Load balancing in multichannel data collection wireless sensor networks

Hamadoun Tall

UNIVERSITÉ CLERMONT AUVERGNE

École Doctorale
Sciences pour l'Ingénieur de Clermont-Ferrand

Thèse

Présentée par

Hamadoun TALL

Pour obtenir le grade de

DOCTEUR D'UNIVERSITÉ

Spécialité : Informatique

Titre :

Load balancing in multichannel data collection wireless sensor networks

Soutenue publiquement le 14 Mai 2018, devant le jury :

Rapporteurs :	M. Hervé Rivano	Professeur à l'INSA de Lyon
	Mme Nathalie Mitton	Directrice de Recherche à l'INRIA de Lille
Examinateurs :	M. Thierry Val	Professeur à l'Université de Toulouse
	Mme Saoucène Mahfoudh	MCF à King Abdulaziz University de Jeddah
Directeur de thèse :	M. Michel Misson	Professeur à l'Université Clermont Auvergne
Co-encadrant :	M. Gérard Chalhoub	MCF-HDR à l'Université Clermont Auvergne

CLERMONT AUVERGNE UNIVERSITY

PHD THESIS

Load balancing in multichannel data collection wireless sensor networks

---

Author:
Hamadoun TALL

Supervisors:
Pr. Michel MISSON
Dr. Gérard CHALHOUB

A thesis submitted in fulfillment of the requirements for the degree of PhD in Computer Science

in the

LIMOS UMR 6158 CNRS Laboratory
Doctoral school of Science for Engineering

Acknowledgements

I am glad to have such an opportunity to express my great gratitude and respect to people who helped me when I was pursuing my PhD. Without their supports and encouragements I cannot go so far.

First of all, I want to express my deep gratitude to Prof. Michel MISSON and Dr. Gérard CHALHOUB, my thesis director and advisor respectively, for their excellent support, their advice and guidance during this research. Their constant availability, academic rigour and experience helped me to develop good research habits and gave me the will and the power to reach success. Thank you a lot for you both.

I would like to thank Prof. Hervé RIVANO and DR Nathalie MITTON for accepting to review my manuscript and be part of my jury. Thank you for your availability and your relevant and constructive comments.

I also want to thank Prof. Thierry VAL and Dr. Saoucène MAHFOUDH for accepting to be part of my jury. Thank you for your availability.

I also would like to thank European Regional Development Fund (FEDER) program of 2014-2020, the region council of Auvergne-Rhône-Alpes and the Digital Trust Chair of the Clermont Auvergne University for the financial support of my thesis.

A big thanks to all my colleagues from C6 office. It was a great pleasure to share the office with my fellow PhD students : Honoré, Thérèse, Rana, Moussa, Malick, Xavier, David, Jinpeng, Sylvestre and Mouna. Thank you for the great time spent together in our office C6. I also would like to thank all researchers in the LIMOS Laboratory and specially "Replic" team members, the team in which I did my thesis. We spent great research and pop times together, it was great!

I also say thank to Prof. Nadir Hakem from LRTCS Laboratory of Val D'Or in Quebec (Canada) who welcomed me in January 2016 for one month research collaboration. Thank you for your hospitality!

I would like to give a special thankfulness to Prof. Hervé RIVANO, Prof. Fabrice VALOIS and Dr. Khaled BOUSSETTA in CITI laboratory (INSA Lyon) where I did my Master internship. I spent a great time in CITI and I learnt a lot thank to you all. In one word, my stay in CITI has been very beneficial!

I am also grateful for many peoples in my former Universities (Université Claude Bernard de Lyon 1 - UCBL1 and Institut Francophone International- IFI). A special thank to M. Alain BOUCHER (IFI), M. Victor Moraru (IFI), M. Quang NGYEN (IFI), M. Van Nam NGYEN (IFI), M. Vinh Ho TO (IFI) and M. Anthony BUCHON (UCBL1) who taught me and guided me into research work.

Finally, and most significantly, I want to give my greatest gratitude to my parents and my dear brother Abdoulaye. They believed in me and supported me. Without their constant support, I cannot go far like that...

Abstract

The popularity of wireless sensor networks (WSNs) is increasing due to their ease of deployment and auto-configuration capabilities. They are used in different application domains including data collection with convergecast scenarios. In convergecast, all data collected in the network is destined to one common node usually called the sink. In case of high carried traffic load and depending on the used routing policy, this many-to-one data collection leads to congestion and queue overflow mainly in nodes located near the sink. Congestion and queue overflow reduce delivery ratio that negatively affects the network efficiency.

Wireless sensor nodes are resource constrained devices with limited buffers size to store and forward data to the sink. Introducing multichannel communication in WSNs helps to increase the carried traffic load thanks to allowing parallel data transmission and reduction of contention and interference. With high traffic load, the number of data packets travelling from leaf nodes towards the sink becomes higher. In case the routing scheme does not balance the traffic load, it will be unfairly distributed between forwarding nodes. Thus, nodes that are in part of the routing will be overloaded while others are less used. Overloaded nodes increase the risk of congestion and queue overflow leading to data loss that reduces the throughput. Therefore, we need to couple the routing protocols with traffic load balancing scheme in high traffic load network scenarios.

The goal of this thesis is to propose an efficient routing solution to prevent congestion and queue overflow in high data rate convergecast WSNs, in such a way, to optimize data delivery ratio at the sink node.

On the one hand, we proposed a single channel traffic load balancing routing protocol, named S-CoLBA (Single channel Collaborative Load balancing routing). It relies on data queueing delay metric and best score (according to the value of the metric) next hop neighbors to fairly distribute traffic load in per hop basis in the network. Since the carried traffic load increases in multichannel communication, on the other hand, we adapted our contribution to cope with multichannel WSNs and we named it as Multichannel CoLBA (M-CoLBA). As broadcasting information is not straightforward in multichannel, we optimize M-CoLBA to use piggybacking scheme for routing information sharing in the network. This enhanced version is called ABORt for Acknowledgement-Based opportunistic Routing protocol and relies on ACK frames to share routing information. Doing so helps to optimize data frame end-to-end delay and to reduce the transmitted beacons in the network. ABORt fairly distributes traffic load in the network and avoids congestion and queue overflow.

We evaluated the performance of our contributions in both simulation using Contiki OS Cooja simulator and experiment (only for S-CoLBA) on TelosB motes. Obtained results in both simulation and experiment confirm the efficiency of our routing protocols in term of packet delivery ratio and queue overflow compared to some existing routing protocols in the literature.

Keywords: wireless sensor networks, multichannel, routing protocols, congestion, queue overflow, load balancing, convergecast.

Résumé

Les Réseaux de Capteurs Sans Fil (RCSF) sont de plus en plus exploités par des applications diverses grâce à leur facilité de déploiement et d'auto-configuration. Les applications de collecte de données qui utilisent les RCSF ont souvent un profil *convergecast* : l'ensemble des données récoltées par tous les capteurs du réseau sont acheminées vers un puits de collecte, grâce à une communication multi-saut.

Pendant l'acheminement des données des nœuds de collecte vers le puits, des goulots d'étranglement sont fréquemment observés, principalement au voisinage du puits. Cela est du à la congestion et au phénomène d'entonnoir couramment observé sur le trafic de données ayant un profile *convergecast*. Outre un risque accrue de collision, cela entraîne le débordement des files d'attente des nœuds concernés conduisant à des pertes de données. Cette perte réduit le taux de livraison au puits entraînant une baisse du débit du réseau. Afin de réduire ces pertes et de permettre un meilleur taux de livraison au puits, le trafic doit être équitablement réparti au niveau de chaque saut pendant l'acheminement.

Dans cette thèse, nous avons d'une part proposé S-CoLBA (*Single channel Collaborative Load Balancing Algorithm*), un protocole mono-canal de routage dynamique avec équilibrage de la charge. Sa métrique de routage est basée sur le délais moyen d'accès au medium radio par nœud. Chaque nœud choisit comme prochain saut à destination du puits, un de ses voisins ayant le délais d'accès le plus court. S-CoLBA intègre également une surveillance permanente des files d'attente des nœuds afin de prévenir la congestion et d'éviter le débordement de ces files.

D'autre part, nous avons adapter S-CoLBA pour le rendre utilisable dans un réseau multicanal. Cette version du protocole s'appelle M-CoLBA (pour Mulitchannel CoLBA). M-CoLBA évite la congestion en équilibrant la charge grâce à une répartition du trafic au niveau de chaque saut du réseau. Dans un réseau multicanal, le problème de support de diffusion se pose. M-CoLBA introduit des périodes de synchronisations où tous les nœuds utilisent le même canal pour échanger les informations de routage. Ces périodes de synchronisation contribuent à allonger les délais de bout en bout des paquets. Nous avons ainsi optimisé M-CoLBA en "surchargeant" les acquittements des trames avec les informations de routage (*piggybacking*) et les états des files d'attente. Cela évite de passer par des périodes de synchronisation pour diffuser ces informations. Cette version optimisée s'appelle ABORt (*Acknowledgement-Based opportunistic Routing protocol*). Dans un cas de trafic de type *convergecast*, ABORt induit une diversité des routes prises par les données collectées, ce qui est bénéfique à la quantité de données transportées et à la robustesse de la solution.

Les contributions ont été évaluées par simulation et expérimentation dans un réseau monocanal et multicanal. Les résultats montrent que nos contributions améliorent le taux de livraison des données au puits, optimisent le délais de bout en bout et réduisent la quantité de trafic de contrôle comparé à des solutions déjà existantes.

Mots-clés : réseaux de capteurs sans fil, communication multicanal, protocoles de routage, congestion, débordement des files d'attente, équilibrage de charge, collecte de données.

Contents

List of Figures

List of Tables

List of Abbreviations

ABORt	Acknowledgement-Based Opportunistic Routing protocol
ACK	ACKnowledgement
AODV	Ad hoc On-Demand Distance Vector
A-TSCH	Adaptive Times-Slotted Channel Hopping
BS	Base Station
CADA	Congestion Avoidance, Detection and Alleviation
CAP	Contention Access Period
CDMA	Code-Division Multiple Access
CMS	Capacitate Minimal Spanning
CODA	COngestion Detection and Avoidance
CoLBA	Collaborative Load Balancing Algorithm
CN	Congestion Notification
CPU	Central Process Unit
CRC	Cyclic Redundancy Check
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DAG	Directed Acyclic Graph
DAO	Destination Advertisement Object
DGRM	Directed Graph Radio Medium
DIO	DODAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination-Oriented Directed Acyclic Graph
DRCS	Distributed Routing and Channel Selection scheme
DSME	Deterministic and Synchronous Multi-channel Extension
EPA	Enhanced Performance Architecture
ESRT	Event-to-Sink Reliable Transport protocol
ETX	Expected Transmission Count
FCS	Frame Control Sequence
FDMA	Frequency Division Multiple Access
FIFO	First In, First Out
GHz	GigaHertz
GloMoSim	Global Mobile Information System Simulator
GMCAR	Grid-based Multipath with Congestion Avoidance Routing

HART	Highway Addressable Remote Transducer Protocol
HMC-MAC	Hybrid Multi-Channel Medium Access Control
ID	IDentifier
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
I2MR	Interference-Minimized Multipath Routing
ISA	International Society of Automation
ISM	Industrial, Scientific and Medical
ITU	International Telecommunication Union
ITU-T	ITU Telecommunication standardization sector
JRCA	Joint Routing and Channel Assignment
LBR	Load Balancing Routing
LLNs	Low-power and Lossy Networks
LP-WPANs	Low Power Wireless Personal Area Networks
MAC	Medium Access Control
MASN	Multichannel Access for Sensor Networks
MCC	Multi-Channel Collection
MC-LMAC	Multi-Channel Lightweight Medium Access Control
M-CoLBA	Multichannel Collaborative Load Balancing Algorithm
MCRT	Multi-Channel Real-Time
MMSN	Multi-frequency Media access control protocol for wireless Sensor Networks
MRM	Multi-path Ray-tracer Medium
NS	Network Simulator
OSI	Open Systems Interconnection
OS	Operating System
PACA	Popularity Aware Congestion Avoidance
PCSS	Predictive Channel Scanning and Switching
PDR	Packet Delivery Ratio
RAM	Random Access Memory
ROLL	Routing Over Low power and Lossy networks
RPL	Routing Protocol for Low-power and lossy networks
RSSI	Received Signal Strength Indicator
SCADA	Supervisory Control And Data Acquisition
S-CoLBA	Single channel Collaborative Load Balancing Algorithm
SINR	Signal to Interference plus Noise Ratio
TARP	Traffic Aware Routing Protocol
TDMA	Time-Division Multiple Access
TMCP	Tree-based Multi-Channel Protocol
TSCH	Time Slotted Channel Hopping

UDGM	Unit Disk Graph Medium
Wi-Fi	Wireless Fidelity
WSN	Wireless Sensor Network

Chapter 1

Introduction and Rationale

Nowadays, Wireless Sensor Networks (WSNs) have become popular thanks to their ease of deployment and their energy autonomy. These networks are more and more considered for deployment in many aspects of our daily life. They are applied in several domains with various applications, among them we can cite: weather or volcanic earthquake monitoring [1], automation monitoring [2], health care monitoring [3] and industrial plan monitoring [4].

A WSN is a network composed of a large number of sensor nodes that sense the environment and communicate this information possibly via multiple hops to one or more collection points called sink [5]. Data transmission is done through wireless links as illustrated in figure 1.1, where an example of a WSN is depicted. Each sensor node is composed of one or several capture units, a processing unit, a wireless transceiver and a source of power as depicted by figure 1.1 (b).

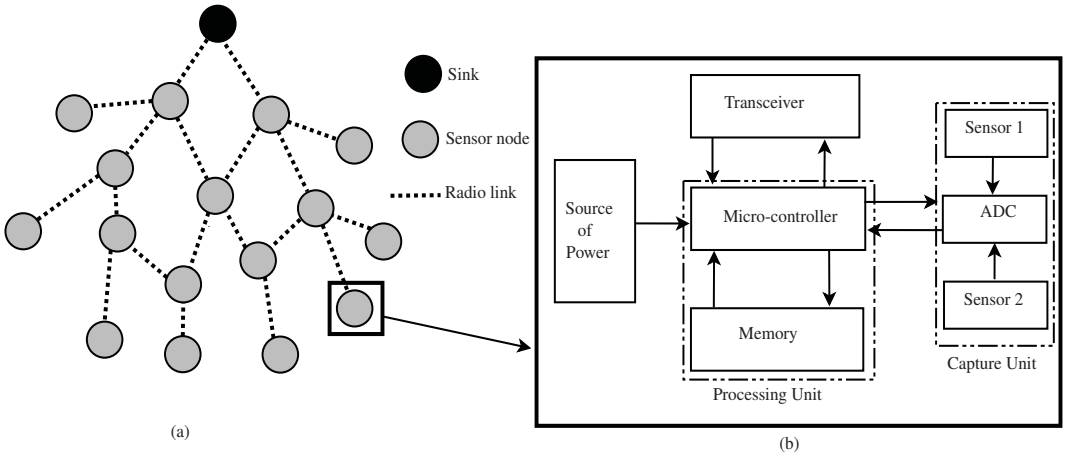


FIGURE 1.1: An example of a Wireless Sensor Network (a) and the typical architecture of sensor node (b).

When it comes to data transmission, WSNs can provide three types of communication:

- point-to-point: that is a communication scenario used between 2 nodes inside the network, one node transmitting data to an other;
- point-to-multipoint: one node transmitting information to several other nodes. This traffic profile is generally used by the sink to broadcast commands or information to nodes in the network;

- multipoint-to-point, usually called convergecast in the literature: several nodes transmit data to one. This communication scenario is used by nodes in the network to send data or network service information (topology, congestion level, residual energy) to the sink.

Nodes composing a WSN are very tiny with limited resources as memory, processing power, energy (they are battery powered) and communication range. Due to the limited communication range, several nodes in the network are out of range of the sink and have to transmit their data using multihop communication which is transmitting data hop by hop until it reaches the destination.

1.1 Convergecast wireless sensor networks

In convergecast WSN nodes sense the environment and collected data in the overall network are transmitted towards the sink. In this many-to-one data transmission, several nodes must use multihop communication to forward their data because they are out of range of the sink. Figure 1.2 presents an example of convergecast data transmission in WSN. In the figure, we notice that the traffic load is increasing as data travel towards the sink.

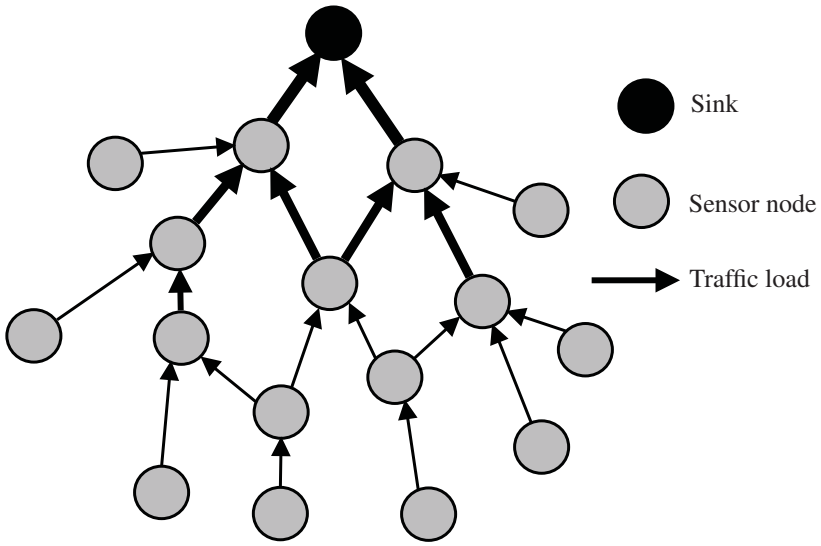


FIGURE 1.2: An example of convergecast Wireless Sensor Network. The thickness of the arrow between two nodes reflects the load of data traffic between them.

Convergecast WSNs are mainly used for data collection applications where nodes have to continuously sense their environment (fire monitoring, seismic activity monitoring, temperatures measurement, pollution monitoring) and forward the sensed data to the sink.

1.2 Wireless sensor networks lower layers

IEEE and IETF working groups have standardized several protocols in order to create a protocol stack for Low-power and Lossy Networks (LLNs) and Low Power Wireless Personal Area Networks (LP-WPANs), including WSNs. Following the concept of Enhanced Performance Architecture (EPA), authors of [6] argue that the number of layers in WSN can be reduced to five by leaving aside Session and Presentation layers of the Open Systems Interconnection (OSI) model. Figure 1.3 depicts comparison between layers of OSI model (a) and those of WSNs (b) presented in [6]. In this figure, we notice that WSNs have five layers, namely: physical, link, network, transport and application layers. In the following sections, we will briefly present physical, link and network layers.

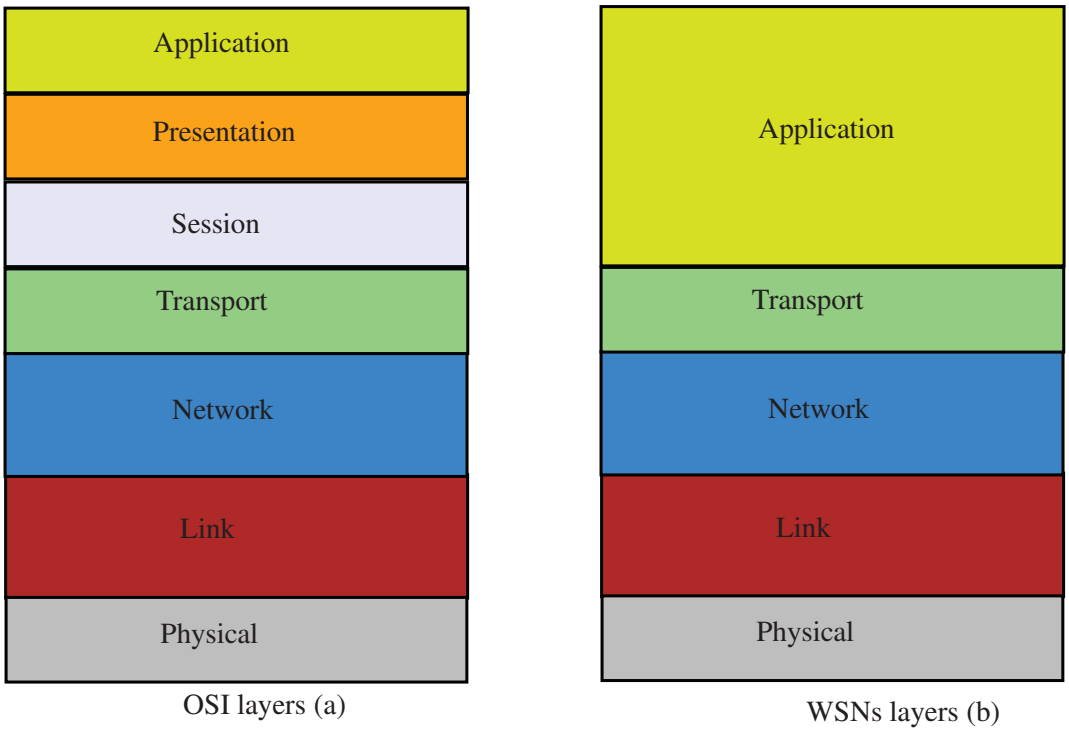


FIGURE 1.3: Comparison between OSI stack and WSNs stack.

1.2.1 Physical layer

In WSN, the physical layer is represented by the radio chip which main role is to receive and transmit data as an electromagnetic signal over the radio medium. Several WSNs physical layers follow the specifications of IEEE 802.15.4 standard [7]. This standard offers a maximum throughput of 250 kb/s when it operates on the shared and unlicensed 2.4 GHz band. Operating on this license-free frequency band, the IEEE 802.15.4 standard offers 16 orthogonal communication channels. That opens the way for multichannel communication in WSNs.

The unreliability nature of the radio link makes the physical layer of WSNs unreliable and sometimes asymmetric, that means the quality of a link is not always the

same in both directions. In this thesis, we do not focus on the physical layer, we are mainly interested in network and link layers.

1.2.2 Link layer

The wireless medium is a shared resource and nodes in the same communication range may interfere whenever they try to access the medium at the same time. Thus, the Medium Access Control (MAC) protocol of the link layer is in charge of regulating the access to the medium. It decides which node can use the medium, when and on which channel.

MAC protocols can be time-based relying on Time Division Multiple Access (TDMA) mechanism, or frequency-based using Frequency Division Multiple Access (FDMA) approach, or code-based following the Code-division multiple access (CDMA). It can also be random access based using the Carrier Sense Multiple Access (CSMA) algorithm or even hybrid combining some of these medium access schemes.

The IEEE 802.15.4 standard allows two types of medium access mechanisms that relies on random access: beacon enabled and non-beacon enabled modes. The latter case uses unslotted carrier sense multiple access with collision avoidance (CSMA/CA), whereas the former uses a slotted CSMA/CA algorithm with a time super frame structure. This algorithm makes each node use a randomized waiting time before trying to access the communication medium. The use of a random access mechanism helps to reduce interferences and collisions in the network. The CSMA/CA of IEEE 802.15.4 standard is widely used as a MAC protocol in WSNs. We use it in this thesis as a medium access protocol. The algorithms of both slotted and unslotted CSMA/CA are presented in appendix A.

1.2.3 Network layer

The network layer plays an important role in data transmission. It has the responsibility to construct paths between nodes in the network and routes data packets from source to destination nodes. In WSNs, the network layer is represented by routing protocols that have to build the network and guide nodes to route their data packets towards destination nodes and particularly towards the sink in case of convergecast data traffic profile. Depending on the needs of the application, the used routing protocol may be different. Thus, several routing protocols have been proposed for WSNs. Among them we can cite static or dynamic routing protocols. In the next chapter 2, section 2.2, we present in details routing protocols in WSNs and highlight some drawbacks of these existing protocols. That lead us to propose new routing protocols for convergecast WSNs.

1.3 Congestion and queue overflow issues in convergecast WSNs

In most cases, WSNs operate on the shared and unlicensed 2.4 GHz band. This band is also used by other communication technologies like IEEE 802.11 (Wi-Fi)

and IEEE 802.15.1 (Bluetooth). The coexistence of different technologies in the same frequency band leads to inter-network interferences in addition to intra-network interferences due to the shared medium [8]. Interference reduces network throughput and leads to congestion.

In convergecast WSNs, the traffic load is increasing as data packets travel towards the sink. Thus, sink neighbor nodes will receive an important amount of data packets to forward to the sink. Due to the constrained resources (computation, memory, energy and communication range) characteristics of sensor nodes and depending on the used routing policy, high data load can lead to congestion and possibly queue overflow for nodes that are not able to store and forward all the received packets. That provokes data loss leading to reduced network throughput.

In this thesis, we focus on how to avoid congestion and queue overflow in high data traffic load network by providing traffic load balancing routing protocol that copes with convergecast multihop WSNs.

1.4 Motivation and Assumptions

WSNs are being used in several domains such as environmental, industrial and medical. Their ease of deployment, low cost and low energy mode of operation make them an attractive solution. When it comes to network performance, the needs can vary from one application to another according to the goal of the application. For low rate and smart home automation applications, in most cases, the available solutions based on the ZigBee standard [9] are satisfactory. Other applications might have end-to-end delay constraints such as industrial Supervisory Control And Data Acquisition (SCADA) applications. Where commands should be taken into account by nodes in a timely and deterministic manner based on more deterministic standards such as ISA100.11a [10] and WirelessHART [11]. In some cases, applications have high data rate requirements [12], such as video surveillance [13] or vibration measurements [14] where a special care should be made for MAC and routing protocols in order to offer acceptable network performance according to application needs.

Research in traditional WSNs aims to satisfy simple applications, such as detecting the occurrence of an event and sending a notice information to the sink. Another can be the periodic (every hour, every minute or every second) transmission of sensed data as temperature or pollution level. The requirements of these traditional applications can be fully satisfied with low data rate. However, nowadays, more and more WSNs are required to be used in such as audio and video field, which allows retrieving video and audio streams, still images, and scalar sensor data [15]. Although, there is a great progress in video and audio signal processing, large amount of data are still required in the transmission in above media streaming field as well as some critical control applications, which means that high data rate WSNs is required.

The use of multichannel MAC protocols significantly improves the throughput and increases the overall data traffic load in the network as it was shown in [16]. The negative consequences of high traffic load is the creation of congestion and possibly packet queue overflow especially in the nodes close to the sink in convergecast scenarios. So, increasing traffic load in the network should be coupled with a traffic load

balancing routing to avoid congestion and packet loss.

The goal of this thesis is to propose an efficient and robust communication protocol for applications that suffer from congestion and queue overflow, mainly at sink surrounding neighbors, in high data rate convergecast WSNs. We focus on convergecast network scenario because, it is the most used one in data collection applications. In this thesis, we consider the following hypotheses:

- nodes are static in the network;
- each node has one radio transceiver except the sink which may have several;
- we refer to low data rate when the average data generation rate by each node per second is lower than 5;
- we called high data rate when the average data generation rate per node per second is higher or equal to 5.

1.5 Contributions

In this thesis, our contributions are related to the network layer. We proposed and analysed routing protocols which use a Collaborative Load Balancing Algorithm (CoLBA) to avoid congestion and queue overflow in convergecast WSNs. CoLBA algorithm is applied in both Single channel (S-CoLBA) and Multichannel (M-CoLBA) WSNs. These routing protocols balance the traffic load and enhance the network performances by avoiding to loose data due to congestion.

The main contribution of both S-CoLBA and M-CoLBA is improving the network throughput by fairly distributing traffic load and avoiding to loose packets due to queue overflow thanks to congestion aware dynamic routing metric. M-CoLBA uses synchronization periods for topology and routing informations exchange between nodes. These synchronization periods lead to longer data packet end-to-end delay. So, we enhanced M-CoLBA by avoiding the costly synchronization periods and this enhanced version is called ABORt for Acknowledgement-Based Opportunistic Routing.

ABORt is a multichannel load balancing routing protocol that uses piggybacking scheme to disseminate routing metric. Unlike M-CoLBA, it does not need synchronization periods for network and routing informations exchange. Hence, it avoids time wastage in order to construct routes. ABORt uses CoLBA load balancing scheme and reduces data packet end-to-end delay thanks to piggybacking routing information in ACK frames. It optimizes the network throughput and evaluation results show that it outperforms several existing routing protocols.

1.6 Structure of the thesis

This thesis is conducted with the financial support of the European Regional Development Fund (FEDER) program of 2014-2020, the region council of Auvergne, and

the Digital Trust Chair of Clermont Auvergne University.

The manuscript is organized in five chapters. The first chapter presents an introduction to Wireless Sensor Networks, as well as the motivations and contributions of this thesis. The second chapter presents an overview of MAC and routing protocols in WSNs, with a focus on multichannel medium access protocols, congestion avoidance and traffic load balancing routing protocols. Chapter 3 presents the contributions of this thesis. We begin with S-CoLBA and M-CoLBA, routing protocols that use dynamic routing metric to balance traffic load. Then we present M-CoLBA enhanced version called ABORt that relies on piggybacking mechanism to reduce data packets end-to-end delays and to optimize the network overhead. In Chapter 4, we present the performance evaluation in both simulation and experiment (only for S-CoLBA) of our contributions where we compared obtained results with some existing protocols in the literature. Finally, Chapter 5 concludes this thesis by summarizing our contributions, presenting remarks and opening up some perspectives to forward this work.

Chapter 2

Routing in multichannel wireless sensor networks

In WSNs, the use of multichannel MAC protocol significantly increases the overall data carried load[1] in the network and enhances the throughput as it was shown in [17]. In many-to-one data collection (convergecast network scenario), all collected data will converge from leaf nodes towards the sink thanks to multi-hop data transmission. The concentration of paths (several nodes share the same path) used to forward data towards the sink leads to congestion, collisions and packet losses mainly in nodes closed to the sink. In most cases, congestion occurs due to the lack of load balancing in the network, because some nodes are overloaded while others are under-loaded. Data transmission from leaf nodes towards the sink is ensured by the routing protocol. So, the used routing policy has an impact on the network performance and must be aware about congestion. Increasing the carried load in WSN thanks to introducing multichannel MAC protocol, should be coupled with a congestion avoidance mechanism. Mainly a traffic load balancing routing which helps to avoid overloading some nodes while others in the same hop are less used.

The remainder of this chapter is organized as follows. In section 2.1 we summarize the existing multichannel MAC protocols for WSNs. We detail different approaches used for static, semi-dynamic and dynamic channel allocation[2] protocols. Section 2.2 presents multichannel routing protocols. Finally, section 2.3 overviews the congestion avoidance approaches used in WSNs.

2.1 Multichannel MAC protocols in WSNs

Interference and collisions are serious issues encountered in single channel WSNs. They lead to packet loss that decreases the network throughput. Introducing multichannel communication in WSNs helps to optimize the network bandwidth by avoiding interference and collisions.

Several multichannel MAC protocols using various channel allocation schemes have been proposed for WSNs. Channel allocation methods allow nodes to decide which channel should be used for communication and can be classified according to the frequency of channel switching. In what follows we classify channel assignment protocols where we adopt a classification based on the frequency of channels assignment. Three families of channel assignment methods are identified: static

[1] The carried load is the amount of data carried by the network

[2] Channel allocation and channel assignment are used for the same meaning in this work.

assignment, semi-dynamic assignment, and dynamic assignment. In the following subsections, we detail the three channel allocation approaches by summarizing the main multichannel MAC protocols that are based on these techniques.

2.1.1 Static channel allocation MAC protocols

In static channel allocation, sensor nodes are often grouped in different clusters and a common communication channel is assigned to each cluster. To prevent inter-cluster interference, different and no adjacent channels are allocated to neighboring clusters. The clustering and channel allocation is done once during the network initialization phase. In the same cluster, nodes communicate using the same channel and applying the CSMA/CA algorithm to access the medium. Protocols presented in [18] and [19] perform static channel allocation.

The authors of [18] propose a Tree-based Multi-Channel Protocol (TMCP) for data collection applications. TMCP is a MAC protocol that performs a logical partition of the physical network into multiple sub-trees. For each sub-tree, a common communication channel is allocated. Authors argue that using adjacent channels by neighboring sub-trees leads to interference. To avoid inter-tree interference, neighboring sub-trees are assigned different and non-adjacent channels. All nodes in the same sub-tree use the same communication channel for data transmission. The medium access in each sub-tree is managed by the CSMA/CA algorithm. Network topology and routing informations are exchanged between nodes in each sub-tree. The collected data from leaf nodes of each sub-tree is forwarded to the root of the sub-tree which transmits them to the sink where data are processed. As we can see in figure 2.1, the sink is equipped with multiple radio interfaces (each color represents an interface, so we have 3 interfaces in the example of figure 2.1) and each transceiver works on one different channel.

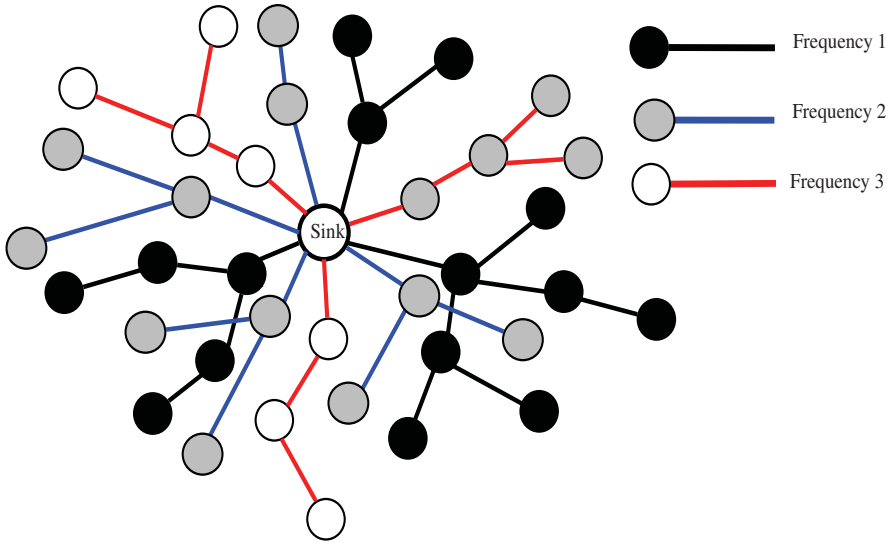


FIGURE 2.1: TMCP channel assignment policy. In the presented scenario, three orthogonal and non adjacent channels are used with frequencies 1, 2 and 3 [18].

The efficient partition of the network into several sub-trees is an NP-complete problem that authors solved using a greedy algorithm. TMCP is mainly efficient in dense networks with a small number of channels. Simulation results using GloMoSim simulator and experiment with Micaz nodes show that TMCP improves the throughput with a reduced packets latency. The drawback of TMCP is that nodes in different sub-trees have no possibility to directly communicate with each other. Moreover, as nodes in the same sub-tree use the same channel, the intra-tree interference remains possible and may affects the performance.

In [19], authors propose a Multi-Channel Real-Time (MCRT) MAC protocol for WSNs that features a flow-based channel allocation strategy. The aim of MCRT is to achieve a bounded end-to-end communication delay for every data flow. To do so, network partitions are formed basing on many-to-one data flows and channel is allocated to each partition. To avoid interference, MCRT splits the network following a convergecast communication profile. After the network partitions are formed, it allocates different channel to each data flow that aims to minimize the channel contention among different flows. The channel allocation algorithm is based on heuristic which tries to maximize the number of disjoint paths in the network that can meet the specified end-to-end communication delay. Data collected in each partition is sent towards a central node with an end-to-end delay constraint. To achieve the delay constraints for every data flow, the channel allocation problem has been formulated as a constrained optimization problem and proved to be NP-complete. MCRT uses a constraint optimization approach to solve this channel allocation NP-complete problem.

Simulation results using NS-2 simulator show that the network is able to better respect the end-to-end constrained delay using MCRT compared to three existing real-time protocols. It also outperforms the other protocols in terms of energy consumption.

The main advantage offered by static channel allocation is the simplicity of the implementation. Protocols do not need to address the dynamic issues due to topology variation or the channels switching. However this channel allocation scheme is not suitable for dynamic network conditions such as frequent link failures or variation of the network topology.

2.1.2 Semi-dynamic channel allocation MAC protocols

Unlike static channel allocation, the main idea of semi-dynamic channel allocation is to attribute one fixed channel to every node and also permits each node to switch between available channels to communicate with its neighbors. The process of channel allocation starts at the network initialization phase but channels can be reallocated whenever needed. Thus, data traffic conditions or interference level can trigger the channel reallocation process. Semi-dynamic channel allocation family includes many protocols such as the ones presented in [20] and [21].

In [20], authors propose a Hybrid Multi-Channel MAC protocol (HMC-MAC) with a multi-interface sink for WSN. HMC-MAC is a semi-dynamic multichannel

MAC protocol where each node selects a reception channel in its 3-hop neighborhood. Nodes are organized in groups and the channel allocation scheme considers nodes inside each group. In order to select the most convenient channel in its 3-hop neighborhood, the node with the highest priority (the priority is related to nodes IDs) searches for free channels in its 3-hop neighborhood in its group, if it does not find one, it looks for a free channel in its 2-hop neighborhood in its group. In case there are no available channels in its 2-hop neighborhood, the node checks for a free channel in its 1-hop neighborhood inside its group. Finally, if it does not find a free channel, it randomly chooses a channel among the list of less used channels in its 1-hop neighborhood in its group. Whenever node chooses a reception channel, this information is shared in its 3-hop neighborhood. To transmit data, each node switches to the reception channel of the next hop neighbor and then forwards the data. The network operation is organized in cycles and the cycle has three phases. A beacon phase in TDMA mode during which nodes exchange information about the topology and channels allocation. The second phase is dedicated to data transfer. In the final phase all nodes are inactive before the cycle resumes again. One of the particularities of HMC-MAC is using a multi-interface sink for receiving data, something which prevents a bottleneck at the sink node.

Results of simulations using NS-2 simulator show that, compared to CSMA/CA, static channel assignment, and 2-hop channel assignment, HMC-MAC improves the network throughput and reduces the number of dropped packets due to medium access problem. The main drawback of HMC-MAC is losing data packets at nodes close to the sink due to queue overflow.

In [21], authors propose a Multi-Channel Lightweight Medium Access Control (MC-LMAC) protocol that uses a TDMA-based approach for channel allocation. Each node is assigned a time slot and a channel to communicate on. An example of MC-LMAC timeslot and channel selection is presented on figure 2.2 where node marked with "?" is searching for a timeslot and other nodes are marked by timeslot/frequency pair that they are using.

To avoid interference, MC-LMAC ensures that the same slot/frequency couple is not allocated to two neighboring nodes at the same time. The number of frequencies is 2 (F1 and F2) and the number of timeslots per frame is 5 where 1 means the timeslot is occupied and 0 means free. The process of slot/channel selection by each node is distributed. Each time slot is divided into a control period and a data transmission period. The control period is used to share control messages during which all nodes switch their radios on the same channel and inform the destination about incoming packets and the channel that they will use. After receiving this information, destination nodes will switch their radio interface to the right channel and wait for data packets.

The performance of MC-LMAC is evaluated by simulation using GloMoSim simulator, and compared to MMSN [22]. Performance evaluation results show that, MC-LMAC outperforms MMSN protocol in terms of packet delivery ratio and network throughput in highly loaded networks. On the other hand, MC-LMAC suffers from control message overhead.

The main interest of semi-dynamic channel allocation over the static one is that,

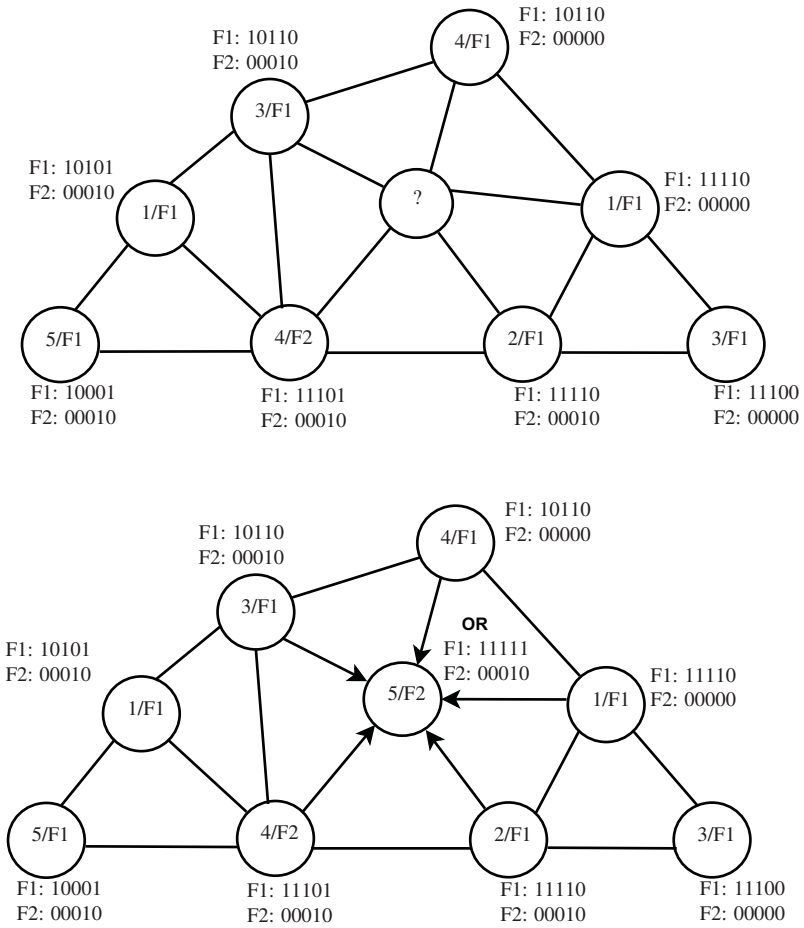


FIGURE 2.2: An example of MC-LMAC timeslot and channel assignment. The node with"?" is looking for the couple timeslot/frequency.

each node can communicate with its neighbors. It also suppresses the network partitioning and isolation. However, semi-dynamic channel allocation requires an efficient coordination between sender and receiver to tune their radio on the same channel at the same time for transmission. Channel switching might also cause deafness[3] problem and makes node to miss data packets destined to it when it is communicating with a neighbor on other channel. Some approaches may be used to mitigate the need of strict nodes coordination and the deafness problem. Without nodes coordination, MAC protocols with multiple transmission attempts like CSMA/CA may be used to mitigate the need of coordination between nodes. It is also possible to make nodes tune their radio on their reception channel whenever they are not transmitting and also after each successful transmission. That will help to reduce the deafness problem and increase the chance of successful transmissions.

[3]Deafness occurs when a sensor node could not receive on its reception channel because it is transmitting on an other channel.

2.1.3 Dynamic channel allocation MAC protocols

The dynamic channel allocation approach is the third and last approach in our channel allocation classification. According to this approach, nodes may switch channels for every transmission. The channel selection can be status-based or measurement-based. The measurement-based technique uses the Signal to Interference plus Noise Ratio (SINR) value before deciding on which channel to switch. In status-based channel selection technique, each node keeps track of status of the channel which may be busy or idle according to the information given by the received control message. In what follows, we describe some of the main protocols using dynamic channel allocation.

To meet some requirements in industrial applications such as reliability of wireless data transmission, the 802.15.4e [23] amendments to the original IEEE 802.15.4 standard introduce several multichannel MAC protocols. These protocols include Deterministic and Synchronous Multi-channel Extension (DSME) and Time Slotted Channel Hopping (TSCH) protocols. These amendments aim to improve throughput and robustness of wireless links already proposed by IEEE 802.15.4. TSCH protocol uses multichannel communication to reduce the effects of interference and multi-path fading through TDMA, channel hopping and scheduled transmissions. Channel hopping helps to avoid blocking wireless link because of inter-network or intra-network interference. TSCH operates in cycles known as slotframe where each slotframe is divided into fixed time periods known as timeslots presented on figure 2.3. Each timeslot is only used to transmit one packet and its acknowledgement. TSCH is an efficient protocol with energy gains, but connecting it to upper layers of the communication stack is not a straightforward task. Figure 2.4 shows an example of a possible link schedule for data collection in a simple 9-node network scenario in a tree topology. In this example, the slotframe consists of 4 timeslots and there are only 5 channels available. Thanks to the multichannel approach used by TSCH, 8 transmissions have been accommodated in a time interval corresponding to 4 timeslots.

The DSME protocol is aimed to support both industrial and commercial applications with stringent requirements in terms of timeliness and reliability. To this end, it combines contention-based and time-division medium access, and offers two different channel diversity modes. It is specifically designed for multi-hop and mesh wireless networks. To enhance the radio link reliability, the DSME MAC protocol uses two types of channel diversity schemes, channel hopping and channel adaptation, while TSCH protocol employs only channel hopping scheme.

In [24] authors propose Adaptive Times-Slotted Channel Hopping (A-TSCH) protocol. A-TSCH is an improved version of IEEE 802.15.4e TSCH protocol. It is oriented to cope with dynamic wireless conditions. A-TSCH uses dynamic blacklisting mechanism relaying on the hardware-based channel energy measurement to select communication channel with less interference. It operates on top of the TSCH protocol. For each TSCH slotframe, it reserves two timeslots to perform energy samplings on the operating channel of these timeslots. No communication will happen in these timeslots therefore the gathered values of energy samplings can be considered as noise level on those channels. The sampling results are used to assign a quality

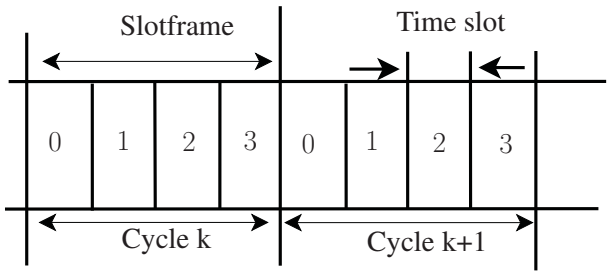


FIGURE 2.3: TSCH slotframe and timeslots.

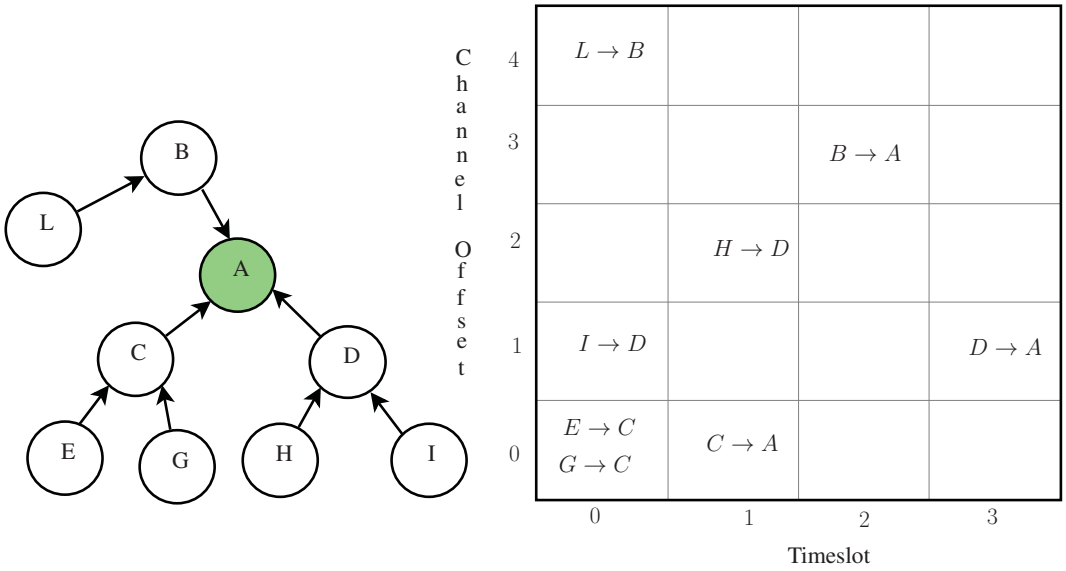


FIGURE 2.4: An example of TSCH possible link schedule for data-collection with 9-node tree-topology scenario.

factor to each channel and thus channels can be ranked according to their availability. A-TSCH improves the reliability of channel hopping scheme and provides better protection from interference in WSNs.

In [25], a Predictive Channel Scanning and Switching (PCSS) algorithm is proposed to fight against interference from Wi-Fi access points and other terminals. To avoid interference due to Wi-Fi, a channel hopping formula is used and the hopping distance is derived from Wi-Fi and IEEE 802.15.4 channel frequencies. PCSS first considers Wi-Fi interference, searches high quality channel and then forces the node to switch to that channel. At the network start phase, each cluster coordinator chooses a default communication channel according to its *ID*. A threshold value of RSSI is established by the network manager and each coordinator determines the channel availability by comparing the computed RSSI value to the threshold value. In case

the coordinator finds that the used channel becomes busy, it will be changed to a channel which is not being used by the nearby cluster. In performance evaluation, simulations results show that PCSS is efficient in term of average scan time required to find available channel. PCSS approach permits to reduce redundant procedures for the channel switching that permits to reduce delay. However, [25] did not present any results concerning packets end-to-end delay.

With dynamic channel assignment the interference level can be reduced, but nodes have to frequently share control information globally or in a large neighborhood to negotiate channel allocation and coordination. Thus, doing so causes considerable communication overhead to WSNs that affects the efficiency of the network.

In table 2.1 we used an array to summarize WSNs multichannel MAC protocols where each protocol is designed by its acronym or its reference number. We compared the following 7 characteristics of all presented protocols:

- The channel allocation method (**Alloc method**), that helps to know if the protocol is static, semi-dynamic or dynamic.
- The usage of control channel (**Cntrl chnn**) for network topology or routing metric value exchange.
- The channel allocation implementation (**Impl**), that gives an idea in relation to the channel allocation like centralized or distributed in the network.
- The synchronisation (**Syn**), that helps to determine if nodes need synchronization phase or not during their operation.
- The medium access (**Med acc**) approach in case two nodes have to use the same channel at the same time. The access may be regulated by contention-base (CSMA) or deterministic way like TDMA.
- The broadcast support (**Broad suprt**) to determine if the protocol needs to frequently broadcast some informations destined to its neighborhood.
- The main objective (**Objctv**) of the protocol that specifies its aim compared to the others.

Observing table 2.1, we notice that except the MASN (Multichannel Access for Sensor Networks) [27], all other protocols have a distributed channel selection approach. Indeed, in large scale WSNs, adopting a centralized channel allocation is not easy to manage. That may generate too much overhead during the channel allocation process. The comparative study also reveals that most of the presented protocols require synchronization phase and broadcast support. Usually, synchronization phase and broadcast support are used for network service information (like network topology and channel availability) sharing. However, synchronization is time consuming and hard to achieve in a large scale WSNs. We also notice that the semi-dynamic channel allocation is more flexible as it allows nodes to switch from their own channels to others. In addition to the fact that it requires less strict coordination than

Protocols	Alloc method	Cntrl chnn	Impl	Syn	Med acc	Broad suprt	Objctv
[26]	Static	Required	Distributed	Required	CSMA/CA	Required	Optimize energy consumption and overhead.
TMCP [18]	Static	Not required	Distributed	Not required	-	Inside branches of the tree	Efficient data collection.
MCRT [19]	Static	Not required	Distributed	Not required	CSMA	Not required	Improve throughput with bounded end-to-end delay.
MMSN [22]	Semi-dynamic	Not required	Distributed	Required	Slotted CSMA	Required	Increase parallel transmissions.
MC-LMAC [21]	Semi-dynamic	Not required	Distributed	Required	LMAC	Required	Reduce interference and improve throughput.
MASN [27]	Semi-dynamic	Required	Centralized	Not required	CSMA	Required	Improve the global throughput.
HMC-MAC [20]	Semi-dynamic	Required	Distributed	Required	TDMA and CSMA	Required	Increase throughput with low interference.
PCSS [25]	Dynamic	Not required	Distributed	Not required	-	Inside the cluster	Avoid Wi-Fi activities and other devices interference.
A-TSCH [24]	Dynamic	Required	Distributed	Required	TDMA	Required	Avoid interference dynamic wireless conditions.
TSCH [23]	Dynamic	Required	Distributed	Required	TDMA	Required	Improve throughput and robustness of wireless link.
DSME [23]	Dynamic	Required	Distributed	Required	CSMA and TDMA	Required	Improve link reliability and throughput.

TABLE 2.1: Taxonomy of channel allocation methods in WSNs.

what dynamic channel assignment needs. In static channel assignment, the interference between nodes sharing the same channel remains unsolved. Moreover, it is not

suitable in dynamic network conditions due to links failure.

In our contribution, we will adopt semi-dynamic channel allocation with distributed channel selection.

The main idea of using multichannel MAC protocols in WSNs is to improve the throughput and achieve a high traffic load. That is possible thanks to the reduction of contention and interference, and collision avoidance. However, multichannel MAC protocols lead to a new challenge when we come to the broadcast support. As nodes do not listen permanently on a common channel, having a broadcast support becomes a new issue which is unknown in single channel networks. In multichannel MAC protocols, we have to find a way for exchanging network information (network topology and routing metric) between nodes in the network. This issue should be considered when designing a multichannel MAC protocol. In most existing multichannel MAC protocols, the problem of broadcast support is solved by introducing synchronization phases in the network or using signalling channel. That consumes energy and time leading to longer end-to-end delay in addition to the fact that it is hard to achieve in a large scale WSN.

Optimizing the MAC layer by using a multichannel MAC protocol helps to reduce contention and collisions. When node accesses the medium, the next step is to forward data packets to a next hop neighbor according to the used routing policy in the network. In multi-hop WSNs, the routing protocol plays an important role in data transmission. It defines a rule that must be followed by each node to select its next hop neighbor to which the next data packet is forwarded towards the sink node. In multichannel WSNs, several routing protocols have been proposed for data transmission. In the following sub-section 2.2, we will present the main routing techniques used in multichannel WSNs for data transmission.

2.2 Multichannel routing protocols in WSNs

In convergecast WSNs, data collected by all nodes in the network is destined to a common final destination, the sink node. Due to the limited communication range of sensor nodes and depending on the topology, some nodes are out of range of the sink. Theses nodes have to use multi-hop communication to transmit their data towards the sink. The multi-hop communication is governed by a routing protocol which defines for each node (out of range of the sink) the next hop neighbor to which the next data packet will be forwarded. Each routing protocol relies on routing metric to select the next hop neighbor that copes with the routing policy.

Routing metrics used in WSNs may be static or dynamic throughout the operation of the network. In the static routing scheme, each node always has the same next hop neighbor to which the data destined to the sink is forwarded. One of the most know static routing metric is the hop count metric [28] where each node finds the shortest path in terms of number of hops from it to the sink. However, in high traffic load WSNs, the static routing is less efficient and leads to congestion and data loss. In the dynamic routing, the value of the routing metric changes over the time that makes nodes to frequently change their next hop neighbor towards the sink. The routing metric may be node-based (as the residual energy of the node or its queue occupancy

rate) [29], [30] or link-based (the expected transmission count of the link or its loss rate) [31] or even related to both link and node [32], [33]. In multichannel WSNs several routing protocols have been proposed for multi-hop data transmission. In what follows, we present some routing protocols proposed for multichannel WSNs.

In [29], authors propose a Multi-Channel Collection (MCC) protocol that aims at optimizing data collection in multichannel WSNs. MCC is a node-based time-scheduled routing protocol with globally synchronized TDMA scheduling. It uses a Capacitate Minimal Spanning (CMS) tree heuristic to build a balanced routing tree for multi-hop data transmission. An example of 10 nodes network scenario used for data dissemination is presented in figure 2.5, where each node scheduled transmission slots are labelled. Note that nodes scheduled transmission slots are relative to their parents first transmission slot. On the example shown in figure 2.5, we notice that nodes are equally distributed among the tree branches belonging to the sink node. The basic idea of the used CMS tree construction algorithm is to minimize the size of each branch. That is done by considering the possible growth that a node might bring when it joints a branch.

Through experiments using Tmote Sky nodes, MCC shows that the network throughput can be improved by mixing channel allocation, tree routing approach and time scheduling. MCC increases the overall network throughput but requires a precise time synchronization, which is hard to obtain in large scale WSNs.

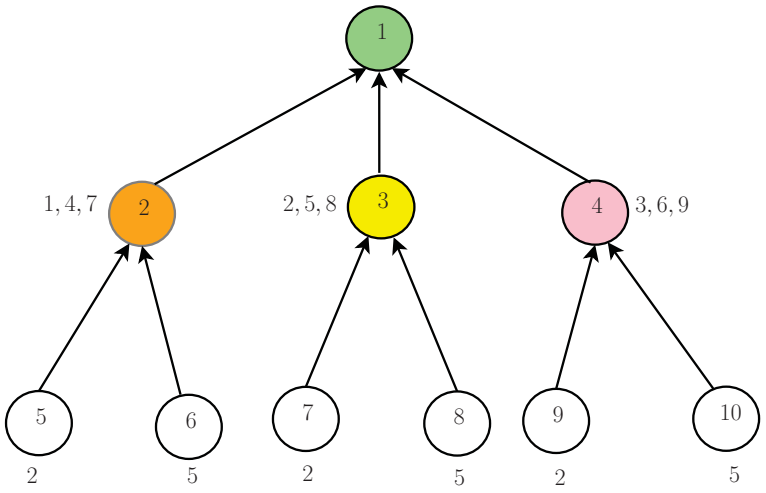


FIGURE 2.5: An example of MCC balanced routing tree with 10 nodes network scenario. Except the node with *ID* 1, the other nodes scheduled transmission slots are labelled.

In [30], authors propose a Hybrid Multi-Channel (HMC) protocol coupled with a tree-based routing scheme for WSNs. The routing scheme in HMC is node-based and it assumes a network where the sink node has 3 radio interfaces and other nodes have only one radio interface. The routing approach follows a tree where each node selects a parent which becomes at the same time its next hop neighbor. The parent of a node is its neighbor that has less children among all its potential next hop neighbors. Nodes activity is organized in cycles and each cycle is divided in 3 periods. The operation cycle of HMC is presented in figure 2.6, where we have a synchronisation

period ([T0;T1]) followed by data exchange period ([T1;T2]) and an inactive period (T2;T0]) where nodes sleep to save energy. During data exchange periods, the data transmission is done by group of nodes and each node has to transmit its generated or forwarded data packet to its parent. The parent selection policy aims to balance the traffic load. That is why each node chooses as a parent a next hop neighbor that has the minimum number of children among its potential next hop neigbors. The network topology is presented on figure 2.7 where nodes are grouped in sub-trees called branches. Nodes in the same branch may belong to different data forwarding groups.

Simulation results using NS-2 simulator show that HMC protocol improves the network throughput and reduces the number of dropped packets due to medium access problem. However, HMC suffers from high end-to-end delay due to the synchronization phase and high number of packets lost due to queue overflow in the sink neighborhood.

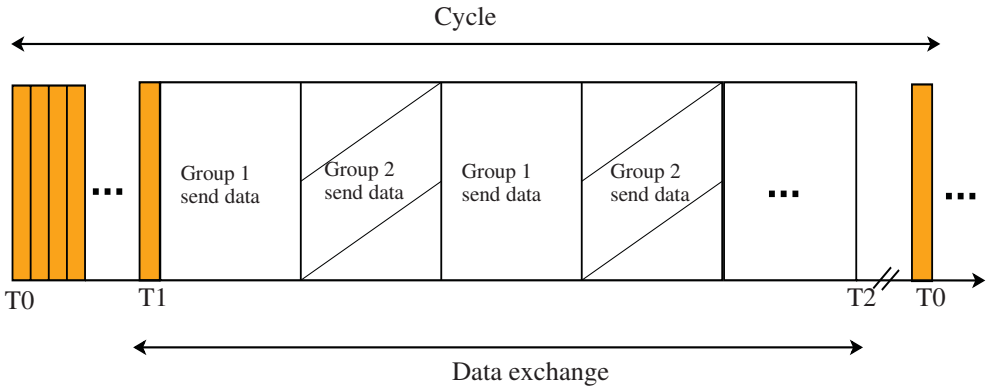


FIGURE 2.6: HMC global cycle. [T0 ; T1] is a synchronization period, [T1 ; T2] is dedicated for data exchange and [T2 ; T0] is sleeping period to save energy.

Authors of [31] propose a Joint Routing and Channel Assignment (JRCA) scheme for wireless mesh networks. JRCA uses a link-based routing metric and aims at improving the quality of the communication in multi-interface and multi-sink wireless mesh networks. The routing scheme is based on the multi-hop route quality in terms of end-to-end probability of success and the delay experienced by packets.

Performance evaluation using NS-2 simulator shows that JRCA improves network throughput, delivery ratio and end-to-end delay compared to a single channel protocol and some existing multichannel protocols. However, JRCA is proposed for wireless mesh networks, applying it in WSNs needs adaptations and modifications due to the limited capacities of sensor nodes. These modifications may concern several aspects such as queue length of 200 packets with a packet size of 1000 bytes, in addition to throughput and channel access differences. Moreover, JRCA suffers from overhead and this aspect was not evaluated.

In [19], authors propose a Multi-Channel Real-Time (MCRT) joint MAC and routing protocol designed for real-time communications in WSNs. MCRT is both link and node based routing protocol. It uses multichannel data transmission to

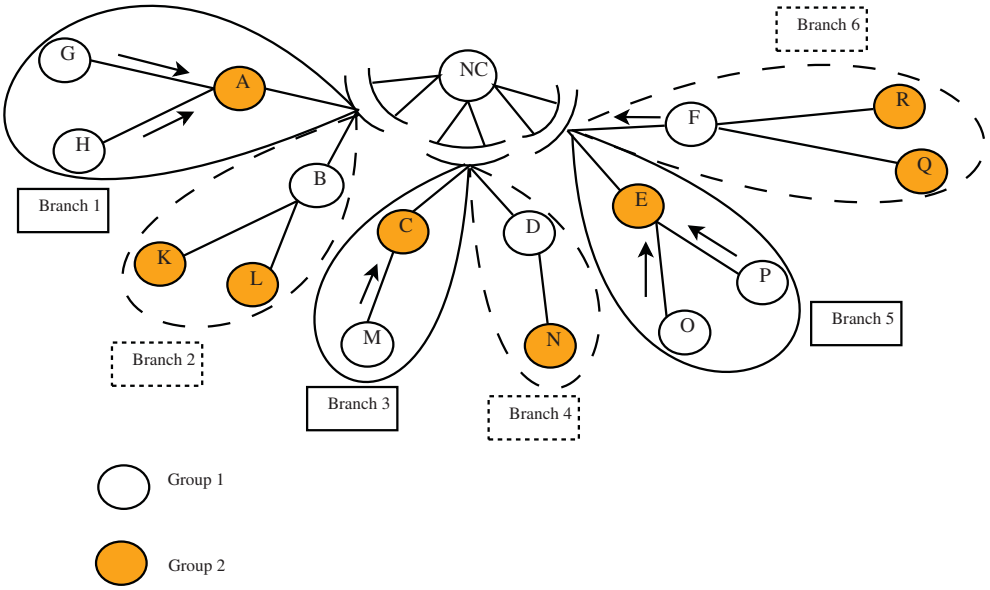


FIGURE 2.7: HMC network segmentation for data forwarding. In this example we have two groups and six branches.

achieve bounded end-to-end communication delay for every data flow in the network. A data flow is composed of a source node and the destination. To establish data flows, the network is partitioned to a set of disjoint paths from source nodes to the destination. The network is organized into several partitions based on data flows such that interference among different data flows can be minimized. That helps to avoid collisions which negative consequence is data retransmission leading to longer end-to-end delay.

MCRT routing scheme is based on packet delay between each node and its upstream neighbors combined with the energy consumption. For each data packet to transmit, the end-to-end deadline of the data flow is inserted in the packet. Whenever a neighbor receives the data packet, it needs to forward the packet to a next hop neighbor based on whether the neighbor can meet the delay requirement of the packet with minimum energy consumption.

Simulation results using NS-2 simulator show that MCRT is able to achieve better end-to-end delays and reduced energy consumption compared to three other existing protocols. However MCRT is not tested in a high traffic load network. In the performance evaluation scenario, traffic is generated by only 10% of nodes in the network generating only 1 packet every 4 seconds. This evaluation scenario does not give an idea about MCRT efficiency in high traffic load situation.

Authors of [33] and [34] propose a Distributed Routing and Channel Selection scheme (DRCS) for multichannel WSNs. DRCS is a joint channel assignment and quality aware routing protocol that aims at improving network lifetime. It uses both node and link based routing approach. The channel assignment and routing are based on the remaining battery level of next hop neighbors and the link quality towards these next hop neighbors according to the ETX metric. Before each transmission, each node needs to select a transmission channel and then chooses a next hop among

all its neighbors that has the best path metric towards the sink according to ETX metric value. An ETX for a link is the expected number of transmission attempts required to deliver a packet successfully over the link. To estimate the quality of a path, DRCS makes the sum of ETXs on each link between source and destination nodes on the route. DRCS optimizes the network energy consumption by reducing the energy consumed from overhearing.

The performance evaluation shows that the proposed approach reduces energy consumption by avoiding overhearing and thus improves the network lifetime. However, DRCS was evaluated under low traffic rate. The performance of DRCS in high traffic load scenario is unknown. Specially its ability to avoid congestion and data loss due to queue overflow is not presented.

Thanks to multichannel routing protocols that allow parallel communications and reduce contention and collisions, high traffic load can be introduced in WSNs. With heavy traffic load, the risk of congestion becomes higher during the data transmission from leaf nodes towards the sink node, mainly at sink surrounding neighbors. Indeed, sensor nodes have limited buffer size and operate on low power computation and consumption Central Process Unit (CPU). In case of heavy traffic load, nodes taking part in the routing can be overloaded and may not handle all the offered traffic load that leads to congestion and to data loss. The network efficiency is affected by data loss because it reduces the delivery ratio and leads to miss some information. Several routing approaches have been proposed to avoid or mitigate congestion in WSNs. Traffic load balancing and congestion control mechanisms are introduced to fight against congestion in WSNs data transmission. In what follows, we present the main approaches used for traffic load balancing and congestion avoidance in WSNs.

2.3 Load balancing and congestion avoidance in WSNs

In data collection WSNs with convergecast profile, introducing multichannel routing protocol helps to have a high carried traffic load in the network. However, due to the constrained resources (computation, memory, energy and communication range) characteristics of sensor nodes, high data load may lead to queue overflow and packet loss because of congestion. In WSNs, the lack of load balancing, where the traffic overloads some nodes while others are under-loaded increases the risk of congestion.

Introducing high traffic load in WSNs should be coupled with a congestion avoidance approach. The congestion issue is generally handled by the routing protocol using techniques based on traffic load balancing or rate control scheme. In the state of the art, several protocols aim at avoiding or alleviating congestion in wireless networks either by load balancing or traffic source flow control or both.

In this section, we present a quick overview on handling congestion issues in wireless networks. We respectively present routing protocols based on traffic rate control and data aggregation, traffic load balancing routing protocols and protocols using both load balancing and rate control to fight against congestion.

2.3.1 Traffic rate control and data aggregation to fight against congestion in WSNs

In convergecast WSNs, nodes have to collect data in their neigborhood and transmit the collected information towards the sink node. Due to the limited communication range, most of nodes are out of range of the sink and have to use multi-hop transmission to forward they collected data. The many-to-one and multi-hop profile of data traffic lead to increased traffic intensity as data packets move closer towards the sink. The concentration of the traffic on some nodes leads to congestion. To fight against the experienced congestion, some protocols as [35] and [36] proceed to traffic rate control, while others as [37] use data aggregation to handle the congestion issue.

In [35], COngestion Detection and Avoidance (CODA) protocol is proposed. It is a routing protocol which is based on two methods to detect congestion in the network. In the first method, sensor nodes in the middle of the routing path monitor their input data traffic. The input traffic is compared to a specified threshold and the result is relayed to the sink node. In the second method, the sink node monitors its own input data traffic to detect traffic congestion. In both cases, when congestion is detected, sink node uses acknowledgement messages to ask nodes detecting the events to reduce their data sensing rate. The reduction of the sensing rate by source nodes helps to alleviate the congestion. Authors argue that without link-layer acknowledgements, buffer occupancy or queue length cannot be an accurate indicator of congestion. This assertion is illustrated on figure 2.8 where a wireless network with five nodes is presented. Each node queue and channel loads are presented. We notice that the channel may be overloaded while the queue is not. Using this network scenario, simulation results show that the buffer occupancy does not provide an accurate indication of congestion even when the ACK is enabled except in some isolate cases where the queue is empty or about to overflow.

Performance evaluation using NS-2 simulator showed that CODA can improve the performance of data dissemination applications by mitigating congestion zones. The main drawback of this protocol is that the sink node will miss events that were not allowed to be sampled or transmitted by event detecting nodes.

In [36], authors proposed an Event-to-Sink Reliable Transport protocol (ESRT). This protocol aims at improving the reliability of event data transmission in WSN scenario as illustrated in figure 2.9. In ESRT, all sensor nodes monitor their internal buffers to detect traffic congestion. When congestion is detected, the result is sent to the sink node using a packet format presented in figure 2.10, where the CN (for Congestion Notification) bit is marked to inform that congestion is experienced. Receiving congestion message, the sink broadcasts the congestion state to all other sensor nodes. After neighbors received the broadcast message, each node reduces its data transmission rate. Due to the reduced data traffic, the congestion problem is gradually alleviated.

Simulation results using NS-2 simulator show that ESRT has an efficient self-configuring aspect that can cope with random dynamic topologies which are frequently encountered in WSN applications. However, since ESRT scheme did not distinguish the event types while undergoing the traffic congestion, the entire flow

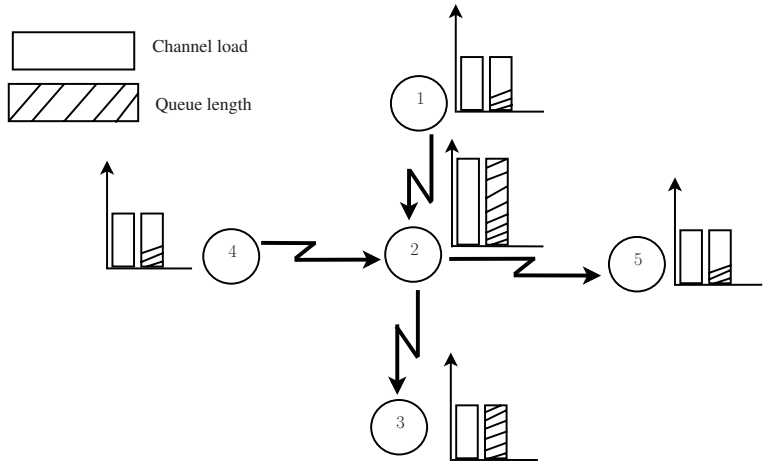


FIGURE 2.8: An example of IEEE 802.11 wireless network scenario with 5 nodes to illustrate receiver-based congestion detection.

of network traffic was restricted, and reliable service for each event type could not be supported. Similarly to CODA, ESRT approach resides in the reduction of data sampling rate when a congestion is detected which leads to information loss.

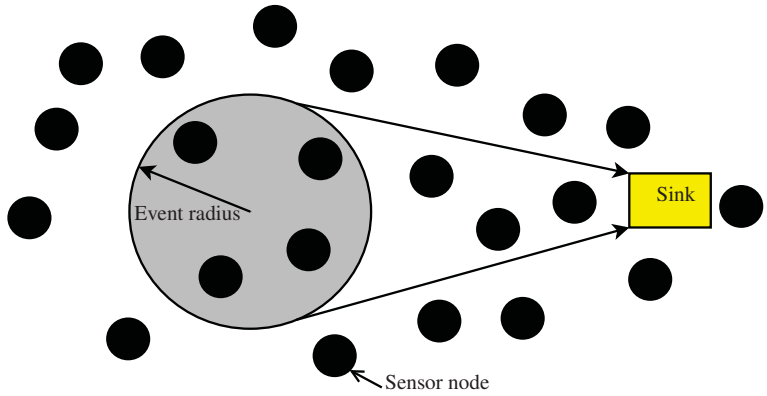


FIGURE 2.9: A WSN scenario with an event to inform to the sink. The sink is only interested to collect information of sensor nodes within the event radius.

The technique of data aggregation leads to use spatial or temporal correlation between sensed data to reduce its quantity and hence prevent congestion in the network [38]. Data aggregation techniques are especially useful in data collection WSNs where we notice a high temporal or spacial correlation on sensed data. Authors of [37] present an aggregation-based congestion control for sensor networks (CONCERT). The main idea of CONCERT is using an adaptive data aggregation technique to reduce the amount of information travelling throughout the network. Authors argue that data event collected by sensor nodes is characterized by a high degree of spatial correlation. In case a specific event occurs, all sensor nodes located in the area where the phenomenon occurs will collect the same information and transmit it towards the sink node. So, they use data spacial-correlation to eliminate redundant

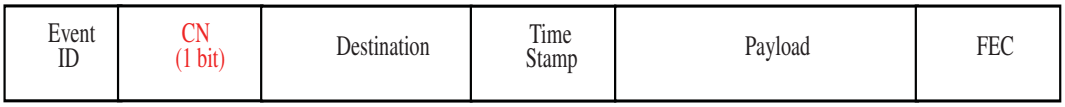


FIGURE 2.10: The format of data packet with CN field, which is marked to alert the sink about congestion [36].

data and that helps to mitigate congestion. Data forwarding nodes which implement spatial aggregation scheme try to find correlation between data received from different sensor nodes and then reduce the size of data to forward. Rather than using source nodes rate regulation to mitigate congestion, authors argue that data aggregation is easier to set up and leverages a characteristic of WSNs to solve the congestion issue.

Simulations results using NS-2 simulator show that the use of data aggregation helps to avoid congestion and improves packet delivery ratio. The main drawback of data aggregation scheme is the additional delay that packets may experience in the queue of node doing aggregation. This delay may affect the network throughput and the timely detection of the occurred event.

In WSN, congestion can be avoided or mitigated by controlling the data traffic rate. That brings to constraint some nodes in the congested area to reduce their data sampling or generation rate. Doing so may reduce the traffic load in the network and also lead to miss some important information. Another approach for congestion avoidance is the traffic load balancing that we present in follows.

2.3.2 Traffic load balancing in WSNs

In convergecast WSNs, all collected data is transmitted towards the sink node. In addition to their sensed data transmission, nodes play a role of router to forward data received from their neighbors towards the sink. Depending on the routing policy, some nodes in the network are more solicited to forward data towards the sink than others. These nodes tend to be overloaded while other potential next hop neighbors are under-loaded. Overloading some nodes leads to congestion, queue overflow and data loss that has negative consequences on the network performance. To avoid overloading some nodes while others are less solicited, different techniques for traffic load balancing are proposed. Load balancing routing tries to fairly distribute the traffic among all potential next hop neighbors. Routing protocols presented in [39], [40], [41], [42] and [43] propose a load balancing approach to prevent congestion in data transmission process.

In [39], authors propose a reliability-constrained routing protocol to balance the traffic load in WSNs. They design and develop a multi-objective optimisation solution to select less congested and optimum end-to-end path. The aim of the proposed approach is to avoid congested paths in the routing scheme. The routing is based on a composite objective function combining the maximum Packet Delivery Ratio (PDR) and the minimum hop count from sources to the sink. Congested paths are identified by capturing nodes traffic load on the path using the delivery ratio. Authors argue that transmission reliability is achieved by selecting least congested nodes along the

path from source to the destination. That also helps to balance the traffic load in the network. The drawback of the proposed routing approach is that the protocol mainly focus on congestion avoidance without proposing congestion alleviation solution in case it happens. The lack of congestion notifications in case congestion appears some where in the network is a limit.

In [40], Traffic Aware Routing Protocol for WSNs (TARP) is proposed. TARP is based on a lightweight genetic algorithm that balances the traffic in case congestion is observed around a node. In this protocol, sensor nodes are aware of the data traffic rate to monitor the network congestion. In case of congestion, by using a dominant gene sets, TARP selects suitable data forwarding nodes to avoid sending data to the overloaded nodes. Each sensor node has traffic information about all other nodes located within 2 hops. When traffic congestion appear in a specific node, a congestion alert message is sent to neighboring nodes which are sending data to the congested node. This message alerts them that their current next hop neighbor is congested. After receiving the congestion message, sender nodes have to share traffic information about their 2 hops neighbors with the congested node. Based on the received information, the congested node creates some chromosomes and allocates a data forwarding rate. Finally, after the chromosome with the highest fitness is selected, the result is sent to the sender nodes and they have to forward their data traffic to the best fitness neighbors.

Simulation results using NS-2 simulator show that TARP can find an alternative next hop to forward the traffic. It also avoids sending data to the congested neighbor in case of congestion due to detection of an urgent event. The main drawback of TARP is that, when congestion occurs, the congested node is in charge to find another under-loaded next hop for its transmitting neighbors. That is a challenge for this congested node to handle the congestion and at the same time computes the function to find an under-loaded next hop.

In [41], a Grid-based Multipath with Congestion Avoidance Routing (GMCAR) protocol is proposed. GMCAR is proactive, hierarchical and multipath routing protocol for WSNs which supports congestion avoidance mechanisms. It works in tree phases. In the first phase, the network grids are formed. An example of a grid sensor network used by GMCAR is shown in figure 2.11, where the sensor field is divided into 16 grids and master nodes are shown in filled squares with red color. The second phase is used to build routing tables and the last phase is the data transmission phase. After forming the grids, the sink initiates a flooding message to enable the master nodes to discover the available paths from each grid to the sink. GMCAR has congestion avoidance and congestion mitigation mechanisms. Its congestion avoidance mechanism is based on queue occupancy. When the buffer of a master node exceeds a certain threshold, it broadcasts invalid route message to the neighboring grid master nodes, and they have to change their routing paths. Other master nodes will stop forwarding data to the congested master node. However, in-grid nodes will not stop transmitting their data to the master as presented of figure 2.12 where grid "6" is invalidated. Even though the congestion avoidance mechanism exists, the congestion may occur. In this case congestion mitigation mechanism is enabled, which is based on splitting the incoming traffic. This is achieved by electing a secondary

master node, based on the residual energy of candidates, so that the load is distributed between the master node and the secondary master. The main drawback of GMCAR is that one node acting as master node until its energy is about to drain can result in quicker network partitioning.

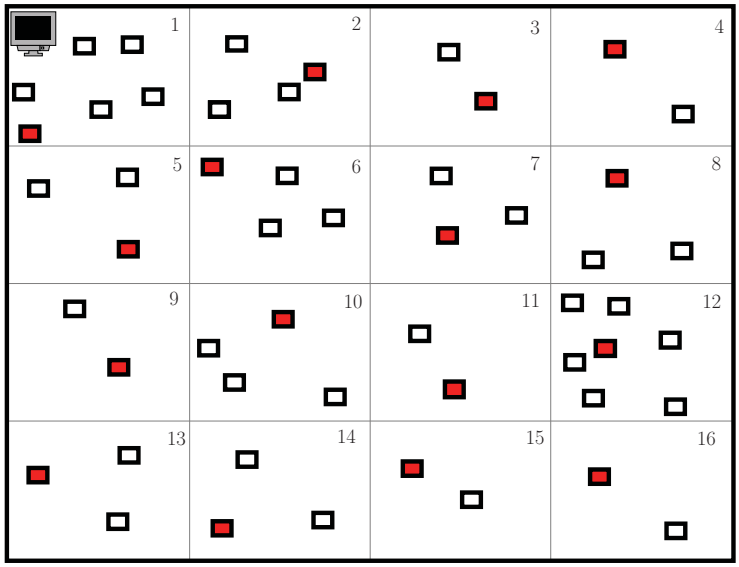


FIGURE 2.11: An example of a grid sensor network scenario used by GMCAR. The network is divided into 16 grids with one master node marked in red per grid.

In [42], a Load Balancing Routing (LBR) protocol is coupled with a multichannel MAC protocol for wireless mesh networks. LBR aims at reducing interference level and balancing traffic load among all potential links in the network. Using a link allocation scheme and a load balancing route selection algorithm, LBR protocol balances traffic load and improves network throughput. LBR is proposed for wireless mesh networks that does not take into account the characteristics of WSNs. Using it on WSNs will require adaptations that may reduce its performances.

Authors of [43] present a cognitive load balancing protocol for single hop multichannel WSNs. The proposed approach alternates communication channels based on the load distribution of the network sinks. This helps to redirect the extra load from overloaded channels to under-loaded ones. Simulation results using a single hop WSN show that the propose scheme can provide a high throughput using a multichannel communication. The drawback of the proposed scheme is that all presented results are obtained in single hop WSNs, so results may be less relevant in multi-hop WSNs.

In [44], authors present a Popularity Aware Congestion Avoidance (PACA) routing protocol that tries to identify upcoming congestion and to avoid its propagation. The proposed approach of congestion control computes a cost function which considers a set of metrics related to the popularity of the alternative nodes and routing

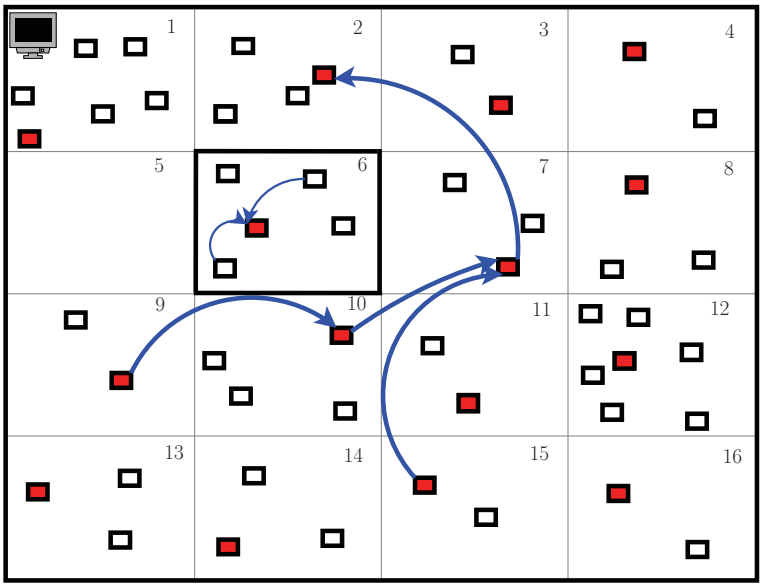


FIGURE 2.12: An example of GMCAR paths after invalidating the grid number '6'.

paths. PACA avoids congestion propagation through multi-hop and multi-path routing. Nodes and routes with less popularity are privileged to transmit data because they have low probability to be congested. Three metrics are used to evaluate the popularity level of each node. (i) The distance between each node and the sink. A high probability of being a packet forwarder is assigned to nodes in the neighborhood of the base station. (ii) The ratio between the upstream neighbors and the total neighbors of each node. Nodes that have less upstream neighbors than downstream neighbors are assigned a high probability to be congested. (iii) The cumulative time of the participation of each node in the network communication. Nodes positioned close to the area where incidents often occur will be confronted to high traffic load situation. The cost of popularity can be individually computed for each node and also for every corresponding potential path. The drawback of PACA is that, the popularity level of a given node is estimated through the position of the node in regard to the detected event which does not take into account the queue occupancy rate or the ratio of packet loss.

The IETF Routing Over Low power and Lossy networks (ROLL) Working Group designed a routing solution for low power and lossy networks (LLNs) including wireless sensor networks. The working group has specified the Routing Protocol for Low power and lossy networks (RPL) [45]. RPL is designed to meet the core requirements for data transmission in LLNs. It is being developed as a standard to be deployed in a number of environments: urban networks, smart grid networks, industrial networks, building and home networks. The routing scheme of RPL consists of constructing a Directed Acyclic Graph (DAG) rooted at the sink, which will minimize the cost of reaching the sink from any node in the network as per the Objective Function (OF). The OF of RPL can consider the ETX or the number of hops from each node towards the final destination to construct the DAG. Building and updating the DAG is done

using four kind of messages : DIO, DAO, DAO-ACK and DIS. DIO messages are generated more frequently than the other and used to propagate information about the DAG construction. DAO is used by nodes to request a parent and DAO-ACK is use by parent nodes to response the request. DIS messages helps nodes to solicit information in their neighborhood. Nevertheless, RPL is not designed for high data rate networks and suffers from congestion because it lacks a load balancing mechanism.

Traffic load balancing is an approach that helps to fairly distribute traffic load among potential next hop nodes. It helps to reduce the risk of congestion in the network. When the traffic load is high in the network thanks to multichannel communication, it is necessary to introduce load balancing routing for efficient data transmission. However, the most existing traffic load balancing routing protocols [44], [39], [40] and [41] are designed and evaluated in single channel networks. These load balancing routing protocols cannot work in multichannel networks without modifications and adaptation.

In multichannel WSNs, when designing a routing protocol, we have to deal with some issues unknown in single channel WSNs. Some of these issues are the channel switching problem, the deafness problem and the lack of broadcast support. So, the load balancing routing protocols proposed in single channel may not cope with multichannel WSNs. Some modifications and adaptations are needed when moved from single channel to multichannel WSNs. In our presented state of the art, only the protocol presented in [43] tries to introduce a traffic load balancing in multichannel routing protocol. However this protocol is evaluated in single hop WSN that is a very simple scenario and cannot give the performance of the protocol in multihop scenario. Which is the most common used network scenario in data collection applications.

2.3.3 Both traffic rate control and load balancing in WSNs

To fight against congestion, some papers [35], [36] control the traffic rate of the source nodes while others [39]–[43] prevent congestion by using traffic load balancing scheme. Each of these solutions has some limits, indeed, source nodes traffic rate control may lead to important information loss while load balancing becomes limited once congestion is observed. So, some authors proposed routing protocols that combine traffic load balancing to avoid congestion onset and source nodes traffic rate control once congestion appears some where in the network.

Authors of [46] proposed a Congestion Avoidance, Detection and Alleviation (CADA) protocol. The main idea followed by CADA is to avoid congestion or timely detects its onset and alleviates it before its extension. The congestion level is continuously monitored by evaluating each node buffer occupancy rate and channel utilization. The channel utilization evaluation is done by measuring channel loading whenever node attempts channel access. In case nodes find that packet delivery ratio decreases while the channel loading reaches the maximum utilization rate, congestion onset is declared and its alleviation method is triggered. CADA alleviates congestion using both resource and traffic control scheme. Resource control is done

by finding multiplexing paths that will be used to redirect data traffic to bypass congested areas. The traffic control consists in detecting event area sources nodes and reduces the number of sources node which are allowed to forward data towards the sink for the same event as presented on figure 2.13.

Performance is evaluated by means of simulations and presented results show that CADA can transmit data traffic by avoiding congestion or alleviating it. However, CADA is evaluated under event-based traffic generation scenario that is a specific scenario. In scenario where all nodes in the network have to continuously generate traffic and transmit it towards the sink node, the scheme proposed by CADA may not be suitable.

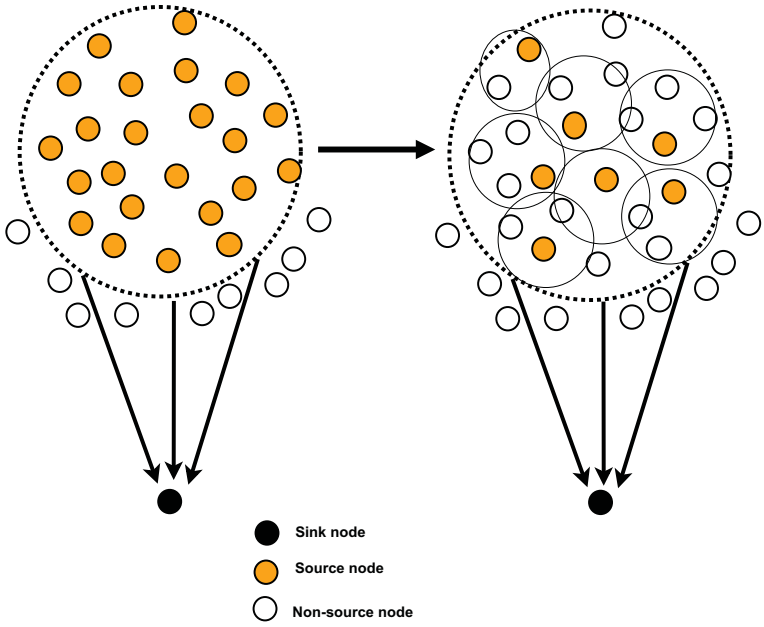


FIGURE 2.13: Representative node selection in the event area. In a given radius, only one node is allowed to transmit towards the sink the event collected data.
[36]

The authors of [47] propose an Interference-Minimized Multipath Routing (I2MR) protocol that integrates source rate adaptation and multipath routing for congestion control in WSNs. I2MR aims to increase the throughput by discovering zone-disjoint paths for load balancing and integrates minimal localization support. I2MR assumes that there is several final destinations (sinks) for data collected in the network. These destinations are connected to a command center via non interfering and high quality links. Sources nodes need to construct three disjoint routes towards final destinations. Sources nodes use two paths (primary and secondary paths) alternatively to transmit data towards the sink and preserve the third one as a recovery path in case of failure. Using primary and secondary paths for data transmission helps to balance the traffic load and avoids congestion in the network. Simulations results show that I2MR outperforms the standard AODV (Ad hoc On-Demand Distance Vector) [48] routing protocol in terms of packet delivery ratio and energy consumption. However, I2MR uses a localization algorithm and the overhead caused by this algorithm

is high. Moreover, to reduce the negative effects of intra-path interference, I2MR constructs shortest paths towards the sink nodes. Shortest paths is constructed using longest hops, and the uncertainly nature of wireless link may lead to frequent path loss when using longest hops.

In table 2.2 we present a comparative study of load balancing and congestion avoidance routing protocols already studied in section 2.3. We used the following five criteria to make our comparative study:

- **Proactive:** that helps to determine if the protocol pro-actively prevents congestion appearance by taking some measurements in advance.
- **Reactive:** we use it to know if the protocol has a mechanism to handle the congestion once it appears some where in the network.
- **Node level:** it helps to know if the congestion is detected based on parameters related to the node, like buffer occupancy rate or packet queueing delay.
- **Link level:** to determine if the congestion detection is done based on the radio link parameters like packet loss rate or channel access duration.
- **Multipath:** that helps to know if the protocol is designed for a multi-hop WSN.

Observing table 2.2, we notice that all presented protocols except [43] operate in multi-hop networks. Indeed, in multi-hop WSN with convergecast data collection, the risk of congestion is very high. Thus, congestion avoidance and mitigation mechanisms should be take into a consideration when designing routing protocols.

The comparative study also points out that even if the routing protocol integrates a congestion avoidance mechanism, in some cases, congestion may appear in the network. Preventing congestion by proactive actions is not enough to guarantee congestion avoidance. Thus, reactive actions against congestion must be taken into account when designing routing protocols. That will help to mitigate the congestion once it appears in the network. However, in the presented study in table 2.2, only CADA [40] combines both proactive and reactive actions to prevent congestion onset and to fight against it once it appears in the network. CADA uses both proactive and reactive approaches to address congestion issues, but it was evaluated in single channel WSN under event-based traffic generation scenario that is a specific scenario. In multichannel WSN with constant high traffic load profile, CADA needs modifications and adaptation to cope with the multichannel environment and the constant high traffic load profile.

In high traffic load scenario, the congestion issue becomes more serious and the routing protocol must prevent the congestion and mitigate it once it appears in the network. We notice that most existing congestion avoidance and traffic load balancing routing protocols are proposed and evaluated in single channel WSNs. Very few works study the traffic load balancing issue in multichannel WSNs. The existing multichannel routing protocols mainly focus on improving the carried load by reducing contention and collisions without treating the congestion issue. However, when the carried load becomes higher in the network, we need to care about congestion and apply traffic load balancing.

Protocols	Proactive	Reactive	Node level	Link level	Multipath
PACA [44]	Yes	No	Yes	No	Yes
[39]	Yes	No	Yes	Yes	Yes
[43]	Yes	No	No	Yes	No
CODA [35]	No	Yes	Yes	Yes	Yes
TARP [40]	No	Yes	Yes	Yes	Yes
ESRT [36]	No	Yes	Yes	No	Yes
I2MR [47]	Yes	No	No	Yes	Yes
LBR [42]	Yes	No	Yes	Yes	Yes
CADA [46]	Yes	Yes	Yes	Yes	Yes
CONCERT [37]	Yes	No	Yes	Yes	Yes
GMCAR [41]	No	Yes	Yes	No	Yes

TABLE 2.2: Summary of load balancing and congestion avoidance protocols in WSNs.

The routing protocol must ensure that the traffic load is fairly distributed between all potential next hop neighbors. That helps to balance the traffic load and avoid overloading some nodes while others are under-loaded. In high traffic load scenarios without traffic load balancing policy, the risk of congestion leading to queue overflow becomes high. So, in multichannel routing protocol, a traffic load balancing is necessary to avoid queue overflow and packet loss and then maintain good network performance. In the following chapter 3, we propose a load balancing routing protocol that prevents congestion and packet loss due to queue overflow in multichannel WSN.

Chapter 3

Contributions

In the Industrial, Scientific and Medical (ISM) unlicensed 2.4 GHz band, the IEEE 802.15.4 standard offers 16 orthogonal channels that can be used for parallel data transmissions. The use of multiple channels helps to mitigate some issues encountered in single channel communications like interference and collisions caused by high contention. With more bandwidth, less interference and less collisions, the network performances will be enhanced and the traffic load in the network will increase. Nodes will have more data to transmit towards the sink and intermediate nodes will have to cope with the traffic rate and find a way to balance the traffic load among them. This is typically the role of a load balancing routing protocol.

Routing in multichannel WSNs has been studied in the last decade [49], [50]. However, having a routing protocol that permits to reach high data rate in multichannel WSNs remains an open research topic. The constrained resources of sensor nodes coupled with a shared wireless medium makes the issue more complex. In single channel WSNs, the interference level is high under high data rate scenarios which leads to higher risks of collisions and packet loss. Multichannel communication has been introduced in order to reduce the interference level and improve the data delivery ratio. In multichannel networks, nodes do not listen on the same channel all the time like in single channel networks. This makes topology and routing information diffusion more challenging. To overcome this issue, some protocols introduce synchronizations phase [20], where all nodes in the network have to tune their radio on the same channel for exchanging control traffic (topology and routing information). However, synchronization is time and energy consuming in addition to the fact that it is hard to achieve in large scale networks. Piggy backing network service (topology and routing) information in data packets or acknowledgement (ACK) frames may be an alternative solution to synchronization.

In this chapter, we present our contributions, S-CoLBA for Single channel Collaborative Load Balancing Algorithm, M-CoLBA for Multichannel CoLBA and its enhanced version ABORt for Acknowledgement-Based Opportunistic Routing Protocol for multichannel WSNs.

S-CoLBA is a single channel load balancing routing protocol that avoids congestion and queue overflow. M-CoLBA is a multichannel routing protocol with synchronization periods. The routing metric of both CoLBA versions is based on the average queueing delays of data packets. In addition, they use multiple next hop nodes towards the destination whenever possible in order to achieve load balancing. We also present M-CoLBA enhanced version, ABORt, which is a routing protocol designed for high data rate multichannel WSNs. ABORt does not rely on periodic synchronization for exchanging topology and routing information. It uses MAC layer ACK

frames in an opportunistic way to exchange control information. Hence, it avoids costly synchronization periods during the operational phase of the network for exchanging routing metric and enhances throughput and access delay by doing so.

The remainder of the chapter is organized as follows. In section 3.1, we present some issues encountered when introducing high data rate in WSNs and ways to mitigate them. In section 3.2, we describe our contributions M-CoLBA and S-CoLBA. Section 3.4 presents the enhanced version, ABORt. Finally, section 3.5 summaries the chapter and introduces the next one.

3.1 Issues in multichannel routing protocols for data collection and ways to mitigate them

Multichannel communication helps to reach high traffic load in WSNs [17]. The need of high data rate applications in WSNs is increasing [51] [52]. Sensor nodes are resources constrained devices including buffer size used to store data. When traffic load becomes too high in sensor network, the risk of congestion and queue overflow increases too. In this section, we study some issues that should be considered when introducing high traffic load in WSNs and way to mitigate these issues.

3.1.1 Congestion in high traffic load WSNs and its impact on data packets queueing delays

In wireless sensor networks, congestion occurs when sensor nodes are carrying more data packets than they can handle. Congestion leads to collisions and packet loss which has negative consequences on the network performance. In multi-hop wireless sensor networks with First In, First Out (FIFO) queue management policy, generated or forwarded data packets are enqueued in packet queue and transmitted using the FIFO policy. When network is congested, it is not straightforward for nodes to access the medium and transmit they data towards the sink because of high contention and collisions. Thus, enqueued data packets for transmission may stay for a long time in queues awaiting their turn to be transmitted. That extends data packets queueing delays.

Wireless radio medium is a shared resource and when the medium access protocol is contention-based like CSMA/CA, nodes are not ensured that they can access to the medium and transmit their packets whenever they try. Collisions may occur and they have to postpone the transmission. Whenever collision occurs, the transmission failed and the packet has to wait in the queue for the next transmission attempt. That extends data packets queueing delay. So, in contention-based medium access protocol, packets queueing delays are affected by the network congestion level. When node neighborhood is congested, collisions become frequent leading to retransmissions, and longer queueing delays. However, when the network is not congested, collisions are rare and nodes can quickly access the medium to transmit their data packets. That helps to reduce data packets queueing delays.

When node neighborhood is congested, its enqueued packets queueing delays increase. Once the congestion drop in its neighborhood, it can transmit its data packets

and queueing delays may be reduced. Observing the evolution of data packets queueing delays may help to detect the congestion level around each node neighorhood. That helps to avoid routing packets through this node, which is useful to reduce data packets queueing and end-to-end delays. With reduced end-to-end packet delays, we can improve the network throughput.

In our contribution, we will use data packets queueing delays as routing metric to detect congested nodes and avoid using them as next hop.

3.1.2 Impact of number of next hop neighbors on traffic load balancing

In multi-hop wireless sensor networks, nodes which are out of range of the sink need the help of next hop neighbors to forward their collected data towards the sink. The choice of the next hop neigbor is governed by a routing metric that may be static (hop count [53]) or dynamic (expected transmission count [54], packet queueing delay). In dynamic routing, to transmit data packets towards the sink, node may select the best next hop neighbor according to the routing metric and forwards its data to it. It is also possible to choose several next hop neighbors according to the routing metric and alternatively forward data packets to each of them. In what follows we study the impact of using one or multiple next hop neighbors on traffic load balancing in WSNs.

Using one next hop neighbor for data transmission

To choose a next hop neighbor, each node is guided by the routing metric. The routing metric indicates which neighbor is efficient according to the routing policy. When it aims to minimize data queueing delays, the neighbor with the smallest queueing delay will be selected as a next hop. Data packets will be forwarded to this next hop neighbor until the routing metric indicates that the choice of an other neighbor becomes better than this one. Thus, node has to drop the current next hop and switch to the new best one and continues its data transmission. The fact of choosing the best neighbor according to the routing metric as a next hop does not favour traffic load balancing. Indeed, in dense network scenarios, several neighboring nodes may select the same neighbor as next hop because this neighbor is the best according to the routing metric in their communication range. All these nodes will forward their data to the same next hop node. Doing so reduces the traffic load balancing and leads to congestion in the network.

Figure 3.1 depicts a network scenario where 6 nodes $(4, 5, 6, 7, 8$ and $9)$ have three common next hop neighbors, 1, 2 and 3, towards the sink (s). Choosing the next hop neighbor with the best routing metric as the next hop will make all nodes in the same communication range to choose the same next hop node, 1 or 2 or 3 as depicted in figures 3.2 and 3.3. In figure 3.2, the six nodes (4, 5, 6, 7, 8 and 9) select node 3 as next hop because it is the best according to the routing metric. However, in figure 3.3, the best next hop becomes node 1 and they select it as their next hop. Doing so, the neighbor with the best routing metric will be quickly overloaded by its downstream neighbors while the other nodes (which do not have the best routing metric value) are less used.

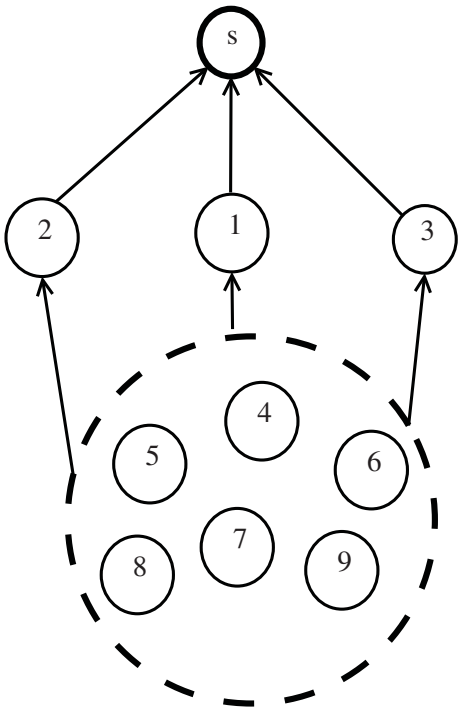


FIGURE 3.1: 10-node network scenario where 6 source nodes forward data packets to the sink (s) via three common next hop neighbors 1, 2 and 3.

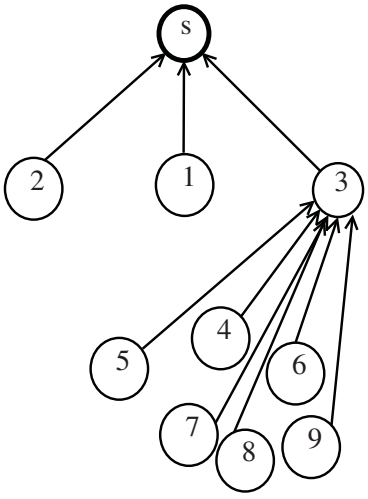


FIGURE 3.2: (a) node 3 is the next hop for all downstream nodes because it is the best according to the routing metric.

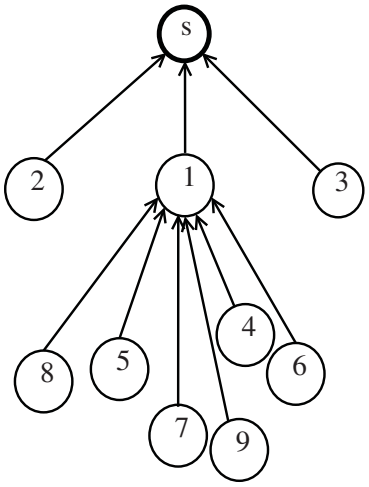


FIGURE 3.3: (b) node 1 becomes the next hop for all downstream nodes because it has now the smallest value of the routing metric

Using a simulated network scenario depicted in figure 3.1, where all nodes except the sink generate 2 data packets per second (with 50-byte packet size), we counted the number of packets transmitted to nodes 1, 2 and 3 by their downstream neighbors

(nodes 4, 5, 6, 7, 8 and 9) for each 5 seconds duration. The used routing policy is to minimize data packets queueing delays. Node with the smallest queueing delays is chosen as a next hop to which data is forwarded. The obtained results are presented in table 3.1. The evaluation is started 30 seconds after simulations start. Results show that, the data traffic is not fairly distributed between the three intermediate nodes 1, 2 and 3. For example, from 30 seconds to 35 seconds, node 1 has not been solicited by its downstream neighbors while node 2 receives 36 data packets from the downstream neighbors. In the time interval 35 seconds to 40 seconds, node 2 received only 6 data packets while node 1 received 34 and 17 for node 3. The unfair distribution of data traffic due to the fact that all neighboring nodes choose the same next hop neighbor leads to overloading some nodes while others are under-loaded. This lack of traffic load balancing provokes congestion and data loss.

Nodes	30 s to 35 s	35 s to 40 s	40 s to 45 s
Node 1	0	34	14
Node 2	36	6	38
Node 3	18	17	2

TABLE 3.1: An example of distribution of data packets received by intermediate nodes 1, 2 and 3. Packets are transmitted by leaf nodes 4, 5, 6, 7, 8 and 9 towards the sink node.

Using multiple next hop neighbors for data transmission

In network scenarios where each node has several next hop neighbors towards the sink, it is possible to alternatively forward data packets to each of them. Instead of selecting the best next hop neighbor (best score neighbor) according to the routing policy, nodes can build a list of next hop candidate neighbors and alternatively use them.

This list may contain the best score neighbor plus other next hop neighbors which exceed the best score neighbor with an acceptable gap. For example, in WSNs where the routing policy aims to select the next hop neighbor which minimizes data packets queueing delays, the list of next hop candidate neighbors will contain the best score neighbor plus all other nodes that exceed this best score within defined threshold (2, 3 or 4 milliseconds). Once this list is built, nodes will alternatively select one next hop in the list and forward its data packets to the selected neighbor. Doing so helps to distribute data packets between several next hop neighbors instead of transmitting only to the best score next hop. The concept of using a list of next hop candidate neighbors helps to balance the traffic load among forwarding nodes and avoids congestion and data loss. Using the 10-node network scenario depicted in figure 3.1, we evaluate the benefit of using a list of next hop candidate neighbors for traffic load balancing. Nodes 1 to 9 generate 2 data packets per second per node and

transmit them to the sink (s). The routing policy aims to reduce data packets end-to-end delays, and the best score neighbor is the neighbor that reduces data queueing delays. Each node builds a list of candidate next hop neighbors that contains the best score neighbor and neighbors that have 3 milliseconds more than it. If a node has a data packet to transmit, it randomly selects one neighbor in the list and forwards its packet to it. At time $t_1 = 35$ s and $t_2 = 40$ s after the start of the simulation, we check the next hop chosen by each node to forward its data packets. Figures 3.4 and 3.5 depict selected next hops. We notice that, in contrast of what we saw in figures 3.2 and 3.3, all leaf nodes do not select the same next hop.

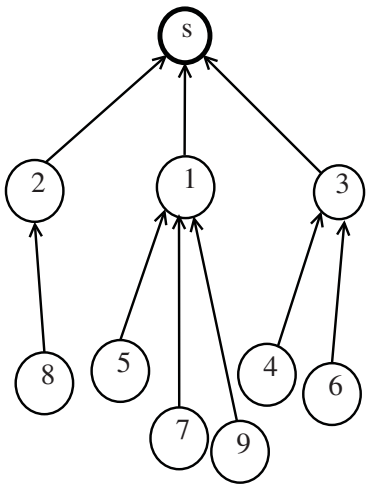


FIGURE 3.4: (a) selected next hop neighbors in time $t_1 = 35$ s by leaf nodes.

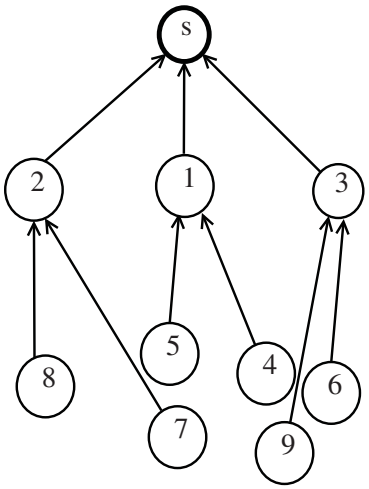


FIGURE 3.5: (b) selected next hop neighbors in time $t_2 = 40$ s by leaf nodes.

We also counted the number of data packets transmitted to each intermediate node by leaf nodes. Table 3.2 contains the number of data packets received by nodes 1, 2 and 3 forwarded by nodes 4, 5, 6, 7, 8 and 9 during 15 seconds data transmission divided in 5-second intervals. Results show that the data traffic transmitted by the six leaf nodes is fairly distributed between the 3 intermediate nodes. So, using alternatively different next hop neighbors to forward data towards the sink helps to balance the traffic load in the network.

Results of this study give an idea on the impact of using one or multiple next hop neighbors to forward data packets to the sink. The best score next hop neighbor according to the routing metric limits the traffic load balancing and leads to congestion in the network. However, having a list of candidate next hop neighbors and alternatively selecting one neighbor in this list to forward data packets helps to balance the traffic. The traffic load balancing is important to help the network to be efficient mainly in high data traffic scenario. Thus, a best trade-off between the routing metric and the traffic load balancing is necessary to improve the network efficiency. In our contribution, we will exploit the idea of using alternatively different next hop neighbors to forward data packets in order to balance the traffic load in the network.

Nodes	30 s to 35 s	35 s to 40 s	40 s to 45 s
Node 1	19	22	22
Node 2	21	20	18
Node 3	18	17	19

TABLE 3.2: An illustrative example of data packets distribution on intermediate nodes (1, 2 and 3) by time intervals. Packets are transmitted by leaf nodes 4, 5, 6, 7, 8 and 9 towards the sink node "s".

3.1.3 Problem of broadcast support in multichannel communications

In WSNs, topology information and routing metric are generally shared using broadcast messages. Hence, broadcast support is used by nodes to share information in their neighborhood.

In single channel communication, nodes in the network tune their radio frequency on the same communication channel. Thus, each node can broadcast and receive topology and routing informations in its neighborhood using the common communication channel. However in multichannel communication, neighboring nodes do not listen the same channel at the same time. So, it is not straightforward to obtain a broadcast support for information diffusion in multichannel networks.

In most existing multichannel communication protocols [17], [55], the problem of broadcast support is solved by introducing synchronization periods in the network. During a fixed time period, nodes in the network have to use the same communication channel in order to shared topology and routing informations. However, the synchronization is hard to obtain in large scale WSNs and it extends data packet end-to-end delays. Data packets generated before synchronization period that did not reach the sink before the starting of the synchronization have to wait in nodes queues during the synchronization period. That increases the end-to-end delays and reduces the network throughput which negatively affects the performance of the network.

In most WSNs communication scenarios, the reception of data frame is confirmed by an ACK frame transmitted by the receiving node. Thus, ACK messages are frequently sent as data packets. ACK messages can be used in an opportunistic way to disseminate topology and routing informations. That is known in the literature as piggy backing. Doing so will avoid introducing synchronization periods to share topology and routing informations and help to optimize packet end-to-end delays in multichannel networks.

3.1.4 Summary

Wireless sensor nodes are resource constrained devices with limited buffers size to store data packets. Introducing multichannel communication in WSNs helps to increase the carried traffic load thanks to the reduction of contention and interference and parallel data transmissions. With high traffic load, the number of data packets travelling from leaf nodes towards the sink becomes higher. And depending on the routing policy, the traffic may be unfairly distributed between forwarding nodes. Nodes that are part of the routing may be overloaded while others are under-loaded. With overloaded nodes, there is a high risk of congestion and queue overflow leading to data loss that reduces the throughput and the network efficiency. Thus, we need to couple the routing protocols with traffic load balancing in high data network scenarios. That will help to fairly distribute data traffic among all potential forwarding nodes and avoid overloading some nodes while others still have memory space to store data.

Dealing with multichannel protocol comes to another issue which is how to get a diffusion support for topology and routing informations exchange. Using synchronization periods gives an answer but it may negatively affect data packet end-to-end delays. In some applications, using synchronization periods may not be suitable because of the longer end-to-end delays that data packets can experience. So, an alternative way to synchronization is necessary. The solution of piggybacking topology and routing informations in acknowledgement frames can be an alternative solution to the synchronization. In following sections, we present in details our contributions, traffic load balancing routing protocols with queue overflow avoidance in multichannel WSNs.

3.2 M-CoLBA: Multichannel Collaborative Load Balancing Algorithm with queue overflow avoidance

The use of multichannel MAC protocols significantly improves the throughput and increases the overall data traffic load in the network [56]. The negative consequences of high traffic load is the creation of congestion and possibly packet queue overflow especially in nodes close to the sink in data collection application scenarios. Increasing traffic load in the network by introducing multichannel communication should be coupled with a traffic load balancing approach to avoid congestion and packet lost. We propose a Multichannel Collaborative Load Balancing Algorithm with queue overflow avoidance (M-CoLBA) that uses queueing delay-based dynamic routing to balance the traffic load. As presented on figure 3.6, in M-CoLBA, nodes activity is organized in cycles. Each cycle is divided in two phases: beacon phase for control messages exchange (topology, channel and routing information), that lasts θt time and data phase dedicated to data transmission towards the sink with ωt time duration. The network starts by a beacon phase dedicated to neighborhood discovery, channel assignment and routes construction towards the sink. This first beacon phase is followed by data transmission and so on and so forth. The main contribution of M-CoLBA is improving the network throughput by fairly distributing traffic load and

avoiding to loose data packets due to queue overflow using a congestion aware dynamic routing metric. In what follows we present in details our multichannel routing protocol M-CoLBA.

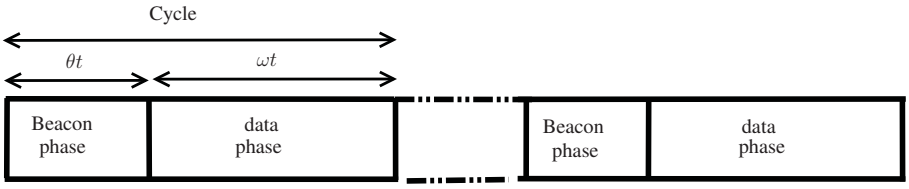


FIGURE 3.6: M-CoLBA beacon and data phases. The network starts by a beacon phase where the topology is built and channels are assigned to nodes.

3.2.1 Neighborhood discovery and predecessor selection in M-CoLBA

WSN is a network consisting of spatially distributed autonomous wireless sensor nodes to monitor physical or environmental conditions in an area of interest. To set the network, neighborhood discovery is necessary. In case multiple channels are used for communication, channel selection may be done following a principle where each node selects a predecessor. The predecessor of a node is its neighbor that must select its channel before it. We present the predecessor selection and neighborhood discovery in what follows.

Neighborhood discovery

Neighbors discovery helps to build the network and prevents nodes selecting channels already used by their neighbors. At the network set up, all nodes share the same communication channel. That allows them to be able to receive and broadcast information used to discover neighbors.

Nodes in the network, except the sink, have to discover their 1-hop, 2-hop and 3-hop neighbors [57]. Neighborhood discovery is done using beacon frames. Each node builds its 1-hop neighbors list on reception of beacon frames from its neigbors. Each node includes the list of its 1-hop neighbors in beacon frames. Thus, receiving nodes can build the list of their 2-hop neighbors using the neighbors lists of their 1-hop neighbors. The list of 2-hop neighbors is also included in beacon frames which will be used by receiving nodes to build their list of 3-hop neighbors. At the end of the neighborhood discovery process, each node in the network except the sink has a list of its 1-hop, 2-hop and 3-hop neighbors. In the network scenario presented in figure 3.7, the 1-hop, 2-hop and 3-hop neighbors of node 8 is coloured in blue. This list contains nodes 4, 5, 11, 12, 14, 15, 16, 17, 21, 25 and 27.

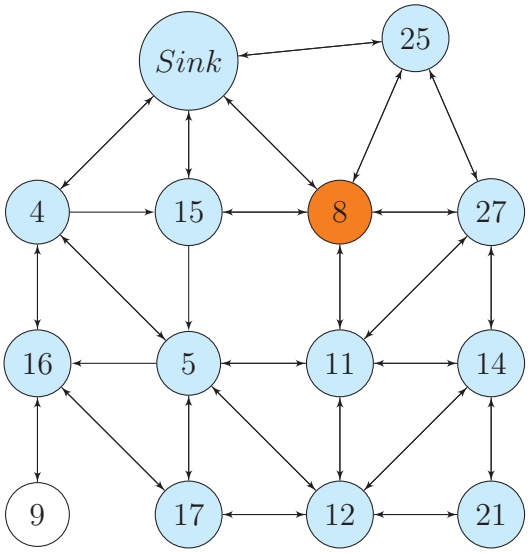


FIGURE 3.7: In this network scenario, 1-hop, 2-hop and 3-hop neighbors of node 8 are coloured in blue.

Predecessor selection

After the neighborhood discovery, each node knows its 1-hop, 2-hop and 3-hop neighbors and has to identify its predecessor. Identifying a predecessor helps to define an order for the channel selection by nodes.

A predecessor of a node N is its 1-hop, 2-hop or 3-hop neighbor M with an IDentifier (*ID*), m that immediately precedes its own *ID*, n. For example, in figure 3.8, the 1-hop, 2-hop and 3-hop neighbors of node 8 are nodes 4, 5, 11, 12, 14, 15, 16, 17, 21, 25 and 27, thus, the predecessor of node 8 is node 5. So, node 8 has to keep this information that will be used during the channel selection process presented in the section 3.8. Node with the smallest *ID* in its 1-hop, 2-hop and 3-hop neighborhhood is a priority node and does not have a predecessor.

3.2.2 Channel allocation scheme

When multiple channels are used in the same network for communication, each node has to know the reception channel of its neighbors to be able to communicate with them. That requires sharing information on channel allocation among neighboring nodes. As shown in [58], channel reuse within 3-hop neighborhood leads to collisions when MAC layer acknowledgement messages are enabled. So, to prevent collisions, nodes avoid as much as possible to select the already used channels within their 3-hop neighborhood. Nodes select channels in an order based on their *ID*s. A node *ID* is a 2-byte integer that is allocated to each node in an unique manner at the network start-up phase. The node with the smallest *ID* within its 3-hop neighborhood (the priority node) selects its channel first. The node that has a predecessor does not select a channel as long as its predecessor has not announced its reception channel. For example in figure 3.9, node 8 cannot select its reception channel until node 5 chooses its and announces it in a beacon frame.

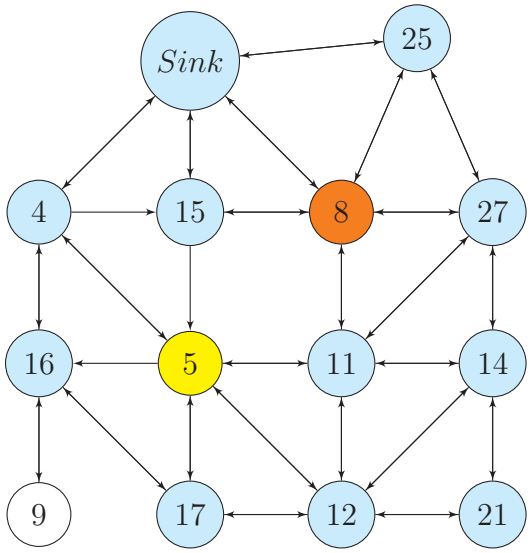


FIGURE 3.8: In this 14-node network scenario, 1-hop, 2-hop and 3-hop neighbors of node 8 are in blue and yellow. Node 8 predecessor is node 5 in yellow.

The main idea behind the channel allocation is to avoid while possible the reuse of the same channel within 3-hop neighborhood. To do so, each node searches for a free channel within its 1-hop, 2-hop and 3-hop neighborhood. If no free channel is available, it searches for a free channel in its 1-hop and 2-hop neighborhood. In case all channels are already taken, it tries to find a free channel in its 1-hop neighborhood. Finally, if no free channel is found, the less used channel in the 1-hop neighborhood will be selected.

At the end of the channel selection process, each node has its reception channel and this choice is also known by its neighbors. In figure 3.9, each node reception channel is labelled. The sink node has three reception channels ($f11$, $f12$ and $f13$) while others have one.

3.2.3 Routing metric computation and diffusion

In WSNs, the selection of the next hop neighbor is done by the routing protocol that relies on a routing metric. M-CoLBA routing metric is based on the average queueing delay of data packets locally computed by each node.

The locally average queueing delay observed by each node is called *node delay* and the end-to-end queueing delay from each node to the sink is the *path delay*.

Node delay computation

The node delay is an average data packet queueing delay locally computed by each node. Except the sink, node delay is continuously computed by each node in the network. Each generated or forwarded data packet is enqueued in packet queue and transmitted using FIFO policy. In figure 3.10, we present an example of node queue where data packets are enqueued and dequeued using FIFO. In M-CoLBA, whenever data packet is enqueued for transmission, the time is recorded using the local clock

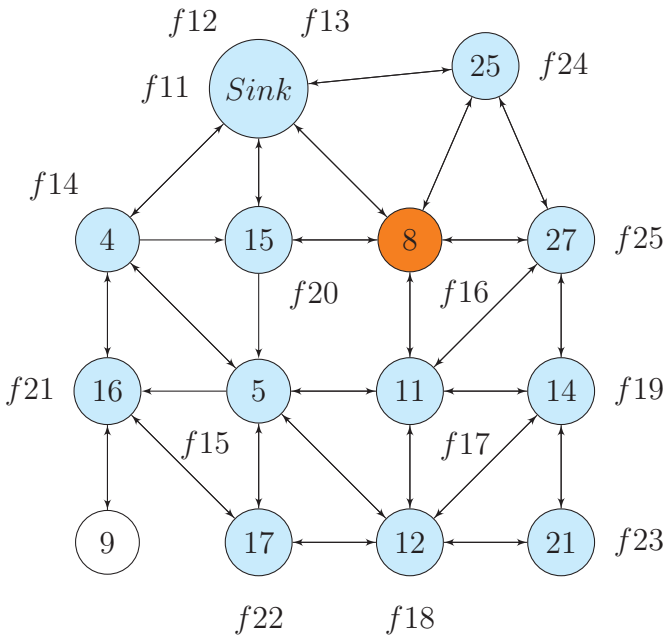


FIGURE 3.9: In this 14-node network scenario, 1-hop, 2-hop and 3-hop neighbors of node 8 are in blue. The reception channel of each node is labelled. Nodes have one reception channel except the sink which has 3.

of the node. When the same packet is dequeued (transmitted or dropped) the time is also recorded. Data packet is considered as transmitted if an ACK frame is received for it. It is dropped when the number of transmission retries exceeds the allowed number by the CSMA/CA algorithm.

The difference between the dequeued and enqueued instants is the packet queueing delay. The average value of the queuing delay of the last dequeued ten [1] packets is called node delay d. To calculate d, each node uses 10-boxe array to keep the ten last dequeued packets node delay as depicted on figure 3.11. Whenever a new queueing delay is computed by node the oldest one in the array is deleted and the new one is inserted. Algorithm 1 presents step by step node delay computation. If node has not dequeued ten packets yet, the node delay is the average value for the already dequeued packets.

The queueing delay of last dequeued packets reflects the up-to-date state of the queue compared to packets dequeued long time ago. When we have a short queueing delay, this means that packets do not spend much time in the queue before their transmission to the next hop neighbor. Thus, when calculating d, we use positives weighing factors α and β (with $\alpha < \beta$) to give more weight to delays of last dequeued data packets compare to others. Figure 3.11 shows an example of how the weight factors are used to compute node delay. Using weigh factors that give more importance to last dequeued packets delays helps to compute node delays that reflect

[1] Ten packets is a value obtained through heuristics showing that smaller values create oscillations, and bigger values prevent nodes from accurately updating its delays.

Algorithm 1: The *node delay* d computation by each node in the network.

```
1  Input: generated or forwarded data packet;
2  Output: node delay d;
3  enqueue the packet;
4  record the enqueue time;
5  the packet waits in the queue until its turn for transmission;
6  Transmit a copy of the packet using CSMA/CA;
7  if ACK received then
8  |   dequeue the packet;
9  |   record the dequeue time;
10 |   compute the queueing delay of the packet;
11 |   delete the oldest queueing delay in the 10-boxe array;
12 |   put the new queueing delay in the 10-boxe array and compute the node
   |   delay d;
13 else
14 |   try retransmission until reach the max retries;
15 end
16 if the max retries is reached then
17 |   execute instructions from 8 to 12;
18 end
```

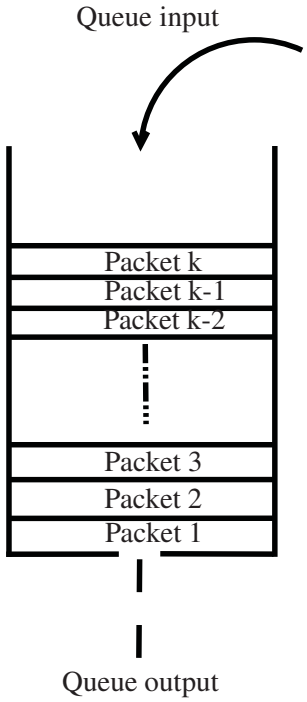


FIGURE 3.10: k data packets in node queue. Packets are enqueued and dequeued using FIFO policy.

the up-to-date state of the queue. The formula of node delays calculation is presented in equation (1).

$$\text{Node delay (d)} = \frac{\sum_{i=1}^{5} \alpha * queueingdelay(i) + \sum_{i=6}^{10} \beta * queueingdelay(i)}{5\alpha + 5\beta} \quad (1)$$

Where *queueing delay (i)* is the difference between dequeuing instant and queueing instant of packet (i). α and β are weight factors.

Once node has already dequeued ten packets, it uses a 10-box FIFO array where the oldest queueing delay is deleted and the new one is inserted as presented on figure 3.11. Using the 10-box FIFO array helps to be sure that each node always records the queueing delays of the most recent dequeued ten packets.

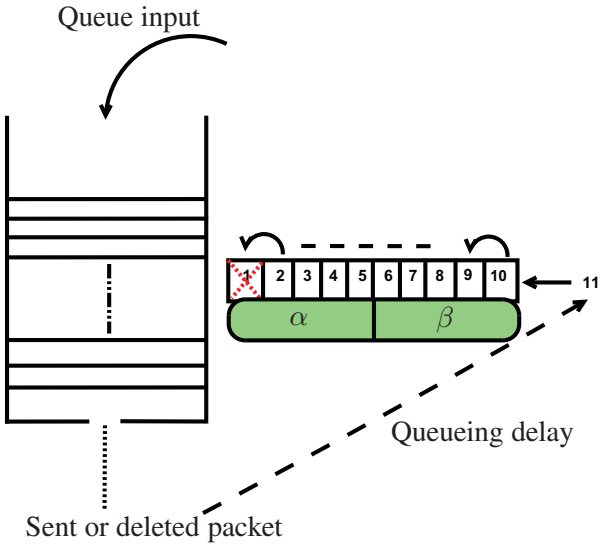


FIGURE 3.11: Node delay d computation, with weight factors α for the 5 oldest queueing delays and β for delays of the newest dequeued 5 packets.

Data transmission and path delay computation

Nodes compute average queueing delays when they transmit data packets to their next hop neighbors or to the sink. After neighbors discovery, sink 1-hop neighbors start transmitting data to the sink and compute their node delay d.

The path delay D is an average delay expressed in milliseconds that each packet will spend to travel from their source node to the sink. The path delay on a path is obtained by summing node delays d on this path. The algorithm 2 presents steps followed by nodes to compute path delays. In figure 3.12, we present a network scenario where each node is labelled with the couple d/D. In this scenario, the path delay of node 13 is 21 milliseconds. This value is obtained by summing node delays on the path with smaller queueing delays from node 13 to the sink. According to node delays, the efficient path from node 13 to the sink is through nodes 9, 4 and 6. As we can see in figure 3.12, node delay d and path delay D are the same for sink 1-hop neighbors. They do not need next hop neighbor to forward their data to the sink, because they can directly reach it. Indeed, sink 1-hop neighbors randomly choose one of the three interfaces of the sink and switch to the reception channel of that interface for transmission. When several nodes have to transmit data to the same

destination, the CSMA/CA algorithm is used for medium access. Once all nodes in the network got a path towards the sink using M-CoLBA routing, each node path delay is updated whenever a smaller path delay is received or a new node delay is computed.

Algorithm 2: The *path delay* D computation by each node in the network.

1 **Input:** neighbors path delays (D);
2 **Output:** the path delay of the node;
3 select the smallest path delay among neighbors path delays;
4 sum its node delay with the smallest path delay;
5 the obtained delay is the path delay D of this node;
6 **if** *a smallest path delay is received from neighbor or a new node delay is computed* **then**
7 | update the path delay by executing instructions from 3 to 5;
8 **end**

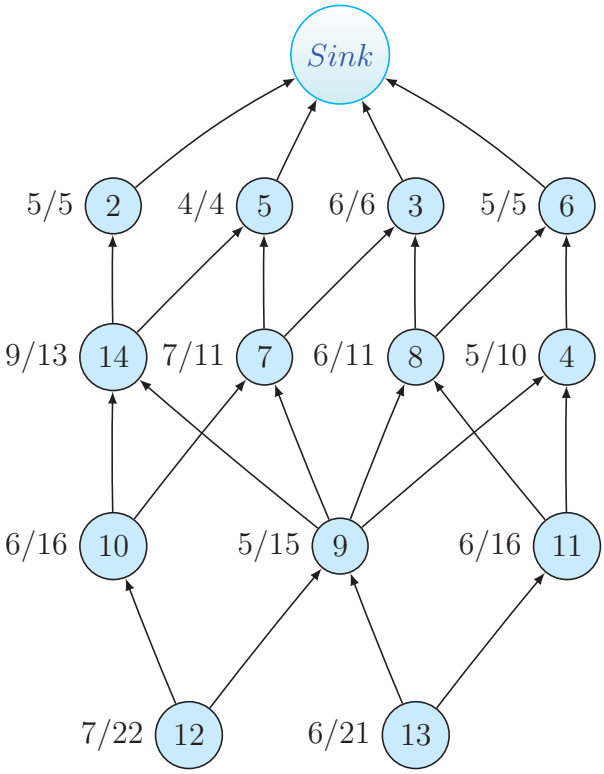


FIGURE 3.12: 14-node network scenario, where node delay d and path delay D are labelled at left of each node with d/D values. d and D are expressed in milliseconds.

3.2.4 M-CoLBA routing metric dissemination

In single channel WSNs, nodes always switch on the same channel and use broadcast messages to exchanged routing information. In multichannel communication, broadcast support is not easy to obtain because nodes are not switched on the same

channel all the time. Thus, we need to find a way that will make all nodes listen on the same channel for information exchange. M-CoLBA uses synchronization periods where nodes have to switch on the same channel and exchange routing information. The synchronization periods last few seconds called θt and during this time, nodes broadcast their routing metric values in their neighborhood. When a node receives the metric value of its next hop neighbors, it updates its path delay and then broadcasts its updated path delay to its neighbors. Each synchronization period is followed by a data transmission period as shown in figure 3.6 where nodes forward their data packets to their next hop neighbors.

3.2.5 Load balancing and queue overflow avoidance in M-CoLBA

In WSNs, congestion has negative consequences because it leads to queue overflow and packet loss that reduces the delivery ratio. To prevent congestion, M-CoLBA proactively applies traffic load balancing and reacts to mitigate the congestion once it appears in the network. In what follows, we present the proactive and reactive approaches used by M-CoLBA to fight against congestion.

Traffic load balancing

Nodes that are out of range of the sink have to forward their data to a next hop neighbor which will also forward them towards the sink. Depending on nodes positions in the network topology and the used routing metric, some nodes have several potential next hop neighbors to forward their data. Using M-CoLBA, nodes build a list of best score next hop neighbors called *top-list*. Nodes in the top-list will be used to forward packets towards the sink. For each packet, a next hop is randomly chosen in the top-list. This way, nodes will achieve load balancing among their next hop neighbors.

The top-list contains neighbor(s) with the smallest routing metric value and neighbors that exceed this smallest value within a certain threshold defined in terms of milliseconds and designed as $\triangle T$. The choice of $\triangle T$ must be done in order to choose most convenient routes and avoid routing loops from occurring. Using a heuristic approach based on simulation results, we fixed $\triangle T$ to $2ms$. Results showed that this value avoids routing loops when transmitting data to the neighbors in the top-list. We detail the top-list construction and updating in algorithm 3.

Algorithm 3: *Top-list* construction and update by each node in the network.

Input : next hop neighbors path delays
Output: top-list members

find the smallest path delay among next hop neighbors path delays;
foreach *path delay of node* i **do**
  if *the path delay of node* $i \leq$ *smallest path delay* $+ \triangle T$ **then**
    add node i in the top-list;
  end
end
if *a top-list member node* i *new path delay is received* **then**
  evaluate the top-list entering condition;
  if *node* i *path delay* $\leq$ *smallest path delay* $+ \triangle T$ **then**
    node i stays in the top-list
  else
    remove node i from the top-list;
  end
end
if *node* j*, no top-list member path delay is received* **then**
  evaluate the top-list entering condition;
  if *node* j *path delay* $\leq$ *smallest path delay* $+ \triangle T$ **then**
    add node j in the top-list
  end
end

The size of the top-list of each node is dynamic and may be different from one node to an other. Once the top-list is constructed, for each data packet to transmit, nodes will randomly choose one neighbor in their top-list and switch to this neighbor reception channel for transmission. When nodes transmit data to their next hop neighbors, they switch back to their reception channel before trying an other transmission. Doing so helps to reduce the deafness problem.

On figure 3.13, we present the top-list concept with a 12-node network scenario. The couple d/D for each node is labelled at left. Node 9 has four next hop neighbors, nodes 13, 7, 8 and 4. Its smallest path delay is 15 milliseconds via node 4. Its top-list contains nodes 4, 8 and 7. Each time its has to transmit packet, it randomly chooses one neighbor from its top-list and switches to this neighbor reception channel for transmission.

The fact that the next hop changes from one packet to another allows load balancing on a per hop basis. Indeed, having a top-list of next hop neighbors and making a random choice in this top-list for each data to transmit avoid forwarding all enqueued packets to the same next hop neighbor. That helps balancing the traffic load and avoiding queue overflow and packet loss in the network. The top-list members of each node change over the time to cope with the variation of the routing metric. Some nodes get in the top-list while other get out. In the 14-node network scenarios presented in figures 3.14 and 3.15, we notice that the top-list of node 9 does not contain the same number of nodes. In figure 3.14, the top-list contains two nodes while in figure 3.15 it contains 3 nodes.

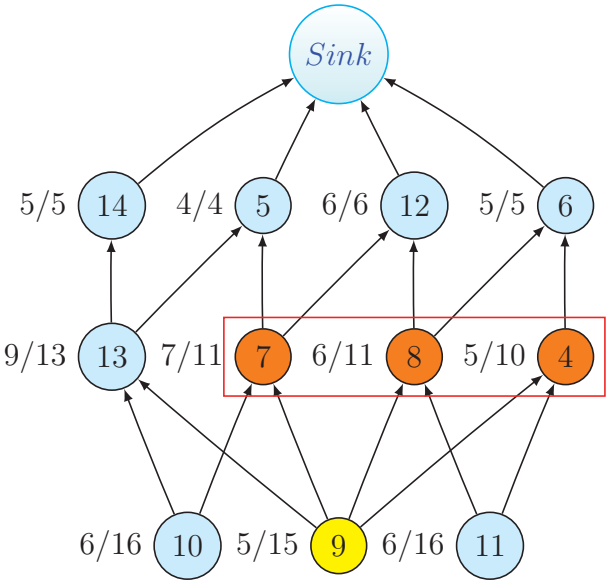


FIGURE 3.13: Node delay d and path delay D. At left of each node is labelled its d/D values. Node 9 top-list contains nodes 7, 8 and 4 encircled in red.

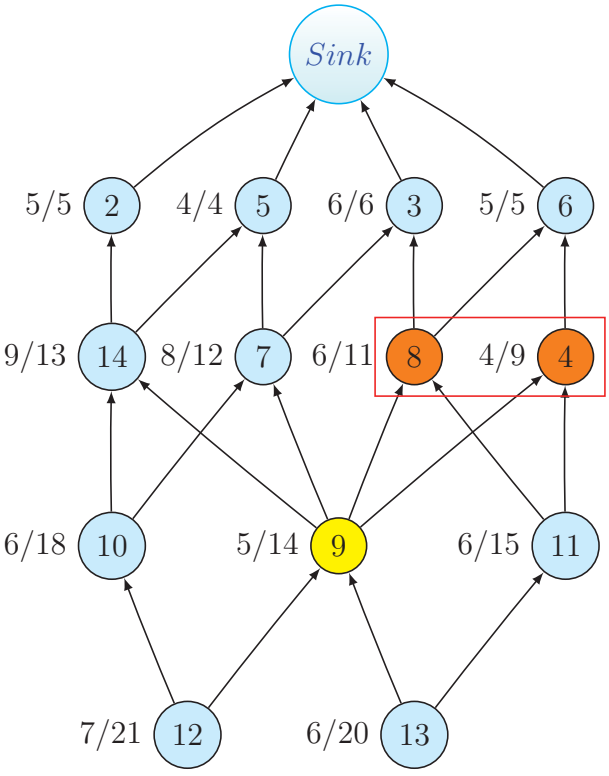


FIGURE 3.14: 14-node network scenario, where node 9 top-list contains 2 neighbors, nodes 4 and 8 encircled in red.

In the network scenario presented in figure 3.15, node 9 has 4 next hop neighbors towards the sink. By means of simulations with queueing delays as routing metric

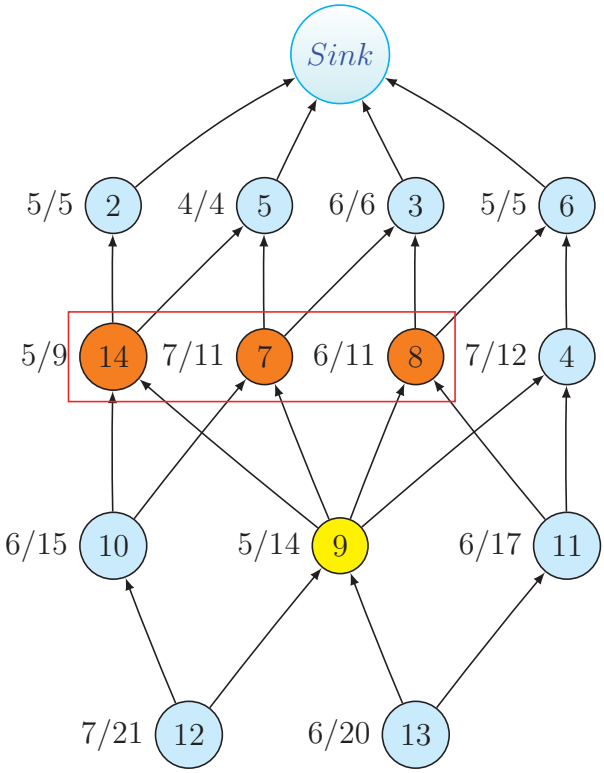


FIGURE 3.15: 14-node network scenario, where node 9 top-list contains 3 neighbors, nodes 14, 7 and 8 encircled in red.

and applying top-list concept to forward data towards the sink, we evaluate node 9 choice distribution among its next hop neighbors. For 100 data packets sent towards the sink by it, we counted the number of packets transmitted to each of its next hop neighbors. Table 3.3 contains the resume of the obtained results. They show that the traffic of node 9 is distributed between its 4 next hop neighbors. As this example shown for node 9, the traffic of other nodes in the network is also fairly distributed among their next hop neighbors until data reach the sink. That helps to improve the network efficiency, mainly the delivery ratio as more extensive results presented in chapter 4 will show.

Queue overflow prevention: packet queue behaviour in dynamic neighborhood WSNs

Wireless sensor nodes are very tiny and have a reduced buffer size. In multi-hop data transmission, packets converge from leaf nodes to the sink through intermediate nodes. In high traffic load scenarios, intermediate nodes receive a lot of data packets and the risk of queue overflow becomes higher despite using a load balancing routing. Queue overflow leads to packet loss that reduces packet delivery ratio. M-CoLBA cares about nodes queue occupancy rate to avoid queue overflow. In what follows, we present node queue behaviour in dynamic WSNs and the queue monitoring approach used by M-CoLBA to avoid overflowing.

Nodes	Percentage of received packets by each node
Node 4	31%
Node 7	19%
Node 8	26%
Node 14	23%

TABLE 3.3: Illustrative results of node 9 traffic distribution among its next hop neighbors. The traffic is balanced between neighbors thanks to the top-list.

When the strategy of the routing protocol uses a metric that varies over time (e.g. expected transmission count, packet queueing delay, etc.), the number of nodes routing their packets to a given next hop neighbor will also vary over time. Thus, for the same time interval, the number of received and transmitted packets by non-leaf nodes is rarely stable [59]. For that reason, the behaviour of non-leaf nodes queues looks like a leaky bucket [60] with both a variable input traffic rate and a variable leak rate. Whenever a node generates or receives a packet from a neighbor to transmit towards the sink, it enqueues it and the number of its enqueued packets increases by 1. When it succeeds a transmission towards the sink, or removes a packet because of transmission failure, its number of queued packets is decreased by 1. We assume that when a node has n packets (with $n \leq k$, where n is a positive integer and k is the packet queue size) in its queue, this node is in state n.
Let $\lambda_{\Delta t}$ and $\mu_{\Delta t}$ be packet enqueueing and packet dequeuing rates per time unit Δt for non-leaf nodes respectively. During Δt time, each non-leaf node may vary its queue occupancy by i packets. The value of i depends on rates of $\lambda_{\Delta t}$ and $\mu_{\Delta t}$.

For a given time Δt, if a node is in state n, the packet queue state transition over time can be represented with a probabilistic Markov chain with three transition probabilities starting from state n. We have the following three possible state transitions:

- the node enqueues more packets than it dequeues ($\lambda_{\Delta t} > \mu_{\Delta t}$) and transits from state n to state $n + i$ with a probability of P_1; the queue is filling up and the state of the node tends to state k,

- the node dequeues more packets than it enqueues ($\lambda_{\Delta t} < \mu_{\Delta t}$) and transits from state n to state $n - i$ with a probability of P_2; the queue is emptying and the state of the node tends to state 0,

- no packets are enqueued or dequeued, or the node enqueues and dequeues the same number of packets ($\lambda_{\Delta t} = \mu_{\Delta t}$) and stays in the same state n with probability P_3, where $P_3 = 1 - (P_1 + P_2)$; the queue remains stable and node stays in state n.

P_1 depends on both packet generation rate and number of received packets during Δt. The number of received packets depends on the number of neighbors that chose this node as a next hop neighbor. P_2 is mainly affected by the channel occupancy in the node neighborhood and probability of successfully transmitting a packet. Non-leaf node state transitions diagram starting from state n over Δt is presented in figure 3.16. When node is in state k, its queue is full and no packet can be enqueued. In state 0, the node queue is empty and k packet slots are available. For a given traffic load, values of P_1 and P_2 are very closely related to the state of the medium and the network congestion.

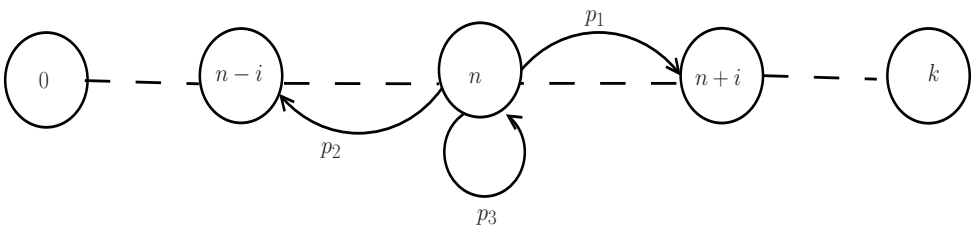
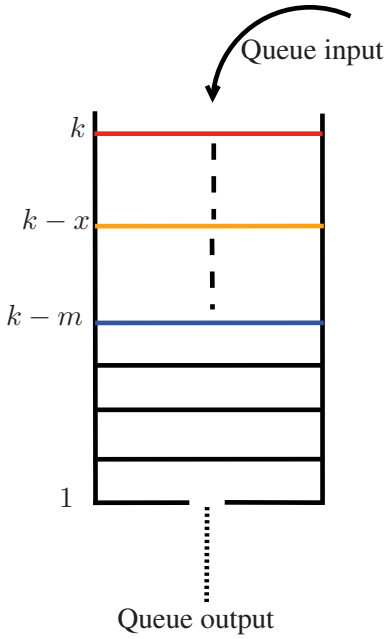


FIGURE 3.16: An example illustrating non-leaf nodes queue state transitions over time.

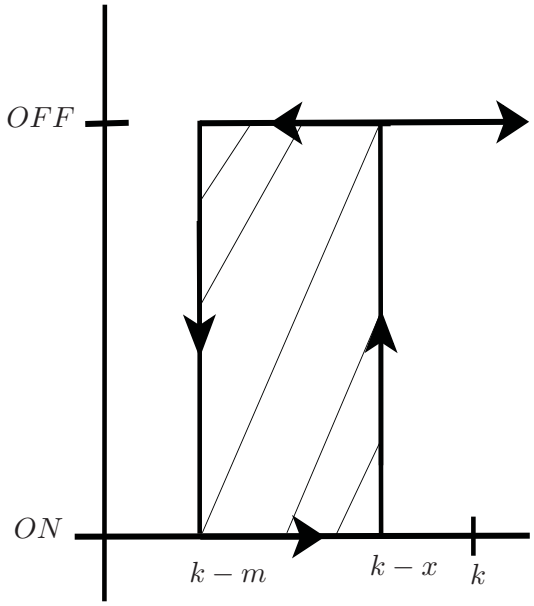
Queue overflow prevention: queue occupancy monitoring

As stated before, k is the packet queue size. When a node reaches the state k, its queue is full and any forwarded or generated packets will be dropped because there is no slot in the buffer to enqueue it. To avoid reaching state k, each non leaf node continuously monitors its queue state and when it reaches state $k-x$ (with x a no null integer smaller that k) it alerts its transmitting neighbors to stop transmitting packet to it. The alert is done by transmitting a beacon frame containing null routing metric to transmitting neighbors. Receiving the alert message, transmitting nodes temporary blacklist this neighbor and select an other next hop in their top-lists. The Blacklisted neighbor will be considered once it get out the queue overflowing situation. Indeed, node has to transmit a beacon frame with a non null metric value when its queue occupancy reach the state $k - m$ (where m is an integer bigger than x but smaller than k). Reaching the state $k - m$ means that there is no risk of queue overflow for this node. The choice of x and m must be done considering the queue size k, the density of the network and the traffic generation rate. We specified the used k, $k - x$ and $k - m$ values in the simulation parameters. Figure 3.17 presents an example of node queue where k, $k - x$ and $k - m$ states are mentioned. With the overflow control, nodes queue behavior follows a hysteresis policy depicted by figure 3.17b. Thanks to the queue monitoring approach used by M-CoLBA, the number of packet loss due to queue overflow is reduced as shown by results in chapter 4.

In this section, we described the multichannel routing protocol relying on our Collaborative Load Balancing Algorithm. In the following section 3.3 we present the single channel version of this protocol.



(A) An example of node queue with k packet slots



(B) An illustration of hysteresis policy

FIGURE 3.17: The $k - x$ indicates the critical threshold of the queue occupancy, $k - m$ the trust threshold and k the queue size. When the critical threshold is reached, transmitting nodes are alerted to stop transmitting to the neighbor. In case it reaches the trust threshold, it is again available to receive data from its neighbors.

3.3 S-CoLBA: the single channel version of CoLBA

As already stated, S-CoLBA is the single channel version of CoLBA and only one communication channel is used by nodes in the network.

3.3.1 Common techniques between M-CoLBA and S-CoLBA

To compute the queueing delay based routing metric and balance the traffic load in the network, S-CoLBA follows approaches used in M-CoLBA. Thus, the node delay d computation 3.2.3, the path delay D computation 18, the top-list construction and the queue overflow avoidance 3.2.5 are the same techniques used in S-CoLBA. Beyond these common approaches, some techniques used by M-CoLBA like channel allocation are not needed in S-CoLBA.

3.3.2 Some approaches used only by S-CoLBA

In single channel networks, nodes communicate using the same channel, therefore, channel allocation is not needed in S-CoLBA. Neighbors discovery is done at the network start ups where nodes exchange beacon frames. When node A receives beacon frame from node B, it considers B as its neighbor. Each node has to build its 1-hop neighbors table instead of 3-hop neighbors table as done in M-CoLBA.

In opposite to M-CoLBA, S-CoLBA does not need synchronisation periods for routing information exchange. Each node routing metric value can be transmitted whenever computed using beacon frame following the scheme described in the following subsection 3.3.2.

Routing metric diffusion and overhead optimization in S-CoLBA

High overhead can reduce the network bandwidth and decrease data throughput. To optimize the network overhead, we introduce a prediction approach to evaluate the necessity of announcing the path delay whenever it is calculated in S-CoLBA, based on the mechanism explained in 20. To do so, we focus on the ratio between the enqueued and dequeued data packets by each node during each Δt time. At each Δt, we compute the ratio between the number of enqueued packets $(\lambda_{\Delta t})$ and the number of dequeued packets $(\mu_{\Delta t})$. We distinguish the two following cases that is used to decide beacon transmission:

- $(\lambda_{\Delta t}/\mu_{\Delta t}) \leq 1$ (1)
- $(\lambda_{\Delta t}/\mu_{\Delta t}) > 1$ (2)

In case (1), if the node queue is not full, the risk of having a queue overflow is low. Even if the node current path delay is different from the previous value the node will not transmit any beacon to announce its path delay variation.

In case (2), node needs to know its residual packet queue capacity and then compares it to the difference between $\lambda_{\Delta t}$ and $\mu_{\Delta t}$. If $(\lambda_{\Delta t} - \mu_{\Delta t})$ is smaller than the residual queue slots, the node does not transmit any beacon. We assume that, in case the difference between the enqueued packets and dequeued ones during the last Δt is less than the residual queue slots, the risk of queue overflow is low, so we do not need to broadcast any beacon. Otherwise, node has to broadcast a beacon frame containing its updated metric value.

3.3.3 Summary for M-CoLBA and S-CoLBA

We presented M-CoLBA, a load balancing multichannel routing protocol. It uses dynamic routing metric that relies on data packets queueing delays and synchronization periods to share routing information. In addition to traffic load balancing, M-CoLBA introduces queue occupancy monitoring to prevent packet loss due to queue overflow. As shown by performance evaluation in chapter 4, M-CoLBA has an optimized packet delivery ratio and outperforms several existing routing protocols.

We also presented the single channel version of CoLBA, designed as S-CoLBA. M-CoLBA and S-CoLBA uses the same routing technique that helps to balance the traffic load. The main difference between them being the number of available channels in the network.

The main drawback of M-CoLBA is the high end-to-end packet delays mainly due to the synchronization periods used for routing information exchange. To optimize this end-to-end delay, we enhanced M-CoLBA by finding a way to share routing information without the need of synchronization. The enhanced version is ABORt (Acknowledgement-Based Opportunistic Routing protocol) that we present in the following section 3.4.

3.4 ABORt: Acknowledgement-Based Opportunistic Routing protocol

ABORt is an enhanced version of M-CoLBA where nodes do not need synchronization periods to share routing information. In ABORt, ACK frames are used in an opportunistic way to exchange routing metric between neighboring nodes. Nodes activity is not organized in cycle like what is done in M-CoLBA. There is only one synchronization period (all nodes tune their transceiver on the same communication channel, used as a semaphore channel) at the network starts up for neighborhood discovery and channel assignment. After that, data packets are continuously transmitted. The neighborhood discovery 3.7, the channel assignment 3.9, the routing metric computation 3.2.3, 18, the load balancing and congestion avoidance approach 3.2.5 are the same to what is done in M-CoLBA. The main enhancement that we did in ABORt is piggy backing routing information in ACK frames that helps to avoid costly synchronization periods once data transmission is started. As what is done in M-CoLBA, nodes start transmitting data to their next hop neighbors using the hop count routing metric that will be progressively replaced by queueing delays routing metric.

3.4.1 Routing metric dissemination in ABORt

ABORt uses ACK frames in an opportunistic way to disseminate the routing metric in the network. We deactivated the MAC layer standard automatic ACK generation and we generate new ACK frames with 2 additional bytes where the routing metric value is piggybacked. At the MAC layer, the standard ACK frame has 3 fields, namely, Frame Control field, Sequence Number field and Frame Control Sequence field (FCS) as presented in figure 3.18.

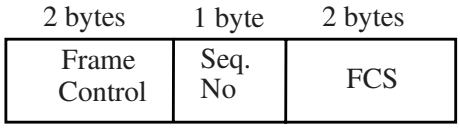


FIGURE 3.18: Fields of the standard ACK frame at the MAC layer. The number of bytes for each field is indicated.

- The Frame Control field is 2 bytes in length and contains information defining the frame type addressing field and other control flags.
- The Sequence Number field has 1 byte in length and specifies the sequence identifier for the data frame.
- The FCS field has 2 bytes and contains 16-bit ITU-T CRC that helps to check the correctness of the frame.

When sink 1-hop neighbors transmit data packets to the sink for the first time, they calculate their path delay D and have to transmit it to their neighbors using the modified ACK frame after receiving data packets from these neighbors. The

structure of this modified ACK frame is presented in figure 3.19 and has one field (Metric) more than the standard ACK.

The Metric field is added to the ACK frame and waiting for data packet reception before to be filled with the up to date metric value.

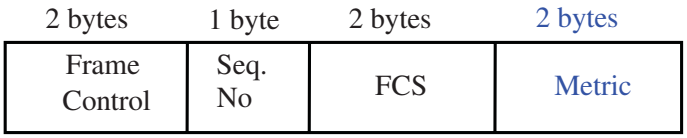


FIGURE 3.19: The MAC layer fields of the modified ACK frame, where a field (Metric) of two bytes is added to piggyback the metric value.

When 2-hop neighbors of the sink receive the ACK frames sent by sink 1-hop neighbors they have the path delay of these neighbors. They select the minimum path delay and add it to they own node delay and that is their path delay. These 2-hop neighbors will also transmit their path delay in ACK frames and the process continues until path delays are computed by leaf nodes.

Like what is done in M-CoLBA, nodes in ABORt also build a top-list of next hop neighbors. Nodes in the top-list will be used for sending frames towards the sink. For each packet, a next hop is randomly chosen from the top-list. This way, nodes will achieve load balancing among their selected neighbors. The top-list contains neighbors with the smallest routing metric within a certain threshold $\triangle T$.

When node receives routing metric value from its neighbor, it has to evaluate the top-list entering condition for that neigbor. So, top-list is continuously updated and nodes that are in the top-list but do not fill any more the entering condition will be removed from it. In case node has more than two next hop neighbors and its top-list contains only one, it must trigger the top-list update process to check if some nodes fill the top-list entering condition. In the following section 3.4.2, we present how to trigger the top-list update.

3.4.2 Triggering top-list update in ABORt

In some scenarios, node top-list may contain only one neighbor while it has several next hop neighbors. That may be due to the fact that when a node is not in the top-list it is not used as next hop to forward data. So, it is not straightforward to get its routing metric value and check the top-list entering condition. To avoid that and keep the top-list always up to date, top-list update is frequently triggered. This update consists of transmitting a data packet to each next hop neighbor and receiving the ACK that contains this neighbor path delay. Doing so helps to get the path delays of all potential next hop neighbors and update the top-list. In order to avoid doing the update process endless when no other path delays are within the threshold, the update process is only triggered after successful transmission of several data packets to the top-list member. The process ends when all neighbors are visited once.

Thanks to piggy backing routing metric values in ACK frames, synchronization periods are not needed any more once the network is set-up and data packets transmission started. ABORt data packet end-to-end delays is optimized (as we can see in performance evaluation in chapter 4) compared to what we got with M-CoLBA.

3.4.3 Number of radio interfaces for the sink in ABORt and M-CoLBA

The sink is the node to which collected data in the network is transmitted. In multichannel communication the sink may be a bottleneck if it has only one channel to receive all transmitted data. Using multi-interface sink helps to reduce the contention level and avoids congestion at sink 1-hop neighbors. Thus, in the multichannel version of our contribution, we use a sink with several radio interfaces where each interface communicate on different channel. The number of radio interfaces of the sink is specified in the simulation parameters in chapter 4

3.5 Conclusion

Introducing high data rate applications in WSNs is not a straightforward task. The limited resources of nodes makes it very challenging to find optimized solution. In this chapter, we presented M-CoLBA, S-CoLBA and ABORt routing protocols to balance traffic load and improve throughput in WSNs. S-CoLBA is the single channel version of CoLBA and ABORt is the enhanced version of M-CoLBA to optimize data packets end-to-end delays. Our contribution fairly distributes the traffic load in the network and avoid losing data due to queue overflow.

M-CoLBA is a communication protocol that uses multichannel and load balancing techniques jointly to enhance network performance. This is achieved by increasing the bandwidth capacity and avoiding data loss due to congestion and queue overflow. Evaluation results presented in chapter 4 shown the efficiency of M-CoLBA compared to some existing protocols.

However, M-CoLBA suffers from high latency essentially due to synchronization for control traffic transmission. So, we enhanced it to ABORt, a multichannel load balancing routing protocol that uses ACK-based control information dissemination. ABORt does not rely on synchronization periods for routing information exchange. Hence, it avoids time and energy wastage in order to construct routes. Performance evaluation shows that ABORt is efficient in terms of delivery ratio, queue overflow, end-to-end delay and overhead compared to other multichannel routing protocols. In the next chapter 4, we present simulation tools and obtained results with S-CoLBA, M-CoLBA and ABORt.

Chapter 4

Results

In chapter 3, we presented our contributions CoLBA in both single channel and multichannel WSNs and also the multichannel enhanced version ABORt. In this chapter, we present the performance evaluation of these contributions by means of simulation and experiment (only the single channel version was experimented). We compare the obtained results of our solutions to some existing routing protocols in the literature. The simulation evaluation is done using Contiki Operating System (OS) [61] and its simulator Cooja [62], with some modifications on the provided CSMA/CA algorithm and the radio propagation model. These modifications aim to provide IEEE 802.15.4 compliant version of CSMA/CA and more realistic radio propagation model for simulations. The chapter is organized in 5 sections. The first section presents Contiki OS and its simulator Cooja and our improvement on them. In section 2, we present the obtained result with our contributions in single channel network (S-CoLBA) and also in multichannel network (M-CoLBA). Section 3 presents results of the improved multichannel version named ABORt. We present single channel experiment results in section 4 and finally section 5 summaries the chapter.

4.1 Contiki Operating System

Contiki is a lightweight and flexible open source operating system for tiny networked devices including sensor nodes. It helps to connect tiny low-cost and low-power micro-controllers to the Internet. It provides a toolbox for building complex wireless systems and has an integrated simulator named Cooja, that helps to simulate different network scenarios.

The MAC layer of Contiki OS provides a version of the CSMA/CA algorithm, but this provided version is not compliant with the IEEE 802.15.4 standard and may cause performance degradation. To overcome this insufficiency, we implemented a compliant version of IEEE 802.15.4 unslotted CSMA/CA in Contiki 3.0 and evaluated its performance [63]. We used this implemented compliant version of CSMA/CA to evaluate our contributions by means of simulation using Cooja that we is present in the next section.

4.2 Cooja simulator

Cooja is a flexible Java-based simulator designed for the simulation of sensor networks running Contiki operating system.

In simulation, the radio propagation model is an important feature. It reflects the environment (interference level and collisions) for which simulations are done. Thus, it is more realistic to use radio propagation model that is close to what may happen where sensor nodes are deployed in real environment.

Cooja simulator provides four radio propagation models: Unit Disk Graph Medium (UDGM) with constant and distance loss, Directed Graph Radio Medium (DGRM), and the Multi-path Ray-tracer Medium (MRM). UDGM with constant loss considers that each node transmission range is an ideal disk. All nodes inside the disk always receive transmitted packets and nodes outside the disk will not receive them. UDGM with distance loss is an improvement of the constant loss model by taking into account interferences. Each packet can be sent and received with a certain probability of success. The probability of success must be configured manually by the user before running the simulation. DGRM radio model allows to define and specify some parameters like success ratio and propagation delay on a per-link basis. The MRM radio model relies on ray tracing approach and integrates capture effect[1]. The signal power at the receiving node is computed relying on the Friis formula presented in equation (1.1).

$$P_r = P_t + D_t + D_r + 20.log_{10}\left(\frac{\lambda}{4\Pi d}\right) \tag{4.1}$$

Where P_r is the signal power at the receiving antenna in dBm. P_t is the signal power at the transmitting antenna in dBm. D_t is the isotropic directivity of the transmitting antenna in the direction of the receiving antenna in dBi. D_r is the isotropic directivity of the receiving antenna in the direction of the transmitting antenna in dBi. λ is the wavelength of the signal and d is the distance between the transmitting antenna and the receiving one.

In real world deployments, the radio signal is subject to many disturbances due to obstacles and interferences. These disturbances provoke the signal attenuation that has an influence on the number of transmission attempts and packet reception rate. This phenomenon makes the transmission range of nodes a distribution that changes from one transmission to another. Thus, representing coverage as an ideal disk like stated by UDGM constant loss is an unrealistic simplification of reality. Moreover, the lack of capture effect in UDGM and DGRM is an unrealistic feature because we have capture effect in real motes.

The variation of nodes transmission range over time leads some changes in the topology of the network. This radio signal disturbance is omnipresent in real environment and becomes worse in urban areas or in industrial regions. In general, to take this phenomenon into account in simulators, the equation of the received signal as a function of emitted power usually includes a random component.

Among the radio propagation models provided by Cooja, MRM is the most realistic one and closest to what will be the radio propagation in real world, however, it

[1]The capture effect is a phenomenon associated with frequency modulation reception in which only the stronger signals with a certain threshold will be demodulated.

lack a random path loss behavior. Thus, we modified MRM radio propagation model to include random component as part of the calculation of the received signal power. We added a path loss exponent φ and a Gaussian random variable X_σ with 0 mean and standard deviation of σ. The formula of the received signal power in dB is presented in equation (1.2).

$$P_r(d) = P_r(d_0) - 10.\varphi.log_{10}(\frac{d}{d_0}) + X_\sigma \tag{4.2}$$

Where $P_r(d)$ is the mean received power at distance d in dB. $P_r(d_0)$ is the reference power at distance d_0 in dB. φ is the path loss exponent and X_σ is a Gaussian random variable having a 0 mean and a standard deviation of σ in dB.

For an accurate simulation, both φ and σ must be measured at the site of the planned deployment. Already existing works show that values for an urban outdoor area range between 2.7 and 5 for the path loss exponent φ and between 4 dB and 12 dB for the standard deviation σ [64].

Adding a random component with a Gaussian standard deviation makes each node transmission range change from one transmission to another. That helps to replicate what may happen when a sensor network is deployed in real environment.

The modified MRM radio propagation model with its new feature is used to evaluate our contribution and compare it to other existing works. The next section present evaluated scenarios and obtained results.

4.3 CoLBA results in both single channel and multichannel networks

We evaluated CoLBA performance in both single channel and multichannel WSNs by means of simulation using Cooja simulator where sensor nodes are running Contiki OS. Each node is simulated with the same sensor node capacities as sky motes to ensure realistic parameters and behavior of nodes.

Simulations are performed using IEEE 802.15.4 MAC and physical layers. Nodes access the medium using IEEE 802.15.4 unslotted CSMA/CA algorithm. Simulation parameters are presented in table 4.1 and all results presented in this section 4.3 are obtained using these parameters.

As mentioned in table 4.1, we used a packet queue length of 8 packets (the default queue size in Cooja simulator) with a 50-byte packet size (a very used packet size in simulations). We also disabled the duty-cycle mechanism to keep nodes awake all the time. We varied the packet generation rate in $\{1, 5, 10\}$ packets/second/node. We used an area of 200 x 200 m^2 where sensor nodes are randomly scattered, but we

ensure that the network is connected (no isolated nodes).

In order to obtain different network densities, we continuously doubled the number of nodes from 10 to 80 with one sink that has 3 radio interfaces (only for multichannel networks) and one interface for other nodes. We have to notice that in single channel communication, the sink has one radio interface.

The threshold to be in the top-list is $\triangle T$= 2 milliseconds more than the smallest path delay (D) with beacon phase θt equal 2 seconds and data phase ωt fixed to 20 seconds. Weighting factors α and β are fixed to 1 and 2 for the first five dequeued packets and last five dequeued ones respectively in the node delay (d) computation.

Parameters	Values
MAC protocol	IEEE 802.15.4 CSMA/CA
Duty-Cycle	Deactivated
Radio Model	MRM with random component
Packet queue size (k)	8 packets
Critical threshold ($k - x$)	6 packets
Trust threshold ($k - m$)	3 packets
Data packet size	50 bytes
Simulation surface	200 x 200 m^2
Number of nodes	10, 20, 40 and 80
Packet generation rate	$\{1, 5, 10\}$ packets/s/node
Available channel in multichannel	16
Transmission power	0 dBm
Reception threshold	-90 dBm
Each simulation duration	2 minutes
Number of iterations	10
Sink interfaces in multichannel	3
Path loss exponent (φ)	2.74
Standard deviation (σ)	5 dB
Threshold for top-list members ($\triangle T$)	2 milliseconds
Beacon phase duration (θt)	2 seconds
Data phase duration (ωt)	20 seconds
Weight factor for first 5 packets (α)	1
Weight factor for last 5 packets (β)	2

TABLE 4.1: Simulations parameters used to evaluate CoLBA performance

From here, we designate the single channel version of CoLBA by S-CoLBA. We compared the efficiency of S-CoLBA of our contribution to the IETF standardized routing protocol RPL [45] with the ETX objective function which also operates on single channel. To show that using multichannel MAC protocols helps to improve network throughput with respect to single channel, we compared M-CoLBA with

both S-CoLBA and RPL. To evaluate M-CoLBA efficiency compared to multichannel routing protocol, we compared it to M-HopCount-Sync. M-HopCount-Sync is a multichannel routing protocol which routing metric selects the shortest path in number of hops. Nodes activity is organized in cycles like what is done in M-CoLBA. The channel allocation in M-HopCount-Sync is semi-dynamic as what we did in M-CoLBA.

In the presented results, each value is an average of 10 different simulations using the same number of nodes. Metrics used to evaluate performance are Packet Delivery Ratio (PDR), throughput, packet loss due to queue overflow, network overhead and end-to-end packet delay.

4.3.1 Data Packet Delivery Ratio

The PDR is the ratio between the number of received packets by the sink and the number of generated packets by all nodes in the network. The delivery ratio gives an idea about the efficiency of the routing protocol to successfully transmit data from nodes towards the sink. For each scenario and each packet generation rate, we evaluate the PDR at the sink.

In the low data traffic scenario (generation of 1 packet per second per node), the PDR of the 4 protocols (M-CoLBA, M-HopCount-Sync, S-CoLBA and RPL) are high (more that 90%) and very close as shown in figure 4.1 with a slight advantage for M-CoLBA. In low traffic load networks, nodes do not have much data to transmit, thus, the medium is free most of the time. That makes the risk of collisions and packet loss low regardless of the routing protocol and the use of multiple channels.

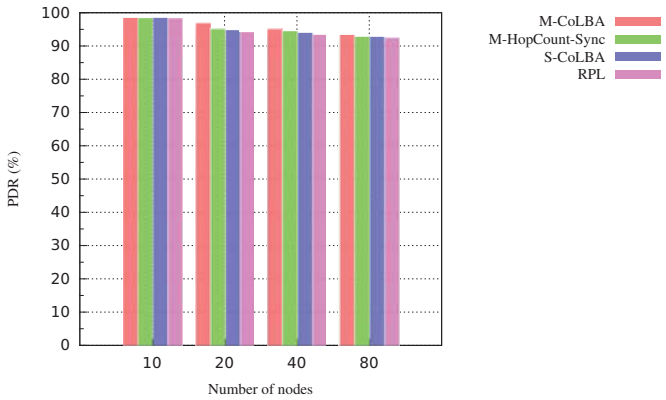


FIGURE 4.1: Packet Delivery Ratio for generation of 1 packet per second per node with different network size.

When traffic load is higher (generation of 5 and 10 packets per second per node), the advantage of M-CoLBA becomes more important as we can see in figures 4.2 and 4.3. We notice that the PDR gap between M-CoLBA and M-HopCount-Sync varies between 2 and 12 points. Which means that, the number of lost packets in

M-HopCount-Sync is high compared to M-CoLBA. The low delivery ratio of M-HopCount-Sync is due to its static routing metric with lack of load balancing. As nodes in the network are not mobile, the smallest hops from each node to the sink is static and the shortest path from each node towards the sink remains the same. Thus, the traffic load is not fairly distributed and same paths are used to forward data towards the sink. Some nodes are under-loaded while others are overloaded which provokes congestion, queue overflow and packet loss.

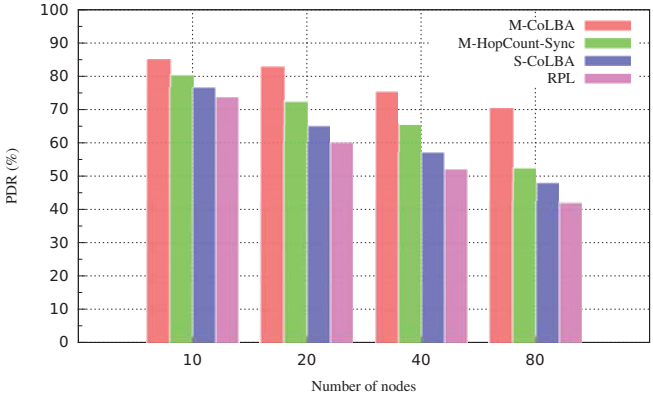


FIGURE 4.2: Packet Delivery Ratio for generation of 5 packets per second per node with different network size.

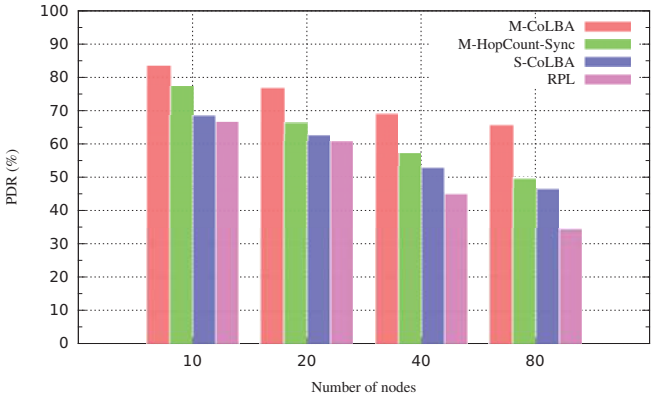


FIGURE 4.3: Packet Delivery Ratio for generation of 10 packets per second per node with different network size.

The low delivery ratio of RPL and S-CoLBA compared to both M-CoLBA and M-HopCount-Sync is mainly due to high contention, interference and collision in the single channel shared medium that leads to packet loss. The delivery ratio of S-CoLBA outperforms RPL and the gap increases with the packet generation rate. Indeed, S-CoLBA is based on a load balancing routing and queue overflow avoidance that helps to mitigate packet loss due to congestion and queue overflow in a high traffic load scenario from which RPL suffers.

The load balancing routing scheme coupled with the queue overflow avoidance thanks to monitoring queue occupancy level help to increase the delivery ratio in both CoLBA versions.

4.3.2 Throughput evaluation in kilobits per second

In this section we evaluate the throughput in kilobits per second (kb/s) at the sink node according to the overall offered load (G) in the network.

By gradually increasing G from 4 kb/s to 320 kb/s, we computed the received throughput at the sink and the obtained results are presented in figure 4.4.

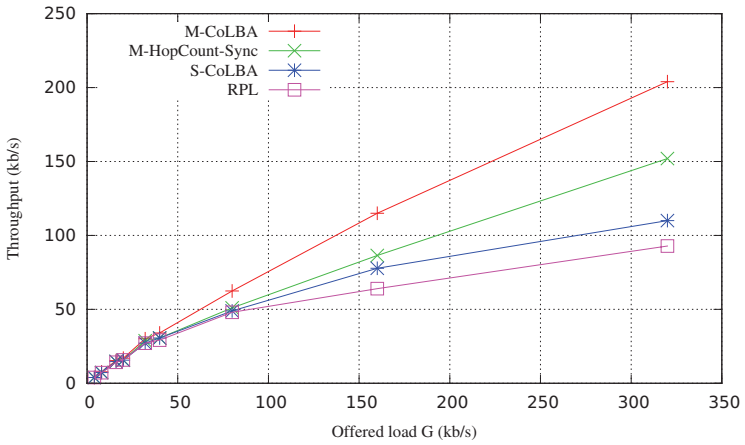


FIGURE 4.4: The received throughput at the sink node according to the offered load G in kb/s. G is gradually increased from 4 kb/s to 320 kb/s.

We notice that as already seen with the delivery ratio, multichannel protocols (M-CoLBA and M-HopCount-Sync) outperform single channel ones (S-CoLBA and RPL). That is thanks to parallel data transmissions, less interferences and collision and especially the fact of using multi-interface sink in these multichannel networks.

The highest throughput obtained with S-CoLBA is 112 kb/s, which represents 44,8% of the theoretical reachable throughput, namely 250 kb/s. With M-CoLBA we can reach 204 kb/s which is about 81.6% of the 250 kb/s reachable throughput. The high throughput of M-CoLBA compared to S-CoLBA is due to using multiple channels and one sink having multiple radio interfaces those can receive data simultaneously.

Increasing the number of interfaces for the sink by X (X a positive integer greater than 1) does not multiply the throughput by X. Indeed, with one sink having one radio interface, S-CoLBA can reach 44,8% of 250 kb/s and with one sink having 3 radio interfaces, M-CoLBA can reach only 81.6% of 250 kb/s.

4.3.3 Packet loss due to queue overflow

Queue overflow causes packet loss, which reduces delivery ratio and network throughput. To ensure that M-CoLBA and S-CoLBA avoid queue overflow, we counted the number of packets dropped due to queue overflow for the 4 protocols. Figures 4.5 and 4.6 present the percentage of lost packets in log scale compared to the total number of data packets generated in the network for generation rate of 5 and 10 packets per second per node. When generating 1 packet per second per node, there were no packets dropped for all 4 network sizes with all 4 protocols.

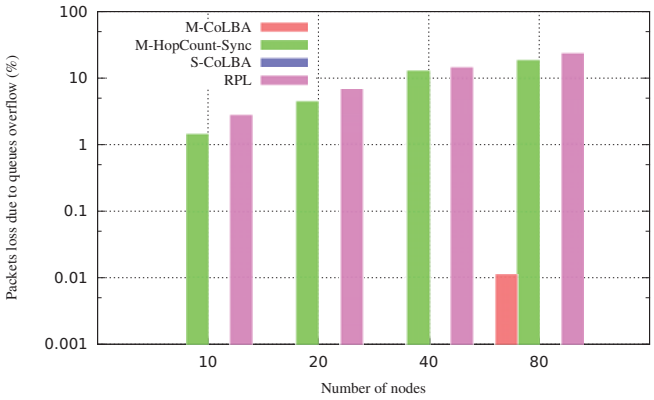


FIGURE 4.5: Percentage of lost packets due to queue overflow for generation of 5 packets per second per node in different networks size.

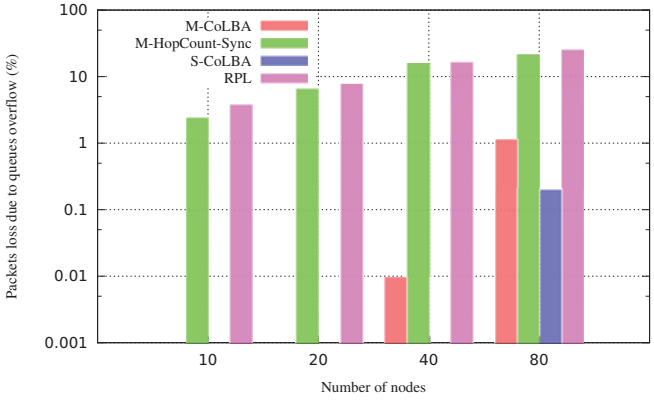


FIGURE 4.6: Percentage of lost packets due to queue overflow for generation of 10 packets per second per node in different networks size.

Queue overflow is almost null for S-CoLBA and M-CoLBA. In the worst simulated scenario (80 nodes with 10 packets per node per second) presented in figure 4.6, S-CoLBA loses 0.042% and M-CoLBA looses 1,08% of the generated packets. To prevent queue overflow, S-CoLBA and M-CoLBA use beacons frames to alert transmitting nodes when the queue of their next hop neighbors is reaching its maximum

capacity. To be efficient, this alert message must be sent to all transmitting nodes at the same time. S-CoLBA being a single channel routing protocol, all nodes in the network communicate using the same channel. They continuously listen to this channel, that facilitates broadcasting an alert message to the transmitting nodes when the receiving node queue is nearly full. However, M-CoLBA uses multiple channels, thus, the broadcast support is more complex and nodes do not get the alert message at the same time. This is the main reason it drops more packets due to queue overflow than S-CoLBA.

The number of lost packets due to queue overflow of RPL and M-HopCount-Sync is higher. RPL is a single channel routing protocol, so there are no parallel transmissions in the same communication range. With high traffic load, the contention level increases and generated or forwarded data packets spend more time in queues, which increases the risk of overflow. Moreover, the load balancing approach of RPL is only metric based (neighbor with the best routing score is selected), which is not enough to fairly distribute data traffic on per hop basis. Thus, some nodes have more traffic to forward while others are under-loaded. In M-HopCount-Sync, packets lost due to queue overflow is also high because the routing metric is static, nodes always chose the same next hop. Moreover, it does not use an alert message to inform transmitting nodes when the receiving node is suffering from queue overflow.

The fair traffic distribution and the overflow alerts mechanism used by M-CoLBA help to significantly reduce lost packets. The fact of using beacon frames to prevent queue overflow may affect the network overhead. In the next subsection 4.3.4, we present results concerning the traffic overhead.

4.3.4 Network overhead

We consider as overhead, beacon frames used to share the network service information (network topology, channel allocation and routing metric).

At the network set-up phase, nodes need to exchange control messages (beacon frames) to discover their neighborhood, in order to build the network and also to allocate channels in the cases of M-CoLBA and M-HopCount-Sync. The use of control messages leads to a certain amount of overhead, which has a direct impact on channel congestion and energy consumption. In this section we evaluate the overhead of each routing approach in terms of number of generated beacon frames.

Figures 4.7, 4.8 and 4.9 present the number of generated beacon frames for each protocol according to the number of nodes in the network and packet generation rate. Results show that on the one hand, except S-CoLBA, the overhead of the other 3 protocols are almost the same for each network size regardless of the packet generation rate. On the other hand, RPL generates more overhead than both M-CoLBA and M-HopCount-Sync for the same network size.

We notice that S-CoLBA overhead increases with the number of nodes and packet generation rate. This is because, when the traffic load becomes high, more packets are enqueued and S-CoLBA metric value varies more often. Thus, each node needs

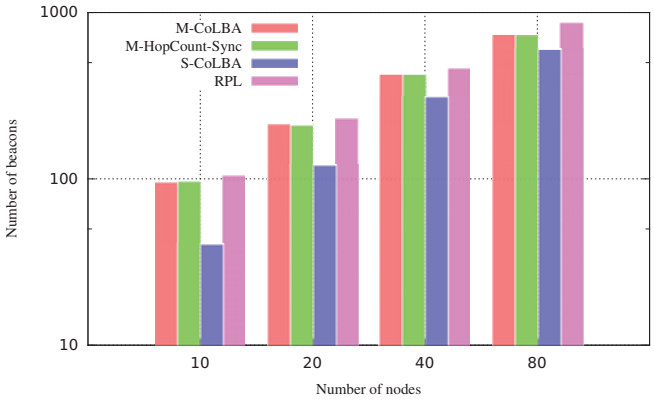


FIGURE 4.7: Overhead for traffic generation of 1 packet per second per node with different network size.

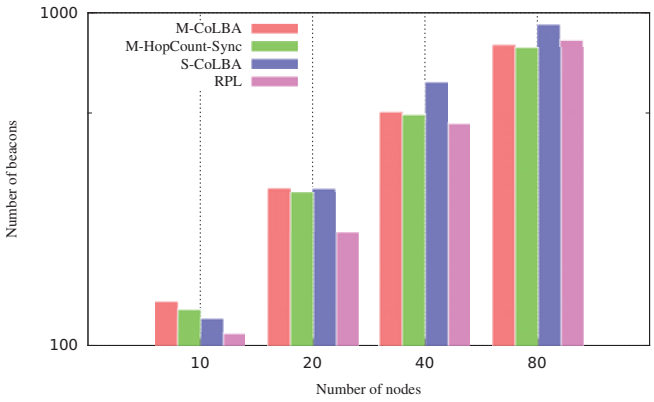


FIGURE 4.8: Overhead for traffic generation of 5 packets per second per node with different network size

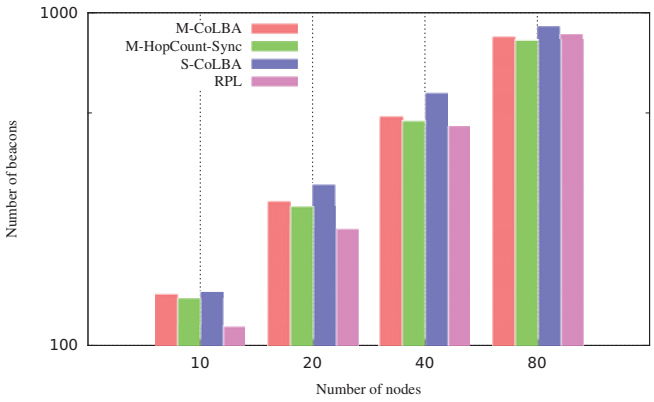


FIGURE 4.9: Overhead for traffic generation of 10 packets per second per node with different network size

to transmit additional beacon frames to inform its neighbors about its new metric value or to alert them about the risk of its queue overflow. We also notice that, in the scenarios with less nodes and light traffic, S-CoLBA has less overhead than the other protocols because the medium is less occupied and S-CoLBA metric value is almost stable with a low risk of queue overflow. So less alert message are transmitted, that

reduced the overhead.

In RPL, overhead is mainly due to DIO, DOA, DAO-ACK and DIS messages. The number of DIO messages is largely higher than the others, because DIO messages are used to build and maintain the network topology. DIO messages generation is governed by the trickle algorithm [65] timer regardless of the data traffic generation rate. This is why the overhead of RPL is not much affected by the packet generation rate. Thus, the overhead of RPL remains stable for the same number of nodes but higher than M-CoLBA and M-HopCount-Sync due to the important number of DIO, DAO and DIS message used to build and maintain the network topology.

In M-CoLBA and M-HopCount-Sync, beacon frames are generated at the network start-up phase and during each beacon phase used for routing metric exchange. In addition to that, M-CoLBA may generate beacon frames during the data transmission phase in case of a high risk of queue overflow. In that case, beacon frames are sent to alert neighbor nodes to stop transmitting packets to overloaded neighbors. That is why in some scenarios, M-CoLBA has slightly more overhead that M-HopCount-Sync. Which does not use alert message to prevent queue overflow.

To sum up, we notice that the overhead of RPL is higher than M-CoLBA and M-HopCount-Sync mainly due to the important number of network service messages (DIO, DAO, DAO-ACK and DIS) generated by it to build and maintain the network topology. S-CoLBA also generates more beacon frames than RPL when the traffic load becomes high in the network.

An other performance metric that we evaluated is data packets average end-to-end delay that we present in the next section 4.3.5

4.3.5 End-to-end packet delay

In this section, we present the average values of end-to-end delays according to the distance in number of hops from node generated the packet to the sink. The end-to-end delay is the time difference between the reception moment of the packet by the sink node and its generation moment by the source node.

Figures 4.10 and 4.11 present the end-to-end delays respectively for the scenario of 10 nodes with 1 packet/second/node and the scenario of 80 nodes with 10 packets/second/node. We chose to present only the results of these two scenarios because all the results follow the same tendency. These results give an idea of the end-to-end delays in a small network with relatively low traffic (10 nodes with 1 packet per second per node) and dense network with heavy traffic situation (80 nodes with 10 packets per second per node).

We notice that M-CoLBA and M-HopCount-Sync have longer end-to-end delay compared to RPL and S-CoLBA. Their long end-to-end delays are mainly due to the beacon phases. Which lasts 2 seconds and during this phase data is not transmitted, only beacon frames are exchange to update routing and channel allocation information. Indeed, data packets which do not reach the sink before a beacons phase need to wait in node queue until to the end of this phase before its transmission towards the sink. However, in RPL and S-CoLBA, we do not need this additional delay because

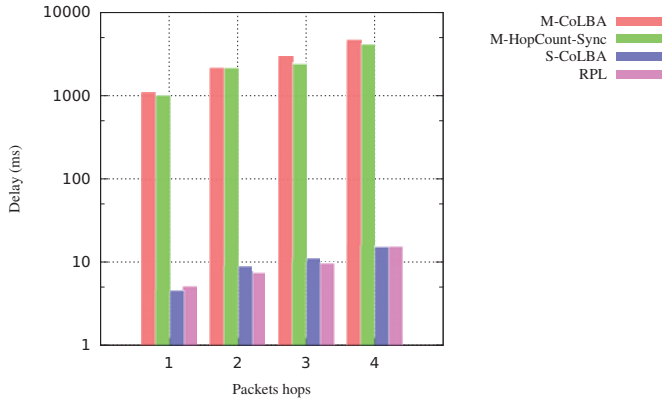


FIGURE 4.10: End-to-end delay, scenario of 10 nodes, generation of 1 packet per second per node.

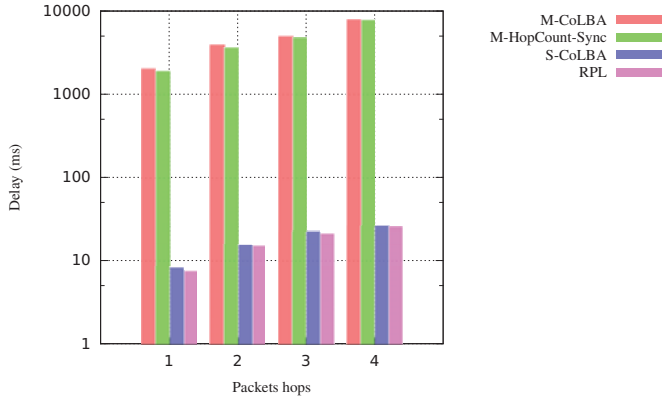


FIGURE 4.11: End-to-end delay, scenario of 80 nodes, generation of 10 packets per second per node.

all nodes use the same channel and there is no need to allocate a dedicated period for control message broadcast.

These results show the difficulty to broadcast messages in multichannel network and the negative consequences of synchronization periods on data packet end-to-end delays. Finding a way to exchange routing information without introducing synchronization periods can help to optimize packet end-to-end delays.

Using high data rate applications in WSNs is a challenging task. We proposed and analysed CoLBA, a communication protocol that uses load balancing and queue overflow avoidance techniques to enhance network performance. Its performance was evaluated in both single channel and multichannel WSNs. Results show that it outperforms some existing routing protocols namely M-HopCount-Sync and RPL in terms of packet delivery ratio and packet lost due to queue overflow. On the other hand, M-CoLBA suffers from high latency compared to S-CoLBA and RPL essentially due to channel and time reservation for control traffic. To mitigate this

drawback, we designed a technique that allows nodes to exchange control information during data exchange without the need to allocate a dedicated timeslot for broadcasting beacons. This enhanced version of M-CoLBA is called ABORt which performance evaluation is presented in the next section 4.4.

4.4 ABORt performance evaluation

In this section, we study the performance of ABORt, the enhanced version of M-CoLBA. ABORt is a routing protocol for multichannel WSNs that avoids synchronization phases once the network is in operational mode. It aims to achieve low packet end-to-end delays and limited overhead. ABORt performance is evaluated by means of simulation using Cooja simulator with parameters shown in table 4.2. These parameters are same to those we have used to evaluate M-CoLBA except the fact that ABORt does not need time periods θt and ωt as the operation of the network is not organized in cycle.

Parameters	Values
MAC protocol	IEEE 802.15.4 CSMA/CA
Duty-Cycle	Deactivated
Radio Model	MRM with random component
Packet queue size (k)	8 packets
Critical threshold ($k - x$)	6 packets
Trust threshold ($k - m$)	3 packets
Data packet size	50 bytes
Simulation surface	200 x 200 m^2
Number of nodes	10, 20, 40 et 80
Packet generation rate	$\{1, 5, 10\}$ packets/s/node
Available channels in multichannel	16
Transmission power	0 dBm
Reception threshold	-90 dBm
Each simulation duration	2 minutes
Number of iterations	10
Sink interfaces in multichannel	3
Path loss exponent (φ)	2.74
Standard deviation (σ)	5 dB
Threshold for top-list members ($\triangle T$)	2 milliseconds
Weight factor for first 5 packets (α)	1
Weight factor for last 5 packets (β)	2

TABLE 4.2: Simulations parameters used to evaluate ABORt performance.

We evaluate ABORt efficiency compared to three multichannel routing protocols, namely M-CoLBA, M-HopCount and M-HopCount-Sync. M-HopCount uses a routing metric that selects the shortest path in number of hops that we implemented using

the same ACK-based metric exchange method of ABORt. M-HopCount-Sync also uses a routing metric based on shortest path in number of hops, but nodes activity is organized in cycles. It is the same protocol that we compared with M-CoLBA in section 4.3. We also compared ABORt with a single channel routing protocols, RPL with ETX objective function.

Presented results in each graph is an average of 10 simulations using the same number of nodes. We evaluated the five following routing metrics: PDR, throughput, packet loss due to queue overflow, network overhead and end-to-end packet delay.

4.4.1 Data Packet Delivery Ratio

Results presented in figure 4.12 show that, in light data traffic scenarios, where each node generates 1 packet per second, the PDRs of all protocols are high (between 92% and 99%) and very close with a slight advantage for ABORt. Indeed, when the network is under-loaded, the medium is free most of the time, which results in collision-free transmissions. Thus, using multiple channels is not very useful and optimizing the routing protocol has very little impact.

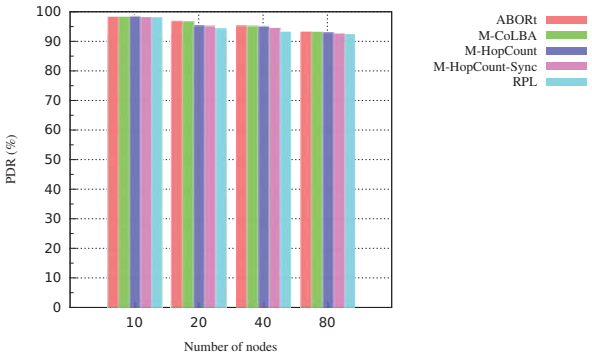


FIGURE 4.12: Packet delivery ratio for generation of 1 packet per second per node with different network size.

When traffic load is higher (generation of 5 or 10 packets per second per node), the advantage of ABORt becomes more important as shown in figures 4.13 and 4.14. In RPL, nodes communicate using the same channel. Thus, its low delivery ratio is due to, on the one hand, contention, interference and collision, leading to packet loss. On the other hand, the lack of traffic load balancing during the data transmission which leads to congestion and queue overflow.

The delivery ratio gap between ABORt and M-HopCount varies between 6 and 20 points. The low delivery ratio of M-HopCount and M-HopCount-Sync compared to ABORt and M-CoLBA is due to their load unbalanced static routing scheme. In a static network, the minimum number of hops from each node to the sink is static and the shortest path from each node towards the sink remains the same. Nodes always use the same next hop to forward their data packets. Thus, some nodes are overloaded, which provokes congestion, queue overflow and packet loss leading to low delivery ratio.

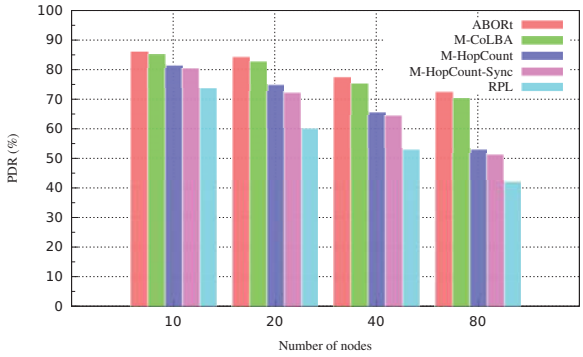


FIGURE 4.13: Packet delivery ratio for generation of 5 packets per second per node with different network size.

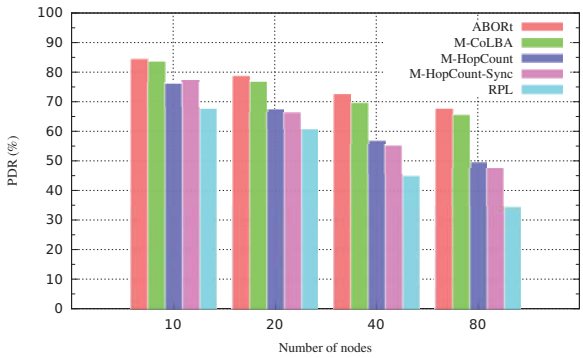


FIGURE 4.14: Packet delivery ratio for generation of 10 packets per second per node with different network size.

When the traffic load becomes high 4.13 and 4.14, ABORt outperforms M-CoLBA. Thanks to piggybacking routing information in ACK frames in ABORt, nodes have updated queue state of their next hop neighbors that helps to keep an up to date top-list with the smallest queueing delay neighbors. Moreover, information about the risk of queue overflow is piggybacked in ACK frames (by introducing null metric value) that helps to reduce the number of transmitted beacon frames and get more time for data transmission. In the following section 4.4.2, we present the received throughput at the sink node according to the offered load.

4.4.2 Throughput evaluation in kb/s

This section focuses on delivery ratio to evaluate the throughput at the sink node in kb/s according to the offered load (G) in the network.

The received throughput at the sink for the 5 protocols (ABORt, M-CoLBA, M-HopCount, M-HopCount-Sync and RPL) is presented in figure 4.15. The throughput is evaluated for 9 different values of G, namely 4 kb/s, 8 kb/s, 16 kb/s, 20 kb/s, 32 kb/s, 40 kb/s, 80 kb/s, 160 kb/s and 320 kb/s.

As already observed in section 4.3.2, we notice that the throughput of multichannel protocols (ABORt, M-CoLBA, HopCount-Sync and HopCount) is higher than

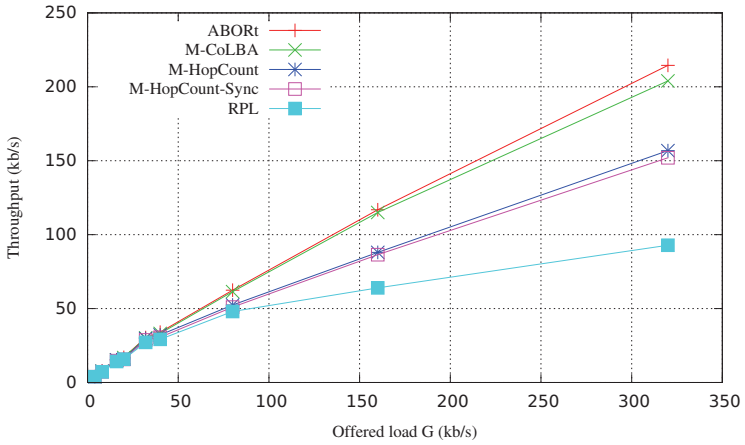


FIGURE 4.15: The received throughput at the sink node according to the offered load G in kb/s. G is gradually increased from 4 kb/s to 320 kb/s.

the single channel one (RPL). Indeed, thanks to using multi-interface sink and multiple channel, the throughput of multichannel protocols is increased.

As already shown by results of delivery ratio, ABORt outperforms all the compared protocols. When G reaches 320 kb/s, the received throughput with ABORt is 214.5 kb/s against 204 kb/s for M-CoLBA. Piggybacking routing information in ACK frames helps ABORt to reduce beacon frames transmission and get more time for data transmission, that improves its throughput.

4.4.3 Packet loss due to queue overflow

One of the negative effects of congestion in WSNs is queue overflow, which leads to packet loss. The number of dropped packets due to queue overflow reflects the congestion level in the network. Figures 4.16 and 4.17 present the ratio of lost packets in log scale compared to the total number of generated packets in the network with the generation of 5 packets per second per node and 10 packets per second per node, respectively. We did not observe any lost packet due to queue overflow in all scenarios with 1 packet per second per node for all evaluated protocols.

Results in figure 4.16 show that the number of lost packets due to queue overflow for ABORt and M-CoLBA is very low. In the worst case of simulated scenarios, M-CoLBA loses 1.11% and ABORt 1.02% of the generated packets. M-CoLBA and ABORt use multiple channels; thus, broadcasting a message is not straightforward to do as in single channel and nodes do not get the alert message about the risk of queue overflow at the same time. That is why we observe some lost packets in M-CoLBA and ABORt in the scenarios of 40 and 80 nodes with 5 and 10 packets per second.

The number of lost packets due to queue overflow of RPL, M-HopCount and M-HopCount-Sync is higher. RPL is a single channel routing protocol, so there are no parallel transmissions in the same communication range. With high traffic load, the contention level becomes high and generated or forwarded data packets spend more

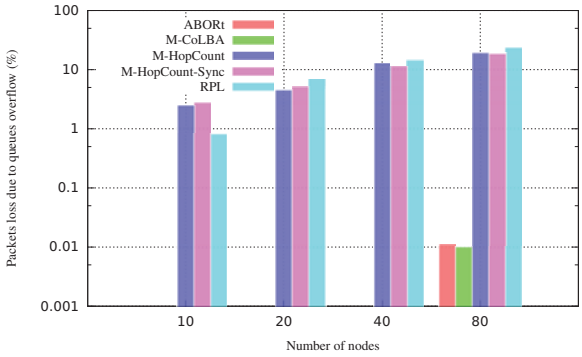


FIGURE 4.16: Queue overflow ratio with packet generation of 5 packets per second per node with different network scenarios.

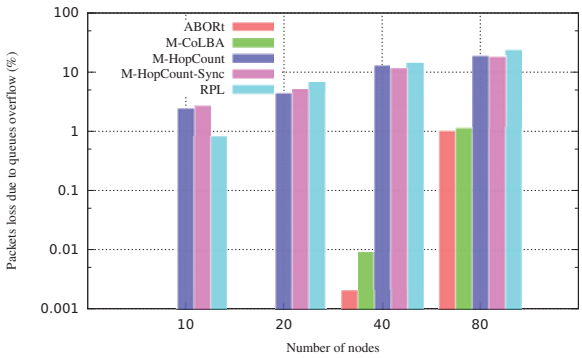


FIGURE 4.17: Queue overflow ratio with packet generation of 10 packets per second per node with different network scenarios.

time in queues, which increases the risk of overflow. Moreover, the load balancing approach of RPL is only metric based, which is not enough to fairly distribute data traffic on per hop basis. Then, some nodes are overloaded leading to queue overflow while others are less used. M-HopCount and M-HopCount-Sync use static routing metrics. Which overloads nodes that are part of the shortest paths of many nodes towards the sink. When the packet generation is high, it leads to queue overflow. Moreover, unlike ABORt and M-CoLBA, M-HopCount and M-HopCount-Sync do not use an alert message to inform transmitting nodes when the receiving node queue is near to full.

Using top-list to balance traffic load and alert message to prevent queue overflow help M-CoLBA and ABORt to reduce the number of lost packets due to queue overflow compared to other protocols. In the next section 4.4.4, we present results about overhead.

4.4.4 Overhead evaluation

Control messages leads to a certain amount of overhead, which has a direct impact on channel congestion and energy consumption. Figures 4.18–4.20 present the number of generated beacon frames for each protocol according to the number of nodes in the network and packet generation rate.

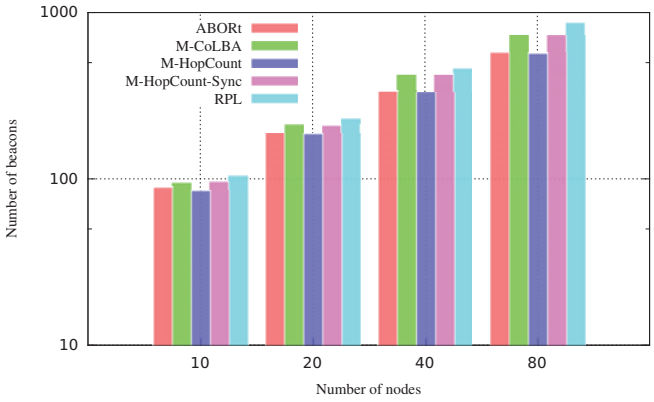


FIGURE 4.18: Overhead for packet generation of 1 packet per second per node.

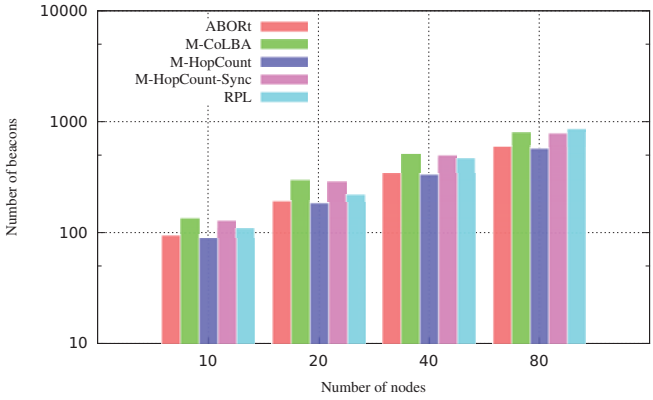


FIGURE 4.19: Overhead for packet generation of 5 packets per second per node.

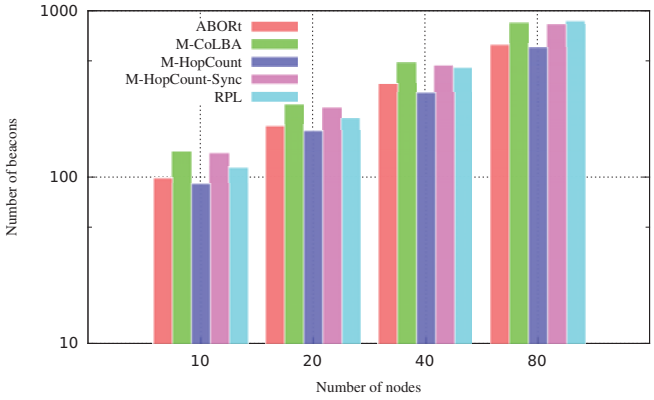


FIGURE 4.20: Overhead for packet generation of 10 packets per second per node.

Results show that the overhead of RPL, ABORt and M-HopCount are almost the same for each network size regardless of the packet generation rate. In RPL, the overhead is mainly due to DIO, DOA, DAO-ACK and DIS messages. DIO messages generation is more higher and follows the trickle timer regardless the traffic load in the network. That is why the overhead of RPL is not much affected by the packet

generation rate. Thus, the overhead of RPL remains stable for the same number of nodes.

In M-HopCount and ABORt, beacon frames are generated only at the network start-up phase. Thus, for the same network size the number of generated beacon frames does not change much. In all 3 packet generation rates, most of the time, ABORt generates more beacon frames than M-HopCount. In some scenarios, M-HopCount has slightly more overhead than ABORt essentially due to the random behavior of beacon frames transmissions during the start-up phase using CSMA/CA algorithm.

In M-CoLBA and M-HopCount-Sync, beacon frames are generated at the network start-up phase and also during each beacon phase used for routing metric exchange. That is why they have more overhead than ABORt and M-HopCount where the synchronization phase is not needed because the routing metric information is disseminated using ACK frames. In M-CoLBA, in addition to the set-up phase and beacon phases, beacon frames can also be generated during the data transmission phase in case of a high risk of queue overflow. In that case, beacon frames are sent to alert neighbor nodes to stop transmitting packets to overloaded neighbors. That is why M-CoLBA has slightly more overhead than M-HopCount-Sync.

The approach used by ABORt and M-HopCount, which consists of piggybacking routing information in ACK frames in an opportunistic way, helps to reduce the network overhead. Avoiding synchronization periods may have an advantage on packet end-to-end delays. In the following section 4.4.5 we present results on packets delays.

4.4.5 End-to-end packet delay

The end-to-end delay is the time difference between the reception instant of the packet by the sink and its generation instant by the source node. Figures 4.21–4.23 present the end-to-end delay according to the number of hops for the scenario of 40 nodes (note that all the results of the other scenarios have the same average tendency).

Results show that M-CoLBA and M-HopCount-Sync end-to-end delays are higher than all other protocols because it uses a synchronization phase for beacon exchange. The synchronization lasts 2 seconds in order to reach all nodes in the network and during that time data packets are not sent. We also notice that the end-to-end delays of the three other protocols are close with a slight advantage for RPL. Indeed, RPL has a slightly lower PDR, which means that more packets are lost and packets that are not lost have less competition and waste less time in queues.

M-CoLBA has longer end-to-end delay than ABORt, the gap varies from 1000 milliseconds to more than 9000 milliseconds. This performance of ABORt is mainly due to avoiding synchronization periods for routing information exchange and piggybacking this information in ACK frames. Thus, ABORt enhances M-CoLBA by reducing data packets end-to-end delay.

Piggybacking routing information in beacon frames avoids allocating periods for beacon frames exchange. That helps to reduce data packet end-to-end delay as results shown for ABORt and M-HopCount.

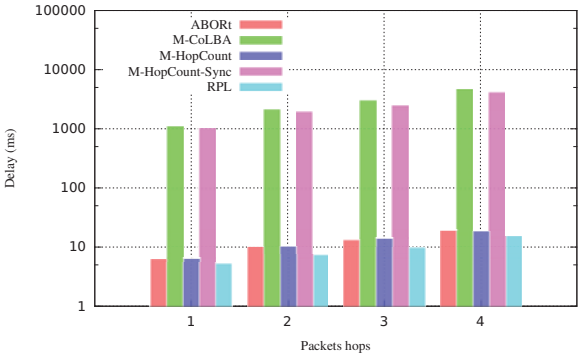


FIGURE 4.21: Packet end-to-end delay according to the number of hops from source node to the sink. Generation of 1 packet per second per node.

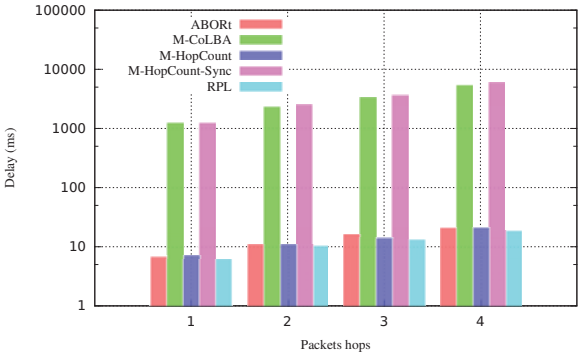


FIGURE 4.22: Packet end-to-end delay according to the number of hops from source node to the sink. Generation of 5 packets per second per node.

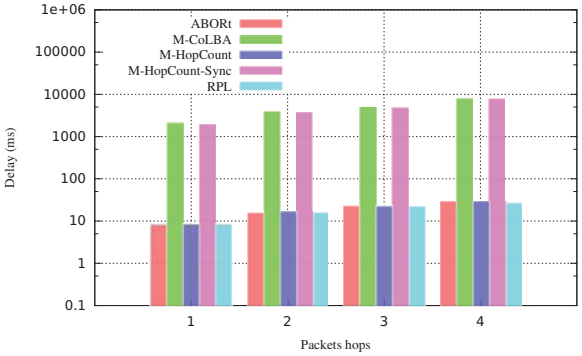


FIGURE 4.23: Packet end-to-end delay according to the number of hops from source node to the sink. Generation of 10 packets per second per node.

We presented obtained results with ABORt, the enhanced version of M-CoLBA compared to several four other routing protocols. ABORt is a multichannel load balancing routing protocol that uses ACK-based control information dissemination. It does not rely on synchronization phase for network and routing information exchange. So, it avoids time wastage in order to construct routes. This performance

evaluation shows that ABORt is efficient in terms of delivery ratio, queue overflow, overhead and mainly end-to-end delay compared to M-CoLBA and three others routing protocols. In the following section 4.5, we present evaluation of the single channel version of CoLBA.

4.5 Experimental evaluation

Evaluating the routing protocol by experiment helps to know its behavior and efficiency in the physical world. In this section we present the experimental evaluation set-up and obtained results with our single channel contribution S-CoLBA using sensor motes. Firstly, we describe the experiment environment and we present the evaluated network scenarios. Secondly we analyse PDR and throughput obtained results by comparing them to those of RPL and S-HopCount. S-HopCount being the single channel version of M-HopCount compared to ABORt in section 4.4. In S-HopCount, the routing metric is static and nodes select path with the minimum number of hops towards the sink to forward their data.

4.5.1 Experiment environment

Our experiment is conducted indoor in practical classrooms. Each room contains around 14 computers equipped with Wi-Fi transceiver in off mode. Multiple Wi-Fi Access Points (APs) were active on the 2.4 GHz band around these rooms. The experiment is done using TelosB motes (they are presented in figure 4.24) communicating on the channel number 26 of IEEE 802.15.4 standard when operating on the unlicensed 2.4 GHz band. The TelosB motes are equipped with the CC2420 transceiver and are battery-powered.

4.5.2 Network set-up and parameters of the experiment

We setted up 10 and 20 motes network topologies in 3 classrooms as show in figures 4.25 and 4.26. We numbered the three rooms with number 1, 2 and 3 as presented in the figures. In each room, motes are placed on tables about 1 meter from the ground. Motes transmission power level was set at -25 dBm to ensure a multi-hop topology network.

In the 10-mote network, the topology is stable and the number of hops from each node to the sink is the same. That is not the case with the 20-mote network, where the topology continues changing due to the higher number of motes in rooms 3 and 2 and the instability of the radio link.

Motes generate respectively 1 packet/second/mote, 5 packets/second/mote and 10 packets/second/mote destined to the sink. These packets are transmitted over single hop or multi-hop to the sink where each received packet is logged with the sender *ID*, its corresponding sequence number and the number of hops it did to reach the sink. The sink is connected to a computer as shown in figure 4.27, where all logs are stored for evaluation purpose. Details about how to upload the code of the routing protocol is presented in appendix B

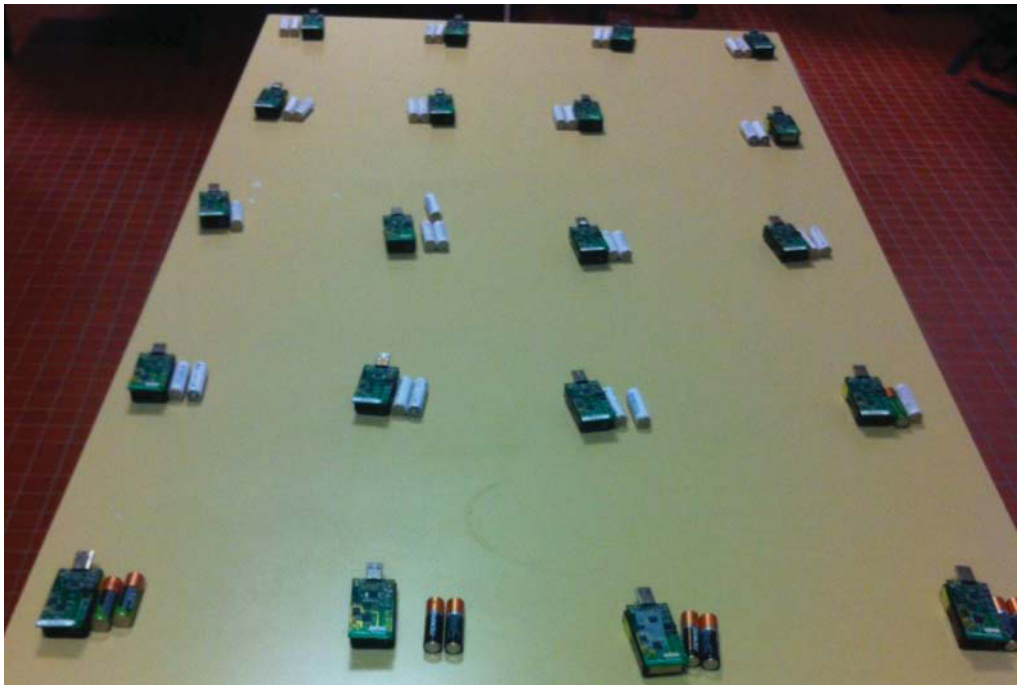


FIGURE 4.24: The used TelosB motes and their batteries for the experimentation.

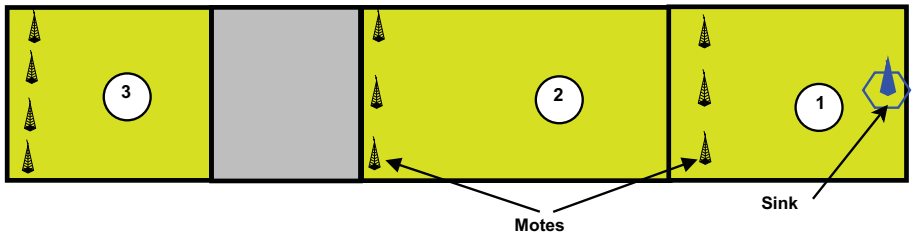


FIGURE 4.25: 10-mote deployed in 3 classrooms for the experiment. When data packet is received at the sink, its number of hops from the source mote until to the sink is printed. We noticed that motes in rooms number 3, 2 and 1 are respectively 3 hops, 2 hops and 1 hop far from the sink. Motes in room 3 forward their data to those in room 2. Motes in room 2 also forward their generated or forwarded data to those in room 1 which transmit them to the sink.

We evaluated the PDR and throughput for both 10 and 20 motes network scenarios with generation of 1, 5 and 10 packets per second per mote. The used parameters for the experiment are presented in table 4.3.

4.5.3 PDR results for experimented scenarios

Figures 4.28–4.30 present the PDR for both simulation and experiment scenarios with S-CoLBA, RPL and S-HopCount respectively for 1 packet/second/mote, 5 packets/second/mote and 10 packets/second/mote. We notice that in low traffic rate (generation of 1 packet/second/mote), the 3 protocols have very close PDR as we can see

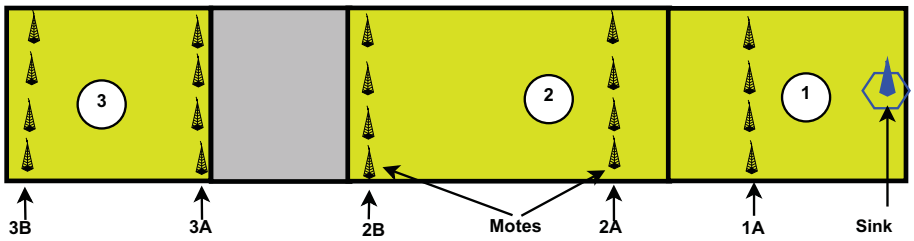


FIGURE 4.26: 20-mote deployed in 3 classrooms for the experiment. In this scenario, the network topology is not stable. For example, the number of hops from motes in row 3B to the sink fluctuates between 2 and 3. Some time motes in row 2B are next neighbors for those in row 3B and another time it changes to motes in row 2A. Most of the time, motes in row 2A can directly transmit their data to the sink but in some rare cases, they forward to motes in row 1A.

FIGURE 4.27: Sink mote connected to the computer to store logs about all received data.

in figure 4.28. However, when the traffic load increases, S-CoLBA PDR outperforms the others in both simulation and experiment. In figures 4.29 and 4.30, results show that S-CoLBA has the highest PDR followed by RPL and then S-HopCount. The tendency of experiment results follows those of the simulation where S-CoLBA is more efficient than RPL and HopCount according to the PDR metric.

We also notice that the simulation results for each protocol outperform those of experiment. Indeed, in physical world, the external environment has a negative impact on the radio signal. Motes are subject to interference leading to collisions and packet loss. Several Wi-Fi access point are deployed around the experiment area.

Parameters	Values
MAC protocol	IEEE 802.15.4 CSMA/CA
Duty-Cycle	Deactivated
Packet queue size (k)	8 packets
Critical threshold ($k - x$)	6 packets
Trust threshold ($k - m$)	3 packets
Data packet size	50 bytes
Number of motes	10 and 20
Packet generation rate	$\{1, 5, 10\}$ packets/s/mote
Number of iterations	10
Number of used channels	1
Threshold for top-list members ($\triangle T$)	2 milliseconds
Weight factor for first 5 packets (α)	1
Weight factor for last 5 packets (β)	2

TABLE 4.3: Experiment parameters used to evaluate S-CoLBA performance.

These access points also operate on the same frequency band 2.4 GHz, which is used by motes in the experiment. That increases the probability of interference and collisions. Moreover, motes in the network placed in different rooms may interfere leading to packet loss and low PDR. So, the main reason of low PDR in the experiment compared to those of simulation is due to both intra-network and inter-network interference and collisions.

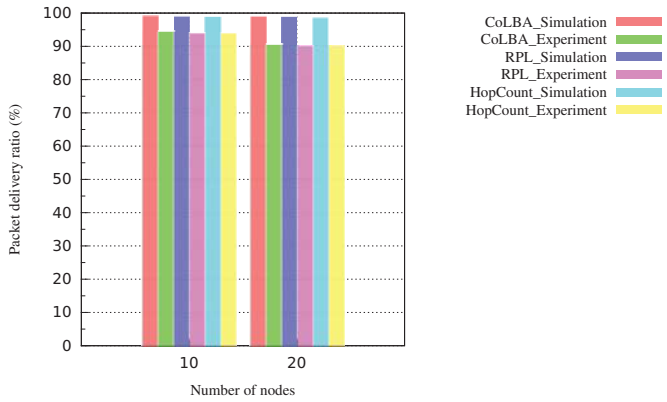


FIGURE 4.28: PDR for both simulation and experiment when generating 1 packet per second per mote.

4.5.4 Throughput evaluation in kb/s

In this section we used results of delivery ratio to evaluate the throughput received by sink node in kb/s according to the offered load (G) in the network.

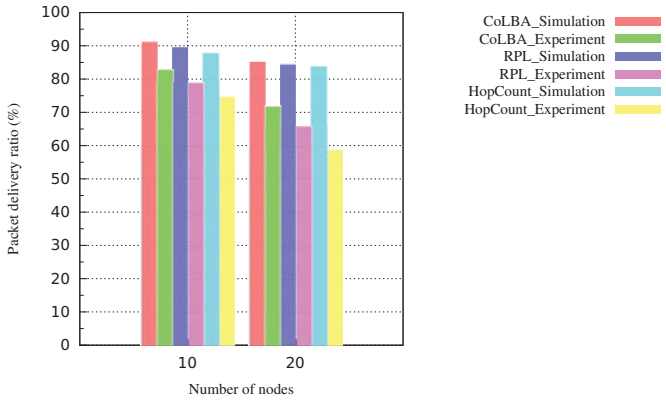


FIGURE 4.29: PDR for both simulation and experiment with generation of 5 packets per second per mote.

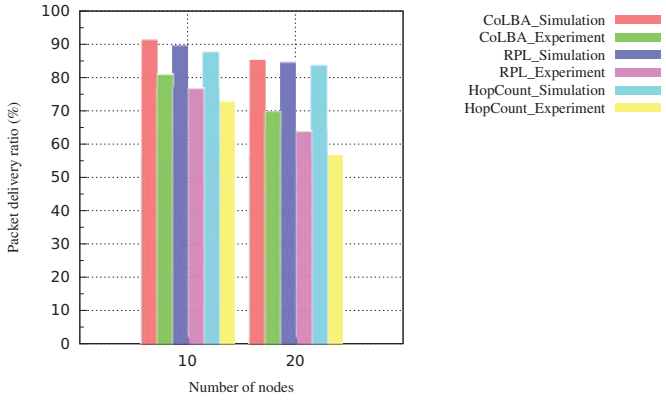


FIGURE 4.30: PDR for both simulation and experiment for generation of 10 packets per second per mote.

Figure 4.31 presents the throughput at the sink node for simulation and experiment evaluation of each protocol according to G. The throughput is evaluated for 5 different values of G, namely 4 kb/s, 8 kb/s, 20 kb/s, 40 kb/s and 80 kb/s.

We notice that for 4 kb/s and 8 kb/s, all evaluated protocols have almost the same values. When G becomes higher than 8 kb/s, throughput obtained by simulation outperforms the throughput in experiment. As already said, this gap is due to interference (inter-network and intra-network) and collisions experienced by sensor motes during the experiment.

In both simulation and experiment, the throughput obtained with our proposed routing protocol CoLBA outperforms the compared routing protocols RPL and HopCount. That is due to the load balancing and queue overflow avoidance schemes used in our protocol.

The experiment evaluation confirms results obtained in simulation where S-CoLBA outperforms RPL in terms of PDR and throughput. These experiment results are a proof of concept and help to validate the efficiency of our contribution in single

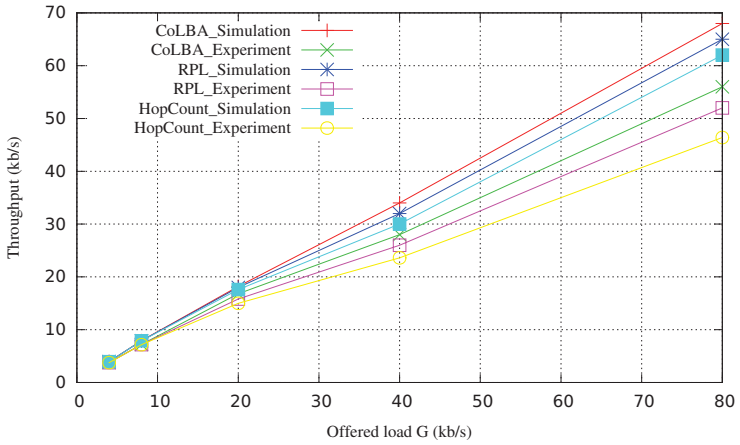


FIGURE 4.31: The received throughput at the sink node according to the offered load G in kb/s. G is gradually increased from 4 kb/s to 80 kb/s.

channel when compared to the well-known IETF standard routing protocol RPL. The future work is to experiment the multichannel version of our contribution.

4.6 Summary

In this chapter we presented simulation results in both single channel (S-CoLBA) and multichannel (M-CoLBA and ABORt) of our contributions and obtained results in experiment with the single channel version. We also compared them to both single channel and multichannel routing protocols existing in the literature.

In the first part, we showed the ability of S-CoLBA and M-CoLBA to improve delivery ratio by reducing packet loss. Results show that, when considering delivery ratio, throughput, queue overflow and overhead, they both outperform the well known single channel routing protocol RPL and M-CoLBA is more efficiency than M-HopCount-Sync. These good performances of both CoLBA versions is mainly due to the traffic load balancing routing technique coupled with queue overflow avoidance scheme. However, the performances of M-CoLBA is limited when considering data packet end-to-end delay. So, we improved it in the enhanced version ABORt.

In the second part, we presented the obtained results with ABORt and we compared them to those of RPL, M-CoLBA and two others multichannel routing protocols, M-HopCount and M-HopCount-Sync. Obtained results show that ABORt outperforms M-CoLBA and the others three protocols on delivery ratio, end-to-end delay and overhead. Moreover, ABORt gains in performance when the traffic load increases. This efficiency of ABORt is due to piggybacking routing information in ACK frames that helps to optimize data packet end-to-end delay and reduces overhead. With less overhead, the protocol have more time to transmit data packets, that enhanced its PDR and throughput.

In the third part, we presented the experiment results of S-CoLBA and compared them to those of RPL and S-HopCount. These results show that S-CoLBA outperforms both of compared protocols.

Obtained results in both simulation and experiment confirm the efficiency of our load balancing dynamic routing scheme in term of packet delivery ratio. That helps to improve the network throughput.

In the next chapter we conclude the thesis and highlight some perspectives to extend this work.

Chapter 5

Conclusion and perspectives

In this chapter we conclude the thesis by reminding the addressed problem, summarizing our contributions, and highlight some perspectives.

5.1 Conclusion

The goal of this thesis was to find an efficient solution against congestion and queue overflow problems mainly at sink neigbors nodes in high data rate convergecast WSNs. It aimed to provide a load balanced routing scheme for data collection in heavy traffic load network that aims to optimize the throughput. Since congestion and queue overflow are key issues that limit reaching high throughout, we focus on how to mitigate them by proposing traffic load balancing routing protocols. The proposed routing scheme fairly distributes the traffic load in per hop basic in the overall network.

5.1.1 Load balancing and queue overflow avoidance

To avoid congestion and queue overflow in data collection WSNs, we proposed a traffic load balancing routing protocol in both single channel (S-CoLBA) and multichannel (M-CoLBA) networks. These protocols fairly distribute the traffic load in per hop basis in the overall network. Using queueing delay based routing metric coupled with a random choice in a list of next hop neighbors, each node that is out of range to the sink, fairly distributes its traffic to its next hops neighbors. Doing so helps to avoid overloading some nodes while others are under-loaded.

In addition to load balancing, both CoLBA versions prevent queue overflow by continuously monitoring packet queue occupancy. Indeed, each node monitors its queue occupancy level, when a critical threshold is reached, which means the queue is close to be full, an alert message is sent to transmitting neighbors and they have to avoid transmitting any more data towards this overloaded next hop neighbor. Thanks to these alert messages, CoLBA reacts faster to avoid queue overflow. Hence, the alerting node will no longer be considered as an eligible node by its neighbors until it gets out of this critical situation. Once its queue occupancy rate drops, it sends new messages which allows its neighbors to reconsider it as potential next hop. This queue occupancy monitoring helps to avoid queue overflowing.

Evaluation results reveal that M-CoLBA suffers from high latency essentially due to synchronization periods dedicated to time reservation for control traffic transmission. So, we enhanced it by designing a technique that allows nodes to exchange

control information during data exchange without the need to allocate a dedicated timeslot for broadcasting beacons. That helped to reduce end-to-end delay and overhead.

5.1.2 Optimization of overhead and data end-to-end delay

In order to optimize packet end-to-end delay, we enhanced M-CoLBA and this optimized version is called ABORt. It provides an alternative approach for routing information sharing in multichannel communication by using MAC layer acknowledgements in an opportunistic way. Routing metric value is piggybacked in ACK frames and transmitted whenever a data packet is received from neighbor. Thus, when node receives an ACK frame it has also the routing metric value of the node that generates this frame. Doing so helps to avoid costly synchronization periods during the operational phase of the network for exchanging control traffic and enhances throughput and access delay. It also helps to reduce the overhead because we do not need other beacon frames to transmit routing information.

The simulation results show that our contributions outperform several existing routing protocol in the literature (namely RPL, M-HopCount-Sync and M-HopCount) when considering performance metric like PDR, throughput and queue overflow. ABORt is also efficient in terms of end-to-end delay compared to M-HopCount-Sync and M-HopCount.

5.1.3 Experiment Evaluation

We also evaluated the single channel version of our contribution by experiment and compared it to 2 other single channel routing protocols. The obtained results show the efficiency of S-CoLBA and its ability to increase the delivery ratio than RPL and S-HopCount. This efficiency is mainly due to load balancing routing approach coupled with the queue overflow avoidance scheme of our contribution.

This experiment is a proof of concept of our contribution. However, the used network scenarios do not put enough emphasis on the addressed problem but these first results obtained with motes remain positive.

5.2 Perspectives

The contributions of this thesis can be extended in several directions. In what follows we enumerate some of them.

5.2.1 Comparing with more multichannel protocols

In the presented results, the performances of the multichannel versions of the contributions (M-CoLBA and ABORt) are compared to routing protocols that use number of hops routing metric, which is a static metric. In the future we aim to compare our contributions to dynamic routing metric multichannel protocols like mDARAL [66].

5.2.2 Modification of mechanisms and parameters

Adding new nodes in ABORt

Some applications may require to add nodes in the network while it is in operation. However, the actual version of ABORt does not allow adding nodes once the network is started. Because neighbors discovery and channel allocation are done once at the network start-up. So, we would like to improve the proposed scheme to allow integrating nodes at any time in the network.

Energy consumption optimization

Energy is a scarce resource in WSNs as nodes are battery powered. Thus, it is necessary to optimize its consumption. The energy consumption of our contributions can be optimized by introducing duty-cycling in nodes operation.

In some circumstances, nodes may have a significant amount of data to transmit for a short time followed by a long period of time when there is no more data to send. Our routing scheme can be modified to follow this kind of operation. Moreover, when nodes are continuously awake it may lead to quickly energy depletion that provokes network partition.

As shown by results, for the same time period, our contribution can transmit more data to the sink than the compared protocols (our contributions have higher delivery ratio). Thus, when nodes haves the same number of data frames to transmit to the sink, those running our contribution will finish first and get more time to sleep and save energy.

Evaluation with longer packet queue size

For the performances evaluation, we used a queue size that can store 8 data packets. This choice was done in relation to the memory size of TelosB mote for the experiment purpose. In fact, TelosB mote has 8 kilobits (KB) Random Access Memory (RAM) that must be shared to store data packets and also for other processing that require memory. With 50-bytes packet size, the 8 packets will need the half of the RAM and the other half will be allocated for others processing. So, we need a trade-off between packet queue size and the amount of memory reserved for processing. That is why we limited the queue size to 8. In future work, we will evaluate the performance of our contribution with a longer data packet queue size, namely queue size more than 10.

5.2.3 Experiment consolidation

The experiment provides more accurate results and allows to reveal more interesting details on the solution. We experimented the single channel version of the contribution. In the future, we would like to perform the experiment with the multichannel versions (M-CoLBA and ABORt) of the contributions using the sixteen available channels in the 2.4 GHz frequency band.

Moreover, all performed experiments are done indoor in classrooms. We would like in future work to run outdoor experiments and compare results to those obtained with indoor experiments.

5.2.4 Topology modification

Evaluation with more than 80 nodes network scenarios

We aim to evaluate our contribution with higher number of nodes network scenarios. In the presented results, the evaluated networks have at most 80 nodes. We would like to increase the number of nodes in the network (around 200) and evaluate the performance.

Evaluation in nodes mobility scenarios

Finally, we would like to evaluate our contributions with node mobility network scenarios. Our contributions has been proposed and evaluated in static networks, where nodes are at the same place during the overall simulation time. However some applications in sensor networks require nodes mobility [67]. Adapting our contribution to cope with node mobility is another aim.

Publications

A. International journal papers

1. **Hamadoun Tall** and Gérard Chalhoub. *ABORt: Acknowledgement-Based Opportunistic Routing Protocol for High Data Rate Multichannel WSNs*. In Journal of Sensor and Actuator Networks, vol. 6, no 4, p. 23, 2017.

2. **Hamadoun Tall**, Gérard Chalhoub, Nadir Hakem and Michel Misson. *Load balancing routing with queue overflow prediction for WSNs*. In Wireless Networks journal, pp. 1-11, 2017

3. **Hamadoun Tall**, Gérard Chalhoub and Michel Misson. *Implementation and performance evaluation of IEEE 802.15.4 unslotted CSMA/CA protocol on Contiki OS*. In Annales des Télécommunications 71(9-10), pp. 517-526, 2016.

B. International conference papers

1. **Hamadoun Tall**, Gerard Chalhoub and Michel Misson. *ComLoB: link quality and queuing delay based Composite routing protocol for traffic Load Balancing in WSNs*. To appear in the proceeding of IEEE MENACOMM 2018, Jounieh, Lebanon.

2. Gérard Chalhoub, **Hamadoun Tall**, Jinpeng Wang and Michel Misson. *DFTR: Dynamic Fault-Tolerant Routing protocol for convergecast WSNs*. In the proceeding of IEEE 86th VTC-Fall 2017, pp. 1-7.

3. **Hamadoun Tall**, Gérard Chalhoub and Michel Misson. *M-CoLBA: Multichannel Collaborative Load Balancing Algorithm with queue overflow avoidance in WSNs*. In the proceeding of IWCMC 2017, pp. 2127-2133.

4. Jinpeng Wang, Gérard Chalhoub, **Hamadoun Tall** and Michel Misson. *Routing Protocol Enhancement for Mobility Support in Wireless Sensor Networks*. In the proceeding of ADHOC-NOW 2017, pp. 262-275.

5. **Hamadoun Tall**, Gérard Chalhoub and Michel Misson. *CoLBA: A Collaborative Load Balancing Algorithm to Avoid Queue Overflow in WSNs*. In the proceeding of DSDIS 2015, pp. 682-687.

6. **Hamadoun Tall**, Gerard Chalhoub and Michel Misson. *Implementation of IEEE 802.15.4 unslotted CSMA/CA protocol on Contiki OS*. In the proceeding of PEMWN 2015, Hammamet, Tunisia.

Appendix A

Slotted and unslotted CSMA/CA algorithms

A.1 Unslotted CSMA/CA

The IEEE 802.15.4 unslotted CSMA/CA aims to reduce interferences and collisions by providing a random access to the medium. The main part of this algorithm is presented in figure A.1 and described by the following paragraph.

For each transmission attempt, each node holds the following two variables: *NB* and *BE*. *NB* (number of backoff) is the number of times the CSMA/CA algorithm was required to backoff while attempting the current transmission. And *BE*, the backoff exponent, which is the maximum number of backoff periods that node will wait before attempting to assess the channel. First, *NB* and *BE* are respectively initialized to 0, and $macMinBE$. At the reception of a data packet, any transmission activity is delayed (backoff) for a random number of backoff periods in the range $[0; 2^{BE} - 1]$ [step 1]. After this delay, channel sensing is performed by doing clear channel assessment (CCA) [step 2]. If the channel is assessed to be busy [step 3], *NB* and *BE* are increased by 1, ensuring that *BE* is not bigger than $macMaxBE$, and the algorithm returns to [step 1]. In case *NB* reaches $macMaxCSMABackoff$, the algorithm will unsuccessfully terminate (failure), which means that the node does not succeed in accessing the channel. If the channel is assessed to be idle, the CSMA/CA succeed and the transmission may start (success).

A.2 Slotted CSMA/CA

The slotted CSMA/CA is a variant of the CSMA/CA algorithm. It uses synchronization to avoid continuously monitoring the state (idle or occupied) of the radio medium that may help to save nodes energy. This variant of CSMA/CA is applied for sending a frame during the Contention Access Period (CAP) of the super-frame, except for sending beacon and acknowledgement frames. The algorithm is based on a time unit called backoff period, which is equal to a *aUnitBackoffPeriod* symbols, which is fixed to 20 symbols.

Slotted CSMA/CA algorithm is based on the three following parameters: *NB*, *BE* and *CW*. *NB* is the Number of Backoffs, it is the number of times node attempt to access the medium for the same packet. *BE* for Backoff Exponent, is the exponent of the backoff that defines the time interval in which the backoff time must be fired. *CW*

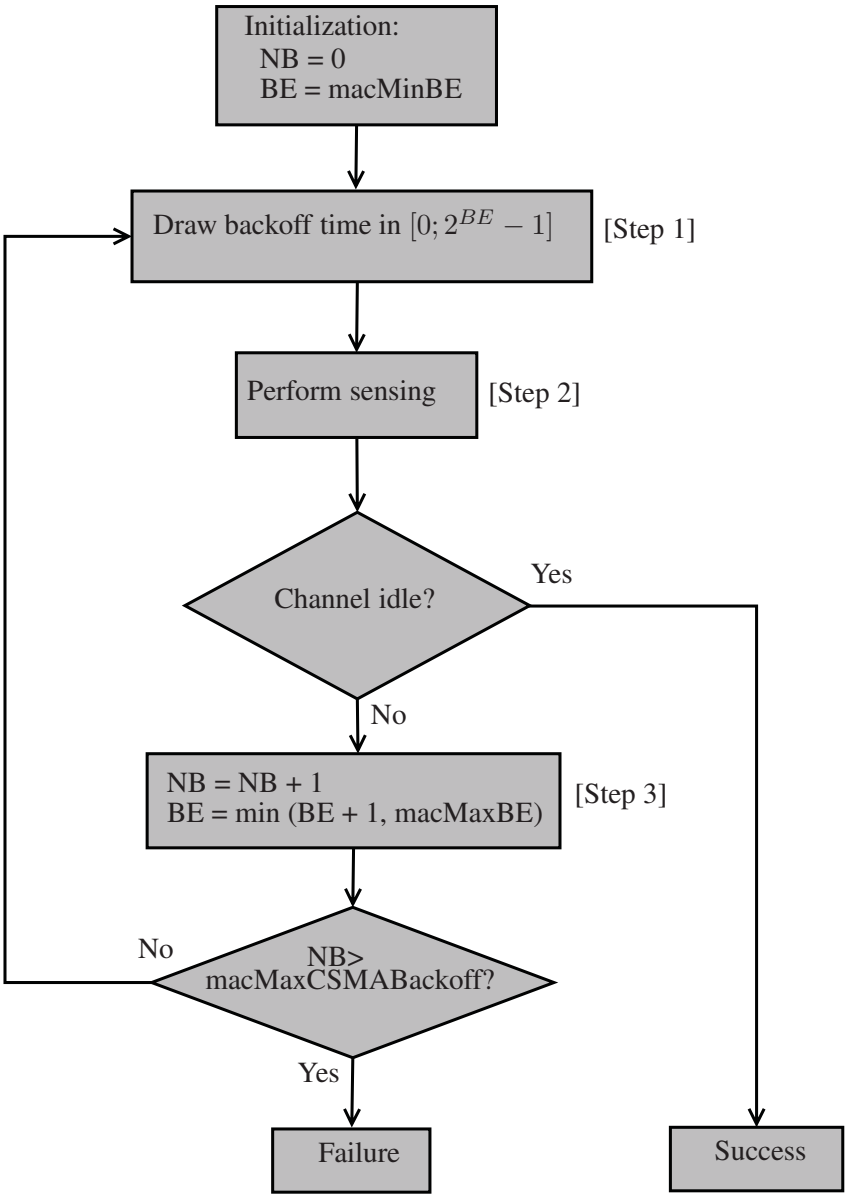


FIGURE A.1: Diagram of IEEE 802.15.4 unslotted CSMACA algorithm.

is the Contention Window which defines the number of consecutive backoff periods after which the medium must be sensed idle before start data transmission.

Figure A.2 presents a simplified version of the different steps of the slotted CSMA/CA algorithm. The first step is the initialization phase. The algorithm starts with $BE = macMinBE$, $NB = 0$ and $CW = 2$. Once these parameters have been initialized, step 2 consists in locating the border of the next backoff period and waiting for a whole number of backoff periods randomly fired in the interval $[0; 2^{BE} - 1]$. If the duration of the fired backoff is longer than the remaining backoff time in the CAP, the backoff time is consumed until the end of the CAP and the remaining time will be used in the next super-frame CAP.

At the end of this backoff time, the MAC layer verifies that the remaining backoff

time in the CAP is enough to perform 2 CCAs, transmit the physical frame and receive an acknowledgement. If this is the case, step 3 is applied. In this step, the MAC layer asks the physical layer to perform a CCA at the boundary of the next backoff period. If the time remaining in this super-frame is enough, the MAC layer carries the CCA back to the beginning of the next super-frame CAP. Starting with a backoff time helps to avoid systematic collisions due to multiple CCAs postponed by other nodes.

If the physical layer detects that the channel is free, the MAC layer decreases *CW* and tests whether *CW* is zero. If this is the case, the MAC layer will ask the physical layer to begin transmitting at the boundary of the next backoff period. If *CW* is not zero, the MAC layer asks the physical layer to do another CCA. The medium is not sensed all the times, but theOn en trouve pas actuellement sur amazon France two consecutive positive CCAs permit to conclude that the medium is free.

If the physical layer detects that the channel is busy, the MAC layer increments *NB* and *BE* by 1 (ensuring that *BE* remains lower or equal to *macMaxBE*) and returns CW to 2. If *NB* = *macMaxCSMABackoff*, the algorithm returns an access failure. It is then up to the MAC layer to request a retransmission if the *macMaxFrameRetries* is not exceeded. By default the *macMaxFrameRetries* is equal to three. If *NB* is smaller than *macMaxCSMABackoff*, the algorithm returns to step 2.

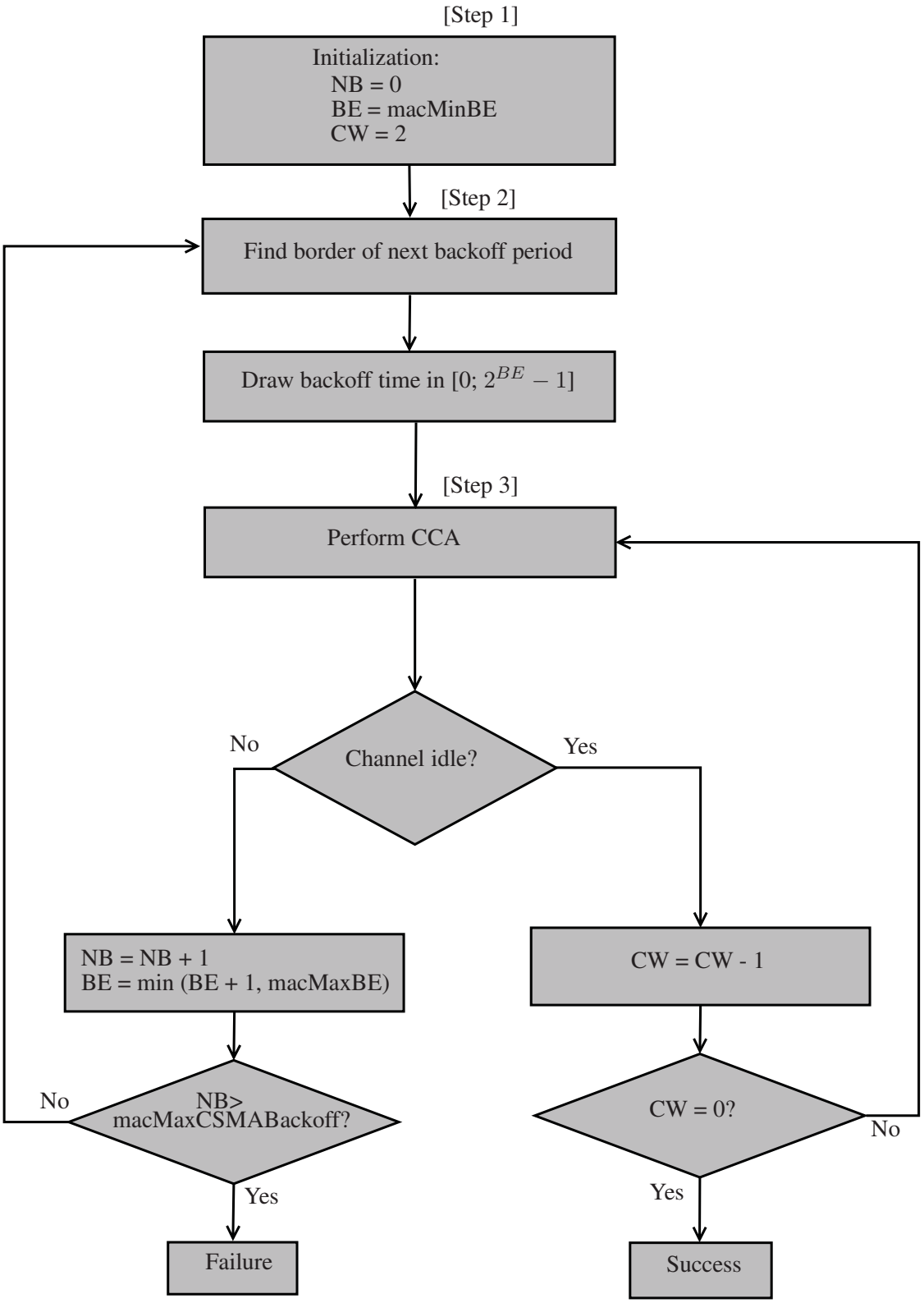


FIGURE A.2: Diagram of IEEE 802.15.4 slotted CSMACA algorithm.

Appendix B

Running experiment on TelosB motes using Contiki OS

In this appendix, we explain step by step how to simply and quickly use Contiki OS on command line to run an experiment or build a sensors network using TelosB motes. We assume that Contiki3.0 or newer version is already installed on the computer.

B.1 Handling before uploading application code on TelosB motes

B.1.1 First step: linking the mote to Contiki OS

We need to connect the mote to a USB port of the computer hosting Contiki OS via the USB port of the mote. A dialogue box is displayed by Contiki OS to signal that the TelosB mote is detected.

B.1.2 Second step: detecting the used serial port by the mote

When the mote is recognized by Contiki OS, we have to find on which serial port the mote will be accessible from command line. For that, we can proceed as follow:

In the sky directory in Contiki OS folders (by default, the path of sky motes directory is /home/user/contiki/example/sky), enter the following command:*make motelist*, and all motes that are currently linked to Contiki OS and the serial ports through which they are accessible will be displayed.

Below is an example of 4 motes connected to the system:

```
sky$ make motelist
../../tools/sky/motelisttelosB
Reference      Device      Description
MXVCYPJ6 /dev/ttyUSB0 XBOW Crossbow Telos Rev.B
XBSF8T0I /dev/ttyUSB1 XBOW Crossbow Telos Rev.B
XBSG1INR /dev/ttyUSB2 XBOW Crossbow Telos Rev.B
XBSF90M3 /dev/ttyUSB3 XBOW Crossbow Telos Rev.B
```

B.1.3 Third step: assign read and write rights to the serial port

Assigning read and write rights to the port connecting the mote to contiki OS permits to send some instructions to the mote throughout the OS. That can be done once for all connected motes in case we have several motes connected at the same time. For that, we execute the following command (still being in the directory /home/user/contiki/example/sky):

sky$sudo chmod 666 /dev/ttyUSB*
Or, do it one by one as follows:

sudo chmod 666 /dev/ttyUSB0,
sudo chmod 666 /dev/ttyUSB1,
sudo chmod 666 /dev/ttyUSB2,
sudo chmod 666 /dev/ttyUSB3

After executing this command, if no error message is displayed, then the motes are well connected and accessible in read/write mode. From now, we can give some instructions to them via command line.

B.2 Assigning an identifier to each mote before experiment

To facilitate the analysis of the results provided by the experiment, it is better to assign an identifier to each mote before uploading your program for experiment. To assign an identifier to mote, we can proceed as follow:

In the directory /home/user/contiki/examples/sky, flash the needed *ID* to the mote using the following command:

sky$ make clean && make burnnodeid.upload nodeid=1 nodemac=1 && make skyreset && make login (here, the *ID* is 1).

If several motes are connected, add the argument of the serial port interface in the command, as shown in the below example:

sky$ make clean && make burnnodeid.upload nodeid=1 nodemac=1 MOTES=/dev/ttyUSB0 && make skyreset MOTES=/dev/ttyUSB0 && make login MOTES=/dev/ttyUSB0

If the code is successfully flashed, the mote will display the following three messages:

Starting 'Burn node id'
Burning node id 1
Restored node id 1

We can repeat this process for each mote connected on the computer in order to have mote with identifiers like 1, 2, 3,...

B.3 Uploading Contiki OS with the protocol code on the mote

Now we need to go in Contiki OS directory containing the C code (e.g. /home/user/contiki/code/CoLBA in our case) that we want to upload on our motes.

Suppose that our program is called colba.c and is located in the directory /home/user/contiki/code/CoLBA. Once in this directory, we have to execute the following two commands:

colba$ make TARGET=sky savetarget
colba$ make colba.upload

NB: For the last above command, add the USB interface option if several motes are connected to the USB ports on the computer.
Example : colba$ make colba.upload MOTES=/dev/ttyUSB0
After executing these commands, if the flash was successful, the mote will display the following message via the command line:
using saved target 'sky'
msp430objcopy colba.sky O ihex colba.ihex
cp colba.ihex tmpimage.ihex
/dev/ttyUSB0
make skyreset skyupload
using saved target 'sky'
make[1]: Entering directory '/home/user/contiki/code/CoLBA'
make k j 20 skyresetsequence
using saved target 'sky'
make[2]: Entering directory '/home/user/contiki/code/CoLBA'
../../../tools/sky/skybslnopic z1 c /dev/ttyUSB0 r
MSP430 Bootstrap Loader Version: 1.39goodfet8
Use h for help
Use fromweb to upgrade a GoodFET.
Reset device ...
Done

make[2]: Leaving directory '/home/user/contiki/code/CoLBA'
make j 20 skyuploadsequence

After that our protocol code is running on the mote. We have to connect the sink to the computer, deploy the others motes and then save logs for analysis.

Below we present some of received logs that we used to evaluate the packet delivery ratio at sink node.

data message received, sender 11, pkt_Num 24, hops: 2
data message received, sender 15, pkt_Num 21, hops: 2
data message received, sender 6, pkt_Num 20, hops: 1
data message received, sender 2, pkt_Num 21, hops: 1
data message received, sender 20, pkt_Num 18, hops: 3
data message received, sender 12, pkt_Num 22, hops: 2
data message received, sender 3, pkt_Num 23, hops: 1
data message received, sender 7, pkt_Num 24, hops: 2
data message received, sender 8, pkt_Num 27, hops: 1
data message received, sender 9, pkt_Num 28, hops: 1
data message received, sender 5, pkt_Num 24, hops: 1

Bibliography

[1] G. Werner-Allen, P. Swieskowski, and M. Welsh, "Real-time volcanic earthquake localization", *In Proceedings of ACM SenSys*, pp. 357–358, 2006.

[2] L. Yang, M. Ji, Z. Gao, and T W. Zhang T. Guo, "Design of home automation system based on zigbee wireless sensor network", *In Proceedings of IEEE ICC Workshops*, pp. 2610–2613, 2009.

[3] S. Nourizadeh, C. Deroussent, and Y. Q. S. J. P. Thomesse, "Medical and home automation sensor networks for senior citizens telehomecare", *In Proceedings of IEEE ICISE*, pp. 1–5, 2009.

[4] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. BishopHurley, "Wireless ad hoc sensor and actuator networks on the farm", *In Proceedings of ACM IPSN*, pp. 492–499, 2006.

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", *In IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[6] ——, "Wireless sensor networks: a survey", *In Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[7] L. S. Committee, "Part 15.4: wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (lr-wpans)", *In IEEE Computer Society*, 2003.

[8] G. Chalhoub, E. P. de La Bathie, and M. Misson, "Overhead caused by wifi on zigbee networks using slotted csmaca", *In Journal of Networks*, vol. 11, no. 2, pp. 39–46, 2016.

[9] Z. Alliance, "Zigbee specification document 053474r17", 2008.

[10] ISA, "Isa100.11a:2009 wireless systems for industrial automation: process control and related applications", *International Society of Automation Std*, 2009.

[11] HART, "Hart field communication protocol specifications, revision 7.1", *HART Communication Foundation Std*, 2008.

[12] N. Zhu, F. Mieyeville, D. N. W. Du, and I. O'connor, "Research on high data rate wireless sensor networks", *In JNRDM*, P.61, 2011.

[13] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Priority assignment for real-time flows in wirelesshart networks", *In ECRTS*, pp. 35–44, 2011.

[14] P. Minet, G. Chalhoub, E. Livolant, M. Misson, B. Rmili, and J. F. Perelgritz, "Adaptive wireless sensor networks for aircraft", *In IEEE WiSEE*, pp. 1–6, 2015.

[15] I. F. Akyildiz, T. Melodia, and R. K. Chowdhury, "A survey on wireless multimedia sensor networks", *In omputer Networks*, vol. 51, no. 4, pp. 921–960, 2007.

[16] R. Diab, G. Chalhoub, and M. Misson, "Enhanced multi-channel mac protocol for multi-hop wireless sensor networks", *In IEEE Wireless Days*, pp. 1–6, 2014.

[17] ——, "Enhanced multi-channel mac protocol for multi-hop wireless sensor networks", *IEEE Wireless Days*, pp. 1–6, 2014.

[18] Y. Wu, J. A. Stankovic, T. He, and S. Lin, "Realistic and efficient multi-channel communications in wireless sensor networks", *IEEE INFOCOM*, pp. 1193–1201, 2008.

[19] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha, "Flow-based real-time communication in multi-channel wireless sensor networks. in european conference on wireless sensor networks", *Springer Berlin Heidelberg*, pp. 33–52, 2009.

[20] R. Diab, G. Chalhoub, and M. Misson, "Enhanced multi-channel mac protocol for multi-hop wireless sensor networks", *IEEE Wireless Days*, pp. 1–6, 2014.

[21] O. D. Incel, L. V. Hoesel, P. Jansen, and P. Havinga, "Mc-lmac: a multi-channel mac protocol for wireless sensor networks", *Ad Hoc Networks*, vol. 9, no. 1, pp. 73–94, 2011.

[22] G. Zhou, C. Huang, T. Yan, T. He, J. Stankovic, and T. Abdelzaher, "Mmsn:multi-frequency media access control for wireless sensor networks", *IEEE Infocom*, pp. 1–13, 2006.

[23] "Ieee standard for local and metropolitan area networkspart 15.4: low-rate wireless personal area networks (lr-wpans) amendment", 2014.

[24] P. Du and G. Roussos, "Adaptive time slotted channel hopping for wireless sensor networks", *IEEE CEEC*, pp. 29–34, 2012.

[25] Y. Baek, J. H. Lee, S. Lee, and H. S. Kim, "Predictive channel scanning and switching algorithm for the coexistence of ieee 802.15.4 and wifi", *IJICIC*, vol. 7, no. 9, pp. 2861–2872, 2013.

[26] A. Gupta, C. Gui, and P. Mohapatra, "Exploiting multi-channel clustering for power efficiency in sensor networks", *IEEE Comsware*, pp. 1–10, 2006.

[27] S. Lohier, A. Rachedi, I. Salhi, and E. Livolant, "Multichannel access for bandwidth improvement in ieee 802.15. 4 wireless sensor networks", *In IEEE Wireless Days*, pp. 1–6, 2011.

[28] O. Yilmaz, S. Demirci, Y. Kaymak, S. Ergun, and A. Yildirim, "Shortest hop multipath algorithm for wireless sensor networks", *In Computers and Mathematics with Applications*, vol. 63, no. 1, pp. 48–59, 2012.

[29] Y. Chen, P. H. Gomes, and B. Krishnamachari, "Multi-channel data collection for throughput maximization in wireless sensor networks", *IEEE MASS*, pp. 443–451, 2014.

[30] R. Diab, G. Chalhoub, and M. Misson, "Enhanced multi-channel mac protocol for multi-hop wireless sensor networks", *In IEEE IFIP WD*, pp. 1–6, 2014.

[31] A. Pal and A. Nasipuri, "Jrca: a joint routing and channel assignment scheme for wireless mesh networks", *In IEEE IPCCC*, pp. 1–8, 2011.

[32] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha, "Flow-based real-time communication in multi-channel wireless sensor networks. in european conference on wireless sensor networks", *Springer Berlin Heidelberg*, pp. 33–52, 2009.

[33] A. Pal and A. Nasipuri, "A distributed channel selection scheme for multi-channel wireless sensor networks", *In ACM MobiHoc*, pp. 263–264, 2012.

[34] ——, "Drcs: a distributed routing and channel selection scheme for multi-channel wireless sensor networks", *In IEEE PERCOM Workshops*, pp. 602–608, 2013.

[35] C. Y. Wan, S. B. Eisenman, and A. T. Campbell, "Coda: congestion detection and avoidance in sensor networks", *In ACM 1st international conference on Embedded networked sensor systems*, pp. 266–279, 2003.

[36] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz, "Esrt: event-to-sink reliable transport in wireless sensor networks", *In ACM MobiHoc*, pp. 177–188, 2003.

[37] L. Galluccio, A. T. Campbell, and S. Palazzo, "Concert: aggregation-based congestion control for sensor networks", *In Proceedings of ACM 3rd international conference on Embedded networked sensor systems*, pp. 274–275, 2005.

[38] K. W. Fan, S. Liu, and P. Sinha, "Scalable data aggregation for dynamic events in sensor networks", *In Proceedings of ACM 4th international conference on Embedded networked sensor systems*, pp. 181–194, 2006.

[39] O. Chughtai, N. Badruddin, A. Awang, and M. Rehan, "Reliability-constrained routing for traffic load balancing in wireless sensor networks", *In IEEE ICIAS*, pp. 1–6, 2014.

[40] F. Ren, T. He, S. K. Das, and C. Lin, "Traffic-aware dynamic routing to alleviate congestion in wireless sensor networks", *In IEEE TPDS*, pp. 1585–1599, 2011.

[41] O. Banimelhem and S. Khasawneh, "Gmcar: grid-based multipath with congestion avoidance routing protocol in wireless sensor networks", *In Ad Hoc Networks*, pp. 1346–1361, 2012.

[42] X. Wang and M. Tan, "A load-balancing routing algorithm for multi-channel wireless mesh networks", *International Journal of Sensor Networks*, pp. 249–255, 2015.

[43] M. Song, Y. Zhao, J. Wang, and E. K. Park, "A high throughput load balance algorithm for multichannel wireless sensor networks", *In IEEE International Conference on Communications*, pp. 1–5, 2009.

[44] D. Kandris, D. J. Vergados, D. D. Vergados, and A. Tzes, "A routing scheme for congestion avoidance in wireless sensor networks", *In IEEE CASE*, 2010.

[45] T. Winter, P. Thubert, A. Brandt, T. Clausen, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, and J. P. Vasseur, "Rpl: ipv6 routing protocol for low power and lossy networks", *IETF Standard*, 2012.

[46] W. W. Fang, J. M. Chen, L. Shu, T. S. Chu, and D. P. Qian, "Congestion avoidance, detection and alleviation in wireless sensor networks", *In Journal of Zhejiang University-Science C*, pp. 63–73, 2010.

[47] J. Y. Teo, Y. Ha, and C. K. Tham, "Interference-minimized multipath routing with congestion control in wireless sensor network for high-rate streaming", *In IEEE Transactions on Mobile Computing*, vol. 9, no. 7, pp. 1124–1137, 2008.

[48] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (aodv) routing", *RFC No. 3561*, 2003.

[49] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha, "Flow-based real-time communication in multichannel wireless sensor networks", *In EWSN*, vol. 9, pp. 33–52, 2009.

[50] A. Pal and A. Nasipuri, "Drcs: a distributed routing and channel selection scheme for multi-channel wireless sensor networks", *In IEEE PERCOM Workshops*, pp. 602–608, 2013.

[51] N. Zhu, F. Mieyeville, D. Navarro, W. Du, and I. O'connor, "Research on high data rate wireless sensor networks", *In 14eme Journees Nationales du Reseau Doctoral de Micro et Nanoelectronique*, p. 61, 2011.

[52] J. Henaut, A. Lecointre, D. Dragomirescu, and R. Plana, "Radio interface for high data rate wireless sensor networks", *In arXiv preprint arXiv:1004.0204*, 2010.

[53] O. Yilmaz, S. Demirci, Y. Kaymak, S. Ergun, and A. Yildirim, "Shortest hop multipath algorithm for wireless sensor networks", *In Computers and Mathematics with Applications*, 2012.

[54] D. S. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing", *In Wirel. Netw*, vol. 11, no. 4, 2005.

[55] Y. Chen, P. H. Gomes, and B. Krishnamachari, "Multi-channel data collection for throughput maximization in wireless sensor networks", *In Proceedings of the IEEE MASS*, pp. 443–451, 2014.

[56] R. Diab, G. Chalhoub, and M. Misson, "Enhanced multi-channel mac protocol for multi-hop wireless sensor networks", *In IEEE Wireless Days*, pp. 1–6, 2014.

[57] R. Diab, "Hmc-mac: un protocole mac hybride et multi-canal pour les réseaux de capteurs sans fil", *Doctoral dissertation, Université Blaise Pascal-Clermont-Ferrand II*, 2015.

[58] R. Diab, G. Chalhoub, and M. Misson, "Hybrid multi-channel mac protocol for wireless sensor networks: interference rate evaluation", *In IEEE VTC Fall*, pp. 1–6, 2013.

[59] T. Qiu, L. Feng, F. Xia, G. Wu, and Y. Zhou, "A packet buffer evaluation method exploiting queueing theory for wireless sensor networks", *In Computer Science and Information Systems*, vol. 8, no. 4, pp. 1028–1049, 2011.

[60] B. V. Patel and C. C. Bisdikian, "End-station performance under leaky bucket traffic shaping", *In IEEE Network*, vol. 10, no. 5, pp. 40–47, 1996.

[61] A. Dunkels, B. gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors", *In Proceedings of the IEEE LCN*, 455–462, 2004.

[62] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja", *In Proceedings of the IEEE LCN*, 641–648, 2006.

[63] H. Tall, G. Chalhoub, and M. Misson, "Implementation and performance evaluation of ieee 802.15. 4 unslotted csma/ca protocol on contiki os", *In Annals of Telecommunications*, vol. 71, no. 9-10, pp. 517–526, 2016.

[64] T. S. Rappaport, "Wireless communications principles and practice edition", *Prentice Hall*, 2001.

[65] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: a self-regulating algorithm for code maintenance and propagation in wireless sensor networks", *In Proceedings of the USENIX NSDI*, pp. 15–28, 2004.

[66] F. J. Estevez, J. M. Castillo-Secilla, J. Gonzalez, J. Olivares, and P. Glosekotter, "Mdaral: a multi-radio version for the daral routing algorithm", *In Sensors*, vol. 17, no. 2, p. 324, 2017.

[67] M. Singh, M. Sethi, N. Lal, and S. Poonia, "A tree based routing protocol for mobile sensor networks (msns)", *In International Journal on Computer Science and Engineering*, vol. 2(1S), 2010.