EMAN SAAD ALI AL-HAWRI

# NETWORK CODING FOR RELIABLE WIRELESS SENSOR NETWORKS


2019

Eman Saad Ali AL-Hawri

# Network Coding for Reliable Wireless Sensor Networks

PhD Thesis in Computer Science

Work done under the supervision of:

Prof. Dra. Noélia Correia

Prof. Dr. Alvaro Barradas


2019

# Statement of Originality

**Network Coding for Reliable Wireless Sensor Networks**

**Statement of authorship**: The work presented in this thesis is, to the best of my knowledge and belief, original, except as acknowledged in the text. The material has not been submitted, either in whole or in part, for a degree at this or any other university.

**Candidate**:

(Eman AL-Hawri)

# ABSTRACT

Wireless sensor networks are used in many applications and are now a key element in the increasingly growing Internet of Things. These networks are composed of small nodes including wireless communication modules, and in most of the cases are able to autonomously configure themselves into networks, to ensure sensed data delivery. As more and more sensor nodes and networks join the Internet of Things, collaboration between geographically distributed systems are expected. Peer to peer overlay networks can assist in the federation of these systems, for them to collaborate. Since participating peers/proxies contribute to storage and processing, there is no burden on specific servers and bandwidth bottlenecks are avoided.

Network coding can be used to improve the performance of wireless sensor networks. The idea is for data from multiple links to be combined at intermediate encoding nodes, before further transmission. This technique proved to have a lot of potential in a wide range of applications. In the particular case of sensor networks, network coding based protocols and algorithms try to achieve a balance between low packet error rate and energy consumption. For network coding based constrained networks to be federated using peer to peer overlays, it is necessary to enable the storage of encoding vectors and coded data by such distributed storage systems. Packets can arrive to the overlay through any gateway/proxy (peers in the overlay), and lost packets can be recovered by the overlay (or client) using original and coded data that has been stored. The decoding process requires a decoding service at the overlay network. Such architecture, which is the focus of this thesis, will allow constrained networks to reduce packet error rate in an energy efficient way, while benefiting

from an effective distributed storage solution for their federation. This will serve as a basis for the proposal of mathematical models and algorithms that determine the most effective routing trees, for packet forwarding toward sink/gateway nodes, and best amount and placement of encoding nodes.

Keywords: Wireless sensor networks, Network coding, RELOAD, CoAP.

# RESUMO

As redes de sensores sem fios são usadas em muitas aplicações e são hoje consideradas um elemento-chave para o desenvolvimento da Internet das Coisas. Compostas por nós de pequena dimensão que incorporam módulos de comunicação sem fios, grande parte destas redes possuem a capacidade de se configurarem de forma autónoma, formando sistemas em rede para garantir a entrega dos dados recolhidos. À medida que mais e mais nós integram estes sistemas e estes se juntam à Internet das Coisas, também é mais expectável que estes sistemas colaborem, ainda que geograficamente distribuídos. As redes *peer to peer* podem ajudar na federação desses sistemas, facilitando essa colaboração. Como nestas redes os *peers/proxies* participantes contribuem quer para o armazenamento quer para o processamento, nenhum servidor específico é sobrecarregado. Além disso, são também evitados estrangulamentos de largura de banda.

O *network coding* pode ser usado para melhorar o desempenho das redes de sensores sem fios. A ideia subjacente a esta técnica é combinar dados de vários *links*, em nós de codificação, antes de efetuar a sua transmissão. Trata-se de uma técnica que provou ter bastante potencial numa ampla gama de aplicações. Também assim é nas redes de sensores, onde os protocolos e algoritmos baseados em *network coding* procuram alcançar um equilíbrio entre baixas taxas de erro nos pacotes e o consumo de energia. Para que estas redes, utilizando *network coding*, sejam federadas usando abordagens *peer to peer*, é necessário assegurar que possam armazenar vetores de codificação e dados codificados, apesar das limitações de capacidade dos seus nós.

Nesta arquitectura os pacotes podem chegar à rede *peer to peer* através de qualquer uma das *gateways/proxies* (*peers* na rede), e os pacotes perdidos podem ser recuperados pela própria rede *peer to peer* (ou cliente) usando os dados originais e codificados que estão armazenados. O processo de descodificação irá exigir a implementação de um serviço de descodificação na rede *peer to peer*. Esta arquitetura, que é o foco desta tese, permitirá que as redes reduzam a taxa de erro de pacotes de maneira eficiente em termos energéticos, enquanto se beneficia de uma solução de armazenamento distribuída e eficaz para a sua federação. Estes aspectos servirão de base para a proposta de modelos matemáticos e algoritmos que determinam não só as árvores de roteamento mais eficazes para encaminhamento de pacotes em direção a nós *gateway*, mas também a quantidade e o posicionamento mais adequado dos nós de codificação.

Palavras-chave: Redes de sensores sem fios, *Network coding*, RELOAD, CoAP.

# ACKNOWLEDGMENT

First, I would like to say thank you very much to my advisers Prof. Dra. Noélia Correia and Prof. Dr. Alvaro Barradas for their support, advices, and patience through my PhD journey.

I would like to acknowledge my home university, Thamar University-Yemen through my doctoral grant (No. 209).

Also, many thanks go to CEOT (Center for Electronic, Optoelectronic and Telecommunications) for partial support of this work within UID/MULTI/00631/2019 project funded by FCT.

My sincere thanks also go to my husband and my son 'Yoyo'. I love you both so much. This work would not have been possible without your love and support. Thank you for being in my life.

Many thanks to my best friend 'Fayza' for her support and kindness. Thank you so much for taking care of my son every time I needed. Thank you so much for your supportive words every time I was disappointed and tired.

Finally, I would like to thank my parents, my sisters, and my brothers for their love and support during this work and throughout my life as well.

# CONTENTS

**Abstract** iii

**Resumo** v

**Acknowledgment** vii

**List of Figures** xiii

**List of Tables** xv

**Nomenclature** xvii

**1 Introduction** 1
- 1.1 Motivation and Scope . . . 1
  - 1.1.1 Network Coding . . . 2
  - 1.1.2 P2P Overlay Networks . . . 3
- 1.2 Contributions . . . 4
- 1.3 Thesis Outline . . . 6

**2 Wireless Sensor Networks** 7
- 2.1 Sensor Architecture . . . 10
- 2.2 Challenges and Constraints . . . 11
  - 2.2.1 Energy . . . 12
  - 2.2.2 Self Management . . . 12
  - 2.2.3 Security . . . 13
  - 2.2.4 Environment . . . 13

2.2.5 Power Consumption . . . 14
2.2.6 Scalability . . . 14
2.2.7 Hardware Cost Constraints . . . 15
2.3 Wireless Sensor Network Applications . . . 15
2.4 Protocol Stack . . . 16
2.4.1 Physical Layer . . . 17
2.4.2 Data Link Layer . . . 17
2.4.3 Network Layer . . . 18
2.4.4 Transport Layer . . . 18
2.4.5 Application Layer . . . 19
2.5 Wireless Sensor Network Topologies . . . 19
2.6 QoS in Wireless Sensor Networks . . . 20
2.7 Routing Protocols in WSNs . . . 22
2.7.1 Cluster based Routing Protocols . . . 22
2.7.2 Chain based Routing Protocols . . . 24
2.7.3 Tree based Routing Protocols . . . 25
2.8 Summary . . . 28

**3 Network Coding** **29**
3.1 Network Coding Benefits . . . 31
3.1.1 Throughput . . . 31
3.1.2 Robustness . . . 32
3.1.3 Complexity . . . 34
3.1.4 Security . . . 34
3.2 Encoding Schemes . . . 35
3.2.1 Binary Network Coding . . . 35
3.2.2 Random Linear Network Coding (RLNC) . . . 35
3.3 Encoding/Decoding Process . . . 36
3.4 Related Work . . . 38
3.5 Summary . . . 48

**4 Federation of Wireless Sensor Networks** **49**
4.1 Federation Related Standards . . . 52
4.1.1 Constrained Application Protocol . . . 52
4.1.2 RELOAD . . . 53
4.2 Related Work . . . 54
4.3 Proposed P2P Architecture . . . 61
4.3.1 Overall Architecture . . . 61
4.3.2 Storing Data and Standard Extensions Needed . . . 63
4.3.2.1 Data Kinds . . . 63
4.3.2.2 Requirements Regarding Network Coding . . . 64
4.3.2.3 Storing and Fetching of Data . . . 65
4.3.2.4 CoAP Option . . . 67
4.4 Summary . . . 67

**5 Network Coding for Reliable WSNs** **69**
5.1 Related Work . . . 71
5.2 Architecture . . . 74
5.2.1 Network Coding . . . 74
5.2.2 RELOAD Overlay . . . 75
5.3 Encoding Node Location Problem . . . 77
5.3.1 Problem Definition . . . 77
5.3.2 Mathematical Formulation . . . 77
5.3.3 Hardness of the Problem . . . 82
5.3.4 Upper Bound and Heuristic Algorithm . . . 83
5.3.4.1 Upper Bound . . . 83
5.3.4.2 Algorithm . . . 84
5.4 Performance Evaluation . . . 85
5.4.1 Scenario Setup . . . 85
5.4.2 Analysis of Results . . . 89
5.4.2.1 Number of Encoding Nodes: Small Topology Tests . 90
5.4.2.2 Number of Encoding Nodes: Big Topology Tests . . 92
5.4.2.3 SenseCode vs Proposed Approach . . . 94
5.5 Summary . . . 96

**6 DAG-Coder for Reliable WSNs** **97**
6.1 Related Work . . . 100
6.1.1 General Data Dissemination Related Work . . . 101
6.1.2 Clustering Related Work . . . 103
6.2 The Data Dissemination Problem . . . 104
6.2.1 Motivation and Architecture . . . 104
6.2.2 Problem Definition . . . 107
6.3 Mathematical Model . . . 107
6.3.1 Notation and Assumptions . . . 107
6.3.2 Formalization . . . 108
6.4 Performance Analysis . . . 112
6.4.1 Scenario Setup . . . 112
6.4.2 Performance Metrics . . . 115
6.4.3 Analysis of Results . . . 115
6.4.3.1 Reliability . . . 115
6.4.3.2 Packet Transmissions . . . 118
6.5 Discussion . . . 121
6.6 Summary . . . 121

**7 Conclusions and Future Work** **123**

**Bibliography** **127**

# LIST OF FIGURES


2.1 Different types of sensors [Ele08]. . . . . 8
2.2 Wireless sensor networks and their communication. . . . . 9
2.3 Sensor node structure. . . . . 11
2.4 Wireless sensor network protocol stack. . . . . 16

3.1 Traditional routing in wired multicast network. . . . . 30
3.2 Binary coding in wired multicast network. . . . . 30
3.3 Traditional transmission in wireless network. . . . . 32
3.4 NC based transmission in wireless network. . . . . 32
3.5 NC robustness in multicast network. . . . . 34
3.6 Random linear coding in multicast network. . . . . 36
3.7 Dynamic network coding with nodes organized as peers in P2P network [dALFMA18]. . . . . 39
3.8 Network coding model from [GCSS14]. . . . . 43
3.9 Pseudo-broadcast in Cope method, applied at intermediate nodes [KRH+08]. . . . . 47

4.1 IoT web stack vs internet web stack. . . . . 52
4.2 CoAP sub layers [CKP15]. . . . . 53
4.3 Gateway main blocks [CSDC11]. . . . . 55
4.4 CoAP and RELOAD architecture for WSNs federation [MBL12]. . . . 57
4.5 P2P based wireless sensor network [AL07]. . . . . 58
4.6 CoHaRT protocol stack [SDA+15]. . . . . 58
4.7 Main components and their interactions [IHVdA+14]. . . . . 59
4.8 Entity manager components [IHVdA+14]. . . . . 60
4.9 Proposed P2P architecture. . . . . 62
4.10 Example of network coding at constrained network. . . . . 66
4.11 Decoding at RELOAD/CoAP P2P overlay network. . . . . 66

5.1 RELOAD/CoAP based architecture. . . . . 76

5.2 Sparse network topologies (ENL Problem). For illustration, maroon links highlight a particular tree (from blue wireless links) while green nodes highlight a particular set of gateways. . . . . 87
5.3 Dense network topologies (ENL Problem). For illustration, maroon links highlight a particular tree (from blue wireless links) while green nodes highlight a particular set of gateways. . . . . 88
5.6 Total number of packets at gateways (original and coded) for 20-node networks (ENL Problem). . . . . 94
5.7 Total number of packets at gateways (original and coded) for 40-node networks (ENL Problem). . . . . 95

6.1 Routing tree in SenseCode [KAAF13]. . . . . 101
6.2 DAG with single sink. . . . . 105
6.3 DAG with cluster heads. Such nodes perform encoding of data packets from associated sources. . . . . 106
6.4 Network topologies (DNC-WSN problem). Maroon nodes highlight a particular set of gateways while weights on links are the relative distances between nodes. . . . . 113
6.5 Reliability for dense topologies. . . . . 116
6.6 Reliability for sparse topologies. . . . . 117
6.7 Total packets for dense topologies. . . . . 119
6.8 Total packets for sparse topologies. . . . . 120

# LIST OF TABLES

5.1 Topologies used in Performance Analysis (ENL Problem). . . . . 86
5.2 Example of Failure Scenarios (ENL Problem). . . . . 89

6.1 Packets Overheard in SenseCode for Scenario in Figure 6.1. . . . . 102

# NOMENCLATURE

**Abbreviations**

| | |
|---|---|
| 6LoWPAN | IPv6 over Low-power Wireless Personal Area Networks |
| ACK | ACKnowledgment |
| AdapCode | Adaptive network Coding |
| ADC | Analog to Digital Converter |
| AMRP | Average Minimum Reachability Power |
| ARQ | Automatic Repeat reQuest |
| CH | Cluster Head |
| CluDDA | Clustered diffusion with Dynamic Data Aggregation |
| CNCNS | Centrality based Network Coding Node Selection |
| CoAP | Constrained Application Protocol |
| CON | CONfirmable |

| | |
|---|---|
| CoRE | Constrained RESTful Environments |
| DAG | Directed Acyclic Graph |
| DNC-WSN | DAG based Network Coding for WSN |
| EADAT | Energy-Aware Data Aggregation Tree |
| ENL | Encoding Node Location |
| FEC | Forward Error Correction |
| GN | Gateway Node |
| GPS | Global Positioning System |
| HEED | Hybrid Energy Efficient Distributed clustering |
| HTTP | HyperText Transfer Protocol |
| IETF | Internet Engineering Task Force |
| INCOR | Inter-flow Network Coding based Opportunistic Routing |
| IoT | Internet of Things |
| LEACH | Low-Energy Adaptive Clustering Hierarchy |
| LN | Local Node |
| MAC | Medium Access Control |
| MCN | Monitoring and Controlling Node |
| mt | measurement time |
| NACK | Negative ACKnowledgment |
| NAT | Network Address Translation |

| | |
|---|---|
| NC | Network Coding |
| NON | NON confirmable |
| NP-hard | Non-deterministic Polynomial-time hardness |
| P2P | Peer-to-Peer |
| PEDAP | Power Efficient Data gathering and Aggregation Protocol |
| PEGASIS | Power Efficient GAthering in Sensor Information Systems |
| PN | Proxy Node |
| QoS | Quality of Service |
| ReDiR | Recursive Distributed Rendezvous |
| RELOAD | REsource LOcation And Discovery |
| REST | REpresentational State Transfer |
| RLNC | Random Linear Network Coding |
| RST | ReSeT message |
| SIP | Session Initiation Protocol |
| SMP | Sensor Management Protocol |
| SQDDP | Sensor Query and Data Dissemination Protocol |
| TCP | Transport Control Protocol |
| TMDAP | Task Management and Data Advertisement Protocol |
| ttl | time-to-live |
| UDP | User Datagram Protocol |

| | |
|---|---|
| URI | Uniform Resource Identifier |
| WN | Wide area Node |
| WSN | Wireless Sensor Network |
| XOR | Exclusive OR |

**Sets**

| | |
|---|---|
| $\mathcal{F}$ | Set of failure scenarios. |
| $\mathcal{G}$ | Set of gateways. |
| $\mathcal{L}$ | Set of wireless links. |
| $\mathcal{L}^*$ | Set of best path tree links. |
| $\mathcal{L}^{\mathcal{F}}$ | Union of all $\mathcal{L}^f$, where $f \in \mathcal{F}$. |
| $\mathcal{L}^f$ | Set of failed links in failure scenario $f \in \mathcal{F}$. |
| $\mathcal{N}$ | Set of nodes. |
| $\mathcal{N}^{\mathrm{e}}$ | Set of encoding nodes. |
| $\mathcal{P}_l$ | Set of alternative paths, where overhearing/encoding nodes would be applied, associated with failing link $l \in \mathcal{L}^{\mathcal{F}}$. |
| $\mathcal{T}^{\mathrm{U}}$ | A universe set of feasible path tree partitions of nodes. |
| $\mathcal{T}^i$ | A specific feasible path tree partition of nodes, including a tree for each gateway/root, from $\mathcal{T}^{\mathrm{U}}$. |
| $\mathbf{p}^{\min}$ | Set of the shortest paths. |

**Variables**

| | |
|---|---|
| $\beta_s^g$ | One if $g \in \mathcal{G}$ is the gateway for source $s \in \mathcal{N} \backslash \mathcal{G}$; zero otherwise. |

$\beta_n$ One if wireless node $n \in \mathcal{N} \backslash \mathcal{G}$ is selected to become a CH, participating in the DAG, zero otherwise.

$\delta_l^s$ One if source node $s \in \mathcal{N} \backslash \mathcal{G}$ uses link $l \in \mathcal{L}$ to send its data towards a gateway; zero otherwise.

$\eta_{l,f}^n$ One if node $n \in \mathcal{N}$ is the last/destination node of the protection path for link $l \in \mathcal{L}^f$; zero otherwise.

$\gamma_l$ One if link $l \in \mathcal{L}$ is used by a path tree; zero otherwise.

$\mathcal{T}^*$ The best path tree partition of nodes.

$\sigma_{l,f}^{l'}$ One if link $l' \in \mathcal{L}$ is used for protection of link $l \in \mathcal{L}^f$; zero otherwise.

$\sigma_l$ One if link $l \in \mathcal{L}$ is to participate in the DAG (used for data delivery), zero otherwise.

$\tau_n$ Topological order of $n \in \mathcal{N} \backslash \mathcal{G}$ in the DAG.

$\vartheta^{n_i}$ One if node $n_i \in \mathcal{N}^{\mathrm{e}}$ is chosen to act as encoding node; zero otherwise.

$\xi_n$ Auxiliar variable to avoid having a fixed DAG outdegree at node $n \in \mathcal{N} \backslash \mathcal{G}$.

$S^*$ The optimal solution.

$S^{\mathrm{UB}}$ Upper bound on the optimal solution.

**Constants**

$\Delta$ A big value which can be set to $\Delta = \mid \mathcal{N} \mid \times \mid \mathcal{L} \mid$.

$H$ Number of hops.

$Hops(n, g)$ The number of hops from node $n \in \mathcal{N}$ towards $g \in \mathcal{G}$.

$w(l)$ The weight of link $l \in \mathcal{L}$.

# 1

# INTRODUCTION

## 1.1 Motivation and Scope

NOWADAYS we are living in two parallel worlds, our physical world and a digital world mainly based on the internet. With this revolution in information technology, our perspective on physical objects has changed. Physical objects are starting to be operated by smart systems, becoming able to communicate with one another, which reduces human intervention for their operation. Smart systems include sensing, actuation and control functions in order to react to the environment, to perform any required analysis and to make adequate decisions. Such systems rely mainly on sensor information sources, meaning that sensors are seen by many as the main part of these systems [JSHG15].

Wireless Sensor Networks (WSNs) rely on concepts that are similar to smart objects [VD10]. WSNs are composed of small nodes, including wireless communication modules, and in most of the cases are able to autonomously configure themselves into networks, to ensure sensing data delivery. Smart objects, however, are more intended to perform tasks, like actuation and control, and are less focused on pure data gathering, while WSNs are primarily focused on data delivery using wireless radio communication modules. Smart objects, on the contrary, are not tied to any particular communication system. WSNs are now used in many applications and are a key element in the increasingly growing Internet of Things (IoT).

### 1.1.1 Network Coding

The performance of WSNs can be affected by channel bandwidth limitation, unstable signal transmission, power constraints, or other network/node characteristics [ZAL+09]. Different approaches have been proposed over the last years to handle one or many of these issues. One of these techniques is Network Coding (NC) where the idea is for information/data from multiple links to be combined at intermediate encoding nodes, before further transmission. This can help improving network throughput and it has demonstrated to have potential in a wide range of applications.

Although NC can be applied to wired and Wireless Mesh Networks (WMNs), many developments in these two kinds of networks cannot be applied to sensor networks. More specifically, NC is used mainly for throughput increase, or efficient use of bandwidth, in one-to-many traffic flow scenarios. Sensor networks, contrarily to WMNs, have the following characteristics [KAAF13, DP10]:

- There are multiple sources sending data notifications to a gateway/sink node, meaning that the traffic flow is mainly many-to-one;
- Energy efficiency is the main concern rather than bandwidth, since nodes typically produce small volumes of data and are most of the times energy constrained;
- The cost of transceivers is a concern in large-scale deployments;
- In tree based many-to-one approaches, widely used in WSNs, messages may not reach the sink although alternative viable paths exist (trees are built based on past network conditions, and alternative paths may not have been discovered yet).

For all these reasons, bandwidth efficiency is often sacrificed to achieve power and/or cost efficiency [DP10]. Since packet error rate is also a critical issue, an adequate criteria when designing NC based protocols and algorithms for WSNs is to achieve a balance between low packet error rate and energy/cost efficiency, which are competing goals (e.g., a packet may be sent through multiple paths to reduce packet error rate but this increases energy consumption and cost). Packet error rate is defined as the fraction of messages generated by the sources that are not successfully communicated to the destination, capturing the ability of the protocols to deliver the original data in the face of packet loss.

### 1.1.2 P2P Overlay Networks

Peer-to-Peer (P2P) systems have proliferated over the last years, and are now the most popular systems for content distribution [FL12]. These systems build an overlay network (on top of an existing network, like the internet) where participating

peers can find the other peers using their logical identifiers. The advantages of these systems are the scalability and not having a single point of failure. This is because files (to be shared) are broken into small pieces and sent to peers, which are allowed to share them too. Since each participating peer shares its own upload bandwidth, the time to distribute/download data reduces significantly. That is, a huge amount of bandwidth (aggregated bandwidth) is available for the overall system [FL12].

From constrained networks' perspective, P2P systems can be used to store and disseminate data. All kinds of smart systems are expected to participate in the IoT, and interactions between these systems are expected too. This means that applications relying on data sharing (for collaboration) will increase, and effective distributed data storage solutions become necessary. This need led to the proposal of a RESource Location And Discovery (RELOAD) Usage for the Constrained Application Protocol (CoAP) [JLVMC15]. RELOAD is a generic P2P protocol that accepts pluggable application layers (Usages), which allows it to fit several applications. CoAP is a web transfer protocol for use with constrained nodes and constrained networks, which is expected to be widely used. RELOAD/CoAP will allow building P2P overlay networks, where constrained systems store their data and clients are able to retrieve it. This can be seen as a distributed caching system.

Since many smart systems rely on WSNs to communicate, where NC can be used to reduce packet error rate in a energy efficient way, RELOAD/CoAP overlays should be prepared to store data from NC based networks. This is the focus on this PhD dissertation.

## 1.2 Contributions

The contributions included in this thesis are the following:

- A CoAP Usage extension is proposed so that NC based constrained networks can use RELOAD/CoAP overlays for data storage, through their proxies/gateways participating as peers. More specifically, NC based constrained networks should be able to store encoding vectors and coded data, for further recovery of lost packets, if necessary. The decoding/recovery can be done by the overlay itself or by the clients. For this to be possible, the CoAP Usage data Kind must be extended. Such extension will allow a scalable and efficient way of discovering cached sensor data, of geographically dispersed sensor networks, allowing large scale applications to emerge, while taking advantage of NC at the wireless section. This is detailed in Chapter 4.

- The efficient design of NC based reliable sensor networks is addressed. More specifically, given a network scenario with certain critical communication channels/links (failure scenarios), ways to plan for the adequate amount and placement of encoding nodes are proposed, so that all data is received at the overlay (even in case of packet loss at critical links). Packets can arrive to the overlay through any of the existing gateways (peers in the overlay), and lost packets can be recovered by the overlay (or client) using both stored data and coded packets. Each failure scenario includes one or more bad quality links that may go down simultaneously. As far as known, the reliability plus network coding node placement problem in many-to-one sensor networks was not addressed by previous authors. This is detailed in Chapter 5.

- A Directed Acyclic Graph (DAG) based dissemination approach, where clustering and NC techniques are applied, called DAG-Coder, is proposed. The main goal is to improve the network reliability while avoiding pre-defined failure scenarios, allowing it (DAG-Coder) to be used in dynamic environments. This approach is detailed in Chapter 6.

## 1.3 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 provides an introduction to WSNs. The societal benefits of sensor networks, together with their types, constraints, and protocols are discussed. Chapter 3 is dedicated to NC principles. The advantages of using NC are discussed, and related research work is presented. In Chapter 4, P2P overlay networks, and required protocols to operate this network, are described. The chapter also discusses a proposal for WSNs to get benefit from P2P overlays and NC simultaneously. In Chapter 5, a proposal for the best placement of encoding nodes, while ensuring reliability in WSNs, is presented. This takes into account failure scenarios and assumes a P2P overlay for data storage. A mathematical model and a heuristic algorithm are developed to achieve such objective, considering either a single or multiple gateways. Work related with this contribution is also presented. In Chapter 6, a DAG based dissemination approach, where clustering and NC techniques are applied, is proposed. The main goal of this contribution is to improve the network reliability, while avoiding failure scenarios to be defined. The chapter also presents related work. Chapter 7 summarizes conclusions and discusses future work.

# 2

# WIRELESS SENSOR NETWORKS

THE recent increase in network requirements affects not only the evolution of network technologies, but also our view of networks in general. Networking has steadily evolved from the simple concept of two or more users using machines for data exchange, to fully machine-machine communication. Therefore, modern networking environments include both user and machine-centric systems.

WSNs play a major role in the previously mentioned communication ecosystem. Depending on the type of sensors (infrared, acoustic, biological, and so on), these networks can be targeted to a specific purpose, like security improvement, productivity increase, wildfire detection, health monitoring, traffic regulation, and other new applications like smart homes and others [ASSC02]. Figure 2.1 shows some sensor types that can be used in different applications.


Figure 2.1: Different types of sensors [Ele08].

A WSN is a network of spatially distributed sensing devices, usually of low cost, that may have self-organizing ability, or not, and may vary in their features and capabilities. These are able to sense their surrounding, communicate with their neighbors, and send their observations to a sink/gateway where the observed data is analyzed, processed and sent to the end user. In other words, sensing devices convert real world parameters and events into signals or data that can be processed and analyzed [DP10]. In WSNs, groups of sensing nodes can be seen as aggregates where members collaborate with each other to provide a service. Hence, each group of devices can be used as an independent data collection entity, while the WSN can be seen as a distributed database in the sense that nodes with the requested data will reply to queries sent by the sink node, usually acting as a intermediate/proxy

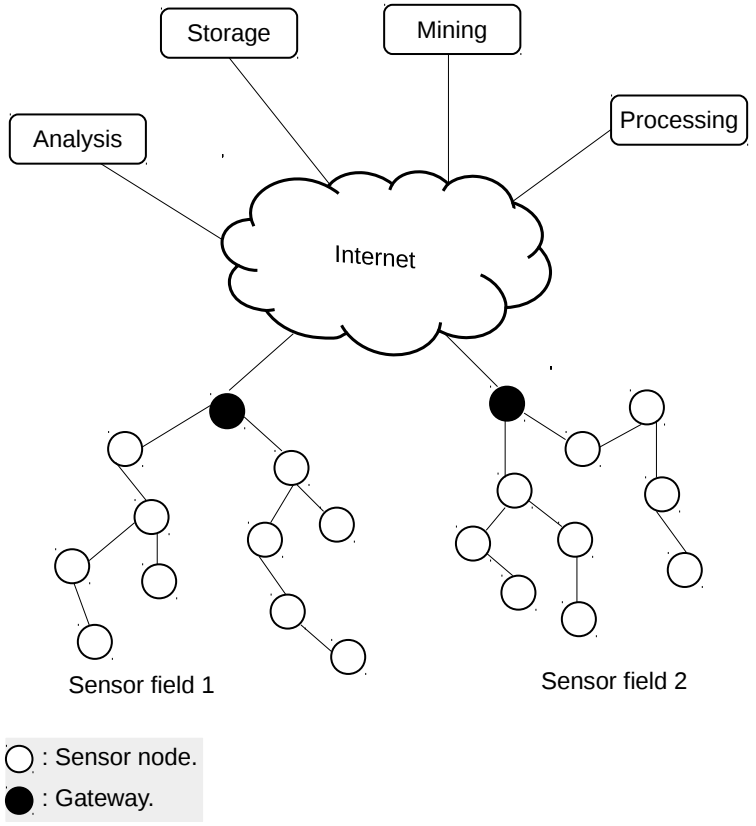



Figure 2.2: Wireless sensor networks and their communication.

between the client/user and the WSN. Figure 2.2 shows some deployed sensors and their communication.

WSNs are usually considered constrained networks because nodes are equipped with restricted processing, memory, and energy capabilities [YMG08]. Such resource limitations make the design of high Quality of Service (QoS) and energy-efficient applications a very challenging task. In fact, most methods and routing protocols try to solve this trade-off between achieving QoS while using as little energy as possible. To develop applications with reasonable long lifetime, energy-saving policies should be followed [MI12].

This chapter is organized as follows. Section 2.1 explains sensor node architectures

while Section 2.2 illustrates the constraints and challenges that should be considered when designing WSNs. WSN applications are presented in Section 2.3, and the WSN protocol stack is presented in Section 2.4. Section 2.5 explains topologies and deployment of WSNs in the environment of interest. The QoS requirements and routing protocols used in WSNs are explained in Sections 2.6 and 2.7, respectively. Finally, Section 2.8 presents a summary of the chapter.

**Contributions:**

- Survey on WSNs. Besides discussing their features and applications, proposals from literature to provide adequate deployment, quality of service and routing are discussed.

## 2.1 Sensor Architecture

Any sensor node consists of four main parts, as shown in Figure 2.3, although additional parts may be required by some applications [ASSC02, ZJ09]. The main components of any sensor node are:

1. Sensing unit: Contains one or more sensors and an Analog to Digital Converter (ADC). When sensors sense the environment, they generate analog signals. These signals are converted to digital signals by the ADC, then passed to the processing unit.

2. Processing unit: Contains a microcontroller or microprocessor that controls the sensor node tasks. The processing unit is usually connected to a storage unit.

3. Transceiver unit: Contains a short range radio that connects the sensor node to the other nodes in its range.

4. Power unit: Contains a battery to supply the nodes with power. In many applications the battery of sensors can not be recharged [MC14].

Other sub-units that may be required by specific applications include:

1. Power generator: Supplies power to the node (e.g., solar cells).

2. Mobilizer: Moves the node to another location so that a specified task is performed.

3. Global Positioning System (GPS): Specifies the node location. Some applications need it to specify the location of network operations.

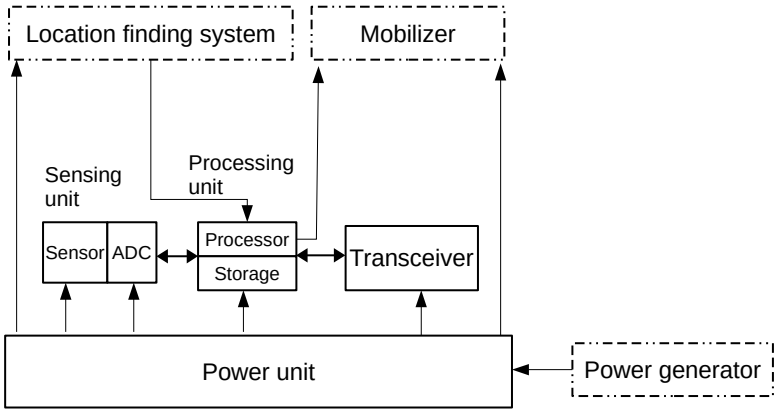



Figure 2.3: Sensor node structure.

## 2.2 Challenges and Constraints

Despite the similarities with other distributed systems, WSNs are subject to special constraints and challenges that affect the design of these constrained networks. Thus,

protocols and algorithms used in WSNs are different from those used in traditional networks. In this section, the main constraints considered when designing WSNs are presented [DP10].

### 2.2.1 Energy

Sensor nodes are usually powered with batteries (limited energy budget) that can be replaced or recharged when depleted. However, replacement or recharging may not be an option for many sensor nodes and depleted nodes are often discarded. Thus, considering the energy limitations of sensor nodes, when designing protocols and algorithms, becomes very critical in WSN applications.

### 2.2.2 Self Management

Because WSNs are usually deployed in harsh environments and unattended places, operated without human intervention, it is critical that their nodes have the ability to collaborate, manage themselves, and operate under some failure scenarios or environment changes. In other words, every sensor node should be a self-managing device that is able to collaborate with its neighbors, should be able to sense and detect events even when the environment changes, and should be able to protect itself from attacks [Mil07]. These features should be considered when designing and implementing WSN applications in a way that excessive energy utilization is avoided.

### 2.2.3 Security

Security techniques used in wired and wireless networks can not be applied to WSNs, due to their unique constraints. In particular, WSNs have specific features which are not presented in other networks, that make the design of security methods very challenging [Yan14]. These features are:

- Sensor networks can contain thousands of nodes with limited energy, memory, processing, and communication capabilities.
- Sensor nodes are often deployed in wide geographical areas, far away from human intervention.
- Sensor networks may interact with people (e.g., health monitoring), animals (e.g., animal tracking), and environment (e.g., detecting wildfire), which brings extra security issues.

### 2.2.4 Environment

Two issues can affect the sensor nodes when it comes to the environment. First, sensor nodes can be deployed very close or directly in the phenomenon under observation. Second, sensor nodes can work under hard conditions (e.g., pressure and very hot or cold weather) or in very harsh environments (e.g., debris and battlefield). They can also work in busy intersection, bottom of ocean, a twister, home or large building, etc. In other words, the environment in most cases makes the management and processing of data (generated by sensor nodes) very hard.

### 2.2.5 Power Consumption

A sensor node is a micro-electronic device that is usually equipped with limited power source and in some cases can be recharged [ASSC02]. However, many applications do not support the recharge. As a result, sensor networks depend mainly on the lifetime of sensor nodes. Usually, sensor nodes consume their energy during data communication and processing. However, the energy consumed by data transmission is much higher than that for data processing. For this reason, it is critical to design nodes that consume very low energy during data transmission [PMEV00]. Furthermore, the amount of traffic and the transmitting distance should be considered when designing routing protocols in WSNs to reduce the energy consumption.

### 2.2.6 Scalability

This property is related with the ability of protocols and techniques to operate and perform well as the number of deployed nodes in the network increases. Depending on the network application, the number of nodes in a network can be quite big. Any new techniques added to sensor networks should be able to work well with any number of nodes. The network density, given the coverage range of sensor nodes, $R$, can be expressed as [ASSC02]:

$$\varphi(R) = \frac{N \times \pi \times R^2}{A} \tag{2.1}$$

where $N$ is the number of nodes deployed in region $A$. The required network density depends on the application. For example, a machine diagnosis application in $5 \times 5$ $m^2$ region can contain 300 sensor nodes [SCI+01]. On the other hand, an habitat monitoring application can contain from 25 to 100 sensors in a large region [CEE+01].

### 2.2.7 Hardware Cost Constraints

When it comes to the design of wireless sensors, the main objective will be to create small, low cost, and efficient devices. These hardware constraints also affect the design of the protocols and algorithms used in WSNs, which should be implemented efficiently to satisfy these constraints.

## 2.3 Wireless Sensor Network Applications

Since sensor nodes can be deployed anywhere and in any number to cover any intended area, WSNs have numerous different applications. Sensor networks have mainly been used in [Sto05]:

- Military applications: The first emergence of WSNs was in military applications. Nowadays, these networks play a significant role in military commands, communication, monitoring either friend or enemy force, intelligence, and surveillance.
- Environmental applications: With sensor nodes scattered in unattainable places or harsh environment, many previously impossible applications became possible and are now useful. Some examples are habitat monitoring, wildfire detection, animal tracking, precision farming, and disaster relief applications.
- Health applications: Recently, sensor networks became an essential part in health care systems. Hospitals are equipped with numerous sensor networks that monitor patients and return their state and location. This can be critical for patients requiring constant monitoring, like diabetic patients.

- Industrial applications: One of the fields getting most benefits from WSNs is the industry. These applications could be in buildings, constructions, and bridge condition monitoring to provide health information of the structure. Furthermore, sensor nodes can monitor the condition of machines, providing in some cases automatic maintenance, very important in production processes. Another vital role of sensor networks is to monitor production performance [EMN+13].

## 2.4 Protocol Stack

Sensor nodes follow a protocol stack when performing tasks like sensing the environment, communicating with each other, and delivering the sensed data to the gateway/sink. Also, the gateway is usually connected to the internet or to the end user (see Figure 2.2). Such protocol stack comes with multiple layers, each layer having its own tasks, and with multiple planes. As shown in Figure 2.4, these are: *i*) physical, data link, network, transport, and application layers; *ii*) power management, mobility management, and task management planes [KB17, ZJ09]. The next subsections detail these layers and planes.

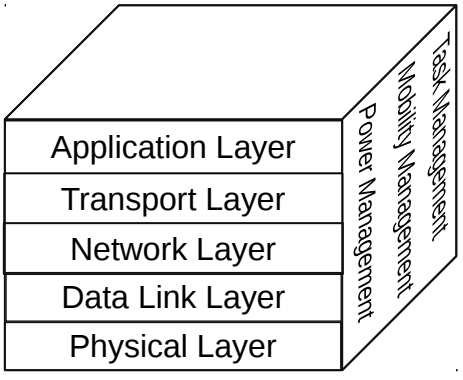



Figure 2.4: Wireless sensor network protocol stack.

### 2.4.1 Physical Layer

This layer converts the bits coming from the data link layer to signals that can be transmitted through the wireless communication medium. For this purpose, the physical layer performs tasks like modulation/demodulation of digital signals, and other. A main concern in WSNs is to find simple and low cost modulation techniques and transceiver architectures. These techniques and architectures should be reliable and efficient enough to produce the intended services [KW07]. Other tasks related to the physical layer are frequency selection, carrier frequency generation, transmission medium, data encryption, and signal detection.

### 2.4.2 Data Link Layer

This layer is responsible for reliable link communication. It consists of a set of subsystems, each one having its own functionalities, as described bellow:

- Medium Access Control (MAC): Two goals of MAC protocols in wireless sensor networks are: *i*) to achieve a direct connection between nodes, so that data can be transmitted to a neighboring node; *ii*) to share the medium between competing nodes in an efficient way.

- Power saving modes: A common way to save power is to turn the transceiver off, when possible. However, if the transceiver is switched off whenever there is an idle period, the overall energy consumed will be greater than if the transceiver is left on. This is because it takes some time for nodes to switch from one state to another and, consequently, more energy is necessary to transmit the same amount of data. For this reason, the transceiver should turn off when the idle time is expected to reach a specific threshold.

- Error control: A main responsibility of data link layer is to correct errors during data transmission. Two ways to control errors are Automatic Repeat Request (ARQ) and Forward Error Correction (FEC). In ARQ, when a receiving node detects an error in the received data, it sends a request to the sender so that the original data is retransmitted. On the other hand, FEC occurs when the receiving node receives enough redundant data that allows the node to correct the errors in the received data, avoiding ARQ technique.

### 2.4.3 Network Layer

This layer is responsible for routing sensing data coming from sensor nodes. Multiple sensor nodes collect data from the surrounding environment and send it to a sink/gateway. This many-to-one traffic pattern differs from the pattern used in other networks, which makes the protocols used in traditional networks not applicable in WSNs. Furthermore, traditional network protocols are not energy efficient, which is a main concern in WSN protocols. Therefore, when designing a network layer and routing protocols for WSNs, energy constraints and traffic patterns must be considered.

### 2.4.4 Transport Layer

This layer is responsible for maintaining the end-to-end data delivery. Contrarily to nodes in unconstrained networks, sensor nodes cannot store much data and acknowledgments (it is very expensive), meaning that end-to-end retransmission and window based congestion techniques, used in Transport Control Protocol (TCP), cannot be applied to WSNs. Therefore, new techniques to ensure the end-to-end communication are required. In addition, sensor networks are application specific, which means

that the reliability required by a certain application may not be required by other applications.

### 2.4.5 Application Layer

This layer includes the applications running at the node, which are usually responsible for data and management tasks. Some application protocols include: *i*) Sensor Management Protocol (SMP); *ii*) Task Management and Data Advertisement Protocol (TMDAP); *iii*) Sensor Query and Data Dissemination Protocol (SQDDP).

## 2.5 Wireless Sensor Network Topologies

Hundreds, and sometimes thousands, of sensor nodes can be deployed in difficult sensor fields [IGE00]. This kind of deployment may lead to unattended and inaccessible nodes, which means that when a node fails the maintenance of the network becomes a challenging task. The network topology, therefore, is a critical parameter that affects performance factors like network latency and robustness [KB17]. Regarding how sensor nodes are deployed in the area of interest, the following possibilities exist:

- **Pre-deployment and deployment phases:** Two ways to deploy the nodes include throwing them randomly (e.g., falling from a plane, dropping by a catapult that throws the nodes from a ship board) or placing them carefully one by one in the environment (e.g., placing one by one at factory, which can be done by humans or robots) [ASSC02]. There are some concerns that should be taken into consideration at pre-deployment phase:

  (i) Minimization of installation cost;

(ii) Maximization of arrangement flexibility;

(iii) Enforcement of fault tolerance and self-organization.

- **Post-deployment phase:** After deployment of nodes, their topology may change when the nodes change their location, reachability, energy, malfunction, or task details [IGE00, MKQP01].

- **Re-deployment of additional nodes phase:** Additional nodes could by deployed to replace the failed nodes or due to any change in tasks. When new nodes are added, the network may need to be re-organized.

## 2.6 QoS in Wireless Sensor Networks

To estimate and enhance the network performance, network QoS parameters like energy consumption, reliability, availability, bandwidth, and latency must be carefully specified. Methods and protocols should try to ensure such QoS requirements. QoS can be application specific or network specific. The application specific relates to how each application differs in its specifications, requirements, and objectives. Network specific, on the other hand, relates to the requirements that a network must try to fulfill, regardless of the applications using it [AKA+17]. Achieving QoS in such networks can be quite challenging because of the following [BSS+10, BZM11]:

- Resource limitations: Energy, processing, bandwidth, and transmission range are very common constraints in WSNs. Consequently, providing QoS in such networks is difficult. For example, depleted batteries cannot be recharged in many sensors, which imposes limitations on protocols and methods for an efficient use of energy.

- Data redundancy: Hundreds and even thousands of sensor nodes may be deployed in the area of interest, which results in a lot of redundant data in most of the cases. Although data redundancy can help in reliability issues, it increases the energy consumption significantly. Therefore, many techniques like data aggregation and data compression are used to minimize the energy consumed by reducing the volume of the transmitted data. However, these techniques bring additional computational activities and delay in some nodes (e.g., cluster heads) and may complicate the design of QoS in WSNs. An alternative solution can be NC, where transmitted data is encoded and decoded in intermediate encoding nodes, as will be detailed in Chapter 3.

- Dynamic network topology: In mobile WSNs, nodes can move and change their position to provide better coverage or due to link/node failure. Mobility complicates the task of achieving QoS requirements.

- Node deployment: From a cost perspective, deploying sensor nodes in a random/unstructured way is better than pre-planned/structured deployment. On the other hand, structured deployment is better to achieve the QoS requirement. When sensor nodes are placed in pre-planned locations, QoS methods will have network information that helps in neighbor and path discovering.

- Heterogeneous sensor nodes: Some applications require different types of sensor nodes, which results in different data generated at different rates. For example, an application that monitors temperature, pressure, and humidity may need three different sensors. Also, different sensors to capture moving, image, or video can be included in a single application. Thus, to fulfill QoS in such applications, different constraints and requirements should be considered, according to the intended application and its desired results.

## 2.7 Routing Protocols in WSNs

This section presents different routing or data dissemination protocols used in WSNs, based on different classification criteria.

### 2.7.1 Cluster based Routing Protocols

Low-Energy Adaptive Clustering Hierarchy (LEACH), proposed in [HCB+02], is one of the most common clustering protocols in WSNs. It is appropriate for applications that need constant monitoring and periodic reporting of information. LEACH considers two steps: a setup step, to arrange the nodes in clusters, and a steady step to perform data routing and aggregation. In the first step, the network is divided into clusters and a cluster head is chosen for each cluster. Then, at the second step, data aggregation is performed inside the clusters for transmission to the sink.

In LEACH, a predetermined percentage of nodes that can be cluster heads, denoted by $P_r$, is defined at setup phase. A random number $y$, between 0 and 1, is generated for each node $n \in N$, where $N$ is the set of nodes that were not cluster heads in the last $1/P_r$ rounds. Such number is compared with a threshold $th$, and if $y > th$ then the node is set as cluster head. The threshold $th$, calculated at each round $\Gamma$, is given by:

$$th = \frac{P_r}{1 - P_r \times (\Gamma \mod (1/P_r))} \tag{2.2}$$

After being chosen, cluster heads send a message to other nodes informing that they are the new cluster heads. When non cluster heads receive such message, they decide which cluster to join according to their coverage range. Even though LEACH

reduces the energy consumed and, accordingly, increases the network lifetime, it has some limitations [GCS14]:

- It does not provide any information related to the number of cluster heads in the network;
- When a cluster head fails, due to any reason like depleted battery or environment, cluster members will never be able to communicate their data to the sink/gateway;
- LEACH considers that all nodes are able to send the data to the sink, which gives the ability for each node to be a cluster head and this may not be advantageous in terms of energy constraints.

In [SSS10], an Hybrid Energy Efficient Distributed (HEED) clustering approach is introduced. The main goal of this protocol is to maximize the lifetime of the network by creating efficient clusters. Two parameters are considered when selecting the cluster heads: the remaining energy of the node and the intra-cluster communication cost. Intra-cluster communication cost reflects the communication between a cluster head and its nodes. Consequently, the minimum communication cost is the mean of minimum power levels that nodes in the cluster consume when communicating with the cluster head. This cost, called the Average Minimum Reachability Power (AMRP), can be calculated as:

$$AMRP = \frac{\sum_{n=1}^{r} P_n^{\min}}{r} \tag{2.3}$$

where $P_n^{\min}$ is the minimum power required by node $n$ in the cluster to send packets to the cluster head, and $r$ is the number of nodes in the cluster.

The clustering process in HEED needs a number of iterations, $N_{\text{iter}}$, to be accomplished. Each iteration takes time $t_c$ which should be enough to receive messages from nodes within the cluster range. Initially, and before running the clustering algorithm, each node sets its probability $P_{\text{ch}}$ of becoming a cluster head to:

$$P_{\text{ch}} = P_r \times \frac{E_{\text{residual}}}{E_{\text{max}}} \tag{2.4}$$

where $P_r$ is the percentage of initial cluster heads, $E_{\text{residual}}$ is the current energy of the node, and $E_{\text{max}}$ is the initial energy of the node, which is supposed to be full. The probability $P_{\text{ch}}$ should not be less than a threshold $th_{\text{min}}$.

Clustered diffusion with Dynamic Data Aggregation (CluDDA) is a new hybrid technique, which has both clustering and diffusion mechanisms [CH03]. In this technique, the sink generates a query (interest message) with a detailed definition of tasks that should be performed on the data to generate a suitable reply. The knowledge of the query is used to reduce the amount of processed data. This mechanism of combining clustering with direct diffusion saves the energy of nodes because only cluster heads and gateways transmit interest message and the rest of nodes will be silent till they have a request to serve. A great feature of CluDDA is the query cache, provided by cluster heads and gateway, that contains the aggregated data and the addresses of neighboring nodes that created data messages.

### 2.7.2 Chain based Routing Protocols

In cluster based data routing protocols, if the nodes are further away from the cluster head then they will consume too much energy when transmitting data to their cluster head. To reduce the energy consumed, each node can send data only to its closest neighbors. The Power Efficient data GAthering protocol for Sensor

Information Systems (PEGASIS) uses such approach and arranges nodes in a linear chain [LRS02].

A chain can be formed by applying a greedy algorithm. Assuming that the nodes have a global knowledge of the network, the idea is for the furthest nodes to start the chain formation, and then each node chooses its closest neighbor to be its successor. For data forwarding, each node receives data from its neighbor, accumulates with its own data and then sends it to the next node in the chain. Sending data to the sink is performed by a node called leader, which is similar to the cluster head in a clustering approach.

In [LRS02], two protocols are proposed: binary chain based scheme and three-level chain based scheme. In binary chain based scheme, each node sends data to the nearest neighbor in a specific level. At each level, the nodes collecting data (from other nodes) build a chain in the next level of the hierarchy. At each higher level, the same process is applied, and in the highest level the leader sends the data to the sink. In a three-level chain based scheme, all $r$ nodes are arranged in a linear chain and divided into $G$ groups, each group containing $r/G$ sequential nodes of the chain. In this hierarchy, only one node from each group will participate in the next level. The $G$ nodes in the second level will also be fragmented into two groups. In this case, there will be only three levels to deal with.

### 2.7.3 Tree based Routing Protocols

These protocols are used in topologies where network nodes are organized as trees rooted at the sink. Data is routed from tree leaves (source nodes) toward the tree root (sink/gateway).

Energy Aware Distributed heuristic Algorithm (EADAT), introduced in [DCX03], builds an efficient energy tree based algorithm. In this algorithm, control messages are propagated by the nodes. Such control messages include:

1. ID: Sensor node ID;
2. Parent: Node's parent;
3. Power: Node power;
4. Status: State of the node (leaf, non-leaf, undefined state);
5. HopCnt: Number of hops from the root.

In EADAT, the sink starts with a broadcasting msg (ID, -, $\infty$, status, $\emptyset$), assuming that the root (sink) has infinite power supply. When a node $n$ receives a control message for the first time, it sets up its time to $t$, which counts down when the channel is idle. During this process, node $n$ will record the parent node that has the highest residual power and the shortest path to the root. When $t$ times out, $n$ broadcasts msg ($ID_n$, $\text{parent}_n$, $\text{status}_n$, $\text{hopCnt}_n$) where the $\text{hopCnt}_n = 1 + \text{hopCnt}_{\text{parent}}$. If a node $\hat{n}$ receives a message from $n$ saying that $\text{parent}_n = \hat{n}$, then $\hat{n}$ will mark itself as a non-leaf node, otherwise $\hat{n}$ is a leaf node. This process continues until the aggregation tree is built.

The advantage of this algorithm is that the nodes with higher energy have a greater chance to be parent nodes. For maintenance, each node has a threshold $th_n$, which is used to check the node power. If the power is less than $th_n$, the node broadcasts a help message for some time units and its radio will shutdown. The child node that receives the help message will switch to a new parent or enters in danger state. When the node, in danger state, receives a HELLO message from a neighbor node $n$ that has shorter distance to the sink, it invites $n$ to join the tree. A great feature

of the EADAT algorithm is that the network lifetime increases linearly with the density of the network.

In [TK03], the goal of the Power Efficient Data gathering and Aggregation Protocol (PEDAP) is to increase the network lifetime in terms of the number of rounds. In each round, data is transmitted from several nodes to the sink. PEDAP is a minimum spanning tree based protocol that can improve the network lifetime even though the sink is inside the sensing area. PEDAP decreases the total energy, expended with communication in each round, by computing the cost of using the minimum spanning tree:

$$C_{ij}(k) = 2 \times E_{\text{elec}} \times k + E_{\text{emp}} \times k \times d_{ij}^2 \tag{2.5}$$

where $C_{ij}(k)$ is the cost of transmitting $k$ bits from node $i$ to node $j$, $E_{\text{elec}}$ is the energy consumed by the transmitter and receiver circuitry, $E_{\text{emp}}$ is the energy consumed by the transmit amplifier, and $d_{ij}$ is the distance between the nodes $i$ and $j$.

To control the load between nodes, the residual energy of each node should be taken into consideration. In this case, a node with low energy should keep its remaining energy to transmit its data to the closest neighbor, and not receiving or forwarding data coming from its neighbors. This is performed in the new version of PEDAP, which is PEDAPAP, by changing the cost to:

$$\hat{C_{ij}}(k) = \frac{C_{ij}(k)}{e_i} \tag{2.6}$$

where $e_i$ is the residual energy at node $i$. This energy $e_i$ is normalized with respect to the battery energy when the node starts sensing.

## 2.8 Summary

In this chapter, a discussion on WSNs is presented. This includes the definition of a WSN, its types and characteristics. The architecture of sensor nodes and their protocol stack is also discussed. The chapter also explains how these sensors can be deployed in the environment of interest, and what are the constraints that should be taken into consideration when using these networks. QoS in WSNs is explained, and the last section gives an overview of the routing protocols used in such kind of networks.

# 3

# NETWORK CODING

BEFORE network coding, the network performance increase efforts were focused on improving network throughput and packet delivery by selecting the best path and the best network topology. Additionally, data retransmission is always an option to ensure transfer reliability. However, in the beginning of the last decade Ahlswede et al., [ACLY00], introduced NC as a new concept to improve network throughput and reliability. Unlike the store-and-forward principle of conventional communication systems, NC allows network nodes to process data before forwarding. More specifically, a node can perform linear combinations on the received packets, and route one or more combined packets.

The combination tasks in network coding can be binary operations like bitwise XOR, which is the simplest coding scheme that can be applied to the packets. Linear operations can also be performed under finite field setting [FLBW06, LYC03]. This

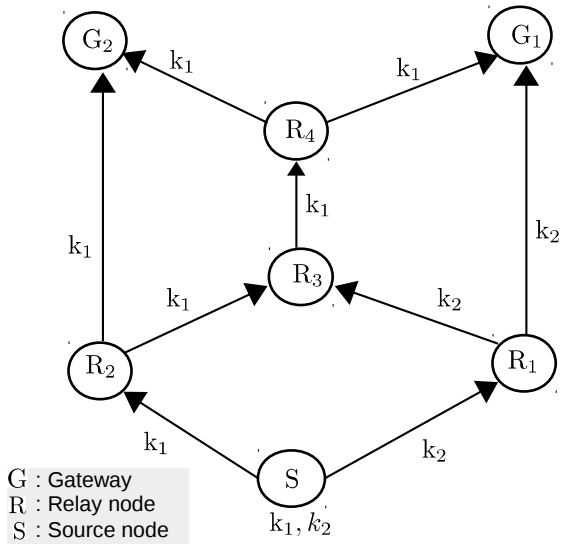



Figure 3.1: Traditional routing in wired multicast network.

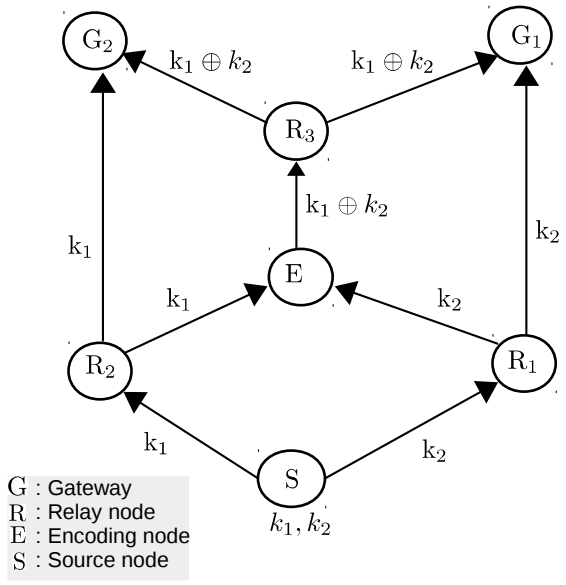



Figure 3.2: Binary coding in wired multicast network.

technique allows destination nodes to decode and retrieve the original data as long as enough linearly independent coded packets are received. Figure 3.1 shows a multicast example in wired network when using traditional routing. In this example the source $S$ sends two packets $k_1$ and $k_2$ to the gateways $G_1$ and $G_2$. As shown in the figure only gateway $G_1$ receives both packets, while gateway $G_2$ receives just

$k_1$, considering one-packet link capacity. On the other hand, when NC is used, both gateways receive $k_1$ and $k_2$, as illustrated in Figure 3.2. With NC, $k_1$ and $k_2$ are XORed and sent as one packet to both gateways, which perform the decoding process and retrieve the original ones. In particular, when using NC both packets sent are received by both destinations in nine packet transmissions. On the other hand, in traditional routing (Figure 3.1) additional transmissions are required to deliver the same data.

The rest of this chapter is organized as follows. Section 3.1 describes the benefits of NC, while Section 3.2 presents binary and random linear NC, respectively. In Section 3.3, the encoding and decoding processes are illustrated. Section 3.4 discusses the work related to NC. Finally, Section 3.5 presents the summary of the chapter.

**Contributions:**

- Survey on network coding theory. Besides describing their benefits and the encoding/decoding process, network coding schemes proposed in the literature are discussed.

## 3.1 Network Coding Benefits

With NC, networks can achieve benefits like better throughput, robustness and security. Such advantages are detailed in the following subsections [HL08].

### 3.1.1 Throughput

With network coding, the fact that packets are combined before their transmission will allow more data to be delivered for the same number of packet transmissions,

when compared with traditional routing. This is illustrated in Figures 3.1 and 3.2 for a wired multicast network. Regarding wireless networks, their broastcast nature also allows throughput benefits. A simple example would be a scenario where two source nodes need to exchange their data but they are not in each other's coverage area. In this case, an additional node is required to act as a relay node between these sources. Figure 3.3 shows that the two sources need four transmissions for the data to be exchanged in traditional wireless network. On the other hand, with NC only three transmissions will be required to accomplish the same task, as illustrated in Figure 3.4.

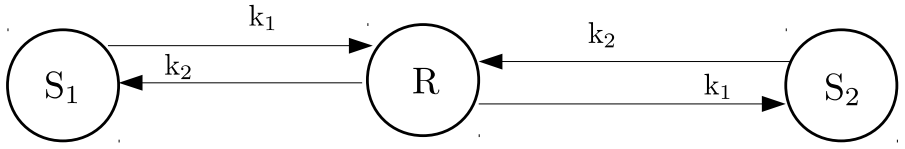



Figure 3.3: Traditional transmission in wireless network.

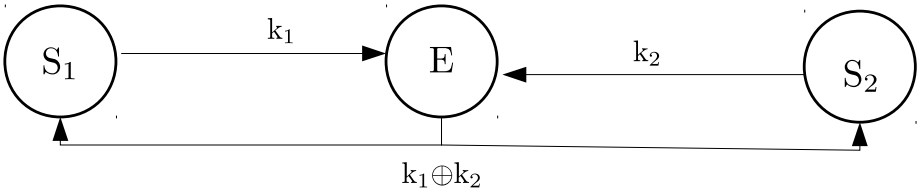


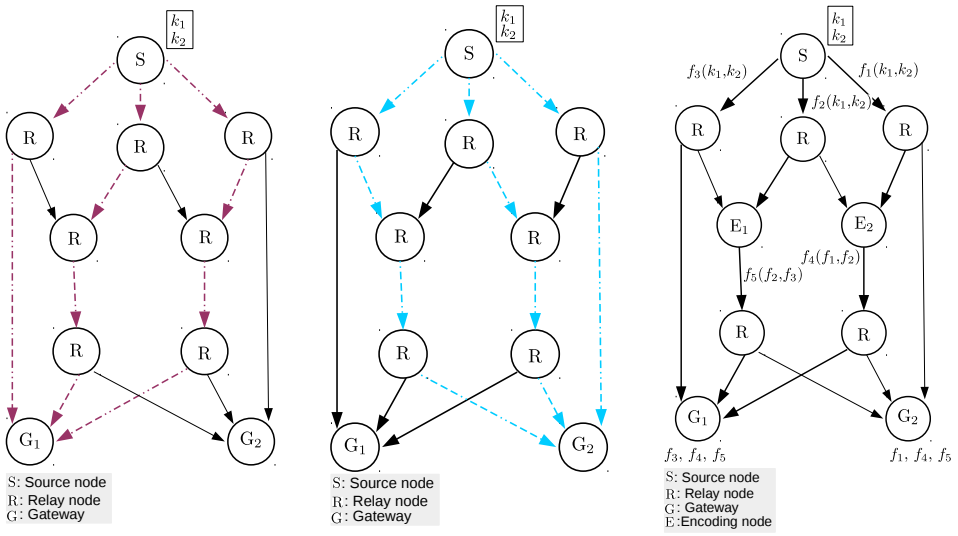

Figure 3.4: NC based transmission in wireless network.

### 3.1.2 Robustness

Packets can be lost due to collision, link outage or buffer overflow. To ensure that packets are received at the destination, positive acknowledgments (ACK) and negative acknowledgment (NACK) messages can be used. With this mechanism, the

destination sends an ACK message to the source to confirm correct receipt, avoiding retransmissions, or sends a NACK to ask for retransmission. When using these mechanisms in NC based WSNs, however, there are tradeoffs. The main problem when dealing with NACKs is deciding which node should respond and send the missing messages. Nodes can not perform decoding tasks until a sufficient number of packets have been received. Also, if NACKs are sent by all nodes then too much energy will be consumed. To overcome this problem, each node with the missing messages, and hearing NACKs, should wait for a period of time to see whether any of its neighbors will send the requested messages. If that does not happen, and before the timeout, the node responds by sending the requested messages.

Network coding can achieve robustness against packet drops resulting from link failures, with no need for retransmission or rerouting. Figure 3.5 shows an example of a multicast network with one source and two gateways. As shown in Figures 3.5a and 3.5b, the maximum flow from source $s$ to any gateway is 3, assuming the link capacity is one unit. Regardless of the output port through which packets $k_1$ and $k_2$ go out, in a link failure scenario one of the gateways may not receive both packets. In case of network coding, illustrated in the Figure 3.5c, the source and nodes $E_1$ and $E_2$ can linearly combine the received data. Consequently, when a link fails, regardless its location, both gateways can decode and recover the lost data. This is so because each gateway will receive at least two out of three independent coded data units [KM14]. Thus, with network coding, a multicast rate of 2 is always achieved no matter which link fails.




(a) Flows towards $G_1$. (b) Flows towards $G_2$. (c) Transmission with NC.

Figure 3.5: NC robustness in multicast network.

### 3.1.3 Complexity

Optimal routing algorithms and NC can take both the best of the available bandwidth, and in some cases give similar solutions. However, complex routing algorithms may be required to obtain adequate solutions. An example of this complexity is the multicast routing when selecting the minimum cost subgraph, which takes us to the Steiner tree problem that is known to be NP-hard. With NC the same problem can be seen as a linear optimization solved with distributed solutions of low complexity [HL08].

### 3.1.4 Security

Data transmitted in plain text form is vulnerable and easy to be hacked. On the other hand, with NC the attackers will have difficulty in getting the original message because data is encoded. For example, in Figure 3.2, when an attacker obtains the

coded data ($k_1 \oplus k_2$) it is difficult to recognize $k_1$ or $k_2$. In some cases, however, using NC cannot protect the data from attacks. Considering Figure 3.2, if node $E$ is a malicious node then it can send a malicious packet, instead of sending the coded packet $k_1 \oplus k_2$. In such scenario, contrarily to original packets, detecting the manipulation performed on the coded packet is more difficult. Thus, NC can be seen as having security drawbacks.

## 3.2 Encoding Schemes

### 3.2.1 Binary Network Coding

In binary NC, bitwise XOR can be used to encode the packets as previously shown in Figure 3.2. In this illustration, the intermediate node $E$ works as an encoding node and applies a XOR between packets $k_1$ and $k_2$, generating a single coded packet which is forwarded to node $R_3$ (works as a relay node). As a final step, node $R_3$ sends this coded packet to both destinations. Therefore, gateways $G_1$ and $G_2$ can retrieve both packets by applying XOR again ($k_1 \oplus k_2$) with $k_1$ and $k_2$, respectively. This scheme is considered computationally light.

### 3.2.2 Random Linear Network Coding (RLNC)

Random linear coding encodes a group of $r$ packets by creating a linear combination of the form $\sum_{i=1}^{r} \alpha_i k_i$, where $\alpha_i$ is a coefficient generated over finite field and $k_i$ is the packet. A finite field is any field containing a finite number of elements. When the field size is large enough (i.e., $2^8$ or $2^{16}$), the probability of retrieving the original packets at the destination increases significantly. Large field size increases

the probability of generating independent coded packets (innovative packets), which helps the destination node in the decoding process. This is because the destination needs to receive enough linear independent coded packets to ensure a successfully decoding. Figure 3.6 shows an example of this kind of coding, where the source node $S$ sends two coded packets to nodes $R_1$ and $R_2$. These coded packets are combinations of the original packets, $k_1$ and $k_2$, and the corresponding coefficient $\alpha_i$. The encoding node $E$ receives the coded packets, re-encodes them, and sends the re-coded packet to gateway $G_1$ and $G_2$ using $R_3$ as a relay node.

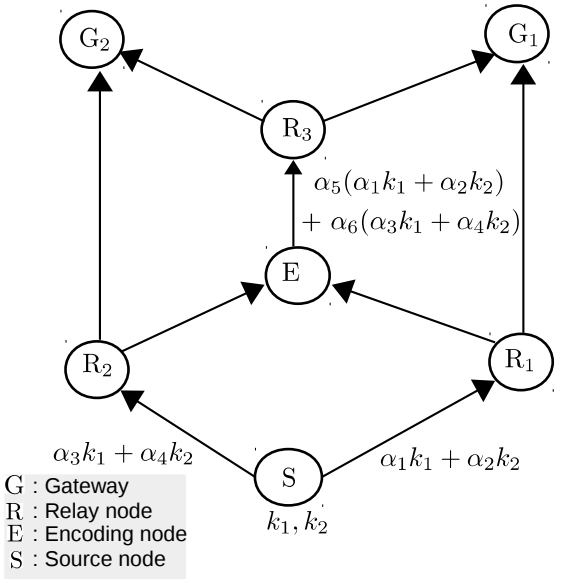



Figure 3.6: Random linear coding in multicast network.

## 3.3 Encoding/Decoding Process

Original packets are divided into symbols of a specific size. Then, packets originating from one or more sources, that need to reach destinations, are linearly combined. Such combination is performed by multiplying each symbol with a scalar coefficient generated randomly from a finite field. This process will generate one or more

coded packets with the same size of the original packets. The coded packets are then transmitted along the network, and encoding can be recursively performed on them. When the packets reach the destination, the decoding process can be applied to extract the original packets. To perform the encoding process the following should be considered:

- Each packet consists of $\ell$ bits, and small packets are padded with 0's;
- $s$ consecutive bits is a symbol over finite field $\mathbb{F}_{2^s}$;
- Each packet is a vector of $\frac{\ell}{s}$ symbols;
- Linearly combined packets of length $\ell$ results into coded packets of length $\ell$.

Now, assume the original packets $(k_1, ..., k_n)$. A coded packet $\omega$ will be:

$$\omega = \sum_{i=1}^{n} \alpha_i k_i \tag{3.1}$$

where coefficients $\alpha_1, ..., \alpha_n$ are in $\mathbb{F}_{2^s}$. Since the summation must occur for every symbol position,

$$\omega^u = \sum_{i=1}^{n} \alpha_i k_i^u \tag{3.2}$$

where $k_i^u$ and $\omega^u$ are the $u^{\text{th}}$ symbol of $k_i$ and $\omega$, respectively. The encoding process occurs recursively through the network. In particular, when any intermediate node receives the coded packets $\omega_1, ..., \omega_m$, it encodes them again to generate a linear combination of these packets as:

$$\hat{\omega} = \sum_{j=1}^{m} \hat{\alpha}_j \omega_j \tag{3.3}$$

where $\hat{\alpha_1}, ..., \hat{\alpha_m}$ are the coefficients over $\mathbb{F}_{2^s}$ generated by the intermediate nodes. The re-coded packet $\hat{\omega}$ is tagged with sequence of encoding coefficients given by:

$$\hat{\alpha_i} = \sum_{j=1}^{m} \hat{\alpha_j} \alpha_i^j \tag{3.4}$$

When a node receives $m$ arrivals $(\alpha^1, \omega^1), ..., (\alpha^m, \omega^m)$, it needs to retrieve the original packets. This can be achieved by solving the system:

$$\omega^j = \sum_{i=1}^{n} \alpha_i^j k_i, \forall j = 1, ..., m \tag{3.5}$$

This linear system has $m$ equations that need to be solved to extract the $n$ unknowns (unknowns are $k_i$). To perform this, it requires that $m \geq n$ to be able to recover all original data, which means that the number of received packets should be at least equal to the number of the original packets. However, this case ($m \geq n$) is not sufficient since the received packets may be linearly dependent [FLBW06].

## 3.4 Related Work

Network coding has been explored for different kinds of networks, which include wired, wireless, sensor, and mobile networks. Each type of network applies NC to achieve a certain goal, which may relate to throughput increase, security improvement, reliability (decrease of packet drops), and energy saving.

In [dALFMA18], dynamic network coding is used in multi-source systems that organize nodes as peers to form a P2P network, shown in Figure 3.7. In these systems, a coordinator server is used to assign the roles for peers, ensuring that the NC process is accomplished successfully. The paper employs dynamic NC, where all

peers in the P2P distributed system participate in the encoding/decoding processes. Specifically, the authors propose a model that contains different types of nodes: *i*) independent sources, each source having a different type of data to distribute to the requesting peers; *ii*) peers that request the data from sources and share this data between them. Network coding is used in this multi-source systems to improve the distribution time.

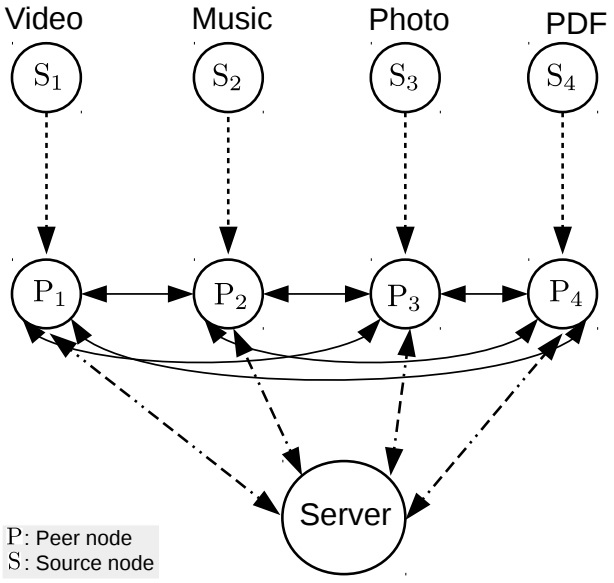



Figure 3.7: Dynamic network coding with nodes organized as peers in P2P network [dALFMA18].

CodeDrip, proposed in [JTV+17], is a data dissemination protocol that utilizes network coding to achieve reliability, energy saving, and to increase the data dissemination speed in WSNs. When network coding is used, packet drops can be recovered in the decoding process, using non dropped packets. Such kind of packet recovery, avoiding retransmissions, increases the speed of data dissemination. The authors state that existing dissemination protocols were not able solve the trade-off between energy consumption and increasing the dissemination speed. In particular, such protocols try to selectively retransmit the lost data, avoiding redundancy, to save

energy but at the same time introduce a large delay. In contrast, CodeDrip achieves a balance between energy consumption and data dissemination speed. To combine packets, binary coding with finite filed $\mathbb{F}_2$ (XOR) is used to reduce the overhead induced by the encoding/decoding process.

In CodeDrip, each node has two buffers one for the original data, and the other for the combined data. When an original packet arrives, the node stores it in the original data buffer, and waits for its time to send. On the other hand, when a combined packet arrives, the node checks if it is possible to decode this packet using the packets in buffers. If this is not possible, the node stores this coded packet in the combined data buffer and waits for other packets to arrive. When an original data packet is to be sent, the node must decide either to send the original packet or to encode this packet with other packets, selected randomly from its buffer, for sending.

The work in [HMS+03] is based on the model presented in [KM03]. Let us assume a network represented as a directed graph $G = (V, E)$, where $V$ is the set of vertices and $E$ is the set of edges. The model contains discrete independent processes $k_1, ..., k_n$, where $k_i$ is generated at source $i$ and needs to be communicated to multiple destinations. Moreover, let the corresponding output process at a destination $D$ be represented as $Z(D, i)$. Note that the data is transmitted as vectors of bits with the same length $\ell$. The model contains $r$ links that are either incoming ($\text{head}(e) = \nu$) or outgoing ($\text{tail}(e) = \nu$) links to/from node $\nu$. Let $Y(e)$ stand for a random process transmitted through the link $e$, such that $\text{tail}(e) = \nu$. Likewise, node $\nu$ can observe the random processes $Y(\check{e})$ for all $\check{e}$ entering node $\nu$. The linear combination at node $\nu$ can be represented as:

$$Y(e) = \sum_{\{i:\ k_i \text{ generated at } \nu\}} \alpha_{i,e} k_i + \sum_{\{\check{e}:\ \text{head}(\check{e})=\nu\}} \beta_{\check{e},e} Y(\check{e}) \tag{3.6}$$

where $k_i$ is the random process generated by node $\nu$, $\alpha_{i,e}$ is a coefficient generated over finite field, and $Y(\check{e})$ plus $\beta_{\check{e},e}$ are the signals/data entering node $\nu$ through the link $\check{e}$, $\text{head}(\check{e}) = \nu$, and the associated encoding vector, respectively. In a similar way, the linear combination of the received processes $Z(D, i)$ at the destination node $D$ is given by:

$$Z(D, i) = \sum_{\{\check{e}:\text{head}(\check{e})=D\}} \delta_{D_{i,\check{e}}} Y(\check{e}) \tag{3.7}$$

where $\delta_{D_{i,\check{e}}}$ are the coefficients. However, in the multicast case considering delay (as packets can arrive in different units of time $t$), memory in destination nodes is needed and the linear coding in Equations 3.6 and 3.7 become as:

$$Y_{t+1}(e) = \sum_{\{i:\ i \text{ generated at } \nu\}} \alpha_{i,e} k_{it} + \sum_{\{\check{e}:\text{head}(\check{e})=\nu\}} \beta_{\check{e},e} Y_t(\check{e}) \tag{3.8}$$

$$Z_{t+1}(D, i) = \sum_{\{\check{e}:\text{head}(\check{e})=D\}} \sum_{l=t-\sigma}^{t} \delta_{D_{i,\check{e}_{t-l}}} Y_l(\check{e}) \tag{3.9}$$

where $\sigma$ is the memory required at the destination and the coefficients $\alpha_{i,e}, \beta_{\check{e},e}, \delta_{D_{i,\check{e}}}$ are generated over $\mathbb{F}_{2^\ell}$.

In [GCSS14], a network model is proposed that takes into account the disjoint routes connecting nodes. Packets are divided into generations, each generation having $r$ packets. For a given encoding vector $\alpha_i = [\alpha_{i1}, \alpha_{i2}, ..., \alpha_{ir}]$ (generated randomly over $\mathbb{F}_{2^8}$) and a given input packet vector $k = [k_1, k_2, ..., k_r]$, the coded packet will be given by:

$$\omega_i = \sum_{j=1}^{r} \alpha_{ij} k_j \tag{3.10}$$

At the destination, the coded packets are combined and represented as a system of linear equations. The destination node must have also the coefficient matrix, to be able to decode the packets. With these two elements (linear equations and coefficient matrix), the destination applies Gauss-Jordan elimination, to put the decoding matrix in reduced row echelon form, and solves the linear equations. In this case, a set of $r$ packets (a given generation) is received together with the encoding vector $\alpha_i = \alpha_1, \alpha_2, ..., \alpha_r$ that was used to encode the $r$ packets. Consider also that a generic element of the decoding matrix $A$ can be represented as $A_{ij} = \alpha_{ij}$, where $i = 1, 2, ..., r$ and $j = 1, 2, ..., m$, where $m$ is the number of received packets in a given generation. Then, when the matrix has a full rank, ı.e rank$(A)= r = m$ in specified generation, the node can solve the linear equations and obtain all the original packets related to this generation. Thus, part of the source's data will be successfully decoded.

Figure 3.8 shows an example of how the just mentioned network coding method performs. In this figure, the source node $S$ has three packets $a$, $b$, and $c$, which are the data to be encoded and sent to the destination $G$. For this purpose, source $S$ encodes these packets (generating $k_1$, $k_2$, and $k_3$) and broadcasts the resulting coded packets. Nodes $E_1$, $E_2$, and $E_3$ receive the coded packets, encodes them again and forwards the recoded packets. More specifically:

- Node $E_1$ receives and encodes $k_1$ and $k_2$, creating the coded data $k_4$ and $k_5$
- Node $E_2$ receives and encodes $k_1$ and $k_3$, creating $k_6$ and $k_7$
- Node $E_3$ receives and encodes $k_1$, $k_2$, and $k_3$ to $k_8$, $k_9$, and $k_{10}$.

When the coded packets reach the gateway, redundant packets are discarded and the linearly independent packets are decoded.

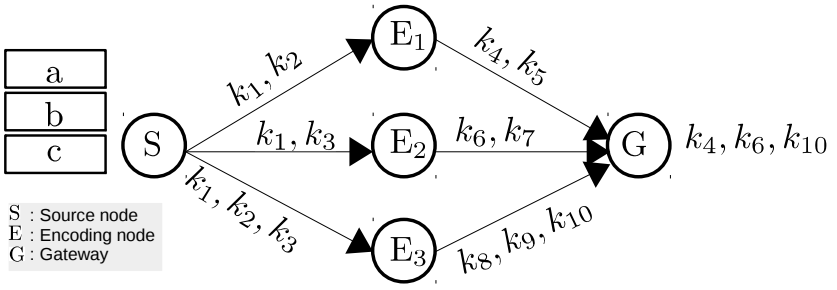



Figure 3.8: Network coding model from [GCSS14].

Adaptive network Coding (AdapCode) [HTAG08] employs network coding to provide reliable data transmission, by keeping the amount of traffic as minimized as possible. Linear combinations are applied to groups $r$ packets. At any node $n$, the coded packet $\omega$ is generated by:

$$\omega = \sum_{i=1}^{r} \alpha_{n,i} k_i \tag{3.11}$$

When the coded packet $\omega$ is ready, the node sends it together with the $r$ coefficients ($\alpha_{n,1}$, ..., $\alpha_{n,r}$), which are included in the packet's header. The procedures in AdapCode can be described as follows:

1. There are $n$ messages of fixed size in the system.

2. Messages are separated into a specific number of pages, where each page has a fixed number of $r$ messages, where $r$ should be a power of 2 ($r$ is set up to 8).

3. Considering one source, this source sends packets periodically and pauses several times to allow other nodes to spread the packets they receive.

4. Intermediate nodes receiving a packet will check if enough messages were received, by applying Gaussian elimination. This enables the node to decode all messages in the packet's page.

5. Intermediate nodes decide their coding scheme $\chi$, which is a factor of $r$, according to the number of its neighbors. Particularly, they send $\frac{r}{\chi}$ packets, where each packet is a linear combination of $\chi$ messages in a page, instead of $r$ messages.

6. The coefficients related to each linear combination are generated randomly from 0 to $1 - q$, where $q$ is a prime number set to 5.

In this method, if a certain node cannot decode the received packets, it will not be authorized to collect and resend these packets. The reason is to a avoid collisions and these packets, which the node could not decode, are likely to be heard by neighbors. Another technique to avoid collisions is to force nodes to wait for a random period of time, before sending their data.

According to [HTAG08], the most important task for a successful coding is to determine the number of packets that can be combined into one packet. This depends both on network and node densities, which describe how nodes are distributed in some area and how many neighbors a node has, respectively. The long-term number of neighbors, calculated by a node after decoding packets successfully, is given by:

$$\psi = \beta \times \psi + (1 - \beta) \times N_{\text{cur}} \tag{3.12}$$

where $N_{\text{cur}}$ is a counter representing the number of sources sending the packets, known by the node, and $\beta$ is a factor that should be small when the network is not stable (fails frequently), and large when the network is stable. This is a flexible way of determining the number of neighbours, as it adapts to topology changes.

To achieve reliability in network coding, it is important to determine the right number of packets to be aggregated into one packet, which is called the 'coding scheme' by the authors in [HTAG08]. Since each node will send one packet (linear combination of received ones) when an enough number of messages has been received then, when compared to traditional flooding where each node sends $\chi$ messages, with NC only $\frac{1}{\chi}$ will be sent.

In Cope [KRH+08], the coding layer is inserted between MAC and Internet Protocol (IP) layers. Cope uses three techniques to provide efficient NC: opportunistic listening, opportunistic coding, and learning neighbor's state. In opportunistic listening, nodes can hear all communications in their coverage area, and keep heard packets for some limited time (set to 0.5 seconds). In opportunistic coding, on the other hand, the method should determine which packets should be combined together to increase network throughput. In other words, any node should ensure that its destinations will receive the maximum number of native packets. The next rule is used to ensure that each next-hop can decode the received coded packet and extract its native packet:

> *Rule:* "To deliver $r$ packets, $k_1, ..., k_r$, to $r$ next-hops, $n_1, ..., n_r$, a node can XOR the $r$ packets together if each next-hop $n_i$ has all $r-1$ packets $k_j$ for $j \neq i$".

In learning neighbor state, any node in the network needs to know its neighbors' packets. This is achieved by a reception report, announced by each node to advertise its packets. However, the reception report sometimes does not reach neighbors, due to network congestion or delay. Therefore, Cope computes a delivery probability between each pair of nodes and uses this probability to predict which packets each neighbor has. Nevertheless, nodes may give (sometimes) a wrong guess of which

packets its neighbor has. In this case, the missing native packet should be retransmitted. When Cope applies its coding algorithm, it takes into account three issues:

- Never delay packets. When a channel is available, the node takes the first packet in its queue and checks if there are other packets to be encoded with it. In this case, it will make the encoding and broadcast the result to its neighbors. If encoding is not done, however, the node does not wait to receive packets. Instead, it sends the packet without any encoding.
- Divide packets based on their sizes, and encoding is done based on this division.
- Packets that will be forwarded to the same next-hop, or those generated in encoding node (current node), are not encoded together. In such cases, coded packets will not be decodable by the next-hop.

Cope gets the benefits from both unicast and broadcast modes in 802.11 MAC to employ the NC. It is known that in unicast the packets are sent to a particular receiver, which in turn replies with ACK if the transmission is successful. This can improve reliability but causes a low throughput. In broadcast, however, throughput increases but there are no acknowledgments, which leads to low reliability. Cope deals with this situation through pseudo-broadcast. In pseudo-broadcast, node deals with: *i*) link layer address; *ii*) XOR-header. The former contains a MAC address that is intended to one of the destinations as unicast. The later contains all next hops of the packet as broadcast. Thus, when a node receives a packet and it is not the intended destination for this packet, it checks the XOR-header to verify whether it is a next hop. If this is the case, it performs the required processing, otherwise the packet is stored for a while in case any neighbor needs it. Pseudo-broadcast, in Cope, is explained in Figure 3.9.

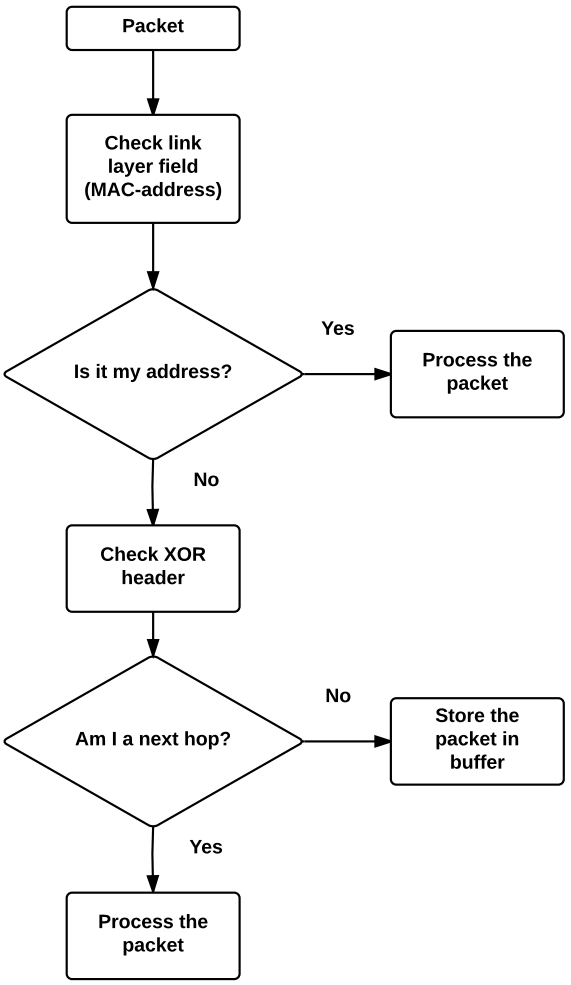



Figure 3.9: Pseudo-broadcast in Cope method, applied at intermediate nodes [KRH+08].

In many network coding works, some centralized knowledge of the network topology is required to compute the broadcast capacity and encoding/decoding functions. However, in [CWJ03], a packet format is proposed to avoid such centralized knowledge.

## 3.5 Summary

This chapter discusses NC principles and how traditional and constrained networks can both benefit from such concept. First, the chapter defines what is NC and its advantages over traditional routing. NC can be binary or random linear NC. In both types there are encoding and decoding processes, which are also presented. The chapter ends with related research work on NC.

# 4

# FEDERATION OF WIRELESS SENSOR NETWORKS

SMART systems appear everywhere and this is expected to increase in the near future [JSHG15]. Such trend, combined with the potential for the IoT to connect tens of billions of objects to the internet, will allow geographically distributed systems to collaborate and smart services to emerge. In such scenarios, P2P networks may have a key role allowing federated peers to collaborate and improve their businesses.

Although cloud storage solutions have emerged, extra storage space always requires extra physical disks and processing, meaning that connecting hundreds of millions of sensors to the cloud will be extremely demanding. Besides this, such places are known as storage places meaning that they are exposed to attacks. P2P approaches

can overcome both these issues because participating peers/proxies also contribute to storage and processing, peers communicate directly posing no burden on a specific set of servers, and bandwidth bottlenecks are avoided. Encrypted communication tunnels can also be built between peers.

In the context of federated constrained systems/networks some open standards become important. Within Internet Engineering Task Force (IETF), the Constrained RESTful Environments (CoRE) working group has focused on the development of CoAP, a data messaging/transfer protocol providing a request/response interaction model between application endpoints [SHB14, CSM+16]. More recently, CoAP Usage for RELOAD was proposed [JLVMC15]. RELOAD is a base protocol that provides a generic self-organizing P2P overlay network service, and uses pluggable application layers, called Usages, which allow RELOAD to fit any purpose [JBLR+14]. Another example of a RELOAD Usage is the one defined for the Session Initiation Protocol (SIP) in [JLR+16]. In a RELOAD/CoAP architecture the proxy nodes form a distributed P2P overlay network to announce resources, allowing clients to discover them. More specifically, the overlay can be used: as a lookup service, to store existing resources, and as cache for data.

Many applications (e.g., industrial, environmental) require smart systems to communicate using wireless constrained networks. Such constrained environments usually have energy efficiency and end-to-end packet error rate concerns, which are competing goals (e.g., a packet may be sent through multiple paths to reduce error rate but this increases energy consumption). An elegant way of achieving a balance between these two goals is to use NC [KAAF13]. In this chapter, an extension of CoAP Usage is proposed so that NC based constrained networks can benefit from RELOAD/CoAP P2P distributed storage. That is, although packets will travel from sources toward sinks/gateways, their final destination will be the P2P overlay where

storage is done. When compared with the work in [JLVMC15], a new data Kind structure is proposed for coded data and encoding vectors to be stored. For the decoding process, required when original data packets are not received, a decoding service is required at the P2P overlay network. Such architecture allows constrained networks to reduce packet error rate while benefiting from an effective distributed solution for data storage. This will serve as a basis for the proposal of mathematical models and algorithms that determine the most effective routing trees, for packet forwarding toward sink/gateway nodes, and adequate placement of coding nodes, as will be detailed in the following chapters.

The remainder of this chapter is organized as follows. In Section 4.1 the relevant standards for the federation of constrained networks are detailed. In Section 4.2 related work is discussed. Section 4.3 presents the architecture being proposed, together with the proposed extension of CoAP Usage, and Section 4.4 presents a summary of this chapter.

**Contributions:**

- An architecture for the federation of network coding based WSNs is proposed. This ensures the recovery of lost packets even when the other non-lost packets (coded or original packets) are forwarded towards different gateways.

**Publications:**

- Eman Al-Hawri, Noelia Correia and Alvaro Barradas, "RELOAD/CoAP P2P Overlays for Network Coding Based Constrained Environments". *In Doctoral Conference on Computing, Electrical and Industrial Systems*, pages 307–315, Springer, 2017.

## 4.1 Federation Related Standards

### 4.1.1 Constrained Application Protocol

CoAP is an application level protocol designed for constrained devices with limited power, processing, memory, and bandwidth (e.g., sensors and actuators) [MBL12]. It is a specialized web transfer protocol that realizes the REpresentational State Transfer (REST) architecture for constrained nodes, and provides a request/response interaction model between application endpoints. The `coap://` and `coaps://` URI schemes are used to identify CoAP resources and provide a mean of locating the resource. For discovery, a default entry point `/.well-known` is defined and the internet media type `application/link-format` is assigned for CoRE Link Format payloads [MC14]. Figure 4.1 shows both the web stack in IoT, including CoAP, and the web stack in the normal internet [LC15].

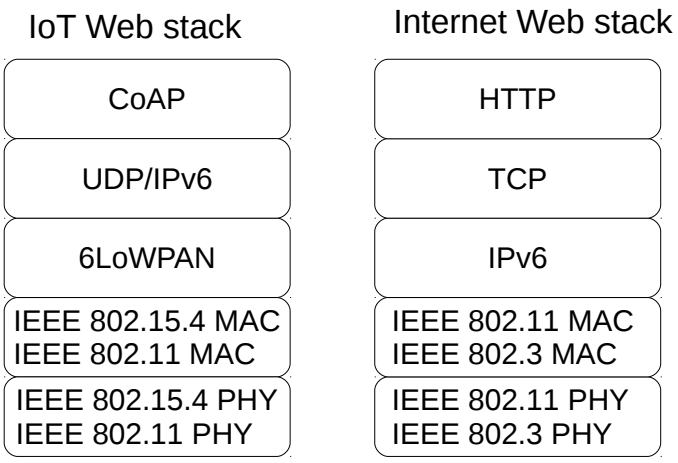



Figure 4.1: IoT web stack vs internet web stack.

CoAP can be seen as having two main layers: request/response layer and messaging layer. The request/response layer deals with the interaction between the client and

the server, in which the client requests a service and gets the response from the server. This is similar to the client/server model in HyperText Transfer Protocol (HTTP) and GET, PUT, POST and DELETE methods are also used. On the other hand, the messaging layer deals with the transmission of messages, using User Datagram Protocol (UDP), and the asynchronous nature of the interactions.

A message in CoAP can be a confirmable (CON), non confirmable (NON), or reset (RST) message. CON message is used when transmission requires reliability, meaning that the server should respond with acknowledgment when receiving a request. Contrarily, when there is no need for reliability, the NON message should be used. However, when the server can not process the client's request, it sends a RST message. Figure 4.2 shows CoAP's sub-layers [CKP15].

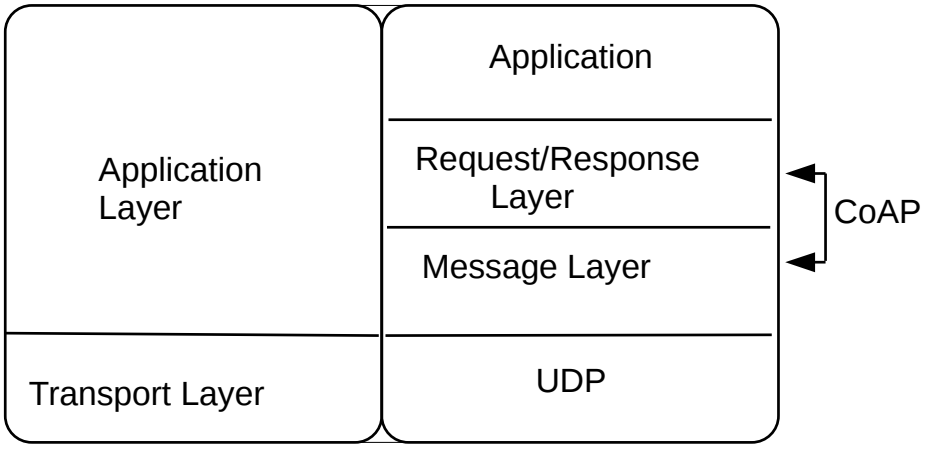



Figure 4.2: CoAP sub layers [CKP15].

### 4.1.2 RELOAD

RELOAD is a generic P2P framework for the management of self-organizing P2P overlay networks, and pluggable application layers (Usages) can be defined so that

it fits specific purposes [JBLR+14]. This protocol has important features, which include security, Usage model, Network Address Translation (NAT) traversal, optimized routing and overlay algorithm extension capability. RELOAD allows the definition of new application Usages that define data types and rules for their use. Thus, RELOAD can have multiple purposes and can be used with new applications through a simple documentation process that supplies the details for each application [JBLR+14]. Recently, a CoAP Usage has been defined in [JLVMC15], which defines a pluggable application layer for constrained networks, allowing a P2P overlay network to be built where sensor networks store their resources and/or data measurements.

## 4.2 Related Work

In [CSDC11], an HTTP over IP network that is integrated with a CoAP wireless sensor network over 6LoWPAN is provided. In this approach, a gateway is used to connect CoAP based WSNs with HTTP based web applications. This gateway consists of three main blocks: web server, database, and CoAP client, as shown in Figure 4.3. Accordingly, the interaction between these blocks starts when the CoAP client stores the obtained data from sensor nodes in the database, for this to be available to a web server. Then, when the web server has a request to obtain sensor data, it retrieves the demanded historical data from the database. In this case, there is no need to communicate with the CoAP client. However, when the web server needs up-to-date data from the WSN, it must communicate with the intended CoAP client for this to retrieve the required data.

The approach previously discussed, from [CSDC11], can be used for local implementations but it is not scalable enough to be applied in federated network scenarios. In

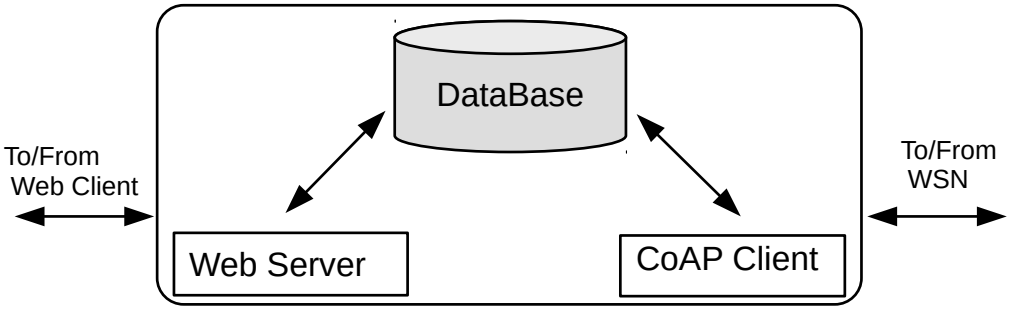



Figure 4.3: Gateway main blocks [CSDC11].

[MBL12], a machine-to-machine communication architecture is proposed to federate distributed WSNs. In this architecture, heterogeneous WSNs can be connected to each other and to the internet as well. Both CoAP and RELOAD protocols are used in the architecture. The proposed architecture contains different types of nodes and different connection technologies, as shown in Figure 4.4. Nodes categories are listed as follows:

- Local Nodes (LNs): Sensors and actuators using 6LoWPAN protocol and radio technology for communication. Work as CoAP endpoints and have resources that are to be stored/discovered in the RELOAD overlay. Since these nodes have limited capacity, not being capable of act as RELOAD nodes, proxies are used as intermediates.

- Non-IP LNs: Legacy sensors and actuators. These nodes also have resources that need to be discovered in the RELOAD overlay and, similarly, proxies are used to register these nodes' resources in the RELOAD overlay network. The difference between these nodes and the previous ones is that these nodes are not using IP-connectivity, and ZigBee protocol is used instead. For each legacy node, the proxy assigns a RELOAD node-ID and CoAP Uniform Resource Identifier (URI), and a RELOAD resource-ID is assigned for each resource hosted by these kind of nodes.

- Wide area Nodes (WNs): Nodes that use cellular technologies for communication because they are geographically distributed. Additionally, they are peers in the RELOAD overlay network, and sometimes work as clients to reduce resources dissipation.

- Proxy Nodes (PNs): Nodes that are placed at the edge of WSNs to connect sensor nodes to the internet. For every WSN, a specific domain in the CoAP URI is provided to differentiate it from other WSNs.

- Gateway Nodes (GNs): These nodes work as peers in the RELOAD overlay network and act as HTTP/CoAP proxies to take care of the translation process between HTTP and CoAP protocols.

- Monitoring and Controlling Nodes (MCNs): Are HTTP clients that access to WSNs resources using a CoAP REST interface.

The architecture has various features, namely: *i*) it provides a federation of WSNs; *ii*) since WSNs are connected by a RELOAD overlay network, they can obtain each other's resources; *iii*) using CoAP and GNs nodes allows WSNs to integrate the web, making their resources available to web clients.

Another work, [AL07], proposes to introduce peer-to-peer overlay sensor networks with no need for proxies or infrastructure support. In this work, each group of nodes is organized as a ring, where each ring contains at least one master node (see Figure 4.5). These masters are nodes with more powerful resources. Each master node can handle the information of its slaves (limited nodes in the same ring) and $\mathcal{O}\log(N)$ other masters. Thus, all the queries are processed in a distributed way with a bound of $\mathcal{O}\log(N)$ messages. The idea is that when a client sends a query, the master node starts searching in its keys (refers to the data stored in its ring) to response to the query. If the lookup operation fails it means that the queried information is

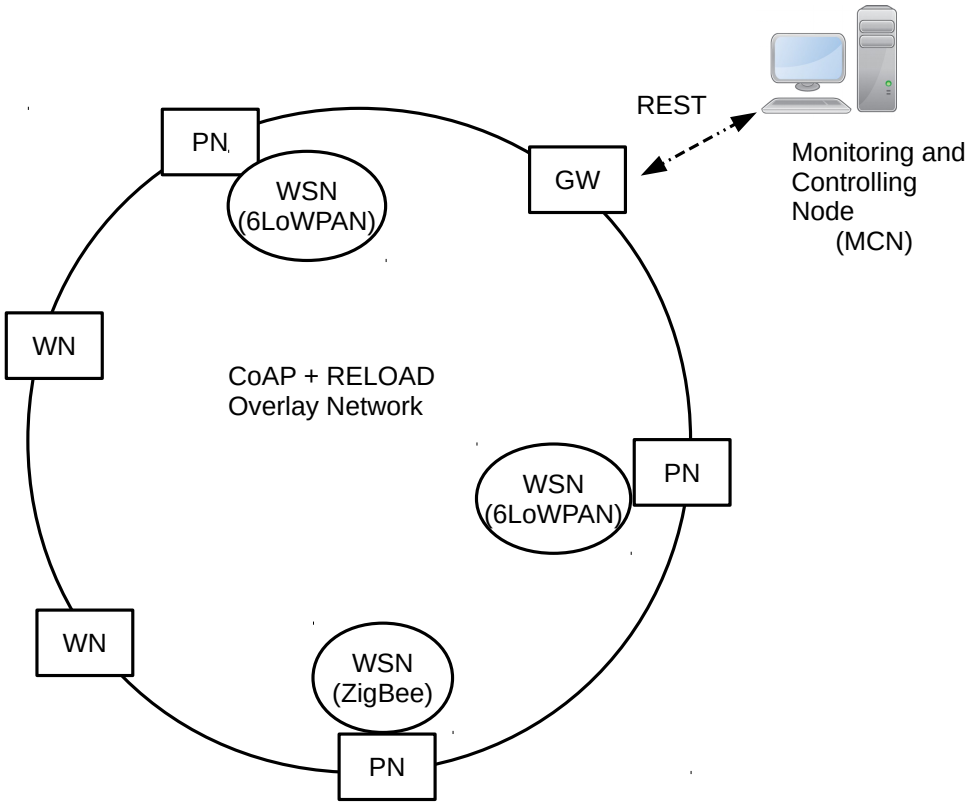



Figure 4.4: CoAP and RELOAD architecture for WSNs federation [MBL12].

not in the master node nor in its slaves. As a result, the master checks if any of the $\mathcal{O}\log(N)$ masters have the information being requested. If this is the case, the query will be forwarded to the master storing the data, otherwise the query will be transmitted to the master closer to the target.

A P2P network using Kad protocol, from [SDAT14], integrated with CoAP protocol is proposed in [SDA+15]. In this approach, called CoHaRT, the CoAP protocol is located at the top of HaRTKad protocol stack, as shown in Figure 4.6. In CoHaRT, the server first divides the payload into blocks. Then, when it receives a request, it sends the first block to the client. Also, CoAP block option (in response's header) is set to inform the client that only a fraction from the actual payload has been sent. This option contains the block number, block size, and whether still more blocks

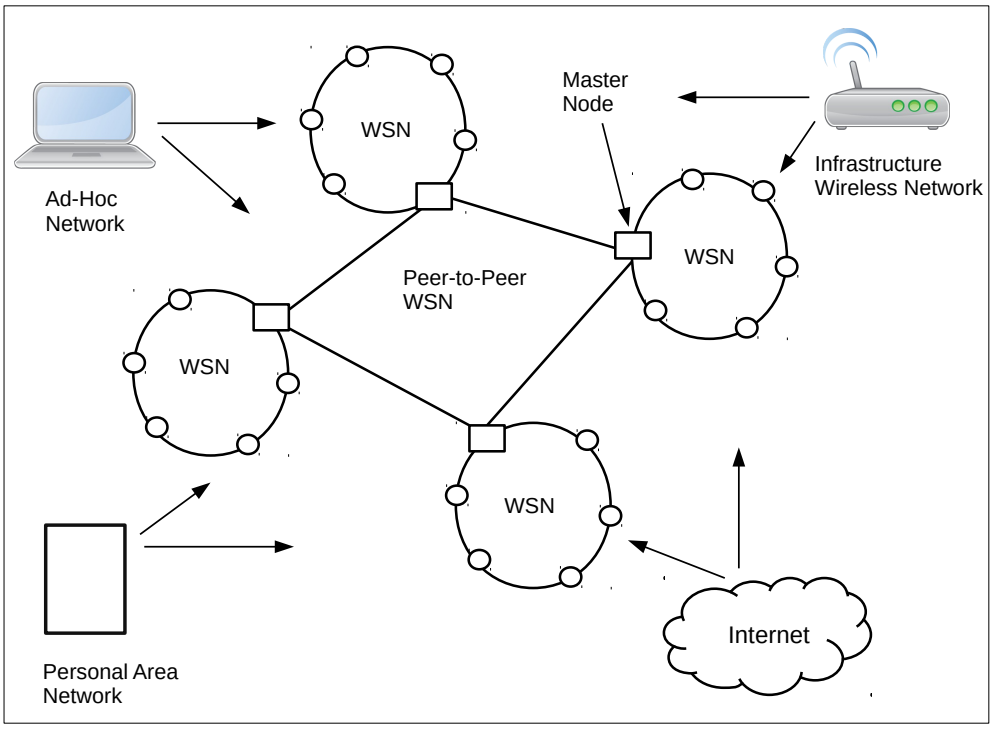



Figure 4.5: P2P based wireless sensor network [AL07].

are to be sent or not. When a client receives the block and sees the block option, it stores the received block and sends a request for the next block.

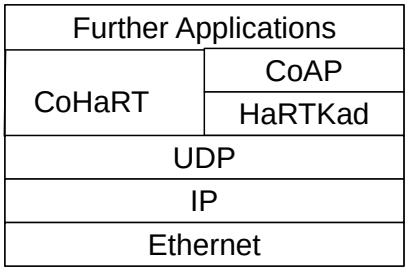



Figure 4.6: CoHaRT protocol stack [SDA+15].

Since smart objects became a significant part of the IoT, many applications try to control and manage the interactions between such objects and their resources. The method in [IHVdA+14] aims to achieve such objective using CoAP protocol. In this method, CoAP devices (smart objects) can be considered as embedded web servers, identified by URIs, with resources demanded by clients. Each group of

resources is called an entity, and each resource in a group is called an entity member. The component managing the entities is called the Entity Manager (EM). Such component is responsible for the creation, management and delete of entities. EM also manages the communication between the entities and the internet user. In this case, the user can communicate with the EM to create new entities or to determine how the existing ones should behave. An advantage of using entities is that their implementation does not require to be placed in a specific device. Rather, it can be located on any CoAP server. Moreover, multiple entity managers can be placed on several places on the network to avoid failure. Figures 4.7 and Figure 4.8 depict the main parts of the system and entity manager, respectively.

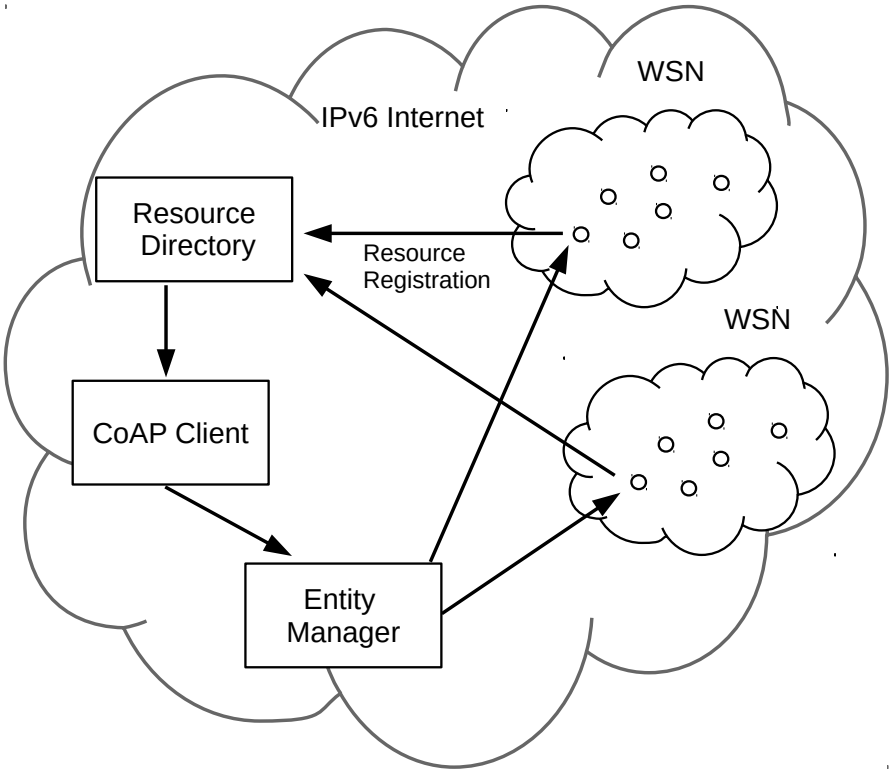



Figure 4.7: Main components and their interactions [IHVdA+14].

When the entity manager receives requests from users, it sends these requests, using CoAP, to the specified sensors. Then it gets the responses, aggregates them in a

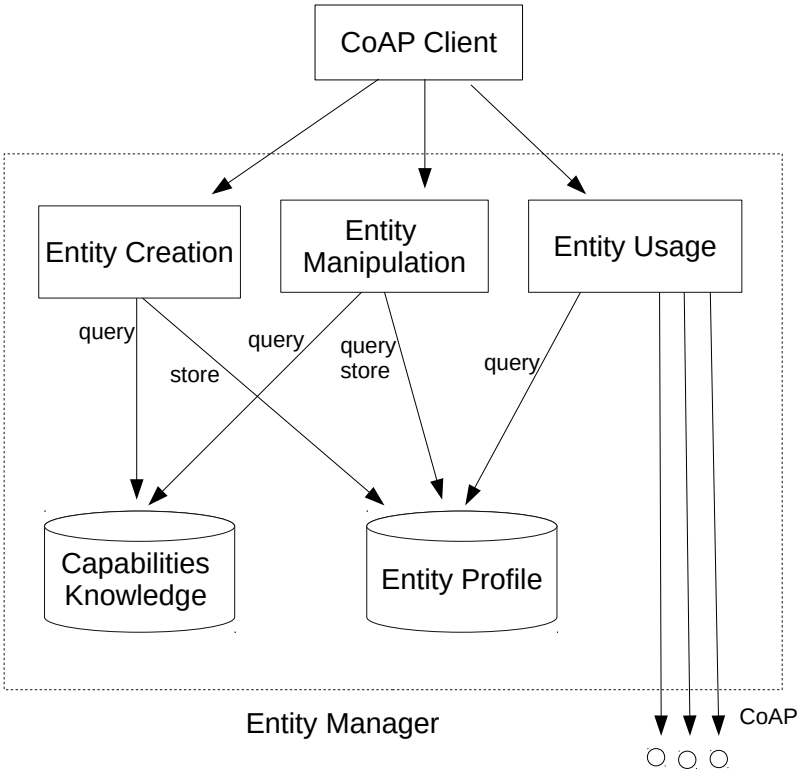



Figure 4.8: Entity manager components [IHVdA+14].

way that satisfies the user requests, and sends it back to the user. As shown in Figure 4.8, two databases are used: *i*) entity database that stores entity profiles, where the profile of each entity characterizes its behavior; *ii*) capability knowledge database (optional) that contains the rules and specifications to ensure sensor's ability of achieving user requests. In this method, the user has two choices when communicating with the system. Either to connect to resources' directory, to know which resources are available, or to connect to the entity manager. The later allows the user to create a new entity and decide how this entity will behave. This can be done when the user issues a CoAP POST request and sends it to the entity manager. Thereafter, the entity manager carries out the following tasks:

- Create the entity and assign a unique URI to it.

- Store the entity in the entity database.
- Validate the entity. In particular, check if the required resources (demanded by the user) are available and can be used.
- Inform the user about the entity URI, to be able to access and use the entity.
- Register the entity resources in the resource directory for these to be available in the lookup process.
- Inform the user about errors that need to be fixed, when the entity did not pass the validation process.

## 4.3 Proposed P2P Architecture

This section explains the proposed architecture and extention to CoAP Usage data Kind, which allows NC based constrained wireless networks to benefit from P2P overlays. A decoding service, to be provided by the P2P overlay, becomes necessary.

### 4.3.1 Overall Architecture

At the wireless section the nodes are organized in a way that packet flowing toward the sink/proxy node is ensured. Figure 4.9 shows the proposed architecture. Depending on factors like location, energy and functionality, nodes can be of type:

- Sensing: Data sources that forward their packets according to some routing table. Nodes can turn to sleep mode to reduce energy consumption.
- Encoding: Nodes able to perform encoding. Besides forwarding original packets, extra coded packets are created by linearly combining received packets

and packets overheard from neighbors. It may be required for them to have more energy, memory and processing capabilities than others.

- Relay: Work just as relays and forward any received packets, either original or coded, toward the sink. Perform no encoding/decoding operations.
- Sink: Destination of data. It is connected to the RELOAD overlay network where packets can be stored and fetched by others.

It is assumed that encoding nodes propagate linear combinations of original packets and packets overheard from neighbors, besides original packets. Sinks/gateways store original and coded packets at the overlay. A specific scenario is discussed in the following sections.

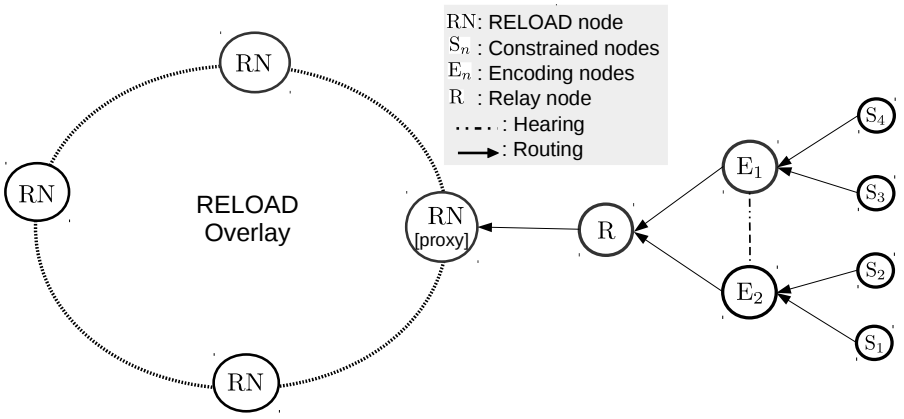



Figure 4.9: Proposed P2P architecture.

### 4.3.2 Storing Data and Standard Extensions Needed

#### 4.3.2.1 Data Kinds

The data Kinds defined by CoAP Usage include the CoAP-REGISTRATION, to announce available resources, and CoAP-CACHING for the storage of sensor measurements [JLVMC15]. Although the CoAP protocol itself supports the use of proxies, for the caching of sensor measurements and consequent reduction of both the response time and network bandwidth consumption (see [SHB14]), the additional caching mechanism of CoAP Usage allows such data to be stored in the P2P overlay, improving even more the response time and network bandwidth utilization because proxies will not become bottleneck points and a distributed access to P2P resources is available.

Regarding CoAP-CACHING, the possibility of storing proxy data and sensor data are both considered. That is, the proxy data structure supports data from multiple sensor nodes, forwarding their data to a proxy, while the sensor data structure stores measurements of a specific sensor node. Listing 4.1 shows the data stored for proxy with NodeID `9996172` and URI `coap://overlay-1.com/proxy-1`, where `mt` is the measurement time of data, `ttl` is the time-to-live, and `v` is the measurement value.

```
Resource-ID=h("coap://overlay-1.com/proxy-1/")
KEY=9996172,
VALUE=[
  </sensor-1/>; {mt=100000;ttl=10000;v=38};
                {mt=100055;ttl=456990;v=42};
                {mt=134000;ttl=234000;v=30}
  </sensor-2/>; {mt=100000;ttl=10000;v=40};
                {mt=400000;ttl=17000;v=25}
     ]
```

Listing 4.1: Storage of proxy data.

The `h(...)` is the hash over the URI, which is used as key for storage in the overlay. The Listing 4.2 shows a direct storage of data from the sensor with URI `coap://overlay-1.com/proxy-1/sensor-1`. The proxy with NodeID `9996172` is the one responsible for sensor-1.

```
Resource-ID=h("coap://overlay-1.com/proxy-1/sensor-1")
KEY=9996172,
VALUE=[
  {mt=100000;ttl=10000;v=38};
  {mt=100055;ttl=456990;v=42};
  {mt=134000;ttl=234000;v=30}
      ]
```

Listing 4.2: Storage of sensor data.

#### 4.3.2.2 Requirements Regarding Network Coding

NC based constrained networks forward original and coded packets toward the proxies participating as peers in the P2P overlay network. Therefore, for NC based constrained networks to benefit from such a distributed storage, the data Kind must also allow the storage of:

- Encoding vectors: Required for the decoding. For deterministic NC these may be stored at the overlay at peer registration time (done once). A proxy stores encoding vectors of sensors for which it is responsible.
- Coded packets: Since decoding may not be done if a sufficient number of coded packets has not arrived, it becomes necessary to ensure that coded packets are stored.

Therefore, the current CoAP Usage data Kind must be extended. Listing 4.3 shows an example when using such extended data Kind. The first value regards to the

current data Kind structure and second value includes predefined encoding vectors and arriving coded data.

```
Resource-ID=h("coap://overlay-1.com/proxy-1/")
KEY=9996172,
VALUE=[
  </sensor-1/>; {mt=100000;ttl=10000;v=38};
                {mt=100055;ttl=456990;v=42};
                {mt=134000;ttl=234000;v=30}
  </sensor-2/>; {mt=100000;ttl=10000;v=40};
                {mt=400000;ttl=17000;v=25}]
ENCODING=[
   {enVector1;enVector2;...};
   {1stcodedData;2ndcodedData;...}]
```

Listing 4.3: Sensor data using extended data Kind.

#### 4.3.2.3 Storing and Fetching of Data

Figure 4.10 illustrates how storage of packets would be done. In this example, encoding nodes $E_1$ and $E_2$ receive packets from their children, namely $E_1$ receives $k_1$ and $k_2$, and $E_2$ receives $k_3$ and $k_4$. Moreover, these encoding nodes can hear each other. In this case, the linear combinations of packets from children plus packets they have heard is done $(k_1 + k_2 + k_3 + k_4)$. When a single packet gets lost, it can be retrieved when the decoding is performed. Please note that encoding can be performed at different areas of the wireless section, and nodes may forward packets toward different proxies/gateways. Therefore, decoding might require coded packets stored by different proxies. Adequate ways of building routing trees, so that the effectiveness of encoding increases, are proposed in the following chapters.

When a client needs to fetch sensor data from the overlay network, such data can be directly available (original data) or there might be a need to perform the decoding

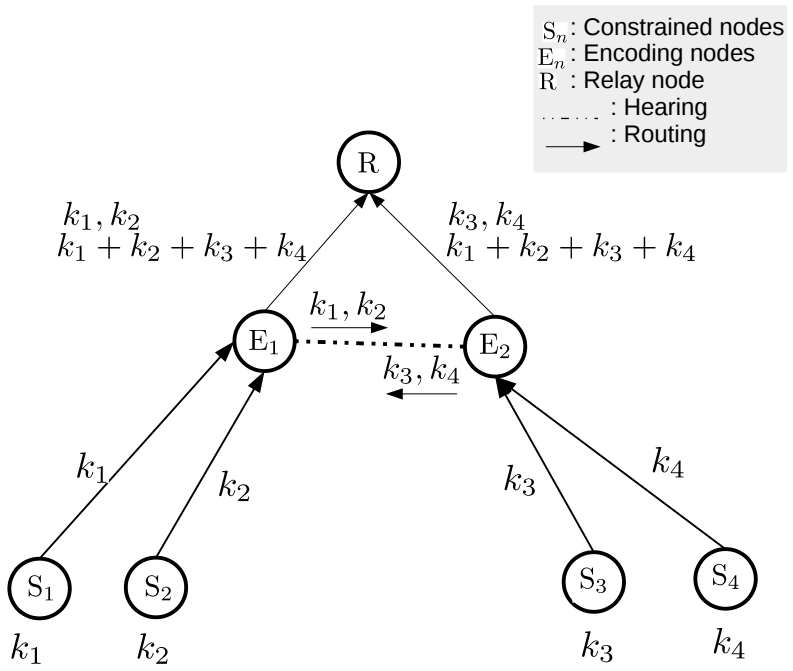



Figure 4.10: Example of network coding at constrained network.

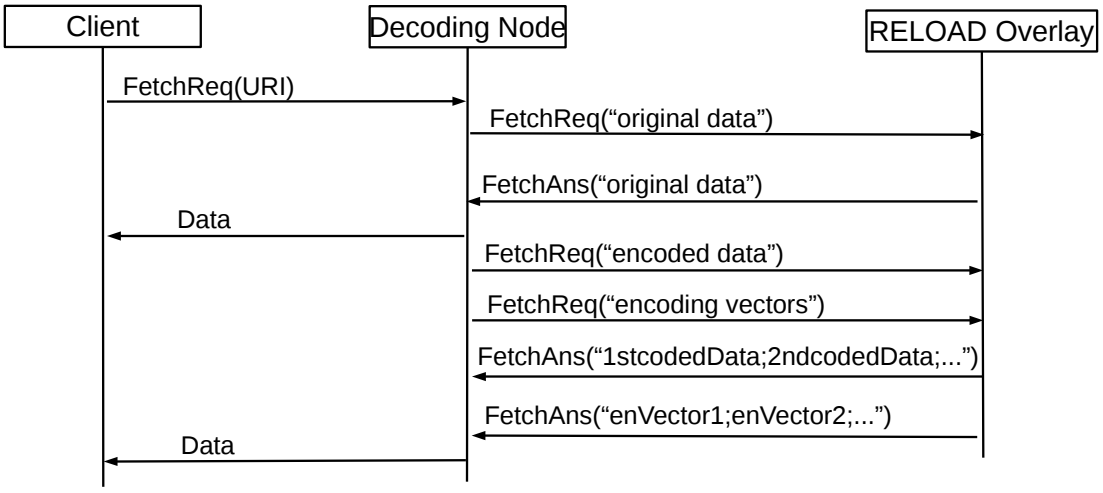



Figure 4.11: Decoding at RELOAD/CoAP P2P overlay network.

process, using stored coded data and encoding vectors, in order to extract the original data. This decoding would be accomplished by RELOAD nodes able to provide such decoding service. Figure 4.11 illustrates a scenario where the decoding service is requested. A fetch request by some client may involve fetching original data and

coded data, which also requires fetching the encoding vectors, so that the highest amount of sensor measurements is returned to the client.

Since peers face the problem of finding the peers providing the decoding service, the Recursive Distributed Rendezvous (ReDiR) service discovery mechanism can be used [MC14]. ReDiR ensures that the load is distributed among the nodes providing the service.

#### 4.3.2.4 CoAP Option

CoAP allows options to be included in a message [SHB14]. Each option instance specifies the option number, length of the option value, and the option value itself. For the proxy to be able to differentiate the type of payload at CoAP packets, which may be of original or coded type, it is necessary to include the corresponding CoAP option number.

## 4.4 Summary

This chapter discusses how to federate NC based constrained networks through RELOAD and CoAP Usage. The proposal is for NC based constrained networks to be able to store encoding vectors and coded data at the P2P overlay network, which requires an extension of CoAP Usage data Kind. This provides a scalable and efficient way of discovering cached sensor data of geographically dispersed sensor networks, allowing large scale applications to emerge, while taking advantage of NC at the wireless section.

# 5

# NETWORK CODING FOR RELIABLE WSNS

In a near future, sensing devices are expected to integrate many applications (e.g., health care, environmental) and in many cases these will have to communicate using WSN [CSM+16]. Such constrained networks usually have energy efficiency and end-to-end packet error rate concerns, which are competing goals (e.g., a packet may be sent through multiple paths to reduce packet error rate but this increases energy consumption). NC can be very useful in such scenarios, allowing a balance between these two goals to be achieved in an elegant way [KAAF13].

In this chapter, a mathematical model and a heuristic algorithm are proposed to plan for the best placement of encoding nodes while ensuring reliability. These approaches are also able to address scenarios where sensor networks, using different

gateways, are federated. In this case a distributed storage system is required to ensure the recovery of packets when related coded packets arrive to the different gateways.

The proposed mathematical formulation is generic, allowing for the use of one or multiple gateways while considering any possible failure scenarios. The case of multiple gateways is relevant when sensor networks are federated, requiring a distributed storage system (P2P overlay) for the storage of data and coded packets. Besides having failure scenarios into account, one should privilege places where encoding nodes would generate more innovative coded packets (nodes with high degree of connectivity). An innovative coded packet is a packet that is linearly independent from the previously received ones, considering a specific generation of packets. These packets bring, therefore, new information that is useful in the decoding process. The adopted approach ensures this issue.

The remainder of this chapter is organized as follows. Section 5.1 discusses related research work. Section 5.2 details the assumed architecture, which can incorporate scenarios of multiple gateways. In Section 5.3, the mathematical formulation for the design of NC based reliable sensor networks is presented, along with a heuristic approach. Section 5.4 includes a performance analysis of the mathematical model and heuristic, and Section 5.5 provides the chapter summary.

**Contributions:**

- A mathematical model is developed to determine the optimal number of encoding nodes, and their location, under certain failure scenarios. Besides reducing cost and improving reliability, sensor networks end up having a higher performance in terms of delay because the overall number of network coding operations decreases;

- A heuristic algorithm is proposed for large-scale network scenarios.
- A comparison between the mathematical model, heuristic and SenseCode (from literature) is performed.

**Publications:**

- E. AL-Hawri, N. Correia and A. Barradas, "Design of Network Coding Based Reliable Sensor Networks", Ad Hoc Networks, Vol. 91, Elsevier, 2019.

## 5.1 Related Work

Regarding node placement in WSNs, [GKJ16], [MMC+17], [NJ14] and [MHXT10] are relevant works. In [GKJ16], the authors propose two methods for relay node placement that give $k$-connectivity to sensor nodes. One of the methods uses a genetic algorithm, while the other stands on a greedy approach. With a given number of potential positions, and number of targets (sensor nodes), both algorithms try to select the location for relay nodes so that the targets become $k$-connected. In [NJ14], an algorithm is proposed for placing the minimum number of relay nodes, working as cluster heads in a two-tier WSN. Full coverage and connectivity of the WSN, and communication cost minimization, are goals to be achieved. The algorithm is based on spiral sequence which is created for stationary sensor nodes to minimize the total number of relay nodes. In [MHXT10], the authors introduce a constrained relay node placement problem. The goal is to minimize the number of relay nodes, which can be located in pre-determined candidate locations.

NC was initially introduced in [ACLY00], and has caught the attention of many researchers. Most of these works try to introduce a good mechanism of coding,

focusing on specific parameters like lifetime, delay, bandwidth, and/or energy to improve the whole network performance.

In [GCSS14], a network model is proposed where nodes are connected by disjoint routes and NC is used to increase reliability. This approach is suitable for adhoc networks, where the network topology, packet loss, and node/link failures are difficult to predict, and no specific gateway/sink nodes is assumed. The authors in [AV11] propose a ARQ MAC protocol for control packets to coordinate a set of relay nodes, acting as helpers in bidirectional communication. These helper nodes apply NC technique (XOR method) to enhance the network performance. In all these works any node can be the source or destination, meaning that the approaches are more adequate for general wireless networks.

In SenseCode, [KAAF13], NC for many-to-one communication is addressed. That is, there are multiple sources sending data packets towards a gateway/sink using tree-based routing. Each node in SenseCode may deal with three kinds of messages: messages received from its children, messages overheard from its neighbors, and its own messages. To perform NC, the node generates linear combinations of these messages. Applying this technique ensures that the sink will be able to recover packets that have been dropped. SenseCode, to the best of our knowledge, is the first one presenting the design and implementation of a collection protocol that is suitable for sensor networks using NC. However, it is assumed that all nodes participate as encoding nodes, which is not a very cost-effective solution. As stated in [CTF10], NC operations increase the delay and node complexity and, typically, the best performance is not achieved when all nodes perform NC.

In [EzAEOL17, EzAEO17] a clustering approach is proposed that takes into account channel conditions and inter-node distance to decide adequate coding, decoding or control usage. The attractiveness of the framework resides in the capacity of

cluster heads to avoid the decoding process if it would not be necessary, and this decision is based on the channel state and distance. The mentioned works do not try to determine the adequate number of encoding nodes, and their placement, while ensuring energy efficiency and reliability, which is addressed here in our work. Our model is also adequate for multi-gateway scenarios, which is not considered by these authors.

Regarding the placement of NC nodes, addressed in this work, [CTF10] and [KCP14] are two relevant works. Authors in [CTF10] take care of such problem in the context of streaming overlays. The article discusses how to decrease the delay of packet delivery while selecting just a few number of nodes to be encoding nodes. Initially, the expected number of duplicate packets is calculated for each node in the network, helping to select the nodes performing the encoding process. Two algorithms are proposed for two different cases: *i*) when a central node knows the whole network information; *ii*) when nodes have just a local information knowledge. The results for both algorithms show maximum profit while minimizing the number of encoding nodes. The approach in [KCP14], called Centrality based Network Coding Node Selection (CNCNS), minimizes the number of encoding nodes, leading to network throughput increase. Nodes at the center of dense neighborhoods are chosen to become encoding nodes, as network throughput is more likely to improve because such nodes can generate more innovative packets.

The problem addressed in this chapter is different from the previous works due to the following:

- NC is applied for reliability, similarly to SenseCode (in [KAAF13]), but here the number of nodes performing encoding is minimized, avoiding unnecessary overhead to some nodes. The optimal location of encoding nodes is also determined.

- The adopted approach takes into account many independent failure scenarios, each leading to one or more inoperable links, which is more realistic than considering that all links can equally fail.

- The possibility for WSNs to be federated is considered. This has the following advantages: *i*) sensor nodes can perform encoding while leaving the decoding processing burden to the gateway or interested users, saving energy; *ii*) a distributed storage system to store original and coded data, together with encoding vectors, allows coded packets from encoding nodes at border zones (with neighbours sending their packets to other gateways) to be useful for the decoding process, improving efficiency.

## 5.2 Architecture

### 5.2.1 Network Coding

Instead of simply relaying the received data packets, an encoding node generates new packets by combining received packets with its own packets [FLBW06]. This approach can be used to improve the throughput, efficiency, scalability, resilience to attacks and eavesdropping in networks, as discussed in detail in Chapter 3. Regarding the decoding process two possibilities exist:

- Packets carry network encoding coefficients, allowing encoding nodes to decode packets first and then make further linear combinations. That is, the encoding vector must be appended to the packet. This may constitute a significant fraction of the payload if the original packet is short, which is typical in WSNs [KAAF13].

- Network encoding coefficients are known at the gateways or stored at a P2P distributed storage system, as discussed in the following section and assumed in this work. Re-encoding can be performed by inner network encoding nodes, without a previous decoding, because recursive decoding can always be performed at the end. Although this is more suitable for static configurations, a node-to-gateway registration protocol (for encoding vector set up) may allow its implementation in more dynamic environments. Scalability is not an issue if a P2P distributed storage system is used because: *i*) in P2P overlays, the load per peer/gateway is independent from the total number of participating peers; *ii*) in WSNs, the number of nodes being served by a gateway is usually kept low in order not to increase the number of hops (towards the gateway) and congestion near the gateway.

### 5.2.2 RELOAD Overlay

In network coding based sensor networks, encoding nodes may end up listening to packets that will not be forwarded to the same gateway, if multiple gateways exist. In this case the recovery of lost packets may only be possible if the gateways share a storage system. For this reason the use of a RELOAD/CoAP based architecture is used. This architecture was detailed in Chapter 4. The operation of the nodes (see Figure 5.1) in this architecture will be:

- Source: Senses the field under observation and sends its data packets to either encoding or relay nodes.

- Encoder: Receives data packets from sources and relay neighbors, encodes the received packets with its own packets, and sends the coded packets towards

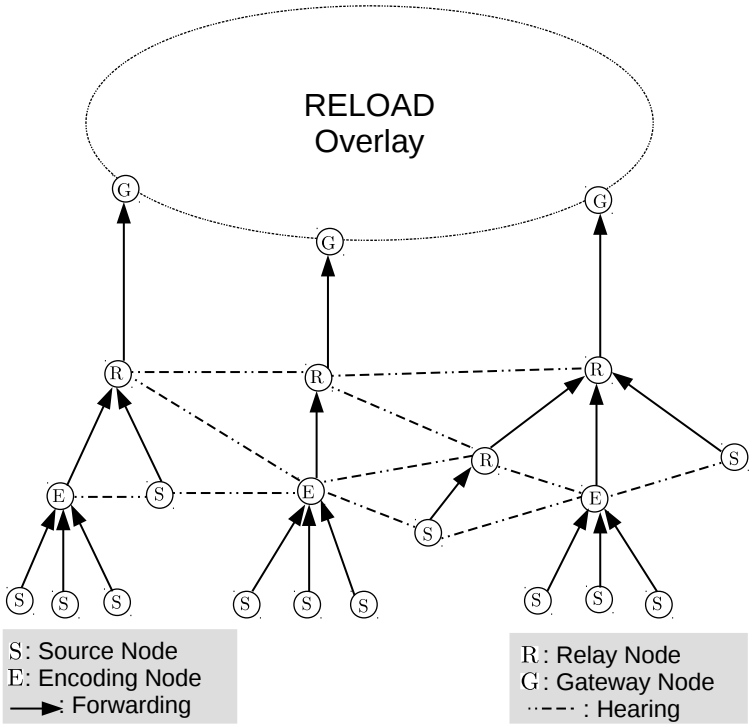



Figure 5.1: RELOAD/CoAP based architecture.

the gateway(s). Nodes of this kind can also send their original data packets, besides coded packets.

- Relay: Forwards the received packets (either coded or original) towards the gateway(s).

- Gateway: Receives the packets (either coded or original) from nodes at the wireless section, and stores them in the P2P overlay. Such data can then be fetched by interested users.

## 5.3 Encoding Node Location Problem

### 5.3.1 Problem Definition

NC induces delay and computational overhead, which will increase with the number of encoding nodes. For this reason the minimum number of encoding nodes, and their location, should be determined. This problem is defined as follows:

**Definition 5.1** (Encoding Node Location (ENL) Problem)**.** Given a constrained sensor network graph $G(\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ are the nodes and $\mathcal{L}$ are the wireless communication channels (links), and given a set of independent failure scenarios $\mathcal{F}$, where failure scenario $f \in \mathcal{F}$ implies that one or more links are down, determine which nodes of $\mathcal{N}$ should perform NC operations, together with packet flowing towards a gateway, so that any lost packet can be recovered through decoding operations.

In other words, failure scenarios reflect critical communication channels. Taking these into account will significantly reduce the end-to-end packet error rate (fraction of messages generated by the sources that are not successfully communicated to the destination).

### 5.3.2 Mathematical Formulation

Having the previous definition in mind, subset $\mathcal{G} \subset \mathcal{N}$ is used to denote the gateway/sink nodes, meaning that $\mathcal{N} \backslash \mathcal{G}$ ends up including just inner network nodes. A link $l \in \mathcal{L}$ between nodes $n_i$ and $n_j$ basically means that $n_i$ and $n_j$ are within the range (coverage area) of each other, i.e., they can hear/reach each other.

The set of failure scenarios is denoted by $\mathcal{F}$, where failure scenario $f \in \mathcal{F}$ includes one or more links that may be inoperable at the same time. Failure scenarios are independent. That is, they do not occur simultaneously. The links included in a failure scenario $f$ are denoted by $\mathcal{L}^f$.

As devices can have different capabilities (multiple classes of devices are envisaged by [BEK14]) only a subset of $\mathcal{N}$ is expected to be able to act as encoding node (high processing capability required). Such set of encoding capable nodes is denoted by $\mathcal{N}^{\mathrm{e}}$, $\mathcal{N}^{\mathrm{e}} \subseteq \mathcal{N}$. An encoding node can be seen as providing reliability to the nodes it can hear.

Path trees are assumed for data to flow, each tree being rooted at a gateway, as in Figure 5.1. The number of hops from data sources towards gateways is limited to $H$. The variables of the problem are the following:

| | |
|---|---|
| $\vartheta^{n_i}$ | One if node $n_i \in \mathcal{N}^{\mathrm{e}}$ is chosen to act as encoding node; zero otherwise. |
| $\delta_l^s$ | One if source node $s \in \mathcal{N} \backslash \mathcal{G}$ uses link $l \in \mathcal{L}$ to send its data towards a gateway; zero otherwise. |
| $\gamma_l$ | One if link $l \in \mathcal{L}$ is used by a path tree; zero otherwise. |
| $\sigma_{l,f}^{l'}$ | One if link $l' \in \mathcal{L}$ is used for protection (data flowing in it will be linearly combined with other data at encoding nodes) of link $l \in \mathcal{L}^f$; zero otherwise. |
| $\beta_s^g$ | One if $g \in \mathcal{G}$ is the gateway for source $s \in \mathcal{N} \backslash \mathcal{G}$; zero otherwise. |
| $\eta_{l,f}^n$ | One if node $n \in \mathcal{N}$ is the last/destination node of the protection path for link $l \in \mathcal{L}^f$; zero otherwise. |

Regarding protection paths, their first node will be the source of the link they are protecting (failing link), their last/destination node will be a node with an operating

uplink (according to the failure scenario under consideration), and both intermediate and final nodes must be encoding nodes in order to ensure that data reaches the operating uplink. Note that protection paths are not used for rerouting, but for overhearing.

**– Objective Function**

$$\text{Minimize} \sum_{\{n_i \in \mathcal{N}^e\}} \vartheta^{n_i}. \tag{5.1}$$

This objective function minimizes the total number of nodes performing encoding. Such goal naturally leads to the selection of nodes having high reachability/overhearing degree, while considering their usefulness as reliability providers (contributing with encoding at any protection path step).

**– Trees for Data Flow**

For data to flow between constrained nodes and gateways, using tree-based routing, the following constraints must be fulfilled:

$$\sum_{\{l \in \mathcal{L}: src(l)=n\}} \delta_l^s - \sum_{\{l \in \mathcal{L}: dst(l)=n\}} \delta_l^s = \begin{cases} 1, & if\ n = s \\ -\beta_s^n, & if\ n \in \mathcal{G} \\ 0, & otherwise \end{cases}, \forall s \in \mathcal{N} \backslash \mathcal{G}, \forall n \in \mathcal{N}. \tag{5.2}$$

$$\sum_{g \in \mathcal{G}} \beta_s^g = 1, \forall s \in \mathcal{N} \backslash \mathcal{G}. \tag{5.3}$$

where $s \in \mathcal{N} \backslash \mathcal{G}$ is a data source, $src(l)$ is the source node of link $l$ and $dst(l)$ is its destination node. Since $\beta_s^n$ denotes if $n$ is serving as gateway for data source $s$, or not, these two sets of constraints ensure, for a specific $s$, the so-called flow conservation law towards a single gateway.

$$\gamma_l \geq \delta_l^s, \forall l \in \mathcal{L}, \forall s \in \mathcal{N} \backslash \mathcal{G}. \tag{5.4}$$

$$\sum_{\{s \in \mathcal{N} \backslash \mathcal{G}\}} \sum_{\{l' \in \mathcal{L} \backslash \{l\} : src(l') = src(l)\}} \delta_{l'}^s \leq (1 - \gamma_l) \times \Delta, \forall l \in \mathcal{L}. \tag{5.5}$$

These constraints ensure that trees are built for routing. More specifically, if a link is performing data forwarding, which is determined by constraints (5.4), then constraints (5.5) ensure that no other link with the same source can be used (node has a single parent node). The $\Delta$ represents a big value and can be set to $\Delta = |\mathcal{N}| \times |\mathcal{L}|$.

$$\sum_{\{l \in \mathcal{L}\}} \delta_l^s \leq H, \forall s \in \mathcal{N} \backslash \mathcal{G}. \tag{5.6}$$

These constraints limit the number of hops for any flow (tree depth is limited).

**– Applying Network Coding Upon Failure Scenario**

$$\sum_{\{l' \in \mathcal{L} \backslash \mathcal{L}^f : src(l') = n\}} \sigma_{l,f}^{l'} - \sum_{\{l' \in \mathcal{L} \backslash \mathcal{L}^f : dst(l') = n\}} \sigma_{l,f}^{l'} =$$

$$= \begin{cases} -1 + \sum_{\{l' \in \mathcal{L} \backslash \mathcal{L}^f : src(l') = n\}} \gamma_{l'}, & if\ n = src(l) \\ -\eta_{l,f}^n, & otherwise \end{cases}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \forall n \in \mathcal{N}. \tag{5.7}$$

For a specific failure scenario, these constraints force data flow through overhearing nodes forming the protection path. More specifically, for each $l \in \mathcal{L}^f$ an alternative path (including overhearing/encoding nodes) must be ensured from the source of $l$ until a node with an active link tree is reached. Note that neighbor nodes may have their uplinks down, if they fail too (simultaneous failure of more than one link is possible). This means that overhearing through one or multiple nodes may be required. An alternative path is built only if there is no active uplink for the source of the failing link, determined by $-1 + \sum_{\{l' \in \mathcal{L} \setminus \mathcal{L}^f : src(l') = n\}} \gamma_{l'}$. The $\eta^n_{l,f}$ variables are determined by the following set of constraints.

$$\sum_{\{n \in \mathcal{N}\}} \eta^n_{l,f} = 1 - \sum_{\{l' \in \mathcal{L} \setminus \mathcal{L}^f : src(l') = src(l)\}} \gamma_{l'}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f \tag{5.8}$$

That is, these constraints are necessary to determine if a destination node, for the protection path of link $l \in \mathcal{L}^f$, will exist. In this case $\sum_{\{n \in \mathcal{N}\}} \eta^n_{l,f}$ will be one, meaning that a node $n$ must be the final destination for the path protection being determined.

$$\eta^n_{l,f} \leq \sum_{\{l' \in \mathcal{L} \setminus \mathcal{L}^f : src(l') = n\}} \gamma_{l'}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \forall n \in \mathcal{N} \setminus \mathcal{G} \tag{5.9}$$

$$\eta^n_{l,f} \leq 1 - \gamma_{l'}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \forall n \in \mathcal{N}, l' : src(l') = n \wedge dst(l') = src(l) \tag{5.10}$$

Constraints (5.9) and (5.10) determine if $n$ can be the final destination node for the

protection path of link $l \in \mathcal{L}^f$, $\eta^n_{l,f} = 1$, or if it can only serve as an intermediate node, $\eta^n_{l,f} = 0$.

$$\vartheta^{n_i} \geq \sigma^{l'}_{l,f}, \forall n_i \in \mathcal{N}^{\mathrm{e}}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \forall l' \in l \backslash \mathcal{L}^f : d(l') = n_i. \tag{5.11}$$

These constraints set the $\vartheta^{n_i}$ variables, which indicate whether a node is acting as encoding node or not, according to the overhearing needs.

– **Binary Assignments**

$$\vartheta^{n_i}, \delta^s_l, \gamma_l, \beta^n_s, \sigma^{l'}_{l,f}, \eta^n_{l,f} \in \{0, 1\}. \tag{5.12}$$

The trees being built will adapt to failure scenarios and chosen encoding nodes.

### 5.3.3 Hardness of the Problem

**Theorem 5.2.** *The ENL problem is NP-hard.*

*Proof.* When considering a single failure scenario, constraints (5.7)-(5.10) come down to the minimum Steiner tree problem, which happens to be NP-hard [Kar72]. The minimum Steiner tree problem can be defined as follows: given an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with non-negative edge costs, find the tree of minimum cost that contains a given subset $\mathcal{T} \subseteq \mathcal{V}$ as terminal nodes. The nodes $\mathcal{V} \backslash \mathcal{T}$ are called Steiner nodes and can be used to build the Steiner tree.

For the just mentioned ENL subproblem to fit the minimum Steiner tree problem, an extra virtual node $v$ is inserted into the network topology. That is, $\mathcal{V} = \mathcal{N} \cup \{v\}$,

where $\mathcal{N}$ are the WSN nodes. Extra links from $v$ to each node in $\mathcal{N}\backslash\mathcal{S}$ are included, where $\mathcal{S} = \cup_{l\in\mathcal{L}^f} src(l)$ includes the sources of failed links for failure scenario $f \in \mathcal{F}$, and failed links in $\mathcal{L}^f$ are excluded. That is, $\mathcal{E} = \mathcal{L}\backslash\mathcal{L}^f \cup \{l = (v, n) : n \in \mathcal{N}\backslash\mathcal{S}\}$. Assuming $\mathcal{S}$ to be the terminal nodes, solving the Steiner tree problem on such modified graph will provide the smallest possible set of encoding nodes, $\mathcal{N}^{\mathrm{e}}$. In case of multiple failures, such ENL subproblem can be seen as an $|\mathcal{F}|$-dimensional minimum Steiner tree problem. Thus, the ENL problem is NP-hard as well. □

### 5.3.4 Upper Bound and Heuristic Algorithm

As stated in [CTF10], finding the most effective subset of network encoding nodes is an NP-hard problem. For this reason the following heuristic is proposed.

#### 5.3.4.1 Upper Bound

Let us assume $\mathcal{T}^{\mathrm{U}}$ as a universe set of feasible path tree partitions of nodes. That is, $\mathcal{T}^{\mathrm{U}} = \{\mathcal{T}^1, \mathcal{T}^2, ..., \mathcal{T}^{|\mathcal{T}^{\mathrm{U}}|}\}$ and $\mathcal{T}^i = \{\mathcal{T}^i_{g_1}, \mathcal{T}^i_{g_2}, ..., \mathcal{T}^i_{g_{|\mathcal{G}|}}\}$, $\forall i \in \{1, ..., |\mathcal{T}^{\mathrm{U}}|\}$. That is, $\mathcal{T}^i$ includes a tree for each gateway/root and $\mathcal{T}^i_{g_i}$ is used to denote the tree having $g_i$ as root, covering nodes $\mathcal{N}^i_{g_i}$ and including links $\mathcal{L}^i_{g_i}$. Each partition $\mathcal{T}^i$ covers, therefore, all nodes. The impact of a path tree partition of nodes is described by the following cost function $f : \mathcal{T}^{\mathrm{U}} \to \mathbb{N}$:

$$f(\mathcal{T}^i) = \sum_{g\in\mathcal{G}} \sum_{n\in\mathcal{N}^i_g} Hops(n, g) \tag{5.13}$$

where $Hops(n, g)$ is the number of hops from node $n$ towards $g$. The best path tree partition of nodes is, therefore, given by:

$$\mathcal{T}^* = \arg \min_{\mathcal{T}^i \in \mathcal{T}^{\mathrm{U}}} \{f(\mathcal{T}^i)\} \tag{5.14}$$

and $\mathcal{L}^*$ will be its links. Let us assume now that the overall set of links that need to be protected is given by:

$$\mathcal{L}^{\mathcal{F}} = \bigcup_{f \in \mathcal{F}} \mathcal{L}^f \tag{5.15}$$

meaning that duplicate links are merged. If the set of alternative paths, where overhearing/encoding nodes would be applied, associated with failing link $l \in \mathcal{L}^{\mathcal{F}}$ is defined by $\mathcal{P}_l = \{p = (src(l), ..., n_i) : \exists (n_i, n_j) \in \mathcal{L}^* \backslash \mathcal{L}^{\mathcal{F}}\}$, then the following cost function $g : \mathcal{P}_{l_1} \times \mathcal{P}_{l_2} \times ... \times \mathcal{P}_{l_{|\mathcal{L}^{\mathcal{F}}|}} \rightarrow \mathbb{N}$ can be defined:

$$g(\mathbf{p} = [p_{l_1}, p_{l_2}, ..., p_{l_{|\mathcal{L}^{\mathcal{F}}|}}]) = |\bigcup_{i \in \{1,...,|\mathcal{L}^{\mathcal{F}}|\}} p_{l_i}| \tag{5.16}$$

Thus, the optimal solution would be:

$$S^* = argmin_{\mathbf{p}}\{g(\mathbf{p})\}. \tag{5.17}$$

Since this is hard to find, an upper bound can be obtained by $S^{\mathrm{UB}} = g(\mathbf{p}^{\min})$, where $\mathbf{p}^{\min} = [p_{l_1}^{\min}, p_{l_2}^{\min}, ..., p_{l_{|\mathcal{L}^{\mathcal{F}}|}}^{\min}]$ is made of shortest paths only. That is, $p_{l_i}^{\min}$ is the path starting at $src(l_i)$ that takes less hops to reach a node connected to an uplink in $\mathcal{L}^* \backslash \mathcal{L}^{\mathcal{F}}$.

#### 5.3.4.2 Algorithm

Based on the previous upper bound, a scalable heuristic algorithm is proposed to determine which nodes should be encoding nodes. This is shown in Algorithm

1. Assuming the best known implementation for the single source shortest path problem, having complexity $O(|\mathcal{L}| + |\mathcal{N}| \log\log |\mathcal{N}|)$ (see [Tho04]), Algorithm 1 will have complexity $O(|\mathcal{G}| + |\mathcal{L}| + |\mathcal{N}| \log\log |\mathcal{N}|)$. More specifically, although Lines 17-22 include a call to the Dijkstra algorithm, meaning that such algorithm could run $|\mathcal{L}|$ times (at most), it is possible to make a single call to the Dijkstra algorithm if a virtual node, connected to every $src(l) \in \mathcal{L}^{\mathcal{F}}$, is added to the topology. This possibility is omitted for the sake of clarity of the algorithm. Therefore, Line 13 plus Lines 17-22 result into complexity $O(|\mathcal{L}| + |\mathcal{N}| \log\log |\mathcal{N}|)$ because coefficients are ignored in the Big-$O$ notation. Lines 14 and 20 are assumed to bring no extra complexity because such information can be ensured during the execution of the Dijkstra algorithm, if adequate data structures are used. Lines 8-11 and Lines 24-26 result into $|\mathcal{G}|$ and $|\mathcal{L}|$ steps (at most), respectively. Therefore, the overall complexity of Algorithm 1 will be $O(|\mathcal{G}| + |\mathcal{L}| + |\mathcal{N}| \log\log |\mathcal{N}|)$, after ignoring coefficients. From the foregoing complexity, Algorithm 1 can be considered scalable since components exhibit a growth lower than quadratic [Ten16].

## 5.4 Performance Evaluation

### 5.4.1 Scenario Setup

Two different topology sizes were used for the analysis of results. These topologies were randomly generated using the algorithm in [OS07]. For each topology, two different connectivity degrees (dense and sparse) were considered. Such topology information is summarized in Table 5.1.

---

*Input:* $\mathcal{N}$, $\mathcal{G}$, $\mathcal{L}$, $\mathcal{F}$, $\mathcal{L}^f, \forall f \in \mathcal{F}$;
*Output:* $\mathcal{T}^*$, $\mathcal{N}^e$;

---

/* *add virtual node and connect it to gateways* */;
$\mathcal{N}^{\mathrm{e}} = \emptyset$;
$\mathcal{N} = \mathcal{N} \cup \{v^X\}$;
**for** $g \in \mathcal{G}$ **do**
    $\mathcal{L} \cup (v^X, g)$;
    $\mathcal{L} \cup (g, v^X)$;
**end**
/* *find path trees rooted at gateways* */;
$(\mathcal{T}^*, \mathcal{L}^*)$=Dijsktra($source = v^X$,$destinations = \mathcal{N} \backslash \mathcal{G}$);
Compute $\mathcal{L}^{\mathcal{F}}$;
$\mathbf{p}^{\min} = []$;
/* *find alternative paths* */;
**for** $l \in \mathcal{L}^{\mathcal{F}}$ **do**
    $destinations = \{n_i \in \mathcal{N} \backslash \mathcal{G} : \exists (n_i, n_j) \in \mathcal{L}^* \backslash \mathcal{L}^{\mathcal{F}}\}$;
    $(\mathcal{T}^{\mathrm{temp}}$,$\mathcal{L}^{\mathrm{temp}})$=Dijsktra($source = src(l)$, $destinations$);
    $p_l$=ExtractShortestPath($\mathcal{T}^{\mathrm{temp}}$,$\mathcal{L}^{\mathrm{temp}}$);
    $\mathbf{p}^{\min} \leftarrow p_l$;
**end**
/* *determine encoding nodes* */;
**for** $l \in \bigcup_{l \in \mathcal{L}^{\mathcal{F}}} \boldsymbol{p}^{min}[p_l]$ **do**
    $\mathcal{N}^{\mathrm{e}} \leftarrow dst(l)$;
**end**

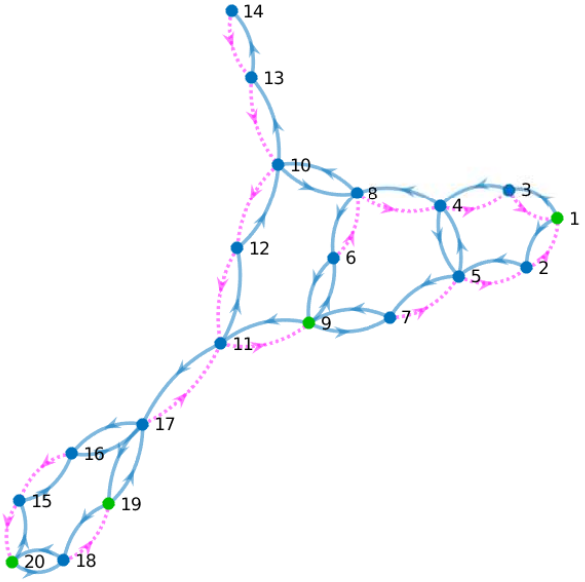
---

**Algorithm 1:** Determining path trees and encoding nodes.

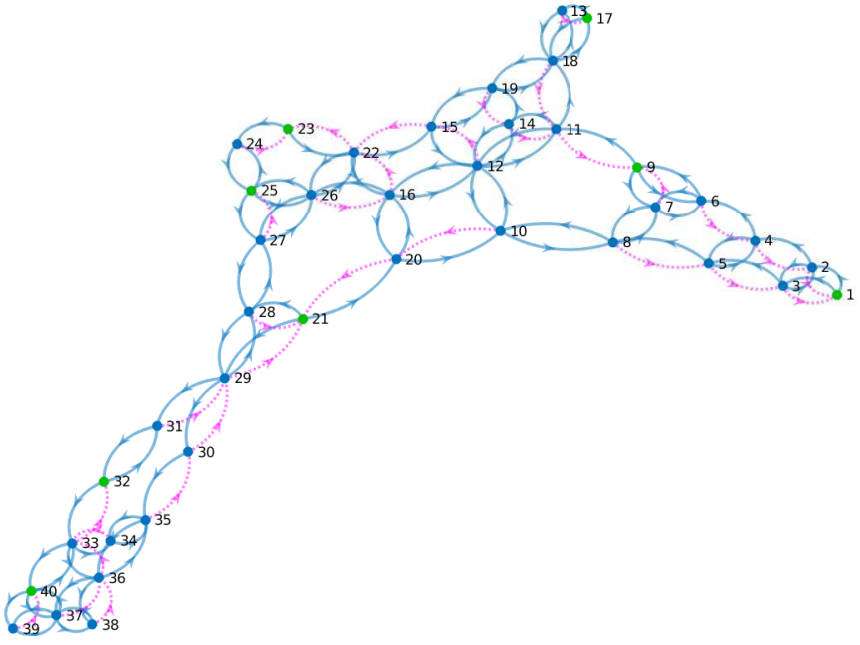
Table 5.1: Topologies used in Performance Analysis (ENL Problem).

| # Nodes | # Links | Connectivity Degree |
|---|---|---|
| 20 | 50 | sparse |
| 20 | 80 | dense |
| 40 | 120 | sparse |
| 40 | 150 | dense |

CPLEX[1] was used to find the results of the optimization model, while the heuristic and the random linear network coding simulation (for comparison of the amount of generated packets by our approach and [KAAF13]) were built in Matlab[2].

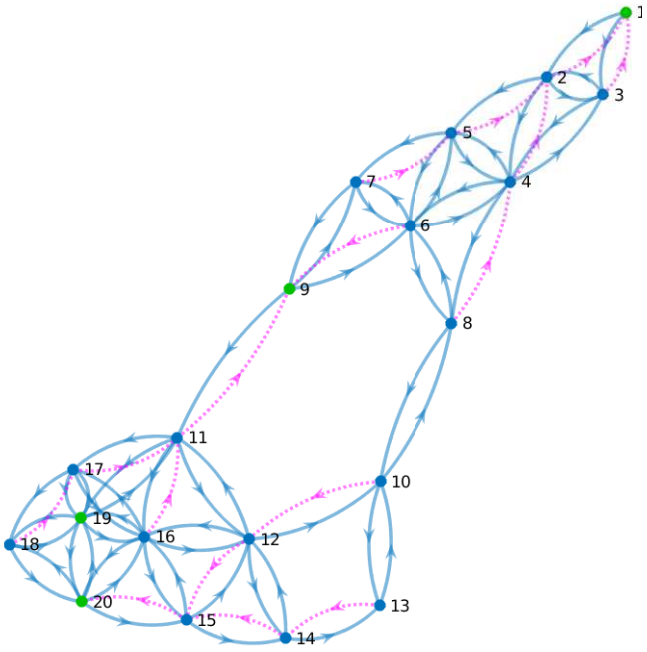



(a) Example of a sparse topology with 20 nodes.

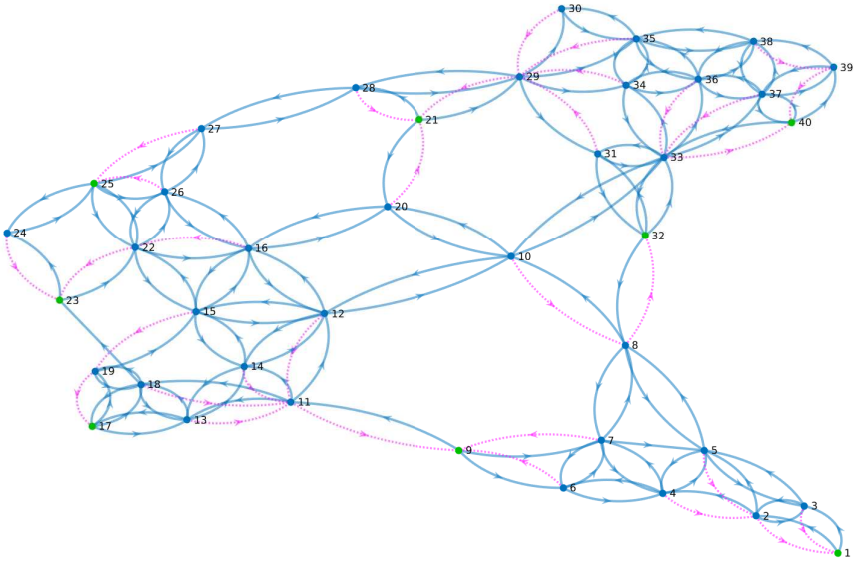



(b) Example of a sparse topology with 40 nodes.

Figure 5.2: Sparse network topologies (ENL Problem). For illustration, maroon links highlight a particular tree (from blue wireless links) while green nodes highlight a particular set of gateways.




(a) Example of a dense topology with 20 nodes.




(b) Example of a dense topology with 40 nodes.

Figure 5.3: Dense network topologies (ENL Problem). For illustration, maroon links highlight a particular tree (from blue wireless links) while green nodes highlight a particular set of gateways.

Table 5.2: Example of Failure Scenarios (ENL Problem).

| Nodes | Links | #fScens | fScens |
|---|---|---|---|
| 20 | 50 | 4 | [40][6][45][18] |
| 20 | 80 | 3 | [28][17][46] |
| 20 | 50 | 5 | [23][45 16][31][47 1][26 30] |
| 20 | 80 | 2 | [78 3][7 68] |
| 40 | 120 | 2 | [44 114][11 93] |
| 40 | 150 | 3 | [9 40][151 36][148 73] |
| 40 | 120 | 2 | [31 41 4 52][62 10 93] |
| 40 | 150 | 1 | [36 74 28 106] |

### 5.4.2 Analysis of Results

To analyze the performance of both the mathematical and heuristic approaches, different tests were performed for randomly generated topologies, illustrated in Figures 5.2 and 5.3. These figures include maroon links to highlight a particular tree for data flow (for illustration), while green nodes highlight a particular set of gateways. Different failure scenarios were also randomly generated, based on specific link failure rates. Failure scenarios do not occur at the same time, but each failure scenario can lead to the failure of one or more links at the same time. More specifically, tests were done considering:

- Number of failure scenarios: 1-5.
- Link failure rate of each failure scenario: 5% or 10%.
- Network density: dense or sparse.
- Number of gateways: 1 or 4 for small topologies, and 1 or 8 for big topologies.

Table 5.2 provides examples of failure scenarios, where $\#fScens$ is the number of failure scenarios and $fScens$ are the failure scenarios (each having one or more link

[1]IBM ILOG CPLEX Optimizer version 12.8.
[2]MathWorks, Inc.

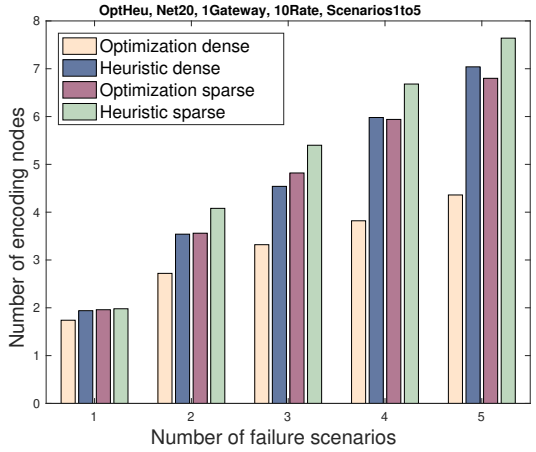
numbers). Each test included 50 runs, and failing links change randomly at every run. The average of such 50 runs is used for the plotting of results.

Note that the mathematical model is making a global optimization, which means that CPLEX will decide for trees that avoid failing links. This way the number of encoding nodes is minimized (objective function). On the contrary, the heuristic algorithm includes two steps: *i*) building the trees; and *ii*) finding alternative paths (requiring encoding nodes) for failing links. Therefore, to fairly evaluate the performance of the heuristic, the mathematical model was also solved in two steps: *i*) building the trees, using just constraints (5.2)-(5.6) and having as goal the minimization of the total number of hops; *ii*) finding alternative paths, using constraints (5.7)-(5.10) and setting the tree related variables according to the output from the first step. The link failure rate of each failure scenario, 5% or 10%, is then applied to tree links.

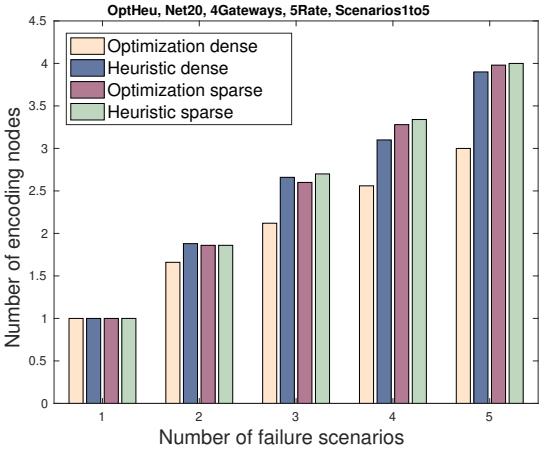
Regarding a possible comparison between the results obtained by the mathematical model, or heuristic, and SenseCode proposed in [KAAF13], it is important to highlight that the last assumes that all nodes are encoding nodes, and for this reason Figure 5.4 and Figure 5.5 do not include SenseCode. Section 5.4.2.3 discusses SenseCode regarding the amount of generated packets.

#### 5.4.2.1 Number of Encoding Nodes: Small Topology Tests

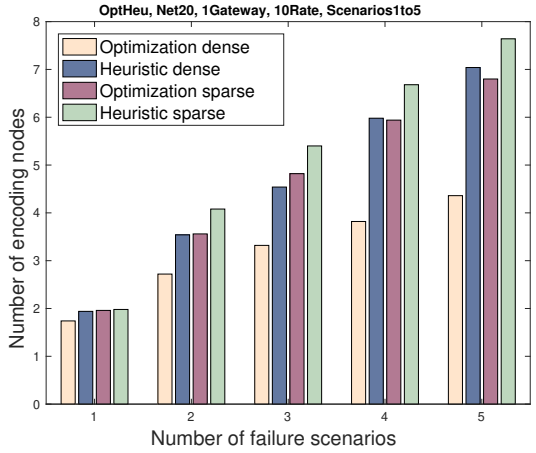
Tests were performed for the sparse and dense 20-node topologies shown in Figures 5.2 and 5.3. Plots 5.4a and 5.4b are results obtained for 5% of failure rate, 1 and 4 gateways, respectively, while plots 5.4c and 5.4d are results obtained for 10% of failure rate, 1 and 4 gateways, respectively. Results from the mathematical



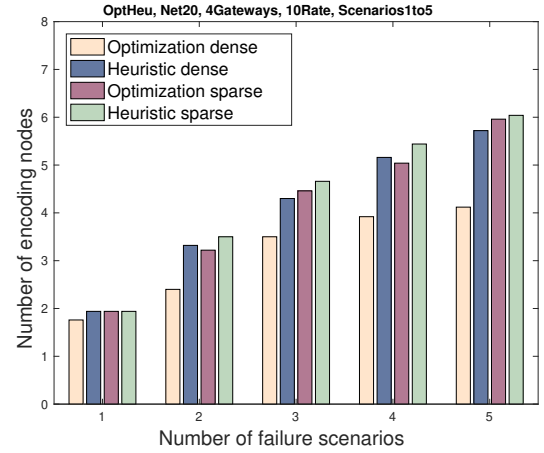

(a) 1 gateway, failure rate of 5%.




(b) 4 gateways, failure rate of 5%.




(c) 1 gateway, failure rate of 10%.




(d) 4 gateways, failure rate of 10%.

Figure 5.4: Number of encoding nodes for the 20-node networks: sparse and dense topology tests (ENL Problem).

model (optimal) and the heuristic are both plotted for a changing number of failure scenarios.

Regarding the sparse topology tests, it is possible to observe in plots 5.4a and 5.4b that there is a slight decrease in the number of required encoding nodes when the number of gateways change from 1 to 4. For plots 5.4c and 5.4d the benefit of using more gateways is much more significant, in particular for larger number of failure scenarios. In general, it is possible to state that the heuristic algorithm is able

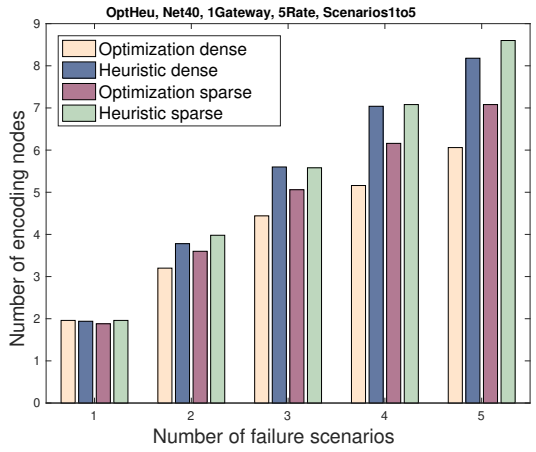
to get close to the optimal results obtained by the mathematical model, although this is less pronounced when there is a single gateway (less flexible scenario). Such behaviour is independent of the number of failure scenarios, which strengthens the scalability of the heuristic algorithm.

Concerning dense topology tests, it is clear that the heuristic does not perform so well, although its performance improves when more gateways are used. Such behavior is similar for both failure rates of 5% and 10%. This means that multiple possible paths between nodes and gateways can make the algorithm diverge from the optimal. The number of encoding nodes is higher for a failure rate of 10%, similarly to the previous tests.

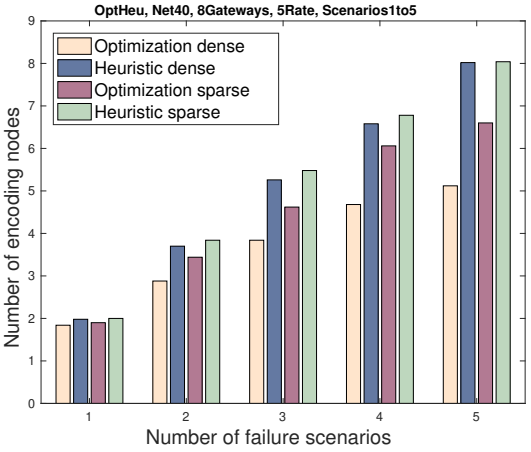
#### 5.4.2.2 Number of Encoding Nodes: Big Topology Tests

Similarly to the previous section, tests were performed for the sparse and dense 40-node topologies shown in Figures 5.2 and 5.3. Plots 5.5a and 5.5b are results obtained for 5% of failure rate, but now for 1 and 8 gateways, respectively, while plots 5.5c and 5.5d are results obtained for 10% of failure rate, also 1 and 8 gateways, respectively. Results from the mathematical model (optimal) and the heuristic are both plotted for a changing number of failure scenarios.

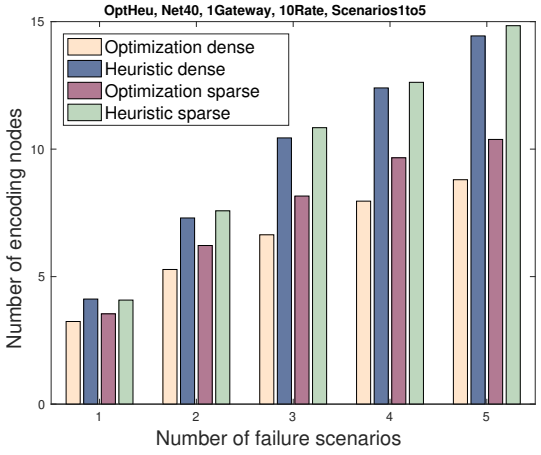
From the sparse 40-node topology tests, it is possible to observe an increase in the number of encoding nodes. This is an expected result because the number of failure scenarios under consideration is greater. The solutions obtained are slightly worse than the ones obtained for the small topology, regarding heuristic to optimal behaviour. That is, for small topologies the heuristic is able to get closer to the optimal values obtained by the mathematical formulation, meaning that the heuristic may have scalability issues regarding the number of nodes. As network size increases,



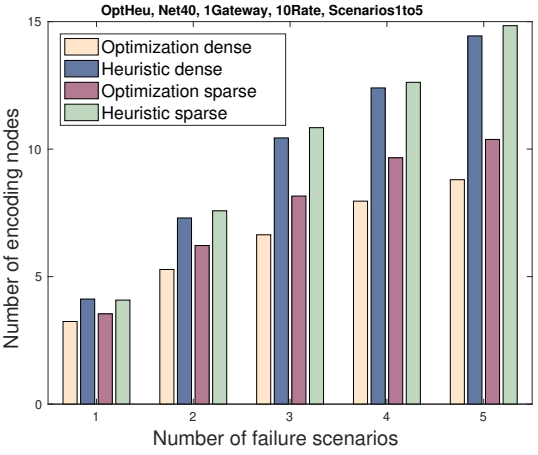

(a) 1 gateway, failure rate of 5%.




(b) 8 gateways, failure rate of 5%.




(c) 1 gateway, failure rate of 10%.




(d) 8 gateways, failure rate of 10%.

Figure 5.5: Number of encoding nodes for the 40-node networks: sparse and dense topology tests (ENL Problem).

however, the benefit of using more gateways becomes much clear, specially for higher failure rates. The gap between the heuristic and optimal also becomes smaller, meaning that increasing the number of gateways, while considering multiple failure scenarios, may compensate the just mentioned scalability issue.

Concerning dense 40-node topology tests, results show that these were the scenarios presenting greater heuristic to optimal gaps, confirming that multiple possible paths, between nodes and gateways, can make the algorithm diverge from the optimal,

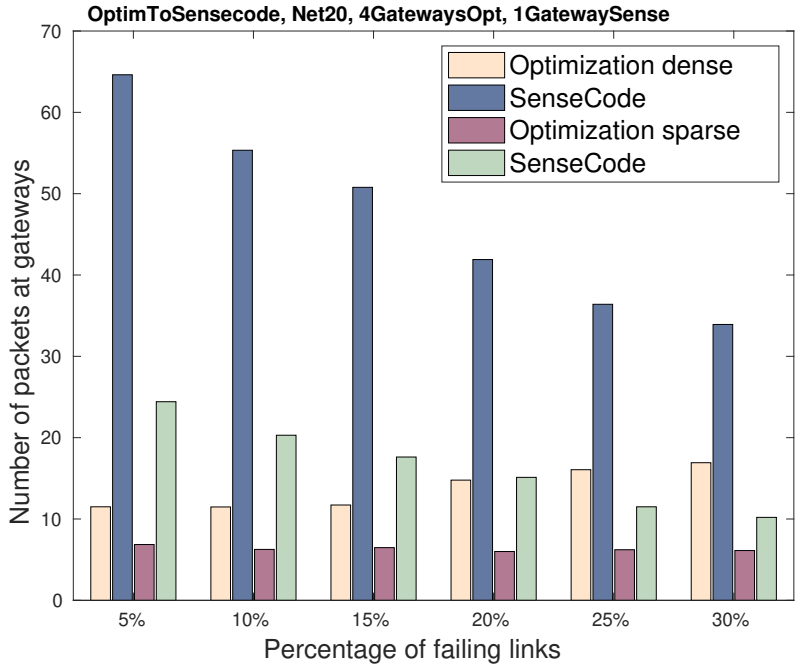



Figure 5.6: Total number of packets at gateways (original and coded) for 20-node networks (ENL Problem).

although performance improves when more gateways are used.

#### 5.4.2.3 SenseCode vs Proposed Approach

SenseCode assumes that all channels have a packet-erasure probability $\epsilon$, but as long the sink receives $N$ linearly independent combinations ($N$ is the number of data sources), generated at intermediate relay nodes performing encoding, then recovery can be performed. The recovery depends on links failing to deliver packets and overhearing neighborhoods. Thus, in general, recovery ends up being more difficult when failing links are closer to the gateway. The proposed approach, on the contrary, considers specific critical links with very high packet-erasure probability (failure scenarios), while ensuring that a node in the neighborhood performs encoding for linear combinations to reach a gateway. The remaining links have zero packet-erasure probability. This way linear combinations are ensured to reach the gateway, allowing

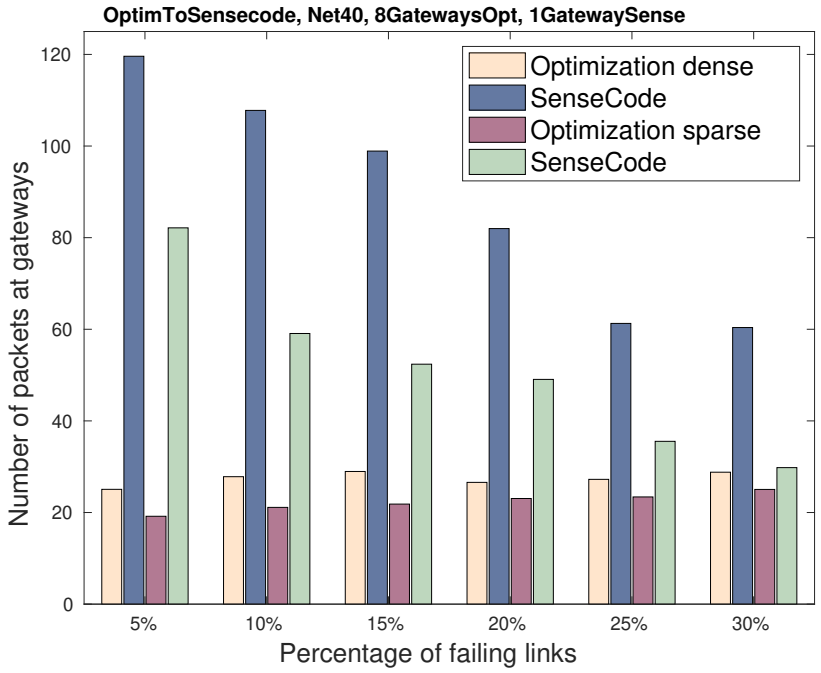



Figure 5.7: Total number of packets at gateways (original and coded) for 40-node networks (ENL Problem).

recovery to be performed. In other words, the proposed approach is suitable for network environments having predictable critical links, while SenseCode is suitable for network environments where packet loss location is not predictable.

It is possible to state that as long as critical/failing links are clearly identified (failure scenarios), the proposed approach has significant advantages because less packets are generated, due to less encoding nodes, allowing for longer network lifetimes and higher goodputs (rate of useful data) to be achieved. Figure 5.6 and Figure 5.7 show the total number of packets (original and coded) at the gateways, in 20-node and 40-node networks, for both approaches. A single failure scenario, within which the percentage of failing links changes, is assumed for comparison with SenseCode to be possible. Also, similarly to [KAAF13], only the tree leaf nodes are data sources, assuming the trees generated by the optimization model discussed in Section 5.3.2. The percentage of failing links relates to tree links.

## 5.5 Summary

In this chapter, the design of network coding based reliable sensor networks is discussed. The aim is to determine a sufficient number of encoding nodes, and their location, taking into account failure scenarios. A mathematical model and a heuristic algorithm are developed to achieve such objective, considering either a single or multiple gateways. When multiple gateways are used, a shared distributed storage system becomes necessary because encoding nodes may listen to packets being forwarded to different gateways. Results show that both the mathematical model and heuristic algorithm can significantly reduce the number of required encoding nodes, when compared with a scenario where all network nodes are encoding nodes. Results also show that the heuristic is able to get closer to the optimal (obtained by the mathematical model) when small and sparse networks are considered. Large and dense networks make the heuristic diverge from the optimal, but this can be avoided if more gateways are considered. The adopted approach also generates a significantly smaller number of packets when compared to SenseCode proposed in [KAAF13].

# 6

# DAG-CODER FOR RELIABLE WSNS

Wireless sensor devices are usually powered with limited non-rechargeable batteries. This energy can easily be depleted when no wise consumption procedures are used, leaving the sensor non-functional. As most of the energy is consumed during the transmission of data, developing efficient data manipulation and transmission approaches is crucial and still an open problem that attracts the attention of many research groups. In this chapter a Directed Acyclic Graph (DAG) based dissemination approach, where clustering and network coding techniques are applied, is proposed. As a main goal, our contribution in this chapter aims at improving the network reliability (ensure recovery of lost packets), while minimizing energy consumption and balancing load at gateways, but now focusing on network environments where failure scenarios (critical links) can not be predicted. That is, as long as critical/failing links are clearly identified (failure scenarios), the approach

of the previous chapter presents significant advantages because less coded packets are generated, allowing for longer network lifetimes and higher goodputs. Note that goodput measures received original data/packets only, while throughput measures all data/packets, even if not useful (e.g., duplicate or coded packets not necessary for the decoding process). The transmission of unnecessary coded packets increases the time required to deliver original data, because bandwidth is being taken, meaning that goodput (received original data per time unit) decreases. The approach from the previous chapter, however, can not be applied when failure scenarios are difficult to determine, which is the case of dynamic environments. This case is addressed here in this chapter.

The main arguments behind reducing packet loss, instead of being concerned with bandwidth, are that: *i*) nodes typically produce small volumes of data, and for this reason bandwidth ends up not being critical in many sensor networks; *ii*) drops are mainly caused by links with high packet-erasure probability (bad link quality) and not due to congestion. For these reasons, multipath communication is explored by many authors. Some authors maintain multiple disjoint paths between communicating end-points, as in [GCSS14], while others propose to disseminate coded packets through all available paths (all nodes are encoding nodes), as in [KAAF13, JTV+17]. The approach in [GCSS14] does not explore all available paths, and packets may not be relayed even though a viable path exists. The approach is also more adequate for wireless mesh networks having both source and destination nodes at the wireless section. Regarding the approaches in [KAAF13, JTV+17], all paths are explored, which means that these are more adequate for dynamic environments. However, only [KAAF13] was designed for many-to-one data dissemination in WSNs. In [JTV+17] (see details in Chapter 3) all nodes are supposed to receive the generated data, being more suited to disseminate control data among all nodes. This could be applied to

many-to-one data dissemination if the stopping criteria is for data to reach one of the gateways.

For the particular case of many-to-one data dissemination, when comparing [KAAF13] and our previous work [AHCB19] the last is suitable for network environments having predictable critical links, while SenseCode in [KAAF13] is suitable for network environments where packet loss location is not predictable. However, SenseCode generates too many packets because a node transmits its own packets, packets from its children (a tree rooted at sink/gateway is assumed), overheard packets and generated coded packets. These packets end up being received by the parent node and heard by all neighbours, meaning that the number of packets will be quite large, in particular near the gateways. Therefore, congestion will be critical near these nodes. For this reason, the goal now is to propose a strategy that improves network reliability (ensure recovery of lost packets) while avoiding having to transmit too many packets (for energy saving) and while balancing load at gateways (reducing congestion), for scenarios where critical links (failure scenarios) are not given. The proposed approach will be compared with the ones from [KAAF13] and [JTV+17].

The remainder of this chapter is organized as follows. In Section 6.1 the relevant related work is discussed. In Section 6.2, the adopted network architecture and data dissemination problem are defined. Section 6.3 presents the mathematical formulation for the DAG-based approach, used to address the data dissemination problem, and Section 6.4 makes the performance analysis of the proposed approach and of two other approaches from the literature. Section 6.6 presents the chapter summary.

**Contributions:**

- A DAG based dissemination approach is proposed. The nodes selected to become cluster heads are the only ones participating in the DAG. Since encoding is performed by cluster heads only, the number of generated coded packets reduces when compared with other approaches.
- A deep comparison between the proposed approach and proposals from the literature is performed.

**Publications:**

- E. AL-Hawri, N. Correia and A. Barradas, "DAG-Coder: Directed Acyclic Graph Based Network Coding for Reliable Wireless Sensor Networks", submitted to IEEE Access journal (revisions suggested by reviewers are in progress).
- E. AL-Hawri, N. Correia and A. Barradas, "Probabilistic Random Linear Network Coding for Reliable Wireless Sensor Networks" submitted to Advanced Doctoral Conference on Computing, Electrical and Industrial Systems.

## 6.1 Related Work

In wired networks, network coding is mainly used to increase the throughput in one-to-many multicast transmissions (see chapter 3). When traffic flow is many-to-one, as in WSNs, network coding can be used to decrease the packet loss, leading to greater network reliability. Thus, the broadcast nature of wireless transmissions, many times seen as a disadvantage, can help ensuring reliability in an elegant way [OWK13]. Any node listening to the packets can work as a next-hop, allowing for an easy tailoring of the flow to the network environment, and accommodating different traffic patterns. As long as the gateway(s) receive enough independent coded packets, packet loss recovery is possible. This decreases the number of packet

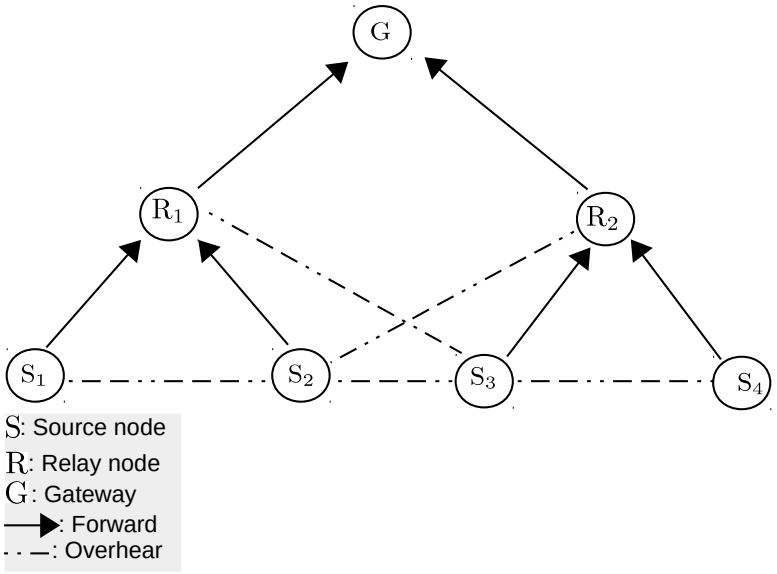



Figure 6.1: Routing tree in SenseCode [KAAF13].

retransmissions required when only opportunistic routing is used. WiFi or WiMAX networks can also benefit from listening and binary network coding, as discussed in [NTNB08].

### 6.1.1 General Data Dissemination Related Work

The CodeDrip proposed in [JTV+17], and already detailed in Chapter 3, is a data dissemination protocol using network coding. The aim of using network coding in CodDrip is to enhance the reliability and speed of dissemination, while reducing the energy consumed, and XOR with Galois field ($\mathbb{F}_2$) is used. Since it has been designed to ensure that all nodes receive the propagated data, it ends up being more suited to disseminate control data in wireless networks.

Another relevant protocol, previously discussed in Chapter 5, is SenseCode [KAAF13]. SenseCode in a many-to-one protocol. That is, multiple sources forward data packets towards a gateway using tree-based routing. A node may deal with three kinds

Table 6.1: Packets Overheard in SenseCode for Scenario in Figure 6.1.

| Node | Overheard | Sent |
|---|---|---|
| $S_1$ | $k_2$ | $k_1, k_1 + k_2$ |
| $S_2$ | $k_1, k_3$ | $k_2, k_1 + k_2 + k_3$ |
| $S_3$ | $k_2, k_4$ | $k_3, k_2 + k_3 + k_4$ |
| $S_4$ | $k_3$ | $k_4, k_3 + k_4$ |
| $R_1$ | $k_3, k_2 + k_3 + k_4$ | $k_1, k_2, k_3, k_2 + k_3 + k_4$ |
| $R_2$ | $k_2, k_1 + k_2 + k_3$ | $k_2, k_3, k_4, k_1 + k_2 + k_3$ |

of messages: messages received from its children, messages overheard from its neighbors, and its own messages. To perform network coding, the node generates linear combinations of these messages. Applying this technique ensures that the sink will be able to recover the packets that have been dropped. Table 6.1 shows an example of how data is forwarded and overheard in SenseCode, assuming the routing tree depicted in Figure 6.1. In the example, each source $S_i$, $1 \leq i \leq 4$, sends its $k_i$ packet towards gateway $G$ using $R_1$ and $R_2$ as relay nodes. It is assumed that all nodes perform encoding, sources included. As seen in the Table 6.1, SenseCode ensures that the gateway $G$ will receive the linear combinations of all packets even if one of the relay nodes fails to communicate its data to the gateway.

The NetCoDer in [VMMdA+16] applies linear network coding in an industrial context. A start topology communication model is assumed where multiple sensor devices send their data, in their assigned slot, to a coordinator at the middle of the topology. Nodes can act as relays, depending on the reliability of communications, retransmitting data during retransmission slots. Such proposal can only be applied to local wireless sensor networks having a star topology.

Inter-flow Network Coding based Opportunistic Routing (INCOR), in [ZYY+15], incorporates both opportunistic routing and inter-flow network coding to increase the performance of WSNs. This approach exploits the broadcast nature of wireless and the spatial diversity of multi-hop wireless networks. The authors present a new

metric to define the prioritization of forwarders in the candidate set of nodes. Then, they design the network coding based opportunistic routing using the defined metric. The authors in [KK16] proposed an algorithm that uses network coding as a solution to reduce the energy consumption and to increase the network lifetime in multicast networks.

From all these proposals, SenseCode and CodeDrip are the only network coding based dissemination approaches with reliability concerns that can be applied in many-to-one scenarios.

### 6.1.2 Clustering Related Work

In [HL17], the authors study a cluster-based WSN model where network coding is applied to nodes located in the overloaded area (near the sink). The network is divided into two areas: cluster and bottleneck/overloaded. At the cluster area, nodes form clusters and every CH receives data from its members. At the bottleneck area, on the other hand, nodes are divided into relay and coding nodes. A relay node is responsible for forwarding data coming from clusters while coding nodes are responsible for coding the data coming from clusters. A similar approach is discussed in [KK18], but the network is divided into a square grid, and then in each square the optimum CH is selected based on the maximum normalized remaining energy. A different CH is selected at every round, in each square, in order to equally distribute the energy load between sensor nodes. This approach increases the network lifetime when compared with LEACH.

In [MSS17], the authors claim that energy efficient clustering protocols like LEACH are concerned with network lifetime at the expense of reduced stability periods.

Therefore, in order to increase the network stability, the authors propose an energy-aware heuristic that balances the load between nodes and, consequently, increases the stability periods. The main idea is to select the CHs in a deterministic way, based on the remaining energy. The concern is also to provide a full network coverage. In [MMC+17], CHs are chosen so that the lifespan of the sensor network is extended.

The previously discussed clustering based approaches do not have reliability concerns and are not suitable for network coding based many-to-one flows.

## 6.2 The Data Dissemination Problem

### 6.2.1 Motivation and Architecture

The network coding based data dissemination protocols that can be adopted in many-to-one scenarios, which is the case of WSNs, are SenseCode and CodeDrip. SenseCode assumes data dissemination through a tree rooted at a sink/gateway. In this case, the failure of a link will affect all traffic coming from the subtree below it. Putting all nodes as encoding nodes is, therefore, a way of increasing the probability of packet recovery when links fail, but a large amount of packets will be generated. Diversity of routing to improve bandwidth utilization, as in Figure 3.6, can not be explored because of the tree type routing structure. Regarding CodeDrip, the aim is to enhance the reliability and speed of dissemination, while reducing the energy consumed, but data dissemination stops when data reaches all nodes, and not gateways in particular. That is, it has not been designed having many-to-one scenarios in mind although, as previously stated, its stopping criteria can change to data reaching one of the gateways.

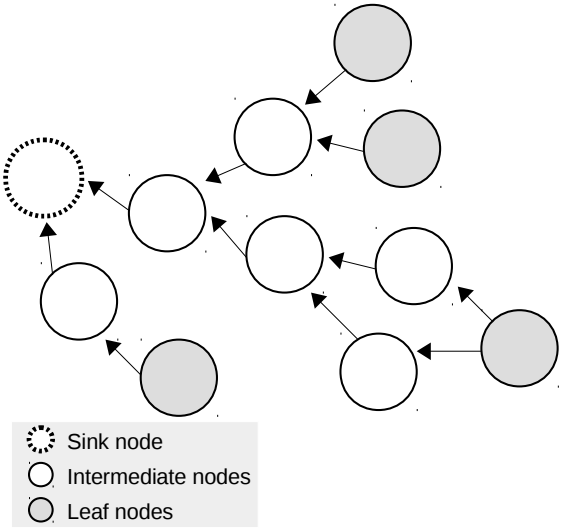



Figure 6.2: DAG with single sink.

Here in this chapter, a DAG-based dissemination approach is proposed that generates less packets than SenseCode and CodeDrip, reducing bandwidth requirements and increasing network lifetime, while increasing packet recovery to achieve reliability. Also, and contrarily to CodeDrip, it has been designed having many-to-one flows in mind. A DAG structure is illustrated in Figure 6.2. Any DAG has at least one topological ordering, which means that for every directed edge $(n_i, n_j)$ the node $n_i$ comes before $n_j$ in the ordering. The proposed approach assumes that gateways are peers in a P2P overlay, which allows diversity in routing (towards different gateways) to be explored, which does not happen in SenseCode. That is, packets reaching different gateways share a storage system that allows the recovery of lost packets even if the required linearly independent combinations have traveled through different routes. Such storage system shared by multiple gateways, and diversity in routing, are the primary advantages of our proposed model over SenseCode and CodeDrip. Such architecture is illustrated in Figure 6.3.

The proposed approach works under the following assumptions:

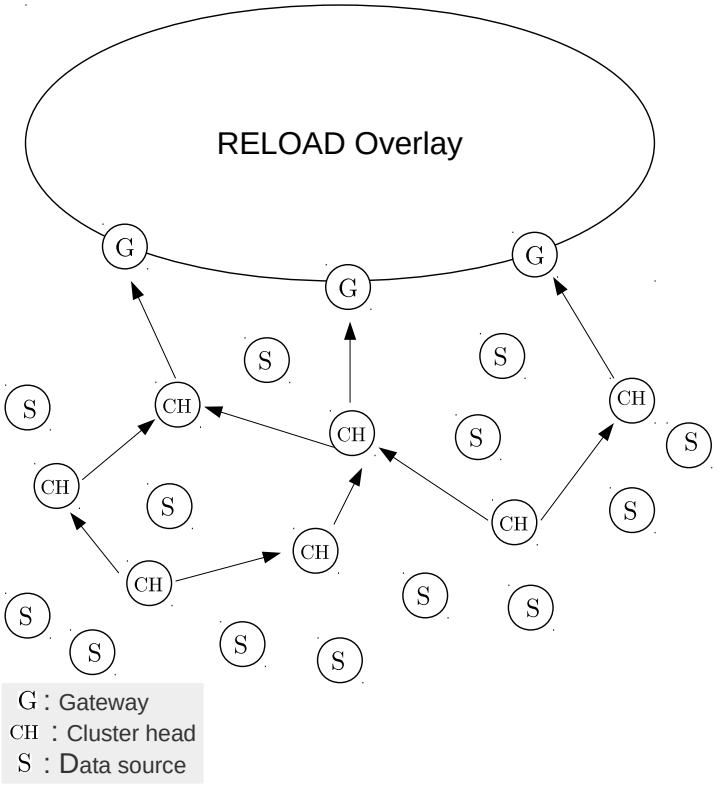



Figure 6.3: DAG with cluster heads. Such nodes perform encoding of data packets from associated sources.

- A wireless node works as a collector (CH) or as a non-collector node.
- Non-collector nodes must be associated with a collector node, and send their data to it. These nodes can hear each other.
- Collector nodes generate coded packets using: *i*) packets received from lower topological order collector nodes; *ii*) packets from their members (non-collector nodes); *iii*) its own packets.
- A collector node has two or more links to other collector nodes. Original data packets, and generated coded packets, are sent through such links. Routing follows the topological ordering of collector nodes.

### 6.2.2 Problem Definition

**Definition 6.1** (DAG-based Network Coding for WSNs (DNC-WSNs))**.** Given a constrained sensor network graph $G(\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ are the nodes and $\mathcal{L}$ are wireless communication channels (links), each with a weight reflecting the required transmission power, determine which nodes of $\mathcal{N}$ should be collector nodes, performing network coding, and which links of $\mathcal{L}$ should be at the DAG, for routing of packets following the topological order of collector nodes, so that energy consumption is minimized while ensuring data flow towards gateways and load balancing at gateways.

Packet loss recovery is possible because all collector nodes will be doing encoding, and sending coded and original packets through multiple paths towards multiple gateways. Energy consumption is minimized because coded packets are generated and transmitted just by CHs.

## 6.3 Mathematical Model

### 6.3.1 Notation and Assumptions

Let us assume a network graph $G(\mathcal{N}, \mathcal{L})$, where $w(l)$ denotes the weight (transmission power) of directed link $l \in \mathcal{L}$. Assume also that $\mathcal{G} \subset \mathcal{N}$ denotes the set of gateways.

A topological ordering of nodes in $\mathcal{N}$ is possible if and only if the graph has no directed cycles. In other words, if it is a DAG. Any topological order of $\mathcal{N}$ is any total order $\tau$ such that if $(n_i, n_j) \in \mathcal{L}$, then $n_i$ precedes $n_j$ in $\tau$. That is, $\tau_{n_i} \leq \tau_{n_j}$.

To incorporate a topological order at an instance of the DNC-WSN problem, it is assumed that $\tau_n$ is predefined for gateways: $\tau_n = 1$, if $n \in \mathcal{G}$. For every other $n \in \mathcal{N} \backslash \mathcal{G}$, since no predefined CHs exist (any $n$ can be selected to become CH) and any node can be a data source (there will be no predefined leafs), while forwarding data from others, a topological order must be dynamically found by the optimizer, while taking into account the objective function (goal) and additional constraints. For every $n \in \mathcal{N} \backslash \mathcal{G}$: $0 \leq \tau_n < 1$.

The variables of the problem are the following:

| | |
|---|---|
| $\beta_n$ | One if wireless node $n \in \mathcal{N} \backslash \mathcal{G}$ is selected to become a CH, participating in the DAG, zero otherwise. |
| $\sigma_l$ | One if link $l \in \mathcal{L}$ is to participate in the DAG (used for data delivery), zero otherwise. |
| $\delta_l^s$ | Percentage of data from source $s \in \mathcal{N} \backslash \mathcal{G}$ that flows through link $l$. |
| $\tau_n$ | Topological order of $n \in \mathcal{N} \backslash \mathcal{G}$ in the DAG. |
| $\xi_n$ | Auxiliar variable to avoid having a fixed DAG outdegree at node $n \in \mathcal{N} \backslash \mathcal{G}$. |

### 6.3.2 Formalization

In this section the mathematical model of the DNC-WSN problem is formalized. This requires choosing CHs and links forming the DAG, a topological order for packet routing, and ensuring flow conservation towards a gateway. Among all possible solutions, the one minimizing energy consumption and balancing load at gateways, while ensuring the recovery of lost packets in case of link failure, should be found. Since some diversity in routing is required when using network coding for packet recovery, and since this can be achieved with two outgoing links from CHs, the

mathematical model should promote solutions with no more than two outgoing links from CHs, for energy saving purposes.

– **Objective Function**

$$\text{Minimize} \sum_{n \in \mathcal{N}} \xi_n + \frac{\sum_{n \in \mathcal{N}} \beta_n + \sum_{l \in \mathcal{L}} w(l) \times \sigma_l}{\Delta}. \tag{6.1}$$

where $\Delta = |\mathcal{N}| \times |\mathcal{L}|$ so that the second component of the objective function does not compete with the first. The first component of the objective function minimizes $\xi_n$ variables, which encourages having more than one outgoing link per CH (see Eq. 6.7 and its explanation), for diversity in routing. The second component is used to minimize energy consumption, which also leads to load balancing at gateways because CHs will use nearer gateways, in order to save energy. The number of gateways and their distribution is pre-planned to serve well a population of nodes. In other words, CHs are selected in a way that energy consumption is minimized and balanced.

– **Cluster Heads**

$$\sum_{l \in \mathcal{L}: src(l)=n} \beta_{dst(l)} \geq 1 - \beta_n, \forall n \in \mathcal{N} \backslash \mathcal{G} \tag{6.2}$$

where $src(l)$ and $dst(l)$ are the source and destination endpoints of directed link $l$, respectively. Constraints (6.2) state that if a node is not CH (collector node) then it must be associated with a CH, for its data to be delivered.

– **Data Delivery**

$$\sum_{\{l\in\mathcal{L}:src(l)=n\}} \delta_l^s - \sum_{\{l\in\mathcal{L}:dst(l)=n\}} \delta_l^s = \begin{cases} \beta_s, & if\ n=s \\ 0, & otherwise \end{cases}, \forall s, n \in \mathcal{N}\backslash\mathcal{G} : n \neq s \quad (6.3)$$

$$\sigma_l \geq \delta_l^s, \forall l \in \mathcal{L}, \forall s \in \mathcal{N}\backslash\mathcal{G} \quad (6.4)$$

$$\sigma_l \leq \frac{1}{2}(\beta_{src(l)} + \beta_{dst(l)}), \forall l \in \mathcal{L} \quad (6.5)$$

Constraints (6.3) ensure flow conservation from any source node $s$ towards any gateway, using CHs as intermediate nodes, and thus avoiding disconnected paths at the DAG. Constraints (6.4) activate links used by the flows in (6.3), while constraints (6.5) ensure that the endpoints of any DAG link are CHs. That is, data flow towards the gateways occurs using CHs only.

– **Acycliness**

$$\tau_{dst(l)} - \tau_{src(l)} > -\Delta + \Delta \times \sigma_l, \forall l \in \mathcal{L} \quad (6.6)$$

Constraints (6.6) define the topological order at the end points of used links, establishing acycliness. This is done for links participating in the DAG, information given by $\sigma_l$ variables.

– **Node Degree**

$$\xi_n \geq 2 \times \beta_n - \sum_{l \in \mathcal{L}: l = src(n)} \sigma_l, \forall n \in \mathcal{N} \backslash \mathcal{G} \tag{6.7}$$

In these constraints, the auxiliar variables $\xi_n$ are used to avoid having a fixed DAG outdegree at CHs. Although an outdegree of 2 should be promoted (see Section 6.2), this might not be possible in certain physical wireless topologies. This impossibility is not only related with the physical topology, but also with the topology order that is imposed to ensure acycliness. Note that, since the goal includes minimizing all $\xi_n$, and following constraints (6.7), the $\xi_n$ variables become: 0, if $n$ is not chosen to become CH; 1, if there is a single outgoing link from $n$; and 0, if there are two or more outgoing links from $n$. Thus, it is of interest to have 2 or more outgoing links (if CH), whenever possible, but the approach is flexible to have a single one, if more outgoing links are not possible. Since the objective function also includes minimizing the number of CHs and energy consumption, through link weights, the solutions tend to use 2 outgoing links at CHs, which avoids increasing packet transmissions more than needed.

– **Bounds and Binary Assignments**

$$0 \leq \delta_l^s, \tau_n, \xi_n \leq 1; \sigma_l, \beta_n \in \{0, 1\}. \tag{6.8}$$

Expression (6.8) states the type of each decision variable, and bounds. Note that, although $\xi_n$ variables have been defined as continuous variables, these will take 0 or 1 value because of expression (6.7). Stating these as continuous variables, instead of

binary, reduces problem complexity and in this particular case does not affect the solution.

CPLEX optimizer is used to solve this problem. The solution found will be the optimal solution for the DNC-WSN problem instance under consideration. Note, however, that this is a Mixed Integer Linear Programming (MILP) problem, which will be difficult to solve for large network instances.

## 6.4 Performance Analysis

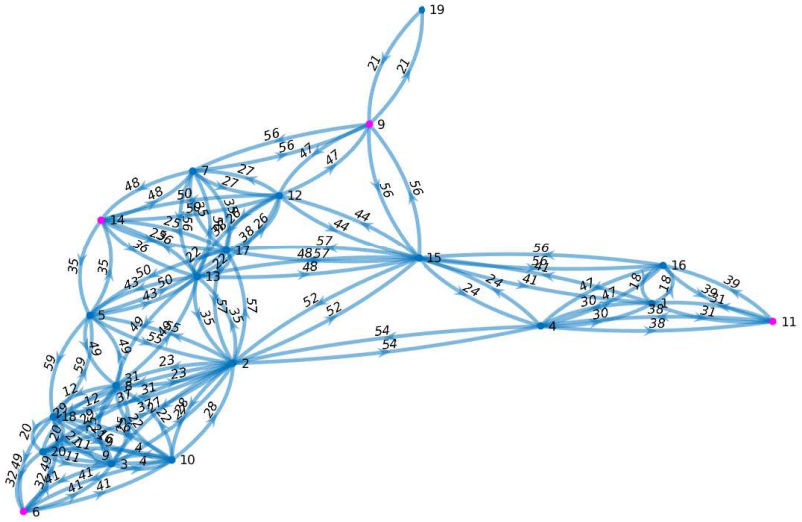
### 6.4.1 Scenario Setup

Randomly generated physical topologies of 20 and 30 nodes were used to evaluate the performance of the DNC-WSN approach, SenseCode and CodeDrip. As previously mentioned, these are the network coding based data dissemination protocols that can be adopted in many-to-one scenarios, which is the case of WSNs. Both dense and sparse topologies were evaluated, and the distance between any two nodes is calculated using the Euclidean distance. Figure 6.4 illustrates 20-node dense and sparse topologies, where distances are shown as link weights. A dense topology is assumed to have a connectivity degree of 0.3, while for a sparse topology this will be 0.2. The connectivity degree is calculated using $\frac{|\mathcal{L}|}{|\mathcal{N}|\times(|\mathcal{N}|-1)}$, where $\mathcal{L}$ is the set of available directed links and $\mathcal{N}$ is the set of network nodes.

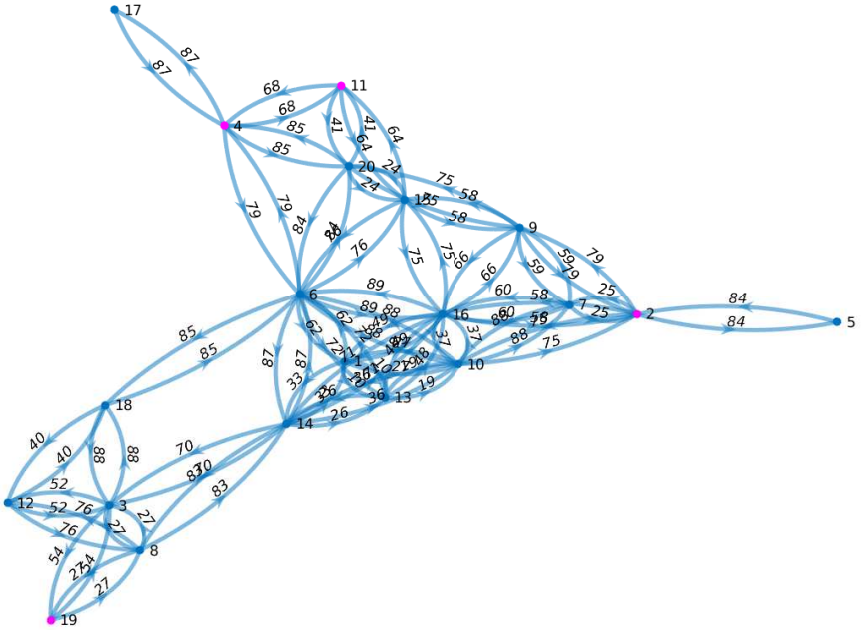
The evaluation follows two steps:

1. Solving the mathematical optimization model to select CHs and generate the DAG for the DNC-WSN approach. This is implemented in CPLEX[1].

[1]IBM ILOG CPLEX Optimizer version 12.8.



(a) Example of a dense topology with 20 nodes (DNC-WSN problem).



(b) Example of a sparse topology with 20 nodes (DNC-WSN problem).

Figure 6.4: Network topologies (DNC-WSN problem). Maroon nodes highlight a particular set of gateways while weights on links are the relative distances between nodes.

2. Implementing random linear network coding for the solutions obtained in Step 1 (DNC-WSN approach). This step also includes the implementation of SenseCode and CodeDrip methods for comparison with DNC-WSN. Such step is implemented in Matlab[2]. In DNC-WSN the encoding nodes will be the CHs, in SenseCode all nodes are encoding nodes, and in CodeDrip a node is an encoding node depending on a certain probability. Two versions of DNC-WSN are generated for testing:

   (a) Non-CHs perform overhearing, and forward any data heard from neighbours (besides their own data) towards the CH to which they are associated.

   (b) Non-CHs perform no overhearing, and forward just their data towards the CH to which they are associated.

The tests performed, in order to compare DNC-WSN, SenseCode and CodeDrip, assume the following parameters:

- Number of gateways: DNC-WSN and CodeDrip consider 4 gateways, for both 20-node and 30-node topologies, while for SenseCode a single gateway is assumed, as in [KAAF13].
- Gateway locations: These are either at the center or at the border of the network.
- Link failure probability: Ranges from 0.05 to 0.5.
- Network connectivity degree: 0.2 for sparse and 0.3 for dense.

[2]MathWorks, Inc.

In each scenario a link failure probability is assumed, so one or more links will be inoperable. For a specific link failure probability, 20 runs are performed (failing links change randomly at each run) and the average is used for the plotting of results.

### 6.4.2 Performance Metrics

In the following plots, two performance metrics are used:

- Reliability: Amount of original packets that are successfully stored at the P2P overlay (not lost or have been recovered). A method achieves 100% reliability if all data packets sent from sources successfully arrive at the gateway(s).
- Packet transmissions: Total number of packet transmissions throughout all the wireless network section. The higher the number of packet transmissions, the greater the amount of energy consumed in each round.

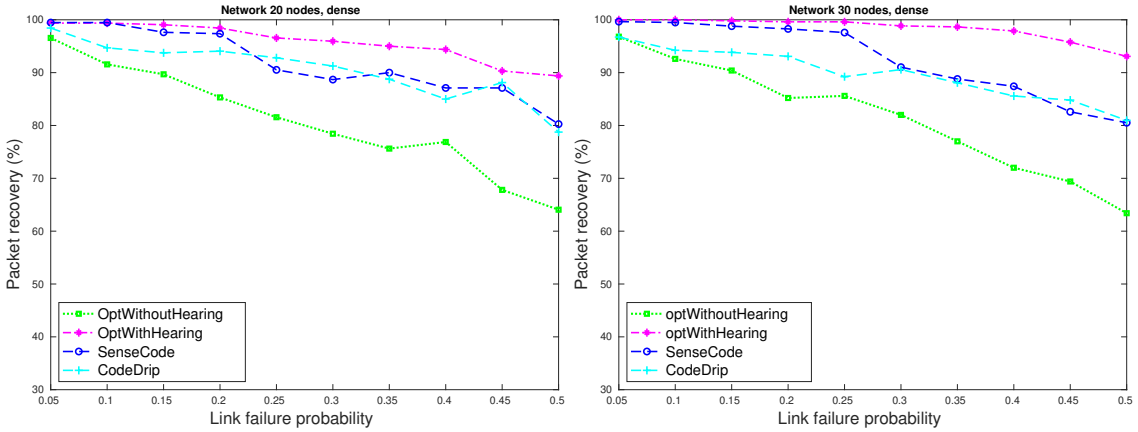
Nodes generates a single packet in each round, and a round ends when not more packets are traversing the network.

### 6.4.3 Analysis of Results

#### 6.4.3.1 Reliability

The results on reliability for dense and sparse network topologies are shown in Fig. 6.5 and Fig. 6.6, respectively. These results include the 20 and 30 node topology cases, for gateways located at the border and center of the network.

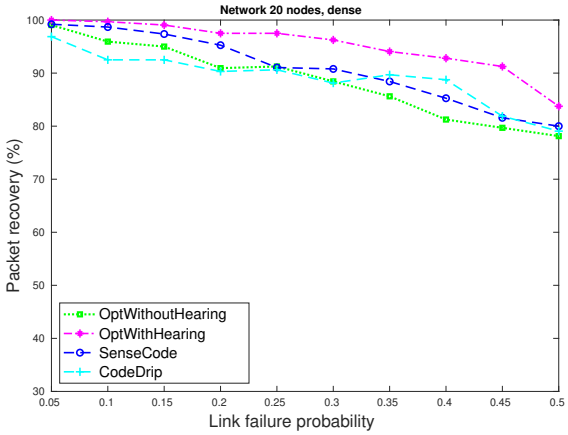
Regarding the impact of gateway location, SenseCode and CodeDrip seem not to perform well, when gateways are located at the border, as link failure probability increases. This is more visible when network topologies are sparse. In addition, while



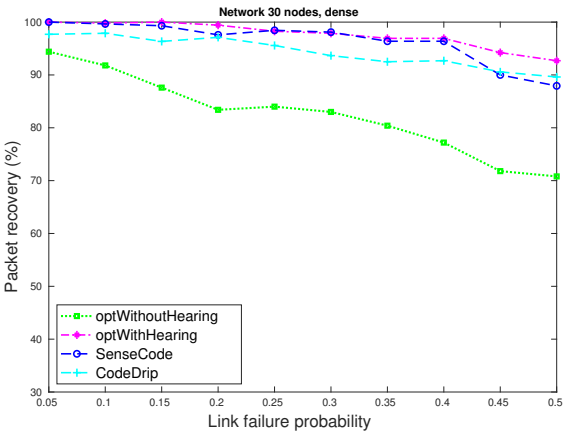

(a) Gateways at the border.




(b) Gateways at the border.



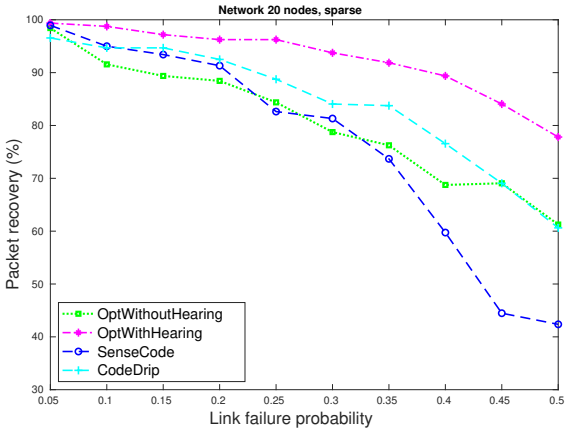

(c) Gateways at the center.



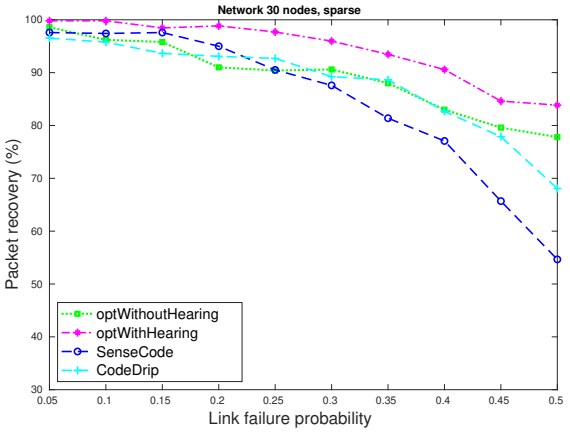

(d) Gateways at the center.

Figure 6.5: Reliability for dense topologies.

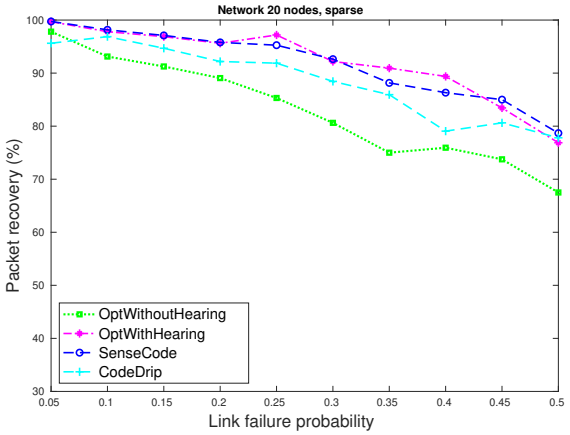
larger networks generally show some improvement compared to their counterparts of smaller size, SenseCode seems to degrade for larger sparse networks when gateways are in the center. The other methods show some stability and improve or keep their performance when the network size increases. SenseCode has, therefore, scalability problems in these scenarios. It has also stability problems, like CodeDrip, because its performance is affected by gateway location and sparseness. Its poor performance in the mentioned scenarios is related with the fact that SenseCode is using a single tree rooted at a single gateway, having no routing diversity. Packets require more hops to get to the gateway, which increases the probability of packet loss, particularly



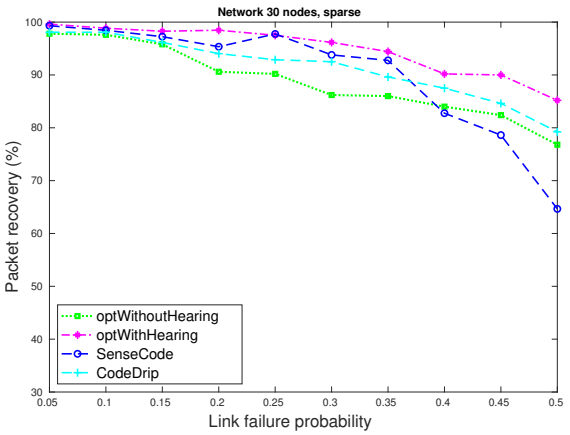

(a) Gateways at the border.




(b) Gateways at the border.




(c) Gateways at the center.




(d) Gateways at the center.

Figure 6.6: Reliability for sparse topologies.

in sparse topologies and when gateways are in the border. That is, packets must successfully traverse more links to reach the gateway. This leads to higher delays and waste of resources because successfully transmitted packets (at the first hops) may still be dropped further ahead, and for their transmission to happen other packets had to stay in queue.

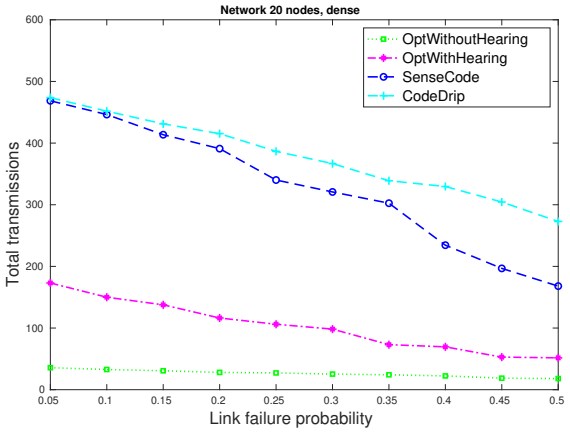
CodeDrip presents no scalability problems (change in network size does not affect its behaviour) because it has routing diversity and explores the multiple gateways, but presents stability problems (performance is affected by the location of gateways and sparseness). Its poor performance in sparse topologies with gateways at the border

is related with CodeDrip's policy, which seems not to favor packet recovery in these scenarios, when compared with DNC-WSN. When a packet arrives, CodeDrip uses a probability to decide whether to send the packet or to combine it with other messages (randomly selected) for sending. XOR is used to combine packets. Although this could save some energy, some packets may not go through the coding process and some lost packets will not be recovered. This also explains the non recovered packets in CodeDrip when the link failure rate is low, which does not happen in SenseCode and DNC-WSN. The fact that SenseCode and CodeDrip are less adequate for gateways located at the border turns out to be a critical issue because such kind of network deployment is very common.

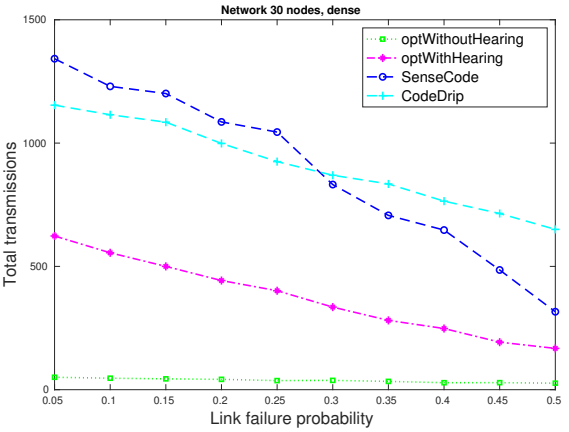
The DNC-WSN with hearing presents the best performance and, contrarily to SenseCode and CodeDrip, high stability since performance is not dependent on gateway location and network size. It is also less affected by network sparseness. This is related with routing diversity towards multiple gateways, explored by both DNC-WSN and CodeDrip, but DNC-WSN's criteria of performing linear encoding using all packets received from lower topological order collector nodes, packets from their members (non-collector nodes) and its own packets, seems to ensure the recovery of more packets than using the probabilistic approach, and XOR, of CodeDrip. The no hearing version of DNC-WSN ends up being ineffective.

#### 6.4.3.2 Packet Transmissions

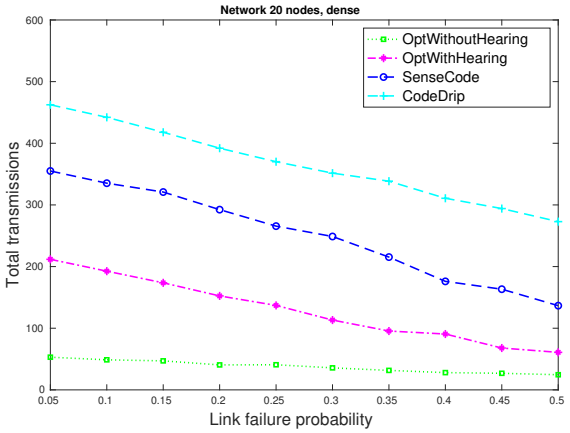
The number of packet transmissions for dense and sparse network topologies, with impact on energy and delay, are shown in Fig. 6.7 and Fig. 6.8, respectively. These include the 20 and 30 node topology cases, for gateways located at the border and center of the network.



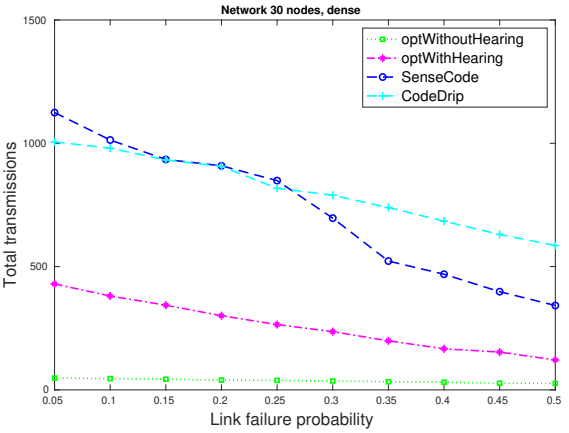

(a) Gateways at the border.




(b) Gateways at the border.



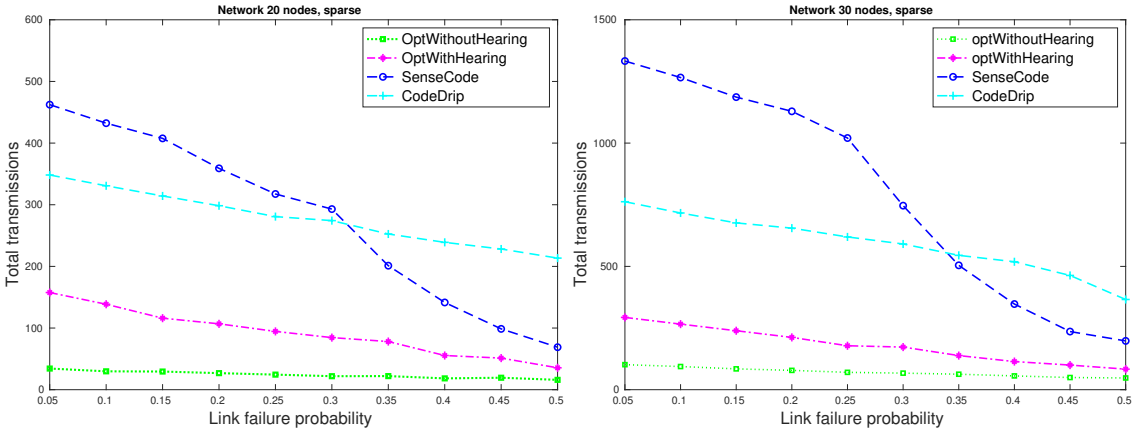

(c) Gateways at the center.




(d) Gateways at the center.

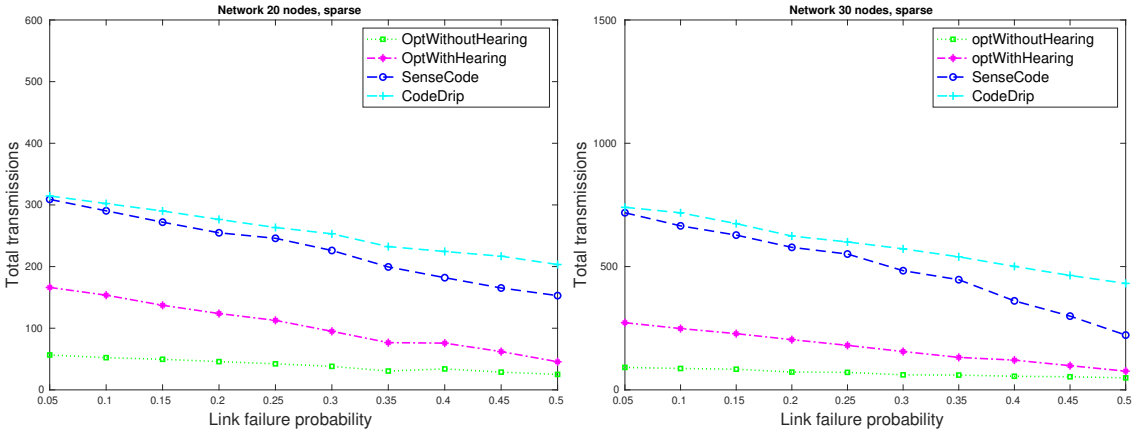
Figure 6.7: Total packets for dense topologies.

From plots it is possible to observe that in SenseCode and CodeDrip there are too many packet transmissions, when compared with DNC-WSN with hearing and no hearing. This is because all nodes are encoding nodes. Since such transmissions do not translate into more packet recoveries than DNC-WSN, these approaches seem not to provide the best tradeoff between packet recovery and energy saving. In SenseCode, the number of packet transmissions is lower when gateways are at the center, due to fewer hops, and packet transmissions reduce significantly when the link failure probability increases, leading to few packet recoveries. This is more evident in sparse topologies, and is basically related with the tree based routing




(a) Gateways at the border.

(b) Gateways at the border.




(c) Gateways at the center.

(d) Gateways at the center.

Figure 6.8: Total packets for sparse topologies.

towards a single gateway.

CodeDrip and DNC-WSN show a linear behaviour, due to diversity in routing. Link failures affect less traffic flows, meaning that their impact is not as drastic as in SenseCode. Still, CodeDrip performs much more packet transmissions than DNC-WSN because all nodes are encoding nodes, while in DNC-WSN only CHs perform encoding. The XOR operations in CodeDrip also involve just two packets, which means that there will be more coded packets when compared with linear encoding. Although the DNC-WSN with hearing shows more packet transmissions than its no hearing version, these are required for packet recovery, meaning that DNC-WSN

with hearing can be seen as the approach having the best packet recovery to energy saving tradeoff.

## 6.5 Discussion

In the previous section a comparison between the proposed DNC-WSN optimization model, SenseCode from [KAAF13], and CodeDrip from [JTV+17], is performed. In the proposed model, the data from sources can be protected against link failures by using overhearing (each source can hear its neighbors and send data to its CH). Furthermore, the data in CHs can be protected against link failures by using network coding (each CH encodes data from its members and sends both original and coded packets to CHs of higher topological order, or to the nearest gateway).

The version of DNC-WSN with better performance is the one with hearing since besides stability it shows the best tradeoff between energy saving and reliability. Its performance results from the fact that gateways act as peers in a P2P overlay network, allowing the recovery of packets even if their related coded packets have traveled towards different gateways. This lowers the required number of encoding nodes, for a certain recovery rate. In networks where failure probability is low, the no hearing variant may be more practical since there is less delay and more energy saving, leading to an increase of network lifetime.

## 6.6 Summary

In this chapter, a DAG-based dissemination approach, using both clustering and network coding techniques, to achieve a balancing between reliability and energy

efficiency is discussed. To solve the DAG-based network coding problem, a mathematical model is developed to select CHs and generate the DAG. These CHs, forming the DAG, are the only nodes in wireless sensor section that perform the encoding operations, while the other Non-CHs nodes perform just hearing. The performance evaluation shows that the DNC-WSN optimization model improves the network reliability, while reducing significantly the packet redundancy when compared to SenseCode and CodeDrip. The proposed DNC-WSN optimization model shows better results in both performance metrics: packet recovery and total number of transmissions.

# 7

# CONCLUSIONS AND FUTURE WORK

This thesis presents several contributions to the wireless sensor networks (WSNs) field. Two main objectives are considered as achieved in this thesis. First, to make these constrained sensor networks scalable by proposing an efficient way of making their data available in the internet. Secondly, to achieve reliability, low energy consumption and low cost, which are main concerns in WSNs. These goals have been achieved using network coding technique.

The thesis introduces a survey on WSNs that covers its definition, characteristics, limitations, components, and routing protocols used in such networks. This is followed by a survey on network coding, covering its definition, characteristics, and benefits over traditional routing. The corresponding related work is also discussed.

For WSNs at different regions to be able to communicate with each other and with the internet, an architecture for the federation of network coding based constrained

networks, using RELOAD overlay and CoAP Usage, is proposed. The goal is for encoding to be applied at the WSN section and decoding at RELOAD/CoAP overlay section. Such architecture proved to be a scalable and efficient way of storing sensing data, while allowing network coding to be applied at the wireless section for network efficiency increase. Packets reaching different gateways share a storage system that will allow recovery of lost data packets, even if the required linearly independent combinations have been forwarded towards different gateways.

Using network coding in constraints networks can bring additional costs, like energy consumption and computation overhead. Therefore, achieving a balance between energy efficiency and reliability in sensor networks is very critical. Here in this thesis, only a subset of network nodes are chosen to act as encoding nodes to achieve such balancing. For this purpose, a mathematical model and a heuristic algorithm are proposed to carefully select the best number and location of encoding nodes, under certain failure scenarios.

After, and in order to consider dynamic scenarios where failure places can be unpredictable, an additional approach is proposed, called DAG-Coder. To achieve reliability, considering the constraints and challenges of such dynamic contraint environments, the DAG-based dissemination approach uses both clustering and network coding techniques. In this approach, the selected cluster heads are the only nodes participating in the DAG and, consequently, the nodes doing the encoding process. Therefore, the number of generated coded packets is reduced, when compared with other approaches from the literature, saving more energy and reducing computation overhead.

Although the proposed methods present very good results, when compared with previous approaches from literature, there are some ideas that can be explored and analysed in future work. More specifically, in a given network, the node degree

(how many neighbors are connected to it) can be used as a criteria when selecting encoding nodes in Chapter 5. That is, higher benefit (or lower cost) can be given to nodes with higher degree, so that these have an higher probability of being selected as encoding nodes.

Some further work will also be focused on reducing the energy consumed and computation overhead in sensor nodes for an higher network lifetime. These goals can be achieved by:

- The impact of using binary coding (XOR) in DNC-WSN, which has lighter computation than linear random network coding, should be carefully analysed. This approach may still have some advantages in some deployments because energy consumption and computation overhead is low.
- Selecting carefully which packets undergo encoding process. A probabilistic approach can be adopted, but the probability to apply may vary from packet to packet. More specifically, the more innovative the packet is, the higher the probability of going through the encoding process should be.

# BIBLIOGRAPHY


[ACLY00] Rudolf Ahlswede, Ning Cai, S-YR Li, and Raymond W Yeung. Network Information Flow. *Information Theory, IEEE Transactions on*, 46(4):1204–1216, 2000.

[AHCB19] E. Al-Hawri, N. Correia, and A. Barradas. Design of network coding based reliable sensor networks. *AD HOC NETWORKS*, 91, AUG 2019.

[AKA+17] Muhammad Asif, Shafiullah Khan, Rashid Ahmad, Muhammad Sohail, and Dhananjay Singh. Quality of Service of Routing Protocols in Wireless Sensor Networks: A review. *IEEE Access*, 5:1846–1871, 2017.

[AL07] Muneeb Ali and Koen Langendoen. A Case for Peer-to-Peer Network Overlays in Sensor Networks. In *International Workshop on Wireless Sensor Network Architecture (WWSNA)*, pages 56–61, 2007.

[ASSC02] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless Sensor Networks: A Survey. *Computer networks*, 38(4):393–422, 2002.

[AV11] Angelos Antonopoulos and Christos Verikoukis. Network-Coding-Based Cooperative ARQ Medium Access Control Protocol for Wireless Sensor Networks. *Distributed Sensor Networks*, 8(1):601321,

2011.

[BEK14] C. Bormann, M. Ersue, and A. Keranen. RFC 7228: Terminology for Constrained-Node Networks. Technical report, IETF, 2014.

[BSS+10] Bhaskar Bhuyan, Hiren Kumar Deva Sarma, Nityananda Sarma, Avijit Kar, and Rajib Mall. Quality of Service (QoS) Provisions in Wireless Sensor Networks and Related Challenges. *Wireless Sensor Network*, 2(11):861, 2010.

[BZM11] Josip Balen, Drago Zagar, and Goran Martinovic. Quality of Service in Wireless Sensor Networks: A survey and Related Patents. *Recent Patents on Computer Science*, 4(3):188–202, 2011.

[CEE+01] Alberto Cerpa, Jeremy Elson, Deborah Estrin, Lewis Girod, Michael Hamilton, and Jerry Zhao. Habitat Monitoring: Application Driver for Wireless Communications Technology. *ACM SIGCOMM Computer Communication Review*, 31(2 supplement):20–41, 2001.

[CH03] Supriyo Chatterjea and Paul Havinga. A Dynamic Data Aggregation Scheme for Wireless Sensor Networks. 2003.

[CKP15] Seung-Man Chun, Hyun-Su Kim, and Jong-Tae Park. CoAP-Based Mobility Management for The Internet of Things. *Sensors*, 15(7):16060–16082, 2015.

[CSDC11] Walter Colitti, Kris Steenhaut, and Niccolò De Caro. Integrating Wireless Sensor Networks with The Web. *Extending the Internet to Low power and Lossy Networks (IP+ SN 2011)*, 2011.

[CSM+16] N Correia, Gabriela Schütz, Andriy Mazayev, J Martins, and A Barradas. An Energy-Aware Resource Design Model for Constrained Networks. *IEEE Communications Letters*, 20(8):1631–1634, 2016.

[CTF10] Nicolae Cleju, Nikolaos Thomos, and Pascal Frossard. Network Coding Node Placement for Delay Minimization in Streaming Overlays. In *International Conference on Communications (ICC)*, pages 1–5. IEEE, 2010.

[CWJ03] Philip A Chou, Yunnan Wu, and Kamal Jain. Practical Network Coding. 2003.

[dALFMA18] Francisco de Asis López-Fuentes and Javier Mendoza-Almanza. Dynamic Network Coding for Collaborative Multisource System. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 378–382. IEEE, 2018.

[DCX03] Min Ding, Xiuzhen Cheng, and Guoliang Xue. Aggregation Tree Construction in Sensor Networks. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 4, pages 2168–2172. IEEE, 2003.

[DP10] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of Wireless Sensor Networks: Theory and Practice*. John Wiley & Sons, 2010.

[Ele08] https://www.electricaltechnology.org/2018/11/types-sensors-applications.htmls, 11 2008.

[EMN+13] Milan Erdelj, Nathalie Mitton, Enrico Natalizio, et al. Applications of Industrial Wireless Sensor Networks. *Industrial wireless sensor networks: applications, protocols, and standards*, pages 1–22, 2013.

[EzAEO17] Imad Ez-zazi, Mounir Arioua, and Ahmed El Oualkadi. On the Design of Coding Framework for Energy Efficient and Reliable Multi-hop Sensor Networks. *Procedia Computer Science*, 109:537–544, 2017.

[EzAEOL17] Imad Ez-zazi, Mounir Arioua, Ahmed El Oualkadi, and Pascal Lorenz. On the Performance of Adaptive Coding Schemes for Energy Efficient and Reliable Clustered Wireless Sensor Networks. *Ad Hoc Networks*, 64:99–111, 2017.

[FL12] Chen Feng and Baochun Li. Network Coding for Content Distribution and Multimedia Streaming in Peer-to-Peer Networks. In *Network Coding*, pages 61–86. Elsevier, 2012.

[FLBW06] Christina Fragouli, Jean-Yves Le Boudec, and Jörg Widmer. Network Coding: An Instant Primer. *ACM SIGCOMM Computer Communication Review*, 36(1):63–68, 2006.

[GCS14] Reenkamal Kaur Gill, Priya Chawla, and Monika Sachdeva. Study of LEACH Pouting Protocol for Wireless Sensor Networks. In *International conference on communication, computing & systems (ICCCS-2014)*, 2014.

[GCSS14] Chao Gui, Hua Chen, Baolin Sun, and Ying Song. Energy Efficient with Network Coding Multipath Routing Algorithm in Wireless Sensor Networks. 2014.

[GKJ16] Suneet K Gupta, Pratyay Kuila, and Prasanta K Jana. Genetic Algorithm for k-Connected Relay Node Placement in Wireless Sensor Networks. In *International Conference on Computer and Communication Technologies (ICCCT)*, pages 721–729. Springer, 2016.

[HCB+02] Wendi B Heinzelman, Anantha P Chandrakasan, Hari Balakrishnan, et al. An Application-specific Protocol Architecture for Wireless Microsensor Networks. *IEEE Transactions on wireless communications*, 1(4):660–670, 2002.

[HL08] Tracey Ho and Desmond Lun. *Network Coding: An Introduction*. Cambridge University Press, 2008.

[HL17] Youying Hong and Xinhua Liu. Extension of Lifetime with Network Coding in Cluster Based Wireless Sensor Networks. In *2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 1219–1223. IEEE, 2017.

[HMS+03] Tracey Ho, Muriel Medard, Jun Shi, Michelle Effros, and David R Karger. On Randomized Network Coding. In *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, volume 41, pages 11–20. The University; 1998, 2003.

[HTAG08] I-H Hou, Y-E Tsai, Tarek F Abdelzaher, and Indranil Gupta. Adapcode: Adaptive Network Coding for Code Updates in Wireless Sensor Networks. In *International Conference on Computer Communications (INFOCOM)*, pages 1517–1525. IEEE, 2008.

[IGE00] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Proceedings of the 6th annual international*

*conference on Mobile computing and networking*, pages 56–67. ACM, 2000.

[IHVdA+14] Isam Ishaq, Jeroen Hoebeke, Floris Van den Abeele, Jen Rossey, Ingrid Moerman, and Piet Demeester. Flexible Unicast-Based Group Communication for CoAP-Enabled Devices. *Sensors*, 14(6):9833–9877, 2014.

[JBLR+14] C. Jennings, Ed. B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. RFC 6940: Resource Location and Discovery (RELOAD) Base Protocol. Technical report, IETF, 2014.

[JLR+16] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, H. Schulzrinne, and Ed. T. Schmidt. RFC 7904: A SIP Usage for REsource LOcation And Discovery (RELOAD). Technical report, IETF, 2016.

[JLVMC15] J Jimenez, J Lopez-Vega, J Maenpaa, and G Camarillo. RFC 7650: A Constrained Application Protocol (CoAP) Usage for Resource Location and Discovery (RELOAD). Technical report, IETF, 2015.

[JSHG15] Albert Jones, Eswaran Subrahmanian, Anthony Hamins, and Casey Grant. Humans' Critical Role in Smart Systems: A smart Firefighting Example. *IEEE Internet Computing*, 19(3):28–31, 2015.

[JTV+17] Nildo dos Santos Ribeiro Júnior, Rodrigo C Tavares, Marcos AM Vieira, Luiz FM Vieira, and Omprakash Gnawali. CodeDrip: Improving Data Dissemination for Wireless Sensor Networks with Network Coding. *Ad Hoc Networks*, 54:42–52, 2017.

[KAAF13] Lorenzo Keller, Emre Atsan, Katerina Argyraki, and Christina Fragouli. SenseCode: Network Coding for Reliable Sensor Networks. *ACM Transactions on Sensor Networks (TOSN)*, 9(2):25, 2013.

[Kar72] Richard M Karp. *Reducibility Among Combinatorial Problems*. Springer, 1972.

[KB17] Mustafa Kocakulak and Ismail Butun. An Overview of Wireless Sensor Networks Towards Internet of Things. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 1–6. IEEE, 2017.

[KCP14] Tae-hwa Kim, Hyungwoo Choi, and Hong-Shik Park. Centrality-Based Network Coding Node Selection Mechanism for Improving Network Throughput. In *International Conference on Advanced Communication Technology (ICACT)*, pages 864–867. IEEE, 2014.

[KK16] Bita Khodabakhshi and Mohammad Khalily. An Energy Efficient Network Coding Model for Wireless Sensor Networks. *Procedia Computer Science*, 98:157–162, 2016.

[KK18] Rajveer Kaur and Gurjinder Kaur. Load Balanced Clustering Protocol for Enhancing the Lifetime of Wireless Sensor Network. *International Journal of Advanced Technology and Engineering Exploration*, 5(46):326–334, 2018.

[KM03] Ralf Koetter and Muriel Médard. An Algebraic Approach to Network Coding. *Networking, IEEE/ACM Transactions on*, 11(5):782–795, 2003.

[KM14] Ahmed E Kamal and Mirzad Mohandespour. Network Coding-based Protection. *Optical Switching and Networking*, 11:189–201, 2014.

[KRH+08] Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Médard, and Jon Crowcroft. XORs in the Air: Practical Wireless Network Coding. *Transactions on Networking (ToN)*, 16(3):497–510, 2008.

[KW07] Holger Karl and Andreas Willig. *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2007.

[LC15] Alessandro Ludovici and Anna Calveras. A Proxy Design to Leverage The Interconnection of CoAP Wireless Sensor Networks with Web Applications. *Sensors*, 15(1):1217–1244, 2015.

[LRS02] Stephanie Lindsey, Cauligi Raghavendra, and Krishna M. Sivalingam. Data Gathering Algorithms in Sensor Networks Using Energy Metrics. *Parallel and Distributed Systems, IEEE Transactions on*, 13(9):924–935, 2002.

[LYC03] S-YR Li, Raymond W Yeung, and Ning Cai. Linear Network Coding. *Information Theory, IEEE Transactions on*, 49(2):371–381, 2003.

[MBL12] Jouni Mäenpää, Jaime Jiménez Bolonio, and Salvatore Loreto. Using RELOAD and CoAP for Wide Area Sensor and Actuator Networking. *EURASIP journal on wireless communications and Networking*, 2012(1):1–22, 2012.

[MC14] J Maenpaa and G Camarillo. Service Discovery Usage for REsource LOcation And Discovery (RELOAD). Technical report, 2014.

[MHXT10] Satyajayant Misra, Seung Don Hong, Guoliang Xue, and Jian Tang. Constrained Relay Node Placement in Wireless Sensor Networks: Formulation and Approximations. *Transactions on Networking (TON)*, 18(2):434–447, 2010.

[MI12] M Matin and M Islam. Overview of Wireless Sensor Network. *Wireless Sensor Networks-Technology and Protocols*, 2012.

[Mil07] Kevin L Mills. A brief Survey of Self-Organization in Wireless Sensor Networks. *Wireless Communications and Mobile Computing*, 7(7):823–834, 2007.

[MKQP01] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu, and Miodrag Potkonjak. Exposure in Wireless Ad-Hoc Sensor Networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 139–150. ACM, 2001.

[MMC+17] J. A. Martins, A. Mazayev, N. Correia, G. Schütz, and A. Barradas. GACN: Self-Clustering Genetic Algorithm for Constrained Networks. *Communications Letters*, 21(3):628–631, March 2017.

[MSS17] Nitin Mittal, Urvinder Singh, and Balwinder Singh Sohi. A Stable Energy Efficient Clustering Protocol for Wireless Sensor Networks. *Wireless Networks*, 23(6):1809–1821, 2017.

[NJ14] K. Nitesh and P. K. Jana. Relay Node Placement Algorithm in Wireless Sensor Network. In *International Advance Computing Conference (IACC)*, pages 220–225. IEEE, Feb 2014.

[NTNB08] Dong Nguyen, Tuan Tran, Thinh Nguyen, and Bella Bose. Wireless Broadcast Using Network Coding. *IEEE Transactions on Vehicular technology*, 58(2):914–925, 2008.

[OS07] Furuzan Atay Onat and Ivan Stojmenovic. Generating Random Graphs for Wireless Actuator Networks. In *International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–12. IEEE, 2007.

[OWK13] Pouya Ostovari, Jie Wu, and Abdallah Khreishah. Network Coding Techniques for Wireless and Sensor Networks. *The Art of Wireless Sensor Networks: Volume 1: Fundamentals*, page 129, 2013.

[PMEV00] A-S Porret, T Melly, CC Enz, and EA Vittoz. A Low-Power Low-Voltage Transceiver Architecture Suitable for Wireless Distributed Sensors Network. In *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, volume 1, pages 56–59. IEEE, 2000.

[SCI+01] Eugene Shih, Seong-Hwan Cho, Nathan Ickes, Rex Min, Amit Sinha, Alice Wang, and Anantha Chandrakasan. Physical Layer Driven Protocol and Algorithm Design for Energy-Efficient Wireless Sensor Networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 272–287. ACM, 2001.

[SDA+15] Jan Skodzik, Peter Danielis, Vlado Altmann, Bjoern Konieczek, Eike Bjoern Schweissguth, Frank Golatowski, and Dirk Timmermann. CoHaRT: Deterministic Transmission of Large Data Amounts Using CoAP and Kad. In *International Conference on Industrial Technology (ICIT)*, pages 1851–1856. IEEE, 2015.

[SDAT14] Jan Skodzik, Peter Danielis, Vlado Altmann, and Dirk Timmermann. Hartkad: A Hard Real-Time Kademlia Approach. In *Consumer Communications and Networking Conference (CCNC)*, pages 309–314. IEEE, 2014.

[SHB14] Z. Shelby, K. Hartke, and C. Bormann. RFC 7252: The Constrained Application Protocol (CoAP). Technical report, IETF, 2014.

[SSS10] Shio Kumar Singh, MP Singh, and DK Singh. A Survey of Energy-Efficient Hierarchical Cluster-Based Routing in Wireless Sensor Networks. *International Journal of Advanced Networking and Application (IJANA)*, 2(02):570–580, 2010.

[Sto05] Ivan Stojmenovic. *Handbook of Sensor Networks: Algorithms and Architectures*, volume 49. John Wiley & Sons, 2005.

[Ten16] Teng, Shang-Hua and others. Scalable Algorithms for Data and Network Analysis. *Foundations and Trends in Theoretical Computer Science*, 12(1–2):1–274, 2016.

[Tho04] Mikkel Thorup. Integer Priority Queues with Decrease Key in Constant Time and the Single Source Shortest Paths Problem. *Computer and System Sciences*, 69(3):330–353, 2004.

[TK03] Hüseyin Özgür Tan and Ibrahim Körpeoğlu. Power Efficient Data Gathering and Aggregation in Wireless Sensor Networks. *ACM Sigmod Record*, 32(4):66–71, 2003.

[VD10] Jean-Philippe Vasseur and Adam Dunkels. *Interconnecting Smart Objects With IP: The Next Internet*. Morgan Kaufmann, 2010.

[VMMdA+16] Odilson T Valle, Carlos Montez, Gustavo Medeiros de Araujo, Francisco Vasques, and Ricardo Moraes. NetCoDer: A Retransmission Mechanism for WSNs Based on Cooperative Relays and Network Coding. *Sensors*, 16(6):799, 2016.

[Yan14] Shuang-Hua Yang. Principle of Wireless Sensor Networks. In *Wireless Sensor Networks*, pages 7–47. Springer, 2014.

[YMG08] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless Sensor Network Survey. *Computer networks*, 52(12):2292–2330, 2008.

[ZAL+09] Jun Zheng, Nirwan Ansari, Victor OK Li, Xuemin Shen, Hossam S Hassanein, and Baoxian Zhang. Network Coding for Wireless Communication Networks. *IEEE Journal on selected areas in communications*, 27(5):577–581, 2009.

[ZJ09] Jun Zheng and Abbas Jamalipour. *Wireless Sensor Networks: A networking Perspective*. John Wiley & Sons, 2009.

[ZYY+15] Donghai Zhu, Xinyu Yang, Wei Yu, Chao Lu, and Xinwen Fu. INCOR: Inter-flow Network Coding Based Opportunistic Routing in Wireless Mesh Networks. In *2015 International Conference on Communications (ICC)*, pages 3666–3671. IEEE, 2015.