Design and Analysis of WPAN Mesh with a Case Study of 802.15.5

Rui Zhang

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

2011

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Date	Professor **Myung J. Lee** Chair of Examining Committee
Date	Dean **Mumtaz Kassir** Executive Officer

Professor **Tarek N. Saadawi**
The City University of New York

Professor **Yi Sun**
The City University of New York

Professor **Bilal Khan**
The City University of New York

Professor **Mehmet Ulema**
Manhattan College

Supervision Committee

The City University of New York

Abstract

Design and Analysis of WPAN Mesh with a Case Study of 802.15.5

Rui Zhang

Advisor: Prof Myung J.Lee

Since the IEEE 802.15.4 has been introduced and widely used as PHY/MAC layer for wireless mesh network and wireless personal area network(WPAN Mesh), the challenge of wireless mesh networks has been sifted to upper layer as traditional network architecture can not meet the requirements posted by wireless mesh networks. Although a lot of research efforts have been put into this area, there is still big gap between theoretical work and industry realization. The motivation of this thesis is to propose the solution that can leverage the real industry application capability of wireless mesh networks and provide the guideline to the next generation wireless mesh network design and implementation.

The topics of this thesis will address both networking and transportation layer challenges. First, we will present the IEEE 802.15.5, the industrial standard mainly developed by our lab which provides mesh capability for wireless personal area network (WPAN) devices. This thesis will focus on the low rate part of IEEE 802.15.5 which uses IEEE 802.15.4 as PHY/MAC standard. We will present the mandatory function design of IEEE 802.15.5, the contents include network formation, addressing scheme, link state information exchange and routing. Finally, the prototype implementation in the test-bed will be

illustrated to demonstrate that the IEEE 802.15.5 will serve well for wireless personal area networks and wireless sensor networks.

The second part of thesis continue and enhance the works of IEEE 802.15.5, we further explore the fundamental capability of IEEE 8021.5.5 by providing a comprehensive analysis of IEEE 802.15.5. We will use four different performance metrics to calibrate the networking layer performance: scalability in view of memory space and energy consumption for routing establishment, fault tolerance represented by the node degree of next hop choices and Mean time to failure of routing path, routing path length compared with the shortest route, and the overhead associated with a mobility support. We will compare the performance of IEEE 802.15.5 with those of Zigbee PRO and the tree based scheme show the consistent superior performances of IEEE 802.15.5

The third part of thesis extends the IEEE 802.15.5 to address mobility case. We will first present the mobility support scheme in IEEE 802.15.5 and propose a distributed mobility support scheme for general wireless sensor networks. Evaluation is preformed and the results confirm the cost efficiency of proposed scheme.

The fourth part of thesis switch the focus to transportation layer, in which we target the most widely used TCP in wireless mesh network scenario. Instead of designing a new TCP algorithm customized to wireless mesh networks, we will propose Correlation-TCP, a TCP scheme based on TCP Newreno with an additional function which adjusts the TCP congestion window size based on correlation of congestion window size and RTT. The advantage of this new design lies in that the proposed TCP is compatible and fairly share bandwidth with current TCP algorithms in general wired Internets while has a much better performance in wireless mesh networks. Moreover, Correlation-TCP has the flexibility to meet various application layer requirements by tuning its parameters.

Final part of this thesis will conclude the thesis and provide future works.

Acknowledgements

First of all, I must express my sincere gratitude and thanks to my advisor and committee chair, Prof. Myung. J. Lee for his guidance, suggestions and encouragement through my Ph.D study and throughout the development of this thesis. Without his constant support, infinite patience and continuous encouragement, this thesis would not be possible. I always feel fortunate for having Prof. Myung. J. Lee as my Ph.D advisor. He showed me the discipline for successful Ph.D research, taught me skills for professional writing, guided me how to conduct excellent research, and spent countless hours, including the sacrifice of many of his spare time, discussing my research work and provided many valuable knowledge.

I want to show my great appreciation to Dr. Tarek N. Saadawi, Dr. Yi Sun, Dr. Bilal Khan and Dr. Mehmet Ulema, the members of my supervisory committee, for taking time to review this dissertation and make valuable comments on my work. I also want to appreciate Dr Ping Ji for her support in my PhD study. I am also grateful to Dean Mumtaz Kassir for providing me administrative advice throughout my graduate study.

I am fortunate to have the opportunity to work with a group of energetic people in Samsung joint research Lab. I have enjoyed every moment that we have worked together. All former and present members of Samsung joint Lab have taught me many things about life. I want especially to thank Dr. Yong Liu, Chunhui Zhu, Dr. Jianliang Zheng, Dr , Dr

Gasheop Ahn, yancheng Liu, June-Seung Yoon, Kim yang, Junjun Li, Baozhi Chen, for their friendships and their collective encouragement to finish this dissertation. I also want to thank my colleagues I have worked with in the Center for Information Networking and Telecommunications (CINT). They are Xijie Liu, Qihua Yang, Xiaohai Li and our secretary Ms. Natia.

Without the scholarships and the financial support from the CUNY, CCNY, Research Foundation of CUNY, and the sponsorship by Electronics and Telecommunications Research Institutes(ETRI) and the Samsung Advanced Institute of Technology, I would not be able to concentrate on this research. I want to thank them for financially supporting me.

Last, but not least, I would like to thank my parents for their love and support of my studying abroad, my wife - Wei Wu - for her love, sacrifice and full-hearted support of my work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Wireless Mesh/Sensor Networks

As the wireless communication technology emerged, devices with different wireless communication technology have been widely used in everyday life. Those large number of wireless device consist of different wireless network, such as most popular cellular networks and wireless LAN (WLAN). As various wireless networks evolve into the next generation to provide better services, a key technology, wireless mesh networks (WMNs), has emerged recently. Wireless mesh networks is a communications network made up of radio nodes organized in a mesh topology. The coverage area of the radio nodes working as a single network is sometimes called a mesh cloud. Access to this mesh cloud is dependent on the radio nodes working in harmony with each other to create a radio network. It is a dynamically self-organized and self-configured, with the nodes in the network automatically establishing and maintaining mesh connectivity among themselves (creating, in effect, an ad hoc network).

Most Wireless Mesh networks(WMNs) work based on the current existing PHY and

MAC. And they actually build a mesh layer on the MAC and PHY to mesh all wireless nodes which operate their own PHY and MAC standard. Right now, a number of new standards are under development in IEEE 802. These standards foresee the integration of routing and frame forwarding into the MAC layer, so that they operate transparently from the perspective of higher layers. To differentiate layer 3 (IP) from layer 2 (MAC)-based wireless routing, the latter one is denoted as path selection. Therefore, an IEEE 802 WMN appears as a single logical broadcast domain, and Address Resolution Protocol (ARP), Dynamic Host Configuration Protocol (DHCP), Spanning Tree, and any other layer 3 protocol can be supported. Furthermore, the MAC layer has the required information on the wireless medium. Thus, seamless WMN solutions have the ability to operate more efficiently than IP-based WMNs. Currently, 802.15.5 [4] and 802.11s [5] develop WMN solutions for wireless personal area networks (WPANs) and wireless local area networks (WLAN), respectively.

1.2 Challenge and Research Topics

As we have seen, the wireless mesh networks bring a new technology while new challenges other than traditional networks and. We briefly list some challenges for the research in Wireless Mesh Networks:

- Physical layer issues

Some advanced physical- layer techniques have been available for WMNs. Wireless radios of existing WMNs are able to support multiple transmission rates by a combination of different modulation and coding rates. With such modes, adaptive error resilience can be provided through link adaptation. Schemes such as orthogonal frequency multiple

access (OFDM) and ultra-wide band (UWB) techniques are being used to support high-speed transmissions. In order to further increase capacity and mitigate the impairment by fading, delay-spread, and co-channel interference, multi-antenna systems such as antenna diversity, smart antenna, and MIMO systems, have been proposed for wireless communications. Although these physical-layer techniques are also desired by other wireless networks, it is a more challenging problem to develop such techniques for WMNs. For example, mesh networking among multiple nodes makes the system model much more complicated than that of a conventional MIMO system in wireless LANs or cellular networks.

Open issues in the physical layer are twofold:

1. It is necessary to further improve the transmission rate and the performance of physical-layer techniques. New wideband transmission schemes other than OFDM or UWB are needed in order to achieve higher transmission rate in a larger- area network. Multiple-antenna systems have been researched for years. However, their complexity and cost are still too high to be widely accepted for commercialization. Frequency-agile techniques are still in their early phase, and tremendous research efforts are needed before they can be accepted for commercial use.

2. To best utilize the advanced features provided by the physical layer, higher-layer protocols, especially MAC protocols, need to work interactively with the physical layer. Consequently, some components in the physical layer must be designed in a way that higher layers can access or control them. This makes hardware design more challenging and also motivates the innovation of low-cost software radio techniques.

- MAC layer issues

There exist differences between the MAC in WMNs and the classical counterparts for wireless networks:

1. MAC for WMNs is concerned with more than one-hop communication.

2. MAC is distributed, needs to be collaborative, and works for multipoint-to-multipoint communication.

3. Network self-organization is needed for better collaboration between neighboring nodes and nodes in multi-hop distances.

4. Mobility is low but still affects the performance of MAC.A MAC protocol for WMNs can be designed to work on a single channel or multiple channels simultaneously.

There exist the following major challenging issues.

1. Scalable MAC. To the best of our knowledge, the scalability issue in multi-hop ad hoc networks has not been fully solved yet. Most existing MAC protocols only solve partial problems of the overall issue, but raise other problems. To make the MAC protocol really scalable, new distributed and collaborative schemes must be proposed to ensure that network performance (e.g., throughput and even QoS parameters such as delay and delay jitter) will not degrade as network size increases. It is obvious that a multi-channel MAC protocol can achieve higher throughput than a singlechannel MAC. However, to really achieve spectrum efficiency and improve the per-channel throughput, the scalable MAC protocol needs to consider the overall performance improvement in multiple channels. Thus, developing a scalable multi-channel MAC is a more challenging task than a single-channel MAC.

2. MAC/Physical Cross-Layer Design. When advanced physical layer techniques, such as MIMO and cognitive radios, are used, novel MAC protocols, especially multi-channel MAC, need to be proposed to utilize the agility provided by the physical layer.

3. Network Integration in the MAC Layer. Mesh routers in WMNs are responsible for the integration of various wireless technologies. Thus, advanced bridging functions must be developed in the MAC layer so that different wireless radios, such as IEEE 802.11, 802.16, 802.15, etc., can seamlessly work together. econfigurable/software radios and the related radio resource management schemes may be the ultimate solution to these bridging functions.

- Network layer (routing) issues

Despite the availability of many routing protocols for ad hoc networks, the design of routing protocols for WMNs is still an active research area. We believe that an optimal routing protocol for WMNs must capture the following features:

1. Multiple Performance Metrics. Many existing routing protocols use minimum hop-count as a performance metric to select the routing path. This has been demonstrated to be ineffective in many situations.

2. Scalability. Setting up or maintaining a routing path in a very large wireless network may take a long time. Thus, it is critical to have a scalable routing protocol in WMNs.

3. Robustness. To avoid service disruption, WMNs must be robust to link failures or congestion. Routing protocols also need to perform load balancing.

4. Efficient Routing with Mesh Infrastructure. Considering the minimal mobility and no constraints on power consumption in mesh routers, the routing protocol in mesh routers is expected to be much simpler than ad hoc network routing protocols. With the mesh infrastructure provided by mesh routers, the routing protocol for mesh clients can also be made simple. Existing routing protocols for ad hoc networks have already considered some of these features. However, none of them has captured all of these features, so routing algorithms targeted for wireless mesh networks are desired.

For a routing protocol of WMNs, several research issues still remain unresolved.

1. Scalability. Hierarchical routing protocols can only partially solve this problem due to their complexity and difficulty of management. Geographic routing relies on the GPS or similar positioning technologies, which increases cost and complexity of WMNs. Thus, new scalable routing protocols need to be developed.

2. Better Performance Metrics. New performance metrics need to be developed. Also, it is necessary to integrate multiple performance metrics into a routing protocol so that the optimal overall performance is achieved.

3. Routing/MAC Cross-Layer Design. A routing protocol needs to interact with the MAC layer in order to improve its performance. Adopting multiple performance metrics from layer-2 into routing protocols is an example. However, interaction between MAC and routing layers is so close that merely exchanging parameters between them is not adequate. Merging certain functions of MAC and routing protocols is a promising approach.

4. Efficient Mesh Routing. With the mesh infrastructure, a much simpler and more

efficient routing protocol than an ad hoc network routing protocol needs to be developed for WMNs.

- Transportation layer issues

To date, no transport protocol has been proposed specifically for WMNs. However, a large number of transport protocols are available for ad hoc networks. Studying these protocols helps in the design of transport protocols for WMNs. Different transport protocols are needed for non-real-time and real-time traffic. Reliable transport protocols can be further classified into two types: TCP variants and new transport protocols. TCP variants improve the performance of the classical TCPs by tackling the following problems:

1. Non-Congestion Packet Losses. The classical TCPs do not differentiate congestion and non-congestion losses. As a result, when non-congestion losses occur, the network throughput quickly drops due to unnecessary congestion avoidance. In addition, when wireless channels return to normal operation, the classical TCP cannot be recovered quickly. A feedback mechanism can be used to differentiate different packet losses.

2. Unknown Link Failure. Link failure occurs frequently in mobile ad hoc networks, since all nodes are mobile. As far as WMNs are concerned, link failure is not as critical as in mobile ad hoc networks, because the WMN infrastructure avoids the issue of single-point-of-failure. However, due to wireless channels and mobility in mesh clients, link failure may still happen. To enhance TCP performance, link failure needs to be detected.

3. Network Asymmetry. Network asymmetry is defined as the situation in which the forward direction of a network is significantly different from the reverse direction

in terms of bandwidth, loss rate, and latency. Thus, it impacts the transmission of ACKs. Since TCP is critically dependent on ACK, its performance can be severely degraded by network asymmetry. Although schemes such as ACK filtering, ACK congestion control, etc. help to solve the network asymmetry problem, whether they are applicable to WMNs needs investigation.

4. Large RTT Variations. Considering mobility, variable link quality, fluctuating traffic load, and other factors in WMNs, the change of routing path may be frequent and may cause large variations of RTT. This will degrade the TCP performance, because the normal operation of TCP relies on a smooth measurement of RTT.

To further improve performance of transport protocols, researchers have started to develop entirely new transport protocols. However, for WMNs an entirely new transport protocol is not a favorable solution. WMNs will be integrated with the Internet and many other wireless networks, and thus transport protocols for WMNs need to be compatible with TCPs.

For reliable transport in WMNs, in addition to better solutions to the above mentioned problems, several other issues need further investigation.

1. Cross-layer Solution to Network Asymmetry. All problems of TCP performance degradation are actually related to protocols in the lower layers. For example, it is the routing protocol that determines the path for both TCP data and ACK packets. To avoid asymmetry between data and ACK packets, it is desired that a routing protocol selects an optimal path for both data and ACK packets. Moreover, the link-layer performance directly impacts the packet loss ratio. In order to reduce the possibility of network asymmetry, MAC and error control need to treat TCP data and ACK packets differently.

2. Adaptive TCP. WMNs will also be integrated with the Internet and various wireless networks such as IEEE 802.11, 802.16, 802.15, etc. The heterogeneity among these networks renders the same TCP ineffective for all networks. Applying different TCPs in different networks is a complicated and costly approach. As a result, adaptive TCP is the most promising solution. Thus, adaptive transport protocols need to be proposed for WMNs. For real-time transport, entirely new RCPs need to be developed by considering the features of WMNs. In addition, new loss differentiation schemes must be developed to work together with RCPs. Since WMNs will be integrated with various wireless networks and the Internet, adaptive rate control protocols are also needed.

- Network management

Many management functions are needed to maintain the appropriate operation of WMNs. Here we just list one major network management issue: Mobility Management. A distributed mobility management scheme is needed for WMNs. However, because of the existence of a backbone network, a distributed scheme for WMNs can be simpler than that for mobile ad hoc networks. How to take advantage of the network backbone to design a lightweight distributed mobility management scheme for WMNs needs further investigation. Mobility management is closely related to multiple layers of network protocols, so developing multi-layer mobility management schemes is another interesting research topic. Location service is a desired feature by WMNs. Location information can enhance the performance of MAC and routing protocols, and it can help to develop promising location related applications. Proposing accurate or efficient algorithms for location service is still an open research topic.

1.3 Industrial Standard Activities

1.3.1 IEEE 802.15.4

A milestone of the development of wireless Mesh PAN networks(WPAN Mesh) and Wireless Sensor networks(WSNs) is the release of IEEE 802.15.4 standard[2] in 2003. IEEE 802.15.4 is the first global standard for wireless personal area networks and wireless sensor networks (simply referred to as wireless sensor networks hereafter).

IEEE standard 802.15.4 intends to offer the fundamental lower network layers of a type of wireless personal area network (WPAN) which focuses on low-cost, low-speed ubiquitous communication between devices (in contrast with other, more end user-oriented approaches, such as Wi-Fi). The emphasis is on very low cost communication of nearby devices with little to no underlying infrastructure, intending to exploit this to lower power consumption even more.

The basic framework conceives a 10-meter communications range with a transfer rate of 250 kbit/s. Tradeoffs are possible to favor more radically embedded devices with even lower power requirements, through the definition of not one, but several physical layers. Lower transfer rates of 20 and 40 kbit/s were initially defined, with the 100 kbit/s rate being added in the current revision. Figure 1.1 shows the IEEE 802.15.4 radio range and power consumption compared with other standards. Devices are conceived to interact with each other over a conceptually simple wireless network. The definition of the network layers is based on the OSI model; although only the lower layers are defined in the standard, interaction with upper layers is intended, possibly using a IEEE 802.2 logical link control sublayer accessing the MAC through a convergence sublayer. Implementations may rely on external devices or be purely embedded, self-functioning devices.

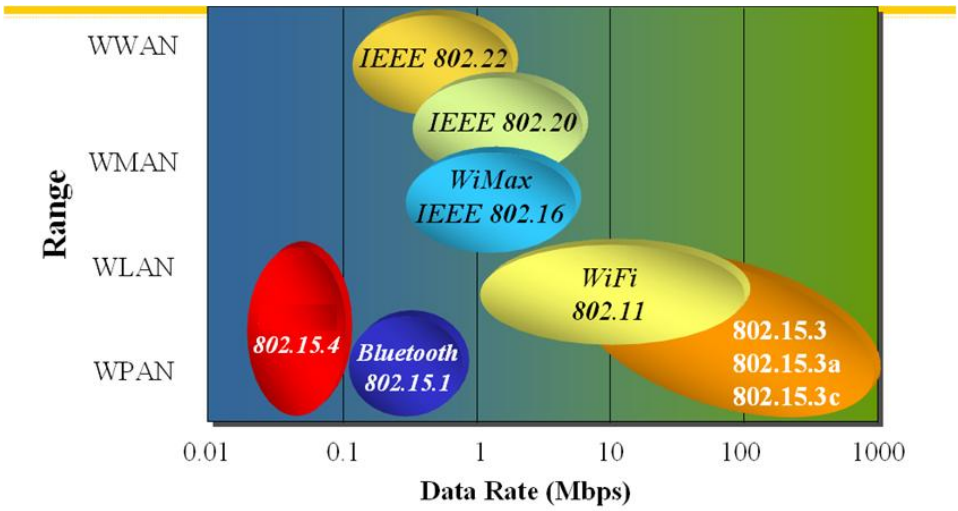


Figure 1.1: IEEE 802.15.4 radio range and power consumption compared with other standards.

- PHY Layer: The physical layer (PHY) ultimately provides the data transmission service, as well as the interface to the physical layer management entity, which offers access to every layer management function and maintains a database of information on related personal area networks. Thus, the PHY manages the physical RF transceiver and performs channel selection and energy and signal management functions. It operates on one of three possible unlicensed frequency bands: 868.0-868.6 MHz: Europe, allows one communication channel. 902-928 MHz: North America, up to ten channels, extended to thirty. 2400-2483.5 MHz: worldwide use, up to sixteen channels. Direct sequence spread spectrum (DSSS) technique and O-QPSK modulation are used in PHY layer, which can support up to 250K bps rate.

- MAC Layer: The medium access control (MAC) enables the transmission of MAC frames through the use of the physical channel. IEEE 802.15.4 provides two different MAC schemes: Beacon enabled and Non-Beacon mode. The first mode en-

able MAC beacon signal to provide the time slot and TDMA based communication. While Non-Beacon mode uses CSMA/CA to random access channel.

As a Physical layer and MAC sublayer standard, IEEE 802.15.4 does not specify how to provision multi-hop routing (a.k.a. meshing) capabilities for wireless sensor networks, assuming it is the task of upper layers. Several other industrial standards organizations also come into play to support mesh functions on top of IEEE 802.15.4. We will briefly go through several widely used ones.

1.3.2 Zigbee(PRO)

ZigBee[5], an industrial alliance, is among the first who have been working on upper layers to provide multi-hop routing functionalities for wireless sensor networks.

ZigBee is a specification for a suite of high level communication protocols using small, low-power digital radios based on the IEEE 802.15.4 standard for Low-Rate Wireless Personal Area Networks (LR-WPANs), such as wireless light switches with lamps, electrical meters with in-home-displays, consumer electronics equipment via short-range radio needing low rates of data transfer. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other WPANs, such as Bluetooth, and provide the mesh capability to extend the coverage of network.

There are three different types of ZigBee devices:

- ZigBee coordinator (ZC): The most capable device, the coordinator forms the root of the network tree and might bridge to other networks. There is exactly one ZigBee coordinator in each network since it is the device that started the network originally. It is able to store information about the network, including acting as the Trust Center and repository for security keys.

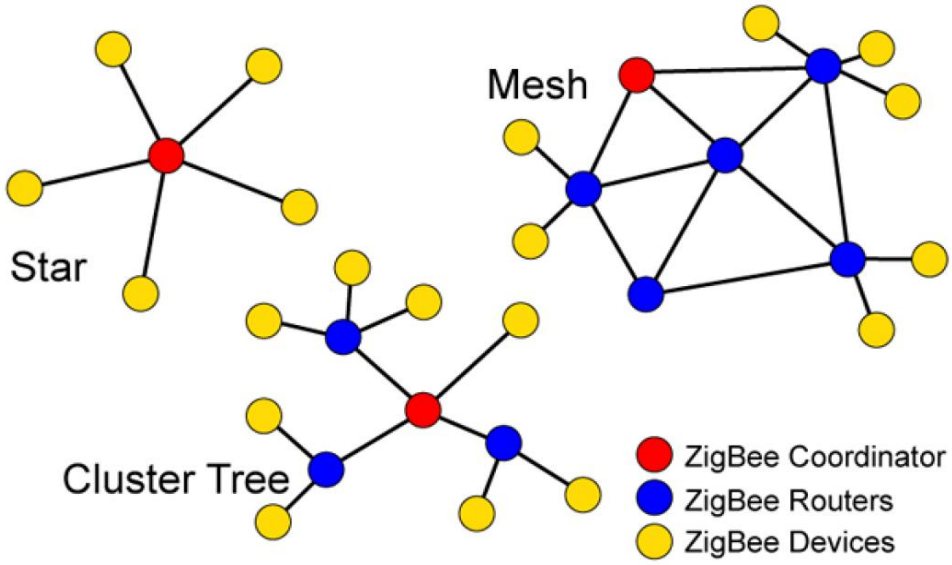


Figure 1.2: Zigbee Network topologies: Star, Cluster Tree and Mesh.

- ZigBee Router (ZR): As well as running an application function, a router can act as an intermediate router, passing on data from other devices.

- ZigBee End Device (ZED): Contains just enough functionality to talk to the parent node (either the coordinator or a router); it cannot relay data from other devices. This relationship allows the node to be asleep a significant amount of the time thereby giving long battery life. A ZED requires the least amount of memory, and therefore can be less expensive to manufacture than a ZR or ZC.

Zigbee network supports three different topologies. Figure 1.2 shows three fundamental topologies of zigbee networks: Star, Cluster Tree and Mesh. Star topology define the one hop network architecture which is similar to bluetooth and Wifi network, but with a much lower data rate and power consumption. In the meantime, Cluster Tree and Mesh topologies and multi-hop and they are unique for Zigbee.

ZigBee is based upon stack architecture that resembles standard OSI seven-layer

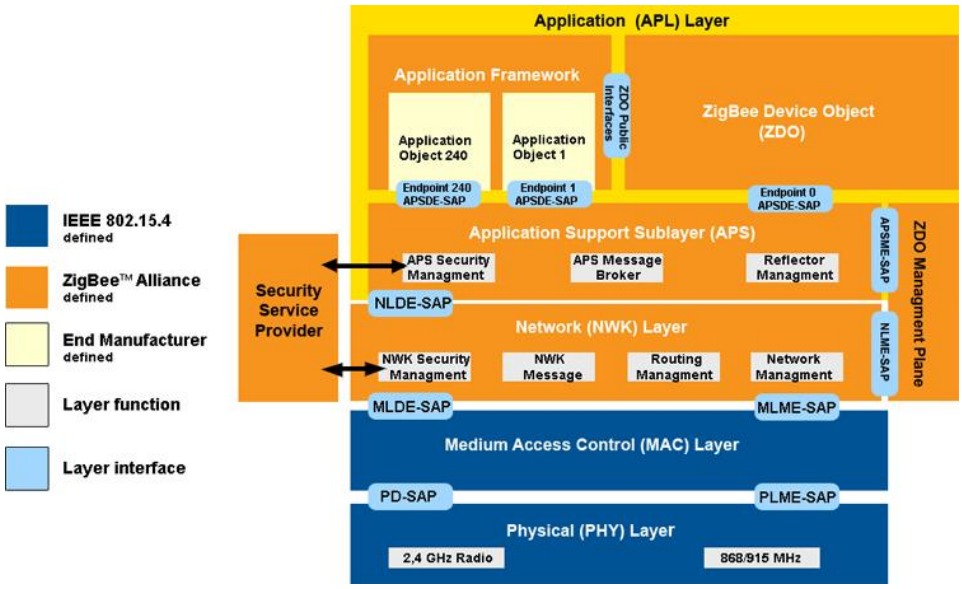


Figure 1.3: ZigBee Stack Architecture.

model but defines only those layers relevant to achieving functionality in the intended scope. Figure 1.3 illustrates the Zigbee stack architecture. The ZigBee stack architecture is made up of a set of blocks called layers. Each layer performs a specific set of services for the layer above: a data entity provides a data transmission service and a management entity provides all other services. Each service entity exposes an interface to the upper layer through a service access point (SAP), and each SAP supports a number of service primitives to achieve the required functionality. IEEE 802.15.4 standard defines the lower two layers: the physical (PHY) layer and the medium access control (MAC) sub-layer. The ZigBee Alliance builds on this foundation by providing the network (NWK) layer and the framework for the application layer, which includes the application support (APS) sub-layer, the ZigBee device object (ZDO) and the manufacturer-defined application objects.

The responsibilities of the ZigBee NWK layer include mechanisms used to join and leave a network, to apply security to frames and to route frames to their intended desti-

nations. In addition, the discovery and maintenance of routes between devices devolve to the NWK layer. Also the discovery of one-hop neighbors and the storing of pertinent neighbor information are performed by the NWK layer. The NWK layer of a ZigBee coordinator is responsible for starting a new network, when appropriate, and assigning addresses to newly associated devices. For more information about the NWK layer please refer to document of ZigBee specification v.1.0. The ZigBee application layer consists of the APS sub-layer, the ZDO (containing the ZDO management plane), and the manufacturer-defined application objects. The responsibilities of the APS sub-layer include maintaining tables for binding, which is the ability to match two devices together based on their services and their needs, and forwarding messages between bound devices. The responsibilities of the ZDO include defining the role of the device within the network (e.g., ZigBee coordinator or end device), discovering devices on the network and determining which application services they provide, initiating and/or responding to binding requests and establishing a secure relationship between network devices.

Zigbee application level Standards were custom-designed by industry experts to meet the specific market needs of businesses and consumers. These standards give product manufacturers a straightforward way to help their customers gain greater control of, and even improve, everyday activities. They take full advantage of ZigBee's many strengths so products using ZigBee low-power wireless standards can be easily installed and allowed to run on harvested energy or batteries for years. This simplicity makes them easy to use and gives consumers and businesses the tools they need to have greener homes and offices. The major Zigbee application level Standards include the following:

- ZigBee Building Automation (Efficient commercial spaces): It offers a global standard for interoperable products enabling the secure and reliable monitoring and

control of commercial building systems. It is the only BACnet03 approved wireless standard for commercial buildings. Owners, operators and tenants can benefit from increased energy savings and ensure the lowest lifecycle costs with this green and easy-to-install robust wireless network. By using ZigBee Building Automation products in your building, you can even qualify for LEED credits because you reduce or even eliminate wiring and conduit.

- ZigBee Remote Control (Advanced remote controls): It provides a global standard for advanced, greener and easy-to-use RF remotes that removes line-of-sight restrictions while also delivering two-way communication, longer range of use and extended battery life. It was designed for a variety of consumer electronic devices including HDTV, home theater equipment, set-top boxes and other audio equipment.

- ZigBee Smart Energy (Home energy savings): It provides interoperable products that monitor, control, inform and automate the delivery and use of energy and water. It helps create greener homes by giving consumers the information and automation needed to easily reduce their consumption and save money, too.

- ZigBee Health Care (Health and fitness monitoring): It provides interoperable products enabling secure and reliable monitoring and management of non-critical, low-acuity healthcare services targeted at chronic disease, aging independence and general health, wellness and fitness.

- ZigBee Home Automation (Smart homes): It provides interoperable products enabling smart homes that can control appliances, lighting, environment, energy management and security, as well as the expandability to connect with other ZigBee

networks.

ZigBee 2007, now the current stack release, contains two stack profiles, stack profile 1 (simply called ZigBee), for home and light commercial use, and stack profile 2 (called ZigBee Pro). ZigBee Pro offers more features, such as multi-casting, many-to-one routing and high security with Symmetric-Key Key Exchange (SKKE), while ZigBee (stack profile 1) offers a smaller footprint in RAM and flash. Both offer full mesh networking and work with all ZigBee application profiles.

1.3.3 6LoWPAN(IPv6 over Low Power WPAN)

6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) and ROLL (Routing Over Low power and Lossy networks), two IETF working groups of the Internet Engineering Task Force (IETF), have been trying to extend IPv6 to wireless personal area networks. The 6LoWPAN concept originated from the idea that ”the Internet Protocol could and should be applied even to the smallest devices,” and that low-power devices with limited processing capabilities should be able to participate in the Internet of Things.

The 6lowpan group has defined encapsulation and header compression mechanisms that allow IPv6 packets to be sent to and received from over IEEE 802.15.4 based networks. The major functions for 6LowPAN are listed as following:

- Adapting the packet sizes of the two networks: IPv6 requires the maximum transmission unit (MTU) to be at least 1280 Bytes. In contrast, IEEE802.15.4's standard packet size is 127 octets. A maximum frame overhead of 25 octets spares 102 octets at the media access control layer. An optional but highly recommended security feature at the link layer poses an additional overhead. For example, 21 octets are consumed for AES-CCM-128 leaving only 81 octets for upper layers

- Address resolution: IPv6 nodes are assigned 128 bit IP addresses in a hierarchical manner, through an arbitrary length network prefix. IEEE 802.15.4 devices may use either of IEEE 64 bit extended addresses or, after an association event, 16 bit addresses that are unique within a PAN. There is also a PAN-ID for a group of physically collocated IEEE802.15.4 devices.

- Adaptation layer for interoperability and packet formats: An adaptation mechanism to allow interoperability between IPv6 domain and the IEEE 802.15.4 can best be viewed as a layer problem. Identifying the functionality of this layer and defining newer packet formats, if needed, is an enticing research area. RFC 4944 [33] proposes an adaptation layer to allow the transmission of IPv6 datagrams over IEEE 802.15.4 networks.

- Addressing management mechanisms: The management of addresses for devices that communicate across the two dissimilar domains of IPv6 and IEEE802.15.4 is cumbersome, if not exhaustingly complex.

- Routing considerations and protocols for mesh topologies in 6lowpans: Routing in 6lowPAN is a two phased problem that is being considered for low-power IP networking. First is the mesh routing in the personal area network (PAN) space. Second is the routability of packets between the IPv6 domain and the PAN domain.

- Device and service discovery: Since IP-enabled devices may require the formation of ad hoc networks, the current state of neighboring devices and the services hosted by such devices will need to be known. IPv6 neighbour discovery extensions is an internet draft proposed as a contribution in this area.

The IPSO Alliance is the primary advocate for IP networked devices for use in energy,

consumer, healthcare and industrial applications.

1.3.4 Wireless HART and ISA 100

Another two important players are the HART (Highway Addressable Remote Transducer) Communication Foundation (HCF) and the International Society of Automation (ISA), both having been retrofitting their control and automation protocols to accommodate automation industries with wireless solutions.

WirelessHART is a wireless mesh network communications protocol for process automation applications. It adds wireless capabilities to the HART Protocol while maintaining compatibility with existing HART devices, commands, and tools. Each WirelessHART network includes three main elements as illustrated in Figure 1.4.

- Wireless field devices connected to process or plant equipment. This device could be a device with WirelessHART built in or an existing installed HART-enabled device with a WirelessHART adapter attached to it.

- Gateways enable communication between these devices and host applications connected to a high-speed backbone or other existing plant communications network.

- A Network Manager is responsible for configuring the network, scheduling communications between devices, managing message routes, and monitoring network health. The Network Manager can be integrated into the gateway, host application, or process automation controller.

Each device in the mesh network can serve as a router for messages from other devices. In other words, a device doesn't have to communicate directly to a gateway, but just

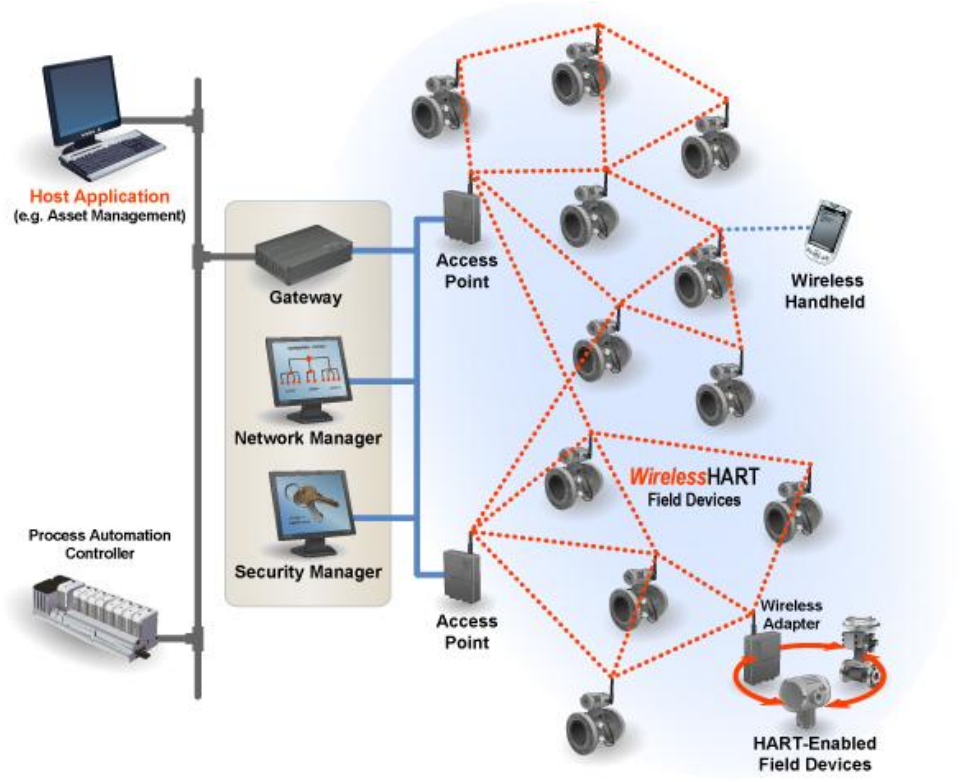


Figure 1.4: Wireless HART Network Architecture.

forward its message to the next closest device. This extends the range of the network and provides redundant communication routes to increase reliability.

The Network Manager determines the redundant routes based on latency, efficiency and reliability. To ensure the redundant routes remain open and unobstructed, messages continuously alternate between the redundant paths. Consequently, like the Internet, if a message is unable to reach its destination by one path, it is automatically re-routed to follow a known-good, redundant path with no loss of data.

The mesh design also makes adding or moving devices easy. As long as a device is within range of others in the network, it can communicate. For flexibility to meet different application requirements, the WirelessHART standard supports multiple messaging

modes including one-way publishing of process and control values, spontaneous notification by exception, ad-hoc request/response, and auto-segmented block transfers of large data sets. These capabilities allow communications to be tailored to application requirements thereby reducing power usage and overhead.

1.4 Overview and Outline of Thesis

In this thesis, we will tackle the WPAN Mesh network challenges from networking and transportation levels and the content will include both research works and industrial standard activity. The organization of this thesis is as below.

Chapter 1 is the introduction of the thesis. It sets forth the background, related research work and industrial standard activities, overview and outline of thesis.

Chapter 2 introduces a new IEEE standard mainly developed by our lab, IEEE 802.15.5, which provides the architectural framework enabling WPAN devices to promote interoperable, stable, and scalable wireless mesh topologies. This chapter will focus low rate part of IEEE 802.15.5, which uses IEEE 802.15.4 as MAC/PHY, present the mandatory functions of IEEE 802.15.5 low rate part, which include tree formation, address block, link state, and uni-casting schemes, and the implementation works based on our test-bed which is deployed over a whole floor (100 x 140 ft2) at CUNY Engineering building. The performance results testify that the IEEE 802.15.5 will serve well what it is designed for.

Chapter 3 will continue the works of IEEE 802.15.5 part, addressing the theoretical analyses on the fundamental capability of IEEE 802.15.5. Four important performance metrics will be investigated: scalability in view of memory space and energy consumption for routing establishment, fault tolerance represented by the node degree of next hop choices and Mean time to failure of routing path, routing path length compared with the

shortest route, and the overhead associated with a mobility support. We will compare the performance of IEEE 802.15.5 with those of Zigbee PRO and the tree based scheme and demonstrate that IEEE 802.15.5 consistently maintains superior performances in all measures investigated and is also capable of meeting different network requirements by adjusting the radius of the local link state.

Chapter 4 extends the former works of IEEE 802.15.5 to a more general wireless mesh/sensor network scenario, in which we will discuss the mobility issue involved. We will first introduce the portability solution in IEEE 802.15.5 and then apply the idea to the sink node mobility support scheme in wireless sensor networks and propose a distributed mobile sink support in wireless sensor networks.

Chapter 5 will switch the research area from networking layer to transportation layer, in which we will target the most widely used TCP and its limitation in wireless mesh networks. This chapter will present a new TCP design-Correlation TCP, which uses correlation between RTT and congestion window size to detect congestion and adjust the congestion window size based on that. Contrast to other TCP designed specifically for wireless mesh network environment, Correlation-TCP is compatible with current widely used Newreno and has much better performance in wireless mesh networks.

Finally, Chapter 6 will conclude the whole works of the thesis and list several future works.

Chapter 2

IEEE 802.15.5-Low Rate Part

2.1 Overview

In this chapter, we will introduce a new IEEE standard, IEEE 802.15.5, which provides mesh capability for wireless personal area network (WPAN) devices. The standard provides an architectural framework enabling WPAN devices to promote interoperable, stable, and scalable wireless mesh topologies. It is composed of two parts: low-rate WPAN mesh and high-rate WPAN mesh. In this paper, we present only low-rate WPAN mesh because it is designed to support wireless sensor networks. IEEE 802.15.5 low-rate part is a light-weight scalable mesh routing protocol that caters well to the requirements of resource-constrained wireless sensor networks. By binding logical addresses to the network topology, IEEE 802.15.5 obviates the need for route discovery. This eliminates the initial route discovery latency, saves storage space and reduces the communication overhead and energy consumption. A distributed link state scheme is further built atop the block addressing scheme to improve the quality of routes, robustness, and load balancing. The routing scheme scales well with regard to various performance metrics. The standard

also provides enhanced functions such as multicast, reliable broadcast, power saving, time synchronization, route tracing and portability. We also present the performance evaluation of major functions performed with a 50-nodes tested deployed over a whole floor (100 x 140 ft2) at CUNY Engineering building. The results testify that the IEEE 802.15.5 will serve well for wireless personal area networks and wireless sensor networks.

2.2 Background and Motivation

A milestone of the development of wireless sensor networks is the release of IEEE 802.15.4 standard[2] in 2003. IEEE 802.15.4 is the first global standard for wireless personal area networks and wireless sensor networks (simply referred to as wireless sensor networks hereafter). As a Physical layer and MAC sublayer standard, IEEE 802.15.4 does not specify how to provision multi-hop routing (a.k.a. meshing) capabilities for wireless sensor networks, assuming it is the task of upper layers. IEEE has recently released IEEE 802.15.5 standard[3] to provide multi-hop mesh functions. This standard is tightly coupled with IEEE 802.15.4 to take full advantage of IEEE 802.15.4 while maintaining simplicity. In this section, we introduce the features and functions of this new standard. The standard is composed of two parts: low-rate WPAN mesh and high-rate WPAN mesh. Our discussion is however confined to the low-rate part because it is designed to support wireless sensor networks. We also present the performance evaluation of the low-rate part using a 50-node tested deployed over a whole floor (100 x 140 ft2) at CUNY Engineering building.

To see where IEEE 802.15.5 stands, let us first highlight the features of the aforementioned solutions in the context of mesh networking. ZigBee ratified its first standard in 2004. Before another distinct standard called ZigBee Pro was released in 2007, ZigBee

routing comprised AODVjr[13] and Cluster Tree [39]. In ZigBee Pro, Cluster Tree has been replaced by a stochastic addressing scheme, which picks up a logical short address randomly from the 16-bit address pool for assignment. This approach solves the ”running out of addresses” problem, but loses the ability to route packets merely based on logical short addresses and requires additional mechanism for resolving possible short address conflictions. Up to date, however, ZigBee and ZigBee Pro are still not scalable enough to support very large scale wireless sensor networks, and they lack the support for battery-powered routers.

One of the key operations of 6LoWPAN is header compression. Out of the 127 bytes of the IEEE 802.15.4 data frame, only 81 to 102 bytes are available to upper layers. This poses a big challenge to 6LoWPAN as how to fit the minimum 1280-byte IPv6 maximum transmission unit (MTU) into the small payload of IEEE 802.15.4 data frame. Simple fragmentation without compression is unlikely viable.6LoWPAN uses both stateless compression[33] and stateful compression[41][68] schemes. By exploiting cross-layer redundancy and leveraging shared state within contexts, 6LoWPAN can compress the adaptation, network, and transport layer header fields all into a few bytes. Notwithstanding that 6LoWPAN only adds 7 to 12 bytes of overhead to a raw IEEE 802.15.4 data frame, which is even less than ZigBee’s 7 to 15 bytes of overhead, each node needs to implement the basic IPv6 protocol stack plus the newly introduced adaptation layer so as to be IP-capable. This consumes the precious memory and makes the operations more complicated. Therefore, the decision to make an IP-cable low power wireless sensor network should be carefully made considering the involved cost and benefit.

Recently an IPv6 Routing Protocol for Low power and Lossy Networks (LLNs), referred to as RPL[65], has been submitted to IETF ROLL working group. RPL is a close comparison to IEEE 802.15.5 mesh routing, since both are based on logical tree structure

and yet try to overcome the shortcomings of early logic tree routing protocols such as Cluster Tree[39]. For example, RPL uses Directed Acyclic Graph (DAG), which is equivalent to logical tree used in IEEE 802.15.5. RPL relies on rank and IEEE 802.15.5 relies on tree level to detect and solve loop problems.

The HART Communication Foundation (HCF)[31] rolled out a wireless mesh networking standard, WirelessHART, in 2007. WirelessHART uses TDMA to provide two important features that are needed for industrial sensing and control but currently missing from ZigBee Pro: deterministic latency and deterministic reliability. A strong competitor of HCF is the International Society of Automation (ISA), who has been developing its own industrial wireless mesh networking standard, officially noted as ISA 100.11a (often referred to as ISA SP 100), which also supports the two features. WirelessHart and ISA 100.11a standards do not serve as upper layer protocols of IEEE 802.15.4 because these standards use only the physical layer of IEEE 802.15.4.

The new standard IEEE 802.15.5, can be delineated by two key words, *mesh* and *simplicity*.

In terms of *mesh*, IEEE 802.15.5 is a light-weight scalable mesh routing protocol that well caters to the requirements of resource-constrained wireless sensor networks. By binding logic addresses to the network topology, IEEE 802.15.5 obviates the need for route discovery. This eliminates the initial route discovery latency, saves storage space compared to routing table, and reduces the communication overhead and energy consumption. IEEE 802.15.5 employs a flexible block addressing scheme for logic address assignment as well as network auto-configuration. The scheme takes into account the actual network topology and thus is fully topology-adaptive. A distributed link state scheme is further built atop the block addressing scheme to improve the quality of routes, in terms of hop count or other routing cost metrics used, robustness, and load balancing. All this is done

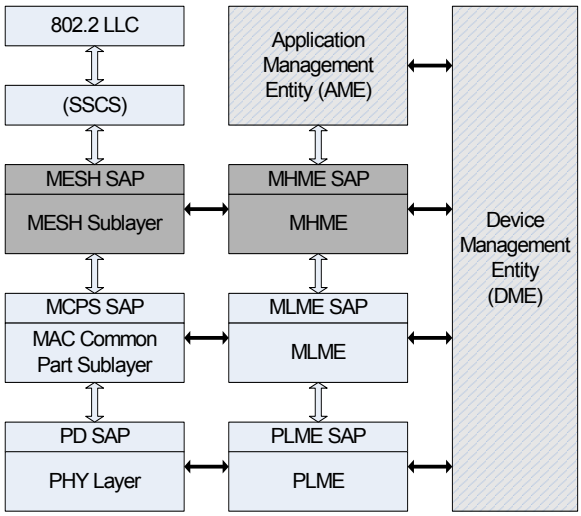


Figure 2.1: Reference Model for IEEE 802.15.5

during network initialization using the native IEEE 802.15.4 association primitives and a couple of IEEE 802.15.5 mesh sublayer commands. The network topology reflected in logic addresses is used as a guideline to tell towards which direction (rather than next hop) a packet should be relayed. The next hop is derived from each relaying node's local link state table. The routing scheme scales well with regard to various performance metrics. The ability to provide multiple paths also precludes the need for explicit route repair, which is the most complicated part in many wireless routing protocols.

For *simplicity*, IEEE 802.15.5 is tightly coupled with IEEE 802.15.4. As shown in Figure 2.1, it adds a thin mesh sublayer between the MAC sublayer and the service specific convergence sublayer (SSCS). It takes full advantage of IEEE 802.15.4 and maintains almost an identical set of service access points as the MAC sublayer. The basic mesh function is accomplished by using 802.15.4 primitives and a couple of mesh sublayer commands. For simplicity, the standard supports only a single PAN. Other enhanced functions like multicast, reliable broadcast, synchronized and asynchronized power saving, route tracing, and portability are done via a few other mesh sublayer primitives and

commands. Compared with other mesh networking approaches, this cross-layer optimization approach not only simplifies the overall protocol stack but also makes it painless for upper layers to migrate from IEEE 802.15.4 one-hop star networks to IEEE 802.15.5 mesh networks. Users can expect most off-the-shelf IEEE 802.15.4 devices will be mesh-ready, at trivial or no additional complexity and cost.

2.3 Basic Mesh Functions of IEEE 802.15.5 Low Rate WPAN Mesh

The design principle of the mesh algorithms for IEEE 802.15.5 focuses on the scalability, which encompasses hierarchy, localization, and minimal overhead. Traditional reactive and proactive algorithms, for example AODV[56] and DSDV[55], fall short of being an efficient and scalable solution for large wireless sensor networks as it relies on network-wide broadcast for route discovery or link state exchange. Also, each node needs to store either routing table or link state information, which grows rapidly along with the network size. Algorithms embedding network hierarchy for a better scalability have been introduced, making use of dominating sets and clusters as seen in OLSR[21]. However, the control overhead and energy consumption for the maintenance of link state table or routing table at each node still falters in resource-constrained wireless sensor networks. In addition, the special role of certain set of nodes (e.g., cluster head) can drain battery unevenly thus shortening the network life time. IEEE 802.15.5 builds in a hierarchy, that is, a tree-based local link state for mesh network. Tree structure provides a simple forwarding while the local link state provides alternative paths and optimized data forwarding. Importantly, the link state information exchanges are done locally, eliminating the overhead

for network-wide broadcast and conserving the memory space of sensor nodes.

IEEE 802.15.5 comprises three mandatory basic functions: tree formation and addressing, local link state, and data forwarding/routing. And several other enhanced functions:multicasting, power saving, reliable broadcasting, and portability support. In this section, we will introduce all mandatory basic functions and the mobility enhanced function.

2.3.1 Tree formation and address block assignment

The tree formation starts with the first node in the network designating itself as the root and beginning to accept association requests from other nodes. After a node is successfully associated, it determines whether to become a parent node which allows other nodes to join the network through it. When a joining node receives multiple association responses, it should choose a proper node to join considering the tree depth or link quality (further details in the next section). After the tree reaches its bottom, that is, no more nodes are waiting to join the network, then, the address reporting process begins from leaf nodes. Leaf nodes will report to their individual parents two numbers: the number of child nodes and additional address space for all future use for example accepting new nodes. Upon receiving these two numbers from all its child nodes, a parent node calculates the same two numbers by summing up all requests from child nodes and its own, and reports to its parent. This process repeats until the root of the tree receives the information from all the branches, after which the root begins to assign addresses. Note that two-byte *short address* is used here. An address management policy is needed to control additional address requests. A simplistic approach may be to allocate all the remnant addresses to all network nodes equally. Figure 2.2 shows an example of the address report

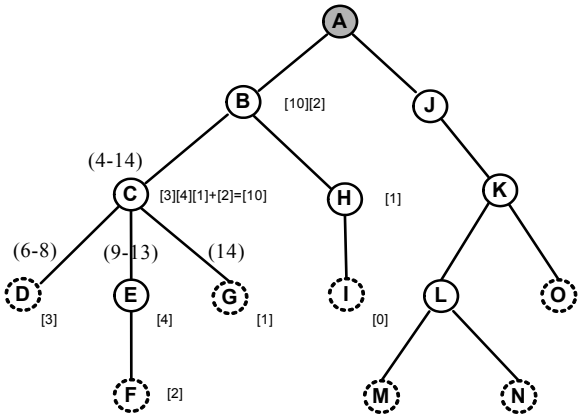


Figure 2.2: Tree Formation and address assignment.

and assignment. [3] means the number of required addresses is 3, and (6-8) means that the address block is 6 to 8. The address assignment takes a top-down procedure. The root will assign a block of consecutive addresses to each branch below it, taking into account the requested number of addresses. This procedure continues until the bottom of the tree is reached. After address assignment, a tree is formed and each node has an address table for tracking its branches. For example, in Figure 2.2, the node C has an address block *(4-14)* because the address blocks for three branches are *(6-8)*, *(9-13)*, *(14)*, respectively. It adds the address block *(4-5)* to use the address 4 for itself and the address 5 reserved for potential expansion of the tree.

2.3.2 Local Link State(LLS) management

After tree formation, every node builds up its own local link state (LLS) information. The additional complexity of combining the block addressing with the LLS scheme is minimal. A node broadcasts several Hello messages to exchange link state information with its immediate neighbors when it receives an address block from its parent. The Hello message contains its own address block (begin and end address), tree level, and the ad-

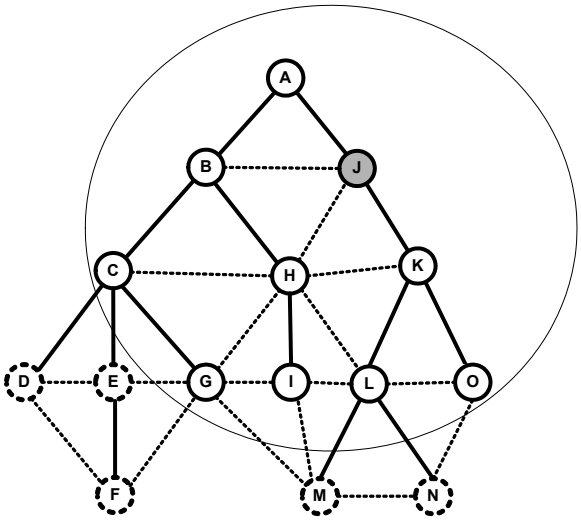


Figure 2.3: The 2-hop neighbors of node J.

dress blocks of one hop neighbors. Upon receiving the Hello message, a node constructs 2-hop neighbor information including peer mesh nodes, optionally with respective link quality. Figure 2.3 shows the 2-hop neighbors of node J. The solid lines constitute a tree and dashed lines indicate mesh links. Both links form a meshed tree. Whenever a node receives the Hello message, it will update its link state information. One can extend this to *k-hop*($k >= 2$) neighbors but at extra costs. A node's LLS information contains two components: neighbor list and connectivity matrix. An example the neighbor list is shown in Table2.1: *begAddri* is the beginning address of the address block owned by neighbor i; *endAddri* is the ending address of the address block owned by neighbor i; *treeLeveli* means the tree level of neighbor i; *hopsi* is the number of hops from "this node" to neighbor i; *linkQualityi* describes the value of link quality such as *RSSI* or *LQI* from the underlying transceiver. Note that the link quality field is only for one hop neighbor for simplicity.

From the 2-hop neighbor information, a node can construct a connectivity matrix. We implement the connectivity matrix using a bitmap to conserve RAM space. For example, Table2.2 illustrates the connectivity matrix for node J. The plus ('+') and minus ('-') at

Table 2.1: Neighbor-List, every row represent one neighbor node's information.

begAddr1	*endAddr1*	*treeLevel1*	*hops1*	*linkQuality1*
begAddr2	*endAddr2*	*treeLevel2*	*hops2*	*linkQuality2*
...	...	...	...	...
begAddrN	*endAddrN*	*treeLevelN*	*hopsN*	*linkQualityN*

Table 2.2: Connectivity Matrix

	J	A	B	C	G	H	I	K	L	O
J		+	+	-	-	+	-	+	-	-
A			+	-	-	-	-	-	-	-
B				+	-	+	-	-	-	-
C					+	+	-	-	-	-
G						+	+	-	-	-
H							+	+	+	+
I								-	+	-
K									+	+
L										+
O										

the cross cell of two nodes indicate they are or are not directly connected. The connectivity matrix shown here is symmetric because only bi-directional links are recorded. The matrix will become non-symmetry if one accounts asymmetric links.

2.3.3 Data Forwarding for Unicast

The data forwarding algorithm uses a divide-and-conquer approach. It first gets a *direction* towards the destination and performs a local search to choose the best next hop along the *direction*. The pseudo code in Figure 2.4 illustrates the procedure of data forwarding. Each node will calculate the next hop from its own address block and local link state information. It checks whether it has "any" information about the destination (category 1) or not (category 2). The category 1 includes the following three cases: the destination is *(i)* one of its descendants, *(ii)* one of its neighbors, *(iii)* one of its neighbors' descendants.

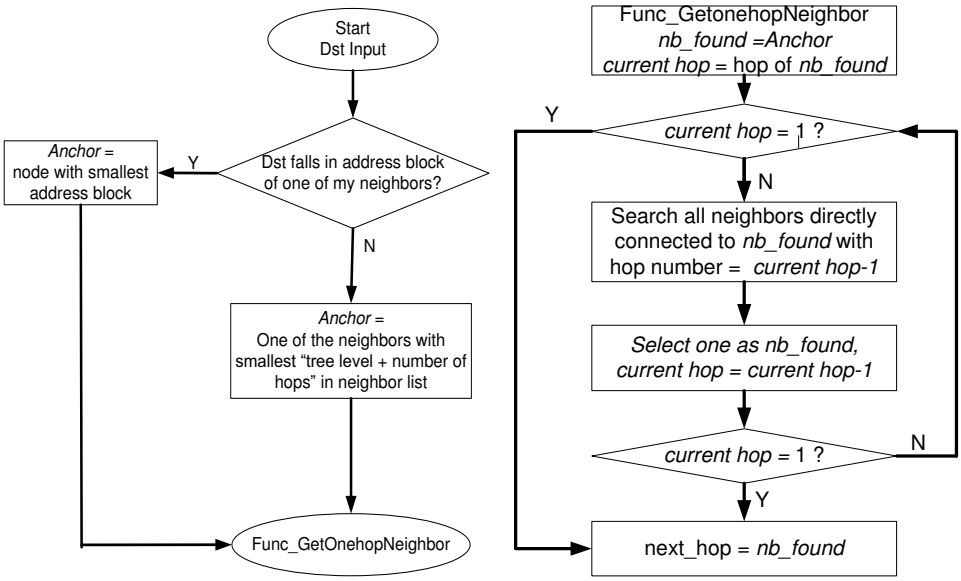


Figure 2.4: Pseudo-code for unicast forwarding

For case (i) it just forwards the packet down to the destination along the tree path. For case (ii), if the destination is a 1-hop neighbor, it just forwards to the destination. If the destination is a 2-hop neighbor, it searches the connectivity matrix for the next hop node connecting the destination. For case *(iii)*, an "anchor" node that can see the final destination as one of its descendants is to be found. Once the anchor node is found, forwarding a packet from the anchor node toward the final destination is the same as the case (i). The anchor node for the case *(iii)* is the node with the smallest address block (calculated by $endaddr - beginaddr$). The rationale is that the node with the smallest address block is the closest node to the final destination. If a relaying node does not have any clue about the destination due to the limited local view (e.g., category 2), the relaying node has to guess a next hop towards the destination. A simple scheme may be to forward the packet to the node's parent node toward the root node. We adopted a heuristics still making use of the neighbor table. The relaying node will examine its 2-hop neighbor table for each node's tree level and the number of hops from this relaying node. The node showing the

minimum sum of the tree level and the number of hops from this node will be the ”anchor” toward the final destination. The rationale is that if a node is closer to the root but far away from this node, it may not be a good anchor. After ”anchor” has been chosen, a ”breadth first” local search (e.g., examining all two hop nodes first) is used to find a next hop node to the chosen anchor node. Note there may be multiple nodes qualifying for the next hop choice, which, in fact, provides resilience for our data forwarding algorithm as shown by line 19 of Figure 2.4.

2.4 Enhanced Mesh functions

Basic mesh functions presented in Section 2.3 are essential to establish and operate a mesh network. For a variety of envisioned applications for wireless sensor networks, the basic mesh functions alone are not sufficient. In this section we present briefly the two major enhanced functions that IEEE 802.15.5 is equipped with: mobility. Readers are encouraged to refer to the standard document for more details.

2.4.1 Mobility Support

Although most wireless sensor networks are stationary, certain network nodes can be mobile. For instance, a data aggregator held by an operator moves to gather information from network nodes or distributes messages to the network. Unlike the MANET's network-wide mobility, this kind of limited mobility is more realistic for wireless sensor networks. 802.15.5 supports the mobility restricted to leaf nodes. Most of device in 802.15.5 network have the energy limitation. And also in wireless mesh networks, there won't be centralized database like in cellular networks. so beyond the traditional mobility management requirement. The mobility management needs to be energy efficient and distributed

manageable. Here we briefly introduce the leaf node mobility scheme. The procedures for the support of a mobile device include the following components: the detection of the movement, the mobile device rejoining the network, the new address assignment, and the notification of the movement to relevant nodes.

- Detection of Movement

The movement of a mobile device can be detected by either the mobile device or the parent of a mobile device. When a mobile device moves away from its parent, it recognizes the movement when it cannot transmit any packets to its parent node. Also, if the periodic hello message, which is an optional feature of this specification, is used in a network, the mobile device can detect the movement when it misses the hello messages from its parent. The mobile device's parent can also detect the movement if it misses the ACK message from the mobile device after sending it a packet.

- Rejoin a mobile device to the network

After a mobile device has been moved to a new location, it shall rejoin the network by issuing a MHMSE-JOIN.request to its mesh layer with the parameter RejoinNetwork set to 0x02, which means the device is joining the network using mesh rejoin procedure. The MHMSE-JOIN.request will generate an active scan at the MAC layer. Any potential parents receiving this active scan shall respond with beacon frames. In each beacon frame, a potential parent shall insert its capability to indicate whether it can accept the mobile device as a new child. The capability of a node is decided by an upper layer that is beyond the scope of this document.

Upon the reception of beacon frames from potential parents, the mobile device will select a "suitable" parent node to join. The join policy for a "suitable" parent is decided by

an upper layer but is beyond the scope of the current specification. After a certain node is selected as the parent to join, the mobile device will trigger the MAC layer MLME-ASSOCIATE.request to perform the association procedure. The parent node that receives this association request will first check if this mobile device was its child by checking its Mesh Address Mapping Table. If the mobile device was its child, then the parent will assign the same address this mobile device has been using before. Otherwise the parent shall assign a new short address selected from its available addressing block.

After the parent node issues the MLME-ASSOCIATE.response to the mobile device, it generates MHSME-JOIN.Indication to the upper layer to inform the join (rejoin) of the mobile device. Upon receiving MLME-ASSOCIATE.response, the mobile node will generate MHSME-JOIN.Confirm to its upper layer to inform the result of the join (rejoin).

- Inform the mobile node to the relevant nodes

After successfully joining the network, the mobile device shall report its new address to relevant nodes. The relevant nodes are defined as the mobile node's old parent and nodes that have the mobile node in their neighbor list. First, it shall unicast the Rejoin-Notify command to its former parent. The command frame sub-type of Rejoin Notify shall have a value of 11110. The DA address field is set to former parent's short address. Upon receiving the Rejoin-Notify command, the former parent of the mobile device shall broadcast the Rejoin-Notify to (TTLOfHello+1) hop-neighbors. Note that each node has (TTLOfHello+1)-hop neighbor table. Therefore, the old parent needs to broadcast to (TTLOfHello+1) hop neighbors. Upon receiving the broadcast Rejoin-Notify, every node that has the mobile device in its neighbor table will set the status of the mobile device to "removed" and update its connectivity matrix.

- Inform the mobile node to the relevant nodes

After successfully joining the network, the mobile device shall report its new address to relevant nodes. The relevant nodes are defined as the mobile node's old parent and nodes that have the mobile node in their neighbor list. First, it shall unicast the Rejoin-Notify command to its former parent. The command frame sub-type of Rejoin Notify shall have a value of 11110. The DA address field is set to former parent's short address. Upon receiving the Rejoin-Notify command, the former parent of the mobile device shall broadcast the Rejoin-Notify to (TTLOfHello+1) hop-neighbors. Note that each node has (TTLOfHello+1)-hop neighbor table. Therefore, the old parent needs to broadcast to (TTLOfHello+1) hop neighbors. Upon receiving the broadcast Rejoin-Notify, every node that has the mobile device in its neighbor table will set the status of the mobile device to "removed" and update its connectivity matrix.

As an example, suppose there is a sender node sending a packet to the mobile device using its previous short address. Once the packet arrives at any node that has the address of the mobile device in its neighbor list with status "removed", the node shall send the Rejoin-Notify with its address field set to the mobile device's new address to the sender and re-route the packet to the mobile device's new address.

- Update the link state information

After the mobile device rejoined the network and informed all relevant nodes via its old parent of its movement, it shall broadcast the Hello message to TTLOfHello-hop neighbors in order to inform its new neighbors its address and connectivity. The following Fig. 2.5 shows how the leaf node mobility is supported.

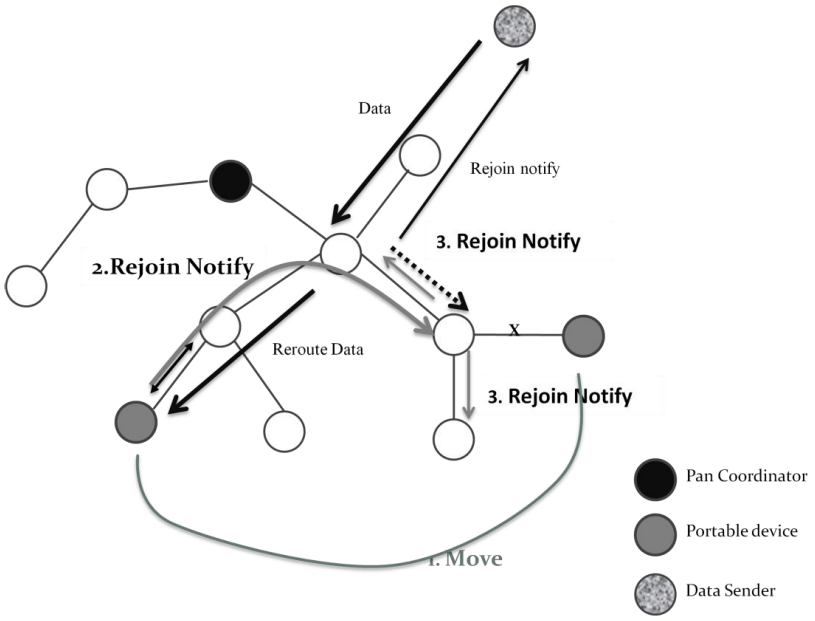


Figure 2.5: mobility support scheme in 802.15.5.

2.5 Performance Analysis using Developed Test-bed

2.5.1 Test-bed Architecture

A simulation study is reported in [72] to assess the performance of basic mesh functions. To further demonstrate the feasibility of the standard protocols in real applications, we implemented major functions (unicast and multicast) in a real testbed, consisting of 50 Micaz nodes each with an MIB600 gateway [51] deployed in the 5th floor of Engineering building. An initial environment of our testbed had been deployed with benchmarking of MotoLab at University of Harvard. Registered users can update their own executable image files and gather specific results from database via a web-based interface. Furthermore, the web-based interface is able to visually display results like network status, tree topologies, or data forwarding. CWSNet consists of mainly three levels: user remote network, control and management, and hardware platform network. Figure 2.6 the user network

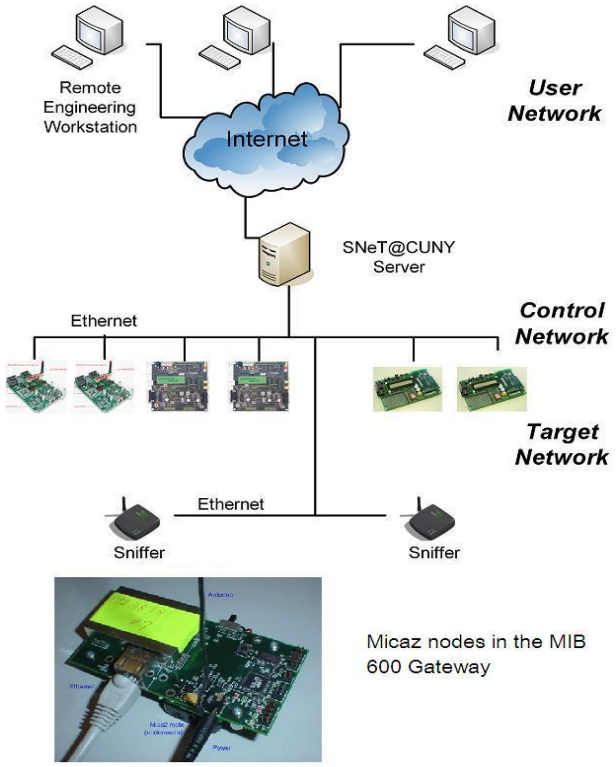


Figure 2.6: User network scenario and sensor board.

scenario and sensor broad used in our test-bed.

Micaz nodes is used as the hardware platform. The major components of Micaz node includes one CC2430 transceiver which has complete IEEE 802.15.4 PHY and MAC stack, one ATmega128 micro processor as MCU, pin connector and antenna. Figure 2.7 shows the architecture structure. A common mesh layer software platform has been developed based on IEEE 802.15.4 PHY and MAC stack provided by TI CC2420[22] using ATmega128 micro processor. The ATmega128 has the features as following:

- 8Bit AVR Microcontroller + 128KB Flash memory.
- AVR core.
- RISC/Harvard MCU architecture.

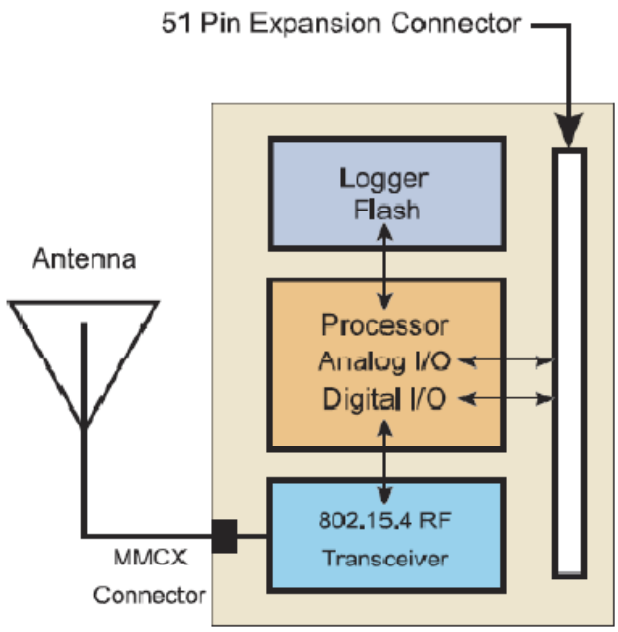


Figure 2.7: Micaz node architecture.

- 133 instructions, single level pipeline.
- Up to 16 MIPS at 16MHz (8MHz for ATmega128L).
- 32*8 general purpose Regs, peripheral control Regs.
- 128KB Flash; 10000 times rewritable.
- 4KB EEPROM; 10000 times rewritable.
- 4KB internal RAM.
- 64KB external memory space.
- Two 8bit timer/counters.
- Two 16bit timer/counters.

- 6 PWM channels.
- 8 channel, 10 bit ADC
- Two USARTs.
- SPI interface.
- Watchdog timer.
- Six sleep modes: idle, ADC noise reduction, power-save.
- power-down, standby and extended standby.

The implementation of the algorithms is written in C and compiled with AVR-GCC. Control Network running TCP/IP protocols is built up to upload binary executables to exchange control messages and testing results between the sensor nodes and the server. The testbed starts as we upload our program binary code beginning from the root. Each node will wait for a time $T = N - i * time_interval$ and then reports the number of children to its parent. The process repeats until the root node collected the number of child nodes. N is a predetermined value chosen as 12 considering the tree depth of 50 nodes and i represents a tree level for each node.It takes about 30s to let all nods join the tree and exchange mesh information.Finally,the experiment results can be displayed by the Java and PHP program running in server with Mysql database. Then the web based language(PHP) could get these network information and display at the remote terminal. Figure 2.8 gives an example of the system GUI to show the networking structure of IEEE 802.15.5. Solid lines represent tree structure and dotted lines represent the routing path in one instance of experiments.

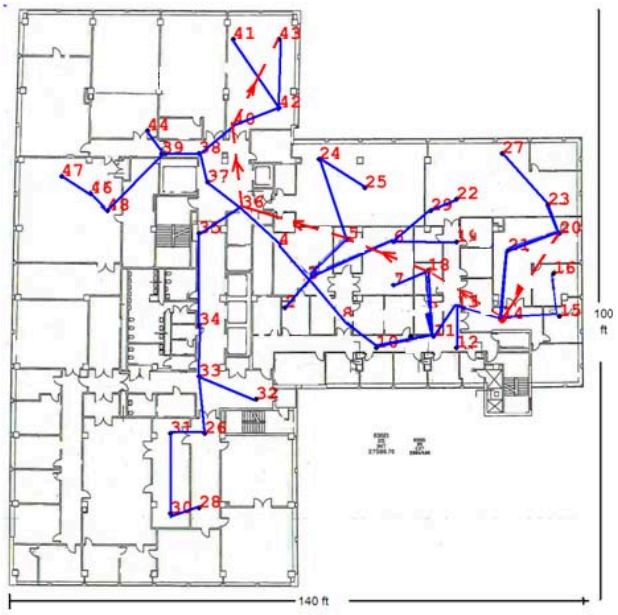


Figure 2.8: Testbed: 50 nodes in the 5th floor of engineering building.

2.5.2 Performance Evaluation of Mandatory Functions

Routing Performance

The most fundament part of the mesh lies in its ability to provide the end-to-end data delivery. Considering this, we have tested the unicast routing by using various scenarios and multiple performance measures. The first group of experiments is performed to show the end-to-end packet delivery ratio, defined as the number of successfully delivered packets divided by the number of packets generated at sources. In order to suppress the bias from a particular topology, we setup 5 different tree topologies staring with different root nodes located in different rooms. For each tree structure, source and destination nodes pairs are selected randomly and final results are averaged. Every node keeps 2-hop local link state information. The data packet size including the IEEE 802.15.4 MAC and PHY is 59 bytes and the maximum MAC layer retransmission is three times. Figure 2.9 shows the packet

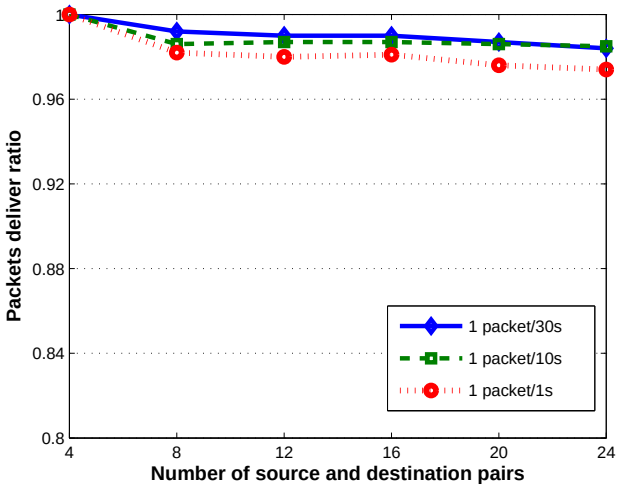


Figure 2.9: The packet deliver ratio vs number of sources/destination pairs.

delivery ratio versus the number of source/destination pairs with different offered loads. The packet delivery ratio maintains over 95% decreasing slightly as the number of nodes and the offered load increases. Analyzing data trace file obtained by the packet sniffer, we find that most packet losses are due to the MAC layer contention and unstable wireless lossy link between sensor devices especially the nodes around the elevator, for example, wireless links between node 4 and node 36.

In Figure 2.10, we examined the packet delivery ratio against different number of hops. Interestingly, increased number of hops between sources and destinations shows almost no negative effect on packet deliver ratio for the scenario tested. The higher the traffic loads, however, the lower the packet delivery ratio gets mainly due to the MAC layer contention. End-to-end delay performance is presented in Figure 2.11, showing increased delay along with the increased number of hops. The data transmission and the MAC layer retransmissions seem to be the main causes.

The second group of experiments considers a typical WSN scenario: multiple source nodes report data periodically to a sink node. Compared with the first experiment, the

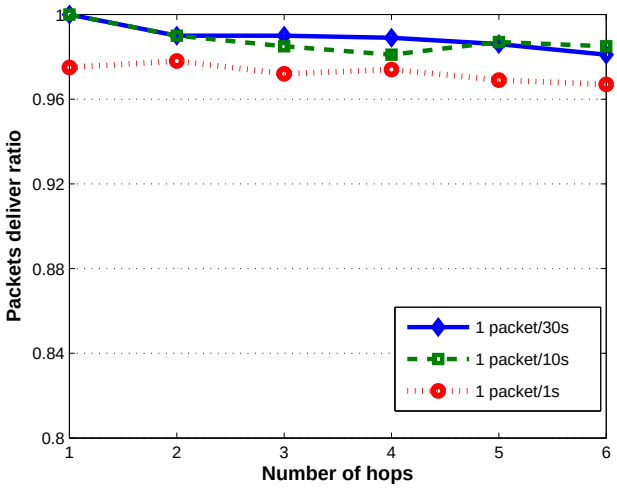


Figure 2.10: Packet deliver ratio vs number of hops.

number of sources and offered traffic impact critically as shown in Figure 2.12. We observe a visible drop(around 10% compared with peer to peer case) in the packet delivery ratio when the number of sources exceeds 20 and data rate is 1 packet/second. This drop is due to IEEE 802.15.4 MAC contention, not the routing algorithm itself.Since the 20 chosen sources are 1-7 hops away from the sink, multiple MAC retransmissions are inevitable and amplified at links for example between node 4 and 36. The funneling effect around the sink node is observed in Figure 2.13. Finally Figure 2.14 shows the end to end delay.

The packet delivery ratio degrades as the traffic increases, but shows a negligible influence by the hop distance. The reason is that regardless of the hop distance from the sink, all traffic generated by the sources will funnel to the sink. The end-to-end delay is almost the same as the first group of experiments, except for the case of 1 packet per second traffic in which the end-to-end packet delivery suffers more contention and thus more MAC layer retransmission.

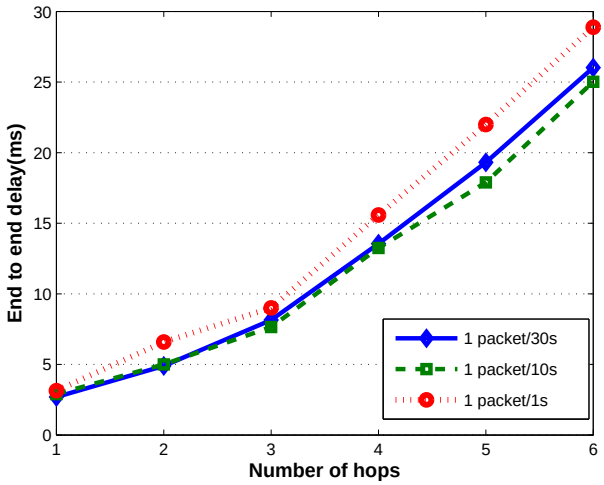


Figure 2.11: Average latency vs number of hops.

Local Link State Information

As explained in section 2.3, the data forwarding is optimized by the local link state information. We investigate the impact of the size (k-hop) of local link state information on the data forwarding decision. k=0 means that the data forwarding is purely done along the tree by parent and child relationship. k =1,2 means one-hop neighbor or two hop information. Every neighbor node information occupies 9 bytes in a neighbor table. The connectivity matrix varies according to the maximum number of nodes in the neighbor list and in our experiment it has two sizes: 16 bytes for neighbors up to 16 nodes or 32 bytes for neighbors up to 32 nodes. For k=1 we use 16 Bytes and 32 bytes for k=2.Table2.3 shows the end-to-end delay, average number of neighbors, and average size of local link state information size regarding the nodes along the data path. With the help of local link state information, the data forwarding strategy has notable improvement. The path optimization is done in our case using the shortest path via higher LQI link. The overhead to accommodate more information is tens or hundreds of bytes, which is reasonable com-

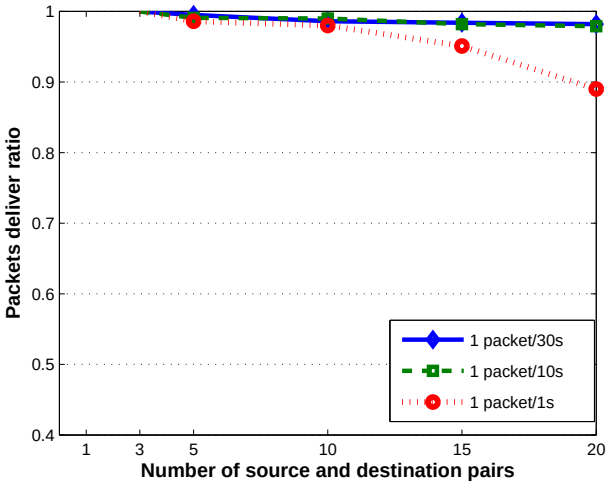


Figure 2.12: Packet deliver ratio in sources to sink scenario with respect to different number of sources nodes.

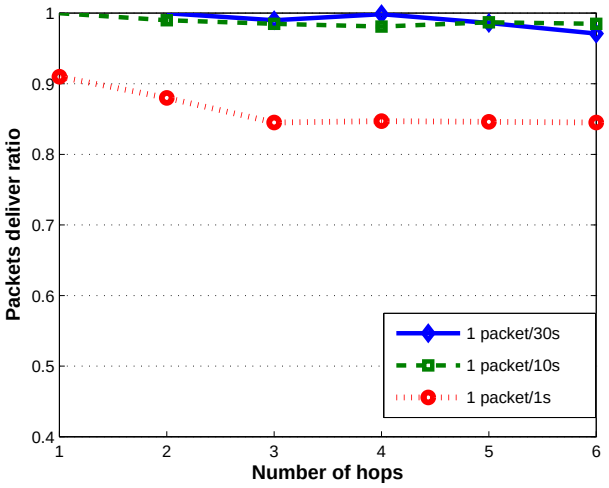


Figure 2.13: Packet deliver ratio vs the number of hops.

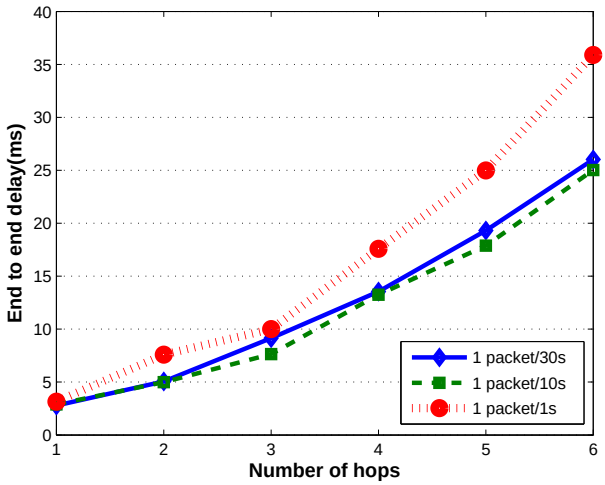


Figure 2.14: End-to-end latency vs different number of hops.

Table 2.3: Performance evaluation with varying local link state sizes.

hop	End-to-end delay	Average number of neighbors	Average LLS Information size
K=0	33.74ms (7.8hops)	3.1	27.9 bytes
K=1	2.85ms (6.0 hops)	5.8	68.2bytes
K=2	7.98ms (2.75 hops)	12.5	144.5 bytes

pared with 4K RAM size in Micaz node. Note here that we do not compare our algorithm with AODV because it is unfair to compare them when they belong to different categories (reactive versus proactive).

2.6 Summary

A new IEEE standard for wireless sensor networks, IEEE 802.15.5 WPAN Mesh (low-rate part), is introduced in this Chapter. The basic mesh functions comprise tree formation, allocation of address blocks, distributed local link state information, and mesh routing. The network formed from these basic building blocks exhibits many desirable properties such as scalability, simplicity, and reliability. In particular, the k-hop local link state informa-

tion and tree structure with address block scheme facilitate the mesh routing by providing a guideline to extract the next hop toward a destination even when the local link state cannot give any information about the destination. One example of enhanced function: mobility support is also presented. The testbed experimental results are very encouraging. The protocols demonstrate satisfactory performance against all the measures applied. For example, the packet delivery ratio maintains over 97% even when the offered traffic reaches 1 packet/sec in the case of multiple source-destination pairs and around 90% in multiple sources-to a sink style traffic.

In the following chapter, we will present more comprehensive study of IEEE 802.15.5 and its expansion to general wireless sensor networks and wireless mesh networks.

Chapter 3

Performance Analysis of IEEE 802.15.5

3.1 Overview

The former chapters and our earlier paper [47][72] have presented the comprehensive performance studies via simulations and the test-bed implementation of IEEE 802.15.5 low rate part, demonstrating it as a viable solution for real environments. This chapter will present theoretical analyses on the fundamental capability of IEEE 802.15.5. Four important performance metrics will be investigated: scalability in view of memory space and energy consumption for routing establishment, fault tolerance represented by the node degree of next hop choices and Mean time to failure of routing path, routing path length compared with the shortest route, and the overhead associated with a mobility support. We will compare the performance of IEEE 802.15.5 with those of Zigbee PRO and the tree based scheme. The analyses and simulation results demonstrate that IEEE 802.15.5 consistently maintains superior performances in all measures investigated and is also capable of meeting different network requirements by adjusting the radius of the local link state (K hops).

3.2 Performance analysis of 802.15.5

Though very useful, our earlier works using 50-node test-bed implementations [47] are inevitably limited in the numbers of nodes and scenarios to reveal the fundamental capability of 802.15.5. Therefore, we perform theoretical analyses for the mandatory functions of 802.15.5. These analyses are approached by the following four performance measures: (a) scalability with respect to memory occupancy and energy consumption for routing establishment, (b) fault tolerance represented by the node degree of next hop choices, (c) routing path length compared with the shortest route, and (d) the overhead associated with the mobility support. We benchmark the performance of 802.15.5 against Zigbee PRO that uses AODVjr as its routing scheme. As 802.15.5 uses a tree structure (a.k.a. meshed tree), we also compare it with the tree based routing. We first list the notations and assumptions used in the analyses.

Notations and Assumptions

We assume a network can be represented by the following parameters:

- N: The number of nodes in the network.
- R: The radius of a network in terms hop distance. R is logical term that is determined by the ratio of physical range of network to transmission/receiving range of wireless node. for a certain node, all nodes within its transmission/receiving range is considered as ”one hop” nodes. In real situation, transmission/receiving range varies due to the lossy and complicated wireless environment situation. In the later simulation, we will apply probabilistic model to distance based model.
- M: The average number of one hop neighbors. It also represents the node density of a network as $M = N\frac{\pi 1^2}{\pi R^2} = \frac{N}{R^2}$

- K: The radius of local link state information of 802.15.5 represented by the number of hops. This notation applies only to 802.15.5. The same assumption for R is applied here.

- L: The average number of parents and children for a node, $L \leq M$. This notation applies the tree based routing and 802.15.5.

3.2.1 Scalability

Wireless sensor networks tend to have a large number of sensor nodes with limited resources regarding memory space, processing power, and energy. The network routing scheme should be designed in the consideration of the scalability in terms of the number of nodes and the node density. Here we consider the network-wide routing capability to provide the routing from any node to any other node, not only from many nodes to a certain gateway aggregator (sink) or vice versa. We examine the scalability issue by two aspects. First we will analyze the memory occupancy of the routing information at a node. Second we will analyze the number of messages used for a network-wide routing establishment, which directly affects the energy consumption of a network. Approximations and the big O notation are used as we are more interested in the growth trend of complexity when the number of nodes in a network increases.

Memory Occupancy for Routing Information

We approximate the memory space occupancy by the number of entries in respective routing table.

- Zigbee PRO

Current Zigbee PRO eliminates the hierarchical tree based addressing and routing schemes. Instead, a random addressing is used and the routing depends on AODVjr scheme [13]. In AODVjr, every node needs an entry in the routing table for each potential destination node. Therefore, the number of entries in the routing table increases linearly with the number of nodes in a network. Zigbee PRO also introduces the source based routing for packets originated from the sink node (i.e., one-to-many). However, this reduces only the entries for packets from the sink. In general, each node needs approximately N entries in the routing table, or $O(N)$ complexity.

- Tree based scheme

In the tree based scheme, the routing information includes only a node's own parents and children whose number is L on average. So the memory occupancy is $L \leq M = \frac{N}{R^2}$ entries, or $O(N)$ complexity if the geographical space of a network is fixed (fixed network size with constant R). If a network expands with the same node density (fixed node density with constant M) but with a larger geographical space coverage (increasing R), the memory occupancy keeps constant, thus only $O(1)$ complexity.

- 802.15.5

802.15.5 uses a tree structure with local link state information. The routing information includes two parts: neighbor table and connectivity matrix, both of which are determined by the number of K-hop neighbor nodes. The average number of K-hop neighbor nodes will be $N \times \frac{\pi K^2}{\pi R^2} = \frac{K^2}{R^2} \times N$, and the number of entries for the neighbor table of a node is $\frac{K^2}{R^2} \times N$. The connectivity matrix, in contrast, consumes far less memory space by using a bitmap in our implementation in [47]. For example, the neighbor table for 16 nodes is 160 bytes while the connectivity matrix is only 32 bytes. Therefore, for the growth trend, we approximate the memory occupancy only to account the neighbor table. Similarly to the

case of the tree based routing, the memory occupancy keeps constant if the node density is fixed. Therefore, the memory occupancy is approximately $\frac{K^2}{R^2} \times N$ or MK^2 entries. Thus $O(N)$ complexity for the case of fixed network size and $O(1)$ complexity for the case of fixed node density.

Message Overhead for Network-Wide Routing Establishment

For the analysis part, the routing message exchange is considered only for network layer, MAC/PHY contention and retransmission will be include in later simulation part based on IEEE 802.15.4 simulator. We also simplify the problem by considering only the messages used for route establishment. We assume all devices have the routing capability.

- Zigbee PRO

We first discuss the route establishment between peer nodes in a network except the routes between nodes and the sink. AODVjr[13] used by Zigbee PRO performs the route discovery by broadcasting an RREQ (routing request) network-wide and only the destination node will respond to the first received RREQ to suppress the broadcast storm. The route establishment between a pair of source and destination involves all nodes to forward RREQ messages (upon receiving an RREQ, a node either broadcasts it if it does not have any information about the destination or uni-casts it towards the destination if it has the destination information). Upon receiving the first RREQ, the destination node sends back an RREP (routing response) message, traversing χ hops back toward the source node, where χ depends on the distance between the source and destination nodes whose value is between maximum $2R$ and minimum 1. The average route establishment message overhead is thus $N+R$. For a single source to get the routing paths to $N-2$ nodes in a network (except the sink and itself), it needs to perform the route establishment $(N+R) \times (N-2)$

times. For $N-1$ nodes(except the sink) to get the routing paths to other $N-2$ nodes in a network, it costs $(N-1)(N-2)(N+R)$ message overhead.

In order to prevent every device in the network from having to discover the sink, separately, ZigBee PRO introduces a scheme, a.k.a "many-to-one scheme" that provides a special case of route discovery. In the many-to-one scheme, a single route request broadcast from the sink establishes a routing entry in every node in the network for the sink as a destination. So it only needs N messages to establish routes from nodes to the sink. Combining together, it is $(N-1)(N-2)(N+R)+N$, the approximate message overhead is $O(N^3)$ complexity.

- Tree based scheme

We assume that the average number of parents and children of a node is L. For every node, it needs to exchange information with only its parents and children; so the message exchange overhead is $L \times N$. As $L \leq M = \frac{N}{R^2}$, the total message overhead is approximately $(\frac{1}{R})^2 \times N^2$ or MN. If the network size is fixed, it results in $O(N^2)$ complexity. If the network density is fixed, $O(N)$ complexity.

- 802.15.5

In 802.15.5, a node needs to exchange messages with its parents and children 4 times to establish the tree structure (association request and response, address report and block address assignment). So it requires $4L$ message exchanges for each node for the tree structure establishment. To build up the local link state information, every node needs to broadcast a packet with its own and one hop neighbors'addresses. At the same time, every node needs to relay(rebroadcast) hello messages generated within $K-1$ hops. So for a node, the number of message exchanges is $\frac{\pi(K-1)^2}{\pi R^2} \times N$. As $L \leq M = \frac{N}{R^2}$, the message overhead for each node is $4L + \frac{\pi(K-1)^2}{\pi R^2} N \leq \frac{4+(K-1)^2}{R^2} N \simeq (\frac{K}{R})^2 \times N$, where we assume $K > 1$.

For N nodes, the message overhead is approximately $(\frac{K}{R})^2 \times N^2$for fixed network size case, thus $O(N^2)$complexity, or MK^2N for fixed node density case, thus $O(N)$ complexity.

3.2.2 Fault Tolerance

Fault tolerance requires the routing scheme to work even when certain nodes in a network fail. It is extremely important as the node failure can be a prevalent phenomenon in wireless sensor networks. In this paper, we evaluate the fault tolerant capability represented by two criterions: The node degree of next hop choices(NDN), and mean time to failure of routing path($MTTF_R$). NDN measures the number of potential next-hop nodes toward a certain destination without re-triggering the route or tree establishment. While $MTTF_R$ illustrates the life time of routing path(H hops) based on the assumption that mean time to failure of single node follows exponential distribution with mean value T_F and failures are independent, So the $MTTF_R$ = *MTTF of single hop/number of hops along routing path*.

- Zigbee PRO

In AODVjr, only the destination node responds to RREQ and only the first RREQ it receives. Therefore, only a single path is established between a source and a destination node, so $NDN = 1$. Once that node fails, that routing path will fail and needs to be re-established, so the MTTF of single hop equals the MTTF of single node, and MTTF of routing path equals to $MTTF_R = T_F/H$.

- Tree based scheme

The tree based routing is a deterministic routing in which the next hop node is either a node's parent or one of its children according to the address of a destination node, so

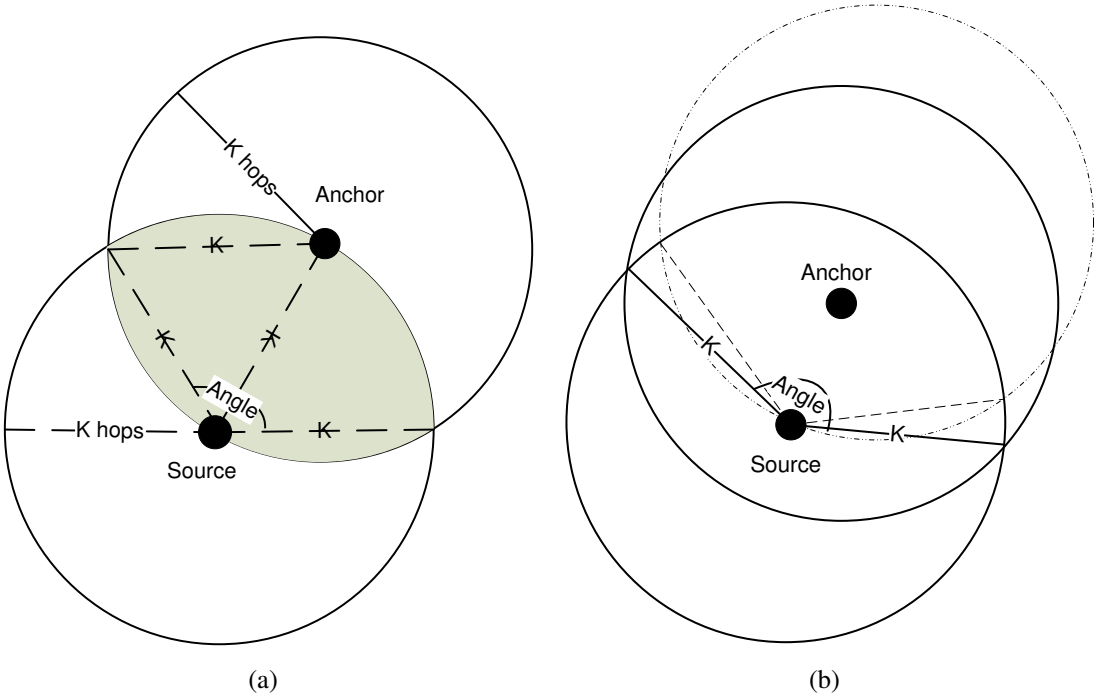


Figure 3.1: (a) is border anchor case: painted area is the information shared by the source and the anchor node. (b)is inside anchor case: the ”angle of view” for the source node to the anchor is larger than the border anchor case.

$NDN = 1$. Same as Zigbee PRO case, the routing path life time is determined by single node failure, so $MTTF_R = T_F/H$.

- 802.15.5

As explained in Chapter 1, the routing scheme for 802.15.5 can be summarized as follows. The node first finds the anchor node within its k-hop neighbors that leads towards the destination. Next, it runs a local link state search to find a next hop node toward the chosen anchor. The anchor node can be K hops away if the destination is beyond K hops from the relay node, or within K hops if the destination node is less than K hops and has been chosen as anchor. We denote the first case anchor as ”border anchor” and second case anchor as ”inside anchor”. We will discuss these two situations separately.

Fig. 3.1(a) depicts the "border anchor" situation where the anchor node is K hops away. The shaded area is the local link state information shared by the source and anchor nodes. For the source node, the next hop can be chosen from its one hop neighbors that is located inside the shaded area with the "angle" of $\frac{2\pi}{3}$. With the average number of one hop node M, the average number of one hop nodes in shaded area is $\frac{M}{3} = \frac{N}{3R^2}$. In this case, the potential next hop choices is irrelevant of the size of local link state information: increasing K cannot provide a larger NDN.

On the other hand, if the anchor node is less than K hops away, the overlapping area created from the two circles of K hop radius centered at the source and anchor nodes is larger than the border anchor case, as Fig. 3.1(b) shows(for comparison, the border anchor case is plotted in dashed line). We can see the angle is larger than $\frac{2\pi}{3}$, so there are at least $\frac{M}{3}$ one hop neighbors included in the shared area. Combined together, $NDN \geq \frac{M}{3} = \frac{N}{3R^2}$.

Since the 802.15.5 has multiple next hop choices, if a single node fails, it can still manage to find alternative next hop immediately. The routing failure happens only when all NDN potential next hop nodes fails, due to the memoryless feature of exponential distribution, the MTTF of first node failure is T_F/NDN, from the failure point on, the MTTF of second node failure is $T_F/(NDN-1)$, until the last one fails. Adding them together, the MTTF of one hop is $\sum_{i=1}^{NDN} \frac{T_F}{i}$. Combing with NDN calculation, $MTTF_R \geq \sum_{i=1}^{\frac{N}{3R^2}} \frac{T_F}{i \times H}$.

This analysis provides a meaningful guideline for potential improvement for fault tolerance. The current 802.15.5 tends to choose "border" anchor. However, if a network requires a high node degree to better cater to the fault tolerance, the anchor should be chosen from the "inside anchor" nodes. By doing this, the anchor gets "closer" to the source so that the overlapping local link state information between the source and the anchor be larger, hence the larger node degree of next hop choices.

3.2.3 Routing Path Length

In order to evaluate whether a routing scheme can achieve the shortest path, we propose a new measure called the Route Length Index *RLI*=(*routing hops/shortest path hops*). A large *RLI* means a poor performance because a route found is much longer than the shortest path, in contrast, $RLI = 1$ means the route found is the shortest path. Based on *RLI*, we define the Worst Route Length Index $WRLI = \max(RLI)$ that models the upper bound of *RLI*. In the following, we will focus on *WRLI* for the three algorithms.

- Zigbee PRO

We simply consider AODVjr can always find the shortest path. Then, $WRLI = 1$.

- Tree based scheme

In the tree based scheme, routing between peer nodes are purely along with tree branches. If both nodes are located on the same branch, the tree based routing gets the shortest path. To consider the *WRLI*, we will focus on the scenario in which the nodes are located at different tree branches so that the routing needs to go through the tree root. Assuming the source and the destination is R_1 and R_2 hops away from the tree root respectively and the shortest distance between them is d, $WRLI = Max(\frac{R_1+R_2}{d})$.

We first consider the case in which the tree root is located at the center of a network. It is easy to demonstrate the *WRLI* occurs when $R_1 = R_2 = R$ and $d = 1$, which means both the source and the destination are located at the boundary of network but belonging to two different tree branches. Thus, the routing path needs to go through the tree root, resulting $WRLI = 2R$ even if they are only one hop away. If the tree root is located at the boundary of a network, the upper bound of route length is $4R$ so that $WRLI = 4R$.

- 802.15.5

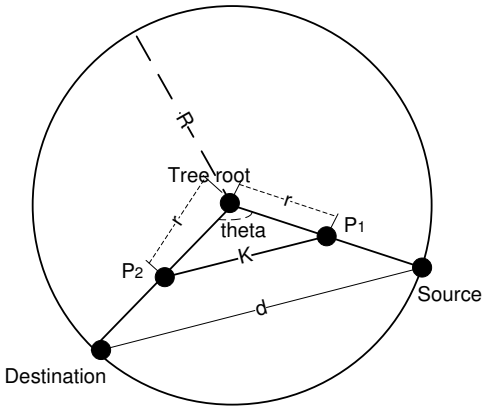


Figure 3.2: An example of routing process when the tree root is at the center of a network.

Similarly to the tree based routing, if the source and destination nodes are located at the same tree branch, then shortest path can be found. Also if the address of the destination node falls into the block address of one of source's neighbors, routing is done by the local search which can find the shortest path. We consider here the scenario in which nodes are located at different tree branches and beyond the local link state region. We discuss two scenarios where tree root is located at the center and boundary a network respectively.

(1) Tree root at the center of a network

Based on the routing scheme, if a source node does not have any clue about the destination, it will choose a route towards the root. Every node along the route will keep on searching its K-hop local link state region for any information leading to the destination. No information about the destination found, the packet will be routed along the direction to the tree root and the next hop node will repeat this search. Fig. 3.2 shows the routing process, in which we denote P_1 for the first node that finds the destination information in its local link state and P_2 for the anchor node. Routing path will be from P_1 to P_2 without going through the tree root. The analysis below shows that we can approximate that P_1 and P_2 have the same tree level (then same distance r to the root) and the route length

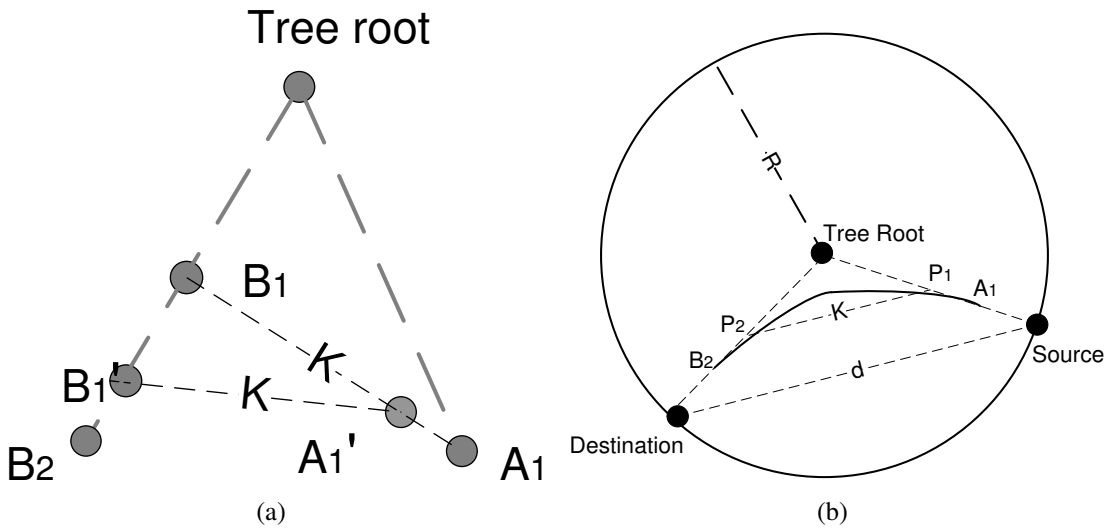


Figure 3.3: (a) shows the detailed routing process once a node finds the destination node's information in its K hop local region. (b) shows how we approximate the routing in Fig. 3.2, where solid line is the real routing path and dashed line is the approximated routing path in calculation.

between them is K.

Fig. 3.3 shows the detailed routing process once a node (denoted as A_1) finds destination's information in its K hops local region, and B_1 is the anchor node it chooses(on the same tree branch with the destination). After selecting the anchor, A_1 will choose a next hop(denoted as $A_1^{'}$) that is close to anchor B_1. $A_1^{'}$ will repeat the algorithm and choose an anchor(denoted as $B_1^{'}$). As the algorithm tends to choose anchor node with smallest address block, $B_1^{'}$ has a deeper tree level than B_1. This process will repeat until a data packet is routed to the node (denoted as B_2) on the same tree branch with the destination so that the actual routing path will resemble an arch-like path. In order to simplify the problem and due to symmetric nature, we approximate A_1 and B_2 having the same tree level and the distance between them is K hops. Fig. 3.3(b) shows the comparison between the actual routing path and the approximated routing path.

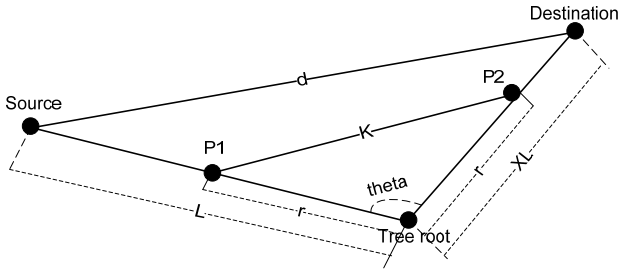


Figure 3.4: General routing in 802.15.5.

We first prove that *WRLI* is obtained when the source and destination nodes are located at the boundary of a network. *WRLI* is obtained when both source and destination nodes are located at the boundary. We assume source and destination node is $L(0 < L \leq R)$ and $XL(0 < X \leq 1)$ hops away from the tree root. We will prove that *WRLI*(*RLI* is maximized) is obtained when L is maximized and $X = 1$, which means source and destination nodes are both located at boundary of a network.

we have $RLI = \frac{L+XL-2r+K}{d}$. According to the law of cosine, $\cos(\theta) = \frac{2r^2-K^2}{2r^2} = \frac{L^2+(XL)^2-d^2}{2XL^2}$, (see Fig. 3.4), then we get $d = \sqrt{(1+X^2)L^2 - \frac{(2r^2-K^2)XL^2}{r^2}}$. As we have $RLI = \frac{(1+X)L-2r+K}{d=\sqrt{(1+X^2)L^2-\frac{(2r^2-K^2)XL^2}{r^2}}}$, taking the derivation of *RLI* over L, $\frac{\partial RLI}{\partial L} = \frac{2r-K}{\sqrt{\frac{L^2(X-1)^2+XK^2}{r^2}}L} > 0$.

RLI increases monotonically via L, so *WRLI* gets when L is maximized($L = R$), which means that the source node is located at the boundary of network. Next, we try to see the relationship between *RLI* and X. *RLI* gets the maximum when $\frac{\partial RLI}{\partial X} = 0$, $X = 1 - \frac{K^2}{2r^2-2rL-LK} > 1$, and $\frac{\partial RLI}{\partial X} > 0$ when $0 < X \leq 1$. *RLI* increases monotonically via X in region $0 < X \leq 1$. *WRLI* can be obtained when $X = 1$, which indicates that the destination node should have the same distance to tree root as the source, thus the destination node also locates at the boundary of network. So we get *WRLI* is obtained when both source and destination nodes are located at the boundary.

So $WRLI = \max(RLI) = \max(\frac{2(R-r)+K}{d})$ with the constraint $\frac{d}{R} = \frac{K}{r}$. We will use Lagrange Multiplier to solve this problem. (Lagrange Multiplier here solves the maximum

not minimum as min(RLI) = 1 when $d = 2R$ and $K = 2r$). Let $L = \frac{2(R-r)+K}{d} + \lambda(dr - KR)$. Here we will solve the Lagrange Multiplier based on mathematical method by assuming R,r,d,K are continuous variables, then apply the physical meaning of these variables in 802.15.5 to interpret the theoretical mathematical upper bound *WRLI*. Taking the gradient for the variable d, r and λ respectively:

$$\frac{\partial L}{\partial d} = -\frac{(2(R-r)+K)}{d^2} + \lambda r = 0 \tag{3.1}$$

$$\frac{\partial L}{\partial r} = -\frac{2}{d} + \lambda d = 0 \tag{3.2}$$

$$\frac{\partial L}{\partial \lambda} = dr - KR = 0 \tag{3.3}$$

It is clearly that L is differentiable with d,r and λ since $\frac{\partial L}{\partial d}$ is continuous with $d(d > 0)$, while $\frac{\partial L}{\partial r}$ and $\frac{\partial L}{\partial \lambda}$ are constant values. From Eqs (3.1)(3.2)(3.3), *RLI* gets maximum when $r = \frac{2R+K}{4}$ and $d = \frac{4RK}{2R+K}$. Therefore, $WRLI = \frac{(2R+K)^2}{8RK} = \frac{1}{2}\frac{R}{K} + \frac{1}{8}\frac{K}{R} + \frac{1}{2}$.

It is clear that $WRLI = 1$ when $K = 2R$. In this case, the K-hop local link state information actually becomes the global link state information so that we can always find the shortest path. Based on this analysis, we further try to see how K can influence the routing performance. Denoting $\rho = \frac{R}{K}$, then $WRLI = \frac{1}{2}\rho + \frac{1}{8\rho} + \frac{1}{2}$, as we have $1 \leq K < 2R$, so $\frac{1}{2} < \rho \leq R$. The gradient of *WRLI* via ρ will be $\frac{\partial WRLI}{\partial \rho} = 1 - \frac{1}{8\rho^2} > 0$, which means *WRLI* monotonically increases via ρ, that is, larger R and smaller K will lead to worse *WRLI*. Therefore, in order to find a shorter routing path, we need to increase K to get enhanced local link state information. However, a large K will cause more memory occupancy and message overhead as pointed in scalability part. In reality, if we assume a network is large, K is much less than R, so we get $WRLI \approx \frac{R}{2K}$. Compared with tree based routing, 802.15.5 is approximately $4K$ times better than the tree routing regarding *WRLI*.

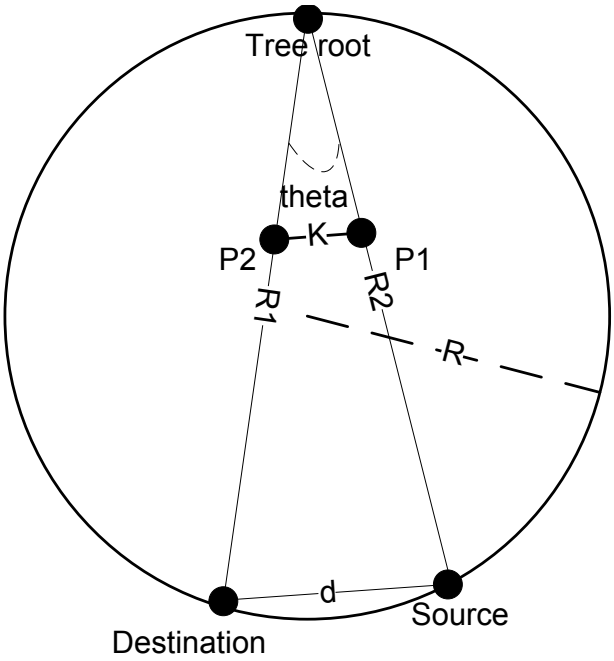


Figure 3.5: An example of routing process when the tree root is at the boundary of a network.

(2) Tree root at the border of a network

From Fig. 3.5, we have $WRLI = \max(\frac{R_1+R_2}{d})$, the condition $R_1 = R_2$ gets *WRLI*. The problem can be solved by using the same approach as the case when the root is situated at the center, except $R_1 = R_2 < 2R$ and $WRLI = \max(\frac{2(2R-r)+K}{d})$ with the constraint $\frac{d}{2R} = \frac{K}{r}$. Using again the Lagrange Multiplier, $WRLI = \frac{(4R+K)^2}{16RK} = \frac{R}{K} + \frac{1}{16}\frac{K}{R} + \frac{1}{2}$ when $r = \frac{4R+K}{4}$ and $d = \frac{8RK}{4R+K}$. Finally we get $WRLI \simeq \frac{R}{K}$. Compared to the tree based routing($WRLI = 4R$), 802.15.5 is still $4k$ times better.

WRLI is obtained when the source and destination nodes are located farthest from the tree root and have the same distance to the root. So $\frac{R}{K}$ is the upper bound of *WRLI* for 802.15.5 regardless of the location of the tree root.

3.2.4 Mobility Support Overhead

Mobility support scheme has been introduced in the former chapter. In this section, we focus on the scenario where a sender in a network tries to find mobile nodes. Since tree based routings do not support the node mobility, we just compare 802.15.5 scheme with that of Zigbee PRO. We will analyze the message exchange overhead for routing re-establishment to mobile nodes for one time movement (i.e., a mobile node from its original place to some other place). The overhead for multiple movements can be easily derived from the result of one time movement. We assume $\alpha(0 < \alpha \leq 1)$ percentage of nodes in a network will send packet to $\beta N(0 < \beta \leq 1)$ mobile nodes.

- Zigbee PRO

Similarly to the scalability analysis, every node needs $N+R$ message exchanges on the average to find a new route to a mobile node. For αN nodes, $\alpha N(N+R)$ message exchanges are consumed to find new routes to one mobile node. To find routes to βN mobile nodes, it needs $\alpha\beta(N+R)N^2$ message exchanges, or $O(N^3)$ complexity.

- 802.15.5

For a sender to find a route to a mobile node, it takes $4R$ message exchanges for the worst case because it costs $2R$ message exchanges for the mobile node to notify its previous parent and $2R$ for the previous parent to notify the sender. Note that the best case costs 0(e.g., the mobile node is still within one hop range of current parents). Thus, on the average $2R$ message exchanges are needed in addition to $2(\frac{K}{R})^2N$ message exchanges to update the local link state information by mobile node and its previous parent. So the total becomes $2R + 2(\frac{K}{R})^2N$. (The 1 time broadcast by sender at the end is ignored to simplify the equation). For αN source nodes to find βN mobile nodes, it requires

$[2R + 2(\frac{K}{R})^2 N] \times \alpha\beta N^2$ message exchanges. Once a sender receives the new address of a mobile node, it will broadcast this information within its one hop neighbors so that those nodes receive this message do not incur any overhead to get the mobile information. Since the average one hop neighbors for a node is M, averagely the total number of message exchanges becomes $\frac{[2R+2(\frac{K}{R})^2 N]\times\alpha\beta N^2}{M}$, which is rewritten as $(2R^3 + 2K^2 N) \times \alpha\beta N$ because $M = \frac{N}{R^2}$. For the fixed network size (constant R)the complexity is $O(N^2)$. For the fixed node density (constant M), $R = \sqrt{\frac{N}{M}}$, the overhead is $2\alpha\beta\frac{N^{\frac{5}{2}}}{M^{\frac{3}{2}}} + 2K^2 a\beta N^2$ or $O(N^{\frac{5}{2}})$ complexity.

If the mobile nodes are not leaf nodes, then the whole tree structure needs to be re-established. The message overhead is the same as the network-wide routing information establishment, as shown in the scalability analysis.

3.2.5 Summary of Analyses

Finally, we summarize all the results of analyses in Table 3.1.

3.3 Simulation

The 802.15.5 has already been successfully implemented on our test-bed and is shown to perform well [47]. In this paper, we will demonstrate the validity of our analyses done in the previous section. We conduct simulations for the network layer behavior using the Matlab. A IEEE 802.15.4 Matlab model is used for underline MAC/PHY [70] and a probabilistic radio model is applied assuming radio range follows the Gaussian distribution with the mean value of 10m and standard deviation of 3m.

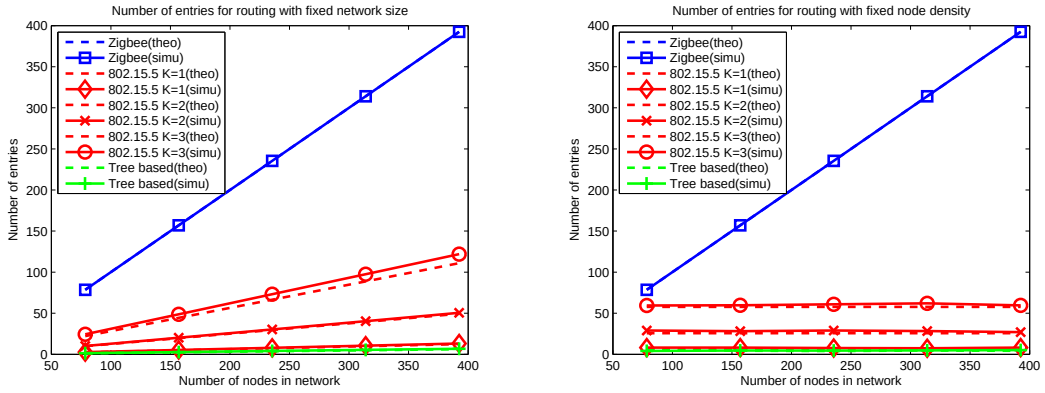
Two different simulation scenarios are used. First, we will fix geographic size of the network so that increasing the number of nodes will increase the node density. We assume nodes are distributed in a circular space with the radius of 50 meters in a grid topology of square cells and then a random shift is added to every node. The direction of the shift is randomly chosen between 0 and 2π and the distance of the shift is randomly chosen between 0 and a half length of the side of a cell. In the second scenario, we will use fixed node density by setting the node density with the length of a cell side 7 meters(constant M) so that increasing the number of nodes will increase the geographic size of the network. Random shifts are added the same way. The tree root is located at the center of a network except the experiments for the shortest path routing where the tree root can be located at the boundary of the network. We will run 10 different topologies for each experiment and take the average for the final results.

- Scalability

Fig. 3.6 shows the memory occupancy versus the number of nodes. Fig.3.6(a) shows that all three algorithms grow linearly with the number of nodes and Zigbee PRO has the largest increasing slope, while in Fig. 3.6(b) only Zigbee PRO shows a linear growth but 802.15.5 and the tree remain constant, the results of which support the analytical results.

Table 3.1: Summary of Analyses

	Zigbee PRO	802.15.5	Tree based scheme
Memory Occupancy	N $O(N)$	$\frac{K^2}{R^2} \times N$	$\frac{1}{R^2} \times N$
		$O(N)$ for fixed network size $O(1)$ for fixed node density	
Routing establishment overhead	$(N-1)(N-2)(N+R)+N$ $O(N^3)$	$(\frac{K}{R})^2 \times N^2$	$(\frac{1}{R})^2 \times N^2$
		$O(N^2)$ for fixed network size $O(N)$ for fixed node density	
NDN	1	$\geq \frac{N}{3R^2}$	1
$MTTF_R$ Mean time of routing failure (T_F:Mean time of node failure) (H: Number of hops of route path)	T_F/H	$\geq \sum_{i=1}^{\frac{N}{3R^2}} \frac{T_F}{i \times H}$	T_F/H
$WRLI$	1	$\frac{R}{2K}$: root at the center $\frac{R}{K}$: root at the border	$2R$: root at the center $4R$: root at the border
Mobility support overhead (α is percentage of sender nodes) (β is percentage of mobile nodes)	$\alpha\beta(N+R)N^2$ $O(N^3)$	$(2R^3 + 2K^2N) \times \alpha\beta N$ $O(N^2)$ Fixed network size $O(N^{\frac{5}{2}})$ Fixed node density	No mobility support

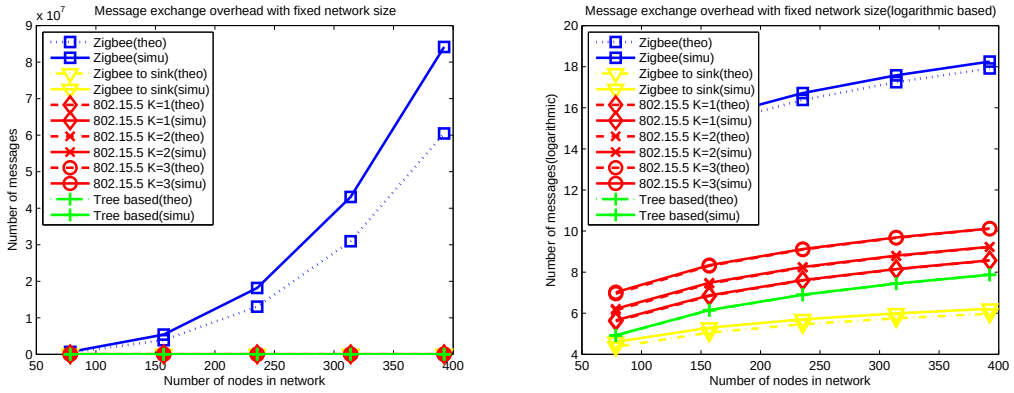


(a) (b)

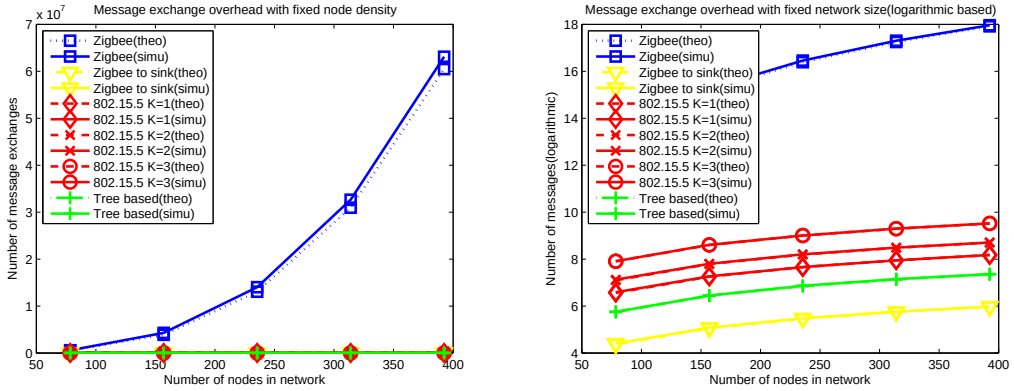
Figure 3.6: Scalability as the routing information size versus the number of nodes in a network. (a) is for fixed network size and (b) is for fixed node density. Dashed lines are theoretical results(theo) and solid lines are simulation results(simu).

Zigbee PRO consumes most memory space and memory increasing speed via number of nodes is the fastest. For 802.15.5, large K leads to high memory occupancy. However, it still performs much better than Zigbee PRO and is comparable with that of the tree based routing especially when K is small. Meanwhile, if the node density is fixed, memory occupancies of 802.15.5 and the tree based scheme remain constant regardless of the number of nodes.

Fig. 3.7 demonstrates the message exchange overhead to establish network-wide routing versus the number of nodes in a network. The analysis in the previous section shows that the three schemes have different big O in term of the number of nodes. To demonstrate the validity of the trends, we give out the original values and also corresponding logarithmic curves presenting the visible differences. Note that the slopes of the logarithmic curves show exactly those big O trends as it represents the power of N in big O notations. Zigbee PRO costs far more overhead (i.e., steepest slope), or energy, compared with 802.15.5 and the tree based routing in building the network-wide routes. And the



(a)



(b)

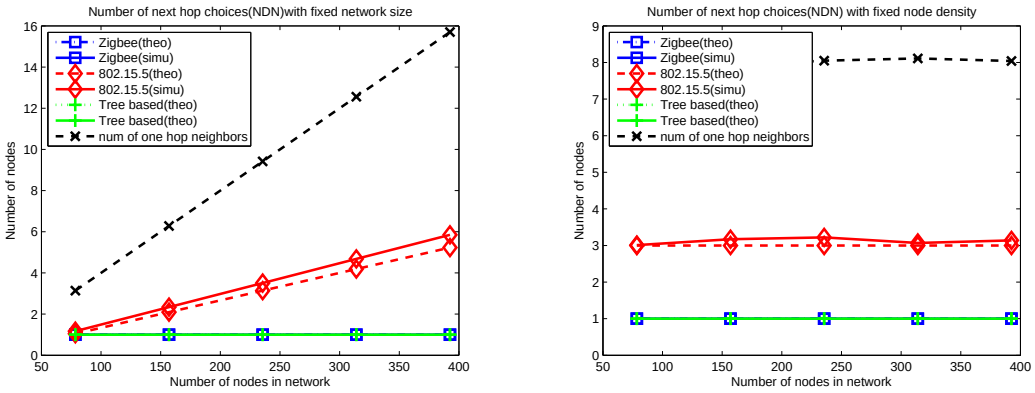
Figure 3.7: Scalability as the number of messages needed for network-wide routing establishment versus the number of nodes in a network. (a) is for fixed network size and (b) is for fixed node density. For a better visible comparison, we plot in log scale for both cases. The left side figures are in the original values and right side figures are in log scale. Dashed lines are theoretical results and solid lines are simulation results. Dashed lines are theoretical results(theo) and solid lines are simulation results(simu).

network wide broadcast cause serious MAC collision so that message exchange in simulation is more than theoretical value, especially in fixed network size case where high node density will cause more MAC collision. Thanks to many-to-one scheme, if routings are required only from sensor nodes to a sink, Zigbee PRO costs the least overhead. For 802.15.5, large K leads to more message exchanges and its value is always larger than the tree based scheme due to the overhead incurred from exchanging the local link state information, but the slopes are the same. As the link state information exchange is done in local region, the influence of MAC collision is much less. Meanwhile, 802.15.5 and the tree based scheme perform better when the node density is fixed, where the slope of the overhead is the same as Zigbee PRO only-to-sink case, while for the fixed network size, its overhead is larger than Zigbee PRO only-to-sink case but still lower than Zigbee PRO network wide routes case.

Overall, 802.15.5 is better than Zigbee PRO by an order of magnitude and comparable to the tree based scheme.In particular, it has a better scalability, i.e., $O(N)$, when the node density is fixed. In reality, this provides a desirable advantage for network developers and operators as most practical networks maintain the same node density when updated to larger area coverage.

- Fault Tolerance(NDN)

Next, we will compare the number of potential next hop nodes for three algorithms (default $K = 2$ used for 802.15.5). 20 pairs of source and destination nodes are randomly chosen. For the NDN simulation, every node in routing path will calculate the node degree of next hop choices (NDN) and the final NDN is the average of all. Fig. 3.8 supports the analytical results and demonstrates that only 802.15.5 can provide multiple next hop choices without route re-establishment: there is only one next hop choice for Zigbee and

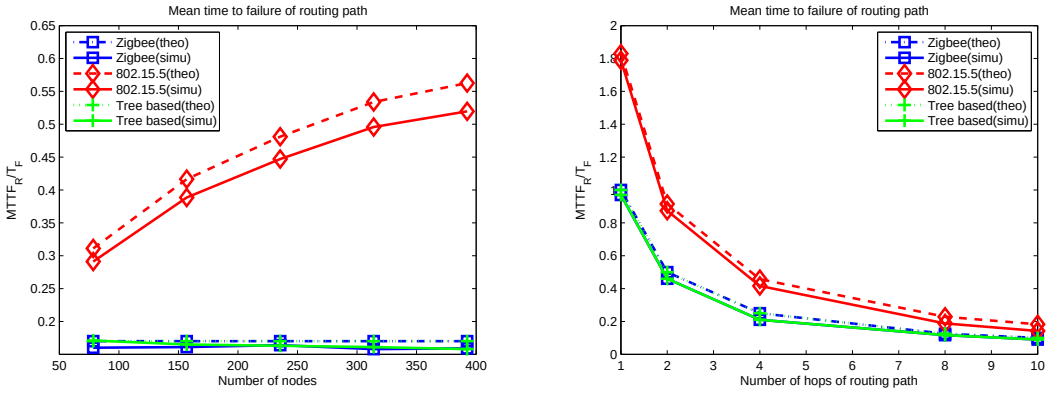


(a) (b)

Figure 3.8: Node degree of next hop choices(NDN). (a) is for the fixed network size and (b) is for the fixed node density. The number of one hop neighbors is also shown to give a better illustration. Note the NDN of tree based scheme and Zigbee PRO are both "1" and overlapped in this figure. Dashed lines are theoretical results(theo) and solid lines are simulation results(simu).

Tree, while for 802.15.5, its NDN grows linearly in the fixed network size case and keeps constant around 3 for the fixed node density case. A large node degree of next hops choices provides more resilience to node failures. In this sense, 802.15.5 is more robust compared with Zigbee PRO and tree based schemes.

Regarding the $MTTF_R$, slightly different experimental setup is used as in NDN case. We will measure the ratio between MTTF of routing path to MTTF of single node($MTTF_R/T_F$), for which the larger ratio means long lifetime of routing path assuming the MTTF of node is predetermined. For the fixed network size case, $MTTF_R/T_F$ is measured along with node number by taking the average of 20 routing paths; For the fixed node density case, $MTTF_R/T_F$ is measured along with hop count of routing path. Fig. 3.9 indicates that 802.15.5 has much larger MTTF regarding routing path so that lifetime of routing path is longer in both cases. Also high node density will extend the lifetime of routing path in 802.15.5 as more next hop nodes are available.

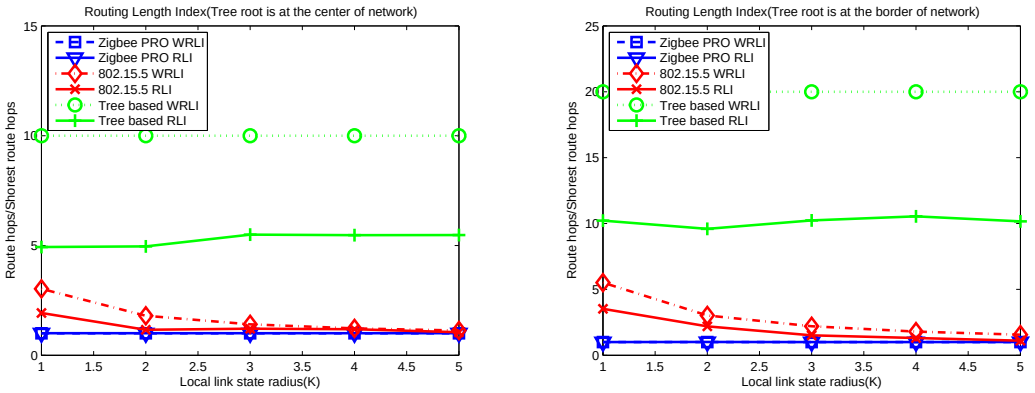


(a) (b)

Figure 3.9: Mean time to failure of routing path($MTTF_R$): ratio between MTTF of routing path to MTTF of single node($MTTF_R/T_F$). (a) is for fixed network size where $MTTF_R/T_F$ is measured along with node number by taking the average of 20 routing paths. (b) is for fixed node density case where $MTTF_R/T_F$ is measured along with hop count of routing path. Dashed lines are theoretical results(theo) and solid lines are simulation results(simu).

- Routing Path Length

For the experiments regarding Routing Length Index, we fix the network radius at 50 meters and the cell side length at 7 meters. Probabilistic radio model is kept the same as before. We randomly choose 50 pairs of source and destination nodes to calculate their *RLI* and take the average for final results. Fig. 3.10 shows the *RLI* performance when the local link state radius K varies, together with pre-calculated theoretical *WRLI*. As expected, Zigbee PRO is the best to find the shortest path, due to probabilistic radio model, the simulation results are slightly larger than 1. The tree based scheme comes the worst. 802.15.5 has much better performance than tree based scheme and is competitive with Zigbee PRO especially when local link state radius K is larger, which gives more chance to find shorter paths, but at the expense of memory and message exchange overhead. In this sense, 802.15.5 can position itself in between Zigbee PRO and tree based scheme



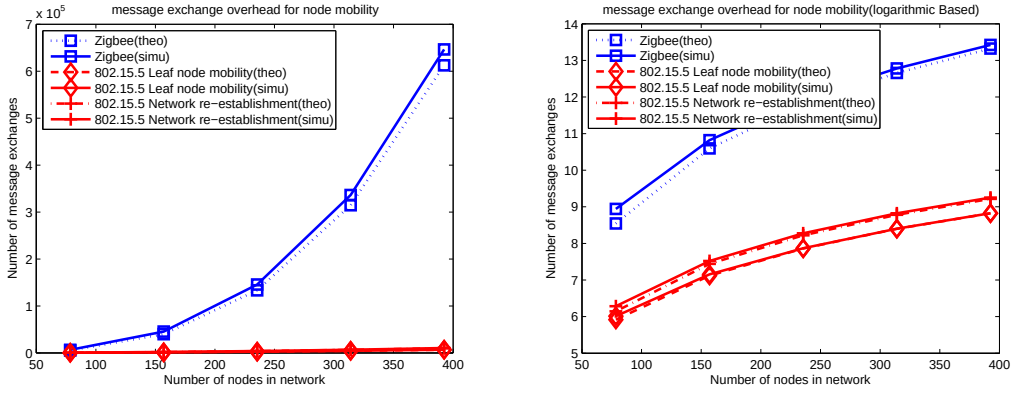
(a) (b)

Figure 3.10: Routing Length Index *RLI* and *WRLI*. (a) is for the case where tree root is located at the center of a network and (b) is for the case where tree root is located at the boundary of a network.

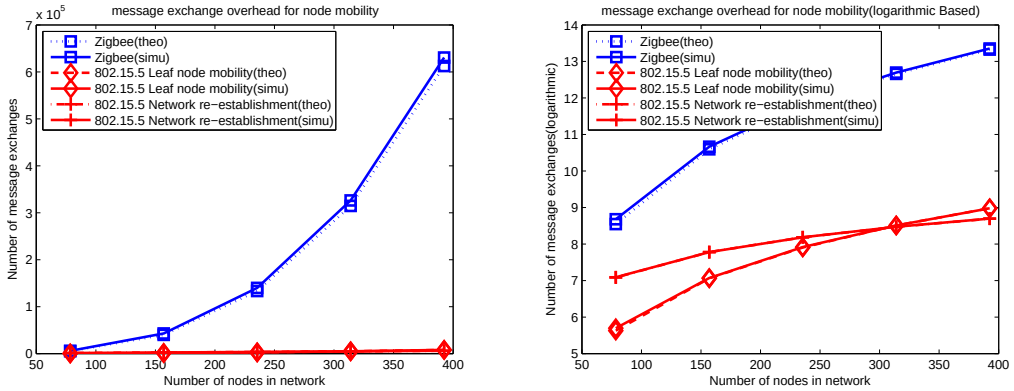
given the required quality of service by controlling local link state radius K.

- Mobility Overhead

For the simulation of the mobility, we assume $\alpha = 0.1, \beta = 0.1$ and consider only one time movement overhead. Mobile nodes are randomly placed inside a network and then the movement for each mobile node is chosen randomly with a direction between 0 and 2π and a distance between 0 and R. We also use a reflection mobility model that if any mobile node has arrived at the boundary of a network it will reflect back and continue to move. Fig. 3.11 shows the message exchange overhead to re-establish the routing from source nodes to mobile nodes after their movements. Same as the scalability part, we plot both original values (left side figures) and logarithmic values(right side figures). The validity of analyses is confirmed by Fig. 3.11(a) and 3.11(b). In both cases, Zigbee PRO pays much more to support the mobility regarding message exchange overhead. In the worst case where mobile nodes are not leaf nodes but intermediate routers, 802.15.5 needs to re-establish the whole tree structure and local link state information. Even in this case,



(a)



(b)

Figure 3.11: Message overhead for mobility support. (a) is for fixed network size and (b) is for fixed node density. For a better visible comparison, we plot in log scale same as in scalability part. left side figures are in the original values and right side figures are in log scale. Dashed lines are theoretical results(theo) and solid lines are simulation results(simu).

802.15.5 still costs less compared to Zigbee PRO. What is not accounted for, however, is the time for network re-initialization. In that sense, we believe 802.15.5 is not suitable to support the network dynamics like in mobile ad hoc networks. In contrast, 802.15.5 is suitbale for low power large scale wireless sensor networks in which most sensor nodes are stationary, mobility is limited, and the network life time is expected to be long (e.g., for years), so the network initialization time is tolerant.

For the case of fixed node density(Fig. 3.11(b)), an interesting observation is that the overhead for the leaf node mobility support exceeds slightly the overhead to re-establish the whole network as the network size increases beyond a certain threshold. We reason this can happen mainly because the leaf node mobility support will pay for routes between previous parents and current parents, mostly leading to a much longer path. If the number of nodes N is small (also R is small considering fixed density), then the overhead for network establishment is $(\frac{K}{R})^2 \times N^2$ which is larger than that of the mobility support $0.02\frac{N^{\frac{5}{2}}}{M^{\frac{3}{2}}} + 0.01K^2N^2$ as $\alpha = \beta = 0.1$. As network size expands via the number of nodes, the mobility support becomes expensive because the extra overhead for current parents to notify previous parents and packet re-route costs excessively. However, the mobility support dose not incur any additional network re-establishment time.

Based on the analyses and simulation results, we can see that 802.15.5 maintains a consistently competitive performance as it is either the best or comparable with best. Furthermore, 802.15.5 dose not suffer any significant shortage, while Zigbee PRO and tree based scheme can achieve best performance in certain metrics by sacrificing much more in others. Another advantage of 802.15.5 is that it can provide flexibility by adjusting local link state K to meet different network requirements, which is hardly offered by Zigbee PRO and tree based scheme.

3.4 Summary

In this chapter, we present a comprehensive analysis on 802.15.5 low rate part. As a low power wireless mesh network standard, it should be simple enough to fit in limited memory and energy to ensure good scalability. Also 802.15.5 needs to provide certain fault tolerant capability. Through the complexity analyses and accompanying simulation studies of network layer performance, 802.15.5 is shown to be the preferred choice over ZigBee Pro and the tree-based scheme. Importantly, it does not suffer from any particular shortage so that it has the potential to be applied to a variety of diversified applications.

Chapter 4

Mobility Support Analysis in WSN

4.1 Sink node mobility in Wireless Sensor Networks(WSN)

In this chapter, we will expand our mobility support scheme in IEEE 802.15.5 to more general case: We will discuss the sink node mobility support scheme in wireless sensor networks and propose a distributed mobile sink support in wireless sensor networks.

The typical application for WSNs is the sensor nodes gathering data and reporting to sink. All data traffics are from sources to a or multiple sink nodes. Recently, data dissemination protocols for WSNs have been extensively studied. In these works, sink nodes are assumed to be stationary and the sink mobility is hardly an issue. Nevertheless, in many real applications, the sinks are expected to be mobile when they are integrated with other mobile devices such as mobile phones and PDAs carried by mobile user. In such cases, sink mobility management becomes an important issue in designing WSNs. How to keep the sensors informed of the current state of the mobile sink is the primary issue for the mobile sink management. In this chapter, we propose a distributed mobility management scheme that uses a set of access points (APs) to support the data transmission

from sensors to mobile sink.

Sink node mobility brings two challenges: to provide means to reach the mobile sink node and how to maintain ongoing session without interruption while on the move. The former is called location management and the latter as handover management. The typical application in sensor network is that sensors report intermittent event or short periodical data to sink, in which one session presumably contains one packet, the problem of which is not as significant as the ones in other networks supporting streaming applications. Therefore in this paper we focus on the location management. Although the issue has been studied intensively for cellular and ad hoc networks, it has very different characteristics in sensor networks. In cellular networks, location management is typically accomplished by using centralized database, e.g., Home Location Register, which stores up-to-date location information of mobile stations. But in WSNs, the realization of such kind of centralized database is not feasible due to signaling overhead for paging method and resource constrained sensor devices. Because the energy consumption is of paramount importance in designing WSNs, a lightweight and distributed approach is preferred for mobility management.

Some earlier works have addressed this problem. TTDD [67] builds a grid topology for each source node and the mobile sink renews the entire path to the source whenever it moves out of range from a local cell. LURP [63] uses local flooding to update location information unless the sink is moving out of range in local cell. Both TTDD and LURP require location aiding devices like GPS for every sensor nodes and it is not cost efficient in many cases. SAFE [45] and SEAD [19] build data dissemination tree which would be reconstructed when the sink is mobile. The tree rebuilding overhead is excessive for sensor network especially when the sink's movement is frequent. EARM [4] tracks the distance of the gateway from the last hop and dynamically adjusts routing.

In this chapter, we propose a distributed mobility management scheme. Instead of broadcasting the whole network its new location information, the mobile sink would choose different access points while moving and update location information through these access points in a distributed way. Potential source nodes will be informed of mobile sink's current location by these distributed access points. Compared with existing works, this distributed management scheme maintains both energy efficiency and balance. Also the proposed method can be integrated with the underlying routing protocols.

4.2 Distributed Mobility Management

The following conditions and assumptions are considered, common in literature for the design of WSNs:

- The sensors nodes are stationary and there exists a certain underlying proactive routing protocol between each node pair.
- Unlike the sensor nodes, the sink has no energy constraint. So, we only take the energy of sensor nodes into consideration.
- Each time a sink moves to a new location, it will choose a sensor node to associate with and this particular sensor node is called access point (AP). All the data traffic toward sink node will first arrive at AP and the AP will relay the data to sink.
- Mobile sink is not a part of the whole network, that is, the movement of mobile sink does not trigger any link state update for the network.
- All sensor nodes will monitor the environment/event with the time period T. Those monitored value exceeding certain threshold will be reported to sink. Therefore,

every node could be potential source, and there may exist multiple sources.

When the network has been deployed, the sink node first chooses an initial AP. Then it broadcasts the initial AP information over the entire network so that all nodes get this initial AP information. Afterward, the sink may be moved along with a mobile robot, a vehicle, or a person. Every T time period, it will choose another sensor node as its new AP and update its status (e.g., current AP) by informing all previous APs.

Whenever a source node wants to send data packet to sink, it first checks its memory for the AP information it has (should be one of previous APs), then sends data to that particular AP. When the data arrives at that particular AP, it will relay data to the current new AP to reach the sink, while sending the current new AP's address back to the source node(s). After the source nodes get the location update, they store the new AP's information and broadcast to their one hop neighbors. In this case, those one hop neighbors of the source nodes will be updated with the new AP information even without sending any data to the mobile sink. This will help when any of these nodes later have data to report. Note here again that we assume an underlying routing protocol taking care of routing packets between any node pair given their addresses. The Figure 4.1 shows an example.

After mobile sink associates with current AP, it will update its location information by informing all former APs using a multicast. Some efficient multicast algorithms such as spanning tree or heuristic Steiner tree may be used to perform the multicast, but the cost of building these trees every time period T will be prohibitively large when the sink is mobile. Hence we just assume a simple multicast algorithm in which current AP unicasts to every former AP.

This distributed method has two major advantages compared with former mobility management scheme. First, no network wide flooding is needed to update the mobility

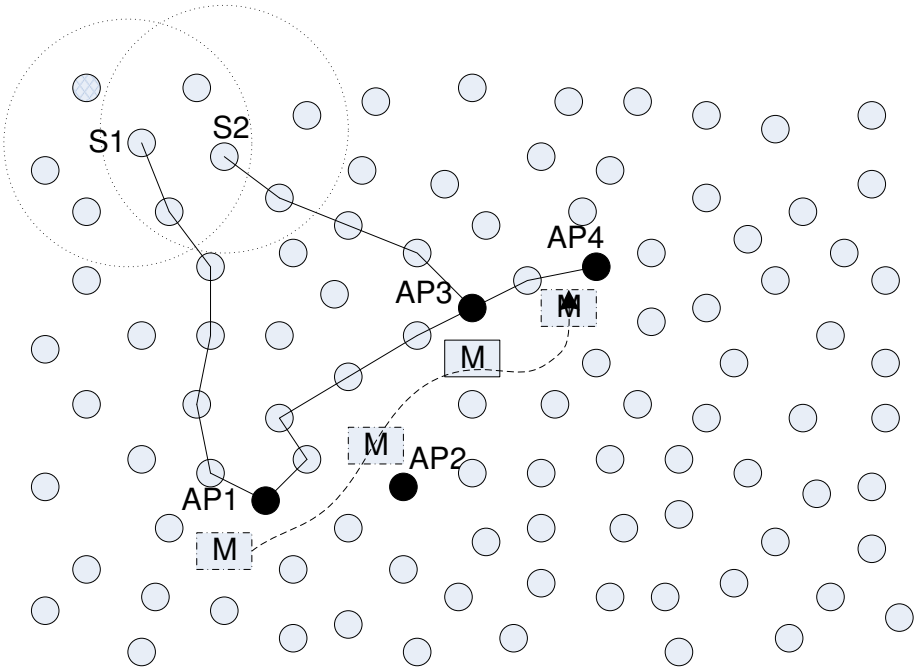


Figure 4.1: Example of Access Point (AP) Management.

information except once for initialization so that it is much more energy efficient. Second, the distributed AP mobility scheme helps balance the energy consumption over all APs. Figure 4.1 shows an example of Access Point (AP) Management: the sink has an initial AP1 and then moves along with the dashed trajectory. After two time periods it will will choose another two APs(AP2 and AP3). At this time S1 wants to send data to mobile sink, but it only has initial AP information(AP1). AP1 receives data from S1 and relay it to AP3, while replies back to S1 with the information of AP3. S1 receives the updated AP information and broadcast to its one hop neighbor so that S2 has AP3's information without sending any data. Next time sink chooses another AP4 as its current new AP, if S2 wants to send data to sink the data packet first arrives at AP3 and AP3 relays it to AP4 while sending back the update with AP4 information. Finally S2 broadcasts this newly updated AP4 to its one hop neighbors.

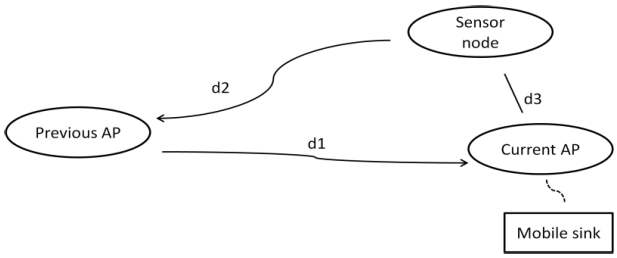


Figure 4.2: A simple model for analysis.

4.3 Performance Analysis

Similar to analysis model used in TTDD and LURP, we consider a squared sensing field whose side length is D, in which N sensor nodes are randomly distributed. Every time T all sensor nodes will perform sensing and those sensing value exceeding a threshold will be reported to mobile sink. In the mean time, the mobile sink will move according to the random waypoint model [12] with velocity v and every time period T it will choose a sensor node as its current AP. The transmission radius of sensor is R. We assume the total operation time of a sensor network is $t\_total = m \times T$. .

Theoretical analysis is done to compare our method with the network wide broadcast , in which mobile sink node will flood the whole network with its new location every time period T , enabling potential source nodes to directly send data to mobile sink. The purpose for this analysis is to compare the communication overhead between these two methods, and to get an approximate threshold below which our method performs better than network wide broadcast.

4.3.1 Analysis Model

We first consider a simple case in which mobile sink just move once and one source reports one packet to mobile sink via a previous AP as showing in Figure 4.2.

- Analysis of Distributed AP Management

For the simple case in Figure 4.2 above the communication overhead is derived as follows:

$$\underbrace{\frac{d_1 * \varepsilon}{R}}_{\text{updateofcurrentAPtopreviousAP}} + \underbrace{\frac{d_2 * (L+\varepsilon)}{R}}_{\substack{source\text{topreviousAPandprevious} \\ \text{APinformsourceofcurrentAP}}} + \underbrace{\frac{d_1 * L}{R}}_{\substack{\text{previousAPrelaypackets} \\ \text{tocurrentAP}}} + \underbrace{\varepsilon}_{\substack{\text{sourcebroadcast} \\ \text{onehoplocally}}} = \frac{(d_1+d_2)(L+\varepsilon)}{R} + \varepsilon \tag{4.1}$$

Where L is the length of data packet and ε is the length of location update packet. d_1 is the distance from current AP to previous AP, d_2 is the distance from source node to previous AP. Also $\frac{d}{R}$ means the hop count between two nodes. Å is the one hop broadcast by source node after it receives updated AP information.

We then extend the simple model to the general case: at each time period T, i nodes are randomly selected to report their data to sink and sink will update its location m times during the entire experiment time $t\_total = m \times T$. Let the communication overhead for our method be E_1, which can be derived as follows:

$$E_1 = \underbrace{\frac{\sum_{k=2}^{m}\sum_{j=1}^{k-1} d_{AP_k}^{AP_j} \times \varepsilon}{R}}_{\substack{current\text{APinformallformerAPs}}} + \underbrace{\frac{\sum_{k=1}^{m}\sum_{j=1}^{i} d_{S_j}^{AP(j)} \times (L+\varepsilon)}{R}}_{\substack{source\text{toAPbasedonitsrecordand} \\ \text{APgivebackcurrentAPupdate}}} + \underbrace{\frac{\sum_{k=2}^{m}\sum_{j=1}^{i} d_{AP(j)}^{AP_k} \times L}{R}}_{relay\text{tocurrentAP}} + \underbrace{m \times i \times \varepsilon}_{\substack{\text{sourcebroadcastto} \\ \text{onehopneighbors}}} + \underbrace{N \times \varepsilon}_{\substack{\text{broadcastnetwork} \\ \text{initialAPinformation}}} \tag{4.2}$$

Where d_i^j means the distance from i to j, AP_i means the AP chosen at i-th time period, $AP(j)$ means the AP information stored in the source node j, and S_j means the source

node j.

- Analysis of Broadcast Based Method

For the simple model in Figure 4.2, the communication overhead of the broadcast based method for a packet from a single source is as follows:

$$N\varepsilon + \frac{d_3}{R}L \tag{4.3}$$

Where N is the number of nodes in network and d_3 is the distance from the source node to current AP. We extend the simple analysis to general case as before, resulting in the communication overhead for broadcast based method:

$$E_2 = m \times N\varepsilon + m \times \frac{\sum_{k=1}^{m}\sum_{j=1}^{i} d_{S_j}^{AP_k} \times (L+\varepsilon)}{R} \tag{4.4}$$

- Analysis of LURP

Based on the broadcast based method analysis and [63], we can extend to the LURP.

$$E_3 = m \times [\alpha(\frac{D_L}{D})^2 + (1-\alpha)] \times N\varepsilon + m \times \frac{\sum_{k=1}^{m}\sum_{j=1}^{i} d_{S_j}^{AP_k} \times (L+\varepsilon)}{R} \tag{4.5}$$

The α means the probability that the sink is inside local cell, D_L is the local cell side length. And $\alpha(\frac{D_L}{D})^2 \times N\varepsilon$ represent the local broadcast communication overhead.

4.3.2 Average Threshold Analysis

An analysis is given out to compare communication overhead of our method with the broadcast based method and local broadcast method. The assumption is the same as

before. To simplify the analysis, we assume $L = \varepsilon$.

In order to measure the approximate threshold for our method compared with broadcast based method, Let $E_1 = E_2$, without loss of generality, we let $d_{S_j}^{AP(j)} = d_{S_j}^{AP_k}$, and $d_{AP(j)}^{AP_k} = \frac{\sqrt{2}D}{2}$. $d_{AP_k}^{AP_j}$ is depend on the velocity of sink node,$d_{AP(j)}^{AP_k} = \frac{v \times t}{2}$. Put all these into Equation 4.2 and 4.4, the average threshold for the number of source node i could be derived:

$$i \approx \frac{NR - \frac{vt}{2}}{\frac{\sqrt{2}D}{2} + R} \tag{4.6}$$

This average threshold analysis shows if the number of source nodes is less than i, averagely speaking our approach performs better than the broadcast approach.

A similar way could be used to derive the threshold for our method compared with LURP. Based on Equation 4.5 and 4.6, the average bounder for our method compared with LURP could be derived as:

$$i \approx \frac{[\alpha(\frac{D_L}{D})^2 + (1 - \alpha)] \times NR - \frac{vt}{2}}{\frac{\sqrt{2}D}{2} + R} \tag{4.7}$$

4.4 Simulation Results

In this section, three methods for mobile sink support are compared: our method, the method with network wide broadcast (flooding), and the one with local broadcast, LURP.. The basic idea of LURP is that it splits the total area into local cells. When the movement of sink is within a local cell, the mobile sink broadcasts its new location only within the local cell, but if the mobile sink moves out of a local cell, it needs flooding the whole network to update its new location information. Both TTDD and LURP are this type of local broadcast based method. We choose LURP for comparison, because the typical application of TTDD differs from ours in that it cares single source to multiple mobile

Table 4.1: Simulation parameters for sink node mobility.

R: Communication range	10m
D: Network side length	100m
t_total: total experiment time	400s
T: mobile sink update period	10s
N: number of nodes in network	1000
L,ε: data and update packet length	59 bytes
J: energy consumption for one bit	0.001 joule
v: sink mobile speed	0.001 joule
D_L: Local cell size	30m
i:source node number	60

sinks and that the data is queried by sink, while our method deals with traffic from multiple sources to sink. Also TTDD requires a location device like GPS. For LURP, the size of the local cell is chosen as 10% of total area (side length of local cell is 0.33D).

The simulation is done with a high level simulator programmed in MATLAB 7, which ignores the PHY and MAC layer. The parameters of network are set as following Table 4.1:

The sink moves following a random waypoint model, in which it randomly chooses to move or stay in each T period. The simulation strategy is to see the impact of each parameter to the communication overhead. we change one parameter every time and keep others fixed so we can see how different parameters influence the communication overhead For each set of parameters, simulation is performed 20 times and the average is taken.

Figure 4.3 shows the impact of source node number to communication overhead. Based on parameters in Table 4.1 and Equation 4.6, the average threshold number for proposed method compared with broadcast is 140. Simulation results show that the bounder is 128, which matches the analysis very well. For LURP, the bounder is 79 based on the simulation. The parameter could be calculated based on Equation 4.7 and TABLE I. in

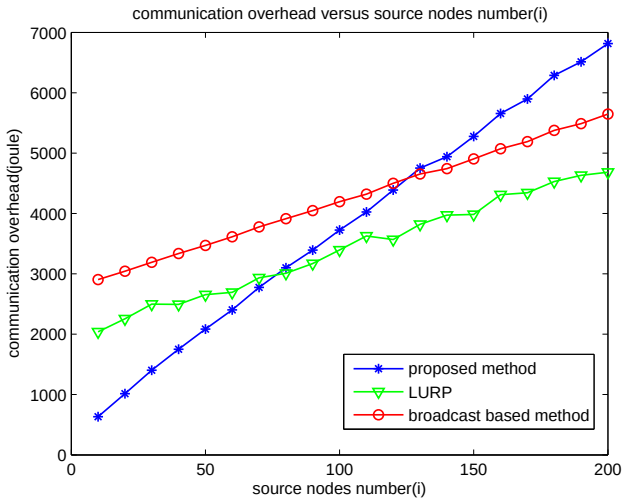


Figure 4.3: The communication overhead vs number of source nodes

this figure, it is about 0.3, which means the mobile sink has 30% to move out of local region. Note here the total number of nodes is about 1000 and there are $t_toal/T = 40$ update times, if we assume every node will report data to sink once in the total experiment time which is reasonable, so every update period 25 sensor nodes will report data, in which case proposed method could perform better than other two.

Figure 4.4 compares the communication overhead versus the update time period T. The communication overhead accounts the total energy associated with movement of the sink during the simulation time. It is clear that communication overhead for all three methods will increase when the update time period T decrease due to the increase of total number of updates $m = \frac{t\_total}{T}$, m is also the number of APs chosen in the whole experiment time. We observe that network wide broadcast incurs worst overhead, regardless of T, as it involves all network nodes. Meanwhile, as T increases (e.g., mobile sink rarely moves), the overheads of all three methods tend to diminish and converge. Extreme case is when the sink does not move, e.g. $t\_total < T$, for which all three methods incur no overheads associated with sink mobility.

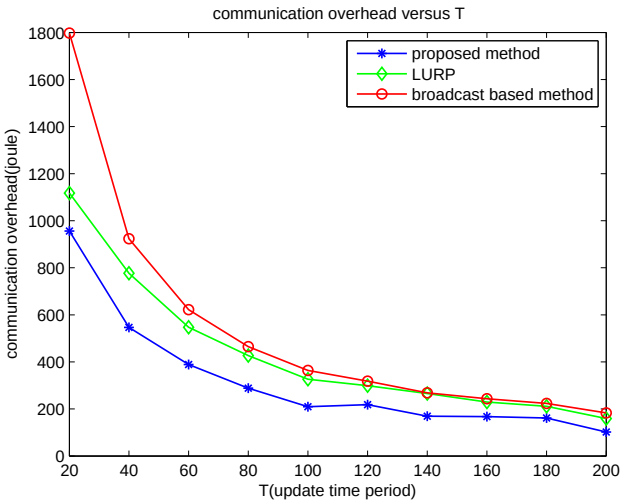


Figure 4.4: The communication overhead vs update time periods

Figure 4.5 shows the communication overhead with different nodes number in network. The communication overhead for broadcast and LURP will increase along with the increase of the number of nodes N. Because a higher N means more broadcast messages and more energy consumption, on the other hand, our methods are not influenced by nodes number N. The results match the theoretical analysis in Equation 4.2 and 4.3 very well.

Figure 4.6 shows the communication overhead with different network size, when D increase, the corresponding distance $d_{AP_k}^{AP_j}$ and $d_{S_j}^{AP(j)}$ for our method in Equation 4.2 will increase, resulting the overhead increase. In the mean time, the increase of $d_{S_j}^{AP_k}$ in Equation 4.3 leads the increasing overhead for network wide broadcast method and LURP. In this case, proposed algorithm still outperforms other two.

Figure 4.7 shows the communication overhead versus mobile sink speed. Broadcast based method will flood the whole network every time T so its overhead is independent of the speed. Although the overhead of our method depends on the mobile speed, the average maintains independent from it because the source nodes are randomly chosen. In

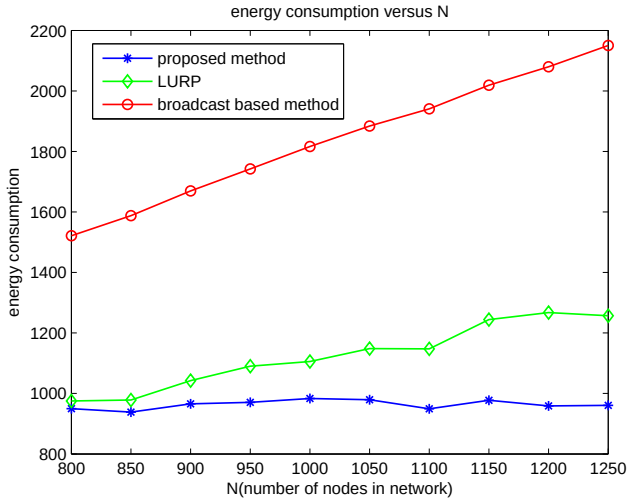


Figure 4.5: The communication overhead vs number of total nodes in network

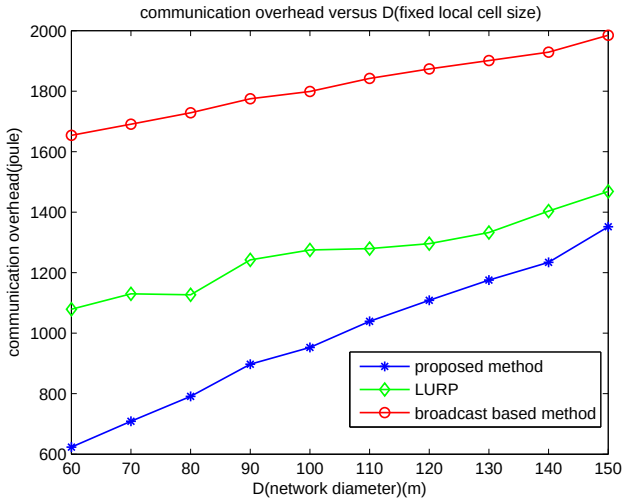


Figure 4.6: The communication overhead vs network diameter D with fixed local cell size (10% of total network size)

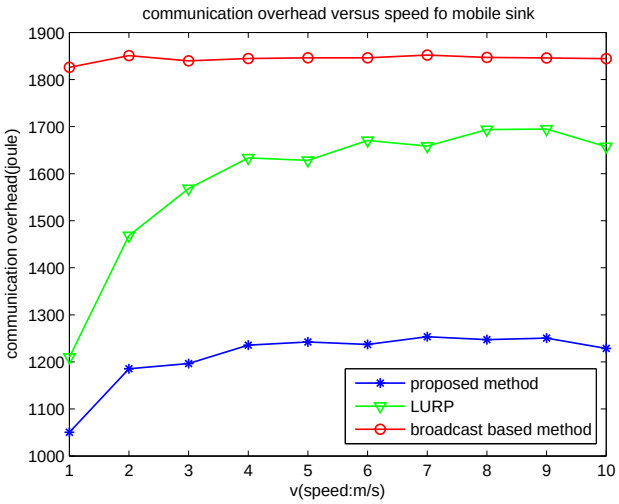


Figure 4.7: The communication overhead vs mobile sink speed

LURP, with higher speed the mobile sink has a lager chance to move out of local cell, causing increased overhead due to more frequent network- wide flooding.

From the above four simulations we can see that our method performs much better than other two methods. It is clear that the broadcast is the dominant factor for communication overhead. Therefore, the reduction of the broadcast should come first in the design of WSNs.

4.5 Summary

In this chapter, we proposed a distributed mobile sink support in wireless sensor networks. Instead of flooding the whole network, the mobile sink would choose different access points while moving and update location information through these access points in a distributed way. Also, potential source nodes will be informed of the sink mobility by these access points. Compared with other algorithms like network wide broadcast and local broadcast method, LURP, our algorithm exhibits better performances consistently

over several network parameters such as network size and mobile speed, supported by both theoretical analysis and simulation.

The future work is to consider multiple mobile sink nodes and different data report period for different source nodes. Also periodic traffic studied in this paper will be extended to include burst and other traffic models.

Chapter 5

Correlation-TCP: TCP Design for Wireless Mesh Networks

5.1 Overview

The former chapters in this thesis give a comprehensive study the IEEE 802.15.5 low rate part and its possible expansion in Wireless Sensor Networks(WSNs), which target the networking layer of Wireless Mesh Networks. In this chapter, we will switch the focus on transportation layer of Wireless Mesh Networks. We will target the most widely used transportation layer protocol: Transmission Control Protol(TCP) over Wireless Mesh Networks. In this Chapter, we will present Correlation-TCP, an enhanced TCP designed for wireless mesh networks based on Newreno. The idea of Correlation-TCP is inspired by the investigation that a high correlation exists between congestion window and RTT for a certain TCP session over wireless mesh networks. This particularly high correlation session occupies most network resources while its throughput is almost saturated; consequently, there is only dminishing gain by continuously increasing its congestion window.

Instead, if we can restrain its congestion window and release excessive network resources it occupies, all other TCP sessions can benefit from lower delays or higher throughputs, with limited throughput loss of high correlation session thus total system performance is improved. Correlation-TCP introduces an additional function in Newreno algorithm to control congestion window by exploring correlation information. Compared with other TCPs proposed for wireless mesh networks that require help and collaboration from intermediate nodes, Correlation-TCP makes use of the information available at the TCP end host. It is compatible with Newreno in Internet and has shown enhanced performance in wireless mesh networks. Extensive experiments will be performed to demonstrate its substantial performance improvement compared with widely used loss (or)and delay based TCPs in wireless mesh networks. Moreover, Correlation-TCP has the flexibility to meet various application layer requirements by tuning its parameters.

5.2 Background and Related Works

5.2.1 TCP Basics

It is well known that TCP is a connection-oriented transport protocol that is aimed at guaranteeing end-to-end reliable ordered delivery of data packets over wired networks. For this purpose, basic functionalities such as flow control, error control, and congestion control are indispensable. While these functions have a clean-cut definition of their own, in practice they are closely coupled with one another in TCP implementation.

In TCP, a sliding window protocol is used to implement flow control, in which three windows are used, namely, Congestion Window, advertised window, and Transmission Window. Congestion window indicates the maximum number of segments (Without caus-

ing confusion, the term segment and packet are used interchangeably henceforth) that the sender can transmit without congesting the network. As shown next in details on congestion control, this number is determined by the sender based on the feedback from the network. advertised window, however, is specified by the receiver in the acknowledgements it. Advertised window indicates to the sender the amount of data the receiver is ready to receive in the future. Normally, it equals to the available buffer size at the receiver in order to prevent buffer overflow. Transmission window means the maximum number of segments that the sender can transmit at one time without receiving any ACKs from the receiver. Its lower edge indicates the highest numbered segment acknowledged by the receiver. Obviously,to avoid network congestion and receiver buffer overflow, the size of transmission window is determined as the minimum of the congestion window and the receiver's advertised window.

To notify the sender that data is correctly received, TCP employs a cumulative acknowledgement (ACK) mechanism. In other words, upon the receipt of an ACK, the sender knows that all previously transmitted data segments with a sequence number less than the one indicated in the ACK are correctly received at the receiver. In the case that an out-of-order segment (identified on the basis of sequence numbers) arrives at the receiver, a duplicate ACK is generated and sent back to the sender. It is important to note that in wired networks, an out-of-order delivery usually implies a packet loss. If three duplicate cumulative ACKs are received, the sender will assume the packet is lost. A packet loss is also assumed if the sender does not receive an ACK for the packet within a timeout interval called retransmission timeout (RTO), which is dynamically computed as the estimated round-trip time (RTT) plus four times the mean deviation. By retransmitting the lost packet, TCP achieves reliable data delivery. It turns out that in wired networks, almost all the packet losses are due to network congestion rather than transmission errors.

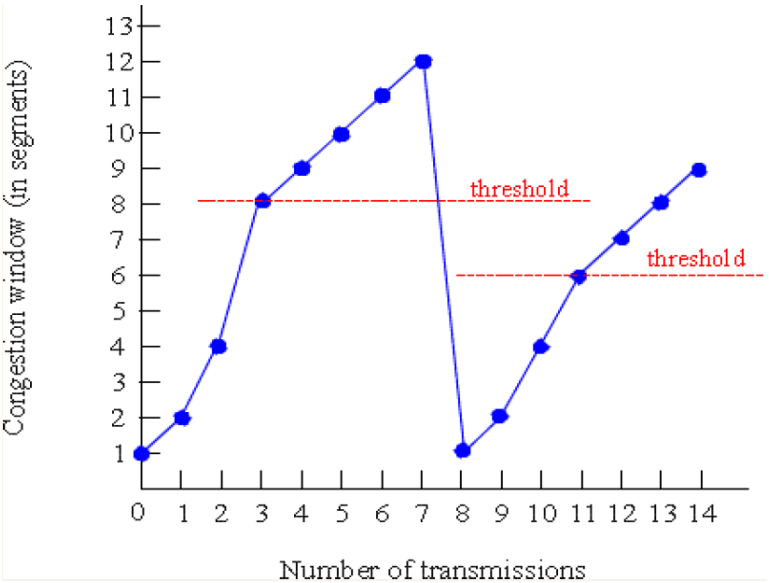


Figure 5.1: TCP congestion window dynamics.

Thus, in addition to retransmission, TCP responds to packet losses by invoking its congestion control mechanism. TCP congestion control is also based on the sliding window mechanism described above and consists of two major phases: slow start and congestion avoidance. In the slow start phase, the initial congestion window size (cwnd) is set to one maximum segment size (MSS) and is incremented by one MSS on each new acknowledgement. After cwnd reaches a preset threshold (ssthresh), the congestion avoidance starts and it is increased linearly, i.e., it is increased by one segment for each RTT. Upon a timeout, ssthresh is set to the half of the current transmission window size (but at least two segments) and the congestion window is reduced to 1 MSS. Then slow start mechanism starts again. This procedure is also called the additive increase and multiplicative decrease algorithm (AIMD). The entire congestion control algorithm is illustrated in Figure 5.1. Note that the sender reacts to three duplicate ACKs in a different way, which is described in fast retransmission and fast recovery in the next subsection.

State-of-the-Art in Standard TCP

Most of the progress made in TCP is centered on error recovery and congestion control. Representative innovations include fast transmissions and fast recovery [27], selective acknowledgements [], random early detection (RED [29]) in routers, and explicit congestion notification (ECN [28]). Notice that depending on what features are included, there are several TCP flavors, including TCP Tahoe, TCP Reno, TCP New Reno, etc. Among them, TCP Reno is by far most widely deployed. Next, we briefly describe these innovations in the following.

- Fast retransmission and fast recovery

As noted earlier, a packet can be assumed lost if three duplicate ACKs are received. In this case, TCP performs a fast retransmission of the packet. This mechanism allows TCP to avoid a lengthy timeout during which no data is transferred. At the same time, ssthresh is set to one half of the current congestion window, i.e., cwnd, and cwnd is set to ssthresh plus three segments. If the ACK is received approximately one round trip after the missing segment is retransmitted, fast recovery is entered. That is, instead of setting cwnd to one segment and starting with slow start, TCP sets cwnd to ssthresh, and then steps into congestion avoidance phase. However, only one packet loss can be recovered during fast retransmission and fast recovery. Additional packet losses in the same window may require that the RTO expire before retransmission.

- Select Ack(SACK)

Owing to the fact that fast retransmission and fast recovery can only handle one packet loss from one window of data, TCP may experience poor performance when multiple

packets are lost in one window. To overcome this limitation, recently the selective acknowledgement option (SACK) is suggested as an addition to the standard TCP implementation.

The SACK extension adopts two TCP options. One is an enabling option, which may be sent to indicate that the SACK option can be used upon connection establishment. The other is the SACK option itself, which may be sent by TCP receiver over an established connection if SACK option is enabled through sending the first option. The SACK option contains up to four (or three, if SACK is used in conjunction with the Timestamp option used for RTTM [24]) SACK blocks, which specifies contiguous blocks of the received data. Each SACK block consists of two sequence numbers which delimit the range of data the receiver has received and queued. A receiver can add the SACK option to ACKs it sends back to a SACK-enabled sender. In the event of multiple losses within a window, the sender can infer which packets have been lost and should be retransmitted using the information provided in the SACK blocks. A SACK-enabled sender can retransmit multiple lost packets in one RTT instead of detecting only one lost packet in each RTT.

- Random Early Detection (RED)

Random Early Detection (RED) is a router-based congestion control mechanism that seeks to detect incipient congestion and notify some TCP senders of congestion by controlling the average queue size at the router. To notify the TCP senders of congestion, the router may mark or drop packets, depending on whether the senders are cooperative. As a response, the senders should reduce their transmission rate. This is done in two algorithms. The first algorithm is to compute the average queue size by using exponential weighted moving average. If we denote by avg and q the average queue size and the current queue size, respectively, then $avg = (1 - wq) \times avg + wq \times q$, where wq

is the queue weight. The other algorithm is to compute the packet-marking or packet-dropping probability pa. If avg falls in between minth and maxth, the packet marking probability $pb = maxp(avg - minth)/(maxth - minth)$ and the final marking probability $pa = pb/(1 - count * pb)$, where maxp and count are design parameters, respectively, denoting the maximum value for pb and the number of packets having arrived since last packet marking or dropping. If avg exceeds maxth, pa = 1, which means that the router marks or drops each packet that arrives. Through control over the average queue size prior to queue overflow, RED succeeds in preventing heavy network congestion and global synchronization as well as improving fairness. Notice that numerous variants of RED have been proposed to improve various performance of the original RED.

- Explicit Congestion Notification (ECN)

Most of current Internet routers employ traditional "drop-tail" queue management. In other words, the routers drop packets only when the queue overflows, which could lead to the undesirable global synchronization problem as well as heavy network congestion. Recently, active queue management (AQM) mechanisms have been proposed since they can detect congestion before the queue overflows at the routers and inform TCP senders of the congestion, thereby avoiding some of these problems caused by the "drop-tail" policy. In the absence of Explicit Congestion Notification (ECN), however, the only choice that is available to AQM for indicating congestion to end systems is to drop packets at the routers.

With ECN, AQM mechanisms have an alternative to allow routers to notify end systems of congestion in the network. ECN requires some changes to the header of both IP and TCP. In the IP header, an ECN field with two bits is used. By setting this field to specific bits, the router can send an indication of congestion to end systems. For TCP, two

new flags in the Reserve field of the TCP header are specified. By manipulating these two flags, the TCP sender and the TCP receiver can enable ECN via negotiation during connection setup; the receiver can inform the sender if it receives congestion indications from intermediate routers; and the sender can inform the receiver that it has invoked congestion control mechanisms

5.2.2 TCP Over Wireless Mesh Networks

TCP performance over wireless multi-hop mesh networks is shown to be poor [66] and many researchers have addressed this problem and proposed various TCP modifications specifically designed for wireless mesh networks, (refer to [48] and [58]). However, most of these TCPs have new designs and/or require help from intermediate nodes or collaboration among network nodes, which make them not compatible with current widely used TCP Newreno. In a well organized network, these special TCPs may be a possible solution. But in a mesh network temporarily established by several portable devices, such as laptops, tablets and smart phones, with WiFi-direct capability [24], it is not practical to implement two different TCPs, regular TCP and mesh specific TCP. Moreover, due to security concern and resource limitation, collaboration and help from intermediate nodes or RED [29] type of active queue management are not always expected. In this paper, our design goal is not a new TCP aiming particularly at wireless mesh networks but an enhancement to the current widely used TCPs with minimal modification in order to fit them in wireless mesh networks. We preserve the host-to-host and distributed TCP schemes without assuming the collaboration from intermediate nodes. The new TCP is compatible and inter-operable with current TCPs, while substantially improving its performance in wireless mesh networks.

As most earlier works have pointed out that the capacity of wireless mesh/multi-hop networks is limited and serious unfairness could happen due to underlying MAC and PHY asymmetry [34] [48], it is really a challenge for wireless mesh networks to become a backbone to support high volume large data transmission. Instead, most traffics in wireless mesh networks are short TCP sessions, e.g. html webpage, emails, interactive data and instant messages. These short sessions are delay sensitive that try to finish transmissions as soon as possible. Only a few long sessions exist that are throughput sensitive concerned more about maximizing the throughput. The purpose of proposed TCP then is to aim at reducing transmission delay for short TCP sessions with limited or no throughput losses for long sessions.

In this paper, we will first investigate the TCP performance over wireless mesh networks, by which we can see that RTT is mostly composed of queuing delay, not transmission or propagation delay as in Internet. More importantly, certain TCP session, by taking advantage of underlying MAC or PHY, can dominate channel bandwidth and buffer space; consequently, its throughput is almost saturated and RTT fluctuates with own congestion window. In this case, high correlation is engendered between congestion window and RTT for that particular TCP session. In view of whole system, the high correlation session hogs network resources and suppresses other sessions. This observation inspires the proposed TCP modification, exploiting correlation information between congestion window and RTT for TCP control. If the growth of congestion window of the session with high correlation is limited, excessive network resource occupied by high correlation session will be released so that other low correlation sessions can benefit at a minimal sacrifice to high correlation session. The Correlation-TCP is based on most widely used Newreno, with only an additional function that adjusts congestion window according to the calculated correlation information between congestion window and RTT . Extensive experiments

are conducted to compare Correlation-TCP with conventional loss based TCP Newreno [27], delay based TCP Vegas [9], and combination of loss and delay based TCP Compound(CTCP) [60]. The results demonstrate that Correlation-TCP can enhance the total system performance by reducing delays of short sessions and increasing throughputs of long sessions with low correlation, while yielding only limited throughput losses of long sessions with high correlation. Furthermore, Correlation-TCP is compatible and interoperable with Newreno and adaptive to different requirements by tuning its parameters.

The remainder of this paper is organized as follows. Section 5.3 reveals the correlation existing between congestion window and RTT for certain TCP sessions over wireless mesh network. We will bring the correlation into TCP model to show that it is a key feature TCP should consider to balance throughput and delay. Section 5.4 will present a new TCP called Correlation-TCP that explores explicitly the correlation information into TCP design based on TCP Newreno. Section 5.5 conducts extensive experiments to show the improved performance of Correlation-TCP and its compatibility with current Newreno. Section **??** concludes the paper and lists future works.

5.3 The Impact of Correlation on TCP Performance

In this section, we will first investigate TCP performance over wireless mesh networks and reveal the impact of correlation. Based on this, we revisit TCP performance in the correlation point of view.

5.3.1 TCP over Wireless Mesh Networks in view of Correlation

Before introducing the impact of correlation, we will first analyze the RTT of TCP session in wireless mesh networks. The RTT of one TCP session could be derived as follows:

$$RTT = T_{Tr} + T_P + \beta win + X \tag{5.1}$$

T_{Tr} and T_P indicate the transmission and propagation delay respectively; βwin represents the queuing delay from its own packets in flight; where β is the parameter that depends on underlying network bandwidth; win is the TCP congestion window; and X accounts for the queuing delay due to other TCP sessions sharing network resources. In Internet, transmission and propagation delay are major components of RTT and Bandwidth Delay product(BDP) is large [43]. Also, the existence of enormous number of flows makes each TCP session's packets have almost no influence on its own queuing delay; thus, βwin is negligible, leading to $RTT = T_{Tr} + T_P + X$. X is roughly treated as a fixed value in most loss based TCPs. In delay based TCP algorithms (or combination of delay and loss), although queuing delay variation is an indication of congestion, these algorithms do not differentiate queuing delays posted by its own congestion window and by other sessions.

In order to show that the behavior of RTT in wireless mesh networks is rapidly different from those of wired networks, we analyze RTT behavior using NS-2 simulator [53]. IEEE 802.11 [66] with 11Mbps PHY rate is used as underlying MAC/PHY protocol. The default values are used for transmission power with which the transmission range is 250 meters while channel sensing and interference ranges are 550 meters. PHY layer capture effect is enabled with 10dB capture threshold. Link loss rate is set as 0.005 per packet and packets can also be lost due to the MAC contention. MAC layer back-off and retransmission are enabled following the IEEE 802.11 standard [1]. RTS/CTS is disabled in our

simulation. TCP data packets size is 1500 bytes, TCP Ack packet size is 60 bytes, and the queue size in each node is 100 packets. Also we assume the advertised window is always larger than the congestion window. Finally, fixed routing is assumed to simplify the problem and simple FIFO scheme is used for queue management. The experiment time is set to 300 seconds.

We select four typical scenarios in wireless mesh networks as shown in Figure 5.2 based on earlier works in [58] [11] [34]. In Figure 5.2, (a) shows cross traffic in which different flows share both queue buffer and wireless medium; (b) is for parallel traffic in which different flows share only wireless medium; (c) is for stack topology which represents Flow In Middle(FIM) problem of wireless mesh networks [34]; and (d) represents typical hidden terminal problem. A TCP session traverses along flow2 with multiple TCP sessions running over flow1 as background traffics during the whole period of simulation. In stack topology, one TCP session also runs over flow3. The experiments reveal interesting and important behaviors of TCP over wireless mesh networks summarized below.

- RTT is mostly composed of queuing delay.

As IEEE 802.11 uses shared medium and interference range is larger than transmission range, the number of simultaneous packet transmissions will be no more than H/4 [16], where H is the number of hops of a flow. The distance between nodes is about hundreds of meters so that propagation delay is in micro second level. Bandwidth-delay-product (BDP) in wireless mesh networks is bounded by the number of hops and is far smaller than that of high speed Internet [16]. Most of the packets in flight will be in queues along end-to-end paths so that queuing delay becomes the major part of RTT. For the topology in Figure 5.2 (a), the average RTT of TCP session in flow2 is 479ms

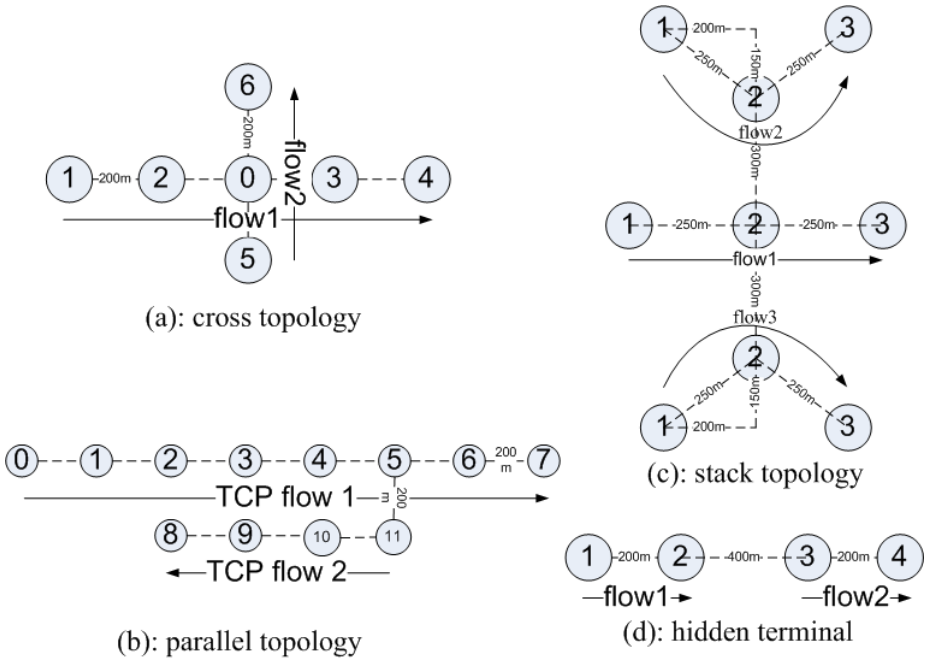


Figure 5.2: Four different traffic scenarios: one TCP session runs over flow2 with multiple background TCP sessions running along flow1. In stack topology, one TCP session also runs over flow3.

with one TCP session in flow1 as background flow. The delay components T_{Tr} and T_P for TCP session over flow2 can be estimated given the packet size, PHY transmission speed, propagation speed of electro-magnetic wave, and retransmission scheme of IEEE 802.11. T_{Tr} is approximately 2.2ms ($2 * (1500 + 60) * 8/11000000 = 2.2ms$ and T_P is $2.5us(4 * 200/300000000)$. If a packet is lost due to channel error or MAC contention, MAC layer will initialize the back-off and retransmit it until successful or drop the packet after the maximum number of retransmissions allowed (four times as default value). The back-off time is randomly chosen from 0 to back-off window size, which has the initial value 32 and will be doubled for every retransmission of that packet. The maximum back-off time for one packet per hop will be $(32 + 64 + 128 + 256) * 20us = 9.6ms$. As flow2 traverses two hops, the maximum delay for MAC layer back-off of a round trip is $4 * 9.6 = 38.4ms$. The combined total time for transmission, propagation, and MAC back-off is still far less than RTT, which means the major part of RTT comes from queuing delay, bearing the fact that large buffers are used to compensate for channel loss in wireless environment [69].

- Queue buffers can be occupied dominantly by certain TCP session whose RTT varies along with its own congestion window fluctuation; thus, high correlation between RTT and congestion window exists for that particular session.

The queuing delay can be induced by a TCP session's own packets in flight represented as βwin and packets from other TCP sessions represented as X in Eq. (5.1). As TCP congestion window controls the number of packets in flight, if the major part of queuing delay comes from its own packets in flight, its RTT will vary along with its own TCP congestion window fluctuation, pointing to a high correlation between them. In order to validate this assumption, we sample the RTT and congestion window of TCP session in

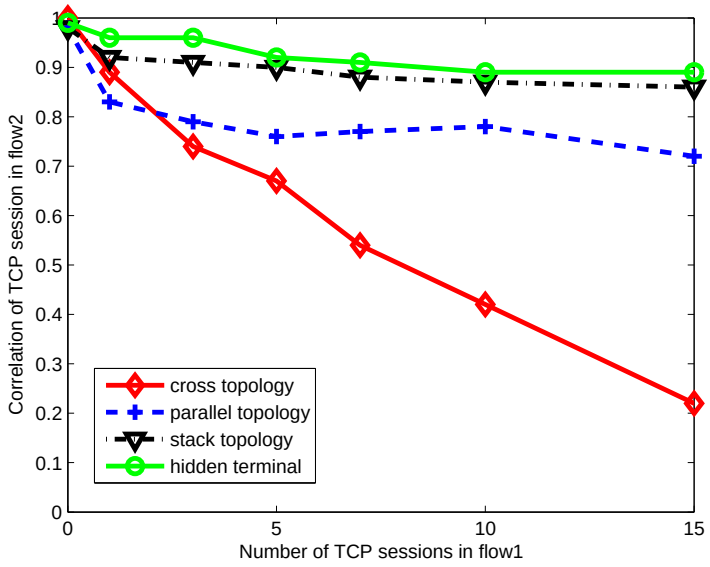


Figure 5.3: The correlation of TCP session over flow2: correlation is influenced by not only background traffics but also network topologies.

flow2 for every 0.5 seconds and measure the correlation between these two time series. The correlation ρ_i for TCP session i is calculated as the following Eq. 5.2:

$$\rho_i = \frac{E[(RTT_i^t - \overline{RTT_i})(win_i^t - \overline{win_i})]}{\sigma_{RTT}\sigma_{win}} \tag{5.2}$$

RTT_i^t and win_i^t are the sampled time series for RTT and congestion window at time t; ($\overline{RTT_i}$,$\overline{win_i}$) and (σ_{RTT},σ_{win}) are average and standard deviation of both values. Figure 5.3 shows the correlation of TCP session in flow2 along with different background traffics in flow1 over four different topologies. Figure 5.4 shows an example of TCP congestion window and RTT evolution of TCP session in flow2 in parallel topology with 10 TCP sessions in flow1.

Experiments illustrate clearly the existence of high correlation between congestion window and RTT of TCP in wireless mesh network, which indicates certain TCP session

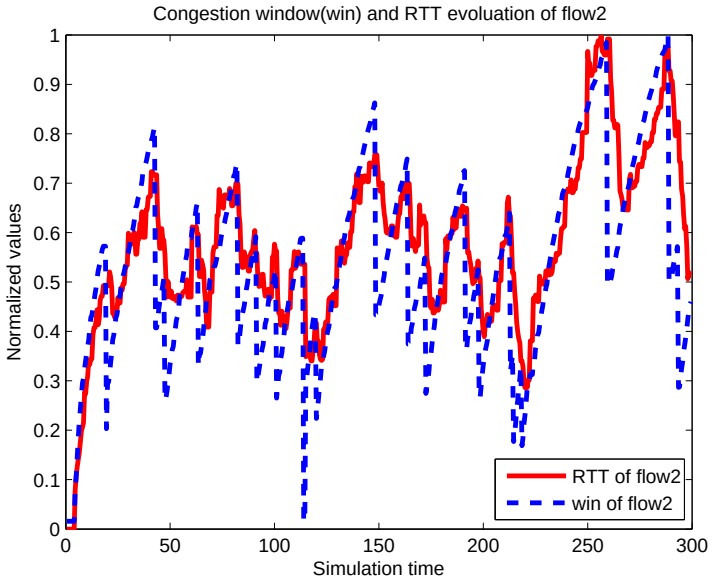


Figure 5.4: An example of TCP session in flow2 is to trace the evolution of congestion window and RTT in parallel topology with 10 TCP sessions in flow1. RTT varies along with congestion window fluctuation, resulting in and high correlation between them. Both values are normalized.

dominates the queue resource. This phenomenon rarely happens in Internet but can be common in wireless mesh networks. In Internet, buffer is shared among TCP sessions and mainly dependent on the condition of data traffic. The large number of sessions will ease individual's influence on its RTT, making RTT represents the statistical average of network condition [54]. In wireless mesh networks, however, traffic is far less than in internet, more importantly, resources shared among TCP sessions depends not only on traffic condition but also on wireless channel occupancy. A more general and extensive study has been done in [35] and later test-bed evaluation [11]. They show that contending flows in multi-hop IEEE 802.11 wireless networks compete with two fundamental asymmetries: (i) channel asymmetry, in which one flow has a stronger signal, potentially yielding physical layer capture, and (ii) topological asymmetry, in which one flow has channel information, potentially yielding an advantage in winning access to the channel

like in hidden terminal case. These asymmetries favor certain TCP session in accessing channel and increasing its congestion window more quickly, hence more queue occupation, while other TCP sessions continue suffering from packet losses and TCP time-outs so that their congestion windows remain small and cannot compete with high correlation session.

The experiments and analyses above demonstrate that queuing delay from a TCP session's own packets in flight is not negligible or even becomes a dominant factor of RTT; hence, high correlation between RTT and congestion window can prevalently exist for TCP over wireless mesh networks. This unique feature of wireless mesh networks is not considered in the eariler TCP models and designs. Next subsection will show that the correlation plays an important role regarding the tradeoff existing between throughput and RTT delay of TCP.

5.3.2 TCP Throughput and Delay Analysis using Correlation Information

The TCP model proposed in [54] has been widely used, which captures the essence of TCP congestion avoidance behavior and expresses the TCP throughput as a function of packet loss rate (p), round trip time (RTT), the incremental pace of TCP window size for every RTT time ($\frac{1}{b}$), and timeout value(T_0) as:

$$BW = \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\{1, 3\sqrt{\frac{3bp}{8}}\}p(1+32p^2)} \tag{5.3}$$

The RTT of TCP session is already given in Eq. (5.1). According to [54] the average window is derived as:

$$\overline{win} = \sqrt{\frac{8}{3bp}} \tag{5.4}$$

Incorporating the Eq. (5.4) into Eq. (5.3) and Eq. (5.1), we get:

$$BW = \frac{1}{\frac{4}{3}\beta + \frac{4(\overline{T_{Tr}}+\overline{T_P}+X)}{3\overline{win}} + T_0 \min\{1, \frac{3}{\overline{win}}\}p(1+32p^2)} \tag{5.5}$$

$$\overline{RTT} = \overline{T_{Tr}} + \overline{T_P} + \beta\overline{win} + X \tag{5.6}$$

Eq. (5.5) and (5.6) show an interesting balance between throughput and RTT delay: if $\beta\overline{win}$ is the major portion for RTT, a large average congestion window increases RTT delay significantly, but the throughput improvement is limited. That is, if the throughput is approaching upper limit, pumping more packets into the network will only incur excessive queuing delay. For a small average congestion window, the throughput may suffer a little but the RTT delay will be reduced significantly. Correlation provides a new degree of freedom for network congestion besides the current packet loss and RTT variation, The fundamental problem for current TCP lies in its lack of feedback mechanism to TCP sender of such network information. This observation has motivated us to design a new TCP protocol by incorporating the correlation information into congestion control. In the next section we will propose a correlation based TCP (Correlation-TCP) algorithm.

5.4 Correlation-TCP

5.4.1 Correlation-TCP algorithm

The design of Correlation-TCP is motivated by the analyses described above: certain TCP session with high correlation dominantly occupies the queue buffer space, leading to larger delays or smaller throughputs for all other sessions sharing network resource while its own throughput approaches the upper bound. Therefore, the key idea of the new design is to properly limit the congestion window of high correlation TCP session. By doing so, packets residing in queues can be considerably reduced and other sessions will have chance to improve their performances. In the meantime, constraining congestion window incurs limited throughput loss for saturated sessions with high correlation. Correlation-TCP achieves this goal by explicitly exploring the correlation information as a signal for TCP congestion control.

The proposed Correlation-TCP is based on Newreno with an additional function used to calculate correlation using sliding window scheme. Correlation-TCP will record RTT and congestion window(win) values as a pair at TCP sender side for every T seconds once TCP enters congestion avoidance stage, where T is a predetermined parameter. As the samples fill the sliding window with sample length K, it will calculate the correlation following the Eq. (5.2). From then on, for every T seconds, it will move forward the sliding window one step to insert a new pair and remove the oldest pair of congestion window and RTT records. If correlation values surge above certain threshold, it will cut its congestion window based on its calculated correlation value, acting early in the incipient queue buildup stage. Other Newreno actions of Correlation-TCP, e.g. Congestion window cut for three Duplicate ACK, Time-out, slow start, are kept the same.

Algorithm 1 provides the pseudo code of Correlation-TCP. *Win* represents congestion

window and *ssthresh* is the threshold between slow start and congestion avoidance. If *win* is less than or equal to *ssthresh*, TCP is in slow start; otherwise, TCP is performing congestion avoidance. For the high correlation session, once its correlation value exceeds ρ_{Threh}, it will reduce its congestion window (sending rate) based on the calculated correlation value: the higher the correlation, the more congestion window TCP will reduce. In the meantime, sessions will not be disturbed as long as they have lower correlations smaller than ρ_{Threh} or (and) they can finish transmission before sample length reaches K, benefitting both long sessions with low correlation and short sessions. α is the parameter used to adjust sensitive level to correlation whose value is between 0 and 1. We can adjust the parameters ρ_{Threh} and α to meet different balance requirements between throughput and RTT, for which a detailed study will be done later in this paper.

Compared with other TCP algorithms designed for wireless mesh networks, Correlation-TCP has some distinct advantages.

- Correlation-TCP explores the current available information of RTT and congestion window and needs a slight modification at end devices without modifying any intermediate node. It is a local and distributed algorithm that can be easily deployed and the overhead is minimal.

- Correlation-TCP is compatible with current TCP. If the correlation is low, the proposed Correlation-TCP performs the same as current TCP Newreno. It can be used both in wireless mesh networks with enhanced performance and Internet with same performance as Newreno (illustrated in next section), making it a perfect candidate for those devices that operate in both environments.

- Correlation-TCP can adjust its performance by tuning the parameters ρ_{Threh} and α. So it is flexible to meet different upper layer requirements.

```
Open linked list COR_TCP after TCP session enters to congestion avoidance stage;
while Every T seconds do
    Sample = (Win, RTT);
    if First Sample then
        Head = Sample;
    else
        Insert Sample at the tail of COR_TCP;
    end
    if Length(COR_TCP)< K then
        Length(COR_TCP)++;
        Continue;
    else
        Newhead = Head− > next;
        Delete(Head);
        Head = Newhead;
        Calculate correlation ρ_TCP based on Eq. 5.2;
        if ρ_TCP > ρ_Threh then
            win = win(1 − ρ_TCP × α);
            if win < ssthresh then
                ssthresh = win;
            end
        else
            Continue;
        end
    end
end
```

Algorithm 1: Correlation-TCP algorithm

Extensive experiments will be performed in the following to evaluate the performance of Correlation-TCP and demonstrate these distinct advantageous features.

5.5 Performance Evaluation

We conduct extensive experiments in order to demonstrate the unique advantages of Correlation-TCP. First we will show the performance of Correlation-TCP compared with other TCPs. Then we will see the compatibility of Correlation-TCP with Newreno in wireless mesh networks and illustrate how Correlation-TCP can adapt different requirements by tuning its parameters. Finally we apply Correlation-TCP to Internet and confirm it has same performance as Newreno.

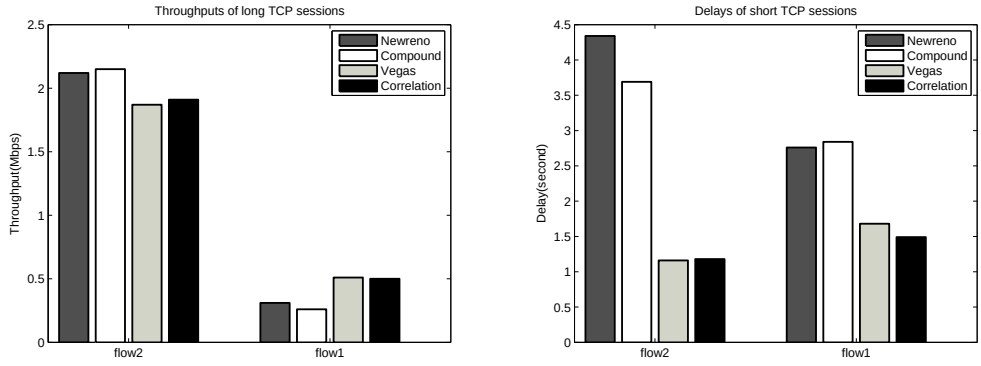
5.5.1 Comparison with Other TCPs

We begin our experiments by evaluating the performance of Correlation-TCP compared with three representative TCP schemes: loss based TCP Newreno, delay based TCP Vegas, and loss and delay combination based TCP Compound(C-TCP). Two different types of TCP traffics are considered here: one is throughput sensitive long TCP sessions running throughout the whole simulation period and the other is delay sensitive short TCP sessions that transmit 1M bytes data. One long session runs along flow2 with 5 long sessions running along with flow1 as background traffics. Meanwhile, 50 short sessions will run over both flow1 and flow2 and the start times for short sessions follow Poisson distribution with average interval of 3 seconds. ρ_{Threh} is set at 0.7 and α is set at 0.6 for Correlation-TCP. Sampling interval T is set to 0.5 sec and K is 20, meaning only those sessions lasting longer than 10 seconds will trigger Correlation-TCP actions. The other MAC and PHY settings are the same as before.

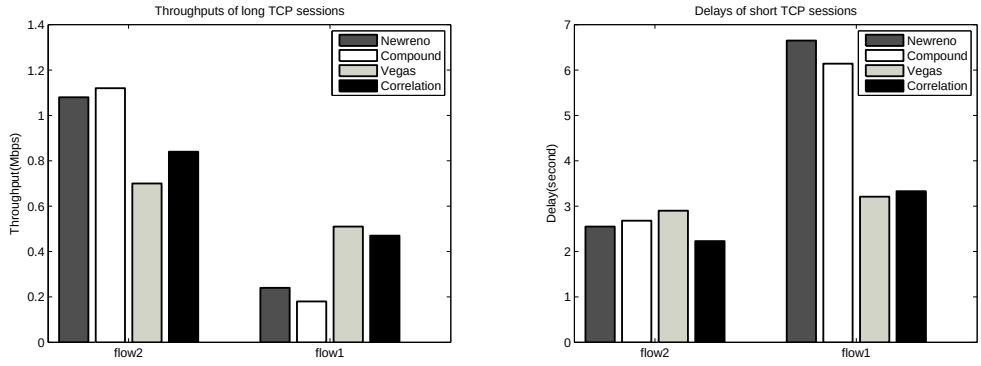
Figure 5.5 shows the comparison over four topologies in Figure 5.2. We consider aggregate throughputs for long sessions over flow1, while delay represents average completion time of transmissions among 50 short sessions.

Regarding cross and parallel topologies in Figure 5.5 (a) and (b), Newreno and Compound post much larger delays for short sessions compared to Vegas and Correlation-TCP. Newreno and Compound both build large queue delays to maximize throughputs of almost saturated long sessions over flow2 (and flow3 in stack topology) and the finishing times of short sessions are longer and throughputs of long sessions in flow1 are largely suppressed. In contrast, both Vegas and Correlation-TCP reduce delays of short sessions significantly, while throughputs of sessions in flow1 increase with limited throughput loss of sessions in flow2. Vegas sessions try to restrain congestion windows once RTT increase is detected; thus, delays of short sessions do not benefit that much as limited congestion windows also prevent them from completing quickly, in the meantime throughput of long sessions could suffer owing to small congestion window such as in parallel topology. Correlation-TCP dose not have this problem as it can target and select the particular long session that occupies most queue resources, while other short sessions act same as Newreno because they can finish before sample lengths reaching K and (or) their calculated correlation value is smaller than ρ_{Threh}. These undisturbed sessions can improve their performance by making use of network resource conceded by high correlation and dominant sessions in flow2. Therefore, we see less delay for short sessions or more throughputs for long sessions in flow1. Meanwhile, throughput reduction of high correlation session is limited as Correlation-TCP will behave same as Newreno. It is because once high correlation session yield its dominant status and correlation value is lower than ρ_{Threh}, high correlation session still has much larger throughput in this case.

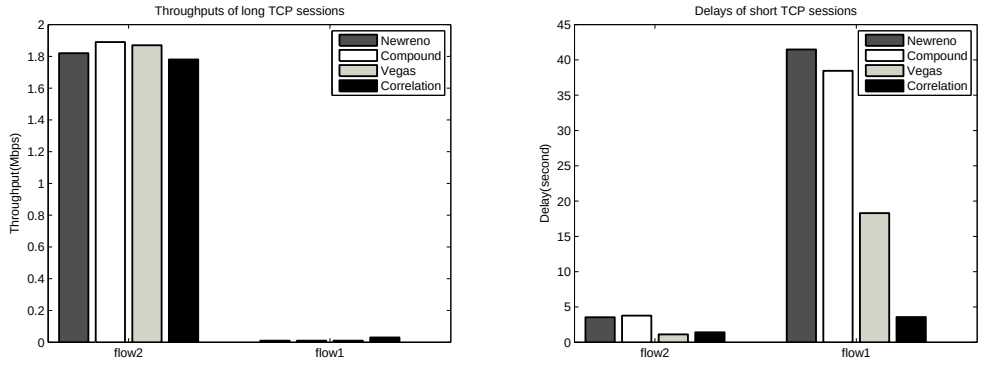
The stack and hidden terminal topologies in Figure 5.5 (c) and(d) represent two well



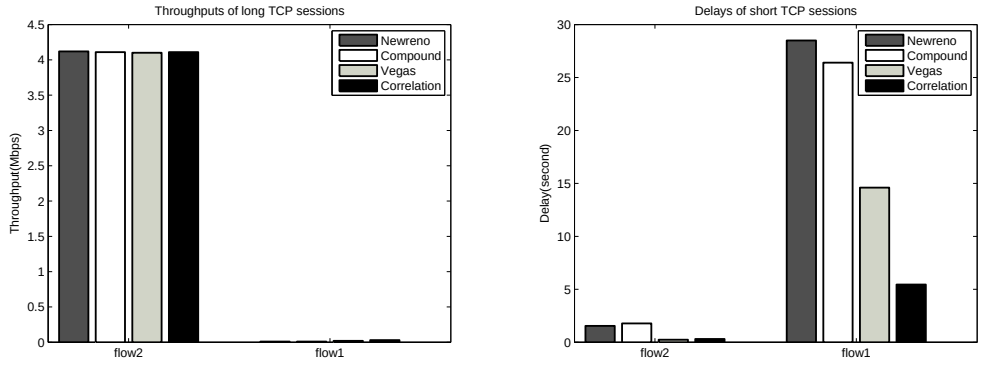
(a) cross topology



(b) parallel topology



(c) stack topology



(d) hidden terminal topology

known MAC asymmetric problems: Flow In Middle(FIM) [34] and hidden terminal problems [66]. In both topologies, TCP sessions in flow2 (also flow3 in stack topology) have a high chance to grasp the channel and continuously transmit; on the other hand, TCP packets of sessions in flow1 always face channel busy (in stack topology) or transmission failure (in hidden terminal topology) in MAC layer. TCP layer can not sense the MAC asymmetric problem without the help and coordination from MAC layer. Newreno and Compound will saturate flow2 (and flow3 in stack topology) and fill the queues in which case almost no transmission in flow1 can squeeze in. As a result, throughput of long session is almost shut down and huge delays ensue for short sessions in flow1. Correlation-TCP can increase throughputs of suffered sessions for 2 to 3 times compared with Newreno and Compound by limiting congestion windows of dominating sessions in flow1. Therefore, delays of short sessions benefit and are reduced dramatically using Correlation-TCP. Vegas cannot achieve such large delay reduction and throughput increase. Sessions in flow1 using Vegas also sense delay increasing and restrain their congestion windows; thus, long sessions have less throughputs and short sessions need more time to finish transmission. Moreover, if a short Vegas session in flow1 starts with a small RTT (mostly because dominant sessions just cut their congestion windows and queuing delays are reduced), then it will sense a large RTT increase and cut its congestion window into a very small level that postpones its finishing time. Moreover, we find short sessions using Vegas have far more TCP time-out than those using Correlation-TCP (about 6-8 times more) as the RTT estimations of short sessions fluctuate with queuing delays in long sessions. These spurious TCP time-out will force TCP to restart congestion window and further postpone the data transmission. The observations show that performance of Vegas highly depends on RTT estimation and is quite unstable in stack and hidden terminal topologies. On the other hand, correlation estimation is quite robust in reflecting

Table 5.1: Performance in a 100 nodes topology.

Average value	delay(sec)	throughput(Mbps)
Newreno	0.769	0.521
Compound	0.764	0.523
Vegas	0.523	0.423
Correlation	0.311	0.476

network status regardless of topologies. In that sense, we believe correlation is a much stable measurement than RTT variation in wireless mesh networks.

As the queuing delay is the major part of RTT, we will see the behaviors of different TCPs by varying node buffer sizes. We perform the experiments by changing node buffer size: 100, 60 and 30 packets. For Vegas and Correlation-TCP, the throughputs of long sessions and the delays of short sessions are almost the same, which demonstrates their capability to keep queue sizes small regardless of node buffer size. For Newreno and Compound, throughputs of long sessions decrease and delays of short sessions increase when the buffer size is smaller. Trace shows this is due to more queue overflow and packet drop causing further cut for congestion window. [23] [69] indicate the same behavior and they both claim that larger buffer size is better. Our later experiments will use 100 packet buffer sizes.

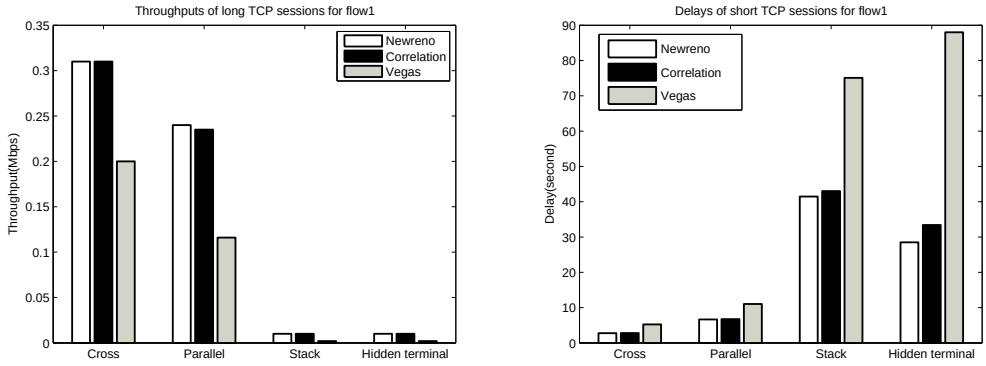
Next we expand the experiment to a large network scenario where 100 nodes are placed in a grid topology (10 by 10) with 200m apart each other and other settings remain the same as before. We apply 60 long sessions and 200 short sessions over 40 pairs of randomly chosen source and destination nodes. Average throughputs of long sessions and delays of short sessions are measured in Table 5.1, which illustrates the same trend as before that Correlation-TCP has the lowest delay for short sessions with limited throughput loss for long sessions.

5.5.2 Co-existence and Fairness Issues

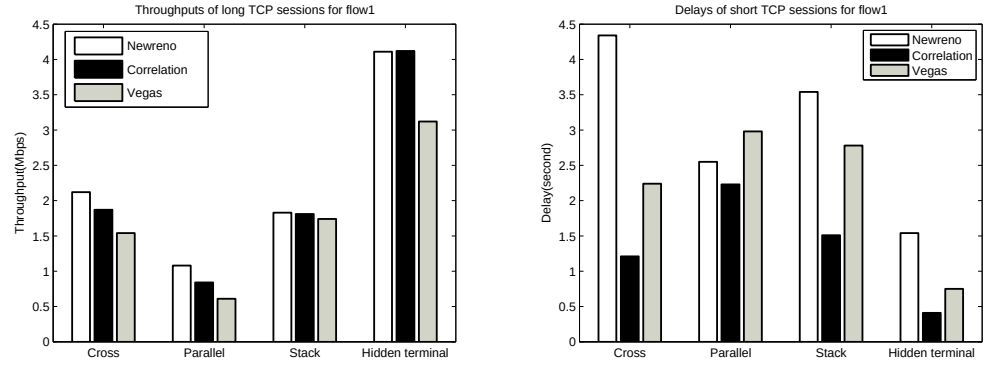
As Correlation-TCP will limit congestion windows of high correlation sessions, whether it will have fair share when co-exists with current Newreno becomes a concern. In order to show teh co-existence between Correlation-TCP and Newreno, we compare Correlation-TCP with Vegas, which is known for its inability to get fair share against Newreno. We conduct two groups of experiments: First, sessions in flow1(and flow3 in stack topology) use Correlation-TCP, Vegas and Newreno respectively while sessions in flow2 use Newreno. Then reversely we let sessions in flow1(and flow3 in stack topology) use Newreno while sessions in flow2 use Correlation-TCP, Vegas and Newreno respectively. Figure 5.6 shows the comparison results.

In Figure 5.6 (a), Correlation-TCP sessions perform almost same as Newreno because their correlation values are below ρ_{Threh} and/or they complete transmission before sample length reaching K. As Vegas cannot get fair share when competing with aggressive Newreno, the throughputs of long sessions decrease and delay of short sessions increase significantly. In Figure 5.6 (b), throughputs of long sessions using Correlation-TCP and Vegas are smaller than those of using Newreno. However, Correlation-TCP experiences much less throughput loss than Vegas as it goes back to Newreno once high correlation sessions have been suppressed and their correlation values become less than ρ_{Threh}. Also Correlation TCP sessions have the lowest delays. Based on these observations, we can see that Correlation-TCP dose not have the same fairness problem that Vegas has because only high correlation long sessions would be affected, and its behavior is more conservative and stable compared with Vegas.

To further demonstrate the observation above, we repeat experiments in 10*10 grid topology as before, in which part of sessions use Correlation-TCP and others use Newreno.



(a) Performance of sessions in flow1



(b) Performance of sessions in flow2

Figure 5.6: Compatibility and inter-operability of Correlation-TCP compared with Vegas. In (a) all sessions in flow2 use Newreno; In (b) all sessions in flow1 use Newreno. Correlation-TCP dose not have the same fairness problem Vegas has.

Table 5.2: 50 TCP sessions in a 100 nodes topology via different percentage of TCP sessions with Correlation-TCP

Percentage of Correlation-TCP	Average short session delay(sec)	Average long session throughput(Mbp
0%	0.817	0.521
20%	0.784	0.505
40%	0.643	0.498
60%	0.516	0.481
80%	0.432	0.473
100%	0.323	0.467

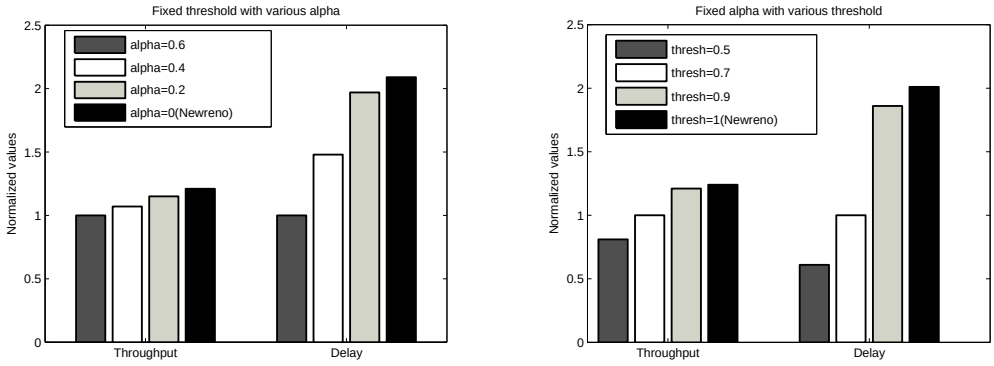
The test was repeated 10 times and averages are taken. The results are listed in Table 5.2. The more sessions use Correlation-TCP, the better the performance achieved regarding delay reduction at the expense of slight throughput loss. Thus, we observe that Correlation-TCP can co-exist with Newreno more fairly.

In term of fairness, we see that Correlation-TCP actually improves the fairness as it reduces the unfair domination of high correlation sessions. Regarding the throughput fairness in view of both Jain's fairness index [18] and proportional fairness term [44], Correlation-TCP shows much better fairness than loss based TCPs and is comparable with delay based TCPs. Another issue is that Correlation-TCP tends to favor short sessions. We will target this concern by three different view points. First, the enhancement of short sessions also comes from the back off of unfairly dominant high correlation sessions. If all sessions have low correlation values, Correlation-TCP actually becomes Newreno and there is no more special favor for short sessions. Secondly, we believe wireless mesh networks are suitable for small data transmissions rather than providing backhual capability for large data, so it is reasonable to favor short sessions. Actually, even in internet community, similar suggestion has been proposed to favor short TCP flows[6]. Finally, Correlation-TCP provides adaptive capability by using different parameters, allowing upper layer applications to select the best choice.

5.5.3 Impacts of Parameters

This subsection will investigate the impacts of the parameters ρ_{Threh} and α on the performance of Correlation-TCP. We implement experiments on grid topology and set $\rho_{Threh} = 0.7$ and $\alpha = 0.6$ as a benchmark. We measure the normalized throughputs and delays compared to the benchmark case with different parameter values. We first fix ρ_{Threh} at 0.7 and vary α, A larger α means larger congestion window reduction. Next, we will fix α at 0.6 and test the impact of different ρ_{Threh} values using the same test scenarios as before. Smaller ρ_{Threh} leads larger reduction of congestion window, and vice versa. Figure 5.7 shows the normalized experimental results. larger α and smaller ρ_{Threh} cut congestion window more heavily and frequently, leading less queue sizes in nodes thus delays of short sessions can be further reduced. The average throughputs of long sessions don't change much as high correlation sessions will suffer but low correlation ones can gain. Meanwhile, smaller α and larger ρ_{Threh} will limit the congestion window cut by Correlation-TCP, making it more like Newreno. When $\alpha = 0$ or $\rho_{Threh} = 1$, Correlation-TCP acts same as Newreno according to Algorithm 1.

Above experiments illustrate that the performance of Correlation-TCP can vary with different parameter values. By tuning ρ_{Threh} and α properly, we can balance the performance of TCP between throughput and delay. This flexibility of Correlation-TCP provides a powerful tool for upper layer to choose the most suitable parameter pairs for different application requirements. Yet, theoretical works are needed for this part, for example, on how to optimize the parameters to achieve a certain throughput or delay bound.



(a) Fixed ρ_{Threh} with various α

(b) Fixed α with various ρ_{Threh}

Figure 5.7: Impacts of Parameters: (a) shows different α with fixed $\rho_{Threh} = 0.7$; (b) shows different ρ_{Threh} with fixed $\alpha = 0.6$. When $\alpha = 0$ or $\rho_{Threh} = 1$, Correlation-TCP acts same as Newreno according to Algorithm 1.

5.5.4 Correlation-TCP in Internet

The above analyses and experiments are all based on wireless mesh networks. As devices equipped with Correlation-TCP is also expected to work in Internet, the performance of Correlation-TCP in general network is of concern. In this subsection we will show such a scenario in Figure 5.8. All links have 10Mbps bandwidth and 10ms delay except link 4-6 and link 5-7. The queue length is 50 packets for all nodes. Long sessions and short sessions coexist with same setting as before. Flow1 has only one long session from node0 to node7; flow2 has 5 long sessions from node 1 to node 6 and 50 short sessions; flow3 is a long session from node 8 to node 13 and 20 short sessions; and flow4 has a long session from node 9 to node 12 and 30 short sessions. Finally there is a UDP background traffics from node 10 to node 11. Fixed routing is assumed. Same as before, throughputs of long sessions and delay of short sessions are measured. We apply the Correlation-TCP and newreno respectively to all sessions and compare their performance. $\rho_{Threh} = 0.7$ and $\alpha = 0.5$ are used for Correlation-TCP. Figure 5.9 shows the throughputs

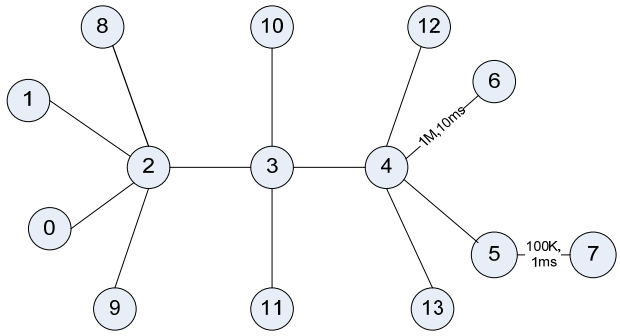


Figure 5.8: A scenario for high correlation in general cases.

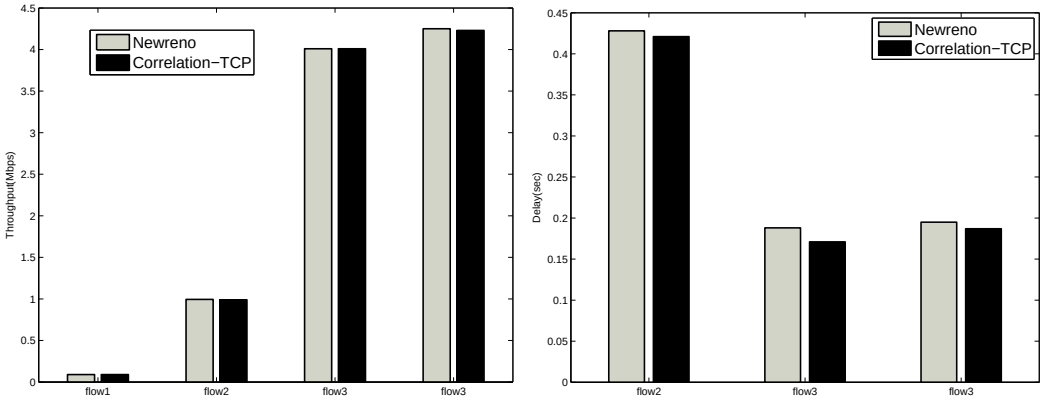


Figure 5.9: Performance of Correlation-TCP compared with Newreno for topology in Figure 5.8. No short sessions over flow1 thus no delay measurement.

of long sessions and delay of short sessions. As the correlations of TCP sessions are relative low except for that over flow1, Correlation-TCP acts the same as Newreno and throughputs of long sessions are the same for both TCP algorithms. High correlation occurs for TCP session over flow1 due to a combined effect of slow link and a buffer exclusively devoted to it . In contrast, although the link between nodes 4 and 6 is the performance bottleneck for flow2, the buffer is shared among 5 TCP sessions so that no TCP session shows high correlation. In this case, Correlation-TCP reduces its queues but its throughput still keeps the same as link 5-7 has been saturated and its throughput is bounded by link bandwidth (TCP throughput is 0.099Mbps while link bandwidth is

0.1Mbps). Also the session with suppressed congestion window over flow1 will reduce the injection rate of packets into network, benefiting the delays of short sessions with a slight decrease. However, due to the large number of sessions, the delay reduction is very limited (about 2-4 percent). Therefore, we claim Correlation-TCP has the same performance as Newreno in this scenario, which makes it a desirable candidate for devices that operate in both Internet and wireless mesh networks.

5.6 Summary

In this paper, we first investigate the throughput and delay of TCP over wireless mesh networks. The analysis reveals that the correlation existing between congestion window and RTT is the key that influences the TCP performance. Based on this observation, we propose a modified TCP based on Newreno called Correlation-TCP to explore the correlation information into TCP design. By doing so, Correlation-TCP can effectively reduce the delay of short TCP sessions and increase the throughputs of long sessions with low correlation, with limited throughput losses of long TCP sessions with high correlation. Also it is compatible with Newreno in Internet and can adjust its performance by tuning different parameters.

There are several future works to be done to improve the performance of Correlation-TCP. First, the correlation calculation is an overhead for end devices; so, fast and memory efficient correlation calculation is needed to improve the Correlation-TCP algorithm. Also we find that sliding window size in Correlation-TCP needs more works. Current Correlation-TCP just provides a framework for throughput and RTT balance and we hope to apply more precise analysis for instance on how to use Correlation-TCP to meet certain delay guarantee requirement. Correlation-TCP optimization with different parameters is

also a potential next step. Another future work would be applying Correlation-TCP to other TCP algorithms such as Compound used in Windows [60] and CUBIC [37] used in Liunx to see their performances.

Chapter 6

Conclusion and Future Works

In this thesis, we present a comprehensive research and development of wireless mesh network with the case study of IEEE 802.15.5 low rate part and the topics of this thesis cover both networking and transportation layer.

We first present IEEE 802.15.5 low rate part by introducing all of mandatory functions that comprise tree formation, allocation of address blocks, distributed local link state information, and mesh routing. The network formed from these basic building blocks exhibits many desirable properties such as scalability, simplicity, and reliability. In particular, the k-hop local link state information and tree structure with address block scheme facilitate the mesh routing by providing a guideline to extract the next hop toward a destination even when the local link state cannot give any information about the destination. Also the prototype implementation in test-bed is demonstrated to demonstrate the advantage of it.

We then further explore the fundamental capability of IEEE 8021.5.5 by providing a comprehensive theoretical analysis of IEEE 802.15.5. As a low power wireless mesh network standard, it should be simple enough to fit in limited memory and energy to ensure good scalability. Also 802.15.5 needs to provide certain fault tolerant capability.

Through the complexity analyses and accompanying simulation studies of network layer performance, 802.15.5 is shown to be the preferred choice over ZigBee Pro and the tree-based scheme. Importantly, it does not suffer from any particular shortage so that it has the potential to be applied to vast diversified applications.

The third part of works target the mobility scenario in WPAN mesh and wireless sensor networks,

Finally, we change the focus on transportation layer of wireless mesh networks and target the shortage of most widely used TCP in wireless mesh network and propose a new Correlation-TCP based on TCP Newreno. We first investigate the throughput and delay of TCP over wireless mesh networks. The analysis reveals that the correlation existing between congestion window and RTT is the key that influences the TCP performance. Based on this observation, we propose a modified TCP based on Newreno called Correlation-TCP to explore the correlation information into TCP design. By doing so, Correlation-TCP can effectively reduce the delay of short TCP sessions and increase the throughputs of long sessions with low correlation, with limited throughput losses of long TCP sessions with high correlation. Also it is compatible with Newreno in Internet and can adjust its performance by tuning different parameters.

As still in its early stage, wireless mesh network has many new topics that can be further explored and investigated. We present several future research areas related to the thesis.

- Fundamental analysis of network layer capability: Although networking layer has been investigated a lot, we believe there is huge room for fundamental analysis of network layer capability. Wireless mesh network has totally networking architecture so that how to balance available resource to establish the most reliable and re-

source network is always a big challenge. Besides the algorithms and development efforts put into this field, some fundamental research is still limit to guide the major design trend and provide insight understanding. In this thesis, we present some analysis of IEEE 802.15.5, the future works could target more general networking scheme£for example, comparing the reactive networking scheme with proactive scheme, and state vs Stateless communication strategies. etc. The goal of these fundamental research is to abstract the useful metrics from complicated diversified networking realization and provide guideline for network design bound.

- Cross layer design with the consideration of application requirements: Layered stack has been demonstrated to be inefficient in wireless mesh network. So how to explore the cross layer information into design is a big chance. Networking architecture and protocol could leverage the cross layer information to best design. In IEEE 802.15.5, we use link quality as a metric for network establishment, we can further consider the application layer traffic and other requirements to optimize network layer design.

- Information and networking co-design: One major application of wireless mesh network is in the wireless sensor network field. As purpose of wireless sensor network is to get information from environment, information oriented networking design could be another topic. How to abstract, interpret, analysis of information from sensor network and how to design the suitable architecture to leverage the information processing capability will be future research.

- IEEE 802.15.5 future supports: Besides IEEE 802.15.4, there are several PHY/MAC enhancements have been proposed: IEEE 802.15.4e targets the time slotted based solution; IEEE 802.15.4f targets RFID application; IEEE 802.15.4g targets smart

utility application, etc. The next step of IEEE 802.15.5 could be the common layer built based on these PHY/MAC standard extensions.

bibliography

[1] IEEE 802.11. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

[2] IEEE 802.15.4. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs).

[3] IEEE 802.15.5. Mesh Topology Capability in Wireless Personal Area Networks (WPANs).

[4] K. Akkaya and M. Younis. Energy-aware routing to a mobile gateway in wireless sensor networks. In *Globecom Wireless Ad Hoc and Sensor Networks Workshop*. IEEE, 2004.

[5] ZigBee Alliance. http://www.zigbee.org/.

[6] Konstantin Avrachenkov, Urtzi Ayesta, Patrick Brown, and Eeva Nyberg. Differentiation between short and long tcp flows: Predictability of the response time. In *INFOCOM*, 2004.

[7] Andrea Baiocchi and Francesco Vacirca. Tcp fluid modeling with a variable capacity bottleneck link. In *INFOCOM*, pages 1046–1054. IEEE, 2007.

[8] Ashwin R. Bharambe, John R. Douceur, Jacob R. Lorch, Thomas Moscibroda, Jeffrey Pang, Srinivasan Seshan, and Xinyu Zhuang. Donnybrook: enabling large-scale, high-speed, peer-to-peer games. In *SIGCOMM*, pages 389–400. ACM, 2008.

[9] Lawrence S. Brakmo and Larry L. Peterson. Tcp vegas: End to end congestion avoidance on a global internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–1480, 1995.

[10] A. T. Campbell C.-Y. Wan and L. Krishnamurthy. Pump-slowly, fetch-quickly (psfq): A reliable transport protocol for sensor networks. 23(4):862–872, 2005.

[11] Joseph Camp, Ehsan Aryafar, and Edward Knightly. Coupled 802.11 flows in urban channels: Model and experimental evaluation. In *INFOCOM*, 2010.

[12] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, 2(5):483–502, 2002.

[13] Ian D. Chakeres and Luke Klien-Berndt. AODVjr:, AODV Simplified. *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, pages 100–101, July 2002.

[14] Mun Choon Chan and Ramachandran Ramjee. Tcp/ip performance over 3g wireless links with rate and delay variation. In *MOBICOM*, pages 71–82. ACM, 2002.

[15] Wu chang Feng, Dilip D. Kandlur, Debanjan Saha, and Kang G. Shin. Stochastic fair blue: A queue management algorithm for enforcing fairness. In *INFOCOM*, pages 1520–1529, 2001.

[16] Kai Chen, Yuan Xue, Samarth H. Shah, and Klara Nahrstedt. Understanding bandwidth-delay product in mobile ad hoc networks. 27(10):923–934, 2004.

[17] CHIPCON. 2.4GHz IEEE802.15.4/ZigBee-ready RF Transceiver datasheet (rev1.2).

[18] Dah-Ming Chiu and Raj Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks*, 17:1–14, 1989.

[19] Seong-Eun Chu, Dae-Wook Kang, and Jae-Nam Kim. Routing and data dissemination over large-scale wireless sensor networks with multiple sources and mobile sinks. In *Advanced Language Processing and Web Information Technology*, pages 444–449. IEEE Computer Society, 2008.

[20] John C.-I. Chuang and Marvin A. Sirbu. Pricing multicast communication: A cost-based approach. In *Telecommunication Systems*, pages 281–297, 1998.

[21] T. Clausen and P. Jacquet. Optimized link state routing protocol (OLSR). *IETF RFC 3626*, October 2003.

[22] CC2420 datasheet. http://www.chipcon.com.

[23] Amogh Dhamdhere and Constantine Dovrolis. Open issues in router buffer sizing. *Computer Communication Review*, 36(1):87–92, 2006.

[24] Wi-Fi Direct. `http://www.wi-fi.org/Wi-Fi_Direct`.

[25] Kevin Fall and Sally Floyd. Simulation-based comparisons of tahoe, reno, and sack tcp. *ACM Computer Communication Review*, 28:5–21, 1996.

[26] S. Floyd, M. Handley, and E. Kohler. Problem Statement for the Datagram Congestion Control Protocol (DCCP), 2006.

[27] S. Floyd, T. Henderson, and A. Gurtov. The newreno modification to tcp's fast recovery algorithm. RFC 3782 (Proposed Standard), 2004.

[28] Sally Floyd. Tcp and explicit congestion notification. *ACM Computer Communication Review*, 24:10–23, 1994.

[29] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1:397–413, 1993.

[30] Bryan Ford. Structured streams: a new transport abstraction. In *SIGCOMM*, pages 361–372. ACM, 2007.

[31] HART Communication Foundation. http://www.hartcomm.org/.

[32] Cheng Peng Fu and Soung C. Liew. Tcp veno: Tcp enhancement for transmission over wireless access networks. *IEEE Journal on Selected Areas in Communications*, 21:216–228, 2003.

[33] J. Hui G. Montenegro, N. Kushalnagar and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. *IETF RFC 4944*, September 2007.

[34] Michele Garetto, Theodoros Salonidis, and Edward W. Knightly. Modeling per-flow throughput and capturing starvation in csma multi-hop wireless networks. In *INFOCOM*. IEEE, 2006.

[35] Michele Garetto, Jingpu Shi, and Edward W. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *MOBICOM*, pages 200–214. ACM, 2005.

[36] Ashvin Goel and Charles Krasic. Low-latency adaptive streaming over tcp. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 4(3):1–20, 2008.

[37] Sangtae Ha, Injong Rhee, and Lisong Xu. Cubic: a new tcp-friendly high-speed tcp variant. *Operating Systems Review*, 42(5):64–74, 2008.

[38] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448 (Proposed Standard), 2003.

[39] L. Hester, Y. Huang, O. Andric, A. Allen, and P.Chen. neurfon netform: A self-organizing wireless sensor network. In *Proceedings of the 11th IEEE ICCCN Conference*, pages 364–369, Miami,FL,USA, 2002.

[40] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. In *In Architectural Support for Programming Languages and Operating Systems(ASPLOS)*, pages 93–104, Cambridge, MA, USA, 2000.

[41] J. Hui and P. Thubert. Compression Format for IPv6 Datagrams in 6LoWPAN Networks. *IETF Internet draft,draft-ietf-6lowpan-hc-06*, october 2009.

[42] J. Hill J. Polastre and D. Culler. Versatile low power media access for wireless sensor networks. In *2th ACM Conference on Embedded Networked Sensor Systems(SenSys 2004)*, pages 95–107, Baltimore, MD, USA, 2004.

[43] V. Jacobson and R. Braden. Tcp extensions for long-delay paths. RFC 1072 (Proposed Standard), 1988.

[44] Frank Kelly, Aman Maulloo, and David Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[45] Hyung Seok Kim, Tarek F. Abdelzaher, and Wook Hyun Kwon. Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In *SenSys*, pages 193–204. ACM, 2003.

[46] E. Kohler, M. Handley, and S. Floyd. Datagram Congestion Control Protocol (DCCP), 2006.

[47] Myung J. Lee, Rui Zhang, Jianliang Zheng, Gahng-Seop Ahn, Chunhui Zhu, Tae Rim Park, Sung Rae Cho, Chang Sub Shin, and Jun Sun Ryu. **IEEE** 802.15.5 wpan mesh standard-low rate part: Meshing the wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 28(7):973–983, 2010.

[48] Christian Lochert, B. Scheuermann, and Martin Mauve. A survey on congestion control for mobile ad hoc networks. *Wireless Communications and Mobile Computing*, 7(5):655–676, 2007.

[49] E. Anderson M. Buettner, G. V. Yee and R. Han. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *4th ACM Conference on Embedded Networked Sensor Systems(SenSys 2006)*, pages 307–320, Boulder, CO, USA, 2006.

[50] Saverio Mascolo, Claudio Casetti, Mario Gerla, M. Y. Sanadidi, Ren Wang, and Politecnico Di Torino. Tcp westwood: Bandwidth estimation for enhanced transport over wireless links. In *MOBICOM*, pages 287–297, 2001.

[51] Crossbow Micaz motes. http://www.xbow.com.

[52] Preethi Natarajan, Janardhan R. Iyengar, Paul D. Amer, All Stewart, and Cisco Systems. Sctp: An innovative transport layer protocol for the web. In *15th International conference on World Wide Web*, pages 23–26, 2006.

[53] NS-2. Network Simulator. `http://www.isi.edu/nsnam/ns/`.

[54] Jitendra Padhye, Victor Firoiu, Donald F. Towsley, and James F. Kurose. Modeling tcp reno performance: A simple model and its empirical validation. *IEEE/ACM Transactions on Networking*, 8:133–145, 2000.

[55] Charles Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *Proceedings of the ACM SIGCOMM*, pages 234–244, London, UK, 1994.

[56] Charles E. Perkins, Elizabeth M. Belding-Royer, and Ian D. Chakeres. Ad hoc On-Demand Distance Vector (AODV) Routing. *IETF RFC 3561*, July 2004.

[57] Maxim Podlesny and Sergey Gorinsky. Rd network services: differentiation through performance incentives. In *SIGCOMM*, pages 255–266. ACM, 2008.

[58] Sumit Rangwala, Apoorva Jindal, Ki-Young Jang, Konstantinos Psounis, and Ramesh Govindan. Understanding congestion control in multi-hop wireless mesh networks. In *MOBICOM*, pages 291–302. ACM, 2008.

[59] R. Sivakumar S.-J. Park, R. Vedantham and I. F. Akyildiz. Garuda: Achieving effective reliability for downstream communication in wireless sensor networks. 7(2):214–230, 2008.

[60] Kun Tan, Jingmin Song, Qian Zhang, and Murari Sridharan. A compound tcp approach for high-speed and long distance networks. In *INFOCOM*. IEEE, 2006.

[61] K. Tang and M. Gerla. Mac reliable broadcast in ad hoc networks. In *Proc. IEEE Military Communications Conference (Milcom'01)*, pages 1008–1013, Mclean, VA, USA, 2001.

[62] Jana van Greunen and Jan Rabaey. Lightweight time synchronization for sensor networks. In *Proceedings of 2nd ACM WSNA*, San Diego, CA, USA, 2003.

[63] Guojun Wang, Tian Wang, Weijia Jia, Minyi Guo, Hsiao-Hwa Chen, and Mohsen Guizani. Local update-based routing protocol in wireless sensor networks with mobile sinks. In *ICC*. IEEE, 2007.

[64] David X. Wei, Cheng Jin, Steven H. Low, and Sanjay Hegde. Fast tcp: motivation, architecture, algorithms, performance. *IEEE/ACM Trans. Netw.*, 16(6):1246–1259, 2006.

[65] T. Winter and P. Thubert. RPL: IPv6 Routing Protocol for Low power and Lossy Networks. *IETF Internet draft,draft-ietf-roll-rpl-04*, october 2009.

[66] S. Xu and T. Saadawi. Does the ieee 802.11 mac protocol work well in multihop wireless ad hoc networks? *IEEE Communications Magazine*, 39(6):130–137, 2002.

[67] Fan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, and Lixia Zhang. A two-tier data dissemination model for large-scale wireless sensor networks. In *MOBICOM*, pages 148–159. ACM, 2002.

[68] J. Hui S. Chakrabarti Z. Shelby, P. Thubert and E. Nordmark. Neighbor Discovery for 6LoWPAN. *IETF Internet draft,draft-ietf-6lowpan-nd-03*, May 2009.

[69] Hui Zang, Majid Ghaderi, and Ashwin Sridharan. Tcp-aware power control in wireless networks. In *ICNP*, pages 334–343. IEEE Computer Society, 2009.

[70] Mohamed-Haykel Zayani and Vincent Gauthier. Usage of ieee 802.15.4 mac cphy model.

[71] Hongqiang Zhai and Yuguang Fang. Distributed flow control and medium access in multihop ad hoc networks. *IEEE Transactions on Mobile Computing*, 5:1503–1514, 2006.

[72] J. Zheng and M. J. Lee. A resource-efficient and scalable wireless mesh routing protocol. *Special Issue of Elsevier Ad Hoc Networks Journal on Wireless Mesh Networks*, 5:704–718, 2007.