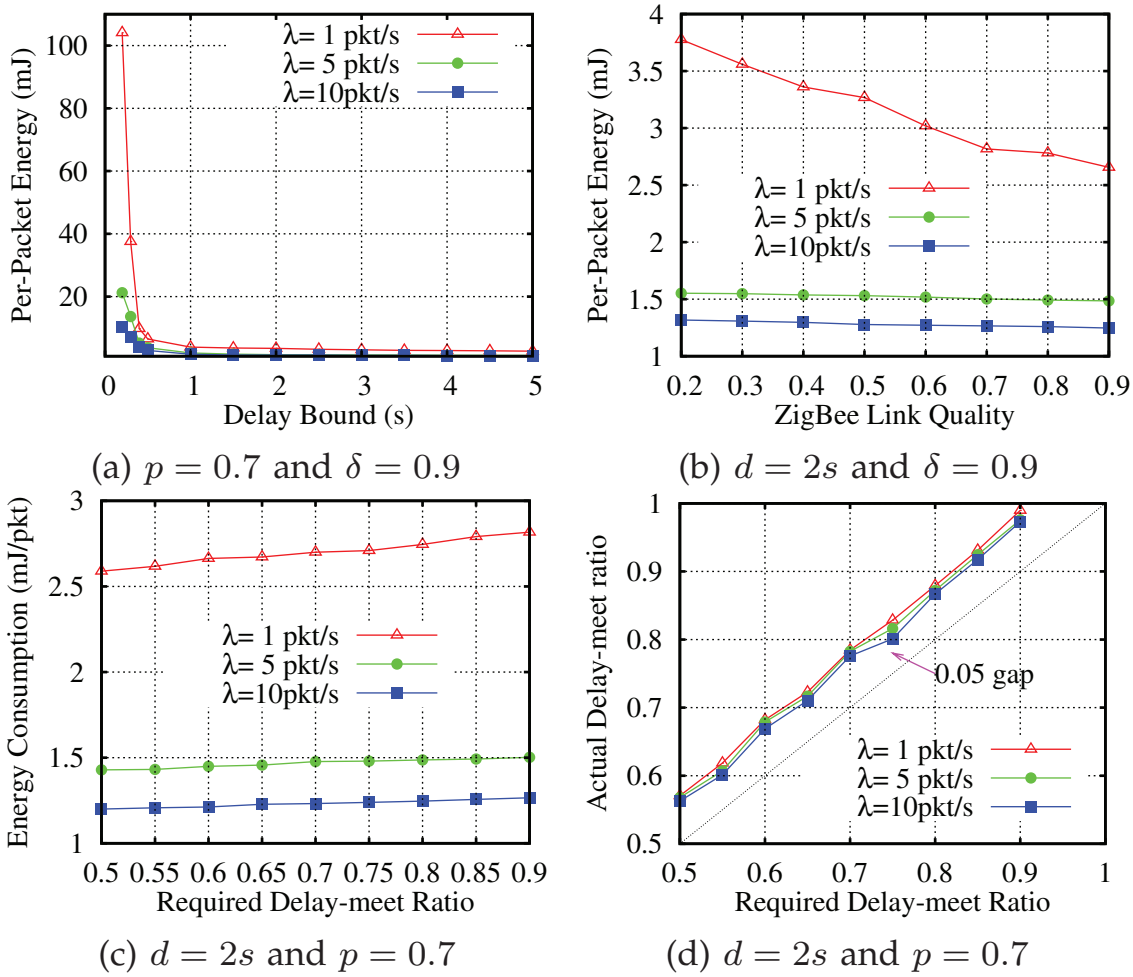


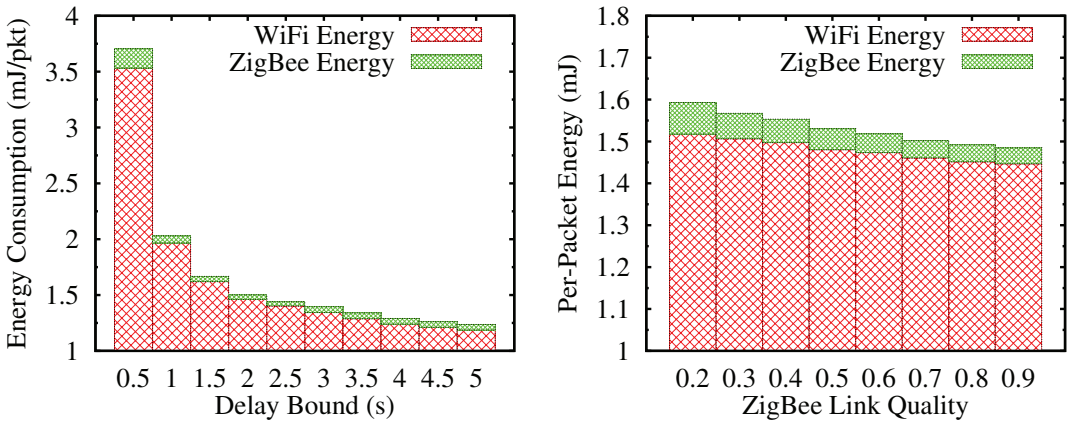
Figure 3.8: Performance of ZPSM in different scenarios

Fig. 3.8b shows that the energy consumption decreases as ZigBee link quality increases. This is because, as link quality increases, clients can have more chances to utilize on-demand wakeup to reduce the wakeup frequency of the WiFi interface and balance transmission workload, which can result in higher energy efficiency. Besides, better link quality can also reduce ZigBee communication overheads for retransmission. We can also observe that, when packet rate is low, this trend is more evident as the saved energy is averaged over few packets. Besides, all clients have an actual delay-meet

ratio above 0.95. Due to space limitation, we do not show the results.

Fig. 3.8c and 3.8d show the energy consumption and actual delay-meet ratio as the required delay-meet ratio increases, respectively. Fig. 3.8c shows that the energy consumption slightly increases as the required delay-meet ratio gets higher. By Eq. (3.6), we can see that as the required delay-meet ratio increases the LI becomes smaller in order to ensure delay requirements. As a result, wakeup overheads get higher, leading to higher energy consumption. In addition, Fig. 3.8d shows that our proposed ZPSM can always achieve an actual delay-meet ratio that is about 0.05 higher than the required one. This is because our design always presumes the worst case, that is, packet transmission is always finished at the end of BI; however, through balancing transmission workload, packet transmission may be finished earlier. Therefore, the actual waiting time can be shorter.

3.5.1.2 Energy Consumption – WiFi *v.s.* ZigBee



(a) $p = 0.7$ (b) $d = 2s$

Figure 3.9: WiFi *v.s.* ZigBee in ZPSM ($\lambda = 5pkt/s$ and $\delta = 0.9$)

Fig. 3.9 plots the portion of energy that WiFi and ZigBee interfaces consume. Generally, in our simulated scenarios, the WiFi interface consumes the energy that is 20$\sim$38 times more than the ZigBee interface does. In Fig. 3.9a, as delay bound becomes smaller, both WiFi and ZigBee interfaces consume more energy due to increased wakeup overheads. In Fig. 3.9b, as link quality gets better, the energy consumption of both interfaces decreases.

3.5.2 Performance Comparison

Moreover, we compare our proposed ZPSM against SPSM, optimal solution (OPT) as well as a simplified ZPSM (S-ZPSM), which uses the same wakeup framework as ZPSM except that each client is always scheduled to wake up at the latest BI (i.e., it reduces wakeup frequency without balancing transmission workload). To distinguish from S-ZPSM, our proposed ZPSM is labeled as advanced ZPSM (A-ZPSM). Particularly, if SPSM client has a delay bound smaller than two BIs, it runs CAM in order to ensure delay requirements.

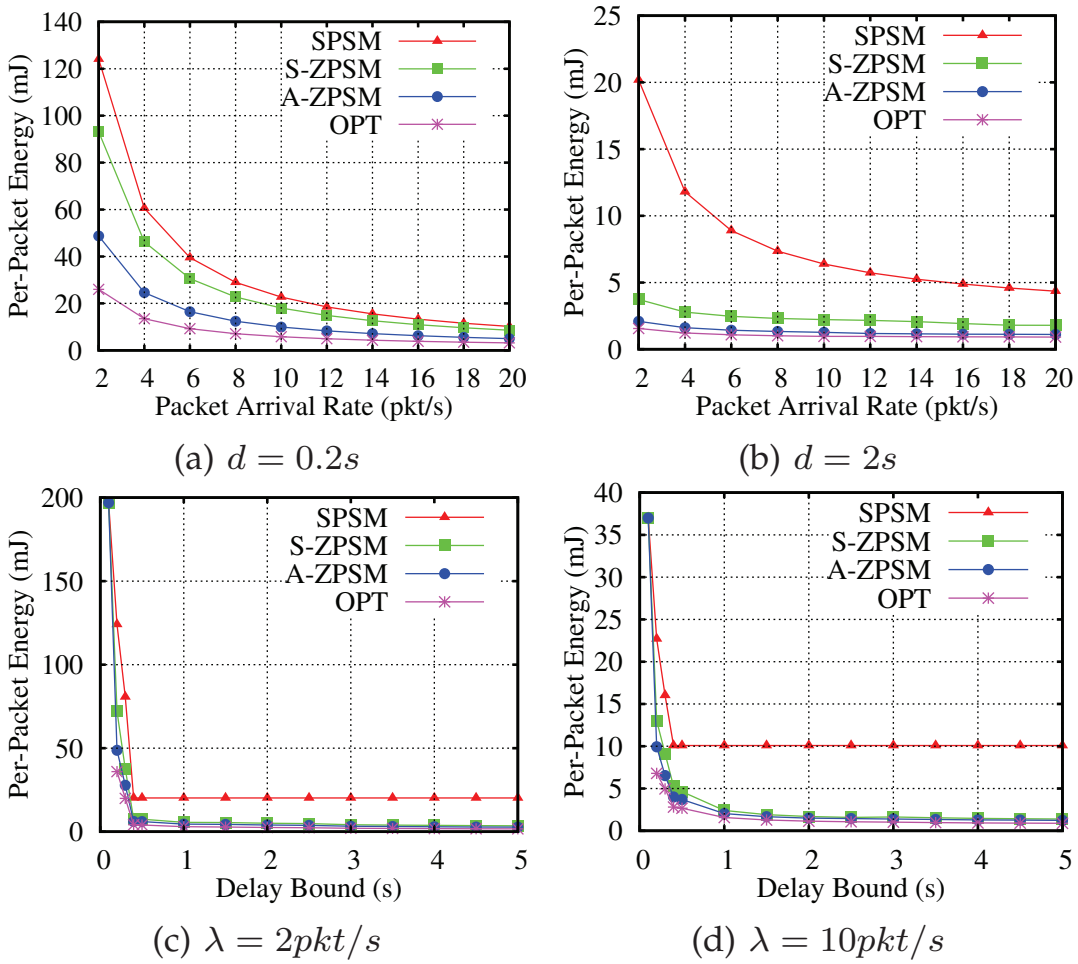
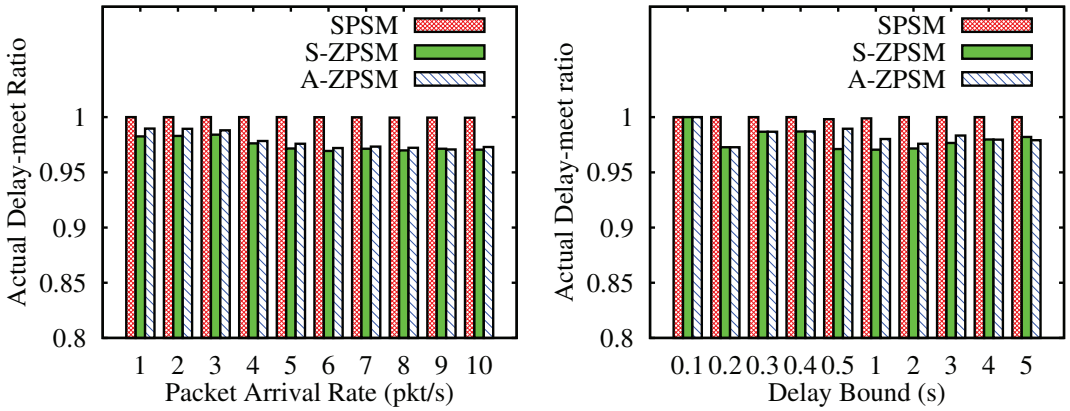


Figure 3.10: Comparison of energy consumption ($p = 0.7$ and $\delta = 0.9$)

3.5.2.1 Per-packet Energy Consumption

Fig. 3.10 compares the energy consumption of different schemes with different packet rates and delay bounds. Generally, our proposed A-ZPSM achieves a level of energy consumption that is very close to OPT and much lower than SPSM and S-ZPSM. In Fig. 3.10a and 3.10b, when traffic rate increases, the per-packet energy efficiency of all schemes decreases because the energy efficiency is averaged on more packets. A-ZPSM outperforms SPSM because SPSM always uses a fixed BI (typically 100 ms), without adapting to actual delay requirements, and hence introduces higher wakeup overheads. A-ZPSM outperforms S-ZPSM since S-ZPSM does not consider transmission workload balancing and therefore may result in more contention among clients, leading to long idle listening and thereby high energy consumption. Note that the energy efficiency of ZPSM in long delay scenarios (e.g., Fig 3.10b) is higher than that in short delay scenarios (e.g., Fig 3.10a), because more packets can be transmitted at each wakeup, resulting in less wakeup overheads. However, in the short delay case, the energy saving of ZPSM over SPSM is still as high as 60% on average.



(a) $d = 2s$ (b) $\lambda = 5pkt/s$

Figure 3.11: Comparison of actual delay-meet ratio ($p = 0.7$ and $\delta = 0.9$)

In Fig. 3.10c and 3.10d, when the average delay bound is below 0.4s, some ZPSM clients are SD clients. With SPSM, these clients must stay in CAM all the time to ensure bounded delay. With ZPSM, these clients only need to enter CAM temporarily when ZigBee link quality is bad, resulting in significantly lower energy consumption. However, when the required delay bound is too short (e.g.,

$d = 100ms$ in the simulation), our proposed ZPSM cannot ensure delay requirements with current ZigBee link quality. In this case, ZPSM clients have to switch to CAM and consume the same amount of energy as SPSM.

In sum, our proposed ZPSM can deliver much higher energy efficiency than SPSM for both SD and LD data traffic.

3.5.2.2 Actual Delay-meet Ratio

Fig. 3.11 shows the actual delay-meet ratio achieved by different schemes. Obviously, SPSM always delivers the largest delay-meet ratio (≈ 1) since it requires the highest wakeup frequency. Nevertheless, when the required delay-meet ratio is 90%, A-ZPSM can also achieve an actual delay-meet ratio above 95% in various scenarios. In particular, A-ZPSM achieves a slightly higher delay-meet ratio than S-ZPSM because of the shorter waiting time resulted from reduced contention.

3.6 Implementation

As a proof of concept, we implement a prototype of our proposed ZPSM system in a testbed with 9 DELL D-Series laptops, each running the Ubuntu Linux 10.04. Each laptop is also equipped with a D-Link WNA-2330 wireless adapter (802.11g, Atheros chipset, PCMCIA) and a Crossbow ZigBee-enabled telosB mote. The system is implemented upon MadWiFi [45], the ZigBee communication is implemented upon TinyOS 2.1.1 platform. Particularly, we have implemented SPSM, S-ZPSM and A-ZPSM. To measure the energy consumption, we first measure the time that each client spends for transmission, reception, sleeping and idle listening, which is a good indicator of the power consumed by the radio interface [46]. Then, the energy consumption is computed according to Table 3.1.

In our experiments, 9 laptops form a wireless LAN, where there is one AP and 8 clients. The WiFi interfaces are tuned to Channel 8 (the least used channel near our testbed to minimize the interference from other co-existing wireless LANs), and the ZigBee interfaces are tuned to Channel 26. Client's ZigBee runs a TDMA-like protocol (i.e., wake up every wakeup slot to receive possible wakeup frame from the AP as described in Section 3.3) while the underlying MAC protocol is CSMA/CA specified by the IEEE 802.15.4 standards.

Figure 3.12: Experiment testbed

To measure energy consumption of WiFi devices, there exist many methods, e.g., reading the supply voltage and current [47], using power monitors [48], installing energy measurement applications such as [49]. To focus on energy consumption of the radio interface, we choose to measure the time that each node spends for transmission, reception, sleeping and idle listening. Then, the energy consumption is computed according to Table 3.1.

3.6.1 Performance Comparison

In the experiment, we also compare A-ZPSM with SPSM and S-ZPSM. Specifically, we first study the performance of the system where all clients are LD clients. Then, we investigate the performance of the system where all clients are SD clients.

3.6.1.1 LD Clients

To emulate the heterogeneity of clients in realistic environment, we configure 8 clients with different settings as shown in Table 3.2. For each client i, δ_i is fixed at 0.9.

From the results shown in Fig. 3.13, we can see that A-ZPSM outperforms SPSM and S-ZPSM in terms of energy consumption and has a delay-meet ratio comparable with SPSM. Similar to the above

Client i	λ_i (pkt/s)	d_i (s)	p_i
1	1	1	0.5
2	1	1	0.9
3	1	5	0.5
4	1	5	0.9
5	10	1	0.5
6	10	1	0.9
7	10	5	0.5
8	10	5	0.9

Table 3.2: Settings of LD clients

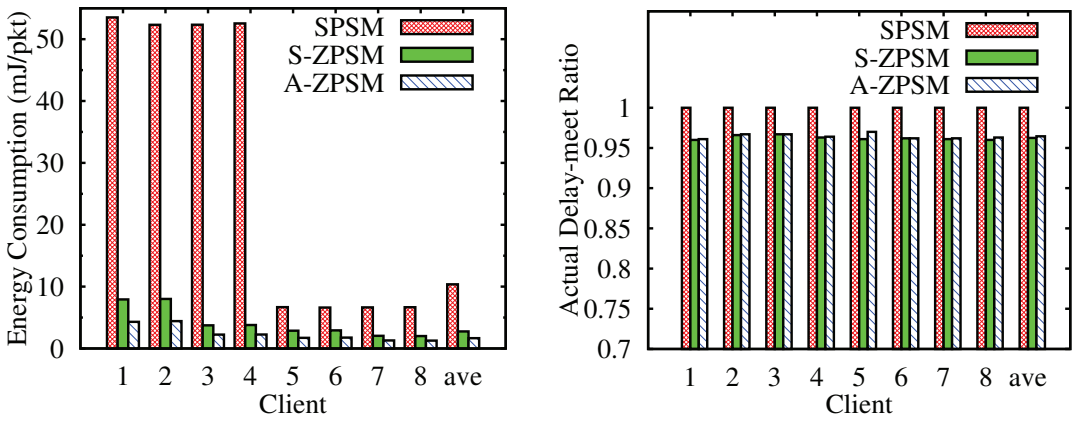


Figure 3.13: Performance comparison

simulation results, the clients with lower packet rate or smaller delay bound have higher per-packet energy consumption. Compared to these two factors, the impact of link quality is less significant, which implies that A-ZPSM is able to tolerate bad ZigBee link quality. On average, our proposed A-ZPSM can save 83.9% and 39.5% more energy than SPSM and S-ZPSM, respectively.

3.6.1.2 SD Clients

To study the performance of A-ZPSM with SD clients, we associate four SD clients with the AP and set their delay bound to 100 ms, 125 ms, 150 ms and 200 ms, respectively. Two sets of experiments (with $\lambda = 1$ pkt/s and $\lambda = 10$ pkt/s, respectively) have been conducted by configuring all clients to A-ZPSM and SPSM, respectively. The results are shown in Tables 3.3 and 3.4.

From the results, we can see that with A-ZPSM only one client (i.e., Client 1) always stays in

CAM. However, with SPSM three clients (i.e., Client 1, 2 and 3) need to run CAM as using only SPSM cannot satisfy their delay requirements. Particularly, the energy consumption of Client 2 with A-ZPSM is slightly higher than that of Client 4 with SPSM. This is because Client 2's delay requirement is very tight. Thus, Client 2 with A-ZPSM sets its LI to 1 to wake up every BI, which is exactly the same as SPSM. Meanwhile, on-demand wakeup is also performed every wakeup slot. However, with SPSM, Client 2 is forced into CAM due to low delay, which consumes much more energy. Also, we can see that when delay bound is set to 100 ms (e.g., Client 1) our proposed ZPSM cannot ensure delay requirements and thereby rollbacks to CAM.

Client i	Delay bound (ms)	Energy (mJ/pkt) A-ZPSM	Energy (mJ/pkt) SPSM
1	100	254.756	254.304
2	125	48.379	255.851
3	150	13.706	253.208
4	200	6.929	44.8392

Table 3.3: Energy consumption with $\lambda = 1$ pkt/s

Client i	Delay bound (ms)	Energy (mJ/pkt) A-ZPSM	Energy (mJ/pkt) SPSM
1	100	22.326	22.473
2	125	6.612	22.549
3	150	4.120	22.103
4	200	2.879	6.395

Table 3.4: Energy consumption with $\lambda = 10$ pkt/s

In addition, with A-ZPSM, Client 1 achieves a delay-meet ratio of nearly 1 as it is always in CAM. The delay-meet ratio of all others are above 0.95. With SPSM, all clients have a delay-meet ratio close to 1. However, much more energy is wasted on staying in CAM. Due to space limitation, we do not show the detailed results.

3.6.2 Impact of Background CAM Traffic

To focus on the impact of coexisting CAM traffic on pure ZPSM system (i.e., no client runs CAM at any time), we use the same experimental setup shown in Table. 3.2. In the experiment, we let the AP

send data traffic to a dummy client, which runs CAM all the time. Fig. 3.14 shows the performance of ZPSM as the CAM traffic rate increases. As we can see, the performance degrades as the CAM traffic rises. This is because as the background CAM traffic increases, the contention between CAM and PSM clients gets more intensive, leading to longer waiting time and thereby lower delay-meet ratio and higher energy consumption. Particularly, in response to the growth of CAM traffic, the delay-meet ratio decreases approximately linearly. The energy consumption is increased to 2.19 mJ, which is still much lower than 10.37 mJ of SPSM.

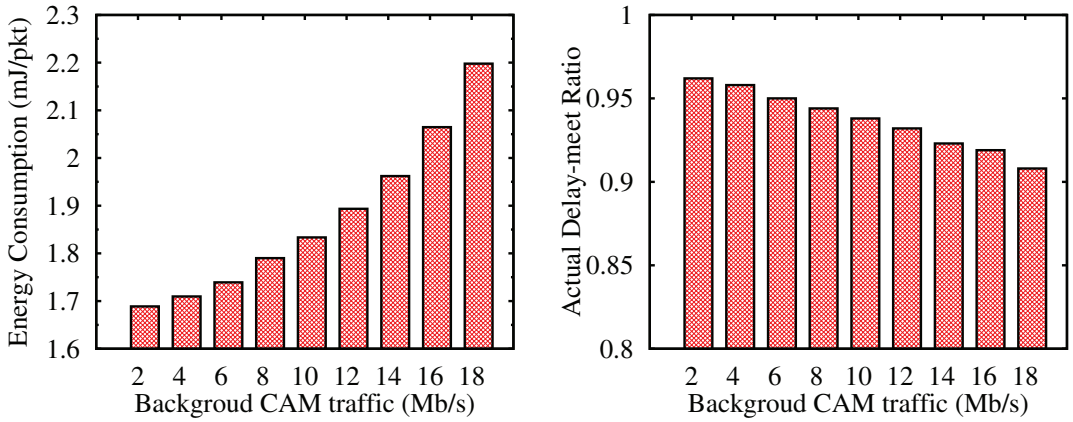


Figure 3.14: Impact of background CAM traffic

3.6.3 Client Joining

To study the stability of our protocol in mobile environment, we let all clients initially run SPSM and then join to ZPSM system one by one. The corresponding energy consumption of SPSM and ZPSM clients is measured and shown in Fig. 3.15a. From the results, we can see that the overall energy consumption decreases significantly as more clients join our proposed ZPSM system. At the same time, the energy consumption of SPSM slightly increases while that of ZPSM slightly decreases. This is because the SPSM clients suffer more from ZPSM traffic while ZPSM clients suffer less from SPSM traffic. while the energy consumption grow obviously. For delay-meet ratio, since the impact is negligible, we do not show the results.

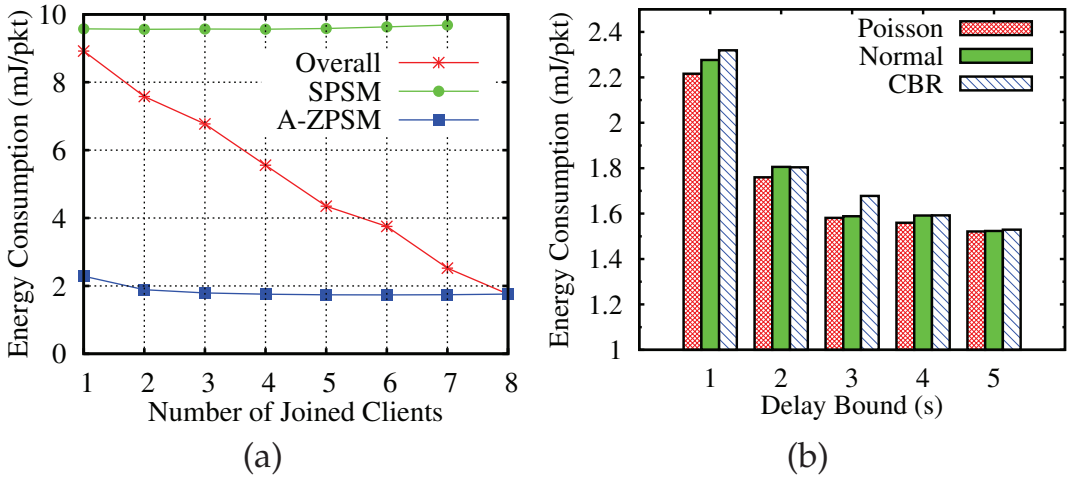


Figure 3.15: Impact of network size and packet arrival models ($\lambda = 5ptk/s$, $d = 2s$, $p = 0.7$ and $\delta = 0.9$)

3.6.4 Packet Arrival Model

Besides, we also study the energy performance of ZPSM under different traffic models. From the results shown in Fig. 3.15b, we observe that the energy consumption with Poisson arrival model is the lowest. The reason is found to be that the Poisson model generates traffic with a larger variance than other models.Thus, it is more likely that some BIs have zero packet arrival while others have many. In this case, our proposed ZPSM can be taken full advantage of to avoid more unnecessary wakeups, resulting in higher energy efficiency.

3.6.5 Wakeup Frequency in Varying Traffic Patterns

To study more detailed wakeup behaviors of individual client, we customize a sequence of packet arrivals, based on which the AP generates traffic to one of the clients in the system. Specifically, the packet sequence consists of four subsequences. Each is a sequence of 500 packet arrivals following a certain packet arrival model, as shown in Fig. 3.16a. The wakeup frequency for the client to retrieve every 100 packets is plotted in Fig. 3.16b.

On average, the overall wakeup frequency is close to our expected wakeup frequency, i.e., $1/d = 0.5$, as our proposed ZPSM adapts wakeup frequency to delay bound for power saving. When the

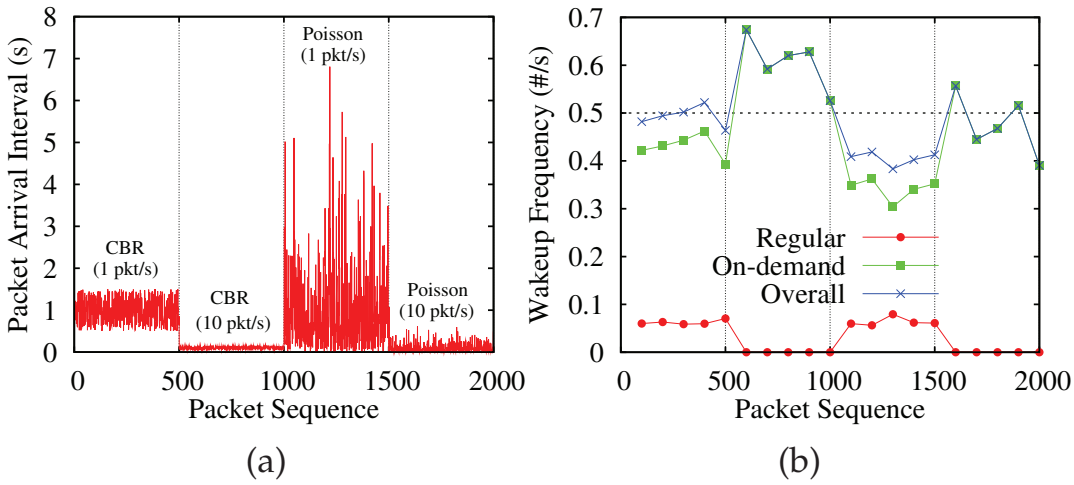


Figure 3.16: Regular *v.s.* On-demand wakeup

Poisson arrival model is used, the wakeup frequency gets lower. This complies with the above results.Besides, the frequencies of on-demand and regular wakeup are also measured to demonstrate how the ZigBee and WiFi interfaces collaborate together to save energy. From the results, we can see that when packet rate is high, ZigBee is used more frequently for on-demand wakeup since the expected interval between two on-demand wakeups (i.e., τ_i) decreases according to Eq. (3.7). As the overall wakeup frequency remains almost the same, the frequency of regular wakeup decreases, resulting in less energy consumption. Particularly, when packet rate changes to 10 pkt/s, the LI is configured to the maximum (i.e., Y_{MAX}) in order to maximize system performance. As a result, the regular wakeup frequency approaches zero.

3.7 Summary

In this chapter, we propose a ZigBee-assisted PSM to improve energy efficiency of WiFi communication in WLAN. Results of simulation and prototype-based experiment have shown significant improvement on energy consumption compared to the standard PSM system, which demonstrates great potential of utilizing ZigBee technology for more energy-efficient data transmission in infrastructure-based wireless networks. Moreover, the proposed wakeup strategies also give a guideline for the follow-up research on energy efficiency in MANET, which will be presented in Chapter 4.

CHAPTER 4. UTILIZING ZIGBEE FOR MORE ENERGY-EFFICIENT MANET

4.1 Overview

In an infrastructure-based wireless network, such as WLAN, mobile devices are connected to base stations (e.g., APs, routers) and the base stations are connected to wired backbone networks. One major drawback of these networks is that they rely on a fixed infrastructure and lack of self-organization capabilities. Thus, when disasters such as earthquakes and hurricanes strike, the network may collapse because of power outage or infrastructure damages. Unfortunately, it is in such emergency situations that responsive and reliable communication is vitally important to obtain accurate and consistent pictures of the situation to minimize casualties and damages.

Featured by quick deployment and self-organization, ad hoc networks enable communication in emergency situations. For example, in the aftermath of a major earthquake, people trapped underground may use their smart phones to form an ad hoc network to communicate with rescuers. However, it is a difficult mission to build a sustainable infrastructure-less communication system out of lightweight peer-to-peer communication devices such as smart phones. The difficulty is mainly due to the requirement of simultaneously addressing the challenges of (i) highly-constrained power supply and short communication range of individual devices and (ii) inherent resource heterogeneity among them.

Power saving management (PSM) has been standardized for IEEE 802.11's distributed coordination function (DCF). But PSM may incur long packet delivery delay [14,16,50], especially in multi-hop networks. This could be intolerable in emergency situations, where responsive communication is essential. To reduce end-to-end delay, protocols adopting proactive or reactive approaches have been proposed. Proactive protocols [12, 13, 51] construct a backbone of constantly active nodes to deliver all traffic while all other nodes are managed by PSM and hence can go to sleep whenever possible.

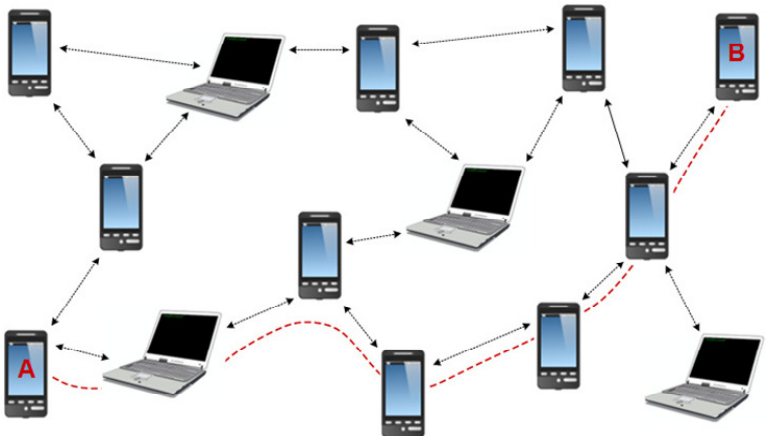


Figure 4.1: A MANET formed by mobile devices

Although these protocols ensure low delay, the energy efficiency may be sacrificed especially when data traffic is light, since backbone nodes always have to stay awake to maintain connectivity. Reactive approaches, such as ODPM [14], allow nodes to be managed by PSM as long as there is no incoming data traffic. However, once a node switches to the constantly active mode (CAM) and transmits data packets, it cannot promptly switch back to the PSM mode because of the wakeup latency caused by the standard PSM. Therefore, the performance improvement may be limited.

To overcome the above limitations, we propose ZigBee-assisted power saving management (ZPSM) for DCF, aiming to maximize network lifetime for *sustainable* ad hoc networks with stringent packet delivery delay requirements. The key ideas are two-fold: (i) the low-power ZigBee radios are utilized to dynamically wake up sleeping high-power WiFi radios for data transmission between nodes, which is similar to the key idea adopted by the ZPSM system proposed for WLAN in Chapter 3; (ii) devices with more energy are managed to wake up more frequently and thereby consume more energy to attain low packet delivery delay, while devices with less energy supply are allowed to wake up less frequently and thereby extend their nodal lifetime, at the cost of longer delay that can be remedied by devices of more energy. ZPSM is built atop the standard PSM and independent of upper layer routing protocol; hence, it is able to communicate with devices running the standard PSM and also compatible with various routing protocols.

To evaluate the performance of ZPSM in large-scale networks, a ns2-based simulator is built and

extensive simulations are conducted. The results show that ZPSM can prolong network lifetime in a wide range of scenarios. To verify the feasibility and the performance of our proposed system in reality, a prototype is built using a testbed of nine laptops equipped with both WiFi and ZigBee interfaces. The experimental results also demonstrate that ZPSM can significantly improve network lifetime (e.g., 168.8% and 140.1% longer than that achieved by ODPM and the standard PSM in moderate traffic scenarios), and accomplish a level of performance close to an upper bound derived though theoretical analysis.

In the following, Section 4.2 presents preliminaries. Section 4.3 describes the design framework, followed by theoretical analysis in Section 4.4. Section 4.5 elaborates the detailed design of ZPSM. The results of simulation and prototype implementation are reported in Sections 4.6 and 4.7. Finally, Section 4.8 summarizes this chapter.

4.2 Preliminaries

4.2.1 Standard Power Saving Management for DCF

Our proposed system is designed based on and compatible with the standard PSM for IEEE 802.11 DCF, called SPSM hereafter. In SPSM, time is divided into beacon intervals (BIs). One beacon interval is composed of an ATIM (Ad-hoc Traffic Indication Message) window and a data transmission window. Every station stays awake during the ATIM window. At the beginning of the ATIM frame, each station waits for a very short random time interval before it broadcasts a beacon frame to announce its presence. Once a station receives a beacon frame from other station, it cancels its pending beacon frame. If a station has packets to send, it transmits an ATIM frame to announce the packets during the ATIM window. Upon receiving an ATIM frame that indicates a packet destined to it, the receiver station replies with an ATIM ACK. During the data transmission window, both sender and receiver stay awake and follow the normal DCF procedure to deliver the packets. Stations having no packet to send or receive instead go to sleep during the data transmission window.

4.2.2 System Model

We consider an ad hoc network of wireless devices. Each device (called node hereafter) has a WiFi and a ZigBee interfaces. The energy of the wireless devices may not be replenished, and thus sustainability is a major concern. Also, delay-bounded communication over the network is desired. However, with the above constraint on energy replenishment and requirement for sustainability, the required delay bound for packet transmission should not be too stringent. Particularly, we assume the desired per-hop delay bound ranges from tens to hundreds of milliseconds (e.g., from 20 ms to 100 ms). Such a range of delay bound may be acceptable for applications such as text/audio messaging/broadcasting, short file transmission, etc., which are useful in emergency scenarios for the purpose of information collection, response direction, rescue coordination and so on.

When a node needs to communicate with another node, a routing protocol may be employed to set up a route between them. For a link on a route, it is called *Z-link* if the two nodes incident to the link are within both the ZigBee and WiFi communication ranges of each other; it is called *W-link* if the two nodes are within the WiFi but not ZigBee range of each other (see Table 1.1). As with SPSM, all nodes are time synchronized through exchanging beacons.

4.2.3 Design Objectives

Our design goal is to adjust the duty cycles of WiFi and ZigBee interfaces of all nodes to increase the network lifetime while satisfying certain end-to-end delay requirements for packets. Here, the network lifetime is defined as the time from the formation of the network till when the first node uses up its energy. The end-to-end delay of a packet is defined as the time elapse from the arrival of the packet at the source node to the receipt of the packet at its destination node. The data transmission along a route is referred to as a *data flow*. Besides, for compatibility, our designed system should be able to communicate with the nodes that run SPSM.

4.3 Design Framework

To maximize network lifetime while satisfying end-to-end delay requirements, we adopt the following principles to design the system.

4.3.1 Lowering Duty Cycles for WiFi Interfaces

As the first principle, the duty cycle of WiFi interfaces should be made as low as possible, through keeping the WiFi interfaces asleep as long as possible and utilizing the ZigBee interfaces to wake up the WiFi interfaces only when necessary.

Let us consider a Z-link from nodes u to v, denoted as Z-link (u, v). To reduce the energy consumption of the two nodes, their WiFi interfaces should stay asleep as long as possible; to ensure that a data packet arriving at u gets to v within a certain required delay bound, however, the WiFi interfaces should be woken up in time to perform the transmission. Considering these aspects simultaneously, we propose a wakeup strategy consisting of two components: *regular wakeup* and *on-demand wakeup*.

4.3.1.1 Regular Wakeup

Like in SPSM, each node in ZPSM wakes up regularly every BI to send or receive beacon. If a node has no packets to send or receive, it immediately puts its WiFi interface to sleep after ATIM window. If a node has packets to send, or has packets to receive, the node puts its WiFi interface to sleep immediately after the sending or receiving has completed.

4.3.1.2 On-demand Wakeup

Once a node puts its WiFi interface to sleep after a regular wakeup, its ZigBee interface is utilized to wake up the WiFi interface on demand when necessary. Specifically, after a data packet arrives at a node (say, node u) when the WiFi interface of its receiver (say, node v) has been put into sleep, the node first buffers the packet and waits for a certain period of time (called *wakeup backoff*). When the wakeup backoff expires, it starts to send an ATIM frame (called *ZATIM*) through its ZigBee interface to the ZigBee interface of the receiver. Due to possible communication failure, a ZATIM frame should be sent by the sender repeatedly during a time window called *rendezvous backoff*. The length of a rendezvous backoff window, i.e., the expected time for the receiver to successfully receive the ZATIM frame, is estimated by the sender based on its local information (i.e., observed quality of the ZigBee link) and is piggybacked in the ZATIM frame. Once the receiver has received the ZATIM frame, it should wake up its WiFi interface after the rendezvous backoff expires, and sends out an ATIM ACK

frame. The sender should also have its WiFi interface awake after the rendezvous backoff expires to receive the ATIM ACK, and then it can send out the buffered packets. Fig. 4.2 illustrates the wakeup strategy used for communication between nodes u and v.

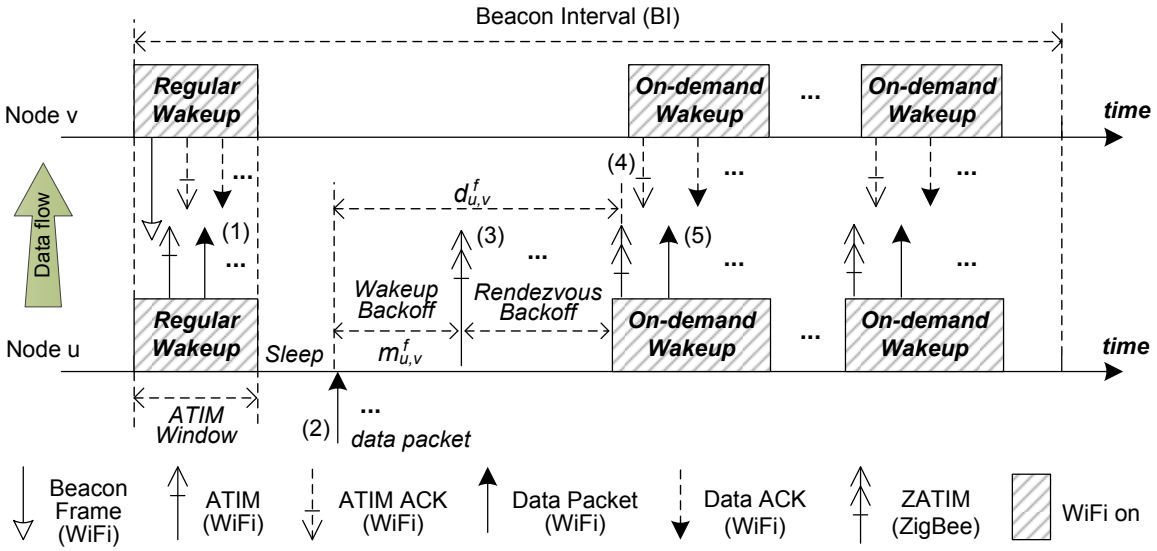


Figure 4.2: Wakeup strategy: interactions between nodes u and v over Z-link (u, v). (1) u and v wake up during each regular wakeup window to exchange beacons and buffered data packets, and then can go to sleep. (2) When a new packet arrives at u, u conducts a wakeup backoff in order to buffer more packets. (3) When wakeup backoff expires, u repeatedly sends ZATIMs to v via its ZigBee interface until rendezvous backoff expires. (4) When rendezvous backoff expires, u turns on its WiFi interface. Meanwhile, if v successfully receives a ZATIM, it also turns on its WiFi interface and sends an ATIM ACK to u. (5) u sends all buffered data packets to v during the on-demand wakeup window and then both nodes go to sleep to save energy.

As we can see, on-demand wakeup can improve energy efficiency as nodes wake up their WiFi interfaces only when necessary. However, due to possible unstable link quality, low rate of ZigBee channel and possible unavailability of ZigBee interface at a node, the wakeup overheads (i.e., transmission/receiving of ZATIM and ATIM ACK, and idle listening for ATIM ACK) may be not negligible. Therefore, choosing an appropriate length for the wakeup backoff to control the frequency of on-demand wakeup is critical, as it offers a tradeoff between energy and delay (as elaborated in next section). Particularly, *longer wakeup backoff results in longer delay but higher energy efficiency*, since more data packets may be buffered during backoff and thus wakeup overheads can be reduced. To achieve our design objective, however, wakeup backoff should be appropriately chosen to also ensure

delay requirements not be violated.

Complementary to the on-demand wakeup, regular wakeup is necessary due to the following reasons: (i) Since the transmission range of WiFi interface is longer than ZigBee interface, some nodes that are within the WiFi range of a node may not be within its ZigBee range. Regular wakeup is necessary not only to maintain connectivity and ensure time synchronization, but also to provide broadcast capability which is existential for routing discovery and maintenance. (ii) Due to unreliable link quality of ZigBee channel, ZigBee transmission may fail occasionally. Adopting regular wakeup ensures a lower bound of performance (i.e., delay and packet delivery ratio). (iii) With regular wakeup strategy, network nodes in our proposed system are compatible with and thereby able to communicate with the legacy nodes which do not have ZigBee interfaces and run the SPSM.

In addition, ZigBee interfaces can also be duty-cycled to further save energy consumption. If not used for other purpose, the ZigBee interface of each node only needs to wake up periodically for sending/receiving ZATIM. The length of the ZigBee wakeup interval is called *wakeup slot*, which is set to 20 ms (typical time to transmit one packet via the ZigBee interface) in our design.

4.3.2 Balancing Nodal Lifetime

For any two nodes incident to a link, the extent to which the duty cycles of their WiFi interfaces can be lowered is essentially determined by the per-hop delay requirement for the link. Hence, the second principle is that, the end-to-end delay of a route should be appropriately distributed to all links forming the route, such that the links connecting shorter-nodal-lifetime nodes are allowed longer per-hop delay bounds; this way, the nodal lifetime can be balanced and thus the network lifetime can be extended. More specifically:

- To prolong network lifetime, the nodes with shorter nodal lifetime should wake up less frequently to transmit data packets, resulting in lower energy consumption but larger link delay.
- To ensure the end-to-end delay of the whole flow, the nodes with longer lifetime should wake up more frequently, resulting in smaller delay but higher energy consumption.

According to the principle, an example is illustrated in Fig. 4.3, where a route $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_5$ is considered, and it is required that each packet be transmitted from v_1 to v_5 within a certain required

end-to-end delay bound D^f.

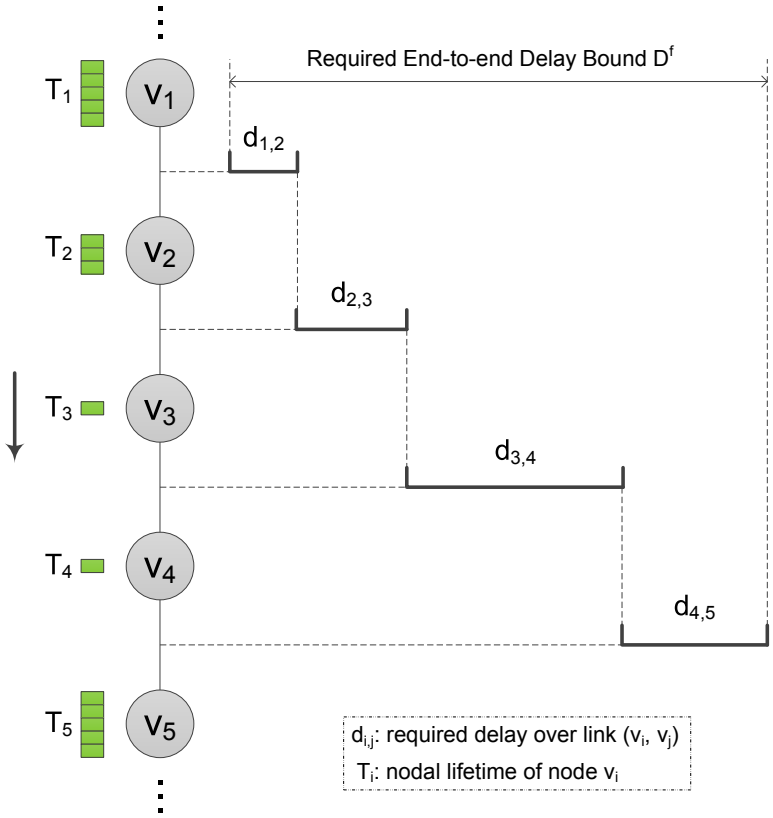


Figure 4.3: Per-link delay allocation for nodal lifetime balancing. Nodes v_3 and v_4 have the shortest nodal lifetime. Thus, link (v_3, v_4) is allocated with the longest packet delivery delay (i.e., $d_{3,4}$) to incur the lowest energy consumption rate. In contrast, as link (v_1, v_2) is associated with the nodes (i.e., v_1 and v_2) with the longest lifetime, the allocated link delay (i.e., $d_{1,2}$) is the smallest to incur the highest energy consumption rate. Although the distribution of nodal lifetime may change over time, the end-to-end delay bound D^f is guaranteed through dynamically adjusting link delay.

4.4 Theoretical Analysis

This section provides theoretical analysis for the proposed design framework, which leads to a performance upper bound that can serve as a reference for evaluating our practical design.

4.4.1 Assumptions

The following assumptions are made to simplify analysis, though our practical scheme presented in the next section does not need these assumptions.

- Ideal WiFi channel conditions (no data packet loss) are assumed. The packet delay due to contention is either negligible or constant.
- Data packet generation follows the Poisson distribution. The size of all data packets is the same.
- The system is not saturated and no packet is dropped due to overflow of queue. Thus, buffered packets will be eventually transmitted.

The notations frequently used are listed in the following table.

$\mathbf{F}$	set of data flows
$\mathbf{V}$	set of all nodes $v \in f, \forall f \in \mathbf{F}$
$\mathbf{E}$	set of all links $(u, v) \in f, \forall f \in \mathbf{F}$
$d_{i,j}^f$	expected delay over link (u, v) for flow f
$m_{i,j}^f$	length of wakeup backoff for link (u, v) on flow f
$p_{i,j}$	ZigBee link quality of link (v_i, v_j)
λ^f	average packet arrival rate of flow f
D^f	required end-to-end delay bound of flow f
e_i	expected energy consumption rate of node v_i
E_i	initial energy capacity of node v_i
B	length of WiFi beacon interval
W	length of ZigBee wakeup slot

Table 4.1: Notations

4.4.2 Delay for Delivering Data Packets over a Link

For an arbitrary Z-link (v_i, v_j) on flow f, we analyze the delay for delivering a packet from node v_i to v_j. When the packet arrives at v_i, node v_i and v_j could be awake or asleep. It is obvious that the largest delivery delay is incurred when both v_i and v_j are asleep. Hence, we only study the worst case. In this case, according to the framework of our design, node v_i needs to wake up itself and v_j through either on-demand or regular wakeup strategies.

4.4.2.1 Delay with On-demand Wakeup

Let $p_{i,j}$ denote the ZigBee link quality of (v_i, v_j), i.e., the probability that a ZigBee packet sent by v_i arrives at v_j successfully. Let $m_{i,j}^f$ denote the length of wakeup backoff of v_i for waking up v_j, T_{ATIM-A} is the time to transmit an ATIM-ACK, and W be the ZigBee wakeup slot. Then, the expected delay for a packet to be delivered from node v_i to v_j through on-demand wakeup, denoted by $d_{i,j}^{f,dem}$, is $m_{i,j}^f + W/p_{i,j} + T_{ATIM-A}$.

4.4.2.2 Delay with Regular Wakeup

Since an on-demand wakeup takes at least $m_{i,j}^f + W + T_{ATIM-A}$ (denoted by $T_{i,j}^f$ for convenience), a packet arriving on or later than $B - T_{i,j}^f$ after the beginning of a BI should be delivered to v_j through next regular wakeup. Thus, the expected delay for a packet to be delivered from node v_i to v_j through regular wakeup, denoted by $d_{i,j}^{f,reg}$, is $T_{i,j}^f + T_{A-win}$, where T_{A-win} is the length of an ATIM window.

In summary, the expected delay $d_{i,j}^f$ for delivering a packet from v_i to v_j can be computed as

$$d_{i,j}^f = \frac{B - d_{i,j}^{f,reg}}{B} \cdot d_{i,j}^{f,dem} + \frac{d_{i,j}^{f,reg}}{B} \cdot d_{i,j}^{f,reg}. \tag{4.1}$$

Note that $d_{i,j}^f = B$ if (v_i, v_j) is a W-link.

4.4.3 Frequency of On-demand Wakeup

We now analyze the expected number of on-demand wakeups in a BI. Specifically, let $X_{i,j}^f$ be a random variable representing the number of on-demand wakeups invoked by node v_i for delivering data packets to v_j along flow f, during a BI. We need to find out the distribution of $X_{i,j}^f$ and then compute its expected value. To facilitate the analysis, we call the interval from when the node v_i was put into sleep to when the next packet arrives as a *doze interval*. Let Y_k^f, where $k = 1, \cdots, X_{i,j}^f$, be a random variable representing the length of the k^{th} doze interval. Suppose the BI begins at time 0 for simplicity. Then, the relations between $X_{i,j}^f$ and Y_k^f can be found as follows.

- **Case I:** The packet transmission of the last on-demand wakeup ends between $B - T_{i,j}^f$ and B.

Then, the following packets arriving after that can be handled by next regular wakeup. Thus,

$$B - T^f_{i,j} \leq X_{i,j} d^{f,dem}_{i,j} + \sum_{k=1}^{X^f_{i,j}} Y_k + T_{A\text{–}win} + B\lambda^f T_{D/A} \leq B, \tag{4.2}$$

where $T_{D/A}$ is the time to transmit/receive a data packet and an ACK.

- **Case II:** The packet transmission of the last on-demand wakeup ends before $B - T^f_{i,j}$ while the next immediate packet arrives after $B - T^f_{i,j}$. Similarly, the following packets are handled by next regular wakeup. Thus,

$$X^f_{i,j} d^{f,dem}_{i,j} + \sum_{k=1}^{X^f_{i,j}} Y^f_k + T_{A\text{–}win} + B\lambda^f T_{D/A} \leq B - T^f_{i,j}, \tag{4.3}$$

and

$$B - T^f_{i,j} \leq X^f_{i,j} d^{f,dem}_{i,j} + \sum_{k=1}^{X^f_{i,j}} Y^f_k + T_{A\text{–}win} + B\lambda^f T_{D/A} + Y^f_{X^f_{i,j}+1}. \tag{4.4}$$

Combining Eq. (4.2), (4.3) and (4.4), we can obtain

$$B - T^f_{i,j} - Y^f_{X^f_{i,j}+1} \leq X^f_{i,j} d^{f,dem}_{i,j} + \sum_{k=1}^{X^f_{i,j}} Y^f_k + T_{A\text{–}win} + B\lambda^f T_{D/A} \leq B. \tag{4.5}$$

Then, the probability that there are n on-demand wakeups during a BI can be computed as follows.

$$P[X^f_{i,j} = n] = P[B - W - T_{ATIM\text{–}A} - m^f_{i,j} - Y^f_{n+1} \leq n \cdot d^{f,dem}_{i,j} + \sum_{k=1}^{n} Y^f_k + T_{A\text{–}win} + B\lambda^f T_{D/A} \leq B]. \tag{4.6}$$

Due to space limitation, we skip the detailed derivations and can get

$$P[X^f_{i,j} = n] = F(B - n \cdot d^{f,dem}_{i,j} - T_{A\text{–}win} - B\lambda^f T_{D/A};\ n, \lambda^f)$$

$$-F(B - W - T_{ATIM\text{–}A} - m^f_{i,j} - n \cdot d^{f,dem}_{i,j} - T_{A\text{–}win} - B\lambda^f T_{D/A};\ n+1, \lambda^f), \tag{4.7}$$

where $F(x;k,\lambda)$ is the CDF of Erlang distribution with parameter k and λ. Specifically, Erlang distribution is the sum of k Exponential random variables with mean λ. Since $x \geq 0$, we have $B - n \cdot d_{i,j}^{f,dem} - T_{A\text{–}win} - B\lambda^f T_{D/A} \geq 0$ for $F(B - n \cdot d_{i,j}^{f,dem} - T_{A\text{–}win} - B\lambda^f T_{D/A};\ n, \lambda^f)$. Then, we have

$$0 \leq n \leq \left\lfloor \frac{B - T_{A\text{–}win} - B\lambda^f T_{D/A}}{d_{i,j}^{f,dem}} \right\rfloor = N. \tag{4.8}$$

Similarly, for $F(B - W - T_{ATIM\text{–}A} - m_{i,j}^f - n \cdot d_{i,j}^{f,dem} - T_{A\text{–}win} - B\lambda^f T_{D/A};\ n+1, \lambda^f)$, we have $B - W - T_{ATIM\text{–}A} - m_{i,j}^f - n \cdot d_{i,j}^{f,dem} - T_{A\text{–}win} - B\lambda^f T_{D/A} \geq 0$, which gives

$$0 \leq n \leq \left\lfloor \frac{B - W - T_{ATIM\text{–}A} - m_{i,j}^f - T_{A\text{–}win} - B\lambda^f T_{D/A}}{d_{i,j}^{f,dem}} \right\rfloor = N'. \tag{4.9}$$

Hence, the expected value of $X_{i,j}^f$, denoted by $x_{i,j}^f$, can be computed as

$$\begin{aligned} x_{i,j}^f &= E[X_{i,j}^f] = \sum_{n=0}^{\infty} n \cdot P[X_{i,j}^f = n] = 0 \cdot P[X_{i,j}^f = 0] + \sum_{n=1}^{\infty} n \cdot P[X_{i,j}^f = n] \\ &= \sum_{n=1}^{N} n \cdot F(B - n \cdot d_{i,j}^{f,dem} - T_{A\text{–}win} - B\lambda^f T_{D/A};\ n, \lambda^f) \\ &\quad - \sum_{n=1}^{N'} n \cdot F(B - W - T_{ATIM\text{–}A} - m_{i,j}^f - n \cdot d_{i,j}^{f,dem} - T_{A\text{–}win} - B\lambda^f T_{D/A};\ n+1, \lambda^f), \end{aligned} \tag{4.10}$$

where $F(x;k,\lambda)$ is given by

$$F(x;k,\lambda) = 1 - \sum_{i=0}^{k-1} \frac{(\lambda x)^i \cdot \exp(-\lambda x)}{i!}. \tag{4.11}$$

4.4.4 Nodal Energy Consumption

Based on the above analysis for on-demand wakeup frequency, we now analyze overall nodal energy consumption rate at each node. Specifically, we need to compute the energy consumption of node v_i *during one BI*, which consists of two parts. One is the basic energy consumption of node v_i for exchanging beacon frames, idly listening in ATIM window and regular ZigBee duty cycling, denoted

by $e_{i,base}$. The other is the energy consumption of node v_i for delivering data flow f, denoted by e_i^f.

Let P_{tx}^w, P_{rx}^w and P_{idle}^w denote the energy consumption rates (in Watt) of WiFi interface for transmitting, receiving and idly listening, respectively. Similarly, let P_{tx}^z, P_{rx}^z and P_{idle}^z denote the energy consumption rates (in Watt) of ZigBee interface for transmitting, receiving and idly listening, respectively.

4.4.4.1 Basic Energy Consumption

Suppose node v_i has n_i neighbors in its vicinity. Then, node v_i consumes energy for exchanging beacons (i.e., $T_B \cdot (P_{tx}^w \cdot \frac{1}{n_i} + P_{rx}^w \cdot \frac{n_i - 1}{n_i})$, where T_B is the time to transmit/receive a beacon frame), idly listening in ATIM window (i.e., $T_{A\text{–}win} \cdot P_{idle}^w$), sensing the ZigBee channel (i.e., $T_{sense} \cdot P_{idle}^z \cdot (B - T_{A\text{–}win})/W$, where T_{sense} is the time to sense the ZigBee channel). Formally,

$$e_{i,base} = T_B(P_{tx}^w \cdot \frac{1}{n_i} + P_{rx}^w \cdot \frac{n_i - 1}{n_i}) + T_{A\text{–}win} \cdot P_{idle}^w + \frac{B - T_{A\text{–}win}}{W} \cdot T_{sense} \cdot P_{idle}^z. \quad (4.12)$$

Note that $e_{i,base}$ is the regular energy consumed by node itself and irrelevant to data flow delivery.

4.4.4.2 Energy Consumption for Delivering Data Flows

To deliver data flow f, each node has to consume energy for receiving and transmitting data packets, which is $\lambda^f \cdot B \cdot T_{D/A}(P_{tx}^w + P_{rx}^w)$ for a BI. Note that, for simplicity, we only consider intermediate node. The computation for source and destination nodes is similar. Then, the additional energy consumed by node v_i for delivering data flow f through *on-demand wakeup* consists of two parts as follows.

- The energy consumed by node v_i for receiving packets from its precursor v_{i-1} on flow f, denoted by $e_{i-1,i}^f$, which consists of the energy consumed for receiving ZATIM from v_{i-1} and the energy consumed for sending ATIM-ACK back to v_{i-1}. Formally,

$$e_{i-1,i}^f = x_{i-1,i}^f(T_{ZATIM} \cdot P_{rx}^z + T_{ATIM\text{–}A} \cdot P_{tx}^w), \quad (4.13)$$

  where T_{ZATIM} is the time to send/receive a ZATIM.

- The energy consumed for transmitting packets to its successor v_{i+1} on flow f, denoted by $e^f_{i,i+1}$, which consists of the energy consumed for sending ZATIM to v_{i+1} and the energy consumed for receiving ATIM-ACK from v_{i+1}. Formally,

$$e^f_{i,i+1} = x^f_{i,i+1}(T_{ZATIM} \cdot P^z_{tx} \cdot \frac{1}{p_{i,i+1}} + T_{ATIM-A} \cdot P^w_{rx}). \tag{4.14}$$

In summary, the energy consumption for delivering data flow f in one BI is given by

$$e^f_i = c_{i-1,i} \cdot e^f_{i-1,i} + c_{i,i+1} \cdot e^f_{i,i+1} + \lambda^f \cdot B \cdot T_{D/A}(P^w_{tx} + P^w_{rx}). \tag{4.15}$$

Here, $c_{u,v} = 1$ if link (u, v) is a Z-link; otherwise, $c_{u,v} = 0$.

Thus, the overall energy consumption rate e_i of node v_i is

$$e_i = \frac{e_{i,base} + \sum_{\forall f \in \mathbf{F}} e^f_i}{B}. \tag{4.16}$$

From above analysis, we can see that *as $m^f_{i,j}$ decreases, $d^f_{i,j}$ decreases (by Eq. (4.1)) and $x^f_{i,j}$ increases (by Eq. (4.5), Eq (4.13) and Eq (4.14)), leading to higher energy consumption on both node v_i and v_j simultaneously*. Thus, $m^f_{i,j}$ is the key system parameter to balance energy and delay.

4.4.5 Optimization Formulation

When the network lifetime is maximized, there must exist one node v_k whose energy reserve is zero while all other nodes' energy reserves are equal or greater than zero. Thus, finding the solution to the original network lifetime maximization problem is equivalent to tentatively solving the following problem for all $v_k \in \mathbf{V}$ and choosing the one that results in the maximal network lifetime as the solution to the original problem.

Objective: Minimize node v_k's energy consumption rate e_k,

s.t.,

$$\sum_{(v_i,v_j)\in f} \Big[c_{i,j} d^f_{i,j} + (1 - c_{i,j})B\Big] \leq D^f, \ \ \forall f \in \mathbf{F}, \tag{4.17}$$

$$\frac{e_i}{E_i} \leq \frac{e_k}{E_k}, \quad \forall v_i \in \mathbf{V} \setminus \{v_k\} \tag{4.18}$$

$$0 \leq m_{i,j}^f \leq m_{max}, \quad \forall (v_i, v_j) \in E, \forall f \in \mathbf{F} \tag{4.19}$$

Here, D^f is the required end-to-end delay bound to flow f and E_i is the initial energy capacity (in Joule) of node v_i. The maximum length of wakeup backoff m_{max} is $B - T_{A\text{–}win} - W - T_{ATIM\text{–}A}$. The set of unknowns is $\{m_{i,j}^f \mid \forall (v_i, v_j) \in E, \forall f \in \mathbf{F}\}$. Particularly, Eq. (4.17) gives the delay constraints for all flows, and Eq. (4.18) ensures that v_k dies first. Thus, if v_k is the bottleneck node, then maximizing the network lifetime is equivalent to minimizing the energy consumption rate of v_k given by the objective function, subjected to the above constraints. Let $L(v_i)$ denote the resulting network lifetime obtained by solving the above problem if $v_k = v_i$. Then, the maximum network lifetime is $\max\{L(v_i)\}_{\forall v_i \in \mathbf{V}}$.

4.4.6 A Performance Upper Bound

Obviously, the above problem is non-linear, as $x_{i,j}^f$ is a non-linear function of $m_{i,j}^f$ as shown in Eq. (4.10). To obtain a computational performance upper bound, we relax our original problem by letting $Y_k = 1/\lambda^f$ in Eq. (4.5) and using the resulting inequality to substitute Eq. (4.10). Then, we can get a loose bound of e_i^f, based on which a linear programming problem can be obtained. Thus, we get a performance upper bound of the solution to our original problem by solving this relaxed problem. The details are elaborated as follows.

First, let $Y_k = 1/\lambda^f$ in Eq. (4.5). We can get

$$B - T_{i,j}^f - 1/\lambda^f \leq x_{i,j}^f (d_{i,j}^{f,dem} + 1/\lambda^f) + T_{A\text{–}win} + B\lambda^f T_{D/A} \leq B \tag{4.20}$$

As $0 \leq m_{i,j}^f \leq m_{max}$, we can get the following loose bound of $x_{i,j}$.

$$g(m_{i,j}^f) = \frac{B - W - T_{ATIM\text{–}A} - m_{i,j}^f - 1/\lambda^f - T_{A\text{–}win} - B\lambda^f T_{D/A}}{m_{max} + W/p_{i,j} + T_{ATIM\text{–}A} + 1/\lambda^f}$$

$$\leq x_{i,j}^f \leq \frac{B - T_{A\text{-}win} - B\lambda^f T_{D/A}}{W/p_{i,j} + T_{ATIM\text{-}A} + 1/\lambda^f} = A_{i,j}^f, \tag{4.21}$$

where $g(m_{i,j}^f)$ is function of $m_{i,j}^f$ and $A_{i,j}^f$ is a constant for convenience.

Then, we get a bound of $e_{i-1,i}^f$ and $e_{i,i+1}^f$, by combining Eq. (4.21), Eq (4.13) and Eq (4.14), as follows.

$$E_{up} \cdot g(m_{i-1,i}^f) \leq e_{i-1,i}^f \leq E_{up} \cdot A_{i-1,i}^f, \tag{4.22}$$

and

$$E_{i,i+1}^{down} \cdot g(m_{i,i+1}^f) \leq e_{i,i+1}^f \leq E_{i,i+1}^{down} \cdot A_{i,i+1}^f, \tag{4.23}$$

where $E_{up} = T_{ZATIM} \cdot P_{rx}^z + T_{ATIM\text{-}A} \cdot P_{tx}^w$ and $E_{i,i+1}^{down} = T_{ZATIM} \cdot P_{tx}^z \cdot \frac{1}{p_{i,i+1}} + T_{ATIM\text{-}A} \cdot P_{rx}^w$.

Combining Eq. (4.22), Eq. (4.23) and Eq. (4.15), we can get a bound of e_i^f

$$c_{i-1,i} \cdot E_{up} \cdot g(m_{i-1,i}^f) + c_{i,i+1} \cdot E_{i,i+1}^{down} \cdot g(m_{i,i+1}^f) + E_{data}^f$$

$$\leq e_i^f \leq c_{i-1,i} \cdot E_{up} \cdot A_{i-1,i}^f + c_{i,i+1} \cdot E_{i,i+1}^{down} \cdot A_{i,i+1}^f + E_{data}^f, \tag{4.24}$$

where $E_{data}^f = \lambda^f \cdot B \cdot T_{D/A}(P_{tx}^w + P_{rx}^w)$.

Applying Eq. (4.24) in our original problem, we can get a relaxed problem as follows.

Objective: Minimize node v_k's energy consumption rate e_k,

s.t.,

$$\sum_{(v_i,v_j)\in f} \Big[c_{i,j} d_{i,j}^f + (1 - c_{i,j})B\Big] \leq D^f, \quad \forall f \in \mathbf{F} \tag{4.25}$$

$$c_{i-1,i} E_{up} \cdot g(m_{i-1,i}^f) + c_{i,i+1} E_{i,i+1}^{down} \cdot g(m_{i,i+1}^f) + E_{data}^f$$

$$\leq e_i^f \leq c_{i-1,i} E_{up} A_{i-1,i}^f + c_{i,i+1} E_{i,i+1}^{down} A_{i,i+1}^f + E_{data}^f, \quad \forall v_i \in V, \forall f \in \mathbf{F} \tag{4.26}$$

$$e_i = \frac{e_{i,base} + \sum_{\forall f \in \mathbf{F}} e_i^f}{B}, \quad \forall v_i \in \mathbf{V} \tag{4.27}$$

$$\frac{e_i}{E_i} \leq \frac{e_k}{E_k}, \quad \forall v_i \in \mathbf{V} \setminus \{v_k\} \tag{4.28}$$

$$0 \leq m_{i,j}^f \leq m_{max}, \quad \forall (v_i, v_j) \in E, \forall f \in \mathbf{F} \tag{4.29}$$

Since $d_{i,j}^f$ and $g(m_{i,j}^f)$ are linear to $m_{i,j}^f$, the above relaxed problem is a linear programming problem with the set of unknowns $\{m_{i,j}^f \mid \forall (v_i, v_j) \in E, \forall f \in \mathbf{F}\} \cup \{e_i^f \mid \forall v_i \in V, \forall f \in \mathbf{F}\}$. Thus, the solution to this relaxed problem is a performance upper bound of the solution to our original problem.

4.5 Design Details

From the above analysis, we can observe that, the wakeup backoff at a node plays a key role in trading off between energy efficiency of the node and packet transmission delay through the node. Hence, in our proposed scheme, wakeup backoff is dynamically adjusted at nodes to balance nodal lifetime without violating packet delivery delay requirements of flows.

The proposed scheme includes the following components: First, *flow initialization* is conducted for each flow to initialize the wakeup backoff for each node on the flow to meet the end-to-end delay requirements. Second, with the assistance of ZigBee interfaces, each pair of neighboring nodes on a flow perform *one-hop adjustments* periodically to balance their nodal lifetime without violating delay requirements. In addition, WiFi interfaces may also be leveraged to perform *multi-hop adjustments* when needed to adjust wakeup backoff for nodes whose nodal lifetime cannot be balanced through one-hop adjustments.

4.5.1 Flow Initialization

To meet end-to-end delay requirements, a flow-wide initialization of wakeup backoff is conducted when a route is being set up. The process is coupled with routing process. Specifically, most of routing protocols (e.g., AODV, DSR, etc.) consist of two steps: *routing request* (RREQ) and *routing*

reply (RREP). When a source node needs to find a route to a destination node, it broadcasts a RREQ packet, which is flooded in a certain scope. Upon receiving RREQ, the destination node can directly or indirectly know a route to the source node. Then, it replies to the source node by sending a RREP packet along the route. The RREP step is leveraged to initialize wakeup backoff for all nodes on the route, as follows.

Let V_w and V_z be the sets of nodes whose outgoing links are W-links and Z-links on flow f, respectively. Then, during the course of relaying RREP, each intermediate node $v_i \in V_z$ piggybacks its expected nodal lifetime L_i (derived from Eq. (4.16)), minimum delay $d_{i,i+1}^{f,min}$ and maximum delay $d_{i,i+1}^{f,max}$ over link (v_i, v_{i+1}) for flow f (derived from Eq. (4.1) by setting wakeup backoff to 0 and m_{max}, respectively). Upon receiving RREP, the source node allocates the wakeup backoff for each node $v_i \in V_z$ according to the procedure shown in Fig. 4.4.

The goal of the procedure is two-fold: meeting the end-to-end delay requirements of the flow and balancing nodal lifetime through allocating wakeup backoff for each node in V_z. To meet the delay requirements, the sum of the delay of all links (including W-links and Z-links) should not exceed the end-to-end delay bound D^f, and the delay of a Z-link (v_i, v_{i+1}) should not be smaller than its minimum value $d_{i,i+1}^{f,min}$. This is achieved by defining the total amount of delay that can be allocated to all Z-links beyond their minimum delay, denoted by D_{alloc}, as $D^f - |V_w| \cdot B - \sum_{v_i \in V_z} d_{i,i+1}^{f,min}$, where the delay of a W-link is B. To balance nodal lifetime, the link incident to nodes with long (short) lifetime should be allocated with a short (long) wakeup backoff so as to delivery low (high) delay with high (low) energy consumption. This is achieved by defining a weight δ_i for distributing link delay $d_{i,i+1}^f$ as

$$\delta_i = \frac{[\sum_{v_k \in V_z}(L_k + L_{k+1})] - (L_i + L_{i+1})}{(|V_z| - 1)\sum_{v_k \in V_z}(L_k + L_{k+1})}, \tag{4.30}$$

which is inversely proportional to the sum of L_i and L_{i+1}. Obviously, $0 \leq \delta_i \leq 1$ and $\sum_{v_i \in V_z} \delta_i = 1$.

Particularly, if the allocated delay to a Z-link is more than its maximum value (i.e., $\delta_i \cdot D_{alloc} \geq d_{i,i+1}^{f,max} - d_{i,i+1}^{f,min}$), the unused delay is saved and then reallocated to other nodes by updating D_{alloc} to $D_{alloc} - (d_{i,i+1}^{f,max} - d_{i,i+1}^{f,min})$. This process continues until no Z-link can be allocated with a delay greater than its maximum delay. Once wakeup backoff is initialized, the source node sends a packet along the route to notify each node of its allocated wakeup backoff. During the course, each node ensures that the

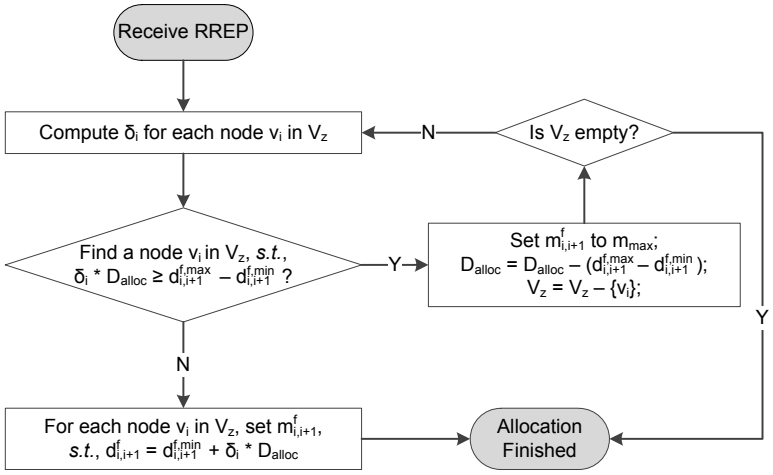


Figure 4.4: Initializing wakeup backoff of nodes on flow f

packet is successfully delivered to its next hop through requesting ACK and performing retransmissions when necessary.

4.5.2 One-hop Adjustment

To balance nodal lifetime, one-hop adjustment is conducted to adjust wakeup backoff between neighboring nodes by leveraging the ZigBee interface. The adjustment is only conducted by node v_i whose incoming link (v_{i-1}, v_i) and outgoing link (v_i, v_{i+1}) on the flow are both Z-links, as shown in Fig. 4.5(a). Moreover, node v_i only adjusts wakeup backoff of itself (i.e., $m^f_{i,i+1}$) and node v_{i-1} (i.e., $m^f_{i-1,i}$), because adjusting wakeup backoff of node v_{i+1} (i.e, $m^f_{i+1,i+2}$) would also affect node v_{i+2}, which is out of the neighborhood of node v_i.

- Each node v_i periodically broadcasts through its ZigBee interface a HELLO message, which contains its expected nodal lifetime (L_i) and wakeup backoff ($m^f_{i,i+1}$).
- Upon receiving a HELLO message from node v_{i-1} or v_{i+1}, node v_i computes $L_{diff} = \max\{L_{i-1}, L_i, L_{i+1}\} - \min\{L_{i-1}, L_i, L_{i+1}\}$.
- If L_{diff} is smaller than a threshold Δ (e.g., 10 min we use in simulations and experiments), the adjustment is canceled to avoid overheads caused by frequent adjustments. Otherwise, node v_i

runs Alg. 2, which goes through all feasible combinations of $\langle m^f_{i-1,i}, m^f_{i,i+1}\rangle$ under current delay constraint (line 6) and chooses $\langle m^{f,opt}_{i-1,i}, m^{f,opt}_{i,i+1}\rangle$ that results in the longest local lifetime L^{opt}_{local}, based on Eq. (4.1), (4.13) and (4.14). Particularly, in our design, wakeup backoff consists of a certain number of time slots of equal length. The slot length is the typical time to transmit a WiFi data packet. Thus, in line 4, wakeup backoff is treated as an integer between 0 and $\lfloor m_{max} \rfloor$.

- If $L^{opt}_{local} - \min\{L_{i-1}, L_i, L_{i+1}\} < \Delta$, then the adjustment is canceled to avoid unnecessary or trivial adjustments. Otherwise, node v_i sends a packet through its ZigBee interface to notify node v_{i-1} of the new wakeup backoff.

- Upon receiving this packet, node v_{i-1} updates its wakeup backoff to $m^{f,opt}_{i-1,i}$ immediately, if it is not currently in the process of making an adjustment with some other node on flow f. Otherwise, no update is made at node v_{i-1}.

- Later, node v_i can tell whether the update is successful or not by overhearing HELLO messages from node v_{i-1}. If it succeeds, node v_i updates its own wakeup backoff to $m^{f,opt}_{i,i+1}$ accordingly; otherwise, the adjustment is canceled.

Algorithm 2 One-hop adjustment on flow f

1: Record the required local delay bound $D_{local} = d^f_{i-1,i} + d^f_{i,i+1}$
2: Define the local lifetime L_{local} as $\min\{L_{i-1}, L_i, L_{i+1}\}$
3: Maximum local lifetime $L^{opt}_{local} = -\infty$
4: **for** $m^f_{i,i+1} = 0$ to $\lfloor m_{max} \rfloor$ **do**
5: Given $m^f_{i,i+1}$, recompute $d^f_{i,i+1}$
6: $d^f_{i-1,i} = D_{local} - d^f_{i,i+1}$ */*Ensure local delay bound*/*
7: **if** $d^{f,min}_{i-1,i} \le d^f_{i-1,i} \le d^{f,max}_{i-1,i}$ **then**
8: Given $d^f_{i-1,i}$, recompute $m^f_{i-1,i}$
9: Given $\langle m^f_{i-1,i}, m^f_{i,i+1}\rangle$, recompute L_{local}
10: **if** $L^{opt}_{local} < L_{local}$ **then**
11: $L^{opt}_{local} = L_{local}$, $m^{f,opt}_{i-1,i} = m^f_{i-1,i}$ and $m^{f,opt}_{i,i+1} = m^f_{i,i+1}$
12: **end if**
13: **end if**
14: **end for**
15: **OUTPUT**: L^{opt}_{local}, $m^{f,opt}_{i-1,i}$ and $m^{f,opt}_{i,i+1}$

4.5.3 Multi-hop Adjustment

With the above one-hop adjustment, the nodal lifetime of two nodes (e.g., node v_i and v_j in Fig. 4.5 (b)) that are interconnected via a W-link cannot be balanced with one-hop adjustment. Even if two nodes (e.g., node v_i and v_j in Fig. 4.5 (c)) are connected via a Z-link, it is also possible that the wakeup backoff of some node (e.g., node u) reaches the minimum value (i.e., 0) or the maximum value (m_{max}) before a balance is attained, and thereby cannot be adjusted any more. In these cases, WiFi interface can be used to perform multi-hop adjustment.

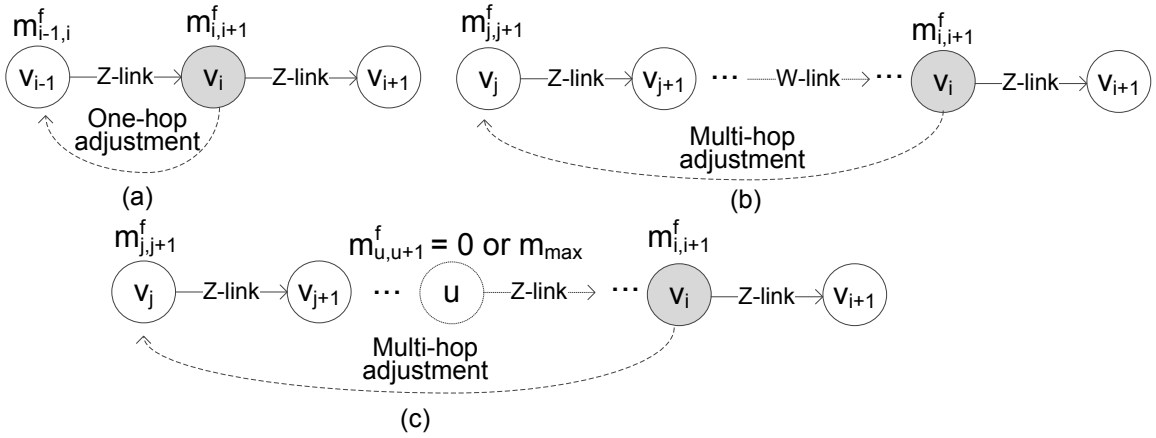


Figure 4.5: Wakeup backoff adjustments on flow f

For a flow f, the multi-hop adjustment works as follows.

- The source node launches the adjustment by setting ADJUST bit in a data packet every certain period of time (e.g., 5 min as we use in our system).
- Upon receiving a data packet with the ADJUST bit set, node $v_i \in V_z$ checks its number of failures on one-hop adjustment (denoted by a_i^f, i.e., the number of *consecutive* times that node v_i fails to find a pair of $\langle m_{i-1,i}^{f,opt}, m_{i,i+1}^{f,opt} \rangle$ that can improve current local lifetime by Δ through running Alg. 2).
- If a_i^f is smaller than a threshold Θ (e.g., 5 as we use), it piggybacks its expected nodal lifetime (L_i), ZigBee link quality ($p_{i-1,i}$), wakeup backoff ($m_{i,i+1}^f$) and a_i^f in the data packet and sends it to next hop.

- Otherwise, it checks the piggybacked information in the data packet to see if it can make an adjustment with one *preceding* node v_j whose L_j, $p_{j-1,j}$ and $m^f_{j,j+1}$ are piggybacked. That is, the resulting local lifetime (i.e., the minimum nodal lifetime among node v_i, v_{i+1}, v_j and v_{j+1}) can be improved by Δ via using Alg. 2. If so, node v_i launches a multi-hop adjustment by sending a packet to notify node v_j of the new wakeup backoff. At the same time, it removes the piggybacked information of node v_j from the data packet to prevent the following nodes from making an adjustment with node v_j. Note that node v_i may find more than one such nodes to adjust, in which case the one that has the largest a^f_j and can improve the lifetime most is chosen. After that, the data packet is sent to next hop.

- Similar to one-hop adjustment, the success of notification can be acknowledged by piggybacking ACK in the following data packets. Upon receiving ACK, node v_i updates its own wakeup backoff. If node v_i cannot receive the ACK for a certain period of time (e.g., the round trip time by assuming each link is a W-link and has a delay of B), this adjustment is canceled.

4.5.4 Practical Issues

4.5.4.1 Links on Multiple Flows

When a link (v_i, v_{i+1}) serves multiple flows, v_i maintains a wakeup backoff for each of the flows. In practice, however, only one wakeup backoff which is *minimal* among all the backoffs for all the flows, is really used in the practical scheme. Note that, the delay requirements of all the flows cannot be violated because the used backoff is minimal among all those maintained and hence the delay introduced over this link does not exceed that allowed by every flow.

4.5.4.2 On-line Backoff Adjustment

From the above discussion, the following phenomenon may occur: a flow f_1 allows a long delay over a link (v_i, v_{i+1}) which however is allowed only a short delay by another flow f_2; hence, all links other than (v_i, v_{i+1}) on f_1 are allowed to have longer delays to save energy at nodes incident to them. Besides, due to random unavailability of the ZigBee interface and irregular traffic patterns, data packet may experience large delay over some links occasionally.

To leverage opportunistically-available extra delay allowance and handle occasional large delays caused, the following on-line adaptive backoff adjustment can be performed, by leveraging available ZigBee interfaces: when a node on a flow finds there is extra delay allowance brought by its precursors and its own nodal lifetime is short, the node can temporarily increase its wakeup backoff to extend its lifetime; when a node finds some packets have experienced longer-than-expected delay before arriving at it, the node can temporarily decrease its wakeup backoff to incur shorter delay to the packets.

4.5.4.3 Estimating Link Quality and Traffic Rate

In the proposed scheme, each node periodically broadcasts HELLO messages through its ZigBee interface to exchange information with its neighbors. The HELLO messages are also used to estimate the link quality and traffic rate between the nodes as follows.

Although different nodes within the ZigBee range broadcast their beacons randomly in our system, the interval is fixed. Thus, each node knows how many beacons each of its neighbors should send within a given time period τ (e.g., 60 s). Suppose node u is a precursor node of node v on a flow, one estimate of the link quality $p_{u,v}$ can be computed by node v as the number of beacons received by node v divided by the number of beacons sent by node u during τ.

In our system, each node v maintains M (e.g., 50) most recently estimates for each precursor node u (i.e., $\hat{p_i}$ for $i = 0, \cdots, M-1$), which are synthesized through *exponential moving average* to derive actual link quality $p_{u,v}$. Particularly, let P_i be the estimated link quality by synthesizing the first i estimates. P_i is computed as

$$P_i = \alpha \cdot \hat{p_i} + (1 - \alpha) \cdot P_{i-1}, \tag{4.31}$$

where $P_0 = \hat{p_0}$ and α is the smoothing factor between 0 and 1, defined as $1 - \exp(-\frac{\tau}{M})$, indicating most recent estimate is more valuable and thereby given a bigger weight if sample size M is small and sampling interval τ is long. Finally, the estimated link quality $p_{u,v}$ is quantified as $p_{u,v} = P_{M-1}$.

Similarly, traffic rate (i.e., packet arrival interval $1/\lambda^f$) of each flow f can be also estimated through moving averaging certain number of most recently packet arrivals.

4.5.4.4 Compatibility with SPSM

As a design goal, ZPSM should be compatible with SPSM; that is, for a link connecting a ZPSM node and a legacy node that do not have ZigBee interface and runs SPSM, the two nodes should be able to communicate with each other. Once the ZPSM node knows that it is connected to a legacy SPSM node, the compatibility is achieved simply as follows: to receive data packets from the legacy node, the ZPSM node stays awake after regular wakeup; to send data packets arriving during the interval between two regular wakeups to the legacy node, the ZPSM node buffers the data packets until next regular wakeup.

4.6 Simulation

To evaluate ZPSM, we conduct extensive simulation based on ns2. In the simulation, we measure following performance metrics:

- *Network lifetime* (unit: hour) is defined as the time from when a network is formed till when one node in the network fails. The initial energy level of each node is randomly chosen between 5000 to 10000 J, which is a typical energy capacity of a mobile phone battery. The energy consumed by the WiFi and ZigBee interfaces is measured according to the specifications of WiFi module [43, 52] and ZigBee CC2530 RF transceiver [44], respectively, as shown in Table 4.2.

- *Delay meet-ratio* is defined as the total number of data packets that meet their delay requirements (at the destination node) divided by the total number of data packets transmitted (by the source node). By default, the packet arrival is modeled as Poisson process.

The simulation uses the network topology shown in Fig. 4.6, where 35 nodes are deployed on campus and there are three data flows: flow 1 from node 7 to 32 (20 hops), flow 2 from node 0 to 22 (16 hops) and flow 3 from node 34 to 0 (14 hops). Also, 84 % of links are Z-link.

The standard IEEE 802.11g is used for WiFi transmission. The detailed settings are shown in Table. 4.2. Average packet arrival rate and average link quality are denoted as λ and p, respectively. Each flow is randomly configured with a packet arrival rate within $(0.5\lambda, 1.5\lambda)$, and each ZigBee link is randomly configured with a link quality $(p-0.05, p+0.05)$. Besides, as mentioned in Section 4.5.1,

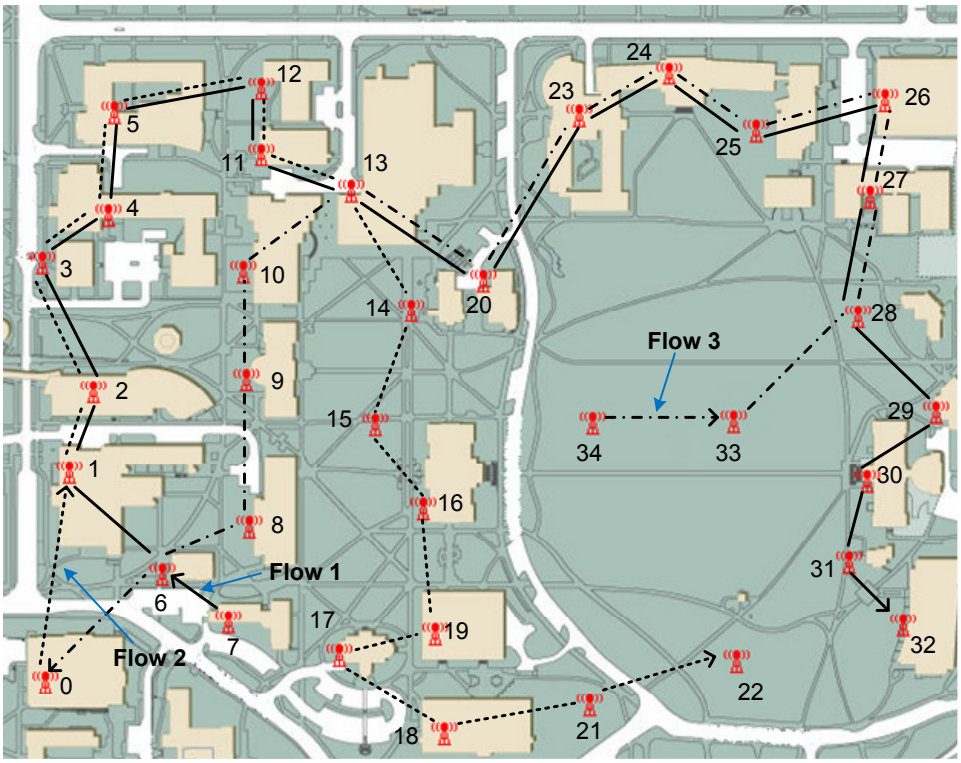


Figure 4.6: Network topology of simulation

the delay that can be achieved by a Z-link (v_i, v_{i+1}) is between $d^{f,min}_{i,i+1}$ and $d^{f,max}_{i,i+1}$. Thus, the feasible end-to-end delay of flow f is between $D^f_{min} = \sum_{v_i \in V_z} d^{f,min}_{i,i+1} + |V_w| \cdot B$ and $D^f_{max} = \sum_{v_i \in V_z} d^{f,max}_{i,i+1} + |V_w| \cdot B$. Hence, the delay bound requirement of flow f is specified as $D^f_{min} + \gamma \cdot (D^f_{max} - D^f_{min})$, where γ, called *normalized delay bound*, is between 0 and 1. Note that, in our simulation and experiments, the minimum per-hop delay, i.e., D^f_{min} divided by the number of hops of flow f, is about 20 ms, while the maximum per-hop delay, i.e., D^f_{max} divided by the number of hops of flow f, is about 350 ms (i.e., one BI). Hence, the simulation considers the per-hop delay bound ranging from about 20 ms to 350 ms. By default, we set $\lambda = 5pkt/s$ and $p = 0.9$; γ is set to 0.5 (i.e., 185 ms per-hop delay bound).

4.6.1 Network Lifetime

By varying normalized delay bound under different traffic rates, we measure the network lifetime and energy consumption of our proposed ZPSM. The results are shown in Fig. 4.7. The energy consumption (J/pkt) is defined as the total energy consumption (including both WiFi and ZigBee) of all

WiFi range	95 m
ZigBee range	75 m
WiFi channel bit rate	54 Mbps
ZigBee channel bit rate	250 Kbps
WiFi beacon interval	350 ms
ZigBee beacon interval	1 s
ATIM window	20 ms
Wakeup slot	20 ms
Data packet size	1023 bytes
Energy capacity	5000∼10000 J
WiFi transmission	1.152 Watt
WiFi reception	0.561 Watt
WiFi idle listening	0.462 Watt
WiFi Power-on	0.544 mJ
WiFi Power-off	0.482 mJ
ZigBee transmission	0.087 Watt
ZigBee reception	0.072 Watt
ZigBee idle listening	0.019 Watt

Table 4.2: System parameters for simulation

nodes divided by the total number of data packets received by all destination nodes. In Fig. 4.7a, we can see that the network lifetime gets longer as delay bound increases. This is because, as delay bound becomes larger, the nodes perform on-demand wakeup less frequently, leading to lower overhead. This trend can also be observed from Fig. 4.7b. It is also shown that the network lifetime decreases as packet arrival rate gets higher, because more data packets are transmitted. However, in Fig. 4.7b, we can see that per-packet energy consumption decreases as packet arrival rate increases. This is because more packets are buffered during one BI and thereby more can be transmitted through one wakeup, resulting in more efficient use of energy.

4.6.2 Delay Performance

Fig. 4.8 shows the delay performance of each flow when delay bounds and packet arrival rates vary. As shown in Fig. 4.8a, the per-hop delay of each flow increases as delay bound increases. Fig. 4.8b shows that, as delay bound increases, the delay meet-ratio of each flow gets higher. Besides, Fig. 4.8c and 4.8d show that, when packet arrival rate increases, per-hop delay goes up and delay meet-ratio

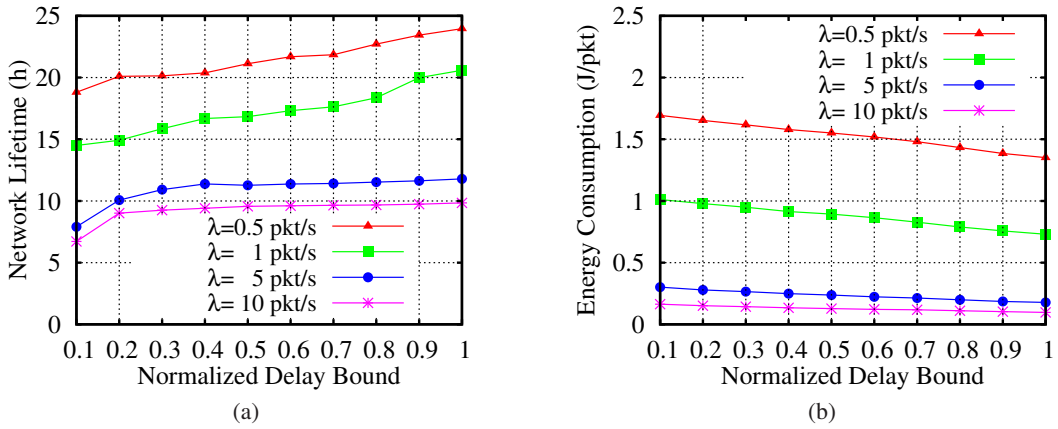


Figure 4.7: Network lifetime and per-packet energy consumption

drops due to more intensive contention among flows. Particularly, the delay meet-ratio is above 0.96 on average.

4.6.3 Impact of ZigBee Link Quality

We also study the impact of ZigBee link quality on system performance, as shown in Fig. 4.9. When link quality is low, the probability that an on-demand wakeup succeeds is also low. Thus, most of packets are eventually delivered through regular wakeup, leading to high delay and thereby low delay meet-ratio, as shown in Fig. 4.9b. Meanwhile, since nodes wake up less frequently, the network lifetime becomes longer. However, when link quality is not too low (e.g., above 0.3 in our simulation), ZPSM can quickly adjust wakeup backoff to overcome the performance degradation caused by ZigBee packet loss and achieve relatively high delay meet-ratio.

4.7 Prototyping

As a proof of concept, we implement a prototype of ZPSM system in a testbed with nine DELL D-Series laptops, each running Ubuntu Linux 10.04. Each laptop is also equipped with a D-Link WNA-2330 wireless adapter (802.11g, Atheros chipset) and a Crossbow ZigBee-enabled telosB mote. The system is implemented upon MadWiFi driver and TinyOS 2.1.1.

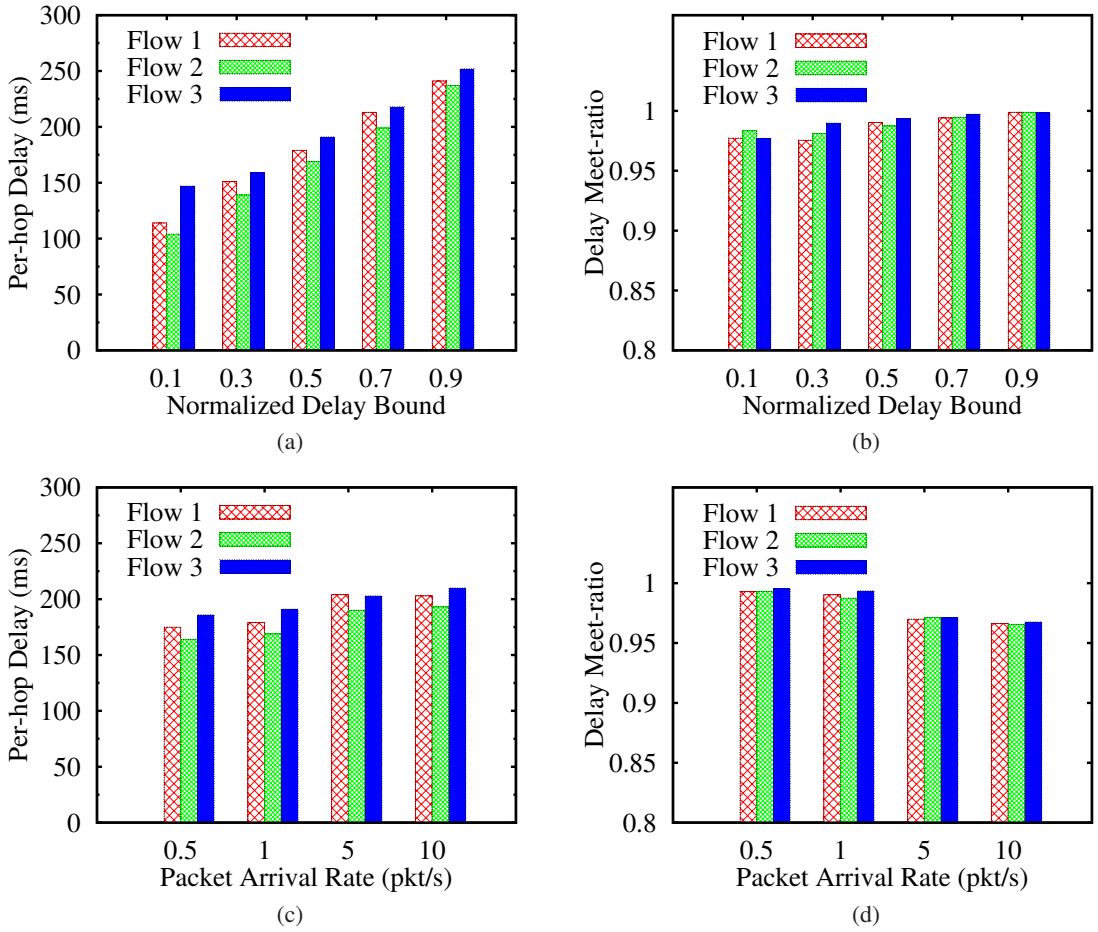


Figure 4.8: Delay performance in different scenarios

To evaluate the performance of the system when interferences between data flows are intensive, the experiment is conducted in a small indoor environment, as shown in Fig. 4.10. The nine laptops are placed in the same room, without the presence of obstacles. They form an ad hoc network, where there are two data flows. The WiFi interfaces are tuned to Channel 8 and the ZigBee interfaces are tuned to Channel 26. Unless specified otherwise, our experiment uses the same default settings as the simulation. Similar to the working presented in Chapter 3.6, WiFi energy is measured by recording the time that each node spends for transmission, reception, sleeping and idle listening. The energy consumption is computed according to Table 4.2.

In addition to ZPSM and SPSM, we also implement ODPM [14], which is a *reactive* protocol that adopts a similar idea of on-demand wakeup of WiFi interfaces except that no ZigBee interface is used.

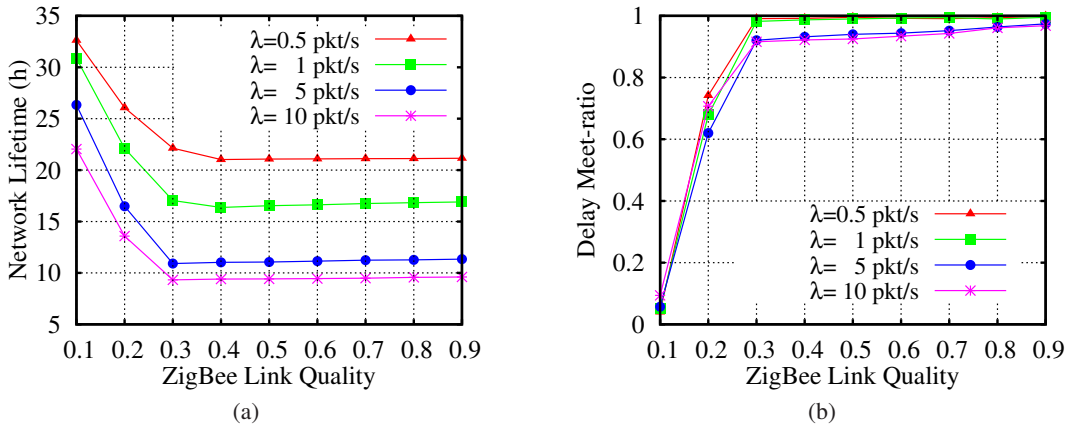


Figure 4.9: Impact of ZigBee link quality

In ODPM, all nodes are initially in SPSM. Upon reception of data packet, the node starts a keep-alive timer and switches to CAM. The keep-alive timer is refreshed whenever a data packet is received. When the timer expires, the node switches back to SPSM. By default, the length of keep-alive timer is set to current packet inter-arrival time. Besides, to make fair comparison, the BI of SPSM in our experiments is set to the maximum value such that the delay requirements are satisfied. ODPM and ZPSM are based on a SPSM with the same configuration.

4.7.1 Performance Comparison

We measure the performance of different schemes when varying delay bound and packet arrival rate. The results are plotted in Fig. 4.11. In Fig. 4.11a and 4.11c, we can see that ZPSM outperforms ODPM and SPSM, and achieves a network lifetime close to the upper bound derived in the theoretical study. With default settings, the network lifetime of ZPSM is 18.1% shorter than the upper bound, and 168.8% and 140.1% longer than ODPM and SPSM, respectively.

As for delay meet-ratio illustrated in Fig. 4.11b and 4.11d, ZPSM is slightly better than ODPM and SPSM, because it checks current delay against expected delay hop by hop to bound the end-to-end delay through on-line wakeup backoff adjustment. Besides, from Fig. 4.11d, we can see that all schemes suffer from contentions under high traffic.

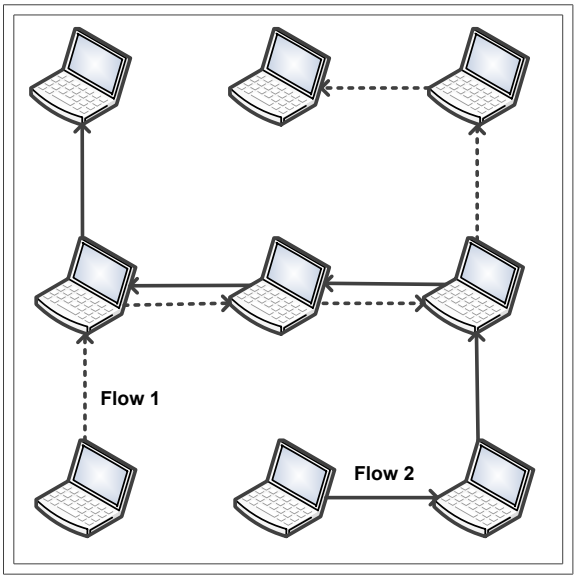


Figure 4.10: Network topology of experiment: nine laptops equipped with ZigBee-enabled sensor motes are placed on the floor of a 4m×7m space, where there are no obstacles.

From above results, we can see that ZPSM can greatly benefit *low data traffic* and/or *relatively long delay bound* applications, such as text/audio messaging/broadcasting, small UDP-based file transfer, and so on.

4.7.2 Delay CDF

The CDF of per-hop delay is also measured. In Fig. 4.12, we can see that ODPM has the best delay performance with mean $\mu = 56$ ms and standard deviation $\sigma = 54$ ms. This is because ODPM sacrifices energy to deliver low delay through intermittently staying in CAM. By leveraging on-demand wakeup strategy, ZPSM ($\mu = 131$ ms and $\sigma = 44$ ms) outperforms SPSM ($\mu = 211$ ms and $\sigma = 23$ ms).

4.7.3 Impact of Packet Arrival Model

In addition to Poisson packet arrival model, we also measured the performance of different schemes under constant bit rate (CBR) traffic. From the results shown in Fig. 4.13a and 4.13b, we can see that both ZPSM and ODPM have similar performance in Poisson and CBR models as they adopt on-demand

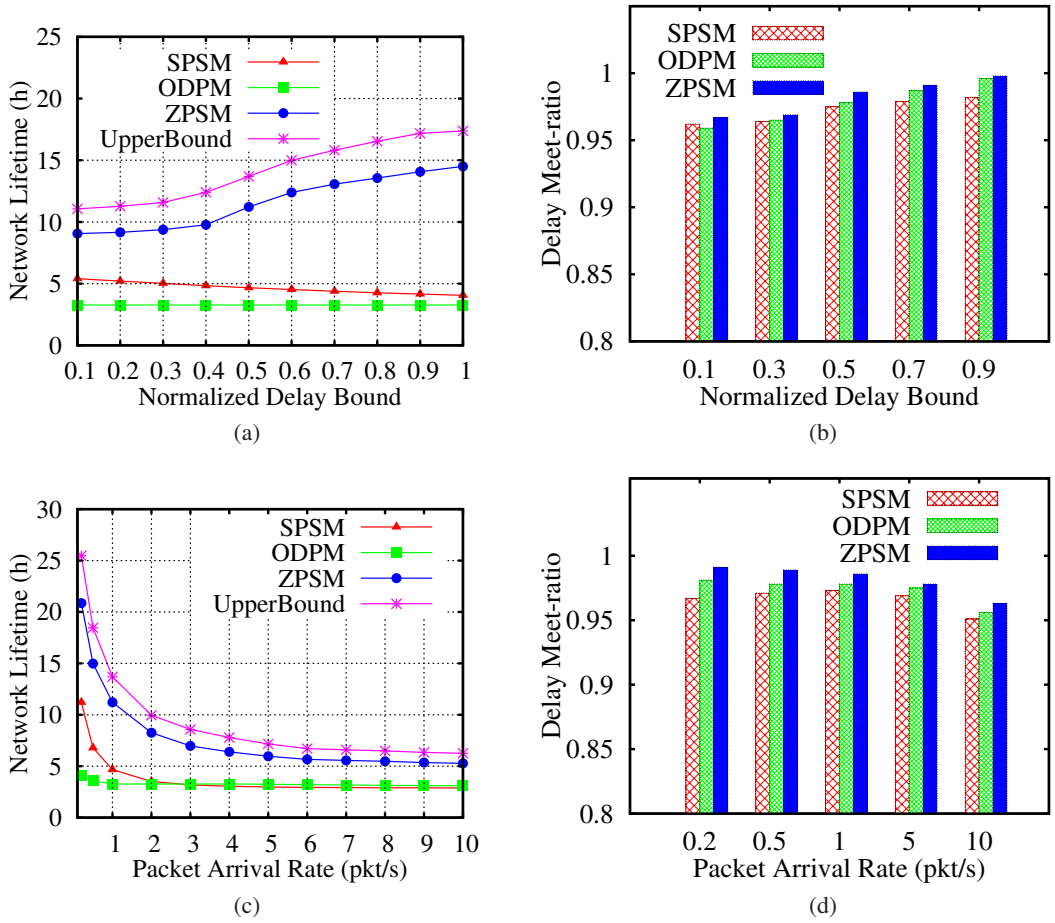


Figure 4.11: Performance comparison

wakeup strategy to adapt to traffic dynamics. However, the performance of SPSM is slightly degraded in CBR model. The reason is as follows. In Poisson model, the traffic is more irregular. Thus, with the same packet arrival rate, the probability that there is no incoming packet during a BI in Poisson model, is greater than if CBR model is used. As a result, the nodes in SPSM wake up less frequently and sleep longer when traffic follows Poisson model.

4.8 Summary

In this chapter, we proposed a ZigBee-assisted power saving management to maximize network lifetime in multi-hop ad hoc networks. The key ideas are similar to those adopted by the ZPSM sys-

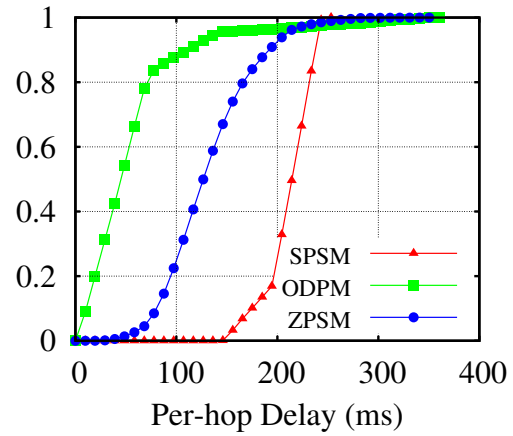


Figure 4.12: Delay CDF of different schemes

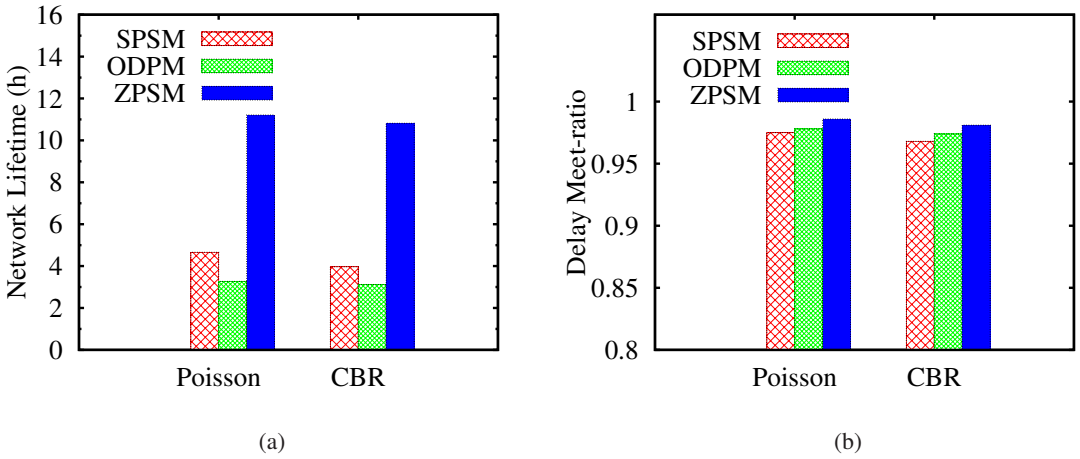


Figure 4.13: Impact of packet arrival model

tem presented in Chapter 3, though the targeted application scenarios and the network architecture are different. Results of simulation and prototype-based experiment have shown that our scheme can significantly improve network lifetime while meet the end-to-end delay requirements.

CHAPTER 5. UTILIZING ZIGBEE FOR MORE BANDWIDTH-EFFICIENT MANET

5.1 Overview

The WiFi interface is the most common interface found in mobile devices for data transmission. However, it may consume a large amount of bandwidth and energy for contention and combating collision, especially when mobile devices located in a small area (e.g., conference room, library, stadium, etc.) all have heavy traffic to transmit. To reduce contention, many protocols have been proposed. However, most of them (e.g., Overlay MAC [21], TDM MAC [22], token-passing MAC [24], etc.) require to either modify the underlying MAC protocol or introduce extra control overhead.

To address the above issue, we propose to use the ZigBee interfaces to coordinate the communication activities of nearby mobile devices in order to reduce contention and collision. The rationales behind the idea are as follows. The ZigBee interface and the WiFi interface can use different channels, and hence the coordination using ZigBee interfaces will not consume the WiFi bandwidth. As the WiFi transmission has higher rate and energy consumption than ZigBee transmission, the utilization of WiFi for large-size data transmission and ZigBee for small-size control message transmission presents an ideal, efficient resource allocation pattern. Such collaboration is possible because ZigBee may not be used frequently in the places, such as conference room, library and stadium, where WiFi traffic could be very heavy.

In this chapter, we propose a simple yet effective ZigBee-assisted WiFi transmission system for the *high traffic density* scenario. In this system, mobile devices leverage ZigBee communication to form clusters where each cluster has a cluster head and multiple cluster members that can directly communicate with the head via the ZigBee interface. According to the communication demands of individual mobile devices, members in the same cluster collaboratively run a TDMA-like protocol with

the ideal goal that, at any moment only one of them attempts to use the WiFi channel so as to eliminate or greatly reduce the contention within a cluster and thus mitigate the contention in the whole network.

The rest of this chapter is organized as follows. Section 5.2 presents the system model. Section 5.3 elaborates our proposed design. The results of comprehensive simulation and prototype implementation are reported in Section 5.4 and 5.5, respectively. Section 5.6 summarizes this chapter.

5.2 System Model

To run our proposed system, each network node (e.g., laptop) has two wireless interfaces: ZigBee and WiFi. We call such nodes **Z-WiFi** nodes. The WiFi interface is for data transmission while the ZigBee interface is for coordinating node transmission activities. Due to current popularity of the IEEE 802.11 protocol, Z-WiFi nodes may co-exist with the nodes that do not have or use ZigBee but use the **S**tandard IEEE 802.11 protocol. We call such nodes **S-WiFi** nodes.

Our design targets mainly at the scenarios where data traffic is heavy due to high node density and/or high packet transmission rate per node. The design objectives are as follows.

- *High Throughput*: By using the information gathered by ZigBee interfaces to carefully schedule the data transmission of WiFi interfaces, our design should reduce the contention among nodes and thereby increase the throughput.

- *Energy Efficiency*: Through reducing the contention experienced by the WiFi interfaces, our design should also decrease the power consumption of nodes.

- *Compatibility*: On one hand, our system should not demand changes in the existing WiFi and ZigBee standards. On the other hand, Z-WiFi and S-WiFi nodes should not harm each other, but should be in the win-win status when co-exist.

- *Fairness*: Our design should organize data transmission of WiFi interfaces in a way that the shared channel is shared relatively fairly among all nodes.

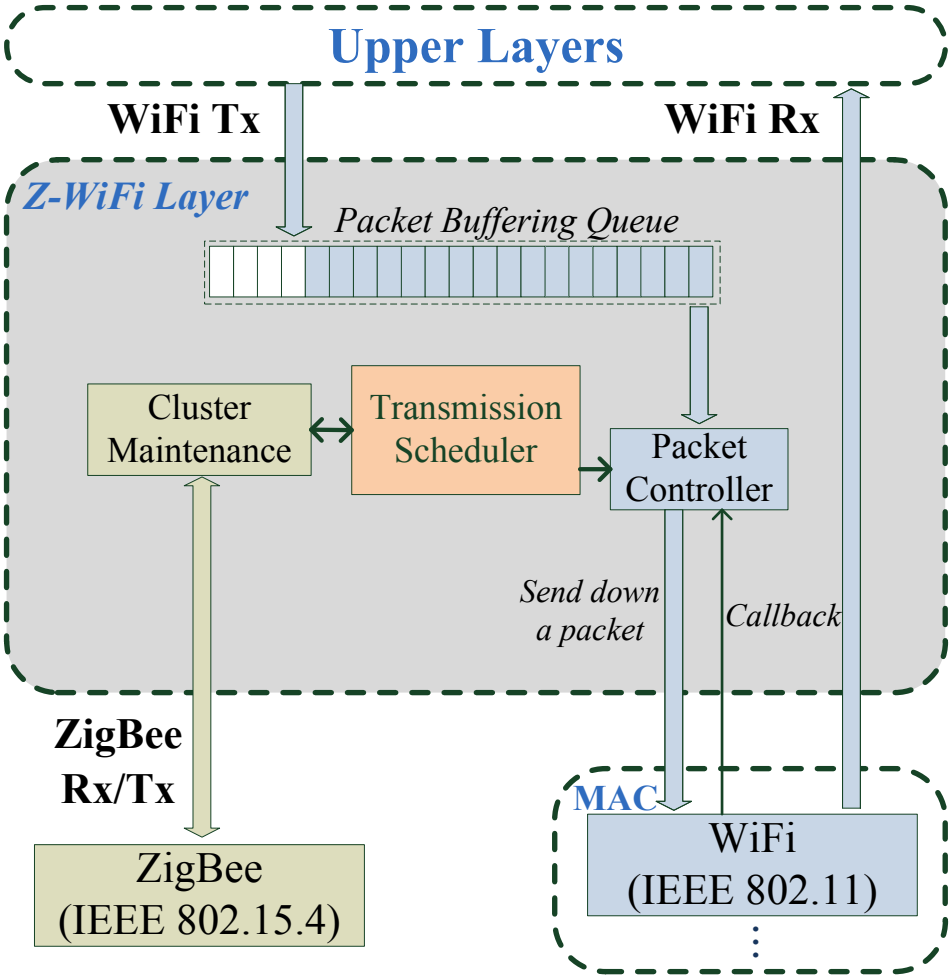


Figure 5.1: System architecture

5.3 Proposed Design

Fig. 5.3 depicts the architecture of our proposed Z-WiFi system, which is built atop WiFi and ZigBee. Thus, it is transparent to and independent of these standards. The *cluster maintenance* component works through communication over the ZigBee interface. A *packet buffering queue* is used to temporarily buffer packets from the upper layer. Through monitoring the status of the queue, packet arrival rate can be inferred, based on which the *transmission scheduler* dynamically computes the TDMA-like schedule for WiFi transmission within a cluster. The schedule is executed by the *packet controller* component which controls the timing and speed for passing packets in the *packet buffering queue* down

to the underlying IEEE 802.11 MAC layer.

In the section, we present the deign details of our proposed Z-WiFi network. Briefly, we first present the cluster formation scheme. Then, the intra-cluster and the inter-cluster coordination are elaborated, respectively. After that, heuristics is designed to deal with practical issues.

5.3.1 Cluster Formation

To facilitate the coordination of WiFi transmission for reduced contention, we propose to organize nodes that have potential need for contention into a single cluster through ZigBee communication. Based on existing cluster formation protocols [53], we propose a cluster formation scheme efficient for the scheduling of WiFi transmission.

Initially, each node marks itself as a free node (denoted as **FN**). To obtain information about neighboring nodes, each node periodically broadcasts a *beacon message*, defined as $\langle Node\_id, CH\_id, i, r_i \rangle$, via its ZigBee interface. Here, $Node\_id$ is the network-wise unique id of the sender, CH_id is the node id of its cluster head (denoted as **CH**) if the sender has joined a cluster (otherwise it is empty), and i is a cluster-wide unique *index* of the sender, assigned by the corresponding CH, when it joins the cluster. Besides, r_i is its current packet arrival rate (in the unit of *bits/second*) of the node with index i, estimated through monitoring the status of its packet buffering queue. Note that, if the sender is a cluster member (denoted as **CM**) or a FN, r_i is the packet arrival rate of its own; if it is a CH, r_i is the sum of packet rates of all nodes in its cluster. The usage of r_i and i is to be detailed later.

Based on beacon exchange, each node can maintain a neighbor information list to record the most recent information about its neighbors. If a FN has heard a beacon from one or multiple CHs, it chooses the one whose cluster has the *smallest* packet arrival rate to join. Otherwise, if a FN does not find any CH after a certain rounds of beacon exchange, it announces itself as a CH candidate by broadcasting a *formation* packet piggybacking the number of FNs in its neighborhood. When a node that is not a CH candidate first receives the formation packet, it waits for a certain period of time to overhear other possible formation packets; when the backoff expires, the candidate CH having the largest number of FNs is chosen as its CH and a *registration* packet is sent back to the candidate to join. Upon receiving a registration packet, the candidate node becomes a new CH. In response to each registration from a new

CM, the CH sends back an *index* packet, in which a cluster-wide unique index i (i is a positive integer) is assigned to the CM. Note that, the index of a CH is 0.

5.3.2 Intra-cluster Coordination for WiFi Transmission

Based on the cluster structure, WiFi transmissions of nodes within the same cluster between CH and CMs for reduced contention are coordinated. Each CM is time-synchronized with its CH.

Besides, each node measures the packet arrival rate (i.e., r_i) at its packet buffering queue, rather than at application layer. When packet buffering queue is full, any incoming packet from upper layer is dropped, which imposes a limit on the value of r_i. Hence, r_i cannot be infinitely large.

With the synchronized time reference, time is divided into frames and each frame is further sliced into slots of equal length. The length of a slot, denoted as τ_w, is the empirical time needed to send a packet through WiFi interface. The CH assigns the slots in each frame to the nodes in its cluster, according to their packet arrival rates. In the following, we show how the CH computes the WiFi transmission schedule (i.e., the slots to transmit), how it is represented and how the CH updates the schedule to its CMs by using the ZigBee interfaces.

A WiFi transmission schedule is represented and sent as a sequence of binary bits, which can be contained in the payload of a single ZigBee packet. A sequence consists of many sub-sequences of 0(s) separated by a 1. For example, sequence

$$\underline{00000}11\underline{0000}1\underline{000}1\underline{00}1\underline{000}100\cdots0$$

represents that a WiFi transmission schedule, where each frame has 17 slots, nodes with indices 0, 1, 2, 3, 4 and 5 are assigned with 5, 0, 4, 3, 2 and 3 slots, respectively. Node 0 (i.e., the CH) can perform WiFi transmission during the first 5 slots of each frame, node 1 may not exist or has no packet to send, node 2 can perform WiFi transmission during the 6th to the 9th slot of each frame, and so on and so forth. WiFi transmission schedule periodically is updated and broadcasted by the CH via its ZigBee interface as the packet arrival rate may change in each node.

Particularly, in our experiments, we set the payload size to 28 bytes, which is the default payload size used by TinyOS. Once the payload size is determined, the maximum number of slots in a frame

is also determined. We denote the maximum number of slots in a frame as f_w^{max}. Also, we use r_i to denote the packet rate of node with index i ($i = 0, \cdots, N-1$) in the cluster, recalling that each node is assigned a unique index. Let δ ($0 < \delta \leq 1$) be a predetermined system parameter. The number of slots allocated to each node i (denoted as n_i) and the actual number of slots composing a frame (denoted as f_w) is computed as follows:

$$n_i = \left\lfloor \min \left\{ \delta \cdot \frac{r_i}{B \cdot \tau_w}, f_w^{max} \cdot \frac{r_i}{\sum_{j=0}^{N-1} r_j} \right\} \right\rfloor > 0, \tag{5.1}$$

$$f_w = \sum_{i=0}^{N-1} n_i \leq f_w^{max}, \tag{5.2}$$

where B is the WiFi bandwidth. Thus, $r_i / B\tau_w$ represents the expected number of packets sent by node i. The rationale behind the slot computation is of three folds:

- For the sake of fairness, the number of slots allocated to a node is proportional to the packet arrival rate of the node while the total number of slots composing a frame should not exceed f_w^{max}.

- The ratio between the number of slots and the packet arrival rate is determined by system parameter δ. The larger is δ, the longer is a frame and the larger number of consecutive slots a node can use for WiFi transmission, and vice versa. Through our experiments, increasing δ leads to decrease in energy consumption and increase in packet delay, and vice versa. To balance energy consumption, δ is set to 0.2.

- Based on Eq. (5.1), the *clustering condition* can be defined as follows: *a FN node can join or form a cluster only if for any node i (including itself) in the resulted cluster $n_i > 0$ can be satisfied*. On one hand, a node with very few packets to send do not need to join or form a cluster and it can just use the IEEE 802.11 protocol as a FN. On the other hand, a node with a high packet rate should not be allowed to join a cluster if its joining makes any existing node in the cluster have *zero* slot to transmit. Thus, after a certain period of time, it will attempt to form a new cluster.

Ideally, each node transmits data through its WiFi interface only during the slots assigned to it, and one packet uses one slot time (i.e., τ_w) to be transmitted. It follows that n_i packets should be sent down to the underlying 802.11 MAC layer in each frame. However, in practice, this is hardly true.

To make full use of each slot, we propose to use the *callback* (i.e., notification of the completion of a packet transmission) from the underlying MAC layer to control the timing for passing packets downwards, as illustrated in Fig. 5.3. Specifically, when the scheduled transmission time (i.e.,$n_i\tau_w$) begins, the packet buffering queue delivers a packet to the MAC layer. As long as the scheduled time does not run out and there is an available packet for transmission, a packet will be pushed down to the MAC layer once the callback of previous packet is received.

5.3.3 Inter-cluster Dynamics for Dealing With Mobility

Due to mobility, a CM may move out the range of its CH and join another cluster; a FN may discover a CH and join the cluster headed by that CH; a CH may move into the range of another CH and their clusters may be merged to reduce the number of co-existing clusters and hence inter-cluster contention.

5.3.3.1 Cluster Switching

When a CM with index i finds it has moved out of the ZigBee communication range of its CH, i.e., failing to receive beacon from its CH for a certain time, it attempts to discover nearby CHs by overhearing beacons. If it finds some CHs, it joins the cluster that has the lowest overall packet arrival rate. If no CH is found in vicinity, it becomes a FN, which can either join another cluster, or form its own cluster. Note that, if a CH fails or is turned off, its CMs will not be able to receive beacon messages from it, in which case they will react as if they have moved out of the communication range of the CH and perform cluster switching as depicted above.

5.3.3.2 Cluster Joining

When a CM or CH becomes a FN, it first tries to join other cluster by turning on its ZigBee and listening for a certain time. If it finds some CHs in the vicinity, a registration packet is sent. Upon

receiving the registration packet, the CH acknowledges that node by replying an *index packet* containing a unique index (typically the *smallest unused* index in the cluster) assigned to that node, if the clustering condition (See Eq. (5.1)) can be satisfied. Once the index packet is successfully received by the FN, it becomes a CM of that cluster. If no CH is found, it starts the cluster formation process as described in Section 5.3.1, if the clustering condition can be satisfied.

5.3.3.3 Cluster Merging

To dynamically minimize the cluster density and hence reduce inter-cluster contention, cluster merging is proposed as follows. As CHs are always awake, they may overhear WiFi transmission schedule packets from nearby clusters. When a CH (CH1) overhears a schedule packet from another CH (CH2), it checks if it can cover more than half of CMs of CH2. If so, merging process will be conducted through the negotiation between these two CHs. As a results, the nodes that are in the cluster of CH2 and covered by CH1 are merged into the cluster of CH1, while the rest of CMs become FNs, which with either join other clusters or form a new cluster later.

5.3.4 Practical Issues

5.3.4.1 Turning on/off ZigBee

Our system is designed mainly to improve WiFi performance in high-contention scenarios, and the IEEE 802.11 protocol can already achieve the optimal throughput when the contention is low. To avoid unnecessary control overhead, we propose a simple heuristic parameter γ for turning off ZigBee interfaces of Z-WiFi nodes when the contention is low and turning on them when the contention is high. The nodes without using ZigBee interface run the IEEE 802.11 protocol.

Specifically, each node records transmission time (i.e., duration from the arrival of a packet to the reception of corresponding ACK) of the most recent outgoing packets. Let T_{pkt} be average transmission time, then

- ZigBee is turned off, if $T_{pkt} < 0.5 \times \gamma\tau_w$;
- ZigBee is turned on, if $T_{pkt} > 1.5 \times \gamma\tau_w$.

$\gamma\tau_w$ represents the expected packet delivery delay when system throughput is saturated. The above conditions attempt to find a range, beyond whose upper bound the ZigBee interface should be turned on to improve performance and below whose lower bound ZigBee interface can be turned off to save energy. Besides, the difference between the upper bound and the lower bound is $\gamma\tau_w$, which is designed to prevent frequent switching caused by random vibration of the network traffic.The selection of γ is to be studied in Section 5.4.1.

5.3.5 Co-existence of Z-WiFi and S-WiFi

In this subsection, we first present an analytical model for our proposed system by taking account of *penetration rate* (i.e., the proportion of Z-WiFi nodes in a system), denoted as λ. Then, based on the model, we enhance our proposed system to be able to operate with the co-existence of S-WiFi nodes.

5.3.5.1 Model

As outlined in [54,55], our model is also built upon the Markov chain model for analyzing saturation throughput. Similarly, we assume that a fixed number (n) of nodes (including λn Z-WiFi nodes and $(1-\lambda)n$ S-WiFi nodes) contend with each others and each node always has a packet available for transmission. Let W denote the minimum contention window, m represent maximum backoff stage (i.e., $2^m W$ is the maximum contention window) and M be the maximum retransmission count (e.g., $m = 4$ for data frame and $m = 7$ for RTS frame in the IEEE 802.11b/g protocol).

Suppose the probability that a packet sent by a node collides with others is p. According to [54], the probability (η) that each node transmits a packet in a randomly chosen slot time can be computed as a function of p and W,

$$\eta = \phi(p, W) = \begin{cases} \frac{2(1-2p)(1-p)}{W \cdot B(m+1) + A(m+1)} & \text{if } m \leq M \\ \frac{2(1-2p)(1-p)}{W \cdot B(M+1) + A(m+1) + W \cdot 2^M p^{M+1} A(m-M)} & \text{if } m > M \end{cases}, \tag{5.3}$$

where $A(x) = (1-2p)(1-p^x)$, $B(x) = (1-p)(1-(2p)^x)$ and $p = 1-(1-\eta)^{n-1}$.

The above model is used to analyze the saturation throughput of homogeneous WiFi systems, where all network nodes have exact the same configurations. In our system, we propose to allow different

settings for the WiFi interfaces on Z-WiFi and S-WiFi nodes. Thus, the model can be extended as follows.

Virtually, each Z-WiFi cluster is represented as a *super* Z-WiFi node that can transmit all the time. Let W_z and W_s be the minimum contention window used by Z-WiFi and S-WiFi node, respectively. Let C denote the expected number of clusters. Then, we can have

- the probability that a super Z-WiFi node (i.e., a Z-WiFi cluster) transmits in a randomly chosen slot time is

$$\eta_c = \phi(p_c, W_z), \tag{5.4}$$

  where

$$p_c = 1 - (1 - \eta_c)^{C-1}(1 - \eta_s)^{(1-\lambda)N}; \tag{5.5}$$

- the probability that a S-WiFi node transmits in a randomly chosen slot time can be computed as

$$\eta_s = \phi(p_s, W_s), \tag{5.6}$$

  where

$$p_s = 1 - (1 - \eta_c)^{C}(1 - \eta_s)^{(1-\lambda)n-1}. \tag{5.7}$$

5.3.5.2 Achieving Fairness Through Adjusting W

Formulation of Fairness From the above-presented model, it is easy to observe that a super Z-WiFi node has the same chance to transmit as a S-WiFi node (i.e., $\eta_c = \eta_s$) if $W_z = W_s$. Suppose there are k nodes in a cluster. Then, the average transmission probability for a Z-WiFi node is η_s/k, which means S-WiFi nodes have an advantage over Z-WiFi nodes in accessing the channel. To deal with this problem and make Z-WiFi nodes have equal chance to access the channel, we propose to dynamically adjust the minimum contention window W_z of Z-WiFi nodes to achieve fair sharing of the channel with S-WiFi nodes.

Assume that all S-WiFi nodes use default the contention window (i.e., 31 for 802.11b and 15 for 802.11g), denoted as W_{def}. That is, $W_s = W_{def}$. We define that Z-WiFi and S-WiFi nodes fairly

access the channel if

$$P_c = k \cdot P_s, \tag{5.8}$$

where P_c and P_s are respectively the probability that a super Z-WiFi node and a S-WiFi node can *successfully* transmit in a randomly chosen slot time. Specifically, we have

$$\eta_c(1-\eta_c)^{C-1}(1-\eta_s)^{(1-\lambda)n} = k \cdot \eta_s(1-\eta_c)^{C}(1-\eta_s)^{(1-\lambda)n-1} \tag{5.9}$$

Combining Eq. (5.4), (5.5), (5.6), (5.7) and (5.9), the optimal value of W_z can be only found *numerically* given a particular set of system parameters, similar to [54, 55].

Heuristics Due to the complexity of solving the above equations, we design a heuristics to choose W_z based on a simplified model, where we assume $m = 0$ (i.e., only the first transmission of a packet is considered). Hence, from Eq. (5.4) and (5.6), we have

$$\eta_c = \frac{2}{W_z + 1} \qquad \text{and} \qquad \eta_s = \frac{2}{W_{def} + 1}. \tag{5.10}$$

Combining these two equations with Eq. (5.9), we can get

$$W_z = \frac{1}{k}(W_{def} - 1) + 1, \tag{5.11}$$

where $0 < 1/k \leq 1$. Since $W_{def} - 1 > 0$, W_z monotonously increases as $1/k$ becomes larger. Thus, when $1/k = 1$ (i.e., only one node in the cluster), $W_z = W_{def}$ which is the same as S-WiFi nodes; when $0 < 1/k < 1$ (i.e., more than one node in the cluster), $1 < W_z < W_{def}$ which enable Z-WiFi nodes to compete with S-WiFi nodes. The value of W_z for a Z-WiFi node is dynamically chosen as follows.

- According to the neighbor information, each Z-WiFi node computes the expected value of minimum contention window $E[W_z]$ by

$$E[W_z] = (1/k)^{\omega}\,(W_{def} - 1) + 1, \tag{5.12}$$

where ω is a *positive* system parameter to adjust the changing rate of W_z as cluster structure alters. Note that $0 < (1/k)^{\omega} \leq 1$ as well.

- If $\lfloor E[W_z] \rfloor = \lceil E[W_z] \rceil$, then $W_z = \lfloor E[W_z] \rfloor = \lceil E[W_z] \rceil$
- If $\lfloor E[W_z] \rfloor \neq \lceil E[W_z] \rceil$, then
  - $W_z = \lfloor E[W_z] \rfloor$ with the probability $\lceil E[W_z] \rceil - E[W_z]$;
  - $W_z = \lceil E[W_z] \rceil$ with the probability $E[W_z] - \lfloor E[W_z] \rfloor$.

By appropriately choosing the value of ω, Z-WiFi nodes can achieve no worse performance than S-WiFi nodes as to be shown in Section 5.4.1.

Remarks: The above heuristics is purely based on the *local* information (i.e., k), and thus is efficient to be implemented. Moreover, it enables fair access of the channel not only between Z-WiFi nodes and S-WiFi nodes but also among Z-WiFi clusters of different sizes. Particularly, the heuristics will force a Z-WiFi cluster of more nodes to use a smaller W to compete with clusters of fewer nodes, so as to achieve fairness for the Z-WiFi nodes in different clusters.

5.4 Simulation

To evaluate our proposed system in a large-scale network, we simulate the system with ns2 simulator. In the simulation, the following major metrics are studied:

- *Network throughput* (Mb/s) is the total amount of data successfully transmitted (i.e., ACKed at sender side) in the network. To measure the throughput, each node runs an application which keeps sending UDP packets and by default totally all these nodes generate the data input with an average rate of 20.4Mb/s (i.e., 22 packets/s at each node on average). All the packets have maximum payload size.
- *Energy consumption* (J/Mb) is computed as the total amount of energy consumed by all network interfaces of all nodes divided by the number of Mbs of data that has been successfully transmitted. The energy consumed by the WiFi interface is measured according to the specified power consumption rate of SX-SDWAG 802.11g wireless module [43] (i.e., 1047mW for transmission,

513mW for reception and 420mW for being idle) and the power consumed by the ZigBee interface is measured according to the specified power consumption of CC2420 RF transceiver [56] (i.e., 52.2mW for transmission, 56.4mW for reception, 1.28mW for being idle, 0.06μW for sleeping and 0.06mW for transition).

- *Throughput fairness* is measured with respect to the fairness index (FI) [57], which is defined as $FI_{tp} = \frac{\mu(\chi)}{\mu(\chi)+\sigma(\chi)}$, where $\mu(\chi)$ and $\sigma(\chi)$ are the mean and the standard deviation of χ at all network nodes. χ is the ratio of throughput to input. Obvious, FI_{tp} is between 0 and 1. The more closer FI_{tp} approaches 1, the better is the fairness.

Unless otherwise specified, our simulation use the settings shown in the table below. Also, we adopt the random waypoint mobility model, where the pause time is fixed to 20s and the maximum speed is 2m/s. Besides collision-caused drops, each node intentionally drops 2% incoming packets on ZigBee communication to simulate the packet loss due to interference from WiFi. The default IEEE 802.11g protocol is used.

Number of nodes	50	Network scale	100m $\times$ 100m
Range of WiFi (R_w)	120m	Range of ZigBee (R_z)	60m
Simulation time	1 hour	WiFi slot length (τ_w)	0.001s
ZigBee on/off parameter (γ)	15	Packet buffer size	50 packets

Table 5.1: Simulation settings

5.4.1 Comparing with S-WiFi system and studying parameter γ

To find the best time to turn on ZigBee so as to maximize the performance, we compare Z-WiFi system, configured with four different values of γ (i.e., 1, 5, 15 and 25), with S-WiFi system.

From Fig. 5.2a, we can see that when network input is below 17Mb/s, S-WiFi system can almost deliver all incoming packets. When input is beyond 17Mb/s, S-WiFi nodes reach the maximum throughput. At this time, ZigBee interface of Z-WiFi nodes should be turned on to assist WiFi transmission. As shown in Fig. 5.2a and 5.2c, $\gamma = 5, 15$ or 25 can precisely render ZigBee turned on at the right time. This is because, due to accumulated waiting delay in the packet buffer queue, packet transmission delay rises up drastically (from less than one millisecond to more than hundreds of milliseconds) once

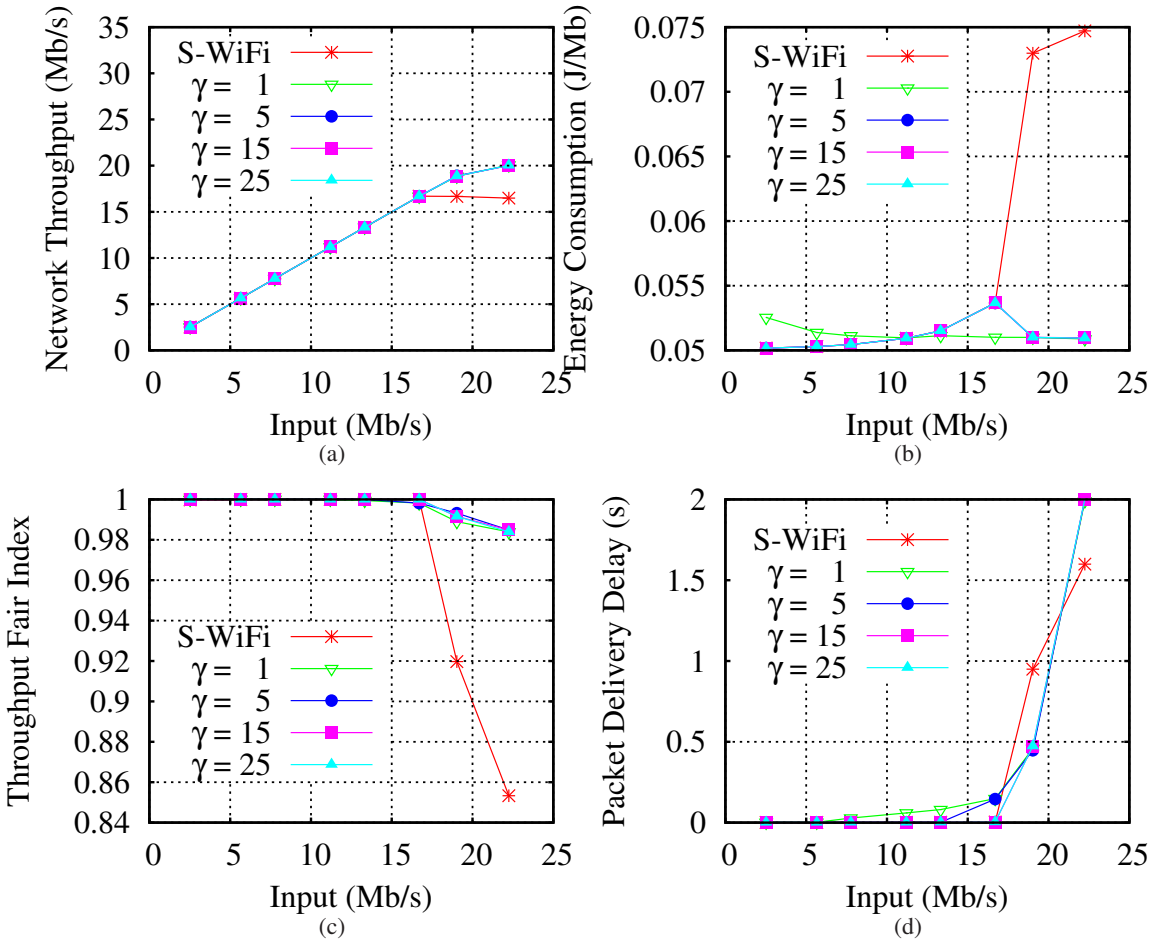


Figure 5.2: Choosing parameter γ by comparing with S-WiFi

S-WiFi system gets saturated. Thus, large values of γ (e.g., $\gamma > 5$) can work appropriately. Particularly, when ZigBee interface is turned on (i.e., input exceeds 17Mb/s), energy consumption drops rapidly, as shown in Fig. 5.2b, which shows that our proposed system can save energy.

When $\gamma = 1$, ZigBee interface is turned on when network input (i.e., contention) is low. At this time, our protocol cannot help, as the S-WiFi system has already achieve the optimal throughput. Hence, the overhead introduced for ZigBee communication makes Z-WiFi systems consume more energy.

In addition, we also measure average packet delivery delay from application layer, as illustrated in Fig. 5.2d. From the results, setting γ to 15 or 25 can guarantee that Z-WiFi system can achieve no longer packet delivery delay than S-WiFi system when input is below 21Mb/s. When input is above

21Mb/s, our system also becomes saturated and thereby packet delivery delay increases. Note that the packet delivery delay of Z-WiFi system is longer than that of S-WiFi system only when the throughput of Z-WiFi is higher than S-WiFi.

To summarize from the above results, *our proposed system can improve the network throughput by 18%, reduce the energy consumption by 32% and provide much better fairness, when the network traffic density is high.*

5.4.2 Co-existence of S-WiFi and Z-WiFi systems

5.4.2.1 Choosing contention window parameter ω

Recall that, we propose parameter ω to dynamically adjust W_z of Z-WiFi nodes to compete with the co-existing S-WiFi nodes. Fig. 5.3a illustrates the impact of ω on overall throughput (including both Z-WiFi and S-WiFi nodes) with different penetration rate. According to the results, the throughput increases as ω becomes larger. This is because using small W can make full use of the channel when contention is very low. When ω is beyond 2.0, the throughput stays approximately constant as the average value W approaches its lower bound.

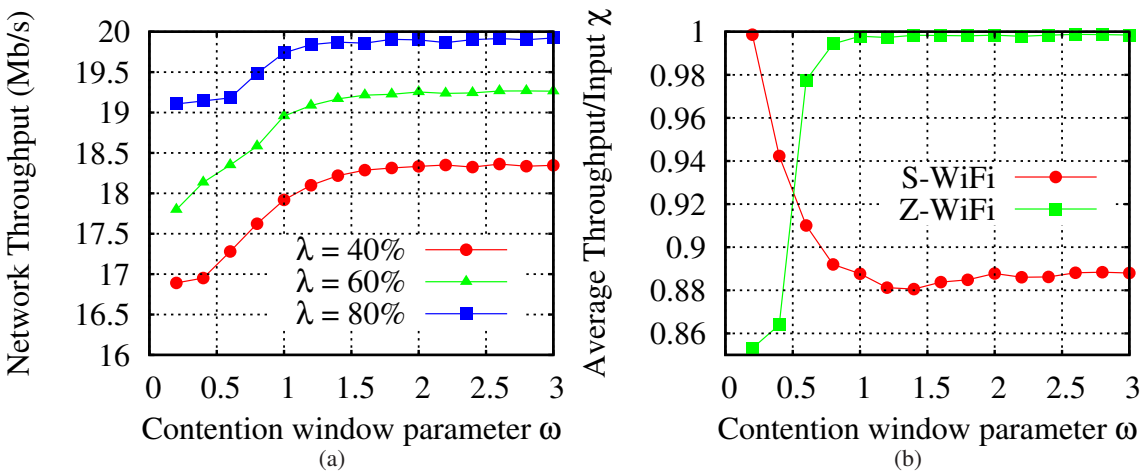


Figure 5.3: Impact of parameter ω on throughput

By fixing $\lambda = 60\%$, we further measured the average throughput/input ratio (χ) of Z-WiFi and S-WiFi nodes, respectively, as ω varies. The result is shown in Fig. 5.3b. Generally, as ω increases,

Z-WiFi node gets more chance to transmit while S-WiFi node gets less chance. When $\omega = 0.5$, Z-WiFi nodes have the similar χ to S-WiFi, which means Z-WiFi and S-WiFi nodes fairly share the channel; when ω is beyond 0.5, Z-WiFi nodes outperform S-WiFi nodes. This also holds for the cases of other penetration rates. To obtain the maximum overall throughput, we choose $\omega = 2$ in our simulation.

5.4.2.2 Impact of penetration rate on performance

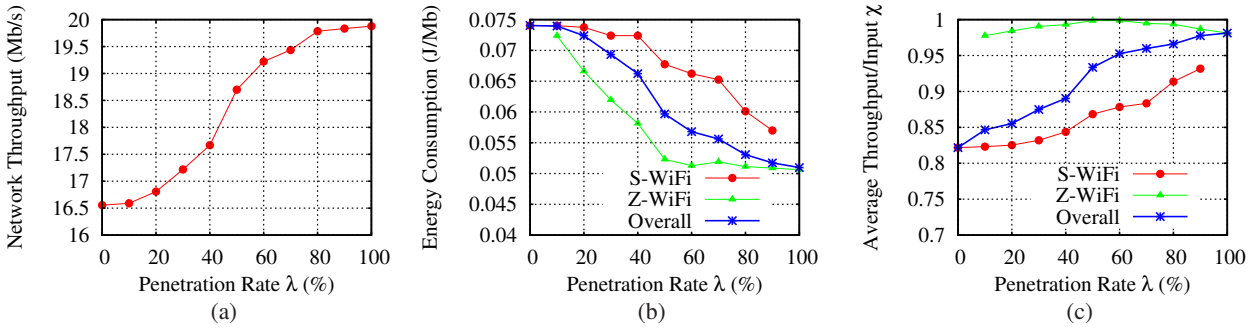


Figure 5.4: Impact of penetration rate λ on performance

Fig. 5.4a gives the overall network throughput, which increases as λ approaches 100%. Fig. 5.4c and Fig. 5.4b show the average throughput/input ratio (χ) and energy consumption of Z-WiFi nodes and those of S-WiFi nodes, respectively, as λ changes. Besides, the corresponding overall averages are also shown. From these figures, we can see that both node throughput ratio and energy consumption of Z-WiFi and S-WiFi nodes are improved, as more and more nodes use our proposed system. This is because by using our system the network contention is reduced, which can also benefit the co-existing S-WiFi nodes simultaneously. Similar trends can be seen from the overall performance.

In Fig. 5.4c, as λ increases, χ of Z-WiFi nodes first reaches 1. Then, it slightly drops below 1. This is because when λ becomes larger, the size of each cluster also increases, leading to more schedule inconsistencies (due to emulated packet loss) within the same cluster. As a result, more transmission slots of the nodes in the same cluster may overlap, resulting in more collisions. Thus, χ slightly drops. Moreover, in Fig. 5.4b, the energy consumption of Z-WiFi nodes decreases very slowly when λ becomes larger. This is because the energy has to be consumed by the regular RTS/CTS of the underlying IEEE 802.11 protocol, which sets an upper bound for the energy saving that can be

accomplished.

As a conclusion of the above analysis, *using our proposed system can benefit all network nodes (both Z-WiFi and S-WiFi nodes) in terms of both throughput and energy. As penetration rate grows, the performance of both individual nodes and the overall networks is improved.*

5.4.3 Performance with Different Network Scale

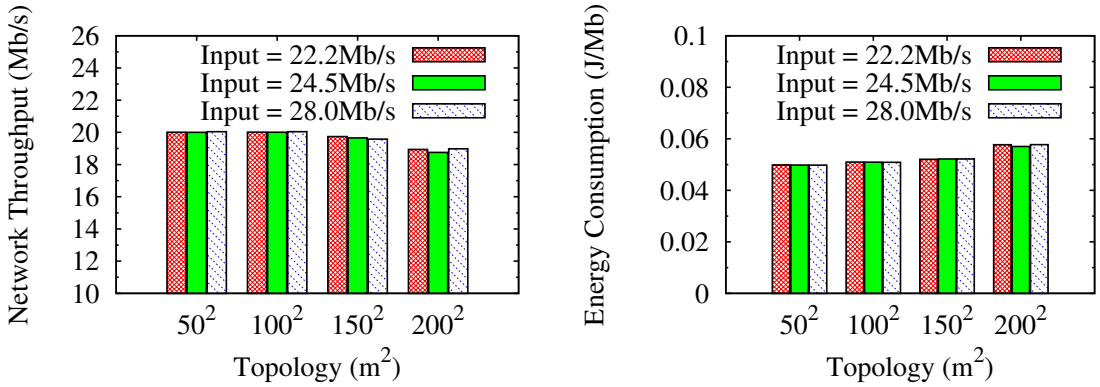


Figure 5.5: Impact of network scale on performance

Fig. 5.5 shows how our system works with different network scale. Generally, the throughput slightly decreases as the scale of the network becomes larger, due to the number of clusters increasing. When the number of clusters within WiFi transmission range increases, contention gets more severe, which degrades the performance. However, the number of clusters will not become too large, since cluster merging mechanism is applied, which can ensure the number of interfering clusters close to $\lceil R_w^2/R_z^2 \rceil$ (e.g. 4 under our simulation). For energy consumption illustrated in 5.5b, more clusters consume more energy in transmission coordination and cluster maintenance.

5.4.4 Impact of ZigBee Packet Loss on Performance

Apart from random collision-caused packet loss, we also study the packet loss due to other environmental phenomena (e.g., interference, obstacle, multipath, etc.). Thus, we conduct a simulation by

varying packet loss ratio from 2% to 20%. As shown in Fig. 5.6, our performance degrades slightly as loss ratio gets larger. For throughput, it is because of the insufficient utilization of channel caused by increasing delay or error in updating WiFi transmission schedule. The energy consumption increases mainly because of the increased energy consumption for contention caused by schedule inconsistencies, resulted from packet loss.

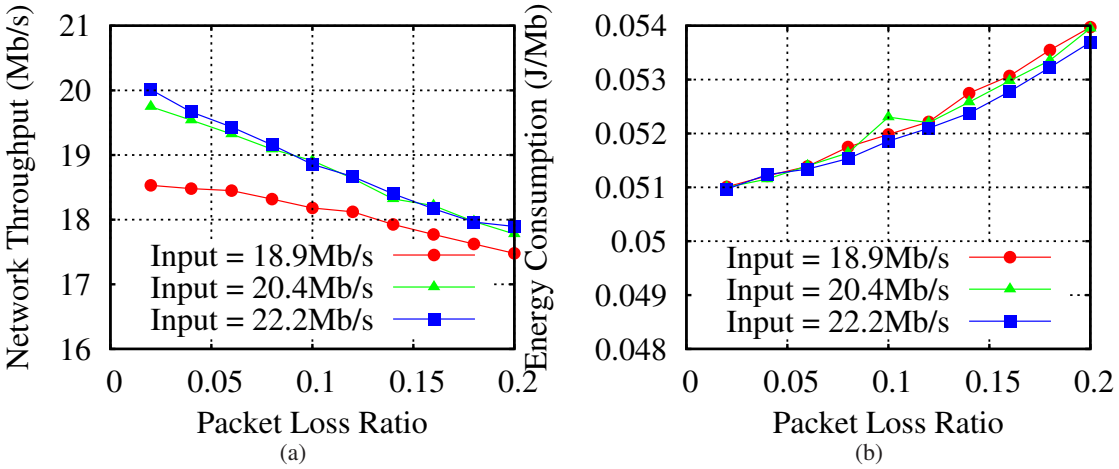


Figure 5.6: Impact of ZigBee packet loss on performance

5.5 Implementation

5.5.1 Prototyping

As a proof of concept, we implement a prototype of our proposed system. We build a testbed with 10 DELL D-Series laptops (called *nodes* hereafter), each running the Ubuntu Linux 8.10 (kernel 2.6.27-17-generic). Each node is also equipped with a D-Link WNA-2330 Wireless G Notebook Adapter (108Mbps, 802.11g, Atheros chipset, PCMCIA) and a Crossbow telosB mote (i.e., ZigBee interface). Note that *the wireless adapter is built with the state-of-the-art technology, which can deliver higher throughput than standard 802.11g devices.* The scheduling of WiFi transmission is implemented upon MadWiFi [45], an open-source driver for Atheros chipset-based 802.11 Wireless LAN devices. The prototyped ZigBee communication is implemented upon TinyOS 2.1.1 platform, where 10 nodes form a cluster. The WiFi interfaces of all nodes run in the ad hoc model and are tuned to Channel 3, and

the ZigBee interfaces are tuned to Channel 26; thus, the interference between them is small. Besides, the implementation of transmission scheduling is based on *software timer* provided by Linux kernel, which can allow a minimum granularity of 1μs.

5.5.2 Experiment Results

Experiments have been conducted on the prototyped system to evaluate the feasibility and the performance of our designed system. For comparison, two sets of experiments are conducted by running the IEEE 802.11 protocol and our proposed system, respectively. Through the experiments, we measure the maximum network throughput as the number of nodes increases from 2 to 10. To measure the maximum throughput, each node generates UDP traffic of 34.8 Mb/s. Each packet has a payload of 1450 bytes, which makes the overall packet to exactly fit into a single MAC-layer frame. The duration of each experiment run is 5 minutes. The experiment is conducted three times. Besides, $n_i = 10$ and $\tau_w = 0.001$s.

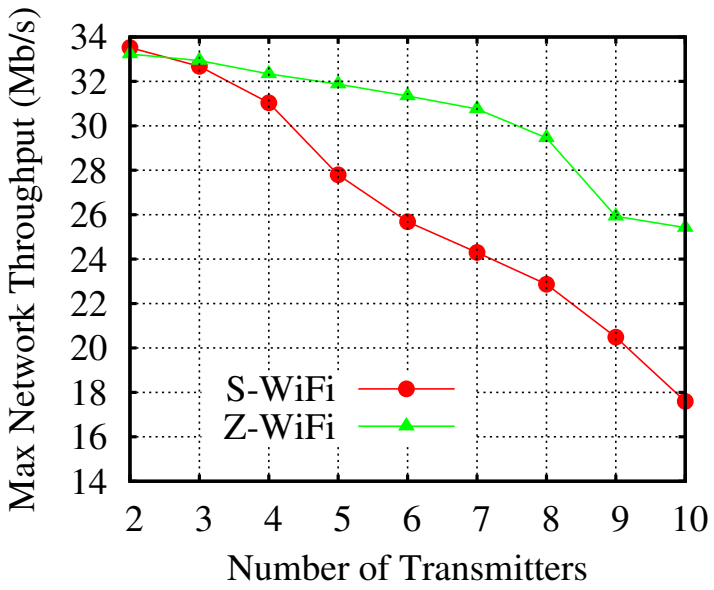


Figure 5.7: Maximum network throughput

The experiment results are shown in Fig. 5.7. In general, compared with the IEEE 802.11 protocol, our proposed system can improve the network throughput significantly. Particularly, when the number of involved nodes reaches 10, the improvement of throughput can be as high as 49.1%. As expected,

our proposed system outperforms the IEEE 802.11 protocol when the number of transmitters is large (e.g., more than 4 nodes in our experiment). As that number keeps increasing, the difference becomes more significant because the IEEE 802.11 protocol suffers from severe contention and the throughput drops fast.

Moreover, the standard deviation (STDV) of throughput among different nodes is also measured, as shown in the table below. From the results, we can see that using our proposed system introduces much lower throughput STDV, which indicates better throughput fairness.

# of transmitters	Throughput STDV of S-WiFi	Throughput STDV of Z-WiFi
4	1.1016	0.1780
6	0.8016	0.1281
8	0.7698	0.1775

Table 5.2: Experiment results

Through the experiments, our proposed system has been shown to be able to improve throughput significantly and provide fair sharing of bandwidth.

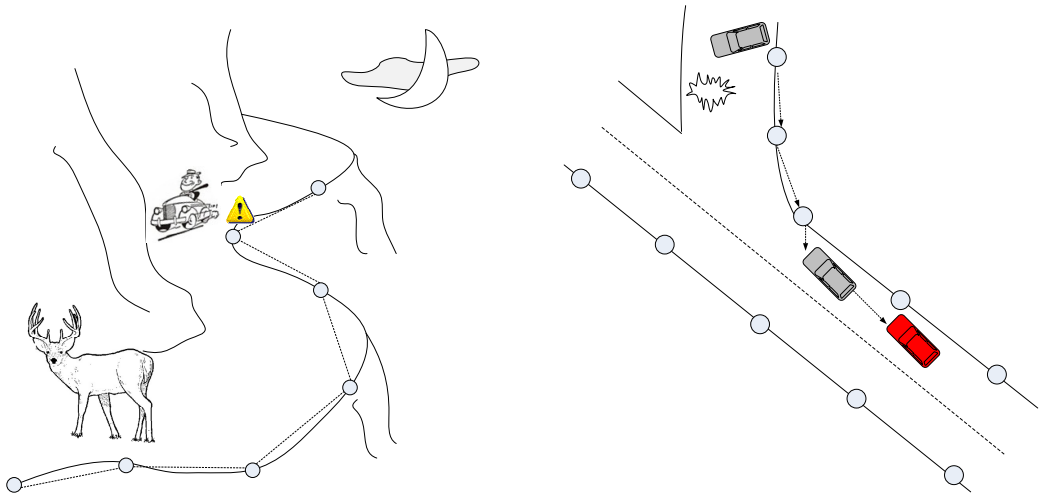
5.6 Summary

In this chapter, we have proposed a simple yet effective system for ZigBee-assisted WiFi transmission. Mobile devices form clusters. Coordinated through ZigBee interfaces, members in each cluster take turns to transmit, resulting in reduced contention and collision. Results of experiment and simulation have shown that our proposed design can improve throughput, power consumption and fairness, among which the improvement of throughput is most significant. In other words, with our design, mobile devices can more efficient use of the limited bandwidth.

CHAPTER 6. UTILIZING ZIGBEE FOR MORE RESOURCE-EFFICIENT VANET

6.1 Overview

Driving is an indispensable part of the life of many people; meanwhile, driving accidents have been a big threat to the lives, health and wealth of the people. The past years have witnessed substantial efforts on improving driving safety. Among them, the most prominent technological one might be the emerging vehicular ad hoc network (VANET) and the safe driving-targeted applications built atop the VANET. The VANET is composed of highly-mobile vehicles and sparsely-deployed roadside stations, each equipped with wireless communication devices and optionally with sensing devices. Wireless communication can be conducted between vehicles and/or between vehicles and roadside stations. On top of the VANET, applications have been developed to collect, process, share and deliver real-time information about road conditions.



(a) (b)

Figure 6.1: Examples of integrated VANET-WSN

These systems sometimes help in accident prevention, but they are not always effective since the underlying VANET does not provide guaranteed real-time detection of road conditions or communication connectivity. Firstly, the VANET only opportunistically monitors road conditions due varying traffic density. That is, only when a vehicle or a roadside station detects or is notified of some conditions, can the information be shared within the VANET. For example, a driver driving at night may not know about a deer roaming on the road ahead because no vehicle or roadside station detects or is notified of the condition. Secondly, the VANET can be disconnected due to high mobility and unpredictable movements of vehicles and the sparse deployment of expensive roadside stations. If the VANET is disconnected, critical information about road conditions known by one partition of the VANET cannot be shared timely with vehicles that need to know it but are in other partitions.

Deploying more roadside stations appears to be a solution. This, however, may significantly increase the investment cost; also, lack of power supply is a big obstacle to do so in rural areas [58]. Wide area wireless networks (such as cellular networks) could be used to connect disconnected segments of the VANET. This approach may achieve communication connectivity, but it does not solve the problem of lacking guaranteed real-time sensing of road conditions.

Towards addressing the above problems, we propose to integrate the VANET with the inexpensive ZigBee-enabled wireless sensor network (WSN) to provide timely detection of road conditions and to help connect partitioned segments of the VANET. Wireless sensor nodes, for example, MicaZ motes [59], are much *cheaper* than roadside stations of current VANETs. Besides, some inexpensive, low-power and small-size sensing modules, for example, the WiEye passive infrared sensors [60], have been commercialized and can be installed on the motes to sense road conditions with low cost. These sensor nodes can be deployed along roadside with higher density than current roadside stations to form a connected network together with the VANET. The sensor nodes can sense the road conditions, collect and process the sensing data to find out information useful for safe driving, and deliver the information to vehicles that need it. The sensor nodes also can buffer the safety-related information generated by vehicles, and forward the information to vehicles in different partitions of the VANET. Due to low cost of sensors, such integrated system is much more investment-efficient than traditional VANET which relies on expensive roadside stations. Meanwhile, by taking advantage of the communication capabil-

ity of sensors (typically equipped with ZigBee interfaces), message propagation among vehicles can be made much more efficient to guarantee driving safety.

Following are some examples showing that deploying WSNs can greatly help in preventing road accidents:

- **Example I.** Deploying WSN along rural roads can help prevent vehicle-animal collision accidents. As shown in Fig. 6.1a, the WSN nodes can detect a deer roaming on the road and propagates the information within the nearby area. Approaching vehicles will get the warning beforehand. The advantage brought by the deployment of WSN is significant. It may help to avoid 1.5 million vehicle-deer collisions happening every year (according to auto insurer State Farm) which result in about 150 deaths and $1.1 billion losses [61].

- **Example II.** Fig. 6.1b shows that, bad road conditions (e.g., slippery surface) detected by an isolated vehicle can be told to nearby roadside WSN nodes, and the WSN nodes can then collaborate with each other to propagate the information to other vehicles approaching this dangerous area. Note that, this cannot be accomplished if only VANET can be used since the VANET is not connected.

To realize the proposed VANET-WSN system, several important issues should be investigated. Firstly, the system should be really *viable* in the real scenarios. The impact of interference, noises and other environmental factors on the performance of the system should be investigated. Secondly, the system should be *highly scalable*, considering the large scale of highway system in the world. As the scale of deployment increases, the difficulty in deploying and maintaining the system should not increase much, and the quality of service and the energy efficiency of the system should remain stable. Thirdly, the system should be *flexible* to changes in the real world. WSN nodes may fail or lose time synchronization, the highways may be extended or reshaped, and traffic pattern may change from time to time. It is desired that the deployment and the working parameters of VANET-WSN system can be adjusted with low overhead as the above changes happen. Fourthly, *energy efficiency* should be maximized for the roadside WSN. Although WSN nodes can be deployed and redeployed by humans and their batteries can be replaced manually when necessary, it is still important to minimize the energy

consumption and maximize the network life time to reduce energy and maintenance costs. Finally, satisfactory *quality of service* should be attained. Dangerous road conditions should be detected and the information about the dangers should be delivered to related vehicles, both in a timely fashion, to ensure driving safety.

Towards tacking the above issues, this research makes the following major contributions:

- We adopt the idea of group-based modular design to achieve *scalability* and *flexibility*. In our design, the roadside WSN is made up of sensor groups. Each group works autonomously and asynchronously, and neighboring groups interact with each other through a gateway node shared by them. Deployment or redeployment of a group does not affect others; topology and working parameter adjustments conducted within each group do not affect others, either. The modularity nature of the network and the autonomy nature of each module enable easy deployment, extension and reconfiguration. Moreover, our design takes the dynamics of traffic flows and the particularity of sensor distribution into full consideration, making our system highly flexible and scalable in various application scenarios.

- The dual objectives of *energy efficiency* and *quality of service* are achieved through (i) an event-driven duty-cycle scheduling strategy which also leverages the VANET to minimize energy consumption in the WSN, and (ii) low-contention and low-delay communication protocols which ensure contention-less communication within a group and can reduce inter-group contentions with certain coordination costs.

- A prototype of our designed system has been implemented and tested in the field to study the *viability* of the system. Based on realistic vehicle traffic traces and roadside sensor-to-sensor communication traces, extensive simulations have also been conducted to study the impact of various factors on the system performance. The results demonstrate various design tradeoffs, and indicate that desired quality of service and energy efficiency can be achieved simultaneously when system parameters are appropriately chosen.

To the best of our knowledge, this is the first work that proposes, implements and evaluates an integrated VANET-WSN system for driving safety.

In the rest of this chapter, Section 6.2 presents an overview of our proposed system, which is followed by the design details in Section 6.3. Section 6.4 and Section 6.5 report our implementation and simulation results. Section 6.6 discusses practical issues. Finally, Section 6.7 concludes this chapter.

6.2 System Framework

6.2.1 Network Deployment

The proposed system consists of vehicle nodes and sensor nodes. Each vehicle node has two communication interfaces: a WiFi interface for communication with other vehicle nodes; and a ZigBee interface for communication with roadside sensor nodes. In our prototype, each vehicle node is an on-car laptop with an embedded WiFi card and a Telosb mote [59] attached to its USB port.

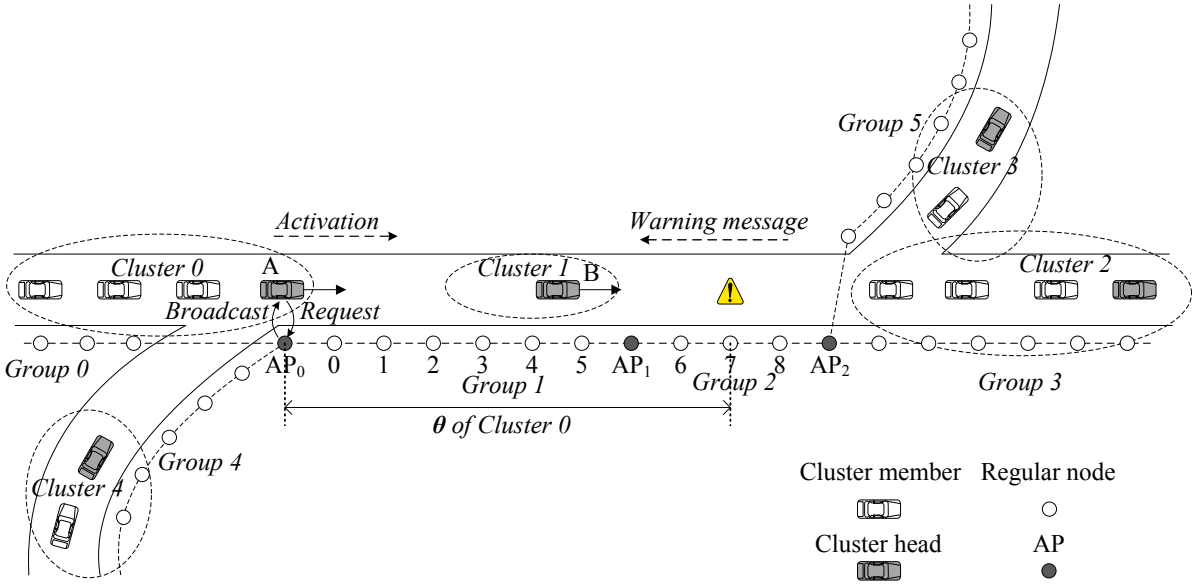


Figure 6.2: Network deployment of the integrated VANET-WSN

Each sensor node has a ZigBee interface used to communicate with other sensor nodes and with vehicle nodes, and in our prototype, each sensor node is a Telosb mote. Sensor nodes are also mounted with sensors which are used to sense road conditions.

As illustrated in Fig. 6.2, sensor nodes are deployed along one side of the highway. We consider only one-way highways, though the system can be extended to two-way roads. The sensor nodes form

a connected network. According to their roles in the system, sensor nodes have two types: the *regular sensor node* and the *access point sensor node* (called *AP* thereafter), and can sense and relay messages, while APs have extra responsibilities of discovering and communicating with vehicles, and managing the network. APs are much fewer than regular nodes. Regular nodes that are deployed between two adjacent APs along the roadside form a *group*. As shown in Fig. 6.2, one highway may merge into another one, two highways may be connected with a ramp, and one highway may branch into two or more highways; hence, the roadside sensor network is not linear. In our design, the node connected with three or more linear segments must be an AP.

In practice, some roads (e.g., in mountain areas) may be more prone to safety-related events than others; hence, sensor nodes may only be deployed along the roads with high risks. This way, deployed sensor nodes do not form a single connected network, but multiple disconnected networks. Our design is flexible and is applicable to such deployment due to the modularity approach adopted.

6.2.2 Duty Cycle Scheduling and Warning Message Forwarding

A connected partition of vehicular nodes on a highway forms a cluster. The formation of cluster has been widely studied [62–64] and is beyond the scope of this research. Each cluster maintains a *cluster head*, a node which is running at the front of the cluster. It is responsible for communicating with roadside sensors on behalf of the whole cluster. As shown in Fig. 6.2, there are five clusters, where cluster 2 and cluster 3 are connected but they are on different highways.

6.2.2.1 On-demand Duty Cycle Scheduling

As illustrated in Fig. 6.3, each AP periodically broadcasts a beacon message. If the AP has buffered some safety-related information that its nearby vehicles should be aware of, it will piggyback these messages in its beacon message. When a passing cluster head hears the message, it sends its registration request to the AP. The request message carries the ID of the cluster.

Responding to the request, the AP activates sensor nodes that are within a certain number (denoted as θ, a system parameter) of hops along the moving direction of this cluster of vehicles (called *forward direction* hereafter), if these nodes have not been activated yet. By this, these waken-up sensor nodes

will be able to proactively monitor the conditions of the roads to detect if there are dangerous conditions that should be known in advance by the drivers of the vehicles.

Note that, a roadside sensor is active only when there is a vehicle cluster approaching to its sensing range within θ hops. If there is no vehicle approaching this sensor, the sensor does not need to work. This is because that, no vehicle needs to know the road conditions nearby this sensor. This duty cycle scheduling scheme can minimize the time that a sensor should be active and hence can save energy.

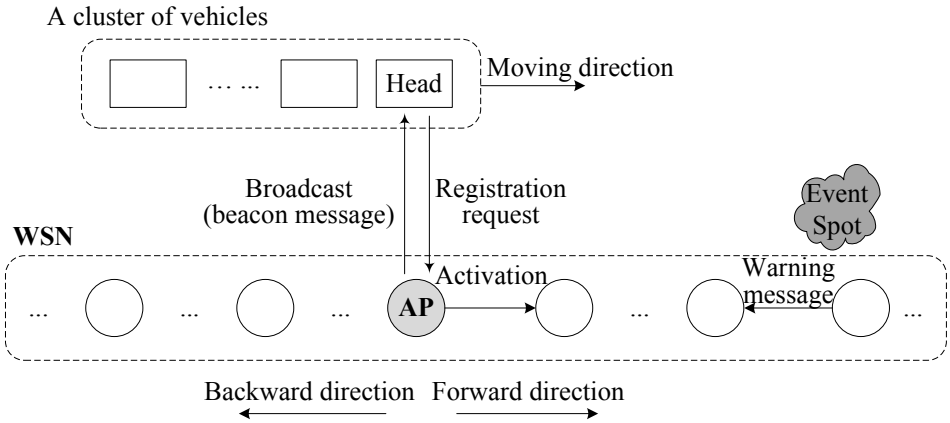


Figure 6.3: Big picture of the integrated VANET-WSN system

6.2.2.2 Warning Message Propagation

If a dangerous condition is detected (e.g., a deer is found roaming on the road), the detecting sensor node will generate a warning message and propagate it along the direction opposite to the moving direction of the vehicles (called *backward direction* hereafter) until the message reaches the heads of all incoming clusters that requested the activation of the sensor node. Then, the warning message is disseminated within the clusters of vehicles by using a certain communication protocol such as [65–67]. This way, VANET nodes are leveraged whenever possible to reduce the workload of the roadside WSN to save its energy. In addition, when the warning message propagation is completed, sensor nodes can temporarily store the warning messages. Later on, they can resume the propagation immediately through the WSN or the VANET when a follow-up cluster activates them. Hence, the message can be disseminated further on demand of the follow-up vehicles until it expires. This method adapts the

warning message propagation to the current traffic, making our system more efficient.

6.3 Detailed Design of the System

The above description on duty cycle scheduling and warning message propagation remains high-level. To realize these functionalities, practical and efficient protocols should be designed for scheduling duty cycles of sensors and for propagating activation messages in forward direction or warning messages in the backward direction. The duty cycle scheduling and message propagation are not independent of each other. Therefore, we will study these two issues together.

One big challenge in designing these protocols is, the forward and the backward propagations take place on the same communication channel, and hence the propagations should be scheduled appropriately to avoid or reduce collisions to minimize both propagation delay and energy consumption. Although CSMA/CA-based protocols are commonly used in wireless ad hoc and sensor networks, TDMA-based protocols are preferred in our system for following reasons:

- As opposed to CSMA-based MAC protocols commonly used in wireless networks, sensor nodes in the proposed system often have little data to transmit (packets are generated only when cluster heads pass APs or events are detected by sensors), and therefore, should ideally stay in the low duty-cycle state during most of time to save energy. Meanwhile, once there is data to transmit, the data should be transmitted in a timely fashion to guarantee quality of service. If CSMA/CA is adopted, time and energy may be wasted for long idle listening, dynamic adjustment of duty cycles, medium contention, etc. On the other hand, with TDMA-based protocol would be a good choice, the shutdown of the sensors when they are not in use and the wakeup when new data come can be deliberately scheduled to save energy and meanwhile ensuring short delay in transmission.

- To improve network throughput and support real-time data delivery in sensor networks, TDMA-based MAC protocols [68–70] have been proposed recently. TreeMAC [68] proposes an innovative localized TDMA MAC protocol to achieve high throughput and low congestion in data collection sensor networks. Although it can achieve real-time transmission, its high data rate and

tree structure of sensor nodes make it unsuitable for our proposed system. Funneling-MAC [69] has been proposed to solve the funneling effect (i.e., the nodes closer to the sink are heavily congested) by using TDMA in the intensity region. TDMA scheduling is managed by the sink node. Essentially, it is a centralized protocol, which is not applicable for our system where every node can only reach a limited number of nodes. Moreover, these protocols are for unidirectional communication, while our system requires bidirectional. Thus, designing a new TDMA-based protocol is necessary for our proposed system.

- Further, the special network topology in the proposed system can facilitate the application of TDMA-based protocol. Each sensor has a limited number of neighboring nodes, which are predetermined, making the assignment of the time slots for communication easier. By carefully assigning the time slot for transmission, we can avoid or greatly mitigate the hidden terminal problem that is hard to be mitigated with CSMA/CA-based protocols.

However, TDMA-based protocols require strict time synchronization among nodes, which is hard to accomplish in large-scale systems. To accommodate two-direction communication in a single channel and meanwhile achieve scalability, we adopt the idea of modularity: sensor nodes are divided into groups; within each group, duty cycles of nodes and two-direction propagations are scheduled to achieve both contention-less communication and energy efficiency; inter-group communication is handled by APs shared by different groups.

In the following, we assume that the roadside WSN has already been deployed and grouping of sensors has been determined, based on which we first present our proposed intra-group and inter-group scheduling schemes. Then, we discuss the choice of system parameters and bootstrapping of the system.

6.3.1 Intra-group Scheduling

Sensors in the same group are time synchronized, and the approach to maintain the synchronization is to be presented in Section 6.4. Therefore, all sensors in the same group share a common time reference. The time is divided into slots of fixed length. During each slot, a packet can be sent from a sensor to its neighbors; hence, we call the length of a slot a *packet time (denoted as τ)*. A certain number

of slots form a period, and the length of a period is denoted as p. Protocols for duty cycle scheduling and medium access control (MAC) are presented in the following such that, every c_f period(s), a packet can be propagated hop by hop from the most back sensor of the group to the most front sensor along the forward direction, and every c_b period(s), a packet can be propagated hop by hop from the most front sensor to the most back sensor along the backward direction. Here, we call c_f the *forward interval* and call c_b the *backward interval*.

6.3.1.1 Scheduling for Forward or Backward Propagation

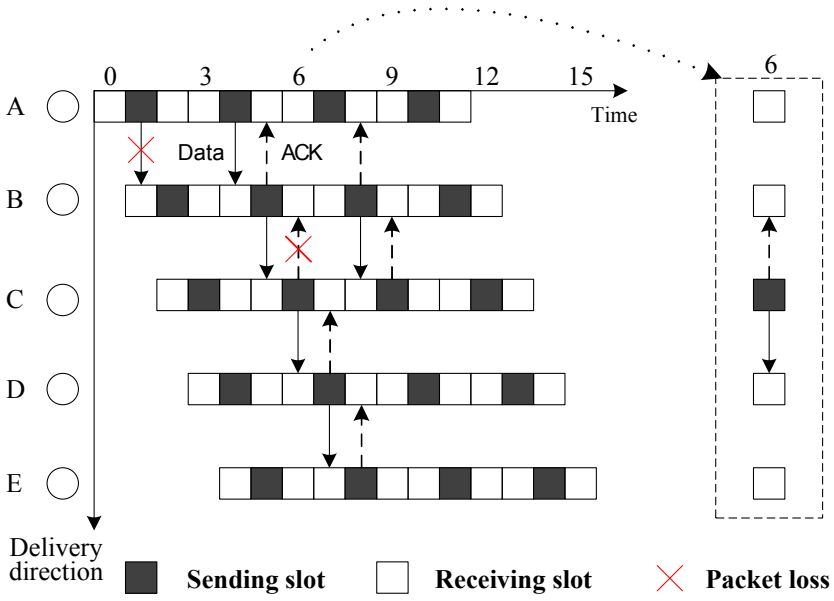


Figure 6.4: Example of scheduling for forward propagation

Without loss of generality, let us consider the example shown in Fig. 6.4, where circles $A, B, \cdots, E$ represent sensors in the same group, A is the most back sensor and E is the most front sensor. We want to schedule the duty cycles of these nodes and their communication behaviors such that a packet can be forwarded from A to E hop by hop.

Taking into account the unique characteristics of the network topology, we adopt the following methods to design forwarding propagation protocol which has no contention, high energy efficiency and low propagation delay:

Firstly, TDMA-based access control is adopted to eliminate contention. For each sensor, a certain number of slots are reserved for it for sending or receiving. The reservation of slots follows the following rule: During the slot reserved for node X for sending, none of its one-hop and two-hop neighbors is allowed to send packets. For the example in Fig. 6.4, during the slots for sensor C to send packets, sensors A, B, D and E are not allowed to send packets. This way, contention (even the hidden terminal problem) can be eliminated.

Secondly, the broadcast nature of transmission is leveraged to speedup packet propagation and reduce acknowledgement overhead. Specifically, after a node has received a data packet from its previous hop, it forwards the packet to the next hop immediately in the next slot. Due to the broadcast nature of transmission, the data packet can also reach the previous hop, serving as the acknowledgement. If the packet cannot reach the previous hop due to errors in the channel, the packet that has arrived at the next hop can be propagated further without waiting for the acknowledgement packet being successfully sent to the previous hop.

Thirdly, reserved retransmission slots can be dynamically shared among sensors in the same group. For reliability, retransmission slots are reserved for sensors. However, the quality of different links may not be the same and may change dynamically. For example, sometimes the link between sensors A and B may be better than the link between D and E, and vice versa in other time. Considering this, our design can enable sensors to dynamically share a certain total number of retransmission slots.

The forward scheduling protocol is detailed as follows.

- *Reservation of slots*: The most back sensor is assigned with $3(r+1)$ sequential slots, where r is the system parameter specifying the maximum times to retransmit a packet by all nodes in the group, which we call *retransmission quota* hereafter. Without loss of generality, we call the first slot assigned to the node as slot 0, and the remaining slots are called slot $1, 2, \cdots, 3(r+1)-1$, respectively. Slots $3i+1$ ($i = 0, \cdots, r$) are reserved for sending while others are reserved for receiving. If we use R to represent a slot for receiving and S to represent a slot for sending, all these slots can be represented as a sequence of $r+1$ RSR's. For each of the remaining sensors in the group, it is also assigned with $3(r+1)$ sequential slots of the same sensing/receiving pattern, except that its first slot is one slot later than that of its previous hop. In the middle of Fig. 6.4,

the scheme for slot reservation is shown for a group composed of 5 nodes and parameter $r = 3$.

- *Sending of a packet*: If a sensor has a packet to propagate, it will send it out at the first sending slot. If it overhears the forwarding of this packet or receives an acknowledgement in the next slot, which is reserved for receiving, from the next hop, the transmission is successful. Otherwise, it will retransmit the packet in the next sending slot. The procedure continues until the transmission is successful or the slots reserved for sending have been used up. In the case that the reserved slots have been used up, the packet can be transmitted in the next reserved propagation time (i.e., nearly $c_f \cdot p$ time later).

- *Receiving/forward of a packet*: If a sensor does not have any packet to send, it will listen in the first slot. If it does not hear anything from its previous hop, it can go to sleep in the following two slots since it can be predicted that it will not have any sending or receiving in the next two slots. If it receives a packet from its previous hop, it will transmit the packet to next hop immediately in the next slot, which is a slot reserved for sending. Then the follow-up procedure for checking if the packet has been successfully sent and retransmitting the packet is the same as in the part of *Sending of a packet*. Note that, if its forwarding is not overheard by its previous hop, the previous hop node may resend the packet. In this case, this forwarding node should be able to identify the duplication; then, it will send a dedicated acknowledgement packet to its previous hop in the next sending slot.

Note that, if multiple sensors in the group have packets to send, these packets can all be propagated except that, some sensor in the middle may have multiple packets to send/forward. In this case, it can merge these packets into one if possible and send it, or send these packets one by one.

An example for packet sending and forwarding is also shown in Fig. 6.4, which is explained as follows. Sensor A wants to send a packet to sensor E. It starts the transmission at its first available sending slot, slot 1. However, this packet gets lost. Hence, A will not receive the acknowledgement from B during the following receiving slot, so it retransmits the packet in the next available sending slot, i.e., slot 4, and it succeeds. Upon receiving the new packet, B immediately forwards the packet, which serves as both data packet to downstream node (C) and acknowledgement to upstream node (A).

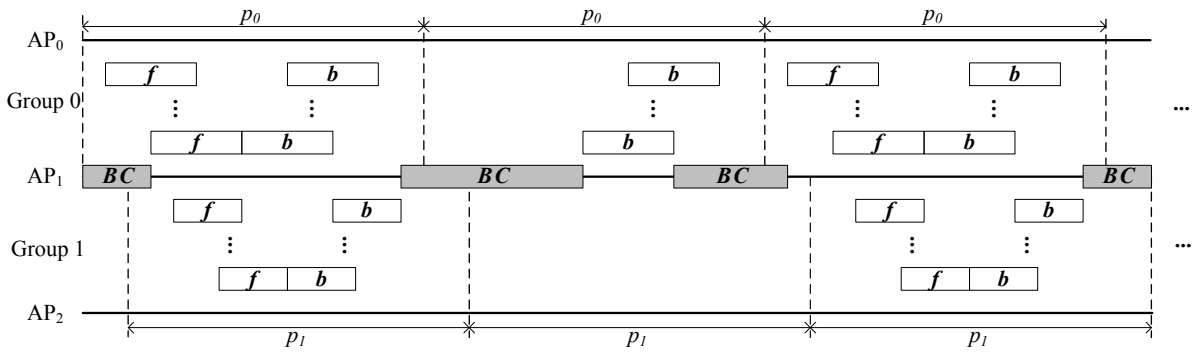


Figure 6.5: Forward and backward propagation scheduling in the system with multiple groups. Group 0: $c_b = 1$ and $c_f = 2$; Group 1: $c_b = c_f = 2$; two groups have the same period $p_0 = p_1$; BC: slots that the AP can use to broadcast beacon messages

This sending packet has been received by C but is not acknowledged successfully. Thus, B assumes that C has not received the packet and retransmits that packet. This retransmission packet can be overheard by A and C. A simply ignores it while C attempts to resend the acknowledgement. Due to the good link quality in following propagation, the packet can eventually reach E even before every packet has been successfully acknowledged.

With the same idea, the protocol for backward propagation can also be designed similarly. Fig. 6.5 has shown examples of backward propagations next to forward propagations. A sensor can turn off its radio during the slots that are not reserved for sending or receiving.

6.3.2 Inter-group Scheduling

The scheduling of each group is made independently. When two groups are connected together at an AP, an issue arises: how can the AP successfully pass packets from one group to another with low delay? To address this issue, the AP needs to cooperate with its neighboring regular nodes (called *boundary nodes*, for example, nodes 5 and 6 are boundary nodes of AP_1 in Fig. 6.2) as follows.

The AP needs to know the schedules of its boundary nodes. For this sake, the boundary nodes periodically tell the AP their schedules, by either explicitly sending the schedule or implicitly piggy-backing it in the data packet. Knowing the schedules of its boundary nodes, the AP should be active when any of its boundary nodes is active. This way, packets sending to the AP will not be missed if no

collision.

The AP also follows the protocol below to ferry packets between groups:

(i) Initially, the AP is in the *idle* state. Suppose the AP is connected with multiple groups, we call a group connects to it on the backward direction as its *upstream* group and a group connects to it on forward direction as its *downstream* group. For example, in Fig. 6.2, Group 1 is a upstream group of AP_1 while Group 2 is a downstream group of AP_1. When the AP receives a forward (backward) packet from its upstream (downstream) group, AP is bound to delivering the forward (backward) packet and hence sets itself to the *forward* (*backward*) state.

(ii) The AP in forward (backward) state is dedicated to delivering the forward (backward) packet. Any incoming backward (forward) will be just buffered and not acknowledged.

(iii) The AP will make an attempt to send the forward (backward) packet to the downstream (upstream) group if (a) any boundary node is in its forward (backward) receiving slots and (b) the last attempt was two time slots away from the current attempt to ensure AP to have enough time to get the possible acknowledgement.

(iv) Step (iii) is repeated until the AP has got an acknowledgement from its downstream (upstream) group. Then, AP will check its buffers to see if there is any packet ready to be delivered. If so, step (iii) and (iv) will be repeated; otherwise, the AP goes back to step (i).

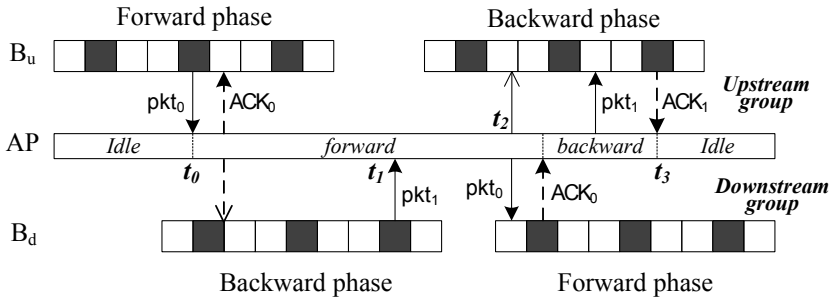


Figure 6.6: Inter-group communication

Fig. 6.6 shows an example. B_u and B_d are two boundary nodes of the AP, whose schedules are shown in the figure. At time t_0, B_u sends a forward packet (pkt_0) to the AP. Since the AP is in the idle

state, it switches to the forward state and sends out the packet acting as acknowledgement. This packet can be also received by B_d as well. However, since B_d is now in backward phase, it will just buffer this packet without acknowledging it. At time t_1, B_d sends a backward packet (pkt_1) to AP. Since AP is now in the forward state, it will also just buffer this packet without acknowledging it. After some time, the forward phase of B_d becomes available. Then, the AP sends pkt_0 at B_d's first available receiving slot. Note that, even if this packet can not be successfully transmitted to B_d, B_d can still send out the acknowledgement for the previously buffered packet. Upon receiving the acknowledgement from downstream group, the inter-group delivery of that packet is accomplished. At that time, the AP checks its buffer and finds the backward pkt_1 is there to be delivered. Then, it changes to backward state and starts another inter-group delivery, which is finished at t_3.

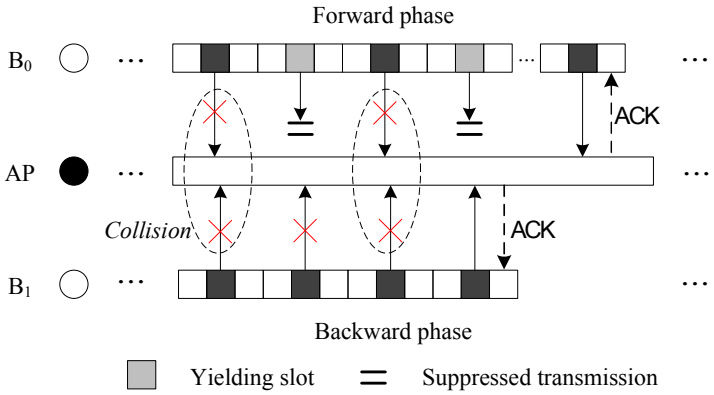


Figure 6.7: Collision resolution at AP

In some extreme case, data packets from two boundary nodes may arrive at the AP simultaneously and the schedules of these two boundary nodes match exactly. Then, the AP may never receive the data packet from either node because collision always exists. In this case, the above scheme fails. Thus, we propose the *yield* mechanism to deal with this situation. The basic idea is to let one boundary node yields to the other when they do not receive the acknowledgement from the AP for a certain number of times (which indicates the possible occurrence of collisions). At this time, one boundary node will start resending the packet once every two sending slots, while the other remains the same. Fig. 6.7 shows a simple example, where boundary node B_0 yields to boundary node B_1. In the first yielding slot, B_1 gets a chance to transmit its backward packet to the AP, but it fails due to packet loss. After one more

collision, B_1 finally gets the backward packet successfully transmitted to AP at the second yielding slot.

Broadcast of AP Beacon Messages When none of the boundary nodes of the AP is in their reserved slots for backward or forward propagation slots, shown as "BC" blocks in Fig. 6.5, the AP can pick some time point to broadcast beacon messages such that passing cluster heads can discover and contact with the AP, as described in Section 6.2.2. The interval between two consecutive beacon messages should be short enough to ensure that a passing cluster head cannot miss it during its stay within the transmission range of the AP.

During the time that none of the boundary nodes is active in their forward/backward propagation and the AP does not broadcast beacon messages, the AP can go to sleep to save energy.

6.3.3 Discussion on System Parameters

In this section, we show the relation between various system parameters by presenting our derived equations.

6.3.3.1 System Parameters n and r

Recall that, the number of sensors in a group is denoted by n and the total times that a packet can be retransmitted in a group (i.e., the retransmission quota) is denoted by r. Given n, parameter r should be carefully determined such that (i) forward or backward propagation of a packet across a group can be finished with less than r retransmissions and (ii) the number of wasted slots reserved for transmission/receiving is as small as possible. The value of r can be derived based on the probability model given in [71].

Since the forward propagation and the backward propagation are symmetric, we only consider the value of r for the forward propagation. Let p_i denote the forward packet loss probability of node i in the group, and $s_i(k)$ denote the probability that a packet is successfully delivered by node i after k transmissions. Then, $s_i(k) = p_i^{k-1}(1 - p_i)$. Thus, the expected number of transmissions required to successfully send a packet by node i is denoted by m_i, where $m_i = \sum_{k=1}^{\infty} k \cdot s_i(k) = \frac{1}{1-p_i}$.

The expected number of *retransmissions* that a packet needs to cross a group equals to $\sum_{i=1}^{n}(m_i -$

1), which should be no greater than r. Form this inequality, we can get the lower bound of r:

$$\sum_{i=1}^{n} \frac{p_i}{1-p_i} \leq r \tag{6.1}$$

Once the network is deployed, n is fixed. Hence, r is only related to p_i. To get p_i, each node i just needs to piggyback a retransmission count in each data packet. When node $i+1$ *first* receives the packet from node i, it can know how many retransmissions have been made to deliver this packet. In this way, each node $i+1$ can estimate the value of p_i and report it to the AP. Thereby, the AP can compute the value of r according to Equation (6.1) and update the schedule of the group when necessary. In addition, we should also not let r be too large, since a large r may result in a large fraction of time that sensors should be active.

If we assume that each sensor node has the uniform packet loss ratio, say $\bar{p}$. Thus, from the Equation (6.1), we can compute the corresponding retransmission quota of each group by

$$r = \frac{n\bar{p}}{1-\bar{p}} \tag{6.2}$$

6.3.3.2 System Parameter c_f and c_b

Recall that, the forward interval is denoted by c_f and the backward interval is denoted by c_b. Without loss of generality, we analyze the impact of c_f on the delay of forward propagation, and the impact of c_b on the delay of backward propagation is similar.

The propagation delay within a group includes two parts: the intra-group delay (among regular nodes), denoted as d_r, and inter-group delay (at AP), denoted as d_a. Assuming that r is appropriately chosen according to Equation (6.2) such that the propagation can be done with a single (forward) phase. At the same time, the (forward) phase of downstream boundary node is just over when the packet reaches the AP (i.e., the delivery stops until the next delivery phase becomes available). Thus, in this *worst* case, we can have $d_r = 3(r+1)\tau + (n-1)\tau = (3r+n+2)\tau$ and $d_a = c_f p$, where τ is the length of a slot and p is the length of a period (defined before). Thus, the propagation speed within

a group (including one AP), denoted by v_p, is

$$v_p = \frac{(n+1)I}{d_r + d_a} = \frac{(n+1)I}{(3r+n+2)\tau + c_f p}, \tag{6.3}$$

where I is the distance between two neighboring sensors. From this equation, we can see that by changing c_f we can dynamically control the propagation speed and further satisfy the delay requirement. Again, c_f and c_b are managed by AP, which can update them to meet the requirement of different applications.

6.3.3.3 System parameter θ

On the vehicle side, the only safety-related parameter specified by drivers is θ. For the sake of safety, it should be proportional to the current velocity of the head vehicle. The head vehicle in high velocity should have a large θ in order to have enough time for drivers to react. Theoretically, θ is equivalent to the number of hops that will be activated for safety detection and it should stay constant after being specified. However, due to dynamics of velocity (v) in driving and deployment of APs, the actual number of activated nodes, denoted as $\hat{\theta}$, is changing from time to time.

Let H be the number of hops between the previous AP, say AP_x, and the next AP, says AP_y. When the head vehicle was passing AP_x, θ nodes would be activated. Suppose the activation needs $\theta I/\bar{v_p}$ time where $\bar{v_p}$ is average propagation speed of activation messages. During this duration, the head vehicle can advance $(\theta I/\bar{v_p}) \cdot v$, where v is the head vehicle's current speed. Actually, when the head vehicle launches an activation at AP_x, the actually activated sensors are:

$$\theta - \frac{\lfloor \theta I v/\bar{v_p} \rfloor}{I}.$$

This value of $\hat{\theta}$ keeps *decreasing* as the vehicle cluster is approaching the next AP, AP_y. When it reaches AP_y, the sensor activation message will be propagated again and some more sensors along the moving direction will be activated. Now, suppose the head vehicle is moving between AP_x and AP_y

and currently H hops away from AP_x. $\hat{\theta}$ is constrained by:

$$\theta - \frac{\lfloor \theta I v / \bar{v_p} \rfloor}{I} - H \leq \hat{\theta} \leq \theta - \frac{\lfloor \theta I v / \bar{v_p} \rfloor}{I} \tag{6.4}$$

From Equation (6.4), we can also see that $\hat{\theta}$ will decrease as the speed is going up.

In practice, what the drivers expect is $\hat{\theta}$ while what the APs use is θ. Thus, when a head vehicle registers a service at AP, it should first employ Equation (6.4) to derive a proper θ and the estimated value of $\bar{v_p}$ should be piggybacked in APs' beacon messages.

6.3.4 System Bootstrapping and Maintenance

System bootstrapping is conducted group by group. Initially, there is only one starting AP_0 on the road. The administrator of the roadside WSN determines the system parameters of the newly deployed group, which include n, r, c_b and c_f. Then the sensors in this group are preloaded with these parameters and are deployed one by one. Next, another AP (denoted as AP_1) is deployed. After the deployment, all nodes in the group synchronize their time clock through exchanging a message between each pair of neighboring nodes. The clocks of different sensors may have different frequencies, resulting in clock drift after sometime. To address this problem, sensors in the same group periodically synchronize their clocks, which is also detailed in Section 6.4

Due to the modularity nature of our system, system maintenance (e.g., sensor addition, replacement, system parameter resetting, rotation of APs etc.) can be performed within groups autonomously. The basic standpoint is that any adjustment to the network can be bounded within two APs (or the group), making the impacts local. Since their underlying principles are similar to the above system bootstrapping, we will not elaborate here.

6.4 Prototyping and Field Tests

We implement the proposed system and test it in the field. In the implementation, three components, namely, AP, regular node and vehicle node, have been prototyped. The vehicle node is implemented atop a laptop injected with a Telosb mote. AP and regular nodes are implemented atop Telosb motes

with WiEye passive infrared sensors mounted. The Telosb motes run TinyOS-2.1.0. The table below shows the image sizes of the software modules developed for these components:

Component	ROM	RAM
AP	33274bytes	1604bytes
Regular node	32194bytes	1600bytes
Vehicle node	17558bytes	1086bytes

For TDMA-based protocols to work, time synchronization is a prerequisite. To realize time synchronization, we use two interfaces provided by TinyOS-2.1.0 library: *TimeSyncAMSend* and *TimeSyncPacket*, which provide the primitives to synchronize a group of nodes through exchanging packets.

The major purposes of field tests are two folds. The first is to test if the proposed system works in field, and the second is to find out the impact of real environmental factors on the proposed system, especially on the communication of the system. Hence, we conduct two sets of field tests in a large open parking lot: One set of experiments are to test how the whole system works. The test is conducted in two scenarios: there exist intensive WiFi traffics nearby and there does not. Another set of experiments are conducted to measure the impacts of environmental conditions on communication between two Telosb motes when the interference level varies. Here, we elaborate the findings and results from the first set of experiments, while the results from the second set of experiments are used as inputs to our simulation which is discussed in Section 6.5.3.

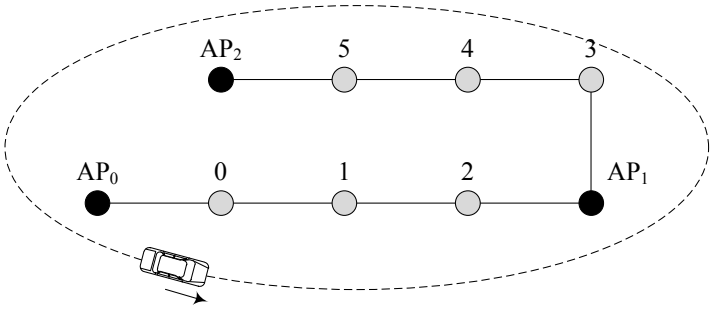


Figure 6.8: Experiment topology

Two groups of Telosb motes (including totally 9 motes) are deployed along the roadside in a large open parking lot, as shown in Fig. 6.8. The motes cover the length of 480 meters, the distance between

two adjacent motes is 60 meters. A vehicle repeatedly runs along the motes. Whenever the vehicle enters the road from one end and is discovered by an AP, the AP will wake up all the rest motes to start sensing. Warning messages are generated by the AP located at the other end of the road at a constant frequency, and the messages are propagated to the AP who discovers the vehicle and then is delivered to the vehicle. Other experimental parameters are presented in the table below.

Radio transmission range	100 meters
AP Beacon period	10 seconds
Event interval	20 seconds
Retransmission quota (r)	3
Number of hops to activate (θ)	8
Vehicle speed	20 miles/hour
Experimental duration	20 minutes

6.4.1 System Performance with Interference

As WiFi communication is expected to co-exist with the proposed system, we first test the working and performance of the system when WiFi communication exists. For this sake, two laptops equipped with WiFi cards are put near each of the APs, respectively, to serve as interferers. To make the interference strong, about 10 Mbps traffic is exchanged between the them. In the experiment, WiFi communication uses channel 6 (the default channel) and ZigBee uses channel 26. For comparison, we also conduct experiment for the situations of no WiFi traffic.

In these experiments, we set the forward/backward interval (c_f/c_b) and group size (n) to 3. We measured the average per-hop delay for the forward/backward message propagation, which are denoted as $D_{Forward}$ and $D_{Backward}$ respectively. The results are shown in the table below. From the results, we can see that the average delay measured with interference is slightly (i.e., between 5% and 9%) higher than that without interference for both forward and backward message propagation. Note that, the simulated interference traffic is intensive. This indicates that the impact of interference on propagation delay is not significant.

$n = c_f = c_b = 3$	With Interference	No interference
$D_{Forward}$(ms)	54.07	**51.06**
$D_{Backward}$(ms)	95.99	**88.31**

Table 6.1: Impact of interference on propagation delay

6.4.2 System Performance with Varying Parameters

Since the impact of interference from WiFi traffic is insignificant, we conduct more extensive experiments without the interference. In the experiments, we vary the system parameters (i.e., c_f, c_b and n) and measure the propagation delay. The results are as follows.

$D_{Forward}$(ms)	$c_f = c_b = 2$	$c_f = c_b = 3$	$c_f = c_b = 4$
$n = 3$	63.04	**51.06**	59.57
$n = 4$	205.19	75.73	157.25
$n = 5$	85.76	122.73	102.65

Table 6.2: Forward propagation delay with different system parameters

$D_{Backward}$(ms)	$c_f = c_b = 2$	$c_f = c_b = 3$	$c_f = c_b = 4$
$n = 3$	111.69	**88.31**	97.30
$n = 4$	540.59	105.39	189.08
$n = 5$	293.40	327.29	424.17

Table 6.3: Backward propagation delay with different system parameters

As we can see the largest forward propagation delay is about 205 ms per hop, which means the speed for propagating activation messages from an AP which detects an incoming cluster of vehicles to other sensors that should be activated is about 293 m/s, i.e., 659 miles/h, which is much faster than the speed of a vehicle. The largest backward delay is about 540 ms per hop, which means the speed to propagate a warning message to related vehicles is about 111 m/s, i.e., 250 miles/h, which is also much faster than the speed of a vehicle.

We can also see that the backward delay is higher than the forward delay. The reason is found to be that, the forward phases of boundary nodes happen to have a better match than their backward phases in our experiments. Consequently, each forward packet arriving at APs can be relayed to the downstream group immediately, while some backward packets have to wait for the next available backward phase of the downstream group. Besides, we can see that the propagation delay goes up as the forward/backward

interval increases most of the time. Occasionally, it varies. By analyzing the collected data at each node, we find that the variations are caused by the random packet loss, which affects the average delays.

6.5 Simulation

NS2-based simulation has been conducted to evaluate our design. We evaluate the impacts of system parameters and environmental factors (reflected by *packet loss ratio* at sensor (p_s) and vehicle (p_v)) on the system performance. The system parameters include *group size* and *forward/backward interval*. Since retransmission quota is decided by group size and packet loss ratio as shown in Section 6.3.3, we do not explicitly consider it in our simulation. The performance metrics include *energy consumption* (the average energy consumption per hour of all APs and regular nodes) and *propagation delay* (the time from when the event occurs to when the cluster head receives the warning message, which is normalized as delay per hop).

We conduct both *theoretical evaluation* and *empirical evaluation*. For the theoretical evaluation, we vary the system parameters within theoretically-possible ranges to evaluate our system performance. For the empirical evaluation, we follow the empirical traffic data to generate traffic and use the packet transmission traces collected from field experiments.

6.5.1 Setup

The following table shows the parameters fixed in the simulation. We simulate a highway with more than 200 sensor nodes deployed along one side. Based on field experimental result, we set the packet time (i.e., length of a slot in the proposed protocols) to 25ms. We assume $c_f = c_b$.

6.5.2 Theoretical Evaluation

Since the system performances are associated with three different parameters, we evaluate each performance metric by varying one parameter while fixing the other two. Besides, the arrival rate of the clusters is set to be 2 cluster per minute and the average speed of the vehicles is 30m/s (67.5 miles/hour).

Road length	18900 meters $\times$ 20 meters
Number of hops to activate (θ)	50 hops
AP Beacon interval	600 ms
Sensor transmission range	100 meters
Inter-node distance (I)	90 meters
Vehicle transmission range	250 meters
Slot length (τ)	25 ms
Interval between two events	6 minutes
Simulated time	1 hour
Sensor power	2$\times$AA batteries $\approx 20,000\ J$

Table 6.4: Simulation settings

6.5.2.1 Group size n

Fix $c_f = c_b = 5$ and $p_s = p_v = Uniform(0, 0.3)$. In Fig. 6.9, we can see that the impact of group size on delay becomes insignificant as it goes up. This is because, by combining Equation (6.2) with (6.3), we know that v_p converges to a constant value as n approaches infinity and c_f (or c_b) and t are fixed. For the energy consumption on sensor side, we can see that a larger group consumes less energy per node. This is because forming a large group can reduce the number of boundary nodes and APs, which consume more energy than regular nodes, in a given area.

However, we can not conclude from the above results that, the larger the group size is the better performance we can achieve. The major problem of forming large group is that the message propagation delay at APs becomes larger as the group size increases. This means that the activation process becomes slow; hence, a vehicle may move ahead of the activation message, which is not desired for safety. Thus, an appropriate group size should be around 25.

6.5.2.2 Forward/backward Interval c_f and c_b

Fix $n = 20$ and $p_s = p_v = Uniform(0, 0.3)$. It is obvious that by using larger forward/backward interval the sensor nodes can get a larger fraction of time to sleep. Therefore, the delay increases and energy decreases accordingly, which is approximately linear, as shown in Fig. 6.10. Since group size is often pre-defined according to the road topology and packet loss ratio is related to environmental conditions, the forward/backward interval should be the key factor affecting our system performance.

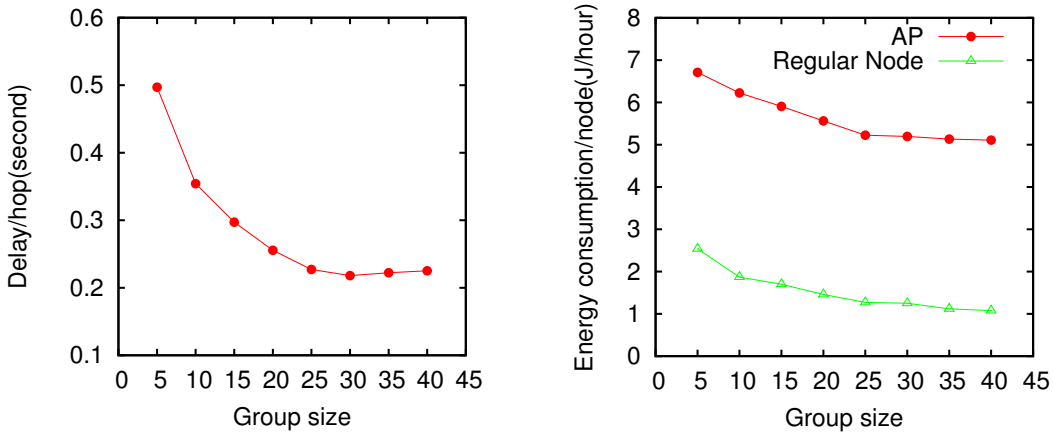


Figure 6.9: Impact of group size on system performance

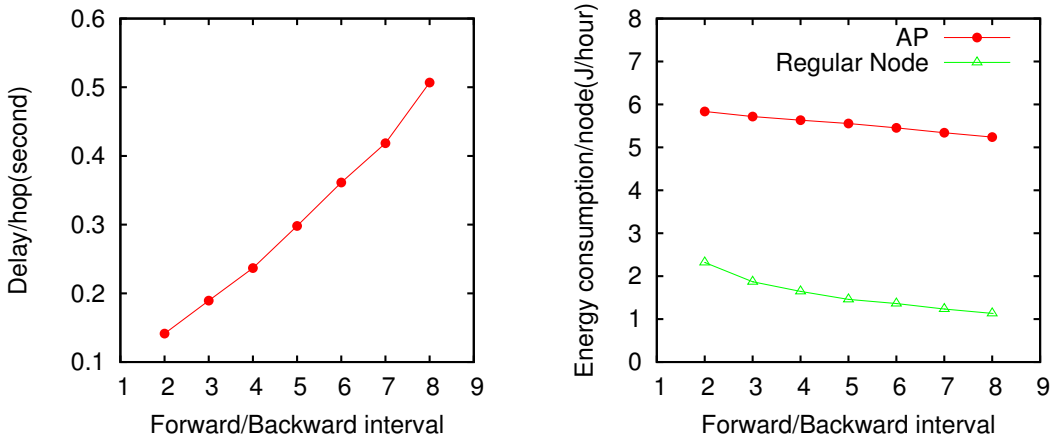


Figure 6.10: Impact of forward and backward interval on system performance

6.5.2.3 Packet loss ratio: p_s and p_v

In reality, packet loss can be caused by either environmental factors or the interference from WiFi or outside sensors. Thus, we measure the impact of packet loss on system performance by adding error model to our simulation, such that the retransmission can be conducted when the packet gets lost. Here, $n = 20$ and $c_f = c_b = 5$.

First, we vary p_s while fixing p_v (also *Uniform(0,0.3)*). In Fig. 6.11, the propagation delay increases significantly as p_s is high, because the number of retransmissions is very likely to exceed r, causing long delay. The energy consumption increases as p_s grows. This is caused by the increasing of the

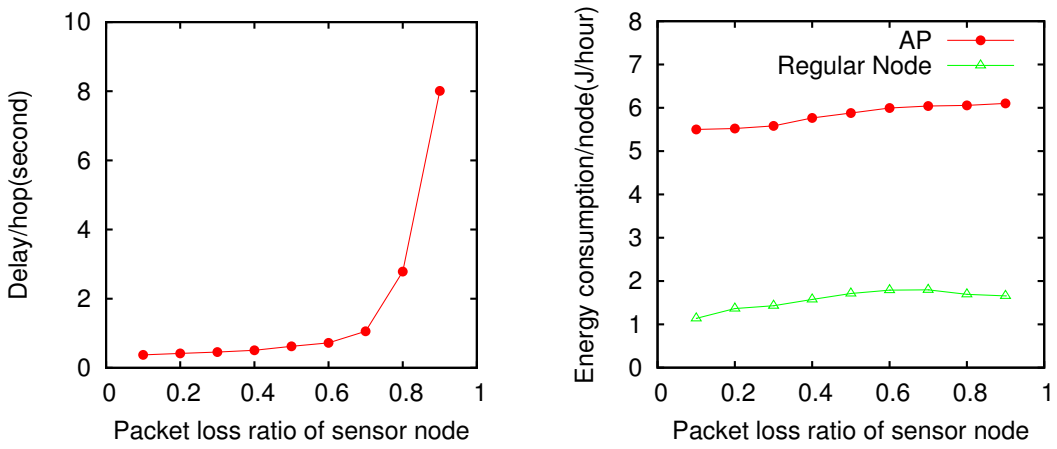


Figure 6.11: Impact of sensor packet loss on system performance

number of retransmissions. However, when the packet loss ratio is high, the energy consumption increases slightly or even decreases due to the significant increasing of the propagation delay. Large delay can reduce the average propagation speed ($\bar{v_p}$) greatly. Fewer sensors will participate in the forward and backward propagations. Hence, less energy is consumed.

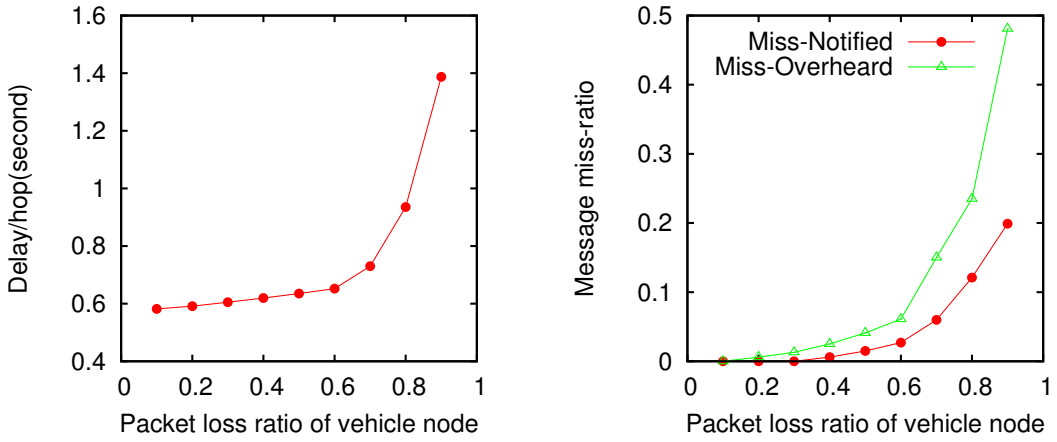


Figure 6.12: Impact of vehicle packet loss on system performance

Second, we consider p_v while fixing p_s (also *Uniform(0,0.3)*). In our system, there are two ways for the cluster head to catch the warning message: one is by overhearing the backward warning propagation, the other is by extracting the warning information piggybacked in the beacon messages when passing APs. Obviously, using the overhearing results in a shorter delay. However, if the packet loss

is severe, the vehicle is very likely to fail to overhear the warning message. In some extreme cases, it even remains unaware of the event when it has passed the event spot. Fig. 6.12 captures the above concerns. Similar to the sensor nodes, the impact of p_v on propagation delay is not significant when p_v is low. The reason is that each vehicle has more than one chance to overhear the warning message since it can be within the transmission range of two sensor nodes simultaneously. In addition, we also show the probability that the warning message fails to be caught by the vehicle, i.e., *miss-notified ratio*, and the probability that the warning message fails to be overheard by the vehicle but succeeds to be informed through the beacon message from APs, i.e., *miss-overheard ratio*.

6.5.3 Empirical Evaluation

In order for the simulation to better reflect the real-world traffic, we use the analysis results of the empirical vehicle traffic data measured on I-80 freeway in California in [72]. Based on that, we compute the cluster arrival rate on I-80 freeway, which is then fed into our simulator to generate traffic. Also in field experiments, we log the packet transmission of the sensor nodes under different traffic scenarios. These logs are transformed into the packet loss traces and then fed into our simulator to determine the reception and dropping of the incoming packets.

6.5.3.1 Traffic generation

In our simulation, the cluster heads are generated following the three traffic categories proposed in [72]: *night traffic* with very low traffic volume and high speed (1 am - 3am), *free-flow traffic* with moderate traffic volume and high speed (10 am - 12 pm) and *rush-hour traffic* with low speed and very high traffic volume (3 pm - 5 pm).

Traffic	$\bar{v}$ [m/s]	Density [veh/m]	Cluster arrival rate [cluster/hour]
Night traffic	30.93	0.0019	133
Free-flow traffic	29.15	0.0125	57
Rush-hour traffic	10.73	0.0364	<1

6.5.3.2 Packet transmission traces

To emulate the road-side interference, we deploy some sensor nodes in the middle of two *regular nodes* to act as the interferers by randomly broadcasting messages (20 packets/second on average). The reason that we choose sensors, rather than WiFi devices, as interferers is to make the interference more intensive since they share the same channel with roadside sensors in our system. The distance between two nodes is nearly 100m. According to different traffic categories, different numbers of interferers are employed: 2 for night traffic, 4 for free-flow and 6 for rush-hour. For comparison, we also tested no interferer scenario. The number of transmissions for a serial of packets (called *packet trace*) are logged, as shown in Fig. 6.13.

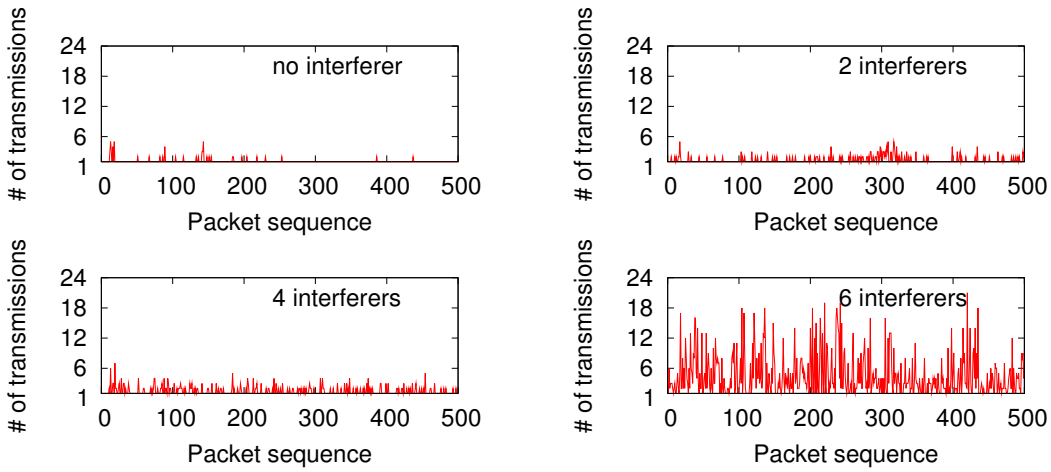


Figure 6.13: Outdoor experiment: packet transmission traces

6.5.3.3 Results

Based on the empirical traffic generation and collected packet transmission traces, simulations are conducted. The first figure in Fig. 6.14 shows the propagation delay under three traffic categories. We see that forward/backward interval is the dominant factor that affects the delay, especially when group size is large. The delay in free-flow traffic is slightly higher than that in night traffic. However, the delay in rush-hour is much higher than the other two cases due to severe interference. Actually, this does *not* lead to a low system performance, since according to our proposed system model only cluster

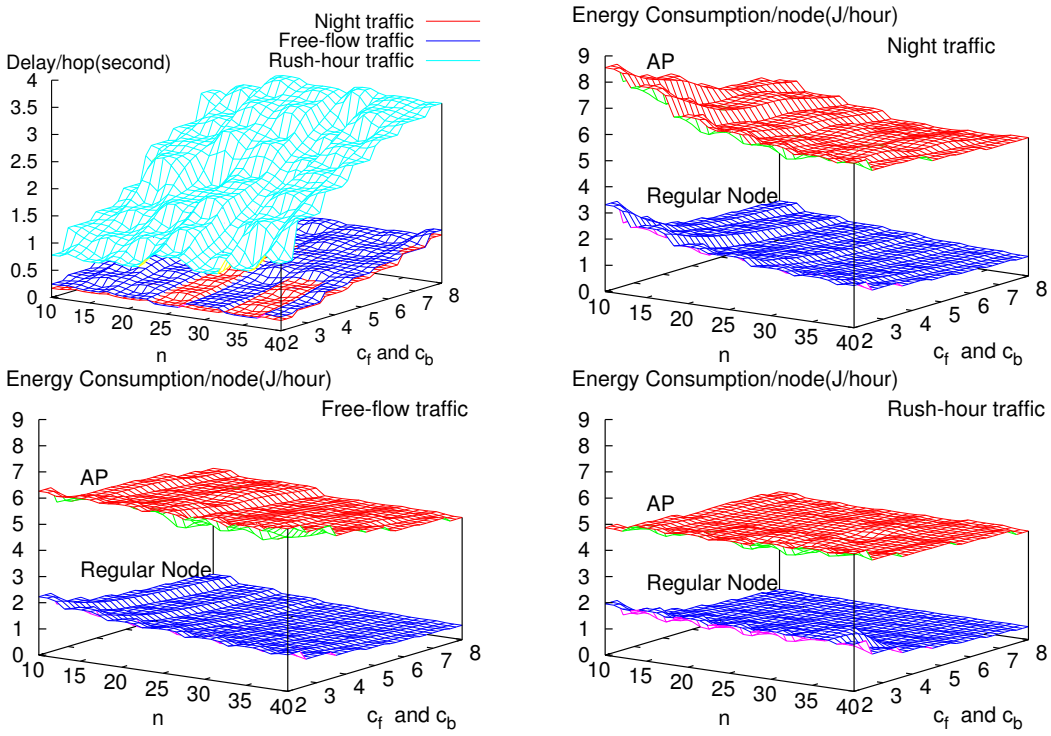


Figure 6.14: Impact of system parameters on performance in different traffic scenarios

head interacts with the WSN. As shown before, cluster arrival rate in rush-hour traffic is less than 1, which means the VANET in rush-hour traffic is almost always connected. The warning propagation can be always conducted via the VANET rather than the WSN.

Therefore, *the utilization of sensor nodes is inversely proportional to the traffic (or cluster) density. The energy consumption of both APs and regular nodes in the rush-hour traffic scenario is the lowest while that in the night traffic scenario is the highest.*

6.5.3.4 Average Propagation Speed and Estimated Node Lifetime

From the simulation, we can also obtain the average message propagation speed in different traffic categories and estimate the lifetime of sensor nodes. The results are shown in the table below. To estimate the lifetime, we take both sensing and communication energy consumption into account under the traffic dynamics of a day. For sensing, experiment has been conducted to measure the sensing energy consumed by passive infrared sensor [60]. Each sampling consumes about 8.7mJ. Each sensor node

samples the road condition every 10 seconds when it is activated. The communication energy consumption is obtained from the simulation by following the specification of Telosb mote [59]. Specifically, we consider the power for receiving(62.1 mW), transmitting (28.1mW), sleeping (3μW), idling (1.41mW) and transition (426μW), when transmission range is by default, i.e., 75$\sim$100m. Each sensor is equipped with 2$\times$AA batteries with $20,000J$ in total. For system parameters, $n = 20$ and $c_f = c_b = 5$.

Traffic scenario	$\hat{v_p}$(miles/hour)
Night traffic	425
Free-flow traffic	356
Rush-hour traffic	115

Table 6.5: Average propagation speed

Node type	Lifetime(days)
AP	145
Regular node	545

Table 6.6: Estimated nodal lifetime

6.6 Discussions

In this section, we discuss some issues related to the practicality of our design and possible enhancements.

Formation of super-group To further reduce the collision at AP, the AP can adjust the starting time of forward/backward propagations in two adjacent groups such that its communications with these groups do not conflict. An example is shown in Fig. 6.15, where three groups are connected by two APs. AP_0 may adjust the starting time of the propagation of Group 1 and then AP_1 can adjust that of Group 2. We refer to this kind of adjustment as *inter-group synchronization*. In this way, we can achieve contention-freeness at both AP_0 and AP_1. If we define a *super-group* as a set of continuous groups, then we can say, in this example, that the super-group formed by Group 0, 1 and 2 is inter-group synchronized and the communications in it is contention-free. Note that inter-group synchronization is much looser than time synchronization with each group because it only requires that the scheduling of two adjacent groups does not conflict at the AP connecting them. Hence, it is less costly to maintain.

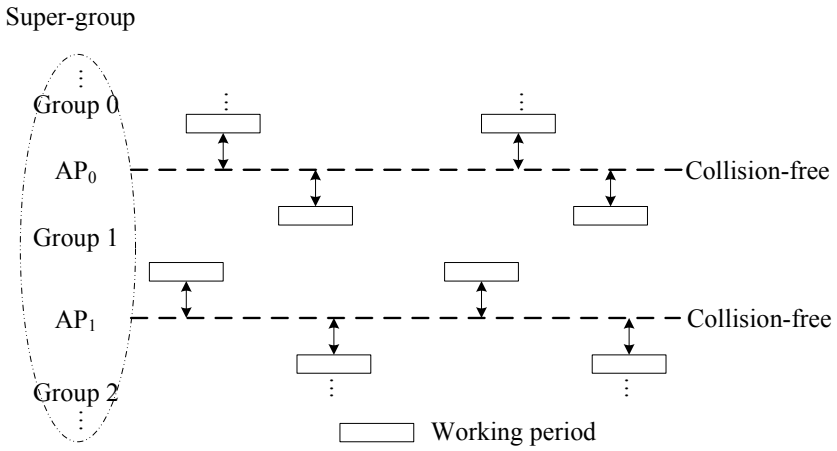


Figure 6.15: Loose time synchronization with a super-group

Communication at the intersection Another scenario we need to consider is the crossroad or intersection, where the sensor nodes on two different roads may interfere with each other. To avoid such interference, two solutions exist. One is that an AP is deployed at the crossing point to coordinate the communication among different groups (see Fig. 6.2) and ensure that no two groups being active in message propagation simultaneously. This is feasible since the actual forward/backward interval is often large; our large-scale simulation shows that typically only one actual forward/backward phase exists every 5 periods or more. Hence, one AP can handle at least 4 groups simultaneously, where each group is from one direction. The other solution is that multiple APs are deployed at the intersection and each of them is responsible for one road segment. CSMA/CA is used for the communication between any two APs to interconnect them.

Two-way road Although our designed protocols are described in the context of one-way highway, they can be extended to the two-way case without significant changes. Specifically, we only need the currently-designed backward propagation to also propagate activation messages and the currently-designed forward propagation to also propagate warning message for vehicles running in the reverse direction.

Adjustment of retransmission quota In Equation (6.1), we show the method to dynamically adjust the retransmission quota, r, for each group. However, in practice, the traffic on most roads follows a relatively stable pattern. Again, let us take the traffic on I-80 free way for example. Night traffic is

from 1AM to 5AM, free-flow traffic is from 6AM to 2PM and from 8PM to 12AM and rush-hour traffic is from 3PM to 7PM. The transition time between the different traffic categories is short. Thus, it is practical to pre-compute three values of r for each group and adjust it correspondingly whenever the traffic category changes, rather than adjusting it all the time.

Cluster formation One assumption used in this work is that the vehicles can form clusters to interact with the WSN. This is feasible in VANET since many research projects [62–64, 73] have investigated schemes for establishing and maintaining such clusters. Different from the existing work, our protocol does not require sophisticated cluster formation schemes. Specifically, each vehicle in our system only has two states, cluster head and cluster member, and the vehicle in the font is selected as the head.

Node failure One of the major problems for our protocol is that if one node fails due to drain of energy or other physical causes, network will be disconnected. To address this issue, a simple solution is to deploy some backup sensors around each network node. These backup nodes are pre-loaded with the same configuration during system bootstrapping. They randomly (with a relatively long period) wake up and check if the working node is still alive. If the working node is dead, that backup node will take over the work of the corresponding network node. In addition, it is possible that the information about node failure along with other information (e.g. power level) can be periodically reported to some administrative APs. In this way, we can keep track of the network status and know when and where sensors or batteries should be replaced. As our future work, we will study more efficient and reliable deployment of roadside WSN to better tolerate node failures.

Power replenishment Power supply for roadside sensors is also an issue. Since these sensors are accessible to people, we suggest that their batteries can be replaced every certain time period. As another direction of our future work, we will study how to extend the time interval between two consecutive battery replacements through rotating the AP roles among sensors to balance their energy consumption or other approaches.

Our current study focuses on communication aspect of the VANET-WSN system, the feasibility of practical, inexpensive, low-power and small-size sensing technology is also critical to the success of the system. Recently, such sensors that can be used to detect the presence of static or mobile targets

within tens to hundreds of feet have been commercially available. For example, EasySen [60] has commercialized WiEye, a tiny sensor board containing long-range passive infrared sensor with about 100 degree wide detection cone. In the near future, we plan to integrate these sensing units into our prototyping system, and study the impact of sensing activities on our design.

6.7 Summary

An integrated VANET-WSN system was proposed in this chapter to overcome the inherent limitations of pure VANET-based system and improve driving safety. Replacing expensive roadside stations with inexpensive roadside sensors makes the monitoring of road conditions more investment-efficient. Meanwhile, with the assistance of roadsides sensors, the communications between vehicles can be improved significantly, which is critical for delivery of safety-related information, especially in light-traffic scenarios. Prototype of the designed system has been implemented and tested in the field to verify its feasibility and efficiency. The simulation results indicate that, with appropriately chosen system parameters, satisfactory safety can be achieved.

CHAPTER 7. CONCLUSIONS AND FUTURE WORK

7.1 Research Contributions

In this dissertation, we have proposed practical solutions to improve resource efficiency for WLAN, MANET and VANET, respectively. Their effectiveness and efficiency have been demonstrated via extensive experimental and simulation studies. The major contributions are summarized as follows.

- *Utilizing ZigBee for more energy-efficient WLAN*

  In Chapter 3, we propose a ZigBee-assisted Power Saving Management (ZPSM) scheme through leveraging low-power ZigBee interface to wake up sleeping high-power WiFi interface on demand of current traffic. With ZPSM, we allow the high-power WiFi interfaces to sleep as long as possible, and use low-power ZigBee to wake up the WiFi interfaces only when they are needed for packet transmission. As our proposed ZPSM is built atop the standard PSM, it is compatible with the IEEE 802.11 standards. To the best of our knowledge, this is the first work that leverages the ZigBee interface to improve energy efficiency while ensuring delay requirements in WLAN. Moreover, our proposed scheme can support both short-delay and long-delay packet delivery, making it suitable for a wide range of applications. Simulation and prototype-based experiment results have shown that ZPSM can save energy significantly without violating delay requirements in various scenarios.

- *Utilizing ZigBee for more energy-efficient MANET*

  In Chapter 4, we propose to design a sustainable ad hoc network on lightweight mobile devices. To build such network, ZigBee-assisted Power Saving Management (ZPSM) has been designed for IEEE 802.11's DCF with the goal of maximizing network lifetime with stringent end-to-end packet delivery delay requirements. This work adopts the same idea of using low-power ZigBee

interface to wake up sleeping high-power WiFi interface. The objective is achieved through balancing nodal lifetime with the collaboration of ZigBee and WiFi interfaces. The simulation and experiment results show that the proposed ZPSM system can sustain for much longer time than the standard PSM and other existing systems under the same delay constraints.

- *Utilizing ZigBee for more bandwidth-efficient MANET*

  In Chapter 5, we propose a simple yet effective system for ZigBee-assisted WiFi transmission in high traffic density scenario. In this system, WiFi-enabled mobile devices leverage ZigBee communication to form clusters. Coordinated through ZigBee interfaces, members in each cluster run a TDMA-like protocol and take turns to transmit, achieving reduced contention and thereby improved bandwidth efficiency. Besides, the designed system is compatible with legacy mobile devices running the standard IEEE 802.11 protocols. Results of experiment and simulation have verified our design by showing that, the throughput, energy efficiency and fairness can be improved significantly at the same time.

- *Utilizing ZigBee for more resource-efficient VANET*

  In Chapter 6, to overcome the inherent limitations of pure VANET, we architecture an integrated network of vehicles and roadside sensors, which can ensure timely detection of dangerous road conditions and responsive communication among vehicles. New challenges are identified and addressed by designing effective and efficient schemes, which are scalable, flexible, energy-efficient and low-delay. To verify the viability of our proposed system, a prototype system has been implemented and tested in the field. Extensive simulations have also been conducted to evaluate system performance. The evaluation results show various design tradeoffs, and indicate that satisfactory safety and energy efficiency can be achieved simultaneously when system parameters are appropriately chosen.

7.2 Future Research Directions

In our research, we have developed a series of practical wireless systems to exploit the technical potential of ZigBee technologies in various aspects. With the aim of exploring new ZigBee-based

applications, we believe that this work opens the door to more extensive research study on ZigBee-based wireless networks. In this section, we briefly discuss about possible future research directions in this area.

7.2.1 Implementation of ZPSM

In this dissertation, two ZPSM schemes have been proposed to improve energy efficiency for WLAN and MANET, respectively. Although we have prototyped these two systems, both are implemented with certain level of simplifications to the originally designed systems. Thus, the performance may be different from more realistic systems. To make more practical evaluations on our proposed systems, a complete implementation is necessary. However, this is not an easy task due to the following reasons.

- *WLAN:* In our design, ZPSM is conceptually built on top of IEEE 802.11's standard PSM. However, concurrent implementation of PSM is accomplished through modifying the 802.11 subsystem as well as underlying drivers. Thus, real implementation of ZPSM on top of PSM is difficult because of the complexity of concurrent PSM implementation.

- *MANET:* Although IEEE 802.11's PSM for DCF has been standardized for years, it has not been fully implemented yet due to immaturity of ad hoc networking technologies. Most of the previous MANET research focuses largely on simulation based analysis or theoretical study. Very little has been done in terms of actual implementation and testing of protocols in real-world ad hoc networks. Thus, to implement ZPSM in ad hoc networks, we need to build up everything from scratch, which is pretty challenging. Possible challenges may include (i) how to synchronize network nodes; (ii) how to efficiently conduct multi-hop coordination; (iii) how to seamlessly integrate the membership management of ZigBee networks with that of WiFi networks running IEEE 802.11 DCF; (iv) how to handle mobility of network nodes with the assistance of ZigBee.

7.2.2 Extension of ZPSM

Although both ZPSM schemes have their specific application scenarios in our proposed systems, they can be further extended for more applications or higher efficiency.

- *WLAN:* In the proposed ZPSM system, we assume that the AP has unlimited energy supply. This is true in most of cases as AP is often static and connected to power supply. However, in some situations, where there are no fixed APs, mobile device may want to become a temporary AP to provide services. For example, people at an airport without Internet or free Internet want to play a client-server based game together. To do this, one mobile device can serve as an AP (i.e., temporary server) to host the game. In this case, the AP is also constrained by power and thereby may need energy-efficient scheme to save its power efficiently. To achieve this, there are a number of issues to be dealt with: (i) how to adapt or redesign ZPSM to improve efficiency for both AP and clients at the same time; (ii) how to coordinate among different APs in the presence of multiple APs; (iii) how to rotate the role of AP so as to maximize network lifetime; (iv) when and how to efficiently involve ZigBee interface for data transmission (especially in light traffic density scenarios).

- *MANET:* Similar to the PSM for DCF, our proposed ZPSM is transparent to upper layer routing protocols. To further maximize energy efficiency, we can design an energy-aware routing protocol to effectively choose energy-efficient routes that work best with ZPSM. Meanwhile, high mobility should be appropriately handled, as it may cause frequent disconnection and route repairing, which could degrade the system performance. Moreover, our designed system should also support multi-cast communication, which is very useful for message dissemination in emergency situations.

7.2.3 More Extensive Integration of VANET and WSN

As the advancement of ZigBee and sensor technology, our proposed VANET-WSN framework can be further extended or enhanced to support a wider range of applications.

- *More extensive interactions between ZigBee-enabled vehicles and roadside facilities:* Due to cost efficiency and easy deployment, ZigBee-based wireless devices are envisioned to replace traditional wired systems, which offers more opportunities for ZigBee-enabled vehicles to interact with ZigBee-based facilities for more applications. For example, ZigBee-based wireless light control system [74, 75]) has been proposed to reduce energy consumption and maintenance

costs. Another example is to attach low-cost ZigBee interfaces to parking meters or place them on the ground at parking lot to facilitate parking information queries and parking reservations.

- *Integration of on-board sensors and roadside sensors:* Typically, vehicles also carry sensors, called on-board sensors. Connecting such on-board sensors with roadside sensors can achieve real-time monitoring of vehicles, which is very important for many safety-related applications in VANETs. With on-board sensors, real-time status of vehicles (e.g., speed, distance from nearby vehicles, etc.) can be monitored by roadside WSN, which can either report these information to some central servers or inform nearby vehicles for more comprehensive situational awareness to avoid collision and other potential dangers.

Bibliography

[1] Wi-Fi Alliance, "WiFi," *http://www.wi-fi.org/*.

[2] Bluetooth SIG, "Bluetooth," *http://www.bluetooth.com/*.

[3] J. Anguelov, L. Zhao, and S. Petrov, "Evaluation of Zigbee Remote Sensor Networks," *http://www.sfu.ca/ lfz2/index3.html*.

[4] ZigBee Alliance, "ZigBee," *http://www.zigbee.org/*.

[5] Wikipedia, "ZigBee," *www.wikipedia.org/wiki/Zigbee*.

[6] "TazTag TPHONE," *http://www.nfc-phones.org/taztag-tph-one/*.

[7] R. Krashinsky and H. Balakrishnan, "Minimizing Energy for Wireless Web Access with Bounded Slowdown," *MobiCom '02*.

[8] D. Qiao and K. G. Shin, "Smart Power-Saving Mode for IEEE 802.11 Wireless LANs," *InfoCom '05*.

[9] P. Agrawal, A. Kumar, J. Kuri, and et al., "OPSM - Opportunistic Power Save Mode for Infrastructure IEEE 802.11 WLAN," *ICC '10*.

[10] V. Namboodiri and L. Gao, "Towards Energy Efficient VoIP over Wireless LANs," *MobiHoc '08*.

[11] Y. Agarwal, R. Chandra, A. Wolman, and et al., "Wireless Wakeups Revisited: Energy Management for VoIP over Wi-Fi Smartphones," *MobiSys '07*.

[12] B. Chen, K. Jamieson, R. Morris, and H. Balakrishnan, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *MobiCom '01*.

[13] Z. Li and B. Li, "Probabilistic Power Management for Wireless Ad Hoc Networks," *MONET '05*.

[14] R. Zheng and R. Kravets, "On-Demand Power Management for Ad Hoc Networks," *InfoCom '03*.

[15] C. Sengul and R. Kravets, "Conserving Energy with On-Demand Topology Management," *MASS '05*.

[16] S. Lim, C. Yu, and C. Das, "RandomCast: An Energy-Efficient Communication Scheme for Mobile Ad Hoc Networks," *TMC '09*.

[17] Y. Wang, W. Song, and et al., "LEARN: Localized Energy Aware Restricted Neighborhood Routing for Ad Hoc Networks," *SECON '06*.

[18] F. Rango, F. Guerriero, and P. Fazio, "Link-Stability and Energy Aware Routing Protocol in Distributed Wireless Networks," *TPDS '12*.

[19] Q. Li, J. Aslam, and D. Rus, "Online Power-aware Routing in Wireless Ad-hoc Networks," *MobiCom '01*.

[20] A. Sankar and Z. Liu, "Maximum Lifetime Routing in Wireless Ad-hoc Networks," *InfoCom '04*.

[21] A. Rao and I. Stoica, "An overlay MAC layer for 802.11 networks," *MobiSys '05*.

[22] D. Koutsonikolas, T. Salonidis, and et al, "TDM MAC Protocol Design and Implementation for Wireless Mesh Networks," *CoNEXT '08*.

[23] X. Lu, G. Fan, and R. Hao, "A Dynamic Token Passing MAC Protocol for Mobile Ad Hoc Networks," *IWCMC*, 2006.

[24] I. Liu, F. Takawira, and H. Xu, "A Hybrid Token-CDMA MAC Protocol for Wireless Ad Hoc Networks," *TMC '08*.

[25] ETSI, "Car 2 Car Communications," *http://www.car-to-car.org/*.

[26] University of California, Berkeley, "PATH project," *http://www.path.berkeley.edu/*.

[27] Government of Canada, "Transport canada," *http://www.tc.gc.ca/*.

[28] Y. Ding, C. Wang, and L. Xiao, "A static-node assisted adaptive routing protocol in vehicular networks," *VANET'07*.

[29] E. Weingatner and F. Kargl, "A Prototype Study on Hybrid Sensor-Vehicular Networks," *GI/ITG KuVS Fachgesprach "Wireless Sensor Networks"*, 2007.

[30] J. Bohli, A. Hessler, O. Ugus, and D. Westhoff, "A secure and resilient wsn roadside architecture for intelligent transport systems," *WiSec'08*.

[31] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *Wireless Communications, IEEE*, Oct. 2006.

[32] T. Kim, S., Y. Lee, and W. Hong, "Design and Evaluation of In-vehicle Sensor Network for Web based Control," *ECBS' 06*.

[33] T. Pering, Y. Agarwal, R. Gupta, and R. Want, "CoolSpots: Reducing the Power Consumption of Wireless Mobile Devices with Multiple Radio Interfaces," *MobiSys '06*.

[34] G. Ananthanarayanan and I. Stoica, "Blue-Fi: Enhancing Wi-Fi performance using Bluetooth signals," *MobiSys '09*.

[35] J. Yoo and et al., "A Cooperative Clustering Protocol for Energy Saving of Mobile Devices with WLAN and Bluetooth Interfaces," *TMC '11*.

[36] T. Jin, G. Noubir, and et al., "WiZi-Cloud: Application-transparent Dual ZigBee-WiFi Radios for Low Power Internet Access," *InfoCom '11*.

[37] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma, "ZiFi: Wireless LAN Discovery via ZigBee Interference Signatures," *MobiCom '10*.

[38] Y. Zhang and Q. Li, "HoWiES: A Holistic Approach to ZigBee Assisted WiFi Energy Savings in Mobile Devices," *InfoCom '13*.

[39] S. Kim, J. Chong, and et al., "Experiments on Interference and Coexistence between Zigbee and WLAN Devices Operating in the 2.4GHz ISM band," *NGPC' 05*.

[40] K. Shuaib, M. Boulmalf, and et al., "Co-existence of Zigbee and WLAN: A performance study," *Proc. IEEE/IFIP Int. Conf. Wireless & Optical Communications Networks*, 2006.

[41] Y. He and R. Yuan, "A Novel Scheduled Power Saving Mechanism for 802.11 Wireless LANs," *TMC '09*.

[42] F. Wang, Z. Liu, and X. Song, "Power-saving Mechanisms for Mobile Devices in Wireless Communications," *IET Commun. '09*.

[43] Silex Technology America, Inc., "Wireless Modules," *http://www.silexamerica.com/*.

[44] Texas Instruments Incorporated., "CC2530 RF transceiver for 2.4-GHz IEEE 802.15.4," *http://focus.ti.com/lit/ds/symlink/cc2530.pdf*.

[45] Multiband Atheros Driver for Wi-Fi, "MadWiFi," *http://www.madwifi.org/*.

[46] E. Rozner, V. Navda, R. Ramjee, and S. Rayanchu, "NAPman: Network-Assisted Power Management for WiFi Devices," *MobiSys '10*.

[47] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," *USENIX ATC '10*.

[48] J. Manweiler and R. Choudhury, "Avoiding the Rush Hours: WiFi Energy Management via Traffic Isolation," *MobiSys'11*.

[49] Nokia, "Nokia Energy Profiler," *www.developer.nokia.com/Resources/Tools_and_downloads/Other/Nokia_Energy_Profiler/*.

[50] C. Hu and J. Hou, "LISP: A Link-Indexed Statistical Traffic Prediction Approach to Improving IEEE 802.11 PSM," *ICDCS '04*.

[51] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad hoc Routing," *MobiCom '01*.

[52] P. Havinga and G. Smit, "Energy-efficient TDMA medium access control protocol scheduling," *AMOC '00*.

[53] J. Wan, D. Yuan, and X. Xu, "A review of cluster formation mechanism for clustering routing protocols," *ICCT '08*.

[54] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis And Enhancement," *INFOCOM '02*.

[55] Q. Ni, T. Li, T. Turletti, and Y. Xiao, "Saturation throughput analysis of error-prone 802.11 wireless," *JWCMC '05*.

[56] Texas Instruments Incorporated., "CC2420 RF transceiver," *http://www.chipcon.com/*.

[57] D. Qiao and K. G. Shin, "Achieving Efficient Channel Utilization and Weighted Fairness for Data Communications in IEEE 802.11 WLAN under the DCF," *IWQoS' 02*.

[58] D. Federal Highway Administration (FHWA), "Rural safety innovation program," *http://a257.g.akamaitech.net/7/257/2422/01jan20081800/edocket.access.gpo.gov/2008/E8-3716.htm*.

[59] CROSSBOW TECHNOLOGY INC., "Wireless Sensor Networks," *http://www.xbow.com/*.

[60] E. LLC, "Wieye - sensor board for wireless surveillance and security applications," *http://www.easysen.com/WiEye.htm*.

[61] S. Farm, "State Farm Statistics," *http://www.philadelphia-accident-lawyers.com/auto-car-accidents/deer-road-safety.html*, 2005.

[62] L. Bononi and M. D. Felice, "A Cross Layered MAC and Clustering Scheme for Efficient Broadcast in VANETs," *MASS'07*.

[63] Y. Gunter, B. Wiegel, and H. P. GroBmann, "Cluster-based Medium Access Scheme for VANETs," *ITSC*, 2007.

[64] I. Leontiadis and C. Mascolo, "Opportunistic Spatio-Temporal Dissemination System for Vehicular Networks," *MobiOpp '07*.

[65] I. Leontiadis, P. Costa, and C. Mascolo, "Persistent Content-based Information Dissemination in Hybrid Vehicular Networks," *PerCom '09*.

[66] P. Costa, D. Frey, M. Migliavacca, and L. Mottola, "Towards Lightweight Information Dissemination in Inter-Vehicular Networks," *VANET '06*.

[67] O. Tonguz, N. Wisitpongphan, F. Bai, P. Mudalige, and V. Sadekar, "Broadcasting in VANET," *INFOCOM '08*.

[68] W. Z. Song, R. Huang, B. Shirazi, and R. LaHusen, "TreeMAC: Localized TDMA MAC Protocol for Real-time High-data-rate Sensor Networks," *PerCom '09*.

[69] G. S. Ahn, E. Miluzzo, A. T. Campbell, S. G. Hong, and F. Cuomo, "Funneling-mac: A localized, sink-oriented mac for boosting fidelity in sensor networks," *SenSys '06*.

[70] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: a hybrid mac for wireless sensor networks," *SenSys '05*.

[71] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "High-throughput path metric for multi-hop wireless routing," *MOBICOM*, 2003.

[72] N. Wisitpongphan, F. Bai, P. Mudalige, V. Sadekar, and O. Tonguz, "Routing in Sparse Vehicular Ad Hoc Wireless Networks," *JSAC*, 2007.

[73] H. Su and X. Zhang, "Clustering-Based Multichannel MAC Protocols for QoS Provisionings Over Vehicular Ad Hoc Networks," *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY*, 2007.

[74] P. Elejoste, I. Angulo, A. Perallos, A. Chertudi, and et al., "An Easy to Deploy Street Light Control System Based on Wireless Communication and LED Technology," *Sensors Journal*, 2013.

[75] F. Leccese, "Remote-Control System of High Efficiency and Intelligent Street Lighting Using a ZigBee Network of Devices and Sensors," *IEEE Transactions on Power Delivery*, 2013.