A SCHEDULING METHOD FOR SPORADIC TRAFFICS IN INDUSTRIAL IOT

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BAVER ÖZCEYLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2017

Approval of the thesis:

A SCHEDULING METHOD FOR SPORADIC TRAFFICS IN INDUSTRIAL IOT

submitted by **BAVER ÖZCEYLAN** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Tolga Çiloğlu
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Buyurman Baykal
Supervisor, **Electrical and Electronics Eng. Dept., METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Cüneyt F. Bazlamaçcı
Electrical and Electronics Engineering Department, METU _____

Prof. Dr. Buyurman Baykal
Electrical and Electronics Engineering Department, METU _____

Prof. Dr. Elif Uysal Bıyıkoğlu
Electrical and Electronics Engineering Department, METU _____

Prof. Dr. Ali Özgür Yılmaz
Electrical and Electronics Engineering Department, METU _____

Prof. Dr. Bülent Tavlı
Electrical and Electronics Engineering Department, TOBB ETU _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: BAVER ÖZCEYLAN

Signature :

ABSTRACT

A SCHEDULING METHOD FOR SPORADIC TRAFFICS IN INDUSTRIAL IOT

Özceylan, Baver
M.S., Department of Electrical and Electronics Engineering
Supervisor : Prof. Dr. Buyurman Baykal

September 2017, 73 pages

Internet of Things technology continues to develop as a commercial value and it has become one of the core elements of Industry 4.0 paradigm. Together with that, IEEE 802.15.4e standard provides Time-Slotted Channel Hopping (TSCH) operation mode especially for industrial applications that have strict QoS requirements. In spite of the fact that the standard defines frame structure in MAC layer, there has been no standardization in scheduling for TSCH frame yet. It brings serious challenge for engineering design since industrial applications have stringent demands such as high reliability, low latency and energy efficiency. Industrial applications can generate two types of traffic flows, which are periodic and sporadic. There have been many studies on periodic traffic flow in which it is proven that dedicated cell property of TSCH can fulfill the requirements. Whereas, there is a lack of research on sporadic traffic, which can be handled with shared cell property of TSCH. To this end, a shared cell scheduling method for sporadic traffic flows in Industrial IoT is introduced in this thesis. Proposed method exploits the tree structure of RPL protocol, which is a decisive standard for Industrial IoT, in order to divide the network into independent subnetworks that consist of one parent node and a number of child nodes. The method takes its basis from the idea that the whole network optimization can be achieved by separate subnetwork optimizations, which corresponds to finding an optimum number of shared cells and distribution of child nodes among these cells to handle traffic, which originated from child nodes, with meeting QoS requirements. The method is

based on estimations according to a heuristic model of the subnetworks. The results obtained by simulations prove that proposed method minimizes energy consumption without violating QoS requirements. It is also observed that optimum results can be accomplished with this method whether traffic generation rate of child nodes in a subnetwork are the same or different. Thus, granting parent nodes the ability to manage trade-offs between energy consumption, latency and reliability is the outcome of this thesis.

Keywords: TSCH, Shared Cell, Scheduling, IoT, Sporadic Traffic

ÖZ

ENDÜSTRİYEL NESNELERİN İNTERNETİNDE DÜZENSİZ TRAFİKLER İÇİN BİR ÇİZELGELEME YÖNTEMİ

Özceylan, Baver
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Buyurman Baykal

Eylül 2017 , 73 sayfa

Nesnelerin İnterneti teknolojisi, ticari bir değer olarak gelişimini sürdürmektedir ve Endüstri 4.0 paradigmasının temel unsurlarından biri haline gelmiştir. Bununla beraber, IEEE 802.15.4e standardına Zaman Dilimli Kanal Atlamalı (TSCH) çalışma modu eklenmiştir. Bu mod özellikle katı hizmet kalitesi (QoS) gereksinimleri olan endüstriyel uygulamalar için tasarlanmıştır. TSCH standardı ortama erişim kontrolü katmanında çerçeve yapısını tanımlamış olmasına rağmen, çerçevenin çizelgelenmesi ile ilgili bir standart bulunmamaktadır. Endüstriyel uygulamalar yüksek güvenilirlik, düşük gecikme süresi ve enerji verimliliği gibi katı taleplere sahip olduğu için TSCH çerçeve çizelgelenmesi ciddi bir mühendislik tasarımı gerektirir. Endüstriyel uygulamada trafik akışı periyodik veya düzensiz olabilir. Periyodik trafik üzerine çokça çalışma mevcuttur. Bu çalışmalar gösteriyor ki TSCH standardında bulunan özel hücre atama yöntemi bahsedilen gereksinimleri karşılayabilmektedir. Buna rağmen, paylaşımlı hücre atama yöntemi ile düzensiz trafikleri çizelgeleme konusunda yeteri kadar araştırma mevcut değildir. Bu tez çalışmasında, Endüstriyel IoT için paylaşımlı hücre kullanımı ile düzensiz trafik çizelgeleme yöntemi sunulmuştur. Önerilen yöntem, ağı bir üst düğüm ve birden fazla alt düğümlerden oluşan bağımsız alt ağlara bölmek için, Endüstriyel IoT teknolojisinde kesinleşmiş standart olan RPL protokolünün ağaç yapısını kullanmaktadır. Yöntem, her bir alt ağın ayrı ayrı optimize edilmesinin tüm ağı optimize edeceği fikrine dayanmaktadır. Bir alt ağı optimize etmek için, QoS ge-

reksinimlerini karşılayan optimum paylaşımlı hücre sayısı ve alt düğümlerde oluşan trafiği karşılayabilecek optimum alt düğüm dağılımı bulunur. Önerilen yöntem, bir alt ağın deneysel modeli ile yapılan tahminlere dayanmaktadır. Deneylerle elde edilen sonuçlara göre, önerilen yöntem QoS gereksinimlerini bozmadan enerji tüketimini düşürebilmektedir. Ayrıca, bir alt ağdaki alt düğümlerin trafik üretim oranlarının eşit ve farklı olduğu durumlar ile yapılan deneylerde, iki durumda da optimum sonuçların elde edilebildiği görülmüştür. Bu tezin sonucunda, üst düğümlere güvenilirlik, gecikme süresi ve enerji tüketimi arasında olan ödünleşme ilişkisini yönetebilme yeteneği verilmektedir.

Anahtar Kelimeler: TSCH, Paylaşımlı Hücre, Çizelgeleme, Nesnelerin İnterneti, Düzensiz Trafik

To my family

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Prof. Dr. Buyurman Baykal for the continuous support, his guidance and encouragement during my M.S. study and related research. I am grateful to him for introducing me this research area that perfectly matches my interests.

I would like to offer my special thanks to Berk Ünlü for his valuable friendship and great support. In addition to his technical support, working with him has always been a pleasure. I am grateful to Ecenaz Erdemir for her enjoying company during this summer. I would like to thank Hasan İhsan Turhan for his support and sharing his experience.

I would like to show my gratitude to my father, Doğan Özceylan and my mother, Filiz Özceylan for their unconditional love and support. They have made education of their children the priority of their lives. I am also grateful to my sister Helin Özceylan for her unconditional love and support. She is the joy of our family.

Finally, I offer my deepest love and thanks to Kardelen Güneş for being with me all the time.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
6TiSCH	IPv6 over the TSCH mode of IEEE 802.15.4e
ASN	Absolute Slot Number
CoAP	Constrained Application Protocol
DAG	Directed Acyclic Graph
DODAG	Destination Oriented Directed Acyclic Graph
DSSS	Direct Sequence Spread Spectrum
EB	Enhanced Beacon
FHSS	Frequency-Hopping Spread Spectrum
HOL	Head of Line
HTTP	Hyper-Text Transfer Protocol
ICMP	Internet Control Message Protocol
IE	Information Element
IoT	Internet of Things
IP	Internet Protocol
LAN	Local Area Network
MAC	Medium Access Control
OF	Objective Function
PDR	Packet Delivery Ratio
PRR	Packet Reception Ratio
QoS	Quality of Service
ROI	Region of Interest
RPL	IPv6 Routing Protocol for Low-Power and Lossy Networks
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TSCH	Time Synchronized Channel Hopping
UDP	User Datagram Protocol
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network

CHAPTER 1

INTRODUCTION

In history, there are three radical industrial developments–i.e. industrial revolutions. It begins with invention of steam power and mechanical production. 2nd industrial revolution brings mass production assembly lines by using electrical energy. 3rd industrial revolution is introduction of automation technologies by digitalization of factories. Although effects of the last revolution is dominant in the present industry, the latest trends such as Cloud Technology, Internet of Things (IoT), Smart Machines and Big Data trigger 4th industrial revolution, namely Industry 4.0. [25]

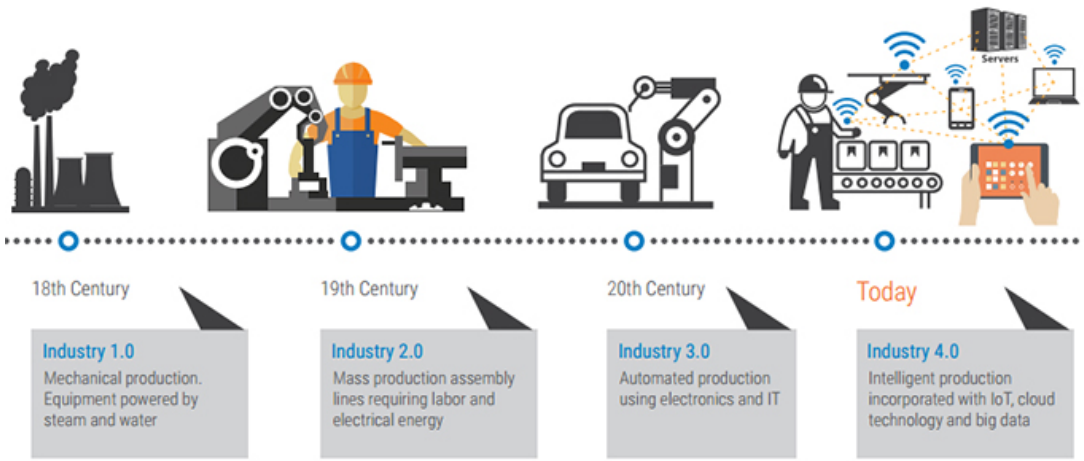


Figure 1.1: Industrial Revolutions [1]

The main idea behind Industry 4.0 paradigm is shifting control system from centralized controller to decentralize intelligent which means self-organized machines. Industry 4.0 requires digitalization of every physical item by using sensors, actuators and RFID tags so that production machines collect and share data in real time. This is the key feature of Industry 4.0 since its main objective is adapting the process to con-

stantly changing market demand, technology option or environmental factors. Thus, IoT technology is essential and plays a major role in Industry 4.0. [42]

Industrial IoT networks have many challenges. Most of them are caused by unreliable nature of wireless medium since industrial applications need high reliability and low-latency. Moreover, one of the objective in Industry 4.0 is achieving minimum energy consumption. Therefore, the communication technology should be energy efficient and support QoS requirements of industrial applications. There have been significant amount of studies [2, 47, 37, 11] that achieve these goals with periodic traffics. However, there is lack of research on sporadic traffics in the literature. The focus of this thesis is creating a scheduling method for sporadic traffics with respect to Industrial IoT standards in order to achieve energy efficiency with meeting QoS requirements of industrial applications.

This thesis is divided into 5 chapters. In Chapter 2, there is a brief information about IoT, well-known solutions for industrial communications and standards for Industrial IoT in detail. Proposed method is explained in Chapter 3 and it is validated by simulations in Chapter 4. Finally, Chapter 5 concludes the thesis, and some important results are highlighted in this chapter.

CHAPTER 2

BACKGROUND

IoT technology provides worldwide connectivity between each device in the world. It means that smart phones, laptop computers, sensors, actuators, *etc.* connect to the Internet so as to communicate with each other. An illustration of IoT Network can be seen in Figure 2.1. It points out that the core of IoT Network is the Internet, which connects geographically separated local networks to each other via edge devices–i.e. access points. Moreover, a local network may connect to the Internet via a edge device in another local network. In IoT Network, local networks use wireless communication due to a large number of mobile devices. Additionally, there are two types of wireless network, WLAN and WPAN, regarding to local requirements.

WLANs are based on IEEE 802.11 [44]–i.e. Wi-Fi, which specifies physical layer and data link layer protocols for wireless medium. The main aim is setting up connections between devices which are seen as a cabled LAN network by upper layers. Thus, a packet can be easily sent to any device in IP network infrastructure. Moreover, Wi-Fi is for computer-to-computer connectivity, which implies high throughput with high power consumption. On the other hand, WPANs do not require high throughput and long transmission range like WLANs [17]. It means that energy efficiency is a major characteristic of WPANs. Unlike Wi-Fi, there is no absolute protocol for WPANs. The most well-known protocols are Bluetooth [5] and ZigBee [53]. Moreover, WPANs are dominant parts of IoT Network due to their energy efficiency feature. There are many application areas of IoT technology–e.g. home network, body area network, *etc*. However, the focus of this thesis is application of IoT technology to industrial networks, namely Industrial IoT.

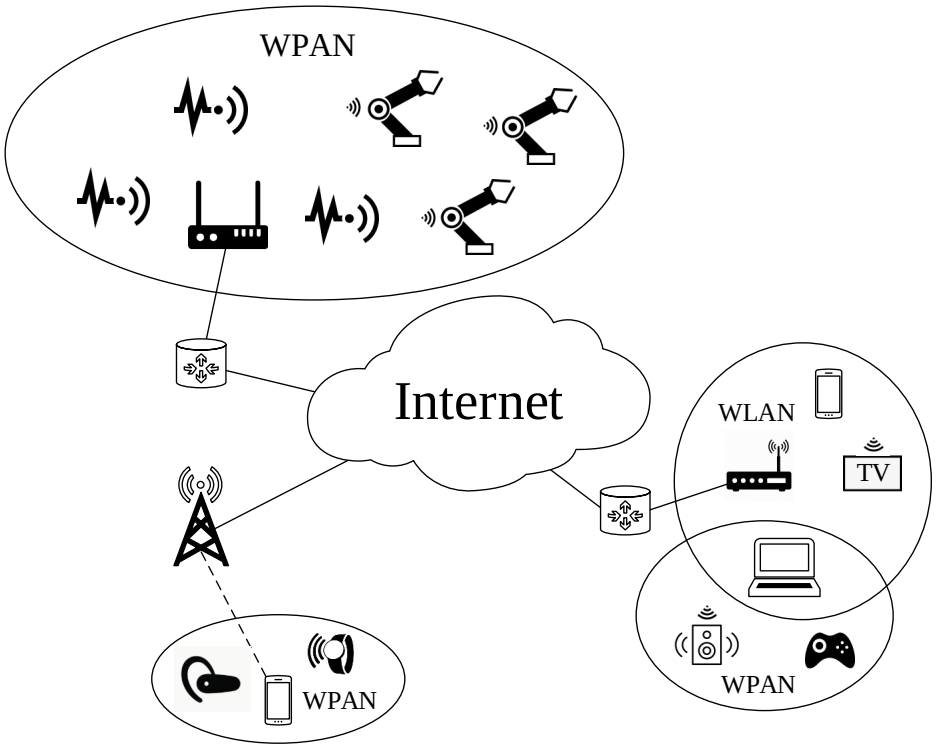


Figure 2.1: IoT Network

In an industrial network, thousands of sensors are used to monitor temperature, pressure or other important parameters. The measured data is processed to control actuators, manage production cycle, or prevent an accident. Thus, the network has strict QoS requirements since any failure will result in catastrophic consequences. In fact, the requirements can be easily achieved by using cables with a proper MAC protocol. However, an increasing number of sensors, actuators and control devices makes it very expensive. For example, drawing cables can cost up to $\$1,000/ft$ for an industrial plant [13]. Additionally, installing and maintaining those cables is a significant expenditure for the business. As a result, wireless solution is inevitable for industrial networks in order to reduce the cost. However, wireless medium is very challenging because interference brings low reliability, antennas consume much more power and broadcast property threats data security. The most important issue in industrial networks is power consumption. Due to the fact that the network mostly consists of low power devices, which are powered by batteries, the lifetime of a low power device depends on power consumption. IEEE 802.15.4 [43] standard was first published

in 2003, then became a common standard for low power wireless device networks. Many protocol stacks for these networks are built on IEEE 802.15.4 standard. Although energy efficiency is a major issue for low power networks, the unreliable nature of wireless medium is also an obstacle in the case of industrial networks. There are two reasons of the unreliability; unscheduled transmissions and independent deployment of the same or different wireless networks. To overcome the first one, Time Division Multiple Access (TDMA) method, which needs time synchronization, is essential for MAC layer. The second one, on the other hand, can be solved with channel hopping technique, which creates frequency diversity without modifying IEEE 802.15.4 physical layer standard. As a result, these two techniques are prerequisites in order to reliable communication for industrial networks [13].

Time synchronization and channel hopping are at the heart of the two most popular wireless industrial network protocol stacks, which are WirelessHART [45] and ISA100.11a [30]. In 2012, the IEEE 802.15.4e amendment introduces TSCH (Time Synchronized Channel Hopping) operation mode for industrial networks. It includes core ideas of time synchronization and channel hopping. IEEE 802.15.4e TSCH mode defines physical layer and MAC constraints. It offers low transmission power and adequate data rate for industrial networks. However, maximum frame size defined by IEEE 802.15.4 is 127 bytes which is very low compare to regular Ethernet. In fact, this issue is not a problem for single industrial network because upper layers can be designed according to this constraint–e.g WirelessHART and ISA100.11a. On the other hand, in case of Industrial IoT, each device should appear as a regular host in the Internet. It means that MAC layer protocol should transmit IP packets and work with existing IP network infrastructure, like Ethernet. Additionally, IPv6 addressing technique should be used due to a large number of end devices in Industrial IoT. For those reasons, IETF 6LoWPAN working group defines a compaction and fragmentation mechanism for transmission of IPv6 packets over a IEEE 802.15.4 network [26, 49]. This mechanism is seen as a first step for realization of IoT concept. To this end, several standards, such as IPv6 Routing Protocol for Low Power and Lossy Networks (RPL) [3], Constrained Application Protocol (CoAP) [41], are designed by IETF working groups in order to enable client-server interaction between a low power wireless device and a regular host in the Internet.

In 2013, IETF working group, called 6TiSCH, was created to build a protocol stack for Industrial IoT by combining TSCH standard which fulfills the requirements of industrial networks and IP-enabled upper layer protocols. Although most of the issues have already been solved, one of the major topics, scheduling for TSCH frame, is still an open issue.

In this chapter, firstly WirelessHART and ISA100.11a are summarized in Section 2.1. Then, 6TiSCH architecture is explained in Section 2.2 with brief information for each layer. Afterwards, IEEE 802.15.4e TSCH mode of operation and 6top protocol (6P) proposed by 6TiSCH working group are clarified in detail in Sections 2.3 and 2.4 respectively. Lastly, state of art scheduling solutions for TSCH frame are given in Section 2.5.

2.1 WirelessHART and ISA100.11a

WirelessHART adds wireless interface to field devices in HART network. It was proposed in 2007 by HART Communication Foundation. On the other hand, ISA100.11a entered the competition in 2009. Both technologies adopt similar layered architectures with slight modifications, which are shown in Figure 2.2. All layers are well defined in both technologies without giving flexibility to the user.

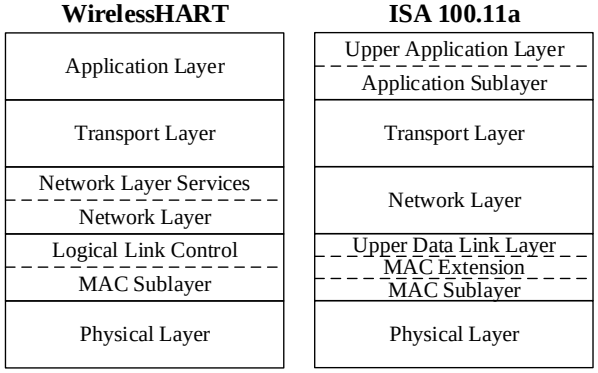


Figure 2.2: Protocol Stacks of WirelessHART and ISA100.11a

Both technologies use IEEE 802.15.4 standard's specifications for physical layer. According to the standard, information signal spreads across the available frequency channels. Both technologies enable Frequency-Hopping Spread Spectrum (FHSS)

modulation technique, which is altering the channel for each packet transmission. FHSS increases the reliability since the effect of noise in one channel can be fairly distributed to the spectrum thanks to frequency hopping.

Data Link Layers of WirelessHART and ISA100.11a are different. WirelessHART has Logical Link Control and MAC Sublayer parts. Whereas, ISA100.11a Data Link Layer consists of Upper Data Link Layer, MAC Extension and MAC Sublayer. MAC sublayer is a subset of IEEE 802.115.4 standard for MAC layer and it provides an integrity with physical layer for both technologies. MAC Extension adds extra features that are not included in IEEE 802.15.4 like CSMA-CA. Although Logical Link Control is responsible for only one-hop links, Upper Data Link Layer in ISA100.11a does not only handle one-hop transmission, it also manages routing within a subnet, unlike WirelessHART where routing is responsibility of Network Layer. Moreover, both technologies use TDMA and frequency hopping for channel access. TDMA implies that time is divided into slots. Certain number of successive slots create a superframe which repeats in time and the length is determined by the number of slots. A time slot in a superframe can be assigned to a link with one transmitter and one receiver or to links with one transmitter and multiple receivers –i.e. broadcast messages. Moreover, different channels can be used for an assigned time slot in each superframe thanks to frequency hopping. Besides, ISA100.11a supports a nondeterministic CSMA-CA based channel access method.

In both technologies, Network management functions are handled by the host devices in a centralized manner. The host can connect to wireless devices via backbone router or gateway. Additionally, the upper layers are designed for only WirelessHART and ISA100.11a networks and cannot allow any flexibility. As a result, these technologies are not suitable for connecting industrial networks to IoT Network. However, TDMA and frequency hopping methods for channel access are promising solutions, and that is why 6TiSCH working group is inspired by WirelessHART and ISA100.11a. [40]

2.2 6TiSCH Architecture

IETF 6TiSCH working group published a document in January 2017 [48]. This is a draft document, not a RFC, which means the working group is still conducting their research on this topic. The working group is studying on a standard architecture, namely 6TiSCH. The goal is to merge industrial networks and IoT technology–i.e. Industrial IoT. In the document, current status of the standardization process is explained. In this section, proposed 6TiSCH architecture, which is shown in Figure 2.3, is summarized according to this most recent document.

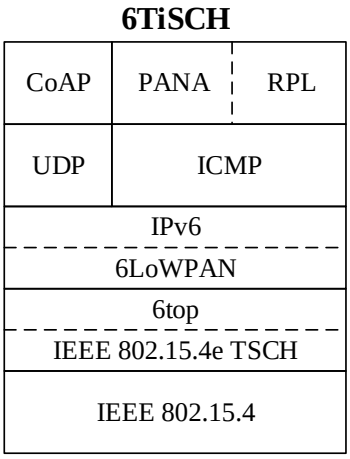


Figure 2.3: Protocol Stack of 6TiSCH

The objective is to combine already constructed IP network and IEEE 802.15.4e TSCH standard whose practicality is proven in process control by ISA100.11a and WirelessHART. Upper layers are application specific like the Internet. For example, UDP and CoAP (Constrained Application Protocol) pair is very similar with TCP and HTTP. Another example is ICMP and PANA (Protocol for Carrying Authentication for Network Access) which are used to secure the join process. In the network layer, datagrams are in IPv6 format since it is expected that there will be a large number of end devices. Thus, an adaptation layer is needed for combining TSCH and IPv6. For this reason, 6LoWPAN is used as an adaptation layer and it is clarified in Section 2.2.1. Additionally, the de-facto routing protocol for low power devices is RPL, which is explained in Section 2.2.2.

IEEE 802.15.14e TSCH manages time synchronization and frequency hopping for IEEE 802.15.4 physical layer. Detailed description of TSCH is in Section 2.3. 6top layer provides link abstraction for IP network layer and management interfaces for scheduling algorithms. Additionally, the working group creates 6top protocol (6P) for this layer and specifications of the protocol are explained in Section 2.4.

2.2.1 6LoWPAN

6LoWPAN is an adaptation layer for transmitting IP packets over IEEE 802.15.4 networks. It is necessary because of the reasons stated below [27];

- Infrastructure of IP network is already constructed and widely-used.
- IP devices can connect to the Internet without the necessity of extra entities like gateways and proxies.
- IP-based technologies are open, free and proven to be working
- There are plenty of study on IP networks.
- IPv6 can support a large number of devices.

Additional layer, 6LoWPAN, is needed since IP packets cannot be directly transmitted over IEEE 802.15.4 networks due to the limitation in frame size. IPv6 and UDP headers are 40 and 8 octets respectively. IEEE 802.15.4 frame size is 127 octets and MAC header can be 46 octets with the security overhead. Thus, remaining space for application data is 33 octets although the maximum transmission unit size for IPv6 is 1280 octets. For those reasons, fragmentation/reassembly of IPv6 packets is essential. Additionally, header compression is also necessary to prevent excessive fragmentation and to make a control packet fit into a single IEEE 802.15.4 frame. Therefore, 6LoWPAN adaptation layer makes using IPv6 possible by compressing and fragmentation/reassembly methods. Although there has been no completed standard yet, requirements and constraints for those methods are given in the document [26] published by IETF 6LoWPAN working group.

2.2.2 RPL

RPL is an IPv6 routing protocol for low-power and lossy networks. It is a standard [3] proposed by IETF in order to meet the requirements stated by 6LoWPAN working group.

RPL procedure begins with discovering the topology due to the fact that a wireless network does not have predefined topology. Then, it selects specific links to get optimum routes. Traffic flows routed by RPL are generally from many nodes to one node, called root node, or from the root to many nodes. It means that both upward and downward traffics are supported. RPL creates Destination Oriented Directed Acyclic Graphs (DODAGs) to simplify the topology hence it is able to cover a large number of nodes. Each DODAG has one root node and many leaf nodes whereby a tree structure is constructed. Single RPL instance can create more than one DODAG on the condition that they are disjoint. Moreover, multiple RPL instances can be defined for one network and they can cooperate with prioritization method. Each instance should have different objective to optimize. An objective is defined by Objective Function (OF) feature of RPL. Although default OF is defined by the standard, the user can easily design an OF to replace the default one.

The aim of default OF is to minimize rank value of each node, which is defined as hop distance to the root. From this point onward, RPL denotes RPL with default OF. Figure 2.4 shows an illustration of route change according to RPL in a sample DODAG. In the figure, arrows denote available links that are used for data traffic routing. Whereas, application data cannot be transmitted over links that are represented by dashed lines. Node 1 is root of the DODAG and rank value of each node is written in the figure. RPL changes DODAG topology from A to B in order to minimize the ranks. After RPL procedure, rank values of nodes 3 and 6 decrease by 1. Moreover, each available link creates a parent-child relationship. A node with lower rank value becomes parent of a node with higher rank value. According to the example, node 2 is parent of nodes 4, 5 and 3 according to the topology A in Figure 2.4. However, parent of node 3 is changed to node 1 in topology B since rank value of node 1 is lower than node 2. As a result, each node, except the root, has a parent and data traffic that can only be between parent-child pairs.

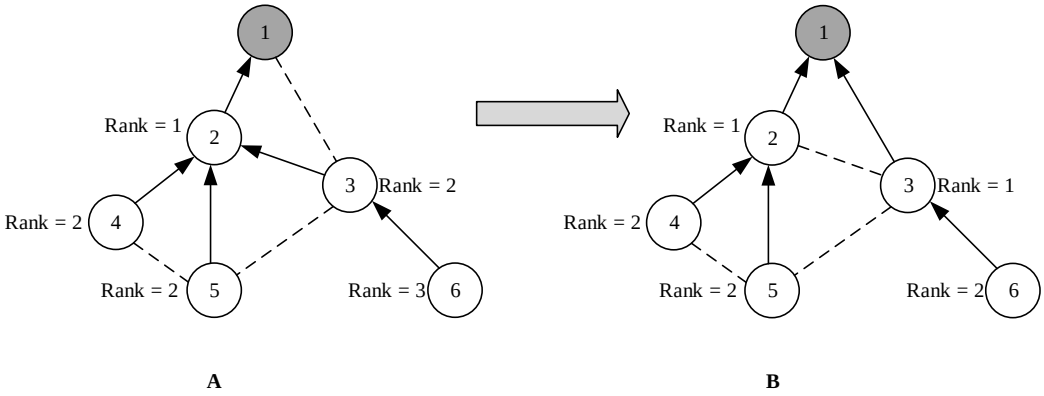


Figure 2.4: RPL Optimization Illustration

A DODAG is constructed and maintained by DAG Information Object (DIO) messages, which are periodically sent as multicast by every node. Nodes should receive and process every DIO messages even if these messages are not transmitted by child or parent of corresponding node. After a DIO is obtained, the node uses the information to join new DODAG or to maintain existing one. Another mechanism is DAG Information Solicitation (DIS) messaging which requests a DIO from specific node so that the node can change its parent when there is a better option.

As a Result, RPL leads a tree structure by creating parent-child relationships. This feature restricts traffic flows and eases to manage routing. Moreover, scheduling method for IEEE 802.15.4e TSCH frame should be designed with respect to RPL features, in fact, the tree structure can also help to manage the schedule.

2.3 IEEE 802.15.4e TSCH

IEEE 802.15.4e standard is enhanced version of regular IEEE 802.15.4 which is used by old-fashion technologies like ZigBee. The physical layer specifications are not modified in order to allow same hardware use. The MAC layer, however, is one of the features that are improved. Moreover, TSCH is actually one of the operation modes defined in the standard and it is structured especially for process automation applications.

TSCH divides time into fixed length slots and assigns each one a unique identification number, called *Absolute Slot Number* (*ASN*). Each node in the network stores an *ASN* to indicate current time and increases it by 1 after a slot passes. It means that every *ASN* indicates the same time value at a time. Thus, it can be thought as a global time reference. TSCH defines repeating slotframes by combining a certain number of successive slots. Each slot in a slotframe is specified by a offset, called *slotOffset*, respect to the first slot of a slotframe. *slotOffset* is computed as in Equation 2.1. *frameLength* is the size of a slotframe and "%" symbolizes modulo operation. An example of frame structure is illustrated in Figure 2.5. Boxes represent time slots and *ASN* value of each time slot is expressed. In the figure, *frameLength* of the slotframe is *FL*. In the middle of time slots, their *slotOffset* values are written according to Equation 2.1. Therefore, a time slot specified by *slotOffset* repeats itself and the period is determined by *frameLength*.

$$slotOffset = ASN \,\%\, frameLenght \tag{2.1}$$

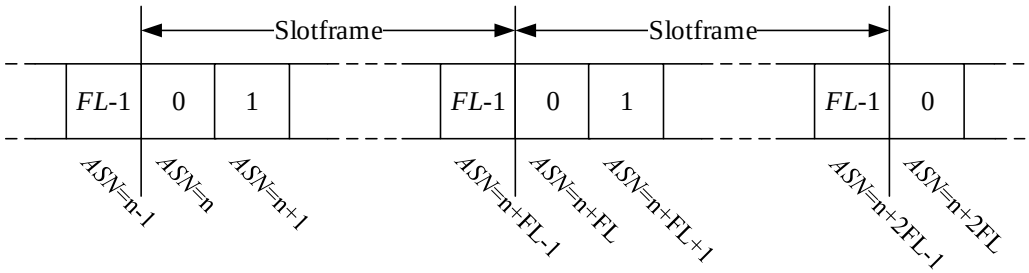


Figure 2.5: TSCH Frame Structure

Exact slot timing is mandatory to maintain TSCH frame. However, slight time shifting is inevitable in such systems. Because of that, listen process starts before transmit process. The time interval between listen and transmit processes is determined by a parameter, namely *GuardTime*, defined in the standard. There is also a time interval between slot and transmit process initiations, which is adjusted by parameter called *TsTxOffset*. If a receiver node does not sense any data within this time interval, it stops listening to save energy. A slot process is shown in Figure 2.6. It is seen that an acknowledgment procedure starts after the data transmission process. It simply means sending *Ack* signal to a transmitter node by a receiver node for reporting about

successful reception of the data. This follows same procedure like data transmission, and the time interval between end of data transmission process and acknowledgment procedure initiation is determined by a parameter called *TsTxAckDelay*.

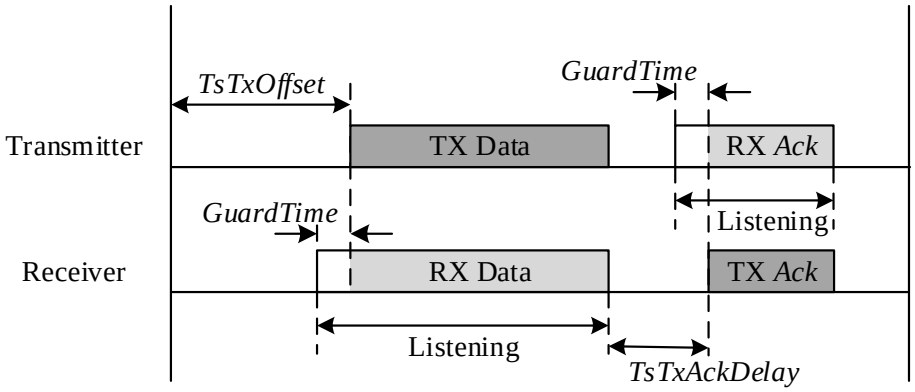


Figure 2.6: TSCH Slot Process

In addition to that, a node re-synchronizes its clock according to one neighbor node which is chosen as time-source neighbor. Since selection of this neighbor is not specifically described in IEEE 802.15.4e TSCH standard, parent node can be used for this purpose. Synchronization can be checked according to data transmission or *Ack* signal starting time since *TsTxOffset* and *TsTxAckDelay* parameters are pre-defined and known by every node. Nevertheless, synchronization fails when a clock shifts more than the time interval specified by *GuardTime*. To prevent this situation, a node should periodically receive packet from its time-source neighbor. *Enhanced Beacon* (*EB*) is suitable for this purpose. Although the standard suggests that each node should periodically send *EB*, it does not provide any scheduling method. An *EB* includes *Information Elements* (*IE*s) which enables information exchange at the MAC layer. *EB*s is also used for many purposes such as neighborhood discovery and joining process. It is also utilized for *ASN* synchronization. Additionally, possibility of adding extra *IE* by the user makes the standard flexible on the MAC layer.

Channel hopping is one of the main features of TSCH. According to the physical layer specifications, a node can use 16 different channels, which represents different frequencies. However, it can only broadcast or listen one channel at a time. Additionally, TSCH gives flexibility to the user on channel selection. Since some of them have

bad reputations, the user can create a channel hopping sequence among high quality ones. Instead of directly selecting a channel, a node selects an offset value, and then, it chooses different frequencies at different times according to following equation;

$$f = F[(ASN + \textit{channelOffset}) \% N_{channel}] \tag{2.2}$$

In the Equation 2.2, F is a function, which can be a lookup table or a hash function. $N_{channel}$ is channel hopping sequence length. Selected offset value is expressed as *channelOffset* and it should be in $[0, N_{channel} - 1]$ interval to avoid overlapping.

Multi-channel mechanism and time division divide a slotframe with respect to frequency and time. This division creates a fix sized frame matrix, which repeats itself in time. In a frame matrix, each element is named as 'cell'. An example is shown in Figure 2.7. Rows and columns are for *channelOffset* and *slotOffset* respectively. Thus, a cell can be denoted with two parameter–i.e $(\textit{slotOffset}, \textit{channelOffset})$. In the example, *frameLength* is 10 and there are 8 available frequencies. Also, 3 cells are colored as an illustration. $(8, 0)$, $(4, 2)$ and $(5, 5)$ cells are marked with blue, green and red colors respectively.

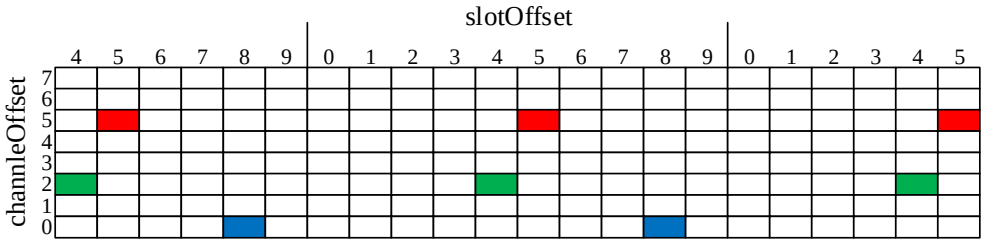


Figure 2.7: An Example Frame Matrix

Although IEEE 802.15.4e TSCH standard defines specifications for a cell, it does not restrict scheduling and link establishment. In a frame matrix, each cell can be assigned to one or multiple links. For this reason, two types of cell, dedicated and shared, are described in the standard. A dedicated cell can be assigned to only one transmitter-receiver pair–i.e. one link, whereas a shared cell is intentionally assigned to more than one link. A dedicated cell does not have collision avoidance property

because proper scheduling guarantees that the link does not share cell with another link. Although there is always a probability of packet loss, which is caused by channel situation like interference, there is no need for collision avoidance thanks to channel hopping mechanism. In the case of shared cell, CSMA-CA mechanism is defined in the standard since collision is inevitable. An example of schedule is illustrated in Figure 2.8 and it corresponds to the topology shown in Figure 2.9. It is clear that 4 channels can be used and *frameLength* is 4. Cell (2,1) is a shared cell and the others are dedicated cells. If nodes E and D try to send packet at the same time, collision occurs. Then, retransmission process follows the backoff algorithm, which is defined by the standard to reduce repeated collisions. [9]

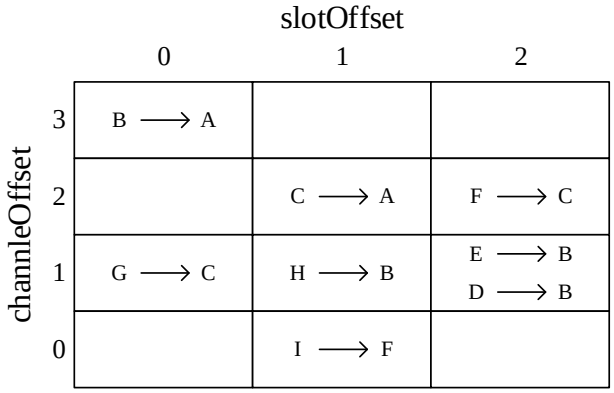


Figure 2.8: A Schedule Example

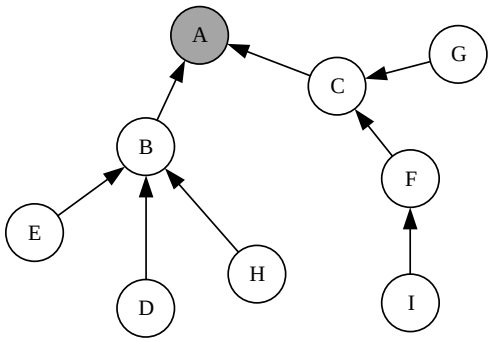


Figure 2.9: Topology for the Schedule Example in Figure 2.8

Link establishment process is also not clarified in the standard although a mechanism to execute a schedule is well defined. That is why 6TiSCH working group asserts 6top layer, which handles signaling between two nodes to establish a link. Additionally, it offers an interface to integrate a scheduling algorithm even though it does not propose one.

2.4 6P

6top layer is inserted above IEEE 802.15.4e TSCH by 6TiSCH working group. The aim is to establish and manage links between neighbor nodes. Although centralized and distributed scheduling options exist in TSCH standard, 6TiSCH working group highlights distributed scheduling, and proposes 6top protocol (6P) [51]. It allows a node to communicate with its neighbors to change schedule by adding, deleting or reallocating TSCH cells without centralized coordination. However, 6P cannot directly decide when to change schedule. This decision is made by a 6top Scheduling Function (SF). Since applications can have different requirements, multiple SFs can coexist. Although some examples of SF are described in the standard, 6P allows the user to add different SFs. This feature provides flexibility on schedule management. 6P is described according to the document [51], which is updated by the working group on June 27, 2017. It is a draft version and regularly updated according to the current status of the working group.

6P Transaction is a negotiation process between two neighbor nodes and 2 types of transaction are introduced. The first one is 2-step 6P Transaction which is showed in Figure 2.10. In the example, source node has a traffic flow to destination node. SF that is responsible for this traffic on source node decides to add extra 2 cells. The number of cells is indicated in *NumCells* part of 6P ADD Request message. For 2-step 6P Transaction, source node also informs destination node about preferred cells in *CellList* part of the message. Additionally, message type and command code are denoted in *Type* and *Code* parts. Source node starts a timer after 6P ADD Request. If 6P Response is not received until timeout, it aborts transaction. When destination node receives 6P ADD Request, it invokes corresponding SF to select desired number of cells from *CellList*. Then, destination node sends back 6P Response message.

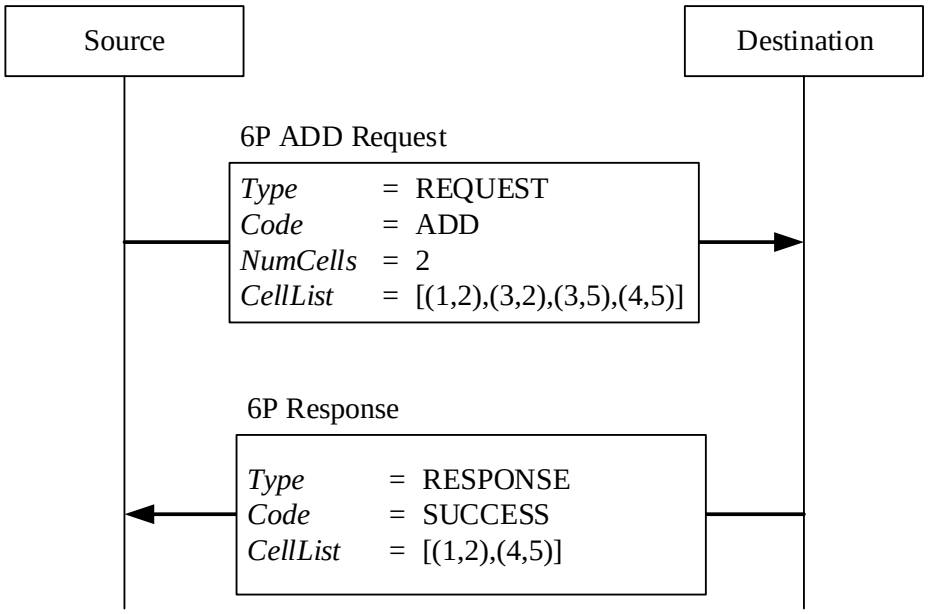


Figure 2.10: 2-step 6P Transaction

The second one is 3-step 6P Transaction. An example for this type is shown in Figure 2.11. Preferred cells are not sent by source node in 6P ADD Request message. On the other hand, destination node offers preferred cells in 6P Response message. Then, source node selects desired number of cells and sends 6P Confirmation message with list of selected cells as *CellList*. Therefore, 6P provides flexibility on cell selection. It means that source or destination node can select cells and the choice is made by the SF.

In 6P, there are 6 commands that a SF can execute. These are Adding, Deleting, Relocating, Counting, Listing cells and Clearing the schedule. For these commands, 3 message types, which are Request, Response and Confirmation, are used. The format of Request message is shown in Figure 2.12. Response and Confirmation messages are in the same format which is indicated in Figure 2.13. Fields are described as follows:

- *Version*: The version number of 6P protocol
- *T*: Type of the message

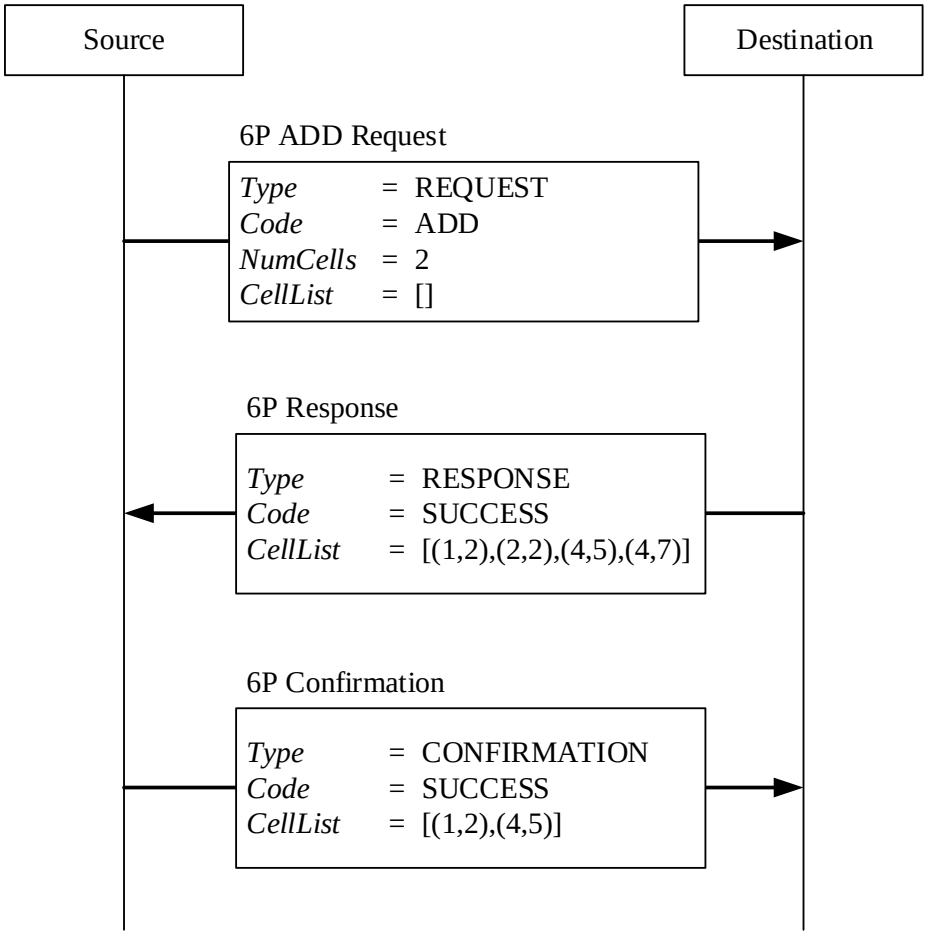


Figure 2.11: 3-step 6P Transaction

- *Code* in Request message: Command identifier
- *Code* in Response and Confirmation messages: Return code
- *SFID*: The identifier number of corresponding SF
- *SeqNum*: Sequence number
- *GEN*: Schedule generation number
- *Metadata*: Extra information to the SF.
- *CellOptions*: The option of cells
- *NumCells*: The number of cells

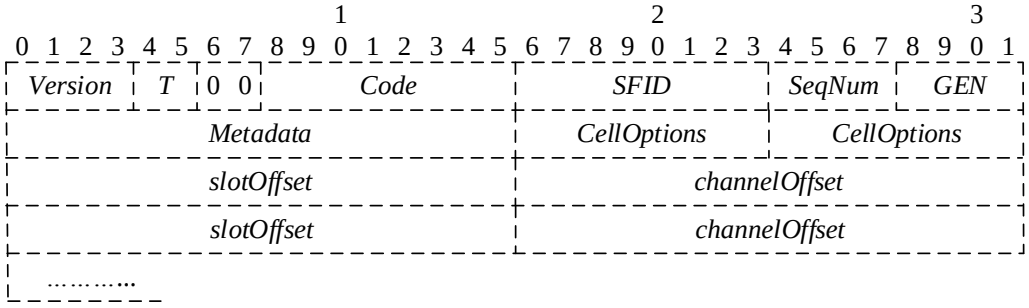


Figure 2.12: 6P Request Message Format

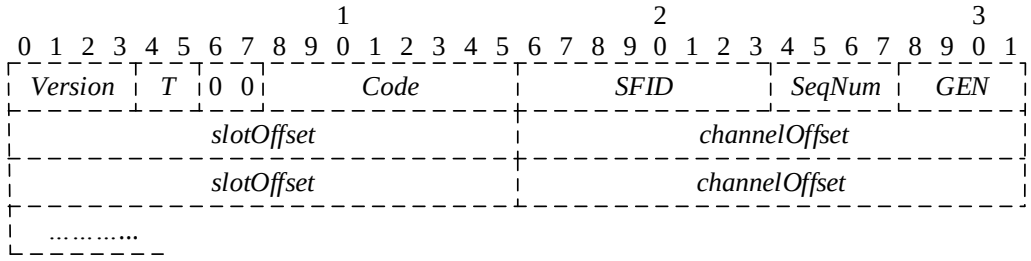


Figure 2.13: 6P Response and Confirmation Message Format

In the both message formats, *T* field is 2-bit and indicates the type of a message with different bit configurations which is shown in Table 2.1. 8-bit *Code* field can take values from 0 to 255 and implies commands or return codes according to the message type. Those are listed in Table 2.2 and 2.3 respectively. *SFID* specifies which SF handles the message. *SeqNum* is used to match messages of the same transaction process, and the value is increased by 1 after a 6P Request is sent to the same neighbor. *GEN* ensures consistency between the schedules among neighbor nodes. Corresponding neighbors increase the value by 1 for each change in the schedule. In addition to those fields, Request message also has 3 extra fields. *Metadata* depends on the SF since *Metadata* directly passes to SF without any modification. This field gives flexibility to the user. *CellOptions* and *NumCells* indicate type and number of cells to be assigned and they also depends on the SF. It means that each SF can define different meanings to this field. Other fields create *CellList*.

Consequently, 6P protocol offers an interface to manage and create a schedule. As a result of SF feature, scheduling algorithms can be easily integrated to 6TiSCH net-

Table 2.1: 6P Message Types

Value	Type
b00	Request
b01	Response
b10	Confirmation

Table 2.2: 6P Command Identifiers

Value	Command
0	Reserved
1	Adding
2	Deleting
3	Relocating
4	Counting
5	Listing
6	Clearing
7-254	Unassigned
255	Reserved

Table 2.3: 6P Return Codes

Value	Return Code
0	Operation Succeeded
1	Generic Error
2	End of List
3	Reset
4	Unsupported 6P Version
5	Unsupported *SFID*
6	Wrong Schedule Generation
7	Busy
8	Not Enough Resources
9	*CellList* Error
10-254	Unassigned
255	Reserved

works. Although, some examples of SF are given by the working group in order to meet simple requirements, it is still an open issue for many applications. In Section 2.5, proposed SFs and other state of art scheduling solutions for TSCH frame are briefly explained.

2.5 Scheduling Methods for TSCH

Scheduling for TSCH frame is mainly assigning cells to links. A scheduling algorithm should decide the number of cells and the position of each cell. It is a challenging issue since inappropriate selection of cells causes packet loss which decreases reliability. Duplex and interference conflicts are two reasons of packet loss. A node cannot transmit and listen at the same time and two nodes should not try to transmit to the same receiver at the same time. Those situations cause duplex conflict. Besides, interference conflict happens when one cell is used by links that are spatially close to each other. Moreover, packet loss also increases power consumption because

a node consumes significant amount of energy when transmitting a packet or listening the channel, and each loss packet triggers retransmission mechanism. In other words, unsuccessful data transmission attempt and unnecessary listening cause waste of energy. Since TSCH is generally used for low power devices, energy efficiency is a major concern in scheduling. Within this context, various scheduling methods exist in the literature. Those methods can be classified into three categories, which are centralized, negotiation based and autonomous scheduling.

Centralized scheduling is the most trivial one. It is based on one coordinator device, which manages schedule according to the gathered data from other devices in the network. WirelessHART and ISA100.11a are commercialized examples which are explained in Section 2.1. In addition to that, Traffic Aware Scheduling Algorithm (TASA) [35, 36] is the most addressed one. TASA builds and manages schedule with respect to topology of the network and the traffic load of each link. Necessary information such as traffic loads is obtained from routing protocol. Afterwards, central coordinator builds schedule by using iterative procedure. Each iteration runs matching and coloring algorithms for one slot. The iteration begins with selecting set of duplex-free links which have traffic loads. Then, coloring algorithm chooses the different channels for each link in order to prevent interference conflict. The authors also prove that only 3 channels are enough to schedule the network thanks to spatial reuse. In addition to TASA, many centralized scheduling algorithms are proposed such as [18, 24, 31, 32]. The differences among them are their primary objectives and algorithm complexities. For example, [18] extends TASA by enabling retransmission to guarantee reliability. Moreover, [24] bounds the delay while energy efficiency is the major concern in [32], unlike TASA which tries to equalize traffic loads with throughput. Although the network becomes more predictable and easy to manage with centralized coordinator, the complexity is very high. It means that these algorithms are not suitable for large and mobile networks. Despite the fact that some of them achieve to decrease the complexity like [31], gathering data from all devices in the network is still a major problem for large networks. For those reasons, 6TiSCH and 6P are designed for distributed scheduling methods, which are negotiation based and autonomous scheduling.

In the case of distributed scheduling, the main challenge is energy efficiency since un-

necessary listening which is named "idle listening" cause waste of energy. Although there are some methods that do not address energy consumption such as [50], this study focuses on low power networks. Distributed algorithms are executed by each node with local information, and overall schedule is created by local decisions. The network reaches a solution with low overhead although this solution is not usually optimum. Distributed scheduling can be created by negotiation or autonomous methods. For example, 6P offers an interface for negotiation by 2-step and 3-step Transactions. However, there are some protocols that do not need any signaling since every node in a neighborhood can reach the same schedule with the same information.

One of the earliest negotiation based scheduling method is proposed in [28]. It takes advantages from label switching. The signaling in this protocol is very similar to Resource Reservation Protocol (RSVP) [52] which is well known for the Internet. It computes the schedule along traffic source and its destination. Although it gives precise results, the signaling overhead is high. On the other hand, DeTAS [2] works with RPL, and each parent computes a schedule for its child nodes. This feature decreases the signaling overhead due to the fact that the schedule of a traffic flow is computed in a distributed manner. Actually, DeTAS is a decentralized version of TASA and it also adjusts a schedule with respect to traffic loads. In [47], algorithm called Wave is proposed. The main goal is to transmit a packet from source to final destination in one frame. To do that, Wave exploits the tree structure of RPL. Each node knows its parent which is one hop destination, set of conflicting nodes and their priority values which implies traffic loads. Wave is an iterative algorithm. In each iteration, a parent node informs its child nodes about current schedule and triggers them to assign a cell to themselves. Child nodes select appropriate cells according to their knowledge and current schedule. After that, each child node reports the schedule to their child nodes in order to start next iteration. This mechanism limits maximum end-to-end delay and the results are very close to the optimum solution. However, the signaling overhead is still high and all procedure starts again when topology or traffic changes. Additionally, hop count of RPL tree structure directly effects the complexity.For those reasons, some modifications to Wave are proposed. For example, [46] allows overlapping while the schedule passes rank by rank. This creates spatial reuse which means spatially separated nodes can assign the same cell to themselves, hence the through-

put increases. Moreover, [20] adds self-healing mechanism to Wave in order to sense and avoid cell collisions.

Palattella *et al.* [37] offer On-The-Fly (OTF) as a SF for 6P. Then, 6TiSCH working group published a document [12] about implementation specifications of OTF. The algorithm has two mechanisms which are bandwidth estimation and resource allocation. A node can calculate required number of cells for a link according to current traffic loads, already reserved cells and QoS requirements. Afterwards, it adds or deletes cells with respect to a hysteresis function since triggering add/delete cell mechanism for every variation in the required bandwidth is undesirable. In other words, the function sets an upper and lower bounds so that cells are added when the required bandwidth increases above the upper band, and cells are deleted if it decreases below the lower band. Additionally, the algorithm always assigns more cells than exact demand to avoid experiencing resource deficiency problem. It is also shown that OTF can adapt time-varying traffics with low latency and high reliability. However, the algorithm schedules all traffic flows between two nodes in one link even though traffic flows can have different requirements. For example, emergency messages need immediate delivery which means low latency; however, periodic messages need low energy consumption. Furthermore, 6TiSCH working group introduced SF1 [4]. It is a hop-by-hop scheduling algorithm and it isolates end-to-end traffic flows by using label switching which means packets are marked with labels. Each label indicate the traffic flow to which the packet belongs. Thus, cells are assigned to labels, not links. Actually, OTF and SF1 are examples given by 6TiSCH working group. They show implementation of a SF and prove that more than one SF can operate in parallel.

Recent studies in the literature develop additional mechanism for SFs like OTF and SF1. For example, close loop control mechanism is proposed in [11]. It tries to adapt traffic variation in time, like OTF. Its difference is using PID controller approach to decrease adaptation time. Another example is [21] which exploits concept in Wave and uses OTF for cell assignment. On the other hand, some latest studies propose complete scheduling algorithms compatible with 6P. In [7], cells, which are located just after cells scheduled for receiving, are scheduled for transmitting in order to decrease buffering delay. However, packet loss is not considered and it is useful only

for linear topologies–i.e. each parent has only one child node. In [23], nodes have a look-up table for cell allocation. The table solves duplex and interference conflicts. A node can find its conflict free cells from table according to its position in the network, which is computed with respect to the information in RPL. The position indicates node's place in the routing tree. Additionally, the table provides pipelined schedule which means packets can be immediately sent to next destination without buffering delay. However, it has several disadvantages like it is not adaptable to traffic variations, look-up table limits the network capacity and irregular RPL tree causes additional power consumption. Additionally, [16] decreases the number of collisions, which are caused by negotiation errors, by selecting cells in the most unused parts of the schedule. To do that, nodes need to monitor the density of the schedule parts. Negotiation errors are also the main issue in [29]. Nodes periodically advertise their schedule in shared cells which are particularly assigned for this purpose. Since advertisement packets are sent as a broadcast, consumed power for this overhead is minimized, and nodes only store information about the neighborhood which is sufficient to avoid negotiation errors. Besides, some proposed methods need modifications in standard protocols such as RPL. For example, Huynh *et al.* [22] allow a node to connect multiple parents in spite of RPL. The aim is to increase the delivery probability since receiving the packet by one of the parents is enough for forwarding to the next hop. It also adjusts IEEE 802.15.4e slot process in order to avoid collision of acknowledgement messages by assigning different priorities to each parent.

Autonomous methods are suitable for highly dynamic and unpredictable networks since central or distributed scheduling entity is not needed. Thus, there is no signaling overhead for adapting network changes. It means fixed schedule is used for all variations of the network. Although it seems that the solution is not an efficient one, signaling to find the most optimum solution creates much more inefficiency. Duquennoy *et al.* [14] introduce Orchestra for this concept. Also, 6TiSCH working group published a document [15] about implementation specification of Orchestra. It bases on building virtual frames with different sizes for each traffic flow–e.g. *EB*s, RPL information or application data. In order to minimize the number of slot collusions in the schedule, Orchestra sets up a rule that the sizes of virtual frames must be coprimes. Duquennoy *et al.* also prove that the probability of collision becomes so

low that it can be neglected. Additionally, nodes calculate cells that are assigned to themselves in a virtual frame by a hash function and their MAC addresses. Cells are configured as shared cell since there is a possibility of assigning one cell to more than one node. Nevertheless, it is shown that Orchestra can reach 99.99% reliability. One of the major contribution of Orchestra is that energy consumption can be decreased by using shared cell feature of IEEE 802.15.4e standard.

CHAPTER 3

A METHOD FOR SCHEDULING TSCH FRAME

3.1 Introduction

An Industrial IoT network can have two types of traffic, which are periodic and sporadic. Periodic traffics can be easily handled by the network since it is predictable. However, sporadic type is difficult to handle because of the long and irregular inter-arrival times. Furthermore, TSCH allows cells to be assigned in a dedicated or shared manner. Most of the scheduling methods consider only dedicated cells. However, dedicated cells can be inefficient in some cases–e.g. a sporadic traffic with high inter-arrival times relative to the average latency constraint.

Dedicated cell assignment can basically be defined as deciding the required bandwidth to each traffic flow or to each neighbor and selecting conflict free cells as required. Scheduling methods given in 2.5 can easily be applied for selecting conflict free cells in the frame, and the required bandwidth is proportional to the packet generation rate, λ. On the other hand, the given bandwidth is proportional to $\frac{1}{frameLength}$ and the number of assigned cells in the frame. Additionally, the period of a periodic traffics or the average inter-arrival time of a sporadic traffic can be expressed as $\frac{1}{\lambda}$, where λ is the packet generation rate.

In the consideration of the ideal case, the easiest solution is building a TSCH frame with $frameLength = \frac{1}{\lambda}$ and selecting one cell in the frame. With this way, the given bandwidth becomes equal to the required bandwidth and 100% energy efficiency can be achieved, which means that every assigned cell is used for packet transmission without idle listening. Energy efficiency can be defined as the ratio of consumed en-

ergy for packet transmissions to total consumed energy. Additionally, idle listening consumes half of the energy that is consumed for packet transmission since transmiting and listening processes consume almost the same amount of energy. Thus, energy efficiency drops when more than the required bandwidth is given. For example, if $\frac{1}{2\lambda}$ is selected as *frameLength*, where λ is the packet generation rate, only half of the assigned cells are used for packet transmission, which means energy efficiency is 66.66%. However, to decrease the average latency, the given bandwidth should be increased since generated packets wait until an assigned cell. In fact, minimum latency can be achieved with 100% energy efficiency if packets are generated at fixed times, which is the case for a periodic traffic. In other words, when the nearest cell to the packet generation is selected, minimum latency with 100% energy efficiency can be achieved. Some of the studies mentioned in 2.5 focus on finding the nearest cell placement. On the contrary, a sporadic traffic generates packets at random times and the packet generation times are independent from each other. For this reason, the average latency is $\frac{frameLength}{2}$ in the case that there is no packet loss, since the average latency is independent from the location of selected cell in the frame. Examples of the two types of traffic and sample schedules for them are given in Figure 3.1 as an illustration. Rectangular boxes and arrows indicate the assigned cells and the packet generations respectively. Red and blue colors are for sporadic and periodic traffics respectively. Packet generation rates are the same and the rate is 0.1 packet per second. Although the latency is constant and equals to the lenght of a slot for the periodic traffic, the average latency is 2.5 seconds for the sporadic traffic since *frameLength* is 5 seconds. Besides, the energy efficiencies can be simply expressed as 100% for the periodic traffic and 66.66% for the sporadic traffic. As a result, although minimum latency can be achieved with 100% energy efficiency for a periodic traffic, there is a trade-off between energy efficiency and the average latency for a sporadic traffic.

One cannot increase energy efficiency without sacrificing from the average latency for sporadic traffics. However, many applications used in the industry have constraint on the average latency. That is why finding a solution, which is energy efficient and meets the average latency constraint, is an important research topic. For that reason, quantizing the trade-off is essential. Since only transmit and listen events consume a significant amount of energy, energy efficiency (η) can be expressed as the ratio of

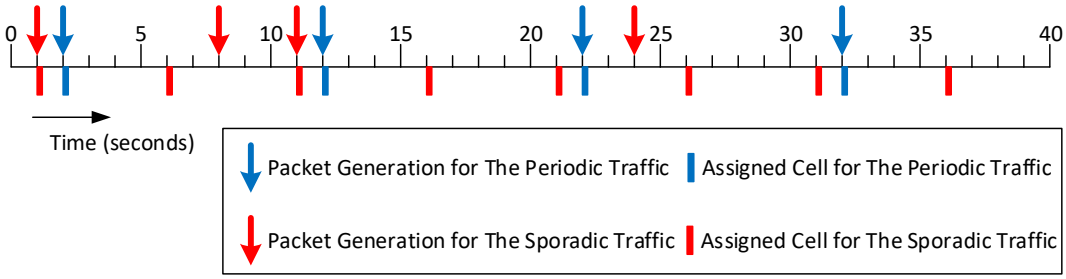


Figure 3.1: Example Dedicated Cell Scheduling

the number of events that result in successful packet transmission to the number of all events. Transmit events includes received packets and collided packets wherase listen events includes all assigned cells. Considering one transmitter node and one receiver node in the ideal environment that there is no packet collision, λ transmit and $\frac{1}{frameLength}$ listen events happen within one second, where λ and *frameLength* are expressed in terms of packet per second and seconds respectively. Additionally, among listen events only λ of them result in successful packet transmission and the others are idle listening. Therefore, energy efficiency (η) is calculated as follows:

$$\eta = \frac{2\lambda}{\lambda + \frac{1}{frameLength}} \tag{3.1}$$

Also, the average latency (L) and the average inter-arrival time (T) for the traffic is assessed as follows:

$$L = \frac{frameLength}{2} \tag{3.2}$$

$$T = \frac{1}{\lambda} \tag{3.3}$$

If the ratio of the average latency to the average inter-arrival time is defined as L', it becomes equal to $\frac{L}{T}$. In fact, it is the average latency in term of the the average inter-arrival time and the upper bound is 0.5 since *frameLength* should be smaller than T. Otherwise, giving less than required bandwidth causes packet loss and this is

inapplicable for industrial networks due to the importance of reliability. Then, η can be written in term of L' as follows:

$$\eta = \frac{4L'}{2L' + 1} \tag{3.4}$$

According to Equation 3.4, the relationship between η and L' is shown in Figure 3.2. It implies the trade-off between energy efficiency and the average latency for a sporadic traffic in the ideal case. According to the figure, if the system allows more delay, higher energy efficiency can be achieved. For example, if an application has a sporadic traffic with $T = 6$ seconds and its average latency constraint is 1 second, energy efficiency can be 50% at most since L' should be smaller then $\frac{1}{6}$. However, many applications require very low average latency values such as 200 microseconds. In this case, L' should be reduced to very low values, which are smaller then $\frac{1}{30} \approx 0.03$ according to the example. This situation results low energy efficiency. As a consequence, assigning dedicated cells to sporadic traffics with low latency constraint relative to the average inter-arrival time is an inefficient solution and also inapplicable since energy consumption is a major concern in Industrial IoT networks.

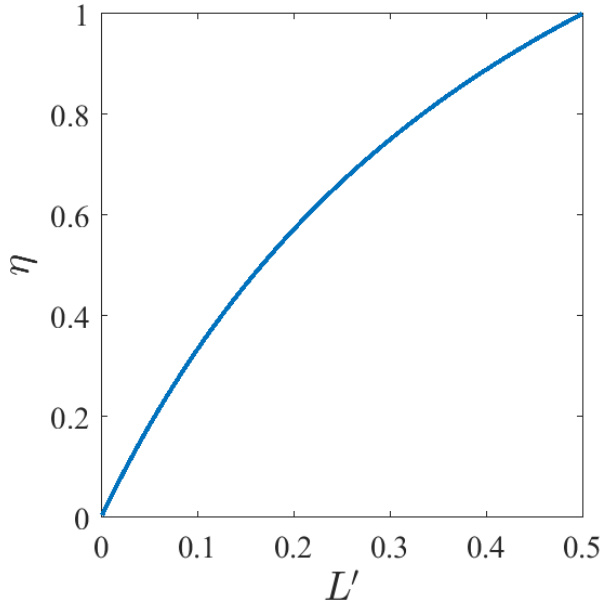


Figure 3.2: The Trade-off Between Energy Efficiency and The Latency

IEEE 802.15.4e TSCH also allows cells to be scheduled in a shared manner. Although dedicated cell assignment is inefficient for sporadic traffics because of a large amount of wasted energy, shared cells are more useful. Moreover, an IoT network is in the tree form due to RPL. Hence, packets are sent from a number of transmitter units–i.e. child nodes, to a receiver unit–i.e. parent node. Using shared cells is also more appropriate for that kind of common receiver networks. Actually, assigning one cell to more than one transmitter node can be thought as decreasing T which causes increase in L'. However, it creates different challenges. Shared cells are open to contention, unlike dedicated cells, hence if more than one transmitter tries to send a packet, collision occurs which causes packet loss and waste of energy. In the ideal case, first-in first-out (FIFO) principle can be applied. It means that if there is a contention, the earliest generated packet have a right to use shared cell. However, this concept cannot be applied in real applications due to the fact that each node can only be aware of its own queue. Nevertheless, FIFO principle can be used to determine the limits. If the same shared cell is assigned to $N_{\text{contender}}$ nodes and each one has a sporadic traffic with T, it can be modeled as one node that the average inter-arrival time is $\frac{T}{N_{\text{contender}}}$. Thus, L' can be increased by adding more contender to the shared cell. However, it has drawbacks since FIFO principle is unfeasible. In the real case, adding more contender causes exponential rise in the number of dropped packets. For those reasons, various contention management methods exist for wireless networks. However, there is a lack of research for shared cells in IEEE 802.15.4e TSCH standard.

In this thesis, the aim is to provide energy efficient shared cell scheduling for sporadic traffics with low latency constraint relative to average inter-arrival time. Shared cell scheduling has pros and cons according to the allowed contention level. If the contention level is high, QoS constraints–e.g. latency and reliability, may not be fulfilled. Otherwise, power consumption becomes very high. Therefore, it is essential to have an optimum shared cell scheduling, in which QoS criteria is met with minimum power consumption. To this end, a method is introduced in this chapter to obtain the optimum schedule. The method treats IoT networks as subnetworks, which consist of one parent and a number of child nodes, and then it tries to optimize each subnetwork independently. In order to achieve the optimum schedule, each parent node should adjust the network parameters, which are *frameLength*, the number of shared cells in

the frame and the number of contender nodes for each cell.

The rest of this chapter is as follows. In Section 3.2, an additional information about dedicated and shared cells is given. In Section 3.3, the network model and estimation of the QoS metrics are described. The method is introduced in Section 3.4.

3.2 Dedicated and Shared Cells

As stated in Section 3.1, there are two types of cell in TSCH standard. Dedicated cells are contention-free, that is to say, exactly one transmission occurrence is guaranteed in these cells. Due to the fact that the most of IoT networks consist of a high number of nodes, dedicated cell scheduling requires a long TSCH frame, in which each node solitarily transmits to its parent node. However, when the traffic is sporadic with low latency constraint relative to average inter-arrival time, bandwidth and energy are wasted in those networks. Therefore, it is better to use shared cell approach rather than dedicated one. In shared cell approach, there is more than one node that attempts to transmit in the cell, which is likely to have contention. After contention resolution, one transmitter node delivers its packet to its receiver. In [14], shared and dedicated cell scheduling methods are compared under a sporadic traffic. The shared cell scheduling method in [14] is that each parent node selects a different cell and then selected one cell is assigned to all of its child nodes whereas the dedicated cell scheduling method is that a parent node assigns a different cell to each of its child nodes. It is also reported that the shared cell scheduling is better than the dedicated cell scheduling. However, proposed shared cell scheduling method in [14] is not an adaptive solution.

To cope with the contention in a shared cell, CSMA-CA algorithm is included to IEEE 802.15.4e TSCH standard. This algorithm is only activated for shared cells. Although the aim is the same, there are differences between traditional CSMA-CA and TSCH CSMA-CA. The common aim is to reduce collision probability in a shared cell. To do that, back-off mechanism is used by both of the algorithms. The mechanism is broadly waiting a random amount of time before trying to send a packet, and if collision occur, the waiting time increases. However, back-off mechanism works

differently in traditional and TSCH CSMA-CA algorithms. Although, newly generated packets can wait to be transmitted in traditional one, TSCH CSMA-CA activates back-off mechanism only if collision has been experienced. Additionally, the mechanism in TSCH CSMA-CA only counts shared cells as back-off waiting time instead of actual time. Clear Channel Assessment (CCA) is another mechanism used by traditional CSMA-CA. The objective is to monitor the medium before trying to send a packet to avoid collision with ongoing transmission process. However, slotted structure does not make CCA possible for TSCH CSMA-CA. Both of the algorithms drop packet if contention level is high. Traditional CSMA-CA decides to drop the packet when a certain number of consecutive sending attempts result in busy medium by CCA mechanism. Since TSCH CSMA-CA cannot use CCA mechanism, it counts retransmissions and if it reaches a certain value, the algorithm decides to drop the packet.

It is clear that CSMA-CA algorithm should be analyzed and modeled in order to achieve optimum shared cell schedule. The majority of traditional CSMA-CA modeling techniques try to model the behavior of a single node in the network through Discrete Time Markov Chain. Park's model [39] is well-known among them. The model provides the ability to set reliability, latency and consumed energy per packet with CSMA-CA parameters. However, it is based on many unrealistic assumptions such as stable and known topology. Additionally, finding optimum parameters with Park's model requires a significant amount of time. In [38], Park *et al.* modify the model and introduce an algorithm to estimate channel busy probability and channel access probability. With those estimations, proposed algorithm tries to minimize energy consumption while guaranteeing reliability and latency constraints. Unlike Park's model, the behavior of an entire network is modeled in [19]. The model is designed to capture even the most unusual events and to make it possible to compute precise probabilities. Inspiring from [19], Guglielmo *et al.* [8] create a model for TSCH CSMA-CA. They perform an algorithm based computation that takes into consideration every possible state that the system can be in. After the exhaustive computation, a probability transition matrix is constructed and the performance metrics can be calculated by using this matrix. However, new packet arrival issue is not considered in [8]. That is to say, it is modeled that the network starts from the state

that nodes have their head-of-line (HOL) packets to send, and ends when all the HOL packets are successfully delivered or dropped. This approach is not realistic due to the fact that collision probability and waiting time of a packet are directly related to the new packet arrivals.

Although analytic model based algorithms provide the most precise results, they are not suitable for real-life scenarios since the analytic models create a significant amount of overhead and they need excessive computational power. For those reasons, a low complexity and distributed algorithm, called ADAPT, is proposed in [10]. It is intended to ZigBee sensor networks and suitable for real-life scenarios since it does not require any knowledge about operating conditions. ADAPT is based on local measurements and provides minimum energy consumption without violating reliability and latency constraints. The main purpose is to control congestion by adjusting CSMA-CA parameters. However, it has oscillating response which results in unnecessary energy consumption. Another algorithm based on local measurements is LEAP [6]. Like ADAPT, LEAP also sets CSMA-CA parameters to find the most energy efficient state without violating QoS constraints. On the other hand, LEAP learns the channel condition with the history instead of ADAPT which only considers current measurements. LEAP creates a table according to learned conditions. Then, it finds optimum CSMA-CA parameters from the table.

In [34], a model and an algorithm based on this model is proposed. Proposed algorithm creates a schedule to decrease energy efficiency with meeting QoS requirements. However, the algorithm checks all possible configuration hence its complexity is high and it is designed for networks that all nodes have the same amount of traffic loads.

As a result, algorithms based on an analytic model are not suitable for real-life scenarios, unlike heuristic methods like ADAPT and LEAP. Although traditional CSMA-CA is well studied, there are very few studies about TSCH CSMA-CA within the current knowledge. Moreover, all of the algorithms try to find optimum CSMA-CA parameters. Based on those results, the aim of this chapter is to design a distributed and low complexity algorithm that based on a heuristic model in order to find optimum shared cell scheduling.

3.3 Model

An Industrial IoT network is composed of multiple subnetworks, which are connected together since a child node can send packets only to its parent node due to the tree structure of RPL standard. In other words, each parent node and its child nodes create a subnetwork. The parent in a subnetwork can be a child node in another subnetwork. With this way, subnetworks can connect to each other. For instance, the network shown in Figure 3.3 consists of 3 subnetworks. Each subnetwork is highlighted by a dashed circle and labeled with a letter. As an example, subnetwork A in Figure 3.3 has 3 nodes which are node 1 as parent node, nodes 2 and 3 as child nodes. Moreover, subnetwork A is connected to subnetwork B via node 2 and to subnetwork C via node 3. In a subnetwork, shared cells can be used to handle sporadic traffics since child nodes in a subnetwork can send packets only to the parent node. Although subnetworks are connected to each other and there is an interference between them, these subnetworks can be thought as independent from each other if shared cells are selected with respect to following rules:

1. Connected subnetworks must use different *slotOffset* values.

2. Unconnected and geographically close subnetworks must use different *channelOffset* values.

The first rule is for avoiding duplex conflict by exploiting time synchronization. The second one is for preventing interference conflict with the help of channel hopping mechanism defined in TSCH. According to example in Figure 3.3. Subnetwork A and subnetwork B can be separated from each other by using different *slotOffset* values. On the other hand, although subnetwork B and subnetwork C are not directly connected to each other, they may suffer from interference conflict since node 3 is very close to node 2. The second rule can prevent this problem. In fact, selecting different *slotOffset* values also solves interference conflict problem. However, the second rule is a better solution since it increase cell reuse and available cells for a subnetwork. Nevertheless, using different *slotOffset* values may be a better solution in the case that the available channels are very limited.

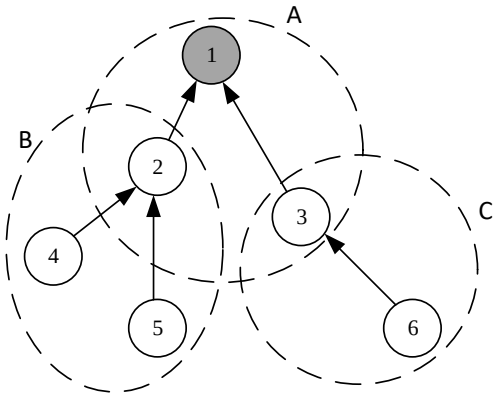


Figure 3.3: 6P Request Message Format

The network performance depends on the subnetworks, which are independent from each other. Thus, modeling a subnetwork and optimizing each subnetwork separately based on this model converges to optimum solution for an Industrial IoT network. Additionally, optimizing each subnetwork separately provides a better solution instead of considering the network as a whole since geographically separated subnetworks can experience different physical layer conditions. Therefore, a model for a subnetwork in an Industrial IoT network is introduced in this section. The model is not an analytic one due to the mentioned reasons in Section 3.2 and it is not for estimating the effects of TSCH CSMA-CA parameters since it is well studied. The model is based on heuristic observations for estimating the effects of the number of child nodes, frequency of assigned shared cells and distribution of these child nodes among the shared cells.

A shared cell assigned to links that are directed from a number of child nodes to the parent node in a subnetwork can be modeled as a bus system. The system consists of 1 parent node and n_{c} child nodes, like in Figure 3.4. Each child node has a first-in-first-out (FIFO) queue, where packet arrivals are independent from each other. The earliest incoming packet occupies HOL area until it is successfully transmited or dropped. If HOL area is not empty, incoming packets wait in packet buffer area for their turns. This area is limited, which means that only *bufferLength* packets can be in this area. Reaching the limit causes loss of further incoming packets. A child node tries to send its HOL packet only in the assigned shared cells. In the case of collision, exponen-

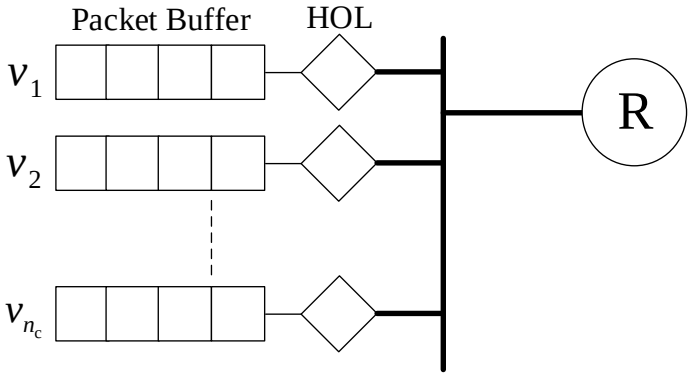


Figure 3.4: Bus Model for Shared Cell

tial back-off mechanism is applied. TSCH CSMA-CA algorithm defines a back-off mechanism as stated in Figure 3.5. The flow chart in the figure is created with respect to the specifications in [43]. New incoming packets are firstly tried to be sent without back-off. If collision occurs, exponential back-off mechanism is triggered. Then the packet waits for a random time. This time is determined with respect to frequency of the shared cells and a random number $w \in [0, (2^{BE} - 1)]$ such that the node waits until w shared cells passes. *BE* is a TSCH CSMA-CA parameter, named back-off exponent, and it adjusts the upper limit of the waiting time. After the waiting, the node tries to send HOL packet again. If collision occurs again, *BE* is increased by 1 and the packet returns to waiting process with new w. *BE* starts from *macMinBE* parameter and can be increased until it reaches the limit that is indicated by *macMaxBE* parameter. If the limit is reached, the node continues to use *macMaxBE* as *BE*. Decision to failure depends on *macMaxFramesRetries* parameter. When the number of back-offs (*NB*) reaches to *macMaxFramesRetries*, the node decides to drop HOL packet. *NB* starts from zero with first retransmission and increased by 1 when collusion occurs. 3.1. One of the important point of this mechanism is that if there is another packet in the buffer when the HOL area becomes empty, *BE* remains to have the same value since there can be still congestion. Otherwise, *BE* is set to *macMinBE*. In the standard definition, these parameters can take integer values within defined ranges. The ranges and default values for the parameters are given in Table 3.1.

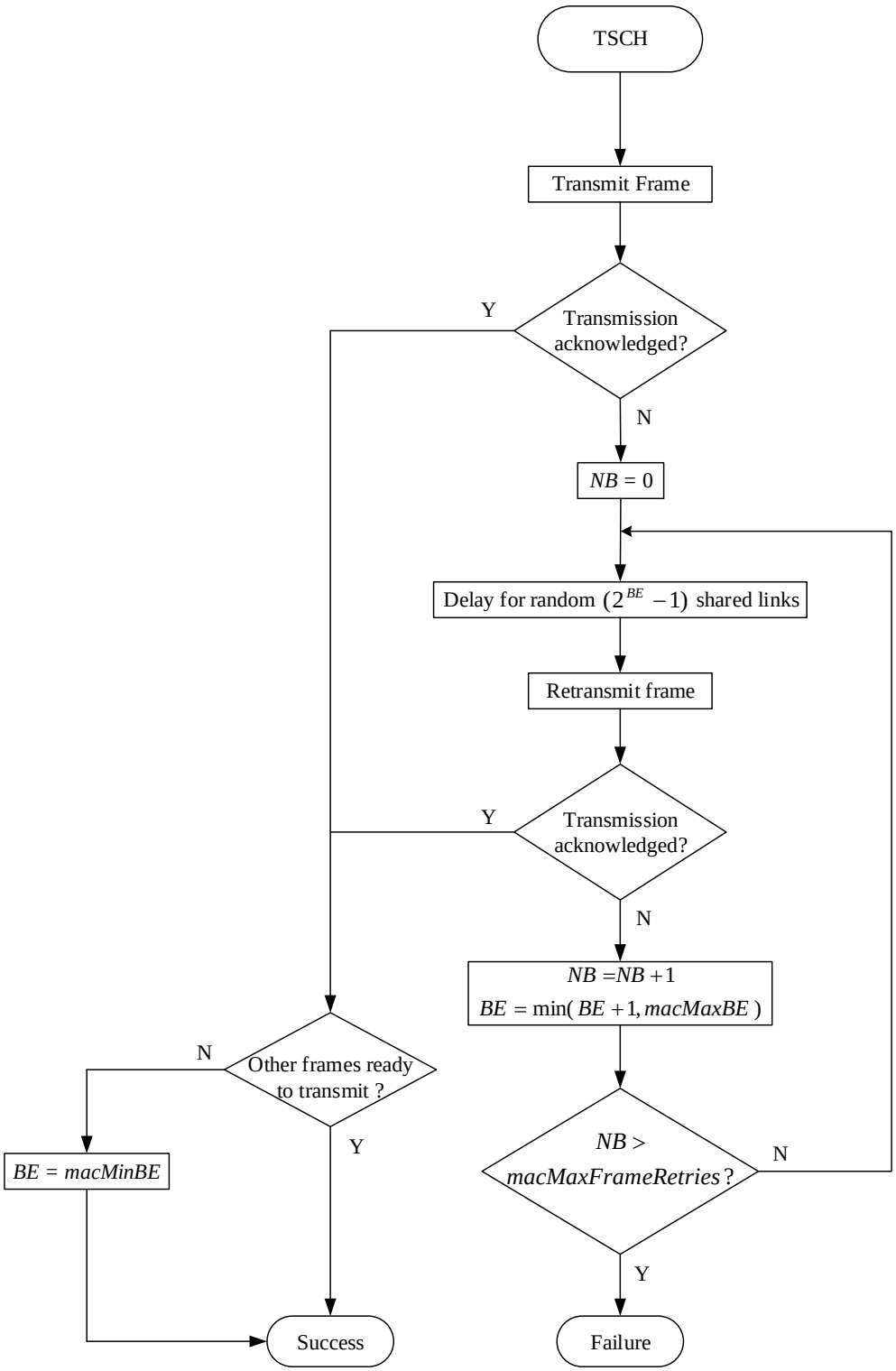


Figure 3.5: TSCH CSMA-CA Retransmission Algorithm

Table 3.1: Parameter Ranges and Default Values

Parameter	Range	Default
macMinBE	0-*macMaxBE*	1
macMaxBE	3-8	7
macMAxFrameRetries	0-7	3

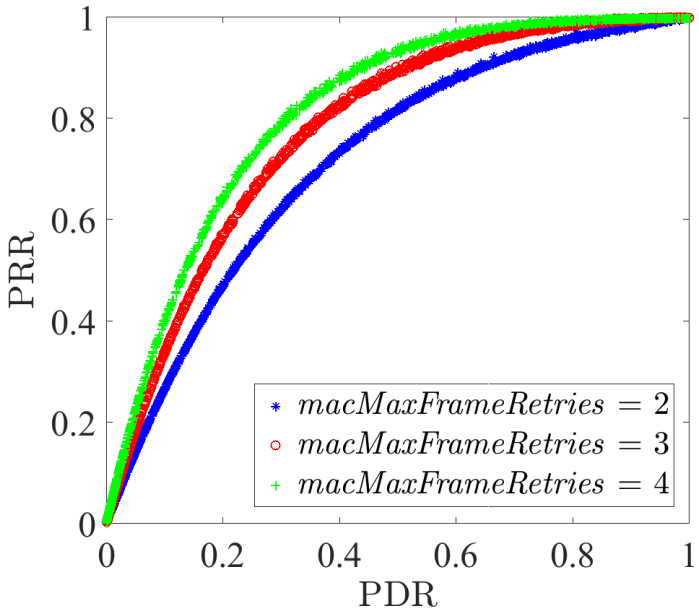
For a sporadic traffic, packet generation is independent from time. However, average packet arrival in a certain time can be known. Thus, each queue randomly generates packets with respect to Poisson distribution. Although it seems to create an analytic model for the system, it is highly complex to model mathematically. High-complexity models are not useful for adaptive algorithms as stated in Section 3.2. To that reason, computer-based simulations are conducted to understand the characteristic of a sub-network. The aim is to estimate the effects of the parameters on performance metrics. The parameters of the model are the number of child nodes, n_{c}, and frequency of the shared cell. The metrics are the average latency and packet delivery ratio (PDR), which refers to the ratio of received packet count to the total trial count to send these packets. In fact, other important performance metrics can be computed with PDR like reliability, which is directly related to PDR. Besides, average consumed energy to successfully send one packet can be calculated by using PDR. Additionally, there are some parameters that cannot be changed to increase the performance such as the average inter-arrival time for a sporadic traffic. On the other hand, TSCH CSMA-CA parameters are not interested in this thesis since there are algorithms like LEAP [6] that focuses on these parameters.

A set of computer-based simulation results are collected in order to clear relationship between the parameters and performance metrics. The simulations are continued long enough to make resultants converge to their average values. In those simulations, it is assumed that one shared cell is assigned to all child nodes in a subnetwork. It means that shared cell frequency is determined by *frameLength* parameter. The competing child count, n_{c}, is changed in range of 1 to 25, while *frameLength* can be in the range of 1 to 50. Therefore, 1250 different configurations come out for the simulations. Apart from objective parameters, the effect of TSCH CSMA-CA parameters are also investigated. On behalf of simplification, $\{2, 3, 4\}$ and $\{3, 7\}$ values are considered for *macMaxFramesRetries* and *macMaxBE* parameters respectively although only the

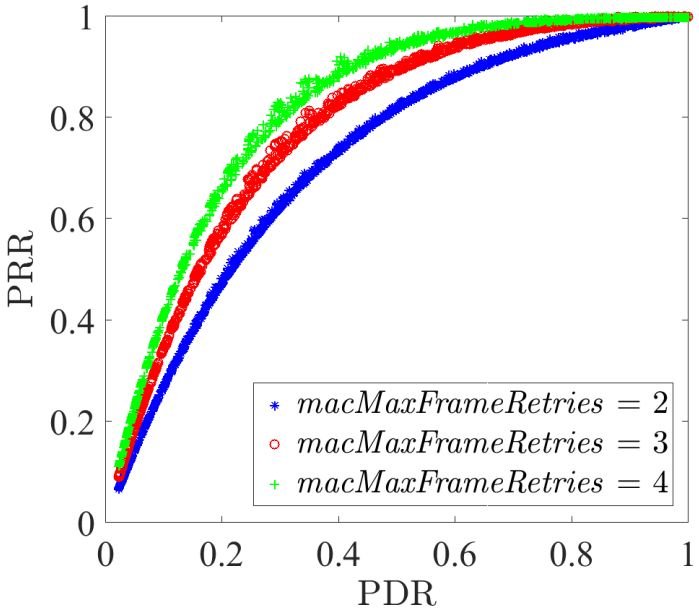
default value is allowed for *macMinBE* since other values are not meaningful. Furthermore, the queues are assumed to be infinite due to the fact that interested traffic type is sporadic with low latency constraint relative to the average inter-arrival time. It means that packet drops due to buffer overflow does not seem possible. The length of a slot is assumed to be 10 milliseconds as stated in the standard. Since the traffic load is very related to *frameLength*, there is no need to investigate them separately. Thus, the average inter-arrival time is fixed for computer-based simulations. The value is 8 seconds and same for all child nodes.

First of all, the relationship between PDR and reliability is examined. Reliability metric is measured as packet reception ratio (PRR), which is the ratio of received packet count to generated packet count. The relationship is shown in Figure 3.6. In the graphs, each configuration is represented by a marker. For *macMaxFramesRetries* parameters, different symbols and colors are used as a marker. The values 2, 3 and 4 are symbolized as '*', 'o' and '+', and colored as blue, red and green respectively. Figure 3.6a shows results when *macMaxBE* parameter is 3, whereas it is 7 in Figure 3.6b.

According to Figure 3.6, it can be stated that even if the PDR depends on the parameters, PRR is directly associated with PDR. Since PRR is the measure of reliability, it should be high enough to meet Industrial IoT service requirements as mentioned in Chapter 2. As a result, PDR should always be kept within a range. For instance, according to Figure 3.6a, it can be seen that in the case of PRR > 0.95, PDR must be greater than 0.77, 0.65 and 0.55 for *macMaxFrameRetries* is 2, 3 and 4 respectively. The values are almost the same for Figure 3.6b. In fact, the reliability constraint defines a region of interest (ROI) on PDR. From now on, the ROI means PRR > 0.95. The relationship between PRR and PDR in the ROI is shown in Figure 3.7. Figures 3.6a and 3.6b show the relationship with different different *macMaxBE* values. According to the figures, the results are very similar to each other hence it can be said that the relationship relationship between PRR and PDR is independent from *macMaxBE* parameter. From now on, *macMaxBE* is fixed to 7, which is the default value.

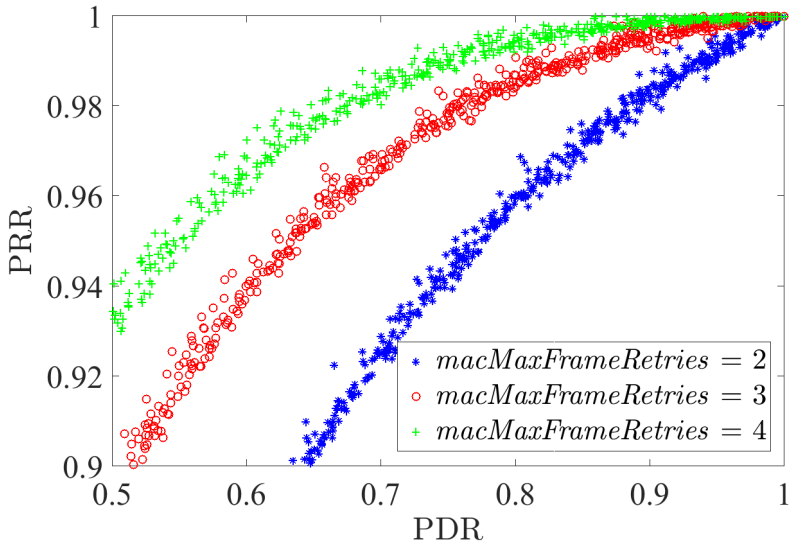


(a) $macMaxBE = 3$

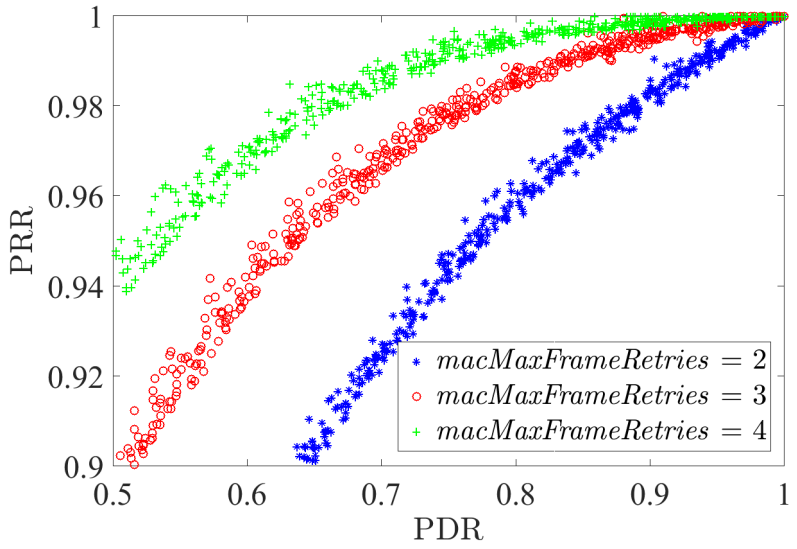


(b) $macMaxBE = 7$

Figure 3.6: The Relationship Between PRR and PDR.



(a) *macMaxBE* = 3



(b) *macMaxBE* = 7

Figure 3.7: The Correlation Between PRR and PDR in the ROI.

PDR can only be controlled by adjusting the adaptable parameters, which are n_c and *frameLength*, since TSCH CSMA-CA parameters like *macMaxFrameRetries* and *macMaxBE* considered as constant in this study. Both the increase and decrease in PDR has its own benefits and drawbacks. If PDR increases, not only PRR increases, but also the latency metric decreases; however, consumed energy significantly increases. Energy consumption metric can be measured by counting transmit and listen events since the major energy consumption in wireless devices occurs while transmitting a packet and listening the channel. Other energy-consuming events can be neglected. Additionally, it can be assumed that idle listening, transmitting and receiving a packet consume the same amount of energy, which are in fact very close. Therefore, the total consumed energy, E_{total}, within the simulation time, T_{sim}, can be expressed as follows:

$$E_{\text{total}} = Q_{\text{listen}} + Q_{\text{transmit}} \tag{3.5}$$

Q is the number of events within the simulation time. While calculating Q_{listen}, each listen event, which can result in successful packet transmission or idle listening, is counting. Similarly, both successful and unsuccessful transmission attempts are included in Q_{transmit}. Q_{listen} only depends on *frameLength* parameter since a parent node has to listen the channel whether there is a sending packet or not. Whereas, throughput and PDR affect Q_{transmit}. Thus, Equation 3.5 can be modified as follows:

$$E_{\text{total}} = \frac{T_{\text{sim}}}{\textit{frameLength}} + \frac{\mu T_{\text{sim}}}{m_{\text{PDR}}} \tag{3.6}$$

In Equation 3.6, m_{PDR} refers to a measured PDR value of the subnetwork for certain configuration and it depends on n_c and *frameLength* parameters. μ is the throughput and it is calculated as average received packet count in one second. If the power is defined as average consumed energy in a subnetwork to receive one packet successfully, it is calculated as follows;

$$P_{\text{packet}} = \frac{1}{\mu_{\text{frame}}} + \frac{1}{m_{\text{PDR}}} \tag{3.7}$$

where P_{packet} is the power and μ_{frame} is average received packet count in one frame. Additonally, $\frac{1}{m_{\text{PDR}}}$ and $\frac{1}{\mu_{\text{frame}}}$ terms in the equation represent the average number of transmit and listen events for one successful packet transmission respectively.

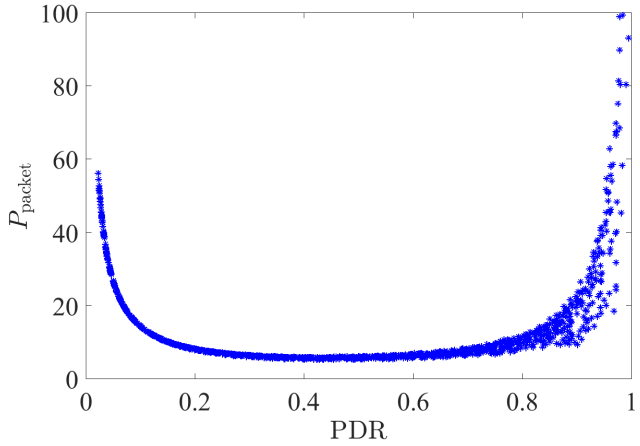
According to Equation 3.7, the relationship between the power and PDR is shown in Figure 3.8. Each P_{packet} and PDR pair, which is a result of the simulations for one configuration, is represented as a marker. Figures 3.8a, 3.8b and 3.8c correspond to different *macMaxFrameRetries* values which are 2, 3 and 4 respectively. According to the figures, P_{packet} increases exponentially with increasing PDR in the ROI for all values of *macMaxFrameRetries*. This situation is not desirable for Industrial IoT networks, in which energy efficiency is crucial as mentioned in Section 2. Therefore, PDR should be set to a value such that latency and reliability constraints are fulfilled with minimum energy consumption. Since parent nodes can control PDR by adjusting n_{c} and *frameLength* parameters, an appropriate system condition can be reached even after the network startup.

Figure 3.9 shows PDR with different n_{c} and *frameLength* values. According to Figures 3.9a, 3.9b and 3.9c, a similar trend occurs between PDR and n_{c} for all *macMaxFrameRetries* values. Not surprisingly, there is a negative correlation between them, since increasing number of the competing child nodes can lead to more collisions and thus lower PDR. Furthermore, *frameLength* value determines the attenuation of this correlation. When the ROI parts are further investigated, which are given in Figures 3.10, it can be seen that there is an almost linear relationship between PDR and n_{c} for all *frameLength* values.

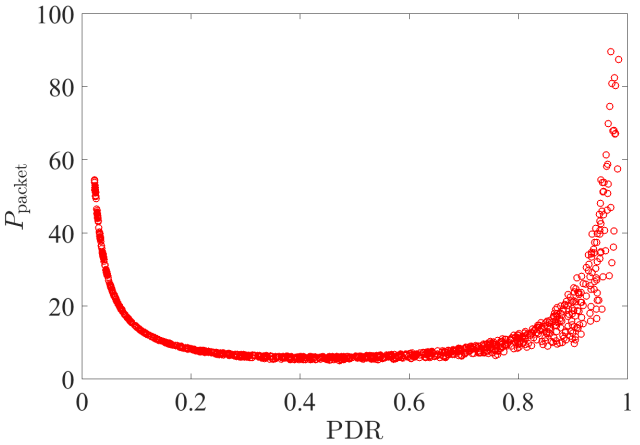
Recalling that it is desired to keep PDR in the ROI, the main focus is to model a subnetwork in this region. Therefore, PDR can be modeled in the ROI with a linear function of n_{c} and *frameLength* parameters. After computations, concluded $P_{\text{pdr}}(n_{\text{c}}, \textit{frameLength})$, which refers to estimated PDR with stated parameters, is expressed as follows:

$$P_{\text{pdr}}(n_{\text{c}}, \textit{frameLength}) = 1 - K_{\text{PDR}}\lambda_{\text{frame}}\big[n_{\text{c}} - 1\big] \tag{3.8}$$

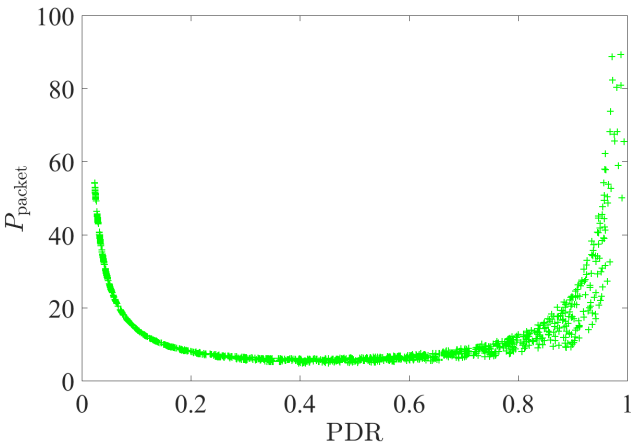
where $\lambda_{\text{frame}} = \lambda \times \textit{frameLength}$ and λ is the packet generation rate for a child node.



(a) *macMaxFrameRetries* $= 2$

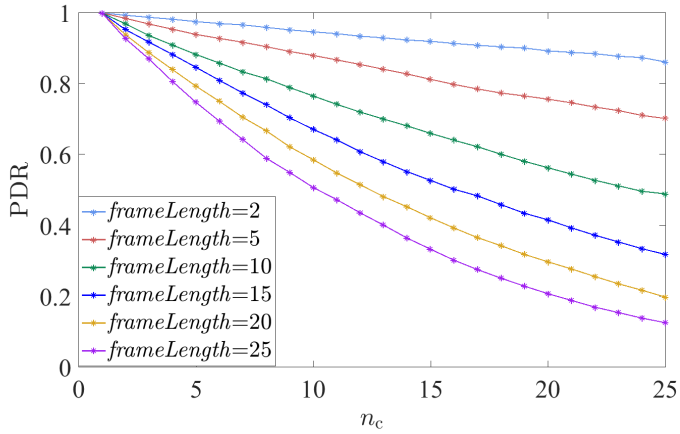


(b) *macMaxFrameRetries* $= 3$

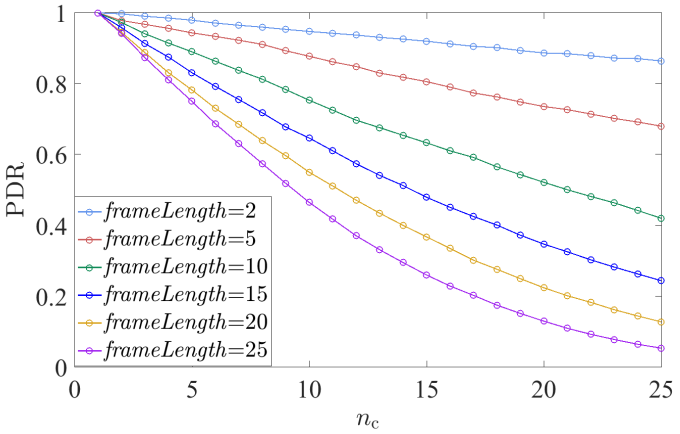


(c) *macMaxFrameRetries* $= 4$

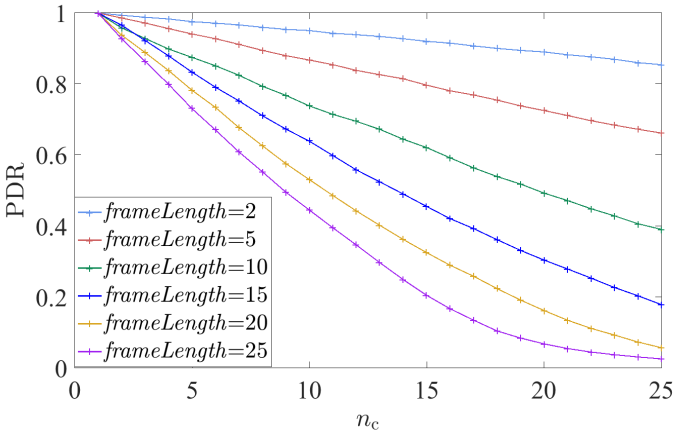
Figure 3.8: The Relationship Between P_{packet} and PDR.



(a) *macMaxFrameRetries* $= 2$

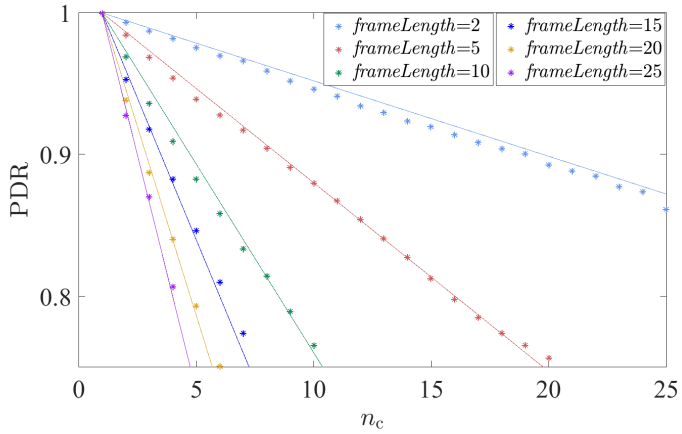


(b) *macMaxFrameRetries* $= 3$

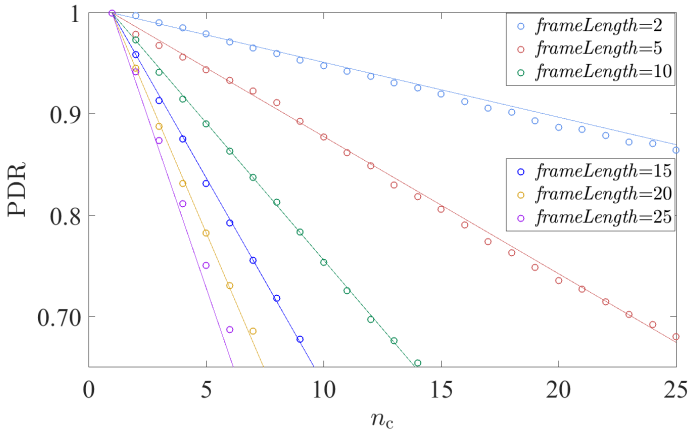


(c) *macMaxFrameRetries* $= 4$

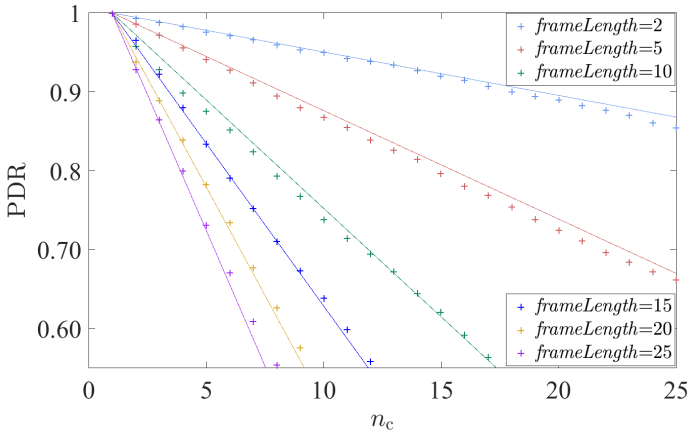
Figure 3.9: The Relationship Between PDR and Adjustable Parameters.



(a) *macMaxFrameRetries* $= 2$



(b) *macMaxFrameRetries* $= 3$



(c) *macMaxFrameRetries* $= 4$

Figure 3.10: The Relationship Between PDR and Adjustable Parameters in the ROI.

Table 3.2: K_{PDR} for different *macMaxFrameRetries* and *macMaxBE* Parameters

macMaxBE	***macMaxFrameRetries***		
	$= 2$	$= 3$	$= 4$
$= 3$	2.13	2.18	2.23
$= 7$	2.13	2.17	2.20

Thus, λ_{frame} corresponds to the average generated packet count in one frame. K_{PDR} term in Equation 3.8 is a coefficient and independent from n_{c} and *frameLength*. In fact, K_{PDR} is not only dependent on pre-defined network parameters like *macMaxBE* and *macMaxFrameRetries*, but also can be affected by several unpredictable factors like intensity of the noise in the wireless medium. Thus, computing K_{PDR} is more appropriate than assigning a fixed value. Table 3.2 shows the most suitable K_{PDR} values for different *macMaxFrameRetries* and *macMaxBE* parameters. Moreover, the linear lines in Figures 3.10a, 3.10b and 3.10c are drawn with respect to Equation 3.8 and K_{PDR} values in the table.

On the other hand, latency is another critical metric for IoT networks. As stated in Section 2, there is an average latency constraint in most of the applications. Hence, latency must also be estimated with the model. Figure 3.11 illustrates the relationship between the latency metric and n_{c}. During the computations, it is assumed that the length of one slot, $slotTime$, is 10 milliseconds in compliance with the standard. From the graphs in Figures 3.11a, 3.11b and 3.11c, it can be seen that d, which is a latency metric and refers to the average latency in term of milliseconds, increases with n_{c} for all *macMaxFrameRetries* values. Obviously, a main factor that affects d is *frameLength*, which extends the waiting time for packet transmission. On the other hand, d is also dependent on PDR. In order to have a better understanding of the correlation between PDR and latency, d_{frame} metric is defined. It is the average latency in term of *frameLength*. Therefore, the correlation can be investigated independently of *frameLength*. In Figure 3.12, the correlation between PDR and d_{frame} is examined in the ROI and it is shown that the correlation is almost independent from n_c and *frameLength*. This figure clearly indicates that d_{frame} exponentially decreases with

Table 3.3: K_{d} for different *macMaxFrameRetries* and *macMaxBE* Parameters

macMaxBE	***macMaxFrameRetries***		
	$= 2$	$= 3$	$= 4$
$= 3$	1.82	2.68	2.91
$= 7$	1.83	2.71	3.14

PDR. Based on the computations, we can formulate this relationship as follows:

$$d_{\text{frame}}(P_{\text{pdr}}) = \frac{1}{2} e^{K_{\mathrm{d}}\ (1-P_{\text{pdr}})} \tag{3.9}$$

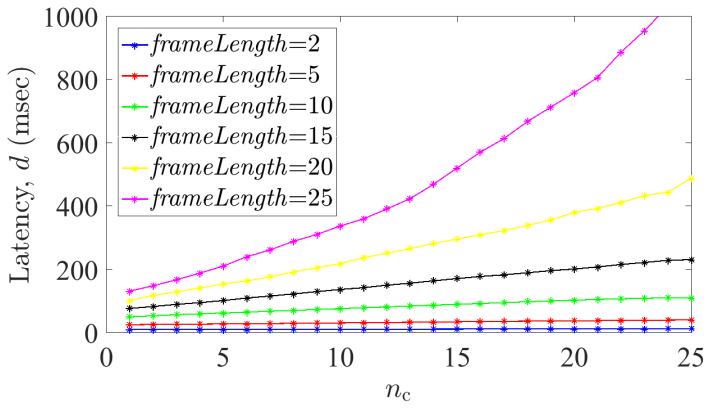
In Equation 3.9, K_{d} is a parameter, which is computed by the nodes due to the same reasons stated in K_{PDR} discussion. For instance, computed K_{d} values for different *macMaxFrameRetries* and *macMaxBE* can be seen in Table 3.3. Furthermore, exponential graphs of the estimated d_{frame} by using Equation 3.9 are drawn Figure 3.12. Besides, we can also compute d as shown in Equation 3.10.

$$d(P_{\text{pdr}}, \textit{frameLength}) = d_{\text{frame}} \times \textit{frameLength} \times \textit{slotTime} \tag{3.10}$$

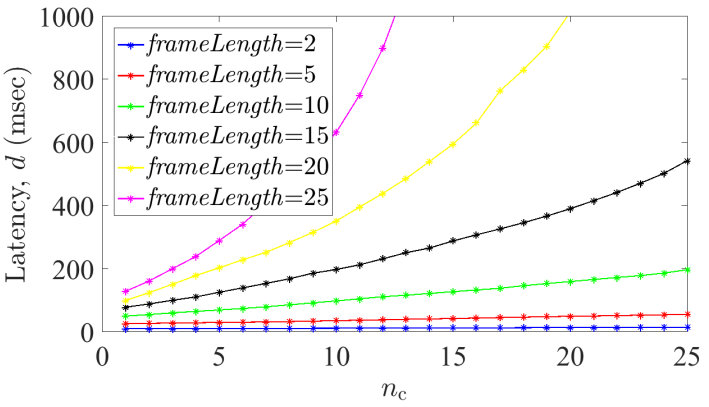
In this section, the equations for estimating PDR, latency and power consumption in terms of n_c and *frameLength* parameters are obtained. Afterwards, a method based on these estimations is proposed in Section 3.4.

3.4 Method

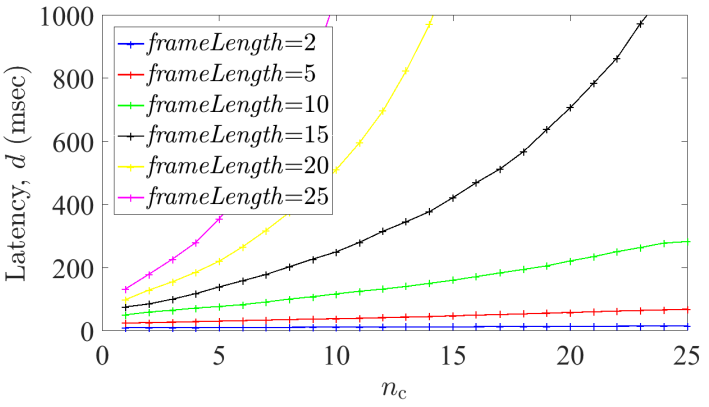
In this section, a method that uses instantaneous network data to arrange the most effective schedule, which meets latency and reliability constraints with minimum energy consumption, is introduced. Proposed model in Section 3.3 can be used for this purpose. The latency and reliability metrics can be estimated and adjusted to meet the constraints by parent node of each subnetwork in a distributed manner. Although the model is for a schedule with one assigned shared cell, it can be extended to a schedule with multiple assigned shared cells by considering each shared cell separately.



(a) *macMaxFrameRetries* $= 2$



(b) *macMaxFrameRetries* $= 3$



(c) *macMaxFrameRetries* $= 4$

Figure 3.11: The Relationship Between Latency and Adjustable Parameters.

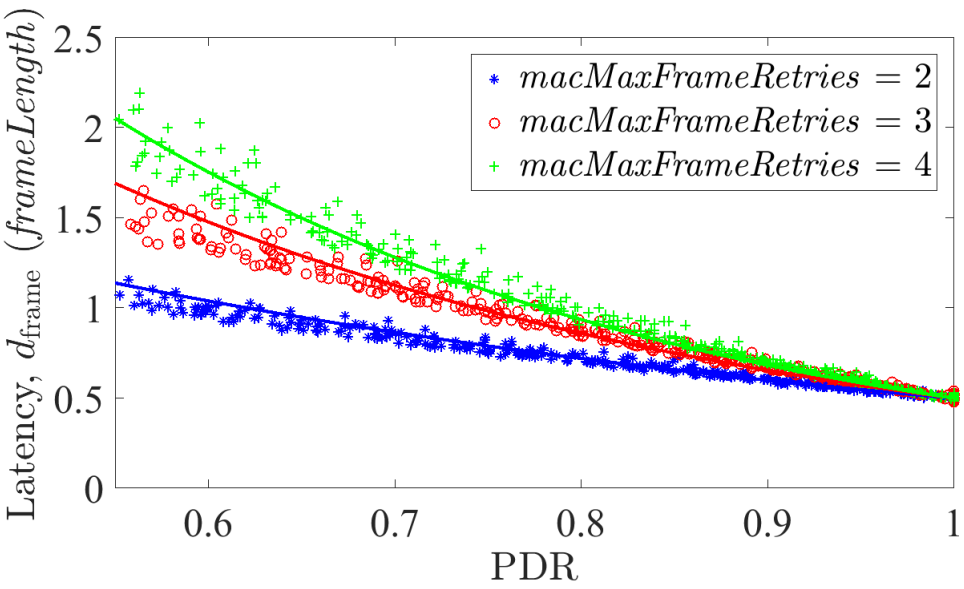


Figure 3.12: The Correlation Between Latency and PDR

In a subnetwork, latency and reliability metrics can be adjusted by changing the number of assigned shared cells, frequency of these cells and distribution of the child nodes among the shared cells. Due to the fact that a trial-and-error method is costly for Industrial IoT networks, those parameters should be adjusted according to an estimation method. Additionally, latency and reliability metrics also depends on the packet generation rates of child nodes and the number of child nodes. However, these constant parameters cannot be changed by a parent node although the values are known. As a result, parent node of a subnetwork can use the model with constant parameters and measurable metrics, such as current PDR and current average latency, in order to adjust the number of assigned shared cells, frequency of these cells and distribution of the child nodes among the shared cells.

The number of assigned shared cells in the frame is an integer parameter and expressed as s and frequency of each cell is determined by *framelength* parameter. Whereas, distribution of the child nodes is specified with a vector, called $\vec{V}$. The v^{th} element of $\vec{V}$ denotes assigned shared cell number to v^{th} child node of the parent. Thus, the size of $\vec{V}$ is the number of all child nodes of the parent and expressed as N_{c}. As a result, The adjustable parameters are s, *framelength* and $\vec{V}$ whereas N_{c} is one of the constant parameters.

The method should distribute child nodes into multiple shared cells in one frame according to introduced model. In fact, shared cells can be thought as subnetworks of the same parent node. Since the model consider each shared cell separately, n_{c}^{i} is defined. It refers to the number of competing child nodes in the i^{th} shared cell, which means the number of elements that equals to i in $\vec{V}$. Moreover, N_{c} can be computed as follows:

$$N_{\text{c}} = \sum_{i=1}^{s} n_{\text{c}}^{i} \tag{3.11}$$

The other constant parameter is the packet generation rates of child nodes. The generated packet rate of v^{th} node is expressed as $\lambda_{\text{frame}}^{v}$ and the packet generation rates of all child nodes can be denoted by a vector, $\vec{\Lambda}_{\text{frame}}$, of which v^{th} element equals to $\lambda_{\text{frame}}^{v}$. Therefore, $\vec{\Lambda}_{\text{frame}}$ and N_{c} are constant parameters.

Although the model is created with an assumption of homogeneous traffic load, which means every child node has same packet generation rate, it can be easily extended to heterogeneous traffic load since the traffic is sporadic with low latency constraint relative to average inter-arrival time. The multiplication of λ_{frame} and n_{c} terms in Equation 3.8 is implies the total packet generation rate of child nodes that assigned to the same shared cell hence the equation can be modified as follows:

$$P_{\text{pdr}}(\theta_{\text{frame}}^{i}) = 1 - K_{\text{PDR}}\theta_{\text{frame}}^{i}\big[1 - \frac{1}{n_{\text{c}}^{i}}\big] \tag{3.12}$$

where $\theta_{\text{frame}}^{i}$ refers to the total packet generation rate of child nodes that assigned to i^{th} shared cell and can be computed as follows:

$$\theta_{\text{frame}}^{i} = \sum_{\substack{v=1 \\ \vec{V}[v]=i}}^{N_{\text{c}}} \lambda_{\text{frame}}^{v} \tag{3.13}$$

A parent node should compute K_{PDR} and K_{d} values for proper estimation. In fact, these values can be different in each shared cell since the computation is based on

the measurements. Thus, a parent node should find approximate values by averaging them as follows:

$$K_{\text{PDR}} = \frac{1}{s}\sum_{i=1}^{s} k^i_{\text{PDR}} \tag{3.14}$$

$$K_{\text{d}} = \frac{1}{s}\sum_{i=1}^{s} k^i_{\text{d}} \tag{3.15}$$

where i refers to the i^{th} shared cell. Additionally, computation of k^i_{PDR} and k^i_{d} values are based on current PDR, m^i_{PDR}, and current average latency, m^i_{d}, measurements in the i^{th} shared cell. Thus, these values can be calculated, by using Equation 3.12 and 3.9, as follows:

$$k^i_{\text{PDR}} = \frac{1 - m^i_{\text{PDR}}}{\theta^i_{\text{frame}}\left[1 - \frac{1}{n^i_{\text{c}}}\right]} \tag{3.16}$$

$$k^i_{\text{d}} = \frac{\ln(2m^i_{\text{d}})}{1 - m^i_{\text{PDR}}} \tag{3.17}$$

Each ($\vec{\Theta}_{\text{frame}}$,*frameLength*) combination represents a state for a parent node, where $\vec{\Theta}_{\text{frame}}$ is traffic distribution vector as $\vec{\Theta}_{\text{frame}} = [\theta^1_{\text{frame}}\ \theta^2_{\text{frame}}\ ...\ \theta^s_{\text{frame}}]$. The size of $\vec{\Theta}_{\text{frame}}$ is shared cell count and each element denotes traffic load to which the cell is assigned.

The power for the i^{th} shared cell can be expressed according to Equation 3.7 and Equation 3.12 as follows:

$$P^i_{\text{packet}} = \frac{1}{\theta^i_{\text{frame}}} + \frac{1}{P_{\text{pdr}}(\theta^i_{\text{frame}})} \tag{3.18}$$

In Equation 3.18, θ^i_{frame} is used as received packet count in i^{th} shared cell although it is the generated packet count. This substitution is meaningful due to high reliability, which means the generated packet count is almost equal to the received packet count.

After the power values for all shared cells are computed, average power (P_{avg}) for a state can be found as follows:

$$P_{\text{avg}}(\vec{\Theta}_{\text{frame}}) = \frac{1}{\Lambda_{\text{frame}}} \sum_{i=1}^{s} \theta^i_{\text{frame}} P^i_{\text{packet}} \tag{3.19}$$

where Λ_{frame} is the total packet generation rate of all child nodes of corresponding parent node and it can be computed as follows:

$$\Lambda_{\text{frame}} = \sum_{v=1}^{N_{\text{c}}} \lambda^v_{\text{frame}} \tag{3.20}$$

P_{avg} is average consumed energy to successfully receive one packet for a parent node. As mentioned before, each shared cell is seen as a subnetwork for a parent node. Hence, it is able to adjust the traffic distribution among subnetworks by altering the child distribution in order to decrease P_{avg}. Furthermore, from Equation 3.18 and 3.19, P_{avg} can be expressed as follows:

$$P_{\text{avg}}(\vec{\Theta}_{\text{frame}}) = \frac{1}{\Lambda_{\text{frame}}} \left[s + \sum_{i=1}^{s} \frac{\theta^i_{\text{frame}}}{1 - K_{\text{PDR}} \theta^i_{\text{frame}} \left[1 - \frac{1}{n^i_{\text{c}}}\right]} \right] \tag{3.21}$$

where Λ_{frame} and s are constant. If ΔP^i is defined as rate of change of P_{avg} when the traffic load, to which i^{th} shared cell is assigned, is increased by δ, it is computed as follows:

$$\Delta P^i = \frac{1}{\Lambda_{\text{frame}}} \left[\frac{1}{1 - K_{\text{PDR}} \theta^i_{\text{frame}} \left[1 - \frac{1}{n^i_{\text{c}}}\right]} \right]^2 \tag{3.22}$$

where $(1 - K_{\text{PDR}} \theta^i_{\text{frame}})$ is $P_{\text{pdr}}(\theta^i_{\text{frame}})$ and it is between 0 and 1. It can be seen form Equation 3.22 that ΔP^i is always positive, which means that decreasing (increase) traffic load of one shared cell always decrease (increase) P_{avg}. However, a parent node cannot decrease a traffic load without increasing another one since a parent cannot change overall traffic load. It means that a parent node is only able to adjust $\vec{\Theta}_{\text{frame}}$, and Λ_{frame} must be constant. Therefore, increasing θ^i_{frame} by δ result in decreasing

θ^j_{frame} by δ. Then, rate of change of P_{avg} becomes $\Delta P^i - \Delta P^j$. Since the aim is to decrease P_{avg}, ΔP^j should be smaller than ΔP^i. From the Equation 3.22, it is obvious that $\Delta P^j > \Delta P^i$ when $\theta^j_{\text{frame}} > \theta^i_{\text{frame}}$. As a result, it can be said that a parent node should try to equalize traffic loads of shared cells by changing child nodes distribution in order to minimize average consumed energy to successfully receive one packet.

The main object of a parent node is to adjust shared cell count, s, and to assign a cell to every child node. As mentioned before, each element of $\vec{V}$ corresponds a child node and $\vec{V}[v]$ indicates the shared cell that v^{th} child node can use. A parent node have s^{N_c} different options for $\vec{V}$ and *frameLength* different options for s. Hence, checking every option to find effective one is highly complex, and devices used in Industrial IoT networks have low computation powers. That is why low complexity heuristic algorithm is needed for solving this problem. The algorithm should try to evenly distribute traffic loads to shared cells since it is proved that equally distributed traffic load give the most efficient solution. Although, there is a disadvantage of the heuristic algorithm that may not find the most efficient solution since it does not check every options, it can give fast response due to low complexity, and the networks can quickly adopt changes–e.g. joining or leaving of a child node.

Although a parent node can easily adjust $\vec{V}$ without any limitation, many application restricts to change *frameLength* parameter. For example, in [14], *frameLength* is set at the beginning and cannot be changed while the network is proceeding. This issue should be taken into consideration while creating the algorithm in order to make it compatible with other protocols and algorithms. Therefore, a set is defined so that it contains permitted values for *frameLength* and it is denoted by $L_{frameLength}$.

Moreover, it is clear that smaller s and bigger *frameLength* result in more effective solution in term of consumed energy. However, these parameters are limited by reliability and average latency constraints. Thus, the algorithm has to select appropriate values for the parameters with respect to these constraints. PDR threshold, Δ_{PDR}, and average latency threshold, Δ_d, are defined for that reasons and the user should adjust these thresholds according to needs of the applications.

Algorithm 1 is defined according to the mentioned specifications. Parent nodes can find the most energy efficient $\vec{V}$ and *frameLength* without violating Δ_d and Δ_{PDR}

constraints. In the algorithm, K_{PDR} and K_{d} values are computed with measurements as mentioned before. The current child node distribution only affects the measurements. After that, the parent node only use $\vec{\Lambda}_{\text{frame}}$, where each element $\lambda^{v}_{\text{frame}}$ indicates the packet generation rate of v^{th} child node. The algorithm checks all possible *frameLength* values in $L_{frameLength}$ set and shared cell count, s. For each *frameLength* and s configuration, it tries to distribute traffic loads among shared cells as equal as possible. This issue implies a well-known problem, namely minimum makespan scheduling. Although there are many algorithms for solving the problem, the solution used in Algorithm 1 is commonly used and the most simple one. In fact, there is no need to use much more complex solutions since variance between the packet generation rates of child nodes is expected to be low. The solution is selecting the child node with the highest traffic generation rate and scheduling to the shared cell with minimum traffic load. The most energy efficient child node distribution can be created by following the stated procedure until every child node is scheduled to a shared cell. Then, the algorithm estimates PDR, π_{PDR}, and average latency, π_d, with respect to the highest loaded shared cell. If the estimated metrics satisfy the constraints, the algorithm proceeds to calculate the power for each *frameLength* and s configuration. After comparing them, the most energy efficient *frameLength* and s configuration is obtained. Output of the algorithm is *frameLength* and $\vec{V}$ which indicates a distribution of child nodes.

Additionally, K_{PDR} and K_d values may be inaccurately estimated due to the physical layer conditions. Because of that, the algorithm should be repeated until a stable solution is found. Moreover, these values may change over time. To overcome this issue, the efficient distribution should be periodically checked by parent nodes.

To summarize this section, it can be said that the most efficient schedule can be achieved by the method. The method can be easily implemented to Industrial IoT networks thanks to 6P mentioned in Section 2.4. Moreover, it is able to coexist with different algorithms even if they impose restriction on *frameLength*.

Algorithm 1 Finding the Efficient Child Node Distribution

procedure

Compute K_{PDR} & K_{d} according to Equation 3.14 & 3.15

Compute N_{c} according to Equation 3.11

Compute Λ_{frame} according to Equation 3.20

$P_{\text{avg}} \leftarrow \infty$ ▷ set to biggest value

for All $l \in L_{frameLength}$ **do**

  for $s \leftarrow 1 : l$ **do**

    $\vec{\Lambda}_{\text{frame}} \leftarrow [\lambda^1_{\text{frame}}\ \lambda^2_{\text{frame}}\ \ldots\ \lambda^{N_{\text{c}}}_{\text{frame}}]$

    $\vec{\Theta}_{\text{frame}} \leftarrow [0]_{1\text{x}s}$

    $\vec{V}' \leftarrow [0]_{1\text{x}N_{\text{c}}}$

    while $\exists \lambda^v_{\text{frame}} \in \vec{\Lambda}_{\text{frame}} : \lambda^v_{\text{frame}} \neq 0$ **do** ▷ equally load the cells

      $v \leftarrow$ index of the maximum value in $\vec{\Lambda}_{\text{frame}}$

      $i \leftarrow$ index of the minimum value in $\vec{\Theta}_{\text{frame}}$

      $\vec{V}'[v] \leftarrow i$

      $\vec{\Theta}_{\text{frame}}[i] \leftarrow \vec{\Theta}_{\text{frame}}[i] + \vec{\Lambda}_{\text{frame}}[v]$

      $\vec{\Lambda}_{\text{frame}}[v] \leftarrow 0$

    $i_{\text{max}} \leftarrow$ index of the maximum value in $\vec{\Theta}_{\text{frame}}$ ▷ the most loaded cell

    $\pi_{\text{PDR}} \leftarrow 1 - K_{\text{PDR}}\ \theta^{i_{\text{max}}}_{\text{frame}} \big[1 - \frac{1}{n_{\text{c}}^{i_{\text{max}}}}\big]$

    $\pi_d \leftarrow \frac{1}{2}\Big[e^{K_{\text{d}}\ (1-\pi_{\text{PDR}})}\Big] \times l \times \textit{slotTime}$

    if $\pi_{\text{PDR}} > \Delta_{\text{PDR}}$ and $\pi_d < \Delta_d$ **then** ▷ check the requirements

      $P'_{\text{avg}} \leftarrow s$ ▷ start to compute average power

      for $i \leftarrow 1 : s$ **do**

        $P'_{\text{avg}} \leftarrow P'_{\text{avg}} + \dfrac{\vec{\Theta}_{\text{frame}}[i]}{1 - K_{\text{PDR}}\ \vec{\Theta}_{\text{frame}}[i]\big[1 - \frac{1}{n_{\text{c}}^i}\big]}$

      $P'_{\text{avg}} \leftarrow \dfrac{P'_{\text{avg}}}{\Lambda_{\text{frame}}}$ ▷ end to compute average power

      if $P_{\text{avg}} > P'_{\text{avg}}$ **then** ▷ find minimum average power

        $P_{\text{avg}} \leftarrow P'_{\text{avg}}$

        $\vec{V} \leftarrow \vec{V}'$

        $\textit{frameLength} \leftarrow l$

return $\vec{V}$ and *frameLength*

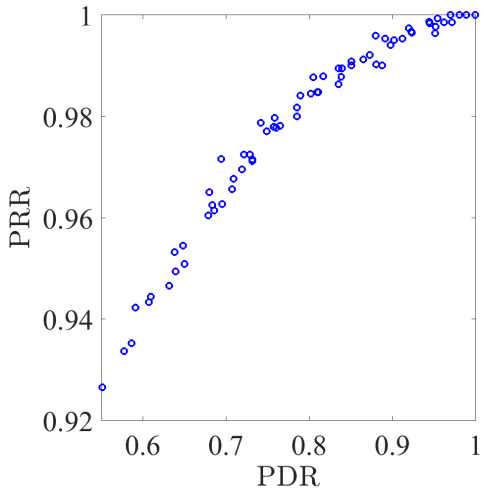
CHAPTER 4

SIMULATION RESULTS

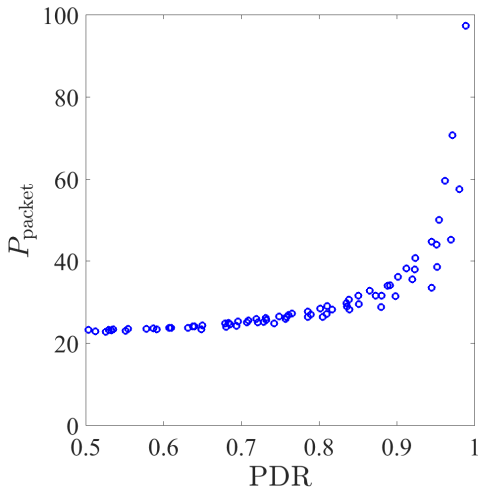
In this section, firstly, it is validated that the model introduced in Section 3.3 is accurate. Then, it is shown that the method approaches the optimum shared cell scheduling. To that end, a number of simulations are conducted by using the Cooja simulator [33], which is a realistic sensor network simulator for Contiki OS. In the simulations, Cooja motes are deployed, TSCH is used as the medium access control and RPL as the network layer protocol with their default parameters. In Cooja, TSCH and RPL protocols have been already implemented in detail. Therefore, only the scheduling algorithm is implemented as a SF in 6P layer. The simulations last 65 minutes, where the network setup is done in the first 5 minutes. Each configuration is shown as a sample point and obtained after averaging 20 independent runs.

4.1 Validation of The Model

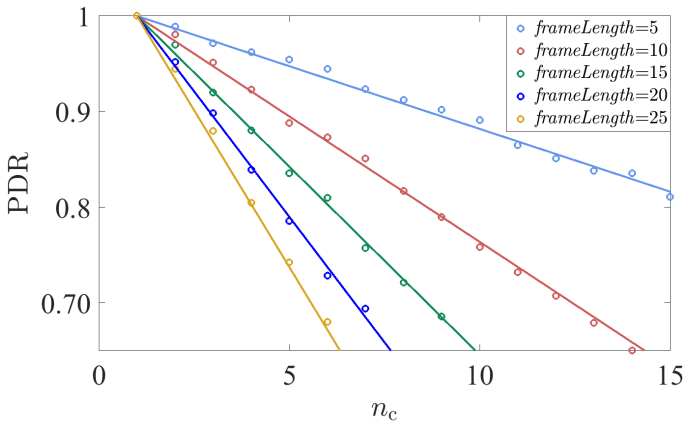
In Figure 4.1, it is shown that estimated metrics presented in Section 3.3 is consistent with the simulation results. In the simulations, a star topology with 1 parent and n_{c} child nodes is constructed, where $n_{\mathrm{c}} = 1, 2, ..., 15$, since the ROI is essential. Also *frameLength* parameter increases from 5 to 25 by 5. Figures 4.1a, 4.1b,4.1c and 4.1d illustrate the corresponding simulation results in the ROI for reliability, energy consumtion, PDR and latency respectively. Moreover, estimations based on Equations 3.8 and 3.9 using the simulation data are also shown as lines in Figures 4.1c and 4.1d. Comparison of the simulation and modeling results reveals that the trend lines are almost the same for all metrics. However, the simulation result values are slightly



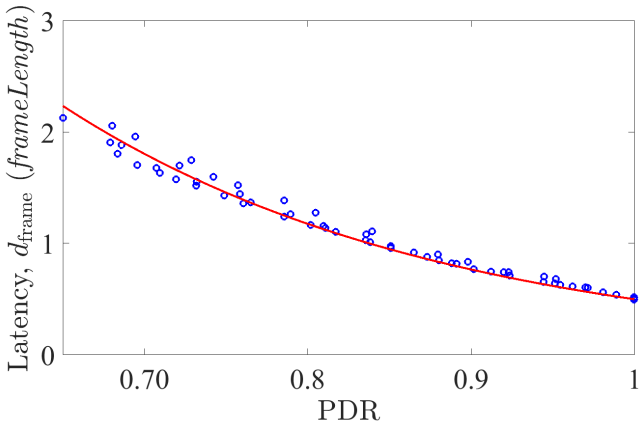
(a) Reliability



(b) Energy Consumtion



(c) PDR



(d) Latency

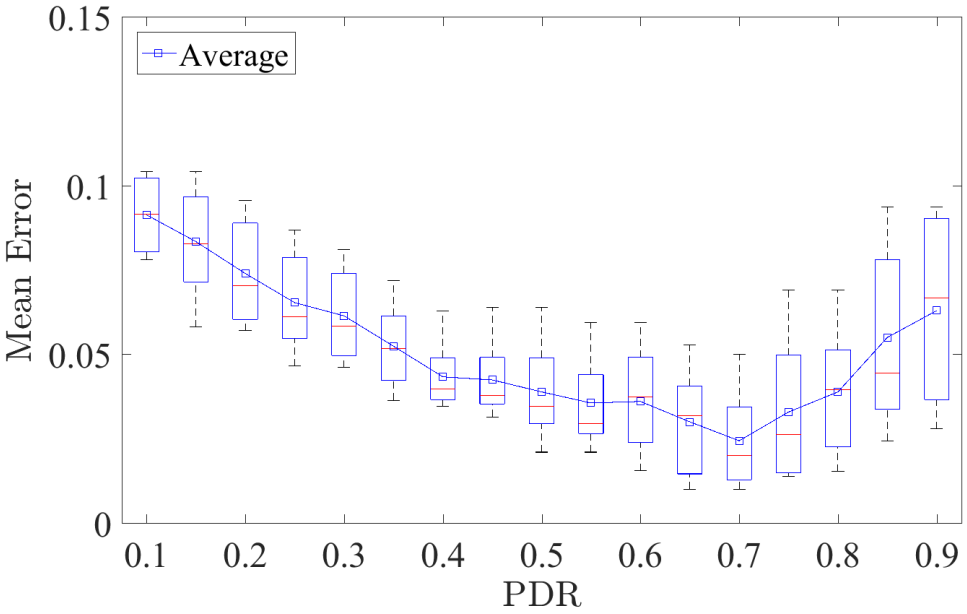
Figure 4.1: Simulation Results for the Network Model.

different. For instance, while comparing Figure 3.8b to Figure 4.1b, it is observed that there is the same trend between P_{packet} and PDR with different power values. It is likely that the reason for this difference is the additional power consumption due to overhead packet transmissions, such as RPL or TSCH control packets, named *EB*. As expected, this reason also cause a difference in latency metric, which can be seen in Figures 3.12 and 4.1d. Notwithstanding the differences in the values, it can be concluded that the model is accurate since obtained trends for all metrics are approximately same to those of the network model. In addition to that, those difference in the values prove that computing K_{PDR} and K_d coefficients, instead of assigning, is more appropriate due to unpredictable nature of the wireless medium.

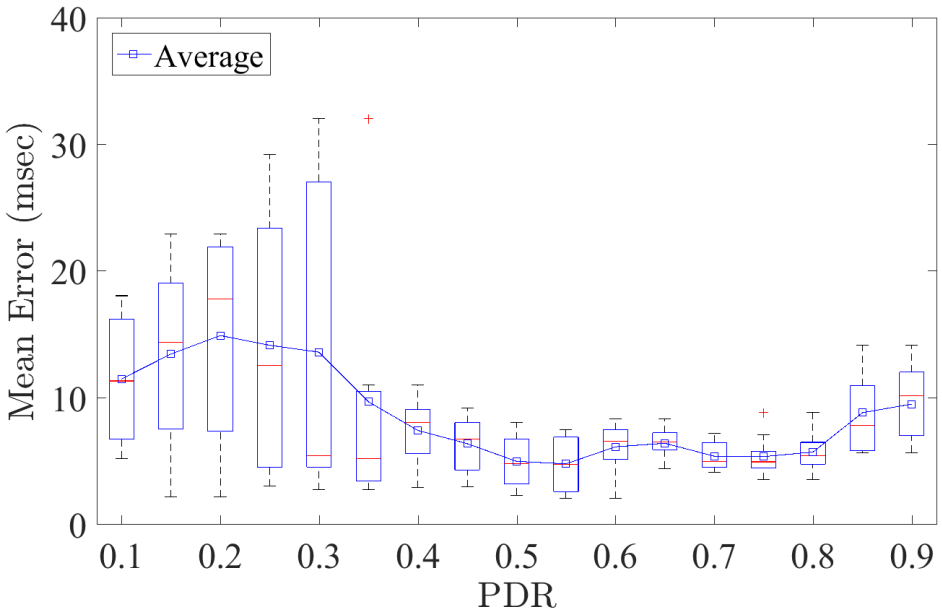
Estimation lines in Figures 4.1c and 4.1d is drawn with respect to 2.1 and 4.35 as K_{PDR} and K_d respectively. These values result in minimum mean error in ROI. However, the parent node compute different K_{PDR} and K_d values for each configuration since it can only measure the metrics within current configuration. Thus, PDR and latency estimations within each configurations result in different mean error. Figure 4.2a and 4.2b shows mean error for PDR and Latency estimations respectively. In the figures, each sample is the results of configurations that have PDR between $C - \frac{W}{2}$ and $C + \frac{W}{2}$. In the graphs, x-axis refers to C and W is 0.1 for each sample. According to Figure 4.2a, the parent node can make good estimations when it experiences PDR that is around 0.7. On the other hand, Figure 4.2a shows that the parent node can make accurate estimation in ROI for latency metric.

4.2 Performance Evaluation

From now on, the simulations are performed to evaluate performance of the method. A star topology is set up with one parent node and 30 child nodes. Δ_{PDR} and Δ_d are chosen as 0.65 and 120 milliseconds respectively. In Figures 4.3 and 4.4, 'o' markers indicate states and each arrow implies change in the state after the method is applied. As mentioned in Section 3.4, a state defined as ($\vec{\Theta}_{\text{frame}}$,*frameLength*) combination. If the method is applied to a state, it gives a new $\vec{V}$ and *frameLength* pair, which means a new state. Simulations are initialized with carefully selected initial states such that there are examples for all regions in the graphs. Figure 4.3 shows changes in P_{avg} with

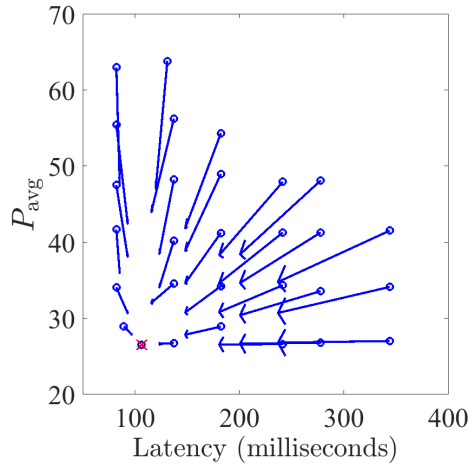


(a) Mean Error for PDR Estimation

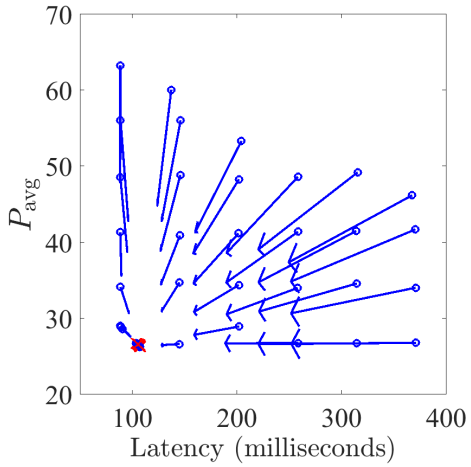


(b) Mean Error for Latency Estimation

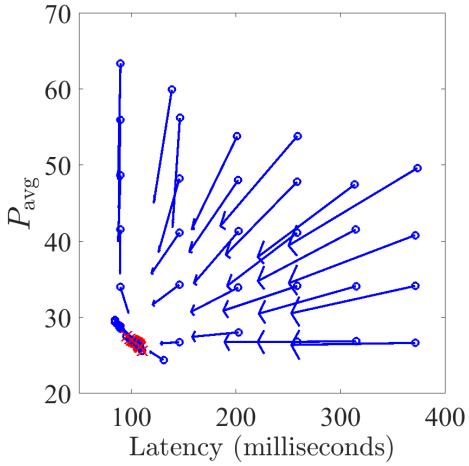
Figure 4.2: Mean Error for Estimations.



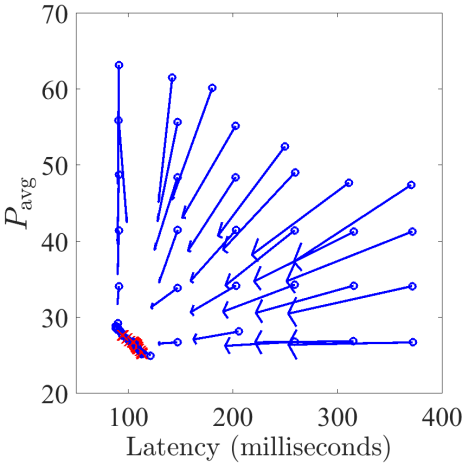
(a) $\sigma = 0$



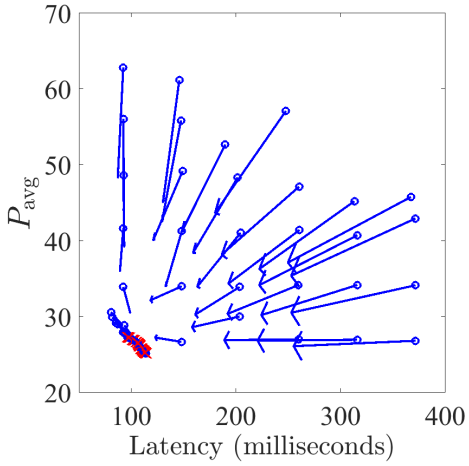
(b) $\sigma = 0.0125$



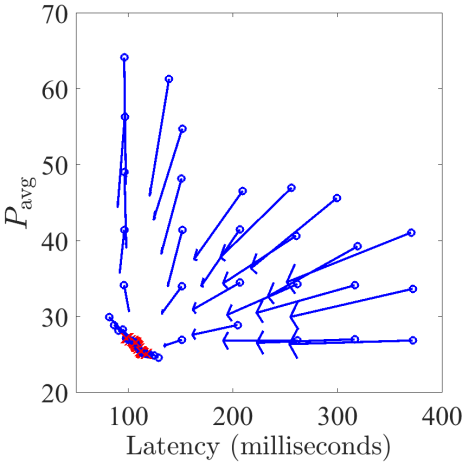
(c) $\sigma = 0.0375$



(d) $\sigma = 0.0625$

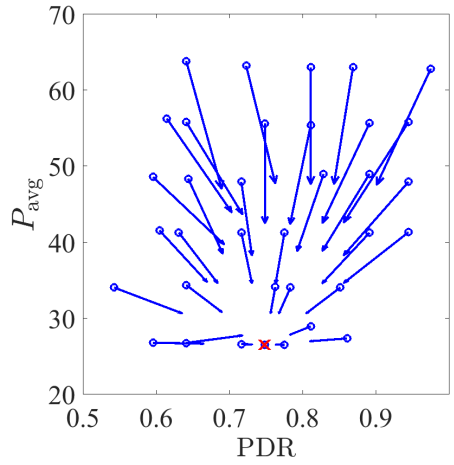


(e) $\sigma = 0.125$

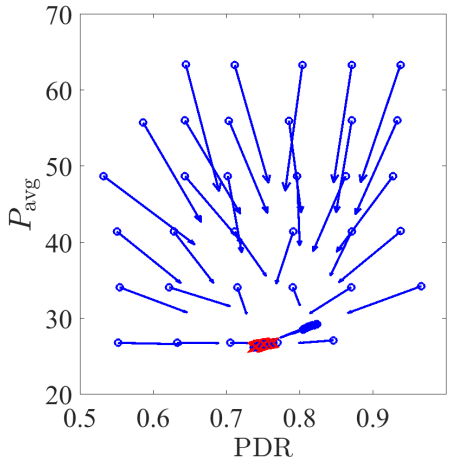


(f) $\sigma = 0.1875$

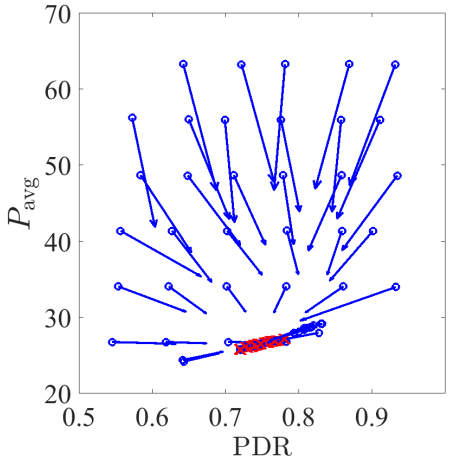
Figure 4.3: Latency Vs Power with Variance.



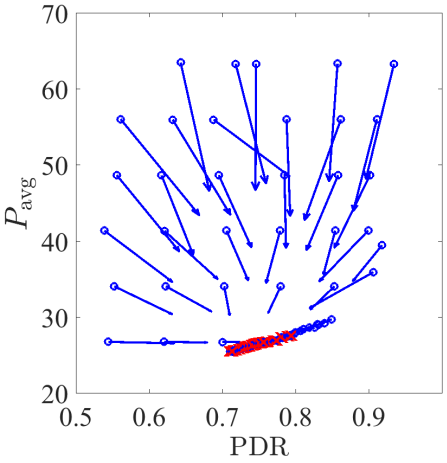
(a) $\sigma = 0$



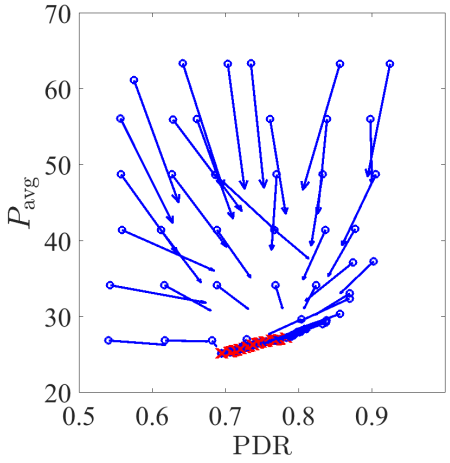
(b) $\sigma = 0.0125$



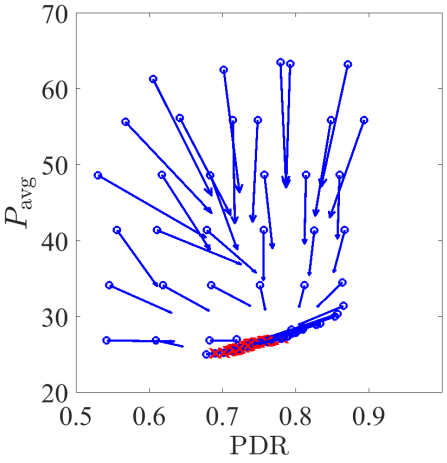
(c) $\sigma = 0.0375$



(d) $\sigma = 0.0625$



(e) $\sigma = 0.125$



(f) $\sigma = 0.1875$

Figure 4.4: PDR Vs Power with Variance.

respect to the average latency in milliseconds, whereas Figure 4.4 shows changes respect to PDR. Additionally, the packet generation rate of each child node is randomly selected with 0.125 second per packet mean, $\mu = 0.125$, and different standard deviations, σ, in order to show the effect of heterogeneous traffic load. Although results in Figures 4.3a and 4.4a are for $\sigma = 0$, which means homogeneous traffic load, from Figures 4.3b to 4.3f and from Figure 4.4b to 4.4f are for $\sigma = 0.0125$, $\sigma = 0.0375$, $\sigma = 0.0625$, $\sigma = 0.125$ and $\sigma = 0.1875$ respectively. Furthermore, the parent node applies the algorithm iteratively until it reaches a stable state, which means the algorithm cannot find a better state than the current state. Reached stable states are indicated by red 'x' markers in the figures. Hence, all other states until stable state is reached called transient states, and they are also shown in the Figures with 'o' markers. In fact, each arrow indicates change with only one iteration, and it is observed that each initial state reaches a stable state at most 2 iterations.

According to the results, each initial state is able to reach stable state with 1 or 2 iterations. In fact, the algorithm does not trigger any change if an initial state is a stable state. It is observed that if an initial state is close to corresponding stable state, the method can make accurate estimation so that it reaches the stable state with 1 iteration. On the other hand, if an initial state is not close to corrseponding stable state, the first iteration make it closer to the stable state, and then, the second iteration reaches it to the stable state.

Expectedly, all initial states reaches the same stable state in the case of homogeneous traffic loads whereas it can be several in the case of heterogeneous traffic load due to different $\vec{\Lambda}_{\text{frame}}$ vectors . Each stable state satisfies the QoS requirements with minimum energy consumption. In other words, the method can decrease the consumed energy without violating reliability and average latency constraints.

Moreover, when an initial state does not satisfy the constraints, the method leads to increase reliability or decrease latency even if it causes an increase in energy consumption. In other words, the methods prioritize satisfying QoS metrics instead of minimizing energy consumption. In fact, this prioritization is necessary for Industrial IoT networks.

From the simulations, it is concluded that analyzed model is accurate for networks with sporadic traffic, latency and reliability constraints. Moreover, the method based on this model gives the most efficient parameters without violating QoS requirements. Additionally, it is shown that proposed method is applicable to both homogeneous and heterogeneous traffic loads and satisfies Industrial IoT requirements.

CHAPTER 5

CONCLUSION

6TiSCH is the standard protocol stack for Industrial IoT networks. It provides an interface to the user for creating and implementing a scheduling algorithm. In 6TiSCH, TSCH and RPL protocols are used for medium access and routing respectively. TSCH allows assigning a cell in a shred or dedicated manner. Although dedicated cells can handle a periodic traffic with low latency and minimum energy consumption, these cells cause a large amount of wasted energy in the case of a sporadic traffic since applications in Industrial IoT requires low latency relative to packet inter-arrival times. Moreover, RPL protocol create parent-child relationship between nodes such that a child node can send a packet only to its parent node and a parent node tends to have multiple child nodes. Therefore, a parent node can assign a shared cell to more than one child nodes for sporadic traffics in order to decrease the wasted energy.

In this thesis, a shared cell scheduling method is presented for sporadic traffics in Industrial IoT networks. Firstly, a network model is created and analyzed. According to the model, it is shown that network characteristics can be estimated. Then, a method is proposed based on these estimations.

The model is for a subnetwork and it is proven in 3.3 that each subnetwork can be thought as independent from each other. A subnetwork is modeled based on observations which means that it is a heuristic model. The model shows that reliability directly depends on PDR. In other words, PDR can be thought as a reliability metric. Moreover, the model focuses on a area that reliability is high since Industrial IoT networks always work with high reliability. In the interested area, it is observed that there are a linear relationship between PDR and the total generated packet in one

frame, and a exponential relationship between PDR and latency. Therefore, the two relationships are exploited to estimate PDR and latency values with different network configurations. The model is also proven to be accurate by simulations.

The aim of the method is to minimize power consumption with meeting reliability and latency requirements. The model tries to find the optimum network configuration for this purposes. Since trial-and-error method is costly, it firstly uses the model to estimate effects of different network configurations. Then, select the most efficient one. Moreover, the method updates the model parameters according to measurements. Thus, it can adapt itself to unpredictable effects of wireless medium thanks to heuristic basis of the model. Since the model is for a subnetwork, the network schedule can be optimized in a distributed manner. Performance of the method is tested by conducting simulations. Obtained results show that the method provides the most ower efficient shared cell scheduling considering the requirements.

Additionally, it is clarified that the method is compatible with different network designs. Therefore, it can be said that the method can be integrated to other TSCH based protocols designed for scheduling.

This thesis can be extended with two ways. Firstly, although proposed method and model are tested with a well-known simulator, called COOJA, they should be tested with real devices. In other words, implementing the method to a real Industrial IoT network and analyzing the experimental data can be a future work. Secondly, this thesis only focus on the number of assigned shared cells and distribution of the child nodes among these cells; however, selection of shared cells are not addressed. Thus, the method can be developed with a distributed conflict-free cell selection algorithm for parent nodes.

REFERENCES

[1] BCM Advanced Research. `http://www.bcmcom.com/solutions_application_industry40.htm`. Accessed: 2017-08-14.

[2] Nicola Accettura, Maria Rita Palattella, Gennaro Boggia, Luigi Alfredo Grieco, and Mischa Dohler. Decentralized traffic aware scheduling for multi-hop low power lossy networks in the internet of things. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, pages 1–6. IEEE, 2013.

[3] Roger Alexander, Anders Brandt, JP Vasseur, Jonathan Hui, Kris Pister, Pascal Thubert, P Levis, Rene Struik, Richard Kelsey, and Tim Winter. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550, March 2012.

[4] Satish Anamalamudi, Mingui Zhang, Abdur Rashid Sangi, Charles E. Perkins, and S.V.R Anand. Scheduling Function One (SF1) for hop-by-hop Scheduling in 6tisch Networks. Internet-Draft draft-satish-6tisch-6top-sf1-03, Internet Engineering Task Force, February 2017. Work in Progress.

[5] SIG Bluetooth. Bluetooth specification, 2003.

[6] Simone Brienza, Domenico De Guglielmo, Cesare Alippi, Giuseppe Anastasi, and Manuel Roveri. A learning-based algorithm for optimal mac parameters setting in IEEE 802.15. 4 wireless sensor networks. In *Proceedings of the 10th ACM symposium on Performance evaluation of wireless ad hoc, sensor, & ubiquitous networks*, pages 73–80. ACM, 2013.

[7] Tengfei Chang, Thomas Watteyne, Qin Wang, and Xavier Vilajosana. LLSF: Low Latency Scheduling Function for 6TiSCH Networks. In *Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on*, pages 93–95. IEEE, 2016.

[8] Domenico De Guglielmo, Beshr Al Nahas, Simon Duquennoy, Thiemo Voigt, and Giuseppe Anastasi. Analysis and experimental evaluation of IEEE 802.15. 4e TSCH CSMA-CA algorithm. *IEEE Transactions on Vehicular Technology*, 66(2):1573–1588, 2017.

[9] Domenico De Guglielmo, Giuseppe Anastasi, and Alessio Seghetti. From IEEE 802.15. 4 to IEEE 802.15. 4e: A step towards the internet of things. In *Advances onto the Internet of Things*, pages 135–152. Springer, 2014.

[10] Mario Di Francesco, Giuseppe Anastasi, Marco Conti, Sajal K Das, and Vincenzo Neri. Reliability and energy-efficiency in IEEE 802.15. 4/ZigBee sensor networks: an adaptive and cross-layer approach. *IEEE Journal on selected areas in communications*, 29(8):1508–1524, 2011.

[11] Marc Domingo-Prieto, Tengfei Chang, Xavier Vilajosana, and Thomas Watteyne. Distributed pid-based scheduling for 6tisch networks. *IEEE Communications Letters*, 20(5):1006–1009, 2016.

[12] Diego Dujovne, Luigi Alfredo Grieco, Maria Rita Palattella, and Nicola Accettura. 6TiSCH 6top Scheduling Function Zero (SF0). Internet-Draft draft-ietf-6tisch-6top-sf0-05, Internet Engineering Task Force, July 2017. Work in Progress.

[13] Diego Dujovne, Thomas Watteyne, Xavier Vilajosana, and Pascal Thubert. 6TiSCH: deterministic IP-enabled industrial internet (of things). *IEEE Communications Magazine*, 52(12):36–41, 2014.

[14] Simon Duquennoy, Beshr Al Nahas, Olaf Landsiedel, and Thomas Watteyne. Orchestra: Robust mesh networks through autonomously scheduled tsch. In *Proceedings of the 13th ACM conference on embedded networked sensor systems*, pages 337–350. ACM, 2015.

[15] Simon Duquennoy, Xavier Vilajosana, and Thomas Watteyne. 6TiSCH Autonomous Scheduling Function (ASF). Internet-Draft draft-duquennoy-6tisch-asf-00, Internet Engineering Task Force, July 2017. Work in Progress.

[16] Thang Phan Duy, Thanh Dinh, and Younghan Kim. Distributed cell selection for scheduling function in 6TiSCH networks. *Computer Standards & Interfaces*, 53:80–88, 2017.

[17] Erina Ferro and Francesco Potorti. Bluetooth and Wi-Fi wireless protocols: a survey and a comparison. *IEEE Wireless Communications*, 12(1):12–26, 2005.

[18] Guillaume Gaillard, Dominique Barthel, Fabrice Theoleyre, and Fabrice Valois. High-reliability scheduling in deterministic wireless multi-hop networks. In *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*, pages 1–6. IEEE, 2016.

[19] Marco Gribaudo, Daniele Manini, Alessandro Nordio, and Carla-Fabiana Chiasserini. Transient analysis of IEEE 802.15. 4 sensor networks. *IEEE Transactions on wireless communications*, 10(4):1165–1175, 2011.

[20] Inès Hosni and Fabrice Théoleyre. Self-Healing Distributed Scheduling for End-to-End Delay Optimization in Multihop Wireless Networks with 6TiSCH. *Computer Communications*, 2017.

[21] Inès Hosni, Fabrice Théoleyre, and Noureddine Hamdi. Localized scheduling for end-to-end delay constrained Low Power Lossy networks with 6TiSCH. In *Computers and Communication (ISCC), 2016 IEEE Symposium on*, pages 507–512. IEEE, 2016.

[22] Thong Huynh, Fabrice Theoleyre, and Won-Joo Hwang. On the interest of opportunistic anycast scheduling for wireless low power lossy networks. *Computer Communications*, 104:55–66, 2017.

[23] Ren-Hung Hwang, Chih-Chiang Wang, and Wu-Bin Wang. A Distributed Scheduling Algorithm for IEEE 802.15. 4e Wireless Sensor Networks. *Computer Standards & Interfaces*, 52:63–70, 2017.

[24] Yichao Jin, Parag Kulkarni, James Wilcox, and Mahesh Sooriyabandara. A centralized scheduling algorithm for IEEE 802.15. 4e TSCH based industrial low power wireless networks. In *Wireless Communications and Networking Conference (WCNC), 2016 IEEE*, pages 1–6. IEEE, 2016.

[25] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4):239–242, 2014.

[26] Gabriel Montenegro, Jonathan Hui, David Culler, and Nandakishore Kushalnagar. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, September 2007.

[27] Gabriel Montenegro, Christian Schumacher, and Nandakishore Kushalnagar. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, August 2007.

[28] Antoni Morell, Xavier Vilajosana, José López Vicario, and Thomas Watteyne. Label switching over IEEE802. 15.4 e networks. *Transactions on Emerging Telecommunications Technologies*, 24(5):458–475, 2013.

[29] Esteban Municio and Steven Latré. Decentralized broadcast-based scheduling for dense multi-hop TSCH networks. In *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture*, pages 19–24. ACM, 2016.

[30] International Society of Automation. Wireless Systems for Industrial Automation: Process Control and Related Applications. *ISA-100.11a-2009 Standard*, 2009.

[31] Mike Ojo and Stefano Giordano. An efficient centralized scheduling algorithm in IEEE 802.15. 4e TSCH networks. In *Standards for Communications and Networking (CSCN), 2016 IEEE Conference on*, pages 1–6. IEEE, 2016.

[32] Mike Ojo, Stefano Giordano, Giuseppe Portaluri, Davide Adami, and Michele Pagano. An energy efficient centralized scheduling scheme in TSCH networks. In *Communications Workshops (ICC Workshops), 2017 IEEE International Conference on*, pages 570–575. IEEE, 2017.

[33] Fredrik Osterlind. A sensor network simulator for the Contiki OS. *Swedish Institute of Computer Science (SICS), Tech. Rep. T2006-05*, 2006.

[34] Baver Ozceylan, Berk Ünlü, and Buyurman Baykal. An Energy Efficient Optimum Shared Cell Scheduling for TSCH Networks. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2017 IEEE 13th International Conference on*. IEEE, (in press).

[35] Maria Rita Palattella, Nicola Accettura, Mischa Dohler, Luigi Alfredo Grieco, and Gennaro Boggia. Traffic Aware Scheduling Algorithm for reliable low-power multi-hop IEEE 802.15. 4e networks. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pages 327–332. IEEE, 2012.

[36] Maria Rita Palattella, Nicola Accettura, Luigi Alfredo Grieco, Gennaro Boggia, Mischa Dohler, and Thomas Engel. On optimal scheduling in duty-cycled industrial IoT applications using IEEE802. 15.4 e TSCH. *IEEE Sensors Journal*, 13(10):3655–3666, 2013.

[37] Maria Rita Palattella, Thomas Watteyne, Qin Wang, Kazushi Muraoka, Nicola Accettura, Diego Dujovne, Luigi Alfredo Grieco, and Thomas Engel. On-the-fly bandwidth reservation for 6TiSCH wireless industrial networks. *IEEE Sensors Journal*, 16(2):550–560, 2016.

[38] Pangun Park, Piergiuseppe Di Marco, Carlo Fischione, and Karl Henrik Johansson. Modeling and optimization of the IEEE 802.15. 4 protocol for reliable and timely communications. *IEEE Transactions on Parallel and Distributed Systems*, 24(3):550–564, 2013.

[39] Pangun Park, Piergiuseppe Di Marco, Pablo Soldati, Carlo Fischione, and Karl Henrik Johansson. A generalized Markov chain model for effective analysis of slotted IEEE 802.15. 4. In *Mobile Adhoc and Sensor Systems, 2009. MASS'09. IEEE 6th International Conference on*, pages 130–139. IEEE, 2009.

[40] Stig Petersen and Simon Carlsen. WirelessHART versus ISA100. 11a: The format war hits the factory floor. *IEEE Industrial Electronics Magazine*, 5(4):23–34, 2011.

[41] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). RFC 7252, June 2014.

[42] Fadi Shrouf, Joaquin Ordieres, and Giovanni Miragliotta. Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm. In *Industrial Engineering and Engineering Management (IEEM), 2014 IEEE International Conference on*, pages 697–701. IEEE, 2014.

[43] IEEE Computer Society. IEEE Standard for Low-Rate Wireless Networks. *IEEE Std.*, 2015.

[44] IEEE Computer Society. IEEEStandard for Information technology-Telecommunications and information exchange between systems Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std.*, 2016.

[45] Jianping Song, Song Han, Al Mok, Deji Chen, Mike Lucas, Mark Nixon, and Wally Pratt. WirelessHART: Applying wireless technology in real-time industrial process control. In *Real-Time and Embedded Technology and Applications Symposium, 2008. RTAS'08. IEEE*, pages 377–386. IEEE, 2008.

[46] Ridha Soua, Pascale Minet, and Erwan Livolant. DiSCA: A distributed scheduling for convergecast in multichannel wireless sensor networks. In *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, pages 156–164. IEEE, 2015.

[47] Ridha Soua, Pascale Minet, and Erwan Livolant. Wave: a distributed scheduling algorithm for convergecast in IEEE 802.15. 4e TSCH networks. *Transactions on Emerging Telecommunications Technologies*, 27(4):557–575, 2016.

[48] Pascal Thubert. An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4. Internet-Draft draft-ietf-6tisch-architecture-11, Internet Engineering Task Force, January 2017. Work in Progress.

[49] Pascal Thubert and Jonathan Hui. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. RFC 6282, September 2011.

[50] Andrew Tinka, Thomas Watteyne, Kris Pister, and Alexandre M Bayen. A decentralized scheduling algorithm for time synchronized channel hopping. In *ADHOCNETS*, pages 201–216. Springer, 2010.

[51] Qin Wang, Xavier Vilajosana, and Thomas Watteyne. 6top Protocol (6P). Internet-Draft draft-ietf-6tisch-6top-protocol-07, Internet Engineering Task Force, June 2017. Work in Progress.

[52] Lixia Zhang, Steve Berson, Shai Herzog, and Sugih Jamin. Resource reservation protocol (RSVP)–Version 1 functional specification. *Resource*, 1997.

[53] Alliance Zigbee. Zigbee specification. *ZigBee document 053474r13*, 2006.