UNIVERSIDADE D
COIMBRA

Marco António Santos Silva

HEURISTIC AND SIMULATION OF ENERGY HARVESTING IOT NETWORKS

Dissertação no âmbito do Mestrado Integrado em Engenharia Eletrotécnica e de Computadores, no ramo de especialização em Computadores, orientada pelo Professor Doutor José Luís Santos e pelo Professor Doutor Hélder Araújo, apresentada ao Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Outubro de 2020

This page intentionally left blank.

1 2  9 0 UNIVERSIDADE D COIMBRA

Heuristic and Simulation of Energy Harvesting IoT Networks

Marco António Santos Silva

Dissertação para obtenção do Grau de Mestre em

Engenharia Electrotécnica e de Computadores

Orientada por: Doutor José Luís Esteves dos Santos
Doutor Hélder de Jesus Araújo

Júri

Presidente:	Doutor Aníbal Traça de Carvalho Almeida
Co-Orientador:	Doutor Hélder de Jesus Araújo
Vogal:	Doutor Jorge Miguel Sá Silva

Outubro de 2020

This page intentionally left blank.

”Faz pela tua conquista, nesta vida o protagonista és tu!”

- André Chapelas, Protagonista

This page intentionally left blank.

Acknowledgments

Em primeiro lugar, agradecer à minha família com especial relevo para os meus pais por terem permitido que a escrita desta dissertação fosse possível e por terem sido os melhores patrocinadores desta vida. Obrigado pelos valores que me passaram e que me permitiram formar como pessoa. Obrigado à minha mãe por me ensinar a ultrapassar sempre todas as adversidades com um sorriso na cara. Ao meu pai, obrigado por seres uma força da natureza que nunca recua perante um desafio. Eternamente grato, mesmo que por vezes não o demonstre. Ao meu avô Domingos, um agradecimento com muito carinho por ter acreditado sempre em mim desde que era pequeno, por me mostrar o valor da humildade, por todas as ajudas a fazer os trabalhos de casa e por me ajudar sempre a encobrir as aneiras dos meus pais. Tenho a certeza que estaria orgulhoso!

A todos os meus amigos de Lousada, obrigado por todas as memórias que trago comigo. Não posso individualizar muito os agradecimentos pois felizmente vocês são imensos e eu só tenho uma tese. Um especial obrigado ao Peixoto por ser o irmão que nunca tive, ao Ricardo por me acompanhar desde que tenho memória e à Pereira e à Vera por serem as melhores farmaceuticas e por todas as conversas.

A todos os meus amigos de Coimbra e de curso, obrigado pelo convívio e pelas vivencias durante estes anos. Que haja muitos mais! Um especial destaque para toda a gente com quem tive o privilégio de trabalhar no NEEEC/AAC e no Bot Olympics. Não posso deixar de destacar o André, o Bento, o Bruno, o Martins e o Moisés, pois ficarão sempre na memória pelo que me ensinaram e porque as nossas conversas acompanhadas de uma caneca certamente vão continuar. Ao Zé Pedro, obrigado pela companhia incansável durante a quarentena e por toda a ajuda a rever esta dissertação. Ao Reber e ao Paulo agradeço o privilégio de continuar a partilhar momentos e por não terem deixado que os rumos das vidas e a dispersão geográfica fosse um problema para os nossos convívios. Ao grande Carlão, obrigado por ser o melhor padrinho de curso que poderia ter escolhido. A ti e a toda a gente dos míticos jantares, obrigado! Ficarão na memória.

Um agradecimento ao pessoal da G6.7 pela forma como me receberam, pela entreajuda e pelo bom ambiente de trabalho criado.

Aos meus orientadores, Prof. Dr. José Luís dos Santos do Departamento de Matemática, Prof. Dr. Marília Curado do Departamento de Informática, obrigado por todo o aconselhamento e por toda a confiança ao longo deste trabalho. Foi um privilégio! Agradecer também ao Prof. Dr. Hélder Araújo, do Departamento de Engenharia Eletrotécnica pelo apoio prestado.

The work performed in this dissertation was realized within the scope of the MobiWise project: From mobile sensing to mobility advising (P2020 SAICTPAC/0011/2015), co-financed by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the project CISUC - UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020

A todos,
Muito Obrigado!

Abstract

The Internet of Things consists of a physical network that interconnects objects, vehicles, buildings, and all things that have communication capabilities. Many of the devices in the Internet of Things are battery powered and thus energy consumption becomes a critical aspect that needs to be tackled. This work aims to contribute to the energy efficiency within the Internet of Things context, with the focus on Energy Harvesting Internet of Things Networks.

These networks comprise devices that can extract energy from the environment and thus prolong the battery lifetime, while being able to perform more frequent and complex operations, including sensing the environment and transmitting the collected data. In such systems, the goal of having a "perpetual operational state" can be achieved through mechanisms that support Energy Neutral Operation. Creating a network that keeps the Energy Neutral Operation is a challenging task because it is crucial to ensure an overall network utility balance, according to the energy resources that the network operation requires.

Therefore, managing the energy level to avoid losing data and deplete the batteries can be achieved considering different features such as rate of data collection, event detection, data transfer and aggregation mechanisms, and network topology. To address this issue, this thesis has analyzed a multi-hop model that uses an enforced acquisition of spatiotemporal notable occurrences and configurable parameters, which can be specified according to the purpose of the network. It is based on a Mixed-Integer Linear Program approach in order to achieve an optimal operation of the Energy Harvesting IoT Network. As this model has limitations in performance and adaptation to more realistic scenarios, in this thesis, an heuristic was conceived to overcome the performance limitations of the optimal model.

The complete system for the heuristic was developed and tested using the' Cooja' software, which allows the creation of a sensor network at a high level while embedding its communication capabilities. One of the main objectives was to improve the solution obtained through methods that allow networks with many sensors to use realistic tools that approach the model's behavior to reality. The experimental results confirm and validate the mentioned theoretical model, allowing the formulation of a distributed heuristic in real-time that considers all the points already mentioned. Beyond this, the developed heuristic enables the user to define all the

network's functionalities without compromising the energy neutral operation.

Keywords

Internet of Things, Energy Harvesting, Energy Neutral Operation, Data Aggregation, Event Capture, Multi-hop Network.

Resumo

Redes de Internet das Coisas consistem numa rede física de objetos, veículos e prédios, dotados de tecnologia capaz de realizar transferência de dados. A maioria dos dispositivos de Internet das Coisas são alimentados energeticamente por uma bateria, por isso, o consumo de energia é um aspeto crítico que precisa de ser controlado e resolvido. Este trabalho visa contribuir para a eficiência energética no contexto de Internet das Coisas, com foco na energia proveniente de "Harvesting", que representa a energia possível de recuperar do meio ambiente.

Assim, estas redes são compostas por dispositivos com capacidade para extrair energia do meio ambiente, prolongando a sua vida útil no que concerne à bateria. Do mesmo modo, são capazes de realizar operações de uma forma mais recorrente e complexa, incluindo a captura de dados do meio ambiente e a transmissão destes entre vários dispositivos da rede. Neste tipo de sistemas, o objetivo é atingir um "estado operacional perpétuo", alcançado através de mecanismos que apoiam a Operação Energética Neutra. A criação destas redes é um dos desafios atuais. Deste modo, é importante garantir o equilíbrio geral da utilidade da rede de acordo com os recursos energéticos que as operações requerem.

Por conseguinte, para gerir o nível de energia de modo a evitar que as baterias esgotem a sua capacidade, o que provoca perda de dados, podemos desenvolver e aplicar diferentes soluções que contribuem para uma melhoria da eficiência energética. Alguns dos referidos trabalhos estudam a melhor periocidade para a recolha e transferência de dados de acordo com a topologia da rede. Outros, por sua vez, analisam a implementação da deteção de eventos e de mecanismos de agregação de dados. Para contribuir para a resolução deste problema analisámos um modelo teórico, com uma topologia de múltiplas etapas baseada em Programação Linear Inteira Mista que visava uma rede de Operação Energética Otimizada, considerando uma aquisição forçada de ocorrências espaço-temporais notáveis e um sistema com parâmetros configuráveis, que podiam ser especificados de acordo com o propósito da rede. Contudo, este sistema continha limitações de desempenho e de adaptação a cenários mais realistas. Assim, nesta dissertação, foi desenvolvida uma heurística especialmente concebida para superar as limitações de desempenho do modelo ótimo analisado.

O sistema para a heurística foi desenvolvido e testado com recurso ao software 'Cooja', que permite a criação de uma rede de sensores de alto nível e que integra as respetivas comunicações.

Um dos principais objetivos foi aperfeiçoar a solução obtida pelo modelo ótimo analisado, através de métodos que permitem a simulação de topologias de redes com um maior número de sensores. As ferramentas utilizadas concedem realismo ao sistema, pois aproximam o comportamento previsto do modelo à realidade que nos rodeia, fazendo com que o sistema fique pronto para uma implementação no mundo real, em especial nas características das tecnologias e protocolos de comunicação. Os resultados experimentais obtidos confirmam e validam os resultados do modelo teórico, permitindo a formulação de uma heurística distribuída no tempo real que considera todos os pontos já mencionados. Para além disto, a heurística permite ao utilizador definir todas as funcionalidades da rede sem comprometer o funcionamento da operação energética neutra.

Palavras Chave

Internet das Coisas, Energia de Harvesting, Operação Energética Neutra, Agregação de Dados, Captura de Eventos, Redes Multi-Hop.

Contents

This page intentionally left blank.

List of Figures

This page intentionally left blank.

List of Tables

This page intentionally left blank.

List of Algorithms

This page intentionally left blank.

List of Acronyms

CCA Clear Channel Assessment

EH-IoTN Energy Harvesting IoT Network

EH-WSN Energy Harvesting Wireless Sensor Network

ENO Energy Neutral Operation

HENO-IoT Heuristic for Energy Neutral Operation in Internet-of-Things

IEEE Institute of Electrical and Electronics Engineers

IoT Internet of Things

ILP Integer Linear Programming

LP Linear Program(ming)

LQI Link Quality Indicator

MILP Mixed-Integer Linear Programming

MAC Medium Access Control

MTU Maximum Transmission Unit

OPL Optimization Programming Language

OS Operating System

RPL Routing Protocol for Low Power and Lossy Networks

RSSI Received Signal Strength Indicator

RDC Radio Duty Cycle

WSN Wireless Sensor Network

This page intentionally left blank.

Theoretical Model Notation

Table 1 Network Symbols

Symbol	Description
$\mathscr{N}$	Represents the set of nodes (sensor nodes + sink node $\odot$). Each member has a sensor ID.
$\mathscr{S}$	Represents the set of sensor nodes.
$\odot$	IoT gateway in $\mathscr{N}$, called sink node. It does not have energy or buffer size limitations.
$\mathscr{L}$	Set of all possible wireless links (i, j) between the nodes $\mathscr{N}$ where all data is transferred.
W_{max}	Maximum capacity of a wireless link.
Δ	Fixed duration, in seconds, of each of the *time slot*s.
T	Operational *time-frame* that represents the finite-horizon period. The initial state of the network is $t = 0$. The model would find a solution for $1 \leq t \leq T$.

Table 2 Energy Symbols

Symbol	Description
$B_i(0)$	Battery level of node $i \in \mathscr{S}$ at the beginning of the simulation. This is also the battery value that the node would have when the *time-frame* operation ends.
B_{min}	Low battery level which can be reached during operation for any node.
B_{max}	Full battery level that can be achieved by any node during service.
$H_i(t)$	Amount of extended battery level of a sensor node $i \in \mathscr{N} \setminus \odot$ in a certain *time slot* t. This amount comes from the energy harvested from the environment.
$C_i(t)$	Total amount of reduced level of the sensor node battery $i \in \mathscr{N} \setminus \odot$ at *time slot* t.
$C^{\mathtt{z}}$	decreased battery level when in *sleeping* (`z`) state.
$C^{\mathtt{s}}$	decreased battery level when in *sensing* (`s`) state.
$C^{\mathtt{Tx}}$	decreased battery level when in *data transmitting* (`Tx`) state.
$C^{\mathtt{Rx}}$	decreased battery level when in *data receiving* (`Rx`) state.
$C_p^{\mathtt{Tx}}$	decreased battery level when transmitting one *data payload*.
$C_h^{\mathtt{Tx}}$	decreased battery level when transmitting one *header* (*data packet*).
$C_p^{\mathtt{Rx}}$	decreased battery level when receiving one *data payload*.
$C_h^{\mathtt{Rx}}$	decreased battery level when receiving one *header* (*data packet*).

Table 3 Events Symbols

Symbol	Description
$E_i(t)$	Binary variable that is 1 if node $i \in \mathscr{N} \setminus \odot$ is close enough to detect an *event* at *time slot* t. Otherwise, is 0.
$v(t)$	Binary variable that is 1 if an *event* has started at *time slot* t. Otherwise, is 0. This means that $v(t) = \max\{E_i(t) : i \in \mathscr{N} \setminus \odot\}$.
δ	Fixed duration, in *time slots*, of all emerging events.
p_E	Event probability.

Table 4 Operating nodes and data transmission

Symbol	Description
$\mathscr{M}$	Collection of all operating states of a single node $i \in \mathscr{N}$. $\mathscr{M} = \{\mathtt{z}, \mathtt{s}, \mathtt{Tx}, \mathtt{Rx}\}$. They are: sleeping `z`, sensing `s`, transmitting `Tx`, or receiving `Rx`.
$Q_i(0)$	At the start of simulation $t = 0$, each node $i \in \mathscr{N}$ has in the *buffer* 0 *data payloads*.
Q_{max}	Maximum number of *data payloads* that a sensor node can have in its *buffer* at any time.
P_{max}	Maximum number of *data payloads* that a transmitted *data packet* can contain.

Table 5 Decision Variables

Symbol	Description
$B_i(T)$	Percentage of the battery the node may have at the completion of the simulation.
$S_i^{\mathbf{m}}(t)$	Binary variable that is 1 if the node $i \in \mathscr{N}$ at *time slot* t is in the state $\mathbf{m} \in \mathscr{M}$. Otherwise, is 0.
$h_{i,j}(t)$	Number of *data packets* (*headers*) being transmitted over wireless link $(i,j) \in \mathscr{L}$ at *time slot* t.
$p_{i,j}(t)$	Number of *data payloads* being transmitted over wireless link $(i,j) \in \mathscr{L}$ at *time slot* t.
$a_{i,j}(t)$	Binary variable that is 1 if the link $(i,j) \in \mathscr{L}$ is active during *time slot* t.
$\bar{f}_{i,j}(t)$	Binary variable that is 1 if there is a *data packet* being transmitted over a wireless link $(i,j) \in \mathscr{L}$ at *time slot* t.
$Q_i(t)$	Represent how many *payloads* has in the *buffer* of node $i \in \mathscr{N}$ at *time slot* t.

1

Introduction

Contents

This chapter introduces the reader to an outline of this document in order to provide easy reading and context about the theme approached. Moreover, it summarizes the motivations needed to perform this work with the corresponding context in a realist application and supply the major goals and key contributions of all the tasks.

1.1 Motivation and Context

The Internet of Things (IoT) is an environment that associates all the physical objects to the internet regardless of their size or computational capability. It is crucial to improve society in several ways, providing many applications in diverse areas like Smart Grids, Industrial Internet or Smart Farming.

In IoT, a Wireless Sensor Network (WSN) incorporates a significant number of tiny devices, called sensor nodes, that are deployed in the sensing domain of interest to gather its data. It is a kind of internet extension that connects enumerable objects, independent of its computational capabilities. Hence, this connection represents an opportunity for the industry and the researchers.

Developing an architecture for the IoT is a hard and complex task due to the wide variety of sensors and services that may be involved in the system. In this context, every device can be equipped with microcontrollers, transceivers for communication, and appropriate protocols stacks that will enable them to interconnect all the instruments developed in the network [1].

One of the essential restrictions of IoT systems is related to energy consumption. Let us suppose that systems have to operate on a bound operational lifetime using conventional batteries. In this case, we need mechanisms that recover the battery level using sources available in the environment, such as solar energy harvesting. Besides, it is vital to use the available energy efficiently in a distributed application, finding a way to balance the application's performance with the consumption and available energy. This can be achieved by employing the mixed amount of performed work allocation to the nodes, which results in different consumption at multiple nodes. Under these circumstances, it is essential to adjust the performed work allocation with the node's energy availability. In some topologies, it is common to find the term "sink node", which represents a powered base station, used to collect and process data in a centralized way.

In the context of smart cities, improving services and quality of life for citizens, developing applications capable of meeting the needs of everyday life and the needs of all people, ensuring the cost of maintaining applications, is a challenging task.

This work approaches the problem of controlling the mechanisms related to production, aggregation, and communication of all the data to perform an Energy Neutral Operation (ENO) in a multi-hop Energy Harvesting IoT Network (EH-IoTN). This work also considers the importance of event detection.

Multi-hop routing is a type of communication in radio networks in which the network coverage area is more significant than the radio range of single nodes. Therefore, to reach some destination, a node can use other nodes as relays [2].

According to Kansal *et al.* [4], Energy Neutral Operation (ENO) represents the capabilities of a node operate in conditions when his energy consumption is inferior or equal to the energy harvesting available on the environment at each moment. ENO and Sustainable Network Operation are terms used to mean no node runs out of energy during an extended period, so the network lifetime prolongs indefinitely. Energy Harvesting IoT Network (EH-IoTN) can be defined as the network that keeps the state that avoids battery overflow and depletion. Both cases must be avoided as much as possible, but depletion is worse than battery overflow since it causes operation outage. From a general perspective, maintain long-term energy neutral operation is the first requirement for the current IoT networks, which means that sensor nodes must maintain a sustainable operation not only for some instants but throughout days. Enabling this resource reduces the maintenance costs in comparison with non-rechargeable battery-powered IoT networks.

This problem was analyzed in [3], proposing a mathematically constrained optimization model to find an optimal solution. However, this proposed work presents some limitations, such as the inability to solve significant problems, the inability to change the network topology at run time, and others explained the current thesis.

As the computational costs to compute the optimal solution of that model can be prohibitive, we offer a heuristic able to approximate the optimal solution within a reasonable time. This heuristic allows to deal with the limitations of the previous model and develops a realistic network that mimics the patterns found in the optimal solution obtained with the mathematical model.

The main task of this work was to construct, involve, and validate the proposed model in a realist scenario. For this reason, we made some computations to improve the performance of an IoT network, considering that we used software that gave us a realist computational performance and make the source code deployable.

1.2 Goals

The most significant purpose of this work is to present a heuristic that reaches an high level of global performance in EH-IoTN to achieve ENO on the whole network, taking in account all the characteristics that an IoT system is supposed to have.

The main objectives are sectioned as follows:

1. Study features and approaches in networking for energy-efficient self-managed networks and define what the model proposed solution accomplish;

2. Implement the heuristic in a realistic environment;
3. Create an autonomous multi-hop network using the heuristic solution;
4. Create a network of easy and quick configurations that allow the user to configure parameters such as data aggregation, event capture, and define the rates for sensing and communication to accomplish the network purpose;
5. Provide framework tool that allows to analyze the network behavior to understand the model simulation results to formulate a heuristic sub-optimal model.

1.3 Key Contributions

This thesis's research was part of an ongoing research project MobiWise that proposes highly efficient communications through the development of 5G in the context of smart cities.

One of the main contributions of this work falls into the development and implementation of an heuristic that mimics the behavior of the optimal solution obtained with the mathematical model. Additionally, it was implemented a framework that approaches the simulation to reality with several computation capabilities and turns the code deployable. Furthermore, an analysis platform was created to perform the result tests and explore all the validations.

A practical example that replicates the findings in this dissertation (Heuristic and Simulation of Energy Harvesting IoT Networks) and the corresponding source code can be found at the repository https://bitbucket.org/marcosantosilva/mobiwise/. More details are presented in appendix II.

One contribution of this dissertation was the submission of the work developed to the workshop:

- **EfS Research Day - Coimbra, May 29, 2019** that aims to demonstrate the ongoing research in the field of Energy for Sustainability.

Furthermore, some of the results will be submitted to an international journal with system review.

- **IEEE Internet of Things Journal** that solicits research papers describing significant and innovative research contributions to IoT communication and networking protocols. All the proposed model and the implemented heuristics will be submitted - see Appendix I. In this appendix, a preliminary version of the submission to be made in November 2020 is represented.
  - M. Silva, J. Torrado, A. Riker, J. Santos, M. Curado, "Extending Energy Neutral Operation in Internet-of-Things", IEEE Internet of Things Journal (awaiting submission).

1.4 Document Structure

This dissertation is divided into seven chapters. The current chapter provides the motivations, goals, and the contributions of this work. Chapter 2 presents the research background and related works. Chapter 3 reports the theoretical model and the correspondent overview, while in Chapter 4 it is explained with the practical implementation of this model and all the changes that we need to do according to this new study environment. Chapter 5 describes the experimental setup, and Chapter 6 discusses the results obtained. Finally, Chapter 7 concludes this dissertation and suggests some future works. Appendix I exposes the draft of the journal that will be submitted to IEEE Internet of Things Journal, and appendix II explains how to compile all the developed code.

This page intentionally left blank.

2

Background

Contents

This chapter shows some relevant scientific works related to IoT, making the proper framework and analogy to our work in order to clarify the purpose and objective of the innovative part of this work.

2.1 Concept

The need to have a group of geographically dispersed sensors communicating with each other, with a high capacity for gathering information from the environment and allowing the construction of increasingly demanding applications, led to wireless networks.

Wireless Sensor Network (WSN) is considered as one of the most critical components from the near-surface of Global Earth Observation. It is expected that WSN becomes a standard in terrestrial applications to enable rapid progress in science and environmental systems [4].

Energy Harvesting Wireless Sensor Network (EH-WSN) is the next step for WSN because it will provide means to enhance the lifetime of wireless sensor nodes considerably. In a realist developed environment, the sensors are geographically distributed, and they need to interconnect. Therefore, the harvesting available on the environment can be different for each node so that it may cause significant constraints for each sensor, and the network should be able to deal with all the situations. Different harvesting levels cause additional performance capabilities that can delay a sensing operation or information transmission on the network. The most common energy harvesting sources are:

- **Radiant Energy -** Harvested from the sun and radio-frequency waves. In recent years, energy harvesting from radio-frequency and sunlight has been explored extensively in an attempt to utilize this energy source to power wireless sensor networks for environmental monitoring applications [5];
- **Mechanical Energy -** Another source of energy that can be harvested to power WSN is mechanical oscillations or vibrations. Typically, mechanical energy is harvested using piezoelectric devices [5, 6];
- **Thermal Energy -** Thermal energy harvesting is based on the fact that when there is a temperature difference between two conductive materials, an electric current is generated [6]. Many environmental studies on thermal powered sensor nodes have been conducted throughout the years [5];
- **Wireless Energy Transfer -** Energy transmission through electromagnetic fields between sensors [7, 8].

One of the particular interests in this work is WSN used for environmental monitoring in the context of smart cities. Battery-powered IoT devices are largely deployed as enablers for buildings control, smart-homes, and smart-cities. The advances in battery technology and related

hardware have allowed these IoT devices to capture and store energy from the environment efficiently. Thus, they can replenish their batteries, as long as there is energy to be harvested from sustainable sources, such as solar, thermal, and vibration.

This environment consists of a group of sensors nodes, distributed in the surroundings, that aggregates and stores the data collected from the sensors to be transmitted into a local sink node. In this topology, sensor nodes are low cost and operate with batteries. The sink node is powered and receives, analyzes and stores data, so that in the future, it can be transmitted to a remote server [9, 10].

In these networks, the sensor node architecture is composed of one or multiple sensing units, a radio transceiver, a processing unit, an energy harvester, one or more energy storage units, a power management system, and possibly an energy predictor [5, 11] so, sensor nodes have some ways to extend the lifetime of the batteries, using harvesting techniques. This is quite important because, in some cases, these sensors are in a difficult place to change the batteries regularly. Therefore, by dealing with these energy issues, we can create an autonomous system that can be used for environment monitoring [12, 13], keeping track of disasters such as earthquakes, fires, hurricanes [14–16], and health monitoring of civil structures such as bridges and buildings [17, 18].

Supplying energy to the entire network of sensors and actuators presents essential design and optimization challenges, which have led to extensive research in recent years. To optimize energy efficiency, we can look for solutions that go through optimizing the activation policies of the sensors and improving the data transmission and aggregation algorithms to maximize the entire network performance under energy allocation restrictions [5, 8]. These networks play a crucial part in achieving the ambitions of many rising concepts, such as Smart Cities, Smart Grids, and the Industries of IoT.

2.2 Related Work

The constant increase in the number of devices connected to WSN raises questions regarding these devices' energy capacity. The number of devices and the amount of data to be processed over the system tends to increase substantially.

Most of the works related to EH-IoTN present a study to create a smart network in terms of Energy Efficiency where all the sensors in the network achieve sustainable operation i.e., no sensor node runs out of energy, based on IoT technology. These techniques promise to compensate for the major network problems, batteries lifetimes, following some approaches:

- **Minimum-Energy Multi-hop Routing -** This technique is a conventional clustering approach to routing using an energy-efficient routing protocol to select the routes that maximize the minimum lifetime of the nodes and consume minimal global energy [19, 20].

The selection of an appropriate well-designed routing protocol increase transmission efficiency and minimize individual transmissions [21, 22];

- **Data Processing -** Each node in the network, having the ability to choose adaptively and dynamically a set of system parameters, such as duty-cycling, sense rate, transmission, power, data processing, or event capture, can achieve greater capacity and performance the network. These parameters need to be chosen carefully in each cycle, always taking into account the next recharge cycle's values to enhance the network life cycle [23–25];

- **Energy Design -** Sensor networks with a battery-powered node may have protocols such as MAC or RDC protocol, that estimate and meet the design goals of energy cost by the sensor and wireless transceivers in a sensing and data communication [6, 26];

- **Energy-aware systems -** By seeking to explore battery charging opportunities to adjust the performance of sensors, based on current and expected values, nodes can solve possible power problems that hinder their performance and service life [6, 27].

ENO, in literature, is a synonym of Sustainable Network Operation, which means that no nodes run out of energy, so the lifetime of the network could be infinity. To study how to optimize network performance achieving sustainable operation, some related works, focus on presenting both theoretical and practical studies similar to this work - see Table 2.1. They propose routing mechanisms and scheduling operations to deal with performance constraints and maximize the major goal - the lifetime of the network.

According to Zareei *et al*. [28] one scheme for transmission power that makes the energy neutral possible consists of increasing the transmission range of devices that have abundant energy compared to their neighbour nodes. Nodes in critical energy conditions normally avoid performing any communication. This proposal does not ensure an ENO because, with these techniques, it allows nodes to reach critical energy conditions. The work improves the end to end performance while maintaining a regular event sensing rate but does not consider any compression or data aggregation in the data transmission.

Yang *et al*. [29] also present solutions for ENO. Authors consider the nodes can execute one of the following actions during a time-slot: collect raw data readings, process data, transmit or receive data packets. This work formulates ENO as a finite-horizon stochastic optimization problem. At each time-slot, the decision variables are sensing rate, wireless transmission rate, and data aggregation. The sensing rate controls the amount of network traffic that will be produced. The wireless transmission rate is associated with energy communication costs. Data aggregation is a means to perform in-network control on the network traffic. In addition to the formulation, the authors also present a distributed and sub-optimal algorithm. Each node decides its sensing rate, data aggregation, and computes a scheduler for wireless transmission

based on the lightweight Longest Queue First. This solution was implemented on a testbed, showing improvements in terms of energy sustainability.

Jackson *et al*. [30] intends to maximize the network's operational performance through the battery life of each node present in the network by an optimization problem related to the degradation of the battery capacity. This work is focused on maintaining a network of neutral operation in each sensor node. However, multihop routing is not addressed. The aggregation of data and the existence and capture of events are also not mentioned in this work. This work proposes a battery model in which its capacity suffer degradation over time.

The work proposed by Le *et al*. [31] focus on adapting the duty-cycling to achieve ENO. The proposed solution is a protocol that was developed for periodically powered indoor IoT devices. The protocol applies the Zero Energy Interval Predictor introduced in [32] to estimate the harvesting and non-harvesting periods. After estimation, it adapts the node's duty cycle according to its residual energy in the non-harvesting periods. Also, the protocol saves a portion of the harvested energy for non-harvesting periods.

Some other approaches apply data regulation, such as data-aggregation, to achieve ENO. This is the case of the work proposed by Gao *et al*. [33]. In the presented data-aggregation solution, each node decides its aggregation percentage considering the energy in its reserve. A node always begins its operation performing the lowest aggregation level and changes it gradually if the battery energy increases.

Jeong *et al*. [34] propose a solution to achieve ENO by controlling data production and managing the transmissions. In this solution, before the transmitted message is processed, the network's devices estimate their remaining energy. If the energy reserve is low, which means the nodes may deplete energy, they avoid data production and transmissions. When the estimated residual energy is high, then the node transmits data. Besides, the devices try to avoid energy wasting, decreasing the occurrence of battery overflow.

To characterize a multi-hop system for utilization in bridges, Gaglione, Rodenas-Herraiz *et al*. [35] developed a system that achieved Energy Neutral Operation in a 4-hop sensor network. However, in this system, vehicle traffic is transient in nature, as in events, so this information is applied to the energy harvesting model and not to the data utility performance. This work also does not recognize data aggregation.

Dehwah *et al*. [36] constructed a system to monitor flood occurrences, creating a policy to optimize the remaining energy in a solar-powered wireless sensor network. This is a multi-hop system that shows good practical results with management policies in a routing scheme. This work considers events (i.e., floods) and energy optimization, but it assumes that no data compression or aggregation is available.

Seeking ENO, some works are focused on the trade-off between event-detection rate and

energy consumption. This tradeoff exists because spending more time sensing the environment enables the nodes to detect the event, decreasing the statistics of not detected events. Zhu *et al*. [37] proposed to control the wake-up and sleep intervals to seek a high rate of event detection and low energy consumption, considering nodes with non-replenishable batteries. Yau *et al*. [38] addressed a similar problem but considering rechargeable batteries. Correia *et al*. [39] and Sacramento *et al*. [40] seek to optimize mainly the energy consumption by applying cache and aggregation schemes on the observe CoAP traffic [41].

Table 2.1 presents a summary of the characteristics analyzed in the principal works referred in this section. As we can see, this work innovates the study of EH-IoTN, taking care of all the essential attributes in a combined way.

Table 2.1 Related Works

Features	[28]	[29]	[30]	[31]	[33]	[34]	[35]	[36]	This Work
ENO	no	no	yes	yes	yes	yes	yes	no	yes
Event Capture	yes	no	no	no	no	no	no	yes	yes
Data Aggregation	no	yes	no	no	yes	yes	no	no	yes
Multi-hop Network	yes	yes	no	yes	no	no	yes	yes	yes

One of the most utilized ways to deal with legitimacy on proposed methods enveloping the points referenced above is defining an optimization problem. Next, suggest an algorithm that can take care of that issue. Thus to this work, a portion of these methods (see Table 2.1) propose routing systems and activities planning to augment accumulated system utility while managing performance constraints. Harvesting IoT Networks is the following stage of the innovative advancement in this field: they join the nearly unending detecting capacities of EH-WSNs with the capabilities of interact with the earth.

It is conceivable to deduce from the general description above (and from Table 2.1) that this work examines these four fundamental IoT main strategies: ENO of a multi-hop coordinate with data aggregation and environmental events capture. Supposedly, this group of four improvements has not been concentrated in one single study yet.

2.3 Research Scope

This work performed at Center for Informatics and Systems of the University of Coimbra, presents a similar line as the works shown in the previous Section 2.2 and it is the follow-up of the work started by Joel Torrado [3] that had some limitations which have now been overcome as well as new features have been introduced.

The main goal of this work is to create a EH-IoTN, which validates the mathematical model presented in [3] and demonstrates that it can be used in a real context, so that no sensor node runs out of energy, by using an energy-neutral scheduling algorithm in a discrete finite-horizon

network. To accomplish this, we studied finite capacity batteries with harvesting capabilities to create a EH-IoTN. Limited capacity batteries power the nodes; still, they are capable of harvesting energy.

Following up on the points covered in Section 2.2, all the main issues for the smooth functioning of the IoT network were explored. In each timeslot, all sensors in the network can decide whether they can introduce new data, (i.e., sensing) and whether they can admit new data from other sensors.

2.3.1 Requirements

The network has the capability to manage the new data that is generated and to manage all communications that are made across the entire network. Sense operations are performed whenever conditions are met and according to a parameter defined in the network, which determines the maximum time allowed between sense on the same node. This parameter was introduced to infer that the network captures information periodically. Similarly, communications are carried out when data is sent to other nodes on the network, always ensuring that they go to the sink node. Creating data on the network is not always as important. The implementation of stochastic spatio-temporal occurrences designed events that are occurrences that may happen during the simulation and that were not initially foreseen, which has higher importance concerning periodic network data collection.

Regarding the ENO scenario requirement, IoT networks must be able to transport data traffic with heterogeneous characteristics. In general, this traffic can be divided into periodic and event-based traffic. Periodic-based traffic is a low-priority data generated following a fixed time interval pattern, and it is a permanent communication for a long time window. In opposite, event-based traffic is a high priority data triggered according to a stochastic occurrence and has a limited duration. To deal with such traffic it is crucial to keep the IoT networks in ENO because they coexist and compete for resources inside the network. For instance, a monitoring application might be implemented to receive data report regularly regarding a facility structure and also to gather data when some critical event occurs, such as smoke detection. Therefore, to achieve ENO in the IoT network, it is required to efficiently use resources and manage the communications aspects for both periodic and event-based traffic.

To minimize the cost of carrying out transmission and reception operations, data aggregation has become a fundamental factor, as a mean to perform in-network data processing control of the traffic and a way to minimize energy consumption once several data payloads can be integrated into the same data packet.

IoT networks produce small payloads, containing short information [42]. However, due to the number of IoT devices, the amount of generated traffic can represent a challenge for the ENO network. Thus, the third requirement for ENO IoT is to apply mechanisms to control

the amount of traffic flowing over the wireless connections. Otherwise, the network resources will become scarce at some time, which would break the ENO. Traffic control to prolong the ENO can be achieved by efficiently controlling the sensing operation and regulating, using data aggregation, the number of forwarded messages.

In addition, the nodes always know the energy restrictions and the battery capacity of all neighbors, which makes it possible to carry out communications along with the network always towards the sink, on a path where the energy conditions are not critical, contributing to a better energy balance between all nodes and causing the critical energy zone to be delayed in all sensors. Therefore, the description of a present network influences the performance that it may have in the future (Minimum-Energy Multi-hop Routing).

Table 2.2 presents a summary of the three IoT requirements to achieve ENO introduced so far.

Table 2.2 Requirements Summary

	Requirements
1.	To enable long-term ENO to achieve a sustainable network operation for a long period of time.
2.	To efficiently manage network's resources to support both periodic and event-based traffic in ENO IoT networks.
3.	To control the amount of traffic generated and communicated by the IoT network, extending ENO.

Given the current requirements, energy efficiency is present in all fields of new technologies and, therefore, the development of a network of ENO has led to the creation of a series of protocols and heuristics to allow achieving this end. This creates some restrictions in the network and the parameters that define the periodicity of the sense and the transmission. Therefore, to perform a ENO, sometimes we have to consider a trade-off between some services' performance in the most critical moments of the network and each sensor's lifetime and the system itself. At all time in the network, the sensor have are aware about state operations costs and can predict the harvesting accumulation that they will receive until the end of the simulation. This allows all sensors to have energy awareness to achieve very realistic energy profiles that always obey all restrictions to which they are subject - Energy-aware system.

2.3.2 Proposed Solutions

Many optimization models have been proposed in the literature to fulfill requirement 1 using deterministic and stochastic models. However, most of these works do not consider requirements 2 and 3. It means that the network does not support periodic and event-based traffic and does not optimize traffic production, communication, and aggregation to prolong ENO.

This work presents an distributed heuristic that seeks to fulfill the requirements 1 - 3. This work has the following main contributions:

1. It describes the Integer Linear Programming (ILP) model proposed in [3] and introduces a distributed heuristic to achieve ENO in IoT networks;
2. It manages communication aspects to support periodic and event-based traffic;
3. It regulates the production and communication of network traffic using energy-aware data aggregation.

Thus, it is essential to mention that this study seeks to jointly investigate these main issues of IoT, contrary to what happened in the works presented in Section 2.2. With this dissertation, we intend to create all the necessary bases for implementation in a realistic environment of all the developed algorithms. For this, we use a network sensor emulator so that all applications are carried out under a realistic scenario which is always ready to implement it in a real situation. This implemented solution show significant improvements in terms of energy sustainability.

2.4 Final Notes

In this chapter, the works related to our proposal were presented. Most relevant approaches of the state of the art were analyzed as well as the advantages and disadvantages of each one. This was the only way we were able to position ourselves correctly and make our work fit the world's current needs. After this analysis, we managed to gather, in a single study the most critical points regarding actual Wireless Sensor Network.

In the next chapter, we will describe the mathematical model proposed in [3], representing the limitations and guidelines that the model must follow to overcome the needs presented during this chapter.

This page intentionally left blank.

3

Theoretical Model

Contents

The theoretical model used throughout the work was initially developed by Joel Möllering Torrado [3] to bring together, in a single model, all the necessary points to obtain an ENO network. However, this model had some limitations that prevented its implementation in the real-world. Therefore, the base model was refined during the execution of this work. Some readjustments to the new reality and development environment were made. Consequently, the identified weaknesses have been fixed, and new features implemented.

The work presented in [3] used CPLEX Optimization Studio Engine[1] to find the optimal solution of the proposed model. This optimizer is quite slow to get results due to the combinatorial part of the model, which produces an exponential number of admissible solutions. In order to solve this issue, it was developed heuristics that, for large networks and long simulation times, analyze all the system's expected behavior in a short time.

Thus, this chapter presents the entire theoretical model developed and the most significant limitations and changes made to the construction of the heuristic.

3.1 Network Model

An energy harvesting network consists of a set of statically-deployed nodes $\mathscr{N}$ that bring together sensor nodes $\mathscr{S}$ and a single IoT gateway (sink node) $\odot$. So $\mathscr{N} = \mathscr{S} \cup \odot$. The network initially created is static, so we cannot change the sensor node position at run time. In our heuristic, we create a dynamic network by quickly removing or adding a sensor to adapt to a real situation. The network itself can automatically adjust to this change, researching their neighbors and updating the sink distance. Therefore, the system does not have any mandatory topology, i.e., a static network can become a dynamic one.

In order to create a network with multi-hop characteristics, each sensor can make a different set of wireless connections, according to what is best for the system at any moment. So $\mathscr{L}$ represents the set of all possible wireless links available at each moment that are directed, though we assume connectivity to be symmetric, i.e., link $(j,i) \in \mathscr{L}$ if and only if $(i,j) \in \mathscr{L}$. Let $N \geq 2$ be the order of the graph (its number of nodes), i.e., $N = |\mathscr{N}|$; and let $L \geq 1$ be the size of the graph (its number of links), i.e., $L = |\mathscr{L}|$. Let $\mathscr{N}_i \subseteq \mathscr{N}$ be the set of all one-hop neighbors of each node $i \in \mathscr{N}$. We can see a network example in Figure 6.1.

The network operates in a finite-horizon period consisting of discrete time slots, i.e., $t \in \{0,1,2,\ldots,T\}$, $T < \infty$, where $t = 0$ is the initial state of the network. Each time slot has a fixed time span Δ, where all the operations related to a state is entirely executed.

There are four potential conditions for a sensor node in a given time slot: sitting idle (sleeping) `z`, taking raw sensor readings (sensing) `s`, transmitting `Tx` or receiving `Rx` data packets over a wireless link. Thus, the set of states is defined as $\mathscr{M} = \{\mathtt{z},\mathtt{s},\mathtt{Tx},\mathtt{Rx}\}$. A binary variable $S_i^{\mathbf{m}}(t)$

[1] https://www.ibm.com/products/ilog-cplex-optimization-studio

indicates if the sensor node i at time slot t is in the state $\mathbf{m} \in \mathscr{M}$ or not, i.e., $S_i^{\mathbf{m}}(t) = 1$ and $S_i^{\mathbf{m}}(t) = 0$, respectively.

Thus, according to what happens in [3], a node may only be in one operating state at each slot moment, as seen in the following Equation 3.1.

$$\sum_{\mathbf{m} \in \mathscr{M}} S_i^{\mathbf{m}}(t) = S_i^{\mathtt{z}}(t) + S_i^{\mathtt{s}}(t) + S_i^{\mathtt{Tx}}(0t) + S_i^{\mathtt{Rx}}(t) = 1 \qquad (3.1)$$
$$\forall i \in \mathscr{N},\ 1 \leq t \leq T$$

In our work, we changed this constraint to introduce greater versatility and speed in the network since, in a single time slot, sensor nodes can do all the operations that they deem most necessary at that moment. For example, in the same time-slot, the nodes can carry out the readings of their sensors, receive new packets over wireless connections and, in the end, send all this new data to the next sensor, which improves the data latency in the network because it has performed more than one operation in a single time slot. So, a node should operate in all states at the same time slot if he needs to - see this change in Chapter 4.2.

In this model, it is considered that the sink node $\odot$ does not have any type of energy restriction, state of operation, or computing capacity to storage data - infinite buffer size. Contrarily, all the sensors nodes $\mathscr{S}$ have harvesting devices and batteries, which causes computational constraints to get some achievements. For simplicity reasons, all the nodes are in the same initial conditions and have the same characteristics.

3.2 Data and Communication

In order to build a network analyzing the main essential factors in EH-IoTN, in a single work, it is necessary to consider all the particularities and changes that the network can affect in, like described below.

1) Wireless Data Communications - Transferring data via wireless communications is a way of sending data using the radium spectrum instead of a physical medium of transfer. Wireless data transfer services represent an essential growth in the communications segment that is renewing the industry [43]. When associated with applications where transfer terminals are objects that can move geographically, this data transfer service raises the problem that the transfer/reception signal may vary. Therefore, security mechanisms for data transfer are essential to guarantee that no data is lost during communications.

For that reason, we consider a model with the following interference in a wireless link communication: we can only transmit and receive, at the same time, if the link does not share a common node, i.e., links that share a common node cannot transmit and receive simultaneously. In this case, the sensor has a schedule that ultimately allows operating one communication before starting another one.

Consequently, a binary variable $a_{i,j}(t)$ is used to indicate if a link $(i,j) \in \mathscr{L}$ is active ($a_{i,j}(t) = 1$) at time slot t. This activity exists only when node i is ready to transmit (`Tx` state) and j is ready to reception (`Rx` state) during that time slot, i.e.,

$$S_i^{\texttt{Tx}}(t) + S_j^{\texttt{Rx}}(t) \geq 2 \cdot a_{i,j}(t), \quad \forall (i,j) \in \mathscr{L}, 1 \leq t \leq T \tag{3.2}$$

Equation 3.2 is presented as inequality and non-equality since, in the same time slot, the model allows a node to be able to send data to one or more neighbors that are ready to receive, in the same way, a node can receive data from one or more neighbors who are prepared to transmit. System operates in a synchronous time-slotted mode where the length of each time slot is Δ seconds. We assume that a link can transmit at a fixed data rate of ψ bits/second. Therefore, a link active during one time slot can achieve a maximum transmission of $\Delta \cdot \psi$ bits in that time slot. On the previous model, presented in [3], a minimum data packets that need to reach the sink at each time slot is defined. This constraint is created precisely because the sensor node can only operate in a single state at a time. However, as we explain in the previous point, the proposed heuristic does not have this limitation, and this restriction will be removed.

2) Network Data Packet - Network data packet loads and helps information to reach its destination. The data packet is assembled into two parts: header and body. Therefore, a network data packet can be considered a data unit that can transport a defined number of bytes. The packet header contains information about the network protocol, while the body contains the data that will be sent from the sender to the receiver. This data consists of one or more payloads. In our model, the header's size is considered constant, and the size of the body varies according to the number of payloads to be transported. The packet body must have at least one payload and load up to a maximum of P_{max} payloads.

Let $p_{i,j}(t)$ and $h_{i,j}(t)$ be the integer number of payloads and headers (data packets) for a successful transmission over the wireless link $(i,j) \in \mathscr{L}$ at time slot t. In our model, it is possible to establish the relationship between the number of data packets and payloads involved in one transmission according this two parameters.

Considering $p_{i,j}(t)$ and $h_{i,j}(t)$ the number of payloads and headers, respectively, for a successful transmission over the wireless link $(i,j) \in \mathscr{L}$ at time slot t, we can model this relationship such as

$$h_{i,j}(t) = \left\lceil \frac{p_{i,j}(t)}{P_{max}} \right\rceil, \quad \forall (i,j) \in \mathscr{L} : a_{i,j}(t) = 1, 1 \leq t \leq T \tag{3.3}$$

Equation 3.3 computes the minimum number of necessary headers to send the number of payloads $p_{i,j}(t)$.

3) Capacity and Cost of Transmissions - All networks have the capacity and cost limitations on their transmissions. Hence, let W_{max} be the maximum integer number of data packets that can be successfully transmitted from i to j during any time slot, that is, the constant maximum capacity of a wireless link $(i,j) \in \mathscr{L}$ for any t. So,

$$h_{i,j}(t) \leq W_{max}, \tag{3.4}$$
$$\forall (i,j) \in \mathscr{L} : a_{i,j}(t) = 1, 1 \leq t \leq T$$

Regarding that, since the proposed heuristics work with real communications between sensors, P_{max} it should be established according to the Maximum Transmission Unit (MTU). Thus, there must be a relationship between the size of the payloads allowed per packet P_{max} and the maximum that wireless communications can transmit in a single transfer W_{max}, as we further explain in Section 6.1.

The increasingly small size of the IoT devices for the next generation has made the topic of energy consumption in wireless communication protocols very important. This reduction in device sizes implies a decrease in battery capacity, which means that wireless systems need to use the battery more efficiently. Considering $C_p^{\mathtt{Tx}}$ and by $C_h^{\mathtt{Tx}}$ a fixed energy cost for successfully transmit a payload and a header, respectively. In the same way, let $C_p^{\mathtt{Rx}}$ and by $C_h^{\mathtt{Rx}}$ be the corresponding cost for a node to successfully receive a payload and a header. Given this, we can obtain the overall energy cost of transmitting/receiving $p_{i,j}$ payloads together with $h_{i,j}$ headers over a wireless link $(i,j) \in \mathscr{L}$ at a time slot t as:

$$C_p^{\mathtt{Tx}} \cdot p_{i,j}(t) + C_h^{\mathtt{Tx}} \cdot h_{i,j}(t) \tag{3.5}$$

and

$$C_p^{\mathtt{Rx}} \cdot p_{i,j}(t) + C_h^{\mathtt{Rx}} \cdot h_{i,j}(t) \tag{3.6}$$

4) Sensing Model - Considering that in each time slot, a node $i \in \mathscr{S}$ can meet conditions to be in the sense state, it collects raw data readings from a hardware sensor. The consumption of this operation must be foreseen in the model. Thus, we denote the total energy cost of the sense operation of a node as a constant $C^{\mathtt{s}}$. It is considered that all sensors in the network have the same type of hardware and, therefore, consumption will be the same in all of them. Each sense generates the creation of a payload with the readings from the sensors ant it is stored in the buffer.

5) Data Aggregation - Generally, the sensor has limited storage capabilities. To avoid buffer packet overflow and to reduce the latency on the network, the node receives data from multiple nodes, like an intermediate, and sends its aggregated information to the sink node. Therefore, we efficiently explore the bandwidth of communication.

The work presented in [3] implicit assumes that some underlying data aggregate mechanism exists to perform this operation in each buffer without any energy cost. This model also considers that packets are always sent using their maximum capacity, which does not correspond to a realistic scenario. However, in our implementation, this aggregation is developed and controlled for our primitives. This allows us to carry a more realistic study, with greater efficiency while maintaining an ENO network.

6) Neighbors List - Contrary to what happened in [3], where the list of neighbors was static and fully defined at the beginning of each simulation, the heuristic created in this work allows this list to be dynamic. Such fact brings more realism to the work, since the structure of a static list of neighbors could be very unrealist. Therefore, with data processing, each node has a dynamic neighbor list. It is capable of storing information about the neighbor, such as the battery level or hops sink distance. As explained in Section 4.3, this list allows the sensor to choose the best neighbor to continue the message transmission at each moment and according to the needs of the network. Consequently, no sensor runs out of the ability to transmit or receive data; all sensors must have at least one neighbor. This represents a significant change - In the previous model, the optimizer always knew the state of all nodes in the network, at each moment, which was very unrealistic and could lead to critical problems.

7) Data Buffer Model - The model considers that sensors have a temporary memory zone to store data before it is sent to another location. This saved data could be data payloads received from its neighbors $\mathcal{N}_i$ or its own data payloads. Therefore, each sensor node $i \in \mathcal{S}$ maintains a *data buffer* $\mathcal{Q}_i(t)$ to store all the necessary data. Consequently, the buffer $\mathcal{Q}_i(t)$ represents the set of data payloads in node i's *data buffer* at time slot t. Given that the headers are only attached and detached at the moment of sending or receiving it is considered that they are not part of the buffer. For this reason, they do not occupy the space allocated for the payloads.

Let $Q_i(t) \geq 0$ be the size of buffer $\mathcal{Q}_i(t)$, i.e., $Q_i(t) = |\mathcal{Q}_i(t)|$. Since sensor nodes normally have limited RAM resources, we consider a finite buffer size Q_{max} in our model, i.e., $\forall i \in \mathcal{S}, \forall t, Q_i(t) \leq Q_{max}$.

Considering the nodes are continually performing sense, sending and receiving operations, the buffer required to store all this data needs to be dynamic. So, the buffer of a node $i \in \mathcal{S}$ can be described as bounded buffer capacity, and we can describe as follows.

$$\begin{aligned} &0 \leq Q_i(t) \leq Q_{max}, \\ &\forall i \in \mathcal{S}, 1 \leq t \leq T \end{aligned} \tag{3.7}$$

$$\begin{aligned} Q_i(t) = Q_i(t-1) - p_i^{out}(t) + p_i^{in}(t) + S_i^{\mathbf{s}}(t), \\ \forall i \in \mathcal{S}, 1 \leq t \leq T \end{aligned} \tag{3.8}$$

Here, Equation 3.8 represents the filling and clearing process of the node buffer; and Equation 3.7 highlights the bounded buffer capacity.

Besides, the following equations

$$p_i^{out}(t) = \sum_{j:(i,j)\in\mathscr{L}} p_{i,j}(t) \quad \text{and} \quad p_i^{in}(t) = \sum_{j:(j,i)\in\mathscr{L}} p_{j,i}(t) \tag{3.9}$$

reflect the cumulative amount of transmitted and retrieved data payloads of the node i at the time slot t.

3.3 Energy Models

Regarding the energy model, each node has three main components in the embedded energy harvesting system: Energy Harvester, Energy Storage, and Energy Consumers.

For each node $i \in \mathscr{S}$, the amount of deterministic harvested energy from the environment in the time slot t is $H_i(t) \geq 0$. Considering a model with batteries on sensors, the residual battery level is $B_i(t) \geq 0$ and it has with finite-capacity, i.e., $\forall i,t,\ B_i(t) \leq B_{max}$.

The total energy consumption for each node $i \in \mathscr{S}$ in time slot t is $C_i(t) \geq 0$. Each operation state $\mathscr{M}$ have a different energy consumption, like as described follow:

- if the node is sleeping (`z`), energy consumption is very low and steady, i.e. $C_i(t) = S_i^{\mathtt{z}}(t) \cdot C^{\mathtt{z}}$;
- if the node is sensing (`s`), the energy consumption is also constant, i.e. $C_i(t) = S_i^{\mathtt{s}}(t) \cdot C^{\mathtt{s}}$ and normally $C^{\mathtt{s}} \gg C^{\mathtt{z}}$;
- Whether the node (`Tx`) or receives (`Rx`) data packets, it would have a component depending on how many headers and payloads it transmits / receives. The highest energy consumption in the reception/transmission is due to the fact that the sensor node needs to turn on its communication antenna to transmit. After the antenna is turned on, the consumption to transfer one or more packets is very small. Nevertheless, it is essential to control the number of payloads and headers transmitted.

The energy consumption when a node is in transmitting or receiving state, can be defined as follows.

$$C_i(t) = S_i^{\mathtt{Tx}}(t) \cdot C^{\mathtt{Tx}} + C_p^{\mathtt{Tx}} \cdot p_i^{out}(t) + C_h^{\mathtt{Tx}} \cdot h_i^{out}(t) \tag{3.10}$$

$$C_i(t) = S_i^{\mathtt{Rx}}(t) \cdot C^{\mathtt{Rx}} + C_p^{\mathtt{Rx}} \cdot p_i^{in}(t) + C_h^{\mathtt{Rx}} \cdot h_i^{in}(t) \tag{3.11}$$

Transmission and reception is considered in Equations 3.10 and 3.11, respectively, where

$$h_i^{out}(t) = \sum_{j:(i,j)\in\mathscr{L}} h_{i,j}(t) \text{ and } h_i^{in}(t) = \sum_{j:(j,i)\in\mathscr{L}} h_{j,i}(t) \tag{3.12}$$

represents the total number of transmitted and received data packets, respectively.

The Equation 3.13 leads to the charging and discharging process of the battery.

$$C_i(t) = C_p^{\mathtt{Tx}} \cdot p_i^{out}(t) + C_h^{\mathtt{Tx}} \cdot h_i^{out}(t) + C_p^{\mathtt{Rx}} \cdot p_i^{in}(t) + C_h^{\mathtt{Rx}} \cdot h_i^{in}(t) + C^{\mathtt{z}} \cdot S_i^{\mathtt{z}}(t) + C^{\mathtt{s}} \cdot S_i^{\mathtt{s}}(t) \quad (3.13)$$

In view of the hardware constraints of the sensor nodes, the total energy absorbed by the time slot should be reduced with a finite value C_{max} (i.e., $\forall i,t, C_i(t) \leq C_{max}$). This value depends on the maximum total power consumption of sensor nodes and the duration of a time slot. Practically, $C_{max} \ll B_{max}$, as a result of C_{max} (mJ) is considerably smaller than B_{max} (kJ).

The energy system of each sensor node can, therefore, be dynamically modeled using the following specifications:

$$B_{min} \leq B_i(t) \leq B_{max}, \quad \forall i \in \mathscr{S}, 1 \leq t \leq T \quad (3.14)$$

$$B_i(t) = B_i(t-1) - C_i(t) + H_i(t), \quad \forall i \in \mathscr{S}, 1 \leq t \leq T \quad (3.15)$$

$$B_i(T) \geq B_i(0), \quad \forall i \in \mathscr{S} \quad (3.16)$$

Here, Equation 3.14 represents the highlights of the limited battery power and Equation 3.15 represents the recharging and discharging of the device. More specifically, the Equation 3.16 ensures ENO, which is supposed to be done by any sensor node in the network, i.e., the node must consume only the energy it has collected; the residual battery level at the end of operation must still be greater or equal to that at the start of the operation.

3.4 Events

Considering that our model represents a network of rechargeable sensors, except for the sink with no limitation in terms of the battery, the events are assumed to be activities of interest that follow a general probability distribution of a stochastic process. Example of a probability distributions widely accepted in the scientific community that events can be modeled with stochastic processes are the Weibull distribution and the uniform distribution [44].

The model considers the spatial and time stochastic process for the events. Consequently events start as input variables, that is, events follow a predictable distribution right at the beginning of each simulation. In our model, an event is a notable occurrence at a given time. The definition of an event is not explored throughout this work as it depends on the general purpose

of the wireless network and the type of hardware that the sensors have access to. For example, if development sensors assist advances in automatic speech recognition systems, changes in the quality of recordings, whether due to background noise or distant speech (events), can make this development a challenging task [45]. Also, suppose the nodes installed fluid control pressure at particular points of the industrial piping structure. In that case, a sudden increase or the decrease in pressure (events) in the proximity of those points may be perceived as blocking or leaking pipes in those regions. Monitoring these occurrences should always be taken into account.

Regardless of the application, the general approach of the events is to ensure that all points in the development and analysis area are covered, at any time, by at least one active sensor. This improves the security and reliability of the network, which can be critical in systems such as, measuring the stresses of a railway bridge where to perform accurate assessments, all vital components responsible for supporting the weight during the passage of a train, must be continuously monitored [46].

The model proposed in [44], consider that the starting time slots and locations of the occurring events are known beforehand, as they follow a predictable distribution. A sensor node only knows the occurrence of an event when it is active. All events have a fixed duration, δ, and location to ensure that can be captured at least one sensor. An event can be captured by more than one node simultaneously and in more than one consecutive time interval for the same subset of sensor nodes. Let $E_i(t)$ be the binary variable that indicates if the sensor node $i \in \mathscr{S}$ is close enough to capture an event $(E_i(t) = 1)$ or not $(E_i(t) = 0)$ at time slot t.

Some algorithms, e.g., [47], are programmed to minimize the possibility that an active sensor would not reach any given point. Therefore, this would increase the detection frequency of any incident and guarantee automatic detection if at least one sensor is on the event range. In certain implementations for intermittent events (e.g., an entity that appears and leaves), the aim might be to increase the detection frequency of events before they vanish. A delay in identification is appropriate.

For an event be captured by at least one node $i \in \mathscr{S}$ in neighborhood, it means that exists at least one node in sensing state while the event is running, i.e.,

$$\exists i \in \mathscr{S}, \exists t \in [\delta_{start}, \delta_{end}] : (S_i^{\mathsf{s}}(t) = 1 \wedge E_i(t) = 1) \tag{3.17}$$

where $\delta_{start} \in \{1, ..., T - \delta\}$ and $\delta_{end} = \delta_{start} + \delta$ correspond to the start and end time slots of the event, respectively.

According to [3] events are generated as inputs to the program. Also, only the monitoring of the detected events is carried out. In our proposed heuristic, events are much more complex. During the network's execution, whenever an event occurs, a flag is added to its payload, symbolizing that it corresponds to an event payload. This way, we were able to establish a series

of priorities for events, such as, sent always a packet associated to an event before a packet that resulted from periodic sense, which allows the latency to be considerable reduced. Also, if the sensor's queue is full, the periodic packets are the first ones to be dismissed. Events should always be on the queue, as they carry very important information, which can only be discarded lastly. More information on this point is demonstrated at Section 6.1.6.

3.5 Initial Theoretical Model

In this work, it is not intended to demonstrate or make any probabilistic and stochastic assumptions of the dynamic states of the network, including energy collection, the costs of each operation state, the ability to transmit and receive packets, the optimization of the input/output buffer or the space-time location of events.

The key purpose of this work is to develop heuristics that enable validation of a theoretical model that solves the problem of energy optimization in a finite horizon, choosing the most fitting state $S_i^{\mathbf{m}}(t) \in \mathscr{M}$ and wireless payload transmission $p_{i,j}(t)$ for-sensor node at each moment $1 \leq t \leq T$, maximizing the data that arrives sink node $\odot$ at the end of the horizon $t = T$. The initial theoretical model developed is described here:

maximize

$$Q_{\odot}(T) \tag{3.18}$$

subject to

Constraints (3.1) - (3.17)

However, during the course of this thesis, some issues were found during this initial theoretical model simulations, which led to the introduction of some constraints that were not originally predicted. One of these restrictions was the obligation to keep the sink node always in the receiving state, ensuring that it does not fluctuate, in order to prevent data loss. Thus, when in a certain time slot the reception of payloads was zero, we defined the energy consumption was non-existent, even though the sensors are operating in the reception state. This was also one of the extra limitations used to conserve the battery.

In the solution obtained with the initial model, most of the generated information was only distributed to the sink node near the end of the considered finite time. This means that the data created on the network was substantially delayed. Consequently, some variables have been added to periodically execute some of the required actions, such as sending data to the sink node.

The initial model assumes that nodes can only operate in one state per time slot. Such fact causes an overload of operations at the closest node to the sink. This means that the nodes closest to the sink are continuously active, in order to make the sense operation and transmissions,

while the peripheral nodes are regularly asleep since the packets generated in these nodes would take a long time to reach the sink. Thus, the information collected was not representative of the entire area where the network was implemented. It was also appropriate to impose constraints that compel all nodes to make sense of action and allow the resulting communication, regardless of distance. Therefore, a penalty was also introduced, which ensured that nodes with more payloads were more likely to make the transmission. This will not only minimize message loss, but also increase the usage of bandwidth in communications.

As a result, the original model was adjusted with the addition of new variables and new constraints, that were not originally predicted. It should be noted that, with the implementation of our heuristics, in a real context, where a sensory communication language is used, many of the limitations mentioned here were corrected by the rules that the network protocols provide, as we will explain in Chapter 6.1.

The next section presents the main variables and the main constraints added to the initial mathematical model, in order to correct the limitations mentioned here.

3.6 Final Theoretical Model

During this chapter, several variables have been introduced in order to control and analyze the entire model. The theoretical model has a few limitations that have been explained throughout this Chapter, which led to the addition of new variables and restrictions. Thus, at the beginning of the document, on the Section "Theoretical Model Notation", the presented tables summarize the variables used. This way, Table 1 represents the network control symbols, while Tables 2 and 3 represent the symbols of energy and events, respectively. The variables related to the operating nodes and data transmission are shown in Table 4. In order to control the entire network, it is necessary to have some control variables, that allow us to define the purpose of the network. We created Table 5 to summarize these decision variables.

Given the notation represented in the tables and taking into account the theoretical model's described, we represent some equations that have not been mentioned in the work until this point.

As explained in the previous section, a constraint has been introduced to ensure that the sink node is still available to accept packets. Such constraint removes the risk of a node trying to send the data and fails, since the sink node was not receiving.

$$S_{\odot}^{\mathtt{Rx}}(t) = 1, \quad 1 \leq t \leq T \tag{3.19}$$

The sink node $\odot$ can always receive packets from nodes.

In order to avoid a wireless link from being unnecessarily active, a constraint has been introduced to ensure that a link is active only when payloads are transmitted through it. Analysing the following equation, if $a_{i,j}(t) = 0$ then $p_{i,j}(t) = 0$, which means that no payloads are transmitted over that connection. If the link is active ($a_{i,j}(t) = 1$), at least one payload must be transmitted.

$$\begin{aligned} a_{i,j}(t) \leq p_{i,j}(t) \leq \infty \cdot a_{i,j}(t), \\ \forall (i,j) \in \mathscr{L}, \ 1 \leq t \leq T \end{aligned} \tag{3.20}$$

To control the payloads a constraint was created based on the following premise: the outgoing packets cannot be higher than the number of its incoming payloads before time slot $t-1$. Thus, the payloads received from other nodes, the payloads created when sensing and the initial payloads in the buffer, should always be lower than the outgoing payloads at a node on a time slot t. Additionally, the number of payloads could not be greater then its maximum buffer capacity. Equation 3.21 translate these requirements in the model.

$$\begin{aligned} p_i^{in}(t) + S_i^{\mathbf{s}}(t) \leq Q_i(t) \leq Q_{max}, \\ 1 \leq t \leq T \end{aligned} \tag{3.21}$$

Events are extremely important activities. It is also important to ensure that there is always at least one sensor in the area where the incident is taking place. Thus, it is guaranteed that events occur close to the sensors.

$$\begin{aligned} \sum_{i \in \mathscr{N} \setminus \odot} \left(E_i(t) \cdot \sum_{\tau=t}^{\min\left\{t+\frac{\delta}{2}, T-t\right\}} S_i^{\mathbf{s}}(\tau) \right) \geq v(t), \\ 1 \leq t \leq T \end{aligned} \tag{3.22}$$

Equation 3.22 ensures that at least one sensor is on the neighborhood to capture a nearby event.

3.7 Main Limitations of the Theoretical Model

The model developed in [3] had some important limitations that have been mentioned throughout this chapter. To overcome these limitations, some solutions were presented. The

model considers a simplification of a real scenario, both in terms of the structure of the network and with regard to communications. Additionally, only small instance are solved with this model, founding a solution that was close to the optimal one. The solution patterns were then analyzed, considering different simulation scenarios and studying all the patterns found, before the heuristics of this work were created. This was the motivation for heuristic construction: building an admissible solution, following the patterns found, that can be applied to larger problems.

The restriction that requires each sensor node to be only in a single operational state, per time slot, is relatively primitive. It forces the nodes to be compelled to add more congestion into the network because, for example, they can not conduct their environmental selection (sensing) and then carry out their transmission (Tx). Therefore, they must require two states that are not authorized to do so. This restriction led to introducing a variable that required that the sensors with more payloads on the queue delivered packets to the sink to prevent a higher delay. Regardless, this solution does not represent a real situation, which makes the theoretical model far from reality - as explained in Section 3.1.

By analysing the results from the theoretical model, it is possible to infer that a static list of neighbors of each node, brings a lot problems. This list was defined at the beginning of each simulation for each sensor and could not be changed at any time. The wireless coverage of each sensor, throughout the entire network operations, can be changed by several factors, such as obstructions in communications or changes in the network topology. Therefore, it is essential to guarantee the immediate update of each list of neighbors, whenever there are reasons for it, for example, avoiding transmission to non-existent neighbors or avoiding forgetting about recently introduced sensors on the network.

Sensing follows the period of time-sensor and it does not happen because of the occurrence of an event. Events are only detected if the time-sensor period coincides with the time slot in which such event occurred. Thus, it was not allowed to signal events differently, giving it higher importance. This assignment of priorities only happens due to the introduction of our heuristic of a protocol, developed from scratch, for data aggregation. The aggregation is introduced in Section 3.2 and explained in more detail in Section 6.1.7.F. Thus, it is possible to assign two different treatments for packets that correspond to periodic senses and those that correspond to senses that originate in events, as explained in Section 3.4.

The theoretical model establishes that it was possible to find some patterns of results analysis. For example, it is possible to infer that the nodes closest to the sink are the ones which have higher failure probability, provoking a cascade effect to nodes which are further away. So, this pattern concludes that using multi-hop networks is an advantage to fight this issue. Another pattern, showed that the aggregation rate increase with the proximity to the sink node. However, considering that this did not recognize the significance of signaling events and aggre-

gation, there are no assumptions on these two points that need to be replicated in our heuristics. In other words, in the development of our heuristic, presented in Chapter 4, we tried to replicate all the patterns that had already been found, in a more realistic way.

In short, the developed heuristics have the great advantage of being distributed. Thus, in each timeslot, the nodes can operate according to the energy harvesting they can collect until the end of the time frame. As this optimization happens in all timeslots, even if the nodes fail in the harvesting forecast, they can compensate it in the next time slot, when they do a new analysis of their status. In the previous model, the sensors did not have the ability to adapt to an error or change in harvesting. Therefore, the theoretical model developed was a good starting point to find patterns to be replicated. However, several weaknesses had to be corrected, as described throughout this thesis.

3.8 Final Notes

At this point, we present all the conditions of the mathematical model, from the communication constraints to the energy conditions. We also introduce all the concepts that will be important in the rest of the dissertation. The model represented here has some limitations that were overcome with the development of the heuristic, explained below.

In the next chapter, the analogy between the developed theoretical model will be made, explaining all the necessary changes to translate the mathematical language model into machine language to build the heuristic.

4

Distributed Heuristic for Neutral Operation

Contents

The development and implementation of the theoretical model directly on the sensor nodes require a readjustment of variables. The translation of the model into a machine with the processing capacity brings additional constraints not initially considered at the finite-horizon optimization problem in Sections 3.5 and 3.6.

Consequently, some additional auxiliary variables are necessary. For example, while the theoretical model considers payloads to be a countable element, in the implementation model, the processing machine measures the number of payloads, by the number of bytes they occupy and not by direct counting of the payloads number.

In this section, we represent new variables constraints and the differences between the theoretical and implementation models.

All the heuristics were replicated to obtain a solution that would mimic the patterns of the optimal solution, on the theoretical model. The work environment was designed to promote a straightforward implementation of this model, in a more natural way, given that the entire application replicates real scenarios. Some complementary flaws of the theoretical model are introduced in this chapter, which served as the basis for heuristic development.

4.1 Motivation

The limitations presented throughout Chapter 3 led to the construction of this distributed heuristic. In addition to the restrictions explained, the solution previously found was primitive, because it did not allow implementation in a real context. It would be very problematic as it did not consider the possibility of failures in the transmissions and the dynamism that a network needs in a real scenario, which can be changed. The heuristic created assumes that the sensors can be in more than one operational state, at the same time slot. Therefore, as already explained, this is significantly advantageous while managing the network's latency time. This, combined with the fact that we are using a distributed and non-centralized heuristic, in a real context, will allow a easy implementation of the system. In centralized heuristics, the optimizer knows, under any circumstances, the current state of all sensor nodes in the network, which is impossible to guarantee in a real context, where the conditions of the network are continually changing. Thus, a centralized heuristic requires that all sensors can communicate directly with the central node, which is very difficult to guarantee in a real scenario.

The previous model would have to overcome a series of constraints, especially the ones concerning communications between wireless sensors, as these require the implementation of network protocols, which was not studied in the previous model, as it is a simplistic version of a real scenario. Such fact led to the neighbor list's adoption, as we explained throughout this chapter and in Section 6.1.3.

The heuristic created aims to extend all sensor nodes' lifetime. It is always trying to capture all vital information from the network, with emphasis on events, while still maximizing the

number of times the sense operation is performed, increasing the number of packets transferred to the sink node. In addition to these measures, it is important to note that the constant operation of the sensor induces a higher loss of battery and overloads the information circulated in the sensor nodes, allowing them to wear out faster. This creates a cascading effect that affects the remaining nodes. From this model, the heuristic, while performing the energy forecast, that is always available until the end of the cycle, manages to optimize all operations to guarantee ENO.

Therefore, heuristics' development intends to bring the theoretical model closer to an implementation that corresponds to a real scenario, given that all the sensory communications part was not foreseen in the previous model. The heuristic aims to replicate the optimal patterns found, while using the new implementation environment, improving the final solution.

4.2 Main Heuristic Rules

The problem presented throughout this dissertation describes some variables as integers (e.g., the number of payloads at the nodes) while others can not be (e.g., battery consumption). In the optimization problem, the types of mathematical relationships between goals, constraints and decision variables determine how difficult it is to solve and determines the methods of solution or algorithms that can be used for optimization, and the confidence that the answer is genuinely optimal. The discrete part of the model means that the number of solutions can grow exponentially. Consequently, only small instances can be solved. Thus, this section will describe heuristic's main decision rules for building a solution that tries to replicate the useful characteristics of the solutions found by the theoretical model described in the previous section.

The aim of this study is not to make any probabilistic and stochastic assumptions regarding complex network states, including energy harvesting, energy costs (for sensing, sleeping, transmitting, and receiving), as well as transmission and data buffering power, or space-time event position. The purpose of this model is to look for an algorithm that can built a solution for the previous problem of finite-horizon optimization by finding the most acceptable state $S_i^{\mathbf{m}}(t) \in \mathscr{M}$ and wireless payload transfer $p_{i,j}(t)$ for each sensor node in $\mathscr{S}$ at each time slot $1 \leq t \leq T$, optimizing the number of data payloads at the end of the horizon $t = T$ at the sink node $\odot$. Accordingly, the resulting heuristic was created based on the theoretical model described in Chapter 3. To do this, we use all the notations already presented in Section 3.6. However, due to the new heuristic rules, it was necessary to develop some other notations. In Table 4.1 we can see the variables that are created exclusively to this heuristic.

Table 4.1 Heuristic Additional Symbols

Symbol	Description
D_i	Distance to the sink node measure by hops distance.
$HT_i(T)$	Amount of increased battery level of a sensor node i until the end the simulation time-frame
$E_i^{R_x}$	Binary variable that is true if the conditions to receive packets is satisfied.
$E_i^{T_x}$	Binary variable that is true if the conditions to transmit packets is satisfied.
$E_i^{S_x}$	Binary variable that is true if the conditions to sense is satisfied.
Tx_{Delay}	Maximum time that a node can retard the transmission.
Sx_{Delay}	Maximum time that a node can retard the sensing.
$N(i,j)$	Best neighbour of the list to transmission between i and j

As mentioned in Section 4.1, the heuristic aims ENO network. For this, at each operation, it is essential to guarantee energy optimization. By considering each operating state's energy consumption together with the available harvesting values, it is possible to achieve ENO. For this, heuristics evaluate the amount of harvesting in two different temporal moments, at the simulation init and at the end of the considered time frame.The first is the amount of harvesting that sensors have at the exact moment intended to carry out the operation. This value is responsible for responding to the immediate needs of the network. The second represents the amount of harvesting available until the end of the cycle, making the network adapt to its long-term needs.

Thus, taking into account the two harvesting values mentioned, whenever a sensor node decides to carry out a transmission, it is necessary to ensure that two constraints are not violated. The first one corresponds to the first condition in Equations 4.1 4.2 and 4.3. It guarantees that the sensor's battery is not less than 10% at the end of its operation. This limit was set to preserve the battery's physical condition and allow the sensors to have a longer lifetime. Otherwise, some performance limitations on the sensor could happen due to the sensor's battery, which means that the sensor can not operate normally. The second constraint is associated with the second condition of Equations 4.1 4.2 and 4.3. It ensures ENO in the long term, which means that when the node is operating, it calculates the entire harvesting that it will still receive until the end of a considered time frame. While making this calculation, the sensor node can check if it has enough energy to carry out the operation at that moment, knowing that, even with this energy waste, it will reach ENO. Therefore, considering these two different harvesting values, knowing the energy cost of the operations and the current value of the battery, we can guarantee that no operation causes the inability to achieve ENO.

Thus, the Equations 4.1, 4.2 and 4.3 are responsible for ensuring that the transmission, reception and sensing operations, respectively, always satisfying the rules imposed by heuristics for guarantee ENO.

$$(B_i(t) + HT_i(T) - C_i^{Tx}(t)) \geq 10\% * B_{max} \textbf{ and } (B_i(t) + HT_i(T) - C_i^{Tx}(t)) \geq B_i(0) \qquad (4.1)$$

$$(B_j(t) + H_j(t) - C_j^{Rx}(t)) \geq 10\% * B_{max} \textbf{ and } (B_j(t) + HT_j(T) - C_j^{Rx}(t)) \geq B_j(0) \quad (4.2)$$

$$(B_i(t) + H_i(t) - C_i^{Sx}(t)) \geq 10\% * B_{max} \textbf{ and } (B_i(t) + HT_i(T) - C_i^{Sx}(t)) \geq B_i(0) \quad (4.3)$$

We would like to emphasize that, whenever a sensor i decides to transfer to a sensor j, it needs to find out if it has enough energy to perform its operation - Equation 4.1 and if its neighbor j has energy conditions that allows it to successfully finish the operation - Equation 4.2. This is possible since the neighbors' battery level is one of the parameters that the sensors keep in their list. Thus, the chosen neighbor's identification to proceed with the transmission must guarantee the same two restrictions mentioned in the previous point, which is not only able to ensure that the neighbor will not keep its battery level below the critical zone (10%), but also to not lose the ability to secure ENO. If the chosen node j does not verify Equation 4.2, a new neighbor is searched.

4.3 Implementation

The proposed heuristic, named Heuristic for Energy Neutral Operation in Internet-of-Things (HENO-IoT), is designed as a time-slotted solution and is based on the standards of the solutions found in [3]. It is obtained without solving an optimization problem (computationally demanding) and introduces changes so that the solution is closer to the real world. Therefore, one of the main changes brings to our heuristic the habilitie to allow sensor nodes to do more than one operation in the same time slot, as we explained in the Section 4.1. For example, this is because they may need to capture important information from the network, send it right away, minimize latency, and the information reaches its destination quickly, making the system more reliable.

Algorithm 1 shows the main HENO-IoT pseudo-code, which runs distributively in each sensor node. This algorithm allows the heuristic to be a distributed one, which means that each sensor node is able to contribute to the network's optimization, in each operation.

Therefore, nodes are not obligated to connect with the central one, where all calculations related to network optimization are performed. This is extremely difficult to achieve in an entire network, so the great advantage of our heuristics is related to the fact that each sensor node has to make decisions independently. Those decisions have to contribute to the network's adequate global functioning.

It is possible to observe that for each slot of time this algorithm checks if the conditions for sensing, transmission and reception are satisfied, in order to guarantee the ENO. Based on Equations 4.1, 4.2 and 4.3, some procedures were created in order to implement the mathematical formulas. Thus, since we demand that the network must transmit data regularly, whenever the timers for the sensing and transmission processes exceed the limit allowed, certain procedures will be used to decide if the sensor node respects the transmission and sense conditions.

Otherwise, if these operations are on the vital battery zone or compromises ENO, the operation will not be carried out. These restrictions are made by each sensor individually, taking into account battery values, the consumption of energy operations, and the values of instant harvesting as well as the cumulative harvesting that happens until the end of the time frame. Each sensor knows its neighbors' battery value at every moment, thanks to the implementation of the list explained in Section 6.1.3. Such fact aims not only to guarantee the ENO service but also to ensure that all receipts from their shipments cooperate with it.

These conditions are checked in the Procedures *SxConditions()*, *TxConditions()*, and *RxConditions()*, presented in lines 8, 11, and 13 respectively, of the Algorithm 1. The binary variables E_i^{Sx}, E_i^{Tx}, and E_i^{Rx} store true to indicate the conditions are meet for these states for a node i. If the variable E_i^{Sx} is true, then the algorithm assigns a sensing operation for that timeslot (see line 9). If the variable E_i^{Tx} is true, it calls the Algorithm 2 to select a neighbour able to receive data from i.

The neighbour selection is a choice considering all neighbors. If the sink node is among the candidates, then it is selected. Otherwise, the algorithm selects the node with a shorter distance to the sink and with more energy in the battery. The list of neighbors is obtained through broadcast messages from the sink, as described in the Section 6.1.3.

Each node, before transmitting, performs aggregation operations with the data available in the queue. More information on this point can be found in Section 6.1.7.F.

Algorithm 1 State of Operation - Decision

Input: *time-slot* ;
Output: Operation state of node i;
/ Each node i runs distributively */*
/ j represents a neighbor of i */*
time-slot $= 0$;
while (*time-slot* $<$ *end*) **do**
    if $AtualTime - TimeLastSx \geq Sx_{Delay}$ **then**
        if SxConditions( ) **then**
            Sense;
    if $AtualTime - TimeLastTx \geq Tx_{Delay}$ **then**
        if TxConditions( ) **then**
            j $\leftarrow$ Select Neighbor(i); /* Call Algorithm 2 */
            if RxConditions( ) **then**
                Transmit(i, j);
        Sleep;
    else
        Sleep;
    time-slot ++;

---

procedure TXCONDITIONS( )
    TxBtt1 = $B_i(t) + H_i(t) - C_i^{Tx}(t)$;
    TxBtt2 = $B_i(t) + HT_i(T) - C_i^{Tx}(t)$;
    if (TxBtt1 $\geq 10\% * B_{max}$ **and** TxBtt2 $\geq B_i(0)$) **then**
        $E_i^{T_x} \leftarrow$ TRUE;
    Return $E_i^{T_x}$;

---

procedure RXCONDITIONS( )
    RxBtt1 = $B_j(t) + H_j(t) - C_j^{Rx}(t)$;
    RxBtt2 = $B_j(t) + HT_j(T) - C_j^{Rx}(t)$;
    if (RxBtt1 $\geq 10\% * B_{max}$ **and** RxBtt2$\geq B_j(0)$) **then**
        $E_j^{R_x} \leftarrow$ TRUE;
    Return $E_j^{R_x}$;

---

procedure SXCONDITIONS( )
    SxBtt1 = $B_i(t) + H_i(t) - C_i^{Sx}(t)$;
    SxBtt2 = $B_i(t) + HT_i(T) - C_i^{Sx}(t)$;
    if (SxBtt1 $\geq 10\% * B_{max}$ **and** SxBtt2$\geq B_i(0)$) **then**
        $E_i^{S_x} \leftarrow$ TRUE;
    Return $E_i^{S_x}$;

Algorithm 2 Select Neighbor

```
1: Input: Neighbors[ ] - Neighbors List of node i;
2: Output: bestN - Best Neighbor to send the information;
3: bestN ← Neighbors[0];
4: for N = 0; N < length(Neighbors[ ]);N++ do
5:     if (Neighbors[N] == SinkNode) then
6:         bestN ← Neighbors[N];
7:         Break;
8:     if (SinkDistance(Neighbors[N]) < SinkDistante(bestN)) then
9:         bestN ← Neighbors[N];
10:    if (SinkDistance(Neighbors[N]) == SinkDistante(bestN))
11:        and (BttLevel(Neighbors[N])>BttLevel(bestN)) then
12:        bestN ← Neighbors[N];
```

It is crucial to notice that the proposed heuristic does not have any specific procedure that guarantees event detection. However, the events are detected if the sense operation is executed during the event occurrence. Therefore, if the sense operation is performed in every time-slot, all events are identified. The rational side of performing the sense operation is that it maximizes the detection of events and it collects information from the environment. Despite having a very low energy consumption, this operation is only performed if the heuristic allows it always safeguarding ENO.

4.4 Final Notes

After presenting the entire model and all the necessary translations from the mathematical language to the machine language, we will choose the development environment and all the reasons that led to this choice, in the next chapter. In this chapter, the heuristic solution's differences and advantages over the simplified solution of the previous model were also highlighted.

Also, all of the software chosen for each stage of the work as well as its characteristics will be presented in the following chapters.

5

Simulation Platform

Contents

This Chapter is responsible for establishing all the system requirements necessary to implement the heuristic. Thus, all features and functionalities of the implemented system will be demonstrated and explained. The model, firstly implemented [3], used CPLEX Optimization Studio Engine to solve it. However, this represents a challenging computational task that takes a considerable time to obtain results. We need to efficiently find an approximation of the previous solution due to the issues already identified. It is important to ensure that the solution can be implemented in a real scenario with a suitable configuration.

To develop the heuristic solution that validate the optimal patterns, found in the solutions obtained using the previous model, we chose a network simulator, specially designed for wireless sensor networks - Cooja Simulator.

This chapter describes the Cooja Simulator specification and functionalities and the platform to scrutinize and confirm the heuristic results.

5.1 Requirements and Development

As mentioned before, the need for this development was evident in the early stages of this research. It was necessary to validate the model, obtaining a proper method to:

- Implement the model in a real-world context, improving the translation from virtual network simulation for real-world scenarios since the sensors already follow the physical rules of an IoT network;
- Interpret the behavior obtained by the model and study results;
- Perform several simulations varying parameters, like harvesting pattern, transmissions and sensing rate, events occurrence, simulations with and without payloads aggregation, and topology scheme;
- Opportunity to expand the model in the future, so the implementation could be as modular as possible.

An additional requirement was to obtain a means to translate the results to a real context efficiently. In other words, we need to get the results in a user-friendly way, according to the following steps:

- Translate the Cooja Simulation output to data-sets in which it is possible to manipulate data for further analysis;
- Provide a way to create a graphical representation of the given network and the data-sets;
- Perform a series of validation experiments.

The solutions were implemented in the Contiki operating system [48]. The Cooja framework [49] was used to embed the Contiki firmware in emulated IoT nodes and simulate the network. MATLAB [50] was used to analyze all the data results.

5.1.1 Contiki Operating System

WSN need to be extremely efficient in terms of energy consumption and memory footprint. Contiki operating system has a modular structure designed for WSN and networks with a large number of tiny devices. This operating system has several benefits, like portability, easy deployment, event-driven systems and power save support. Also, Contiki provides preemptive multithreading to apply in an individual process. It further allows finding mechanisms and abstractions that provide execution environments that meet the limitations of restricted devices and make this deployment as close as possible to reality.

The Contiki system is composed by the kernel, libraries, program loader, and a set of processes. To emulate the sensor behavior, Contiki supports system components, like a sensor, data handling, communication and device drivers. Every process leads to an event handler function and may need a poll handler function. To execute some processes, we need to run all these handlers. Each process should maintain his state between the calls of this handler functions in his private memory, due the stack's restoration after a handler function returns [48].

5.1.1.A Portability

Due to the implementation in C language, Contiki has been used in small micro controller architectures, given that it is a lightweight operating system. Moreover, Contiki offers support to download program code into the network dynamically. So, it can load and unload individual services or applications on the network at run time which is essential for large networks. Another advantage of this operating system is the opportunity to only transfer the application into the devices. Regarding that, a single application is much smaller than an entire operating system.

Furthermore, this run time compilation allowed us to correct some bugs in an operational network. Some related works presented methods for distribution code through wireless updates when each sensor network is running [51–53].

5.1.1.B Event-Driven Systems

Considering event-driven systems, processes are implemented like events handlers execution until they are completed. This type of system design is found to work for many kinds of sensors applications, although they still have some problems [54]. For example, in an event-driven system computation, cryptographic operations can hold all the CPU capacity, making the system unable to respond to external events. This is one disadvantage related to preemptive multi-threading system where this problem could be prevented.

Contiki uses a hybrid model to combine the best of these two worlds (event-driven systems and preemptive multi-threading). This model is based on an event-driven kernel that supports preemptive multi-threading implemented as an application library. Given this, the kernel is allowed to support two types of events - asynchronous and synchronous events. The difference is that synchronous events immediately cause the target process to be scheduled; on the other hand, asynchronous events are not in queue.

5.1.1.C Power save

Contiki offers the sensor networks the ability to power down the energy consumption of nodes when inactive. Contiki's Kernel does not have an abstraction layer that explicitly economizes energy, but instead, it authorizes an application or the system to implement that.

Besides, considering the event schedule queue, the operating system can power down the sensor whenever there are no scheduled events.This does not mean that the processor cannot handle an interrupt anymore once the poll handlers are run in case of an external event.

5.1.1.D Communication Stack

Contiki OS supports two different communications stacks: uIP and RIME. The uIP composes a lightweight TCP/IP stack and implements a minimum set of necessary resources for a full TCP/IP stack, providing TCP, UDP, ICMP, and IP protocols [55].

On the other hand, RIME is a lightweight networking stack optimized for low-power radio networking with explicit layers, that allows best effort and reliable transmission. The layers present very low overhead in the network when compared to uIP. In a multi-hop network, packets can be routed over neighbor nodes; RIME attributes the programmer the possibilities of defining their routing protocols. When compared to TCP/IP-based networks, one RIME improvement is the new layer called Radio Duty Cycle (RDC), that are created to save power consumption as much as possible, turn off the transceivers whenever they are not needed. ContikiMAC is the default mechanism for Contiki, adapted to the 802.15.4 radio and the CC2420 radio transceiver, which keeps the nodes sleeping 99% of the time while maintains all the communication on the network. This layer contributes to power efficiency. To ensure that the node wakes-up whenever needed, ContikiMAC uses real-time to set some call-back schedule functions that run as a protothread [56].

The communication protocol stack implemented in this work is shown in Figure 5.1 [55].

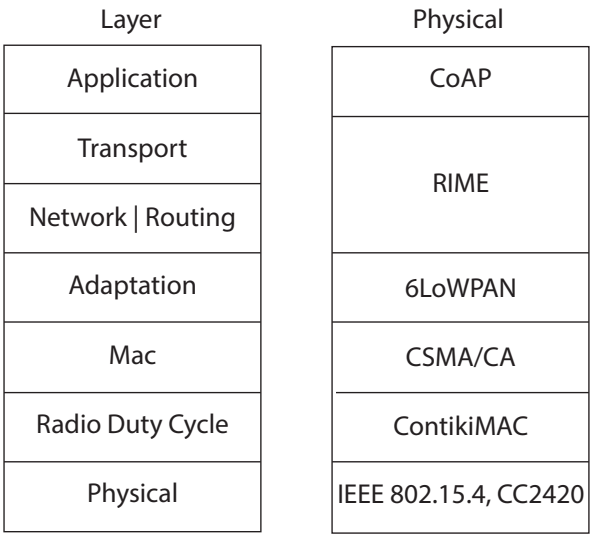


Figure 5.1: Contiki Communication Stack

As we can see, RIME combines the functions of the transport layer and network routing layer, and the RDC layer improves power efficiency without lowering too much the overall performance.

More details about these layers are explored in Chapter 6.1.

5.1.1.E Contiki OS vs TinyOS

Contiki OS and TinyOS are both specialized in WSN. Before we decided what was the best for our work, we compared the similarities and significant differences according to our requirements by analyzing the following points:

1. **Limited Resources:** Both operating systems can run on at hardware level, with restricted resources. However, Contiki's Kernel and scheduler have more complexity and higher resource requirements, allowing them to load and unload dynamically. Additionally, both offer libraries to achieve multi-threading.

2. **Flexibility:** This type of operating system is designed for many application types. The difference between them is the memory management. Contrary to Contiki OS, the memory is allocated statically on TinyOS, which means that it does not support dynamic memory management. The most significant difference in this point is for memory replacement on the programs of an application. Contiki OS supports a mechanism to replace only part of the programs of some application although, on TinyOS, all the applications should be updated, including operating systems.

3. **Low Power:** For our model, this would be the least important decision variable. All energy consumption is shaped by our algorithms to match the model developed. Therefore,

due to the ease of changing Contiki's applications, we thought it would continue to be a better option.

Some of the other advantages of Contiki OS are already mentioned, like software architecture, programming language, communication stack, and platform variety.

5.1.2 Cooja Framework

Based on Java, Cooja is a simulator specifically designed for sensor network simulations in the Contiki OS [48]. One of the main advantages is the possibility to create one simulation when each node can be of a different type. When in a real-world scenario, getting results of some experience may take more time - which turns this unpractical. So, it is essential to test and validate all the findings in a controlled environment before testing them in a real word.

Software development for sensors can be simplified using a system simulator that allows developing some algorithms to study the system and to observe its interaction with the environment in a controlled and faster way [57]. Therefore, one particular sensor platform at the hardware level is essential. It enables the development of low-level software, such as device drivers or communications protocol. This type of system may cause a longer simulation time and higher code complexity due to low-level programming language. In different circumstances, to get short simulation times, we can only use high-level algorithms that do not include the node hardware model. To overcome these two simulations levels, we took advantage of the Cooja's main contribution since it enables this type of cross-level simulation, allowing, in the same simulation, these levels of the system. It combines the low-level simulation of sensor hardware with high-level behavior.

All levels of the Cooja system can be changed and replaced. It is very flexible and extensible, i.e., we can model to our favor the sensor node operating system, radio transceivers, communications protocols, and radio transmissions.

The graphical user interface is user-friendly (see Figure 5.2), so the interaction with nodes and the simulations can be done using the mouse cursor. This makes the simulator easy to understand and to enable users to add some functionality faster.

Emulating sensor nodes gets important details for the fine-grained execution, making the code very deployable. Another vital factor of Cooja is its ability to concern the authorization to the medium access control, radio device drivers and the duty cycles of the sensors. This is very important for us as we need to manipulate all these characteristics to improve our results and make sure that there are no external interference's with our model.

In the end, due to its efficiency, scalability, flexibility, and extensibility, Cooja software is useful for networks where fine-grained execution details are necessary and to get our code deployable in a real world.

5.1.3 Matlab

To analyze all the results with a graphical interface, we use Matlab. This process is wholly intertwined with the process of building-up the simulation. During the code implementation, some particularities of MATLAB results lead to the need to rewrite some constraints to focus on new features.

Developing the Matlab platform was not linear until we got some symbiosis between the optimization model and analysis platform. This symbiosis can be observed in Section 5.3. Matlab's advantages include the ease in creating some data-based algorithms and matrix manipulation, which provides a simple interface that allows interconnecting with programs written in another languages or with externally created data.

5.2 Features and Functionalities

To create a correct study about HENO-IoT we analyzed all the features and the functionalities to fulfill the requirements. For this, we started to think about a system architecture that allows the analysis of network behaviors. Therefore, it was intended to interconnect optimized software to execute sensory networks with software that would analyze all results in user-friendly form.

Our baseline is structured as follows:

1. **Set simulation parameters and storage:** As we presented in Sections 3.6 and 4.2, there is a large set of data used in our implemented model. It is important to create a way to set or modify some simulation parameters quickly. Making the simulation easier to configure allows an external user to understand how the network works and configure it to his network propose. So, whenever we need to do some different tests, we do not need to change the code that runs on the sensors node, we just promptly adapt the parameters and all the simulation runs according to new definitions. Besides, Cooja allows that all the generated data can be stored in a file to later analyze with Matlab;

2. **Graph Representation:** Cooja provides the grid topology in a graph representing the modeled network with all the nodes $\mathscr{N}$. The application can also represent and differentiate the sink node $\odot$. Therefore, we have some available plugins that are used to interact with the simulation. For example, a plugin can manage some counter variables of a node and pause the simulation if the variable reaches some defined value. Other available plugin can pause or resume the simulation and select his run speed. After this, we get a 2D representation of the network. This way, we can visualize the network behavior and do some basic instructions with a comfortable visual environment. This makes the simulation easier to understand. Cooja visual can be observed at the following Figure:

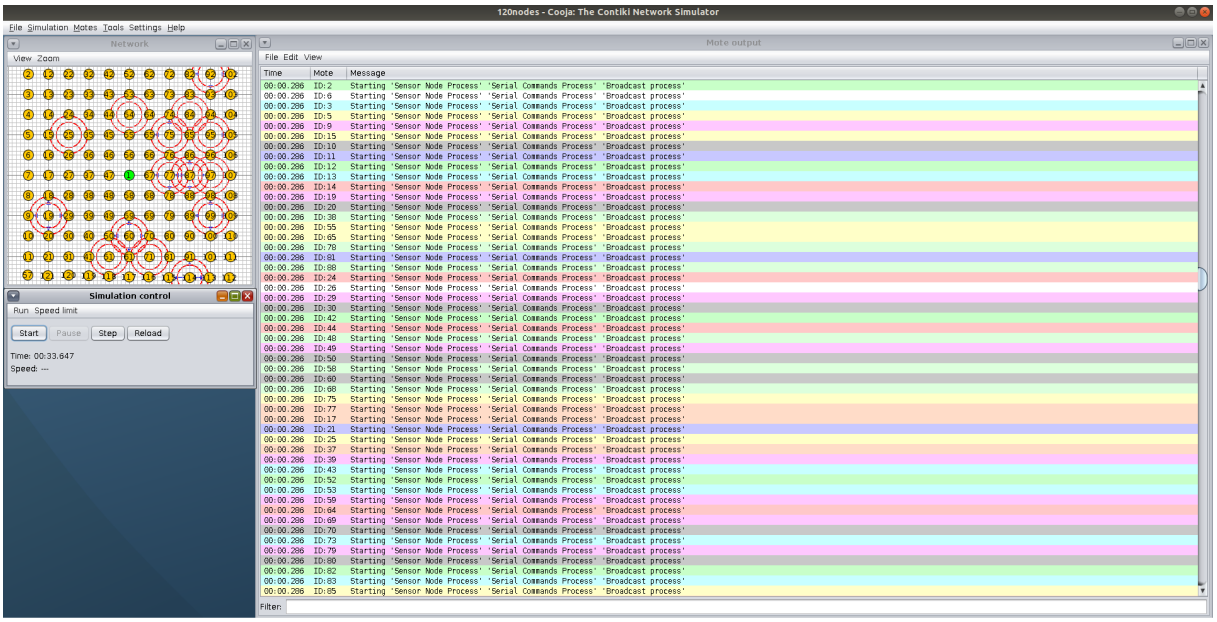


Figure 5.2: Frame capture using Cooja

In Figure 5.2 we can observe three essentials features. Some other features are available, but we decided to turn them off to keep the representation simple and with the essential ones;

(a) **Network:** On the left top corner, we can observe a graphical topology representation. The sensor nodes are distinguished from the sink node, with a different color to quickly follow. Furthermore, the red circles presented on the square illustrate each sensor's actuation range and the blue arrows the communication path. These circles appear whenever a node realizes some operation. Besides, we also can see the node ID of each node;

(b) **Mote output:** This is the most significant square present. Here we can observe all the simulation details related to all nodes - separated in three columns (time, mote ID, and message details), find some variable counters and see at the run time everything that is happening on the network. So, the mote output allows saving all the information to a log file to analyze later. This is the way we can manage the data with the Matlab. We choose what we need to export in mote output to save to one file that serves as input to Matlab;

(c) **Simulation control:** To pause, reload, resume or run the simulation step-by-step, we use the simulation control that enables using a mouse cursor. Moreover, this plugin allows us to choose the simulation speed limit. We can accept simulation without speed limit, taking advantage of the maximum processor capacity.

3. **Results Visualization and Validation:** It is our intention to validate server topics related to the network behavior like the operational state of the sensor, aggregation ratio, amount of data payloads on the node's buffer, the battery level of each sensor, use of each wireless links, data packets received through sink node, events capture and events delivery on the sink. When Matlab receives the solution exported by Cooja it can display the analysis and validate the results. The validation constraints ensure that those results are consistent and the model constraints are respected. For example, the battery level should always be between two defined value of battery percentage. These functions present several different graphical data of the metrics intended;

5.3 System Overview

As we already highlighted, all the evaluations of this work was obtained through Cooja Framework and Matlab. To successfully implement the model presented in Section 4, we need to interconnect them.

Figure 5.3 presents a simple high-level functional flowchart representing the steps to obtain results. Besides, this figure also represents the main modules used.

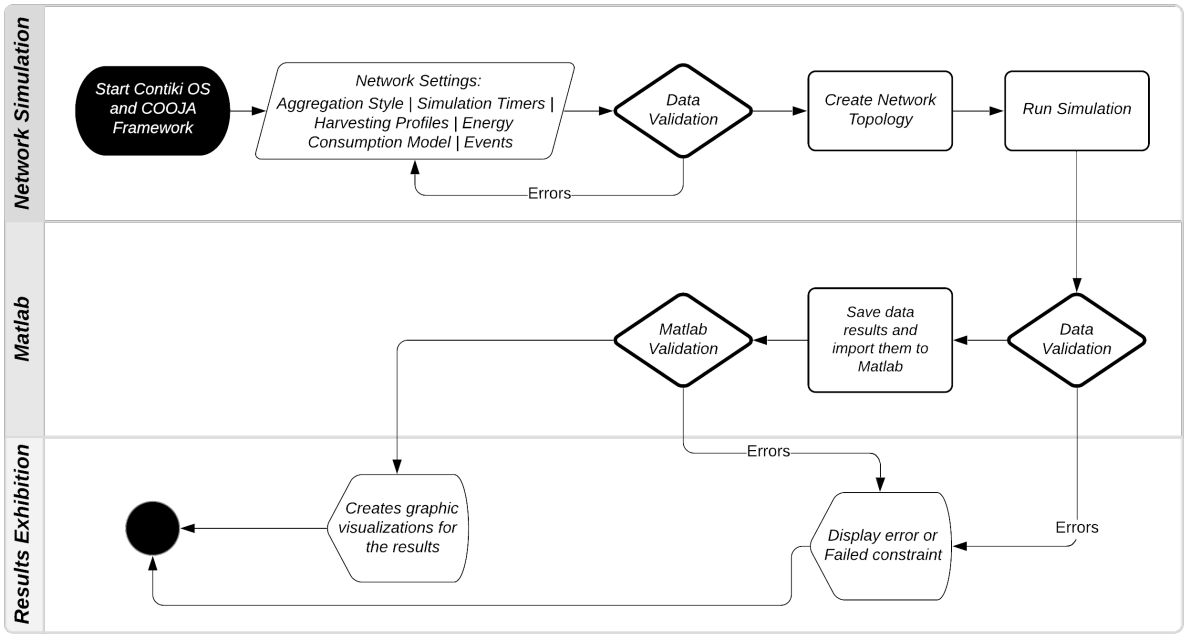


Figure 5.3: Functional flowchart of a model

As we can see, all the settings are firstly defined in Cooja that should run the simulation with the network settings defined and save the results, in a log file that serves as Matlab input. Afterwards, Matlab analyses all the data obtained and, if all the verifications are correct, it displays the graphic visualization about all the metrics described in Section 6.2. Depending on network topology and the simulation settings, results can take a longer time to be obtained.

5.4 Final Notes

Given that we have already presented all the necessary procedures for the execution of the work, we will explain each one with more detail. It is essential to refer all the technical aspects and show all the necessary changes regarding network protocols and data communication wireless. In the next chapter, we report and discuss the results obtained.

6

Evaluation and Results

Contents

This section introduces the environment configuration that we used to test the distributed heuristic and the discussion about the obtained results. These results can find the best proposal for the model presented in Chapter 4 and mirror a realistic IoT network that is ready to be deployable in a real world scenario.

The evaluation was carried out by simulations to demonstrate the solution's performance in terms of ENO. The reasoning behind all choices is explained, as well as the metrics evaluated. The environment setup, the performance metrics, and the results are also presented.

The model must keep the consistence in all the moments. It has to reproduce the real behavior expected from an EH-IoTN, and it should ignore logical flaws like the battery physical bounds, even if this improves the optimization objective.

One of the main objectives of this work is to discover the real behaviour of this type of network in order to improve the optimization objective. Furthermore, we studied the dynamics that this type of system adopts, so that we reach an efficient resource utilization. However, these resources should keep in mind all the premises established before.

Therefore, the network that we created should prove the advantages of concepts like multi-hop design, data aggregation, events capture and needs to determine whether it can operate without battery outage or not, as well as maintaining neutral operation using harvesting energy.

6.1 Environment Setup

The solutions were implemented in the Contiki operating system [48]. The Cooja framework [49] was used to embed the Contiki firmware in emulated IoT nodes and simulate the network. Besides, this model uses some temporal-related data to be deterministic, like energy harvesting profiles or the event's occurrence. Most of the parameters analyzed are close to a real environment.

6.1.1 Simulation Settings

Due to the model complexity, which turns possible many input combinations, we decided to vary only the most important. Hence, we can analyze some parameters' influence with more detail if we compare simulations with each others, where only each parameter is changed at a time. For example, we can compare simulations with and without aggregation modifying payloads data aggregation style, with constant and dynamic harvesting or even study the dynamics of network change some timer like transmission - what varies the transmission ratio. Defining all the simulation parameters can be problematic because some of them have an extensive range of possible values in especially those that correspond to data communications. Regardless, this dissertation's primary goal is not achieving the optimal values for all the parameters. The used values were enough to validate the model and consolidate several conditions for some settings, as we described in this chapter. The introduction of the optimal configuration of the network

parameters is suggested as future work, as a continuation of this dissertation.

The settings in Table 6.1 were used for Contiki and Cooja simulations. The results were obtained from an average of 5 simulation rounds. All parameters of the table are explained in the course of this chapter.

Table 6.1 Fixed simulation parameters

Parameter	Value
Network Nodes	121
Network Topology	Square Grid
Simulation Area	500x500m (meters)
Wireless Range	50m
Sensor Mote	Sky Mote
Wireless Network Protocol	Rime Stack
Radio Duty Cycling	ContikiMAC
MAC and Physical	IEEE 802.15.4 CSMA/CA
Time Slot Duration	$\Delta = 5$ min
Time Frame	$T = 60$
Maximum sensing delay	$Sx_{Delay} = 5$ min
Maximum transmission delay	$Tx_{Delay} = 5$ min $\vee$ 10 min
Maximum Battery	$B_{max} = 648$ J
Minimum Battery	$B_{min} = 0.1 \cdot B_{max} = 64.8$ J
Initial Battery	$B_{\mathcal{N}}(0) = 0.5 \cdot B_{max} = 324$ J
Consumption when in sleeping state	$C^{\mathtt{z}} = 0.63$ mJ
Consumption when in sensing state	$C^{\mathtt{s}} = 7.67$ mJ
Consumption when in transmitting state	$C^{\mathtt{Tx}} = 7.63$ J
Consumption when in receiving state	$C^{\mathtt{Rx}} = 7.63$ J
Consumption when transmitting a data payload	$C_p^{\mathtt{Tx}} = 9.46$ mJ
Consumption when transmitting a packet header	$C_h^{\mathtt{Tx}} = 217.96$ mJ
Consumption when receiving a data payload	$C_p^{\mathtt{Rx}} = 6.28$ mJ
Consumption when receiving a packet header	$C_h^{\mathtt{Rx}} = 375.85$ mJ
Data buffer size	$Q_{max} = 10$ Packets
Maximum data packet size	$P_{max} = 102$ bytes
Initial data payloads inside nodes	$Q_i(0) = 0$
Event Duration	$\delta = 4$ time slots
Events Proximity	Always nearby to at least one sensor node
Events probability	$P_E = 36\%$

6.1.2 Network Topology

Cooja framework allows us to define some performed evaluations, like recreate some topology with some algorithms explicitly created for this effect. So, we can determine the sparsity of the distributed sensor nodes and define the radio transceiver range. With more sparsity, we can create a network with lower node density, i.e., creating a looser network. Therefore, increasing the transceivers range means a more extensive neighbor list for each node, improving the average amount of links between them.

For our work, we needed to study the network behavior in a controlled environment since the evaluation metrics are dependent on the network structure. This means that we needed to know, in every moment, all the $\mathscr{L}$ wireless links possibles between all the nodes. In order to do that, we cannot consider a random network topology. We understand that this type of topology mirrors a network that could easily exist in a practical IoT setting, although this does not serve our model's purpose. This random distribution leads to a constant changing connectivity

between simulations, i.e., varying wireless links between nodes and between sink nodes. So, it makes difficult to immediate evaluation of network dynamics. This type of judgment is recommended as future work given that, in this dissertation, we already validated the results in a controlled topology.

We considered 121 nodes in a fixed squared grid topology, i.e., 120 sensor nodes with 1 sink node placed at the center of the network - see Figure 6.1 where arrows show the possible network path flow until reaching the sink node. These topology satisfies the heuristics and metrics for this model. We can study the sensors performance in the network with many devices and analyze the performance along all the network to evaluate nodes' performance across all the sink hops distance. Due to the grid topology, we can evaluate the nodes of the network up to 10 hops distance. Briefly, we found a network with 220 bidirectional non-overlapping links and with a central sink node. This topology is one of the differences presented, when compared with the one from the work presented in [3] where only 4 hops sink distance were considered and wireless links are significantly lower since they only consider 25 nodes in the topology used.

In this heuristic development, we first use the same topology of the theoretical model to validate all the algorithms and results. However, once the results were validated, we decided to study the most significant grid topology, validating the heuristic in a more realistic environment and taking advantage of the heuristic computational capabilities. This way, we decided to not include the graphical results about the small topology on this dissertation, since they have the same pattern results and because this represents a very preliminary study of the heuristic. So, we represent only the most important ones to make this dissertation's message as straightforward as possible. In the previous version, since the model was simplistic and far from a real scenario, the simulation time was too high, concerning the time frame and the number of devices on the network, which limited the analysis performed.

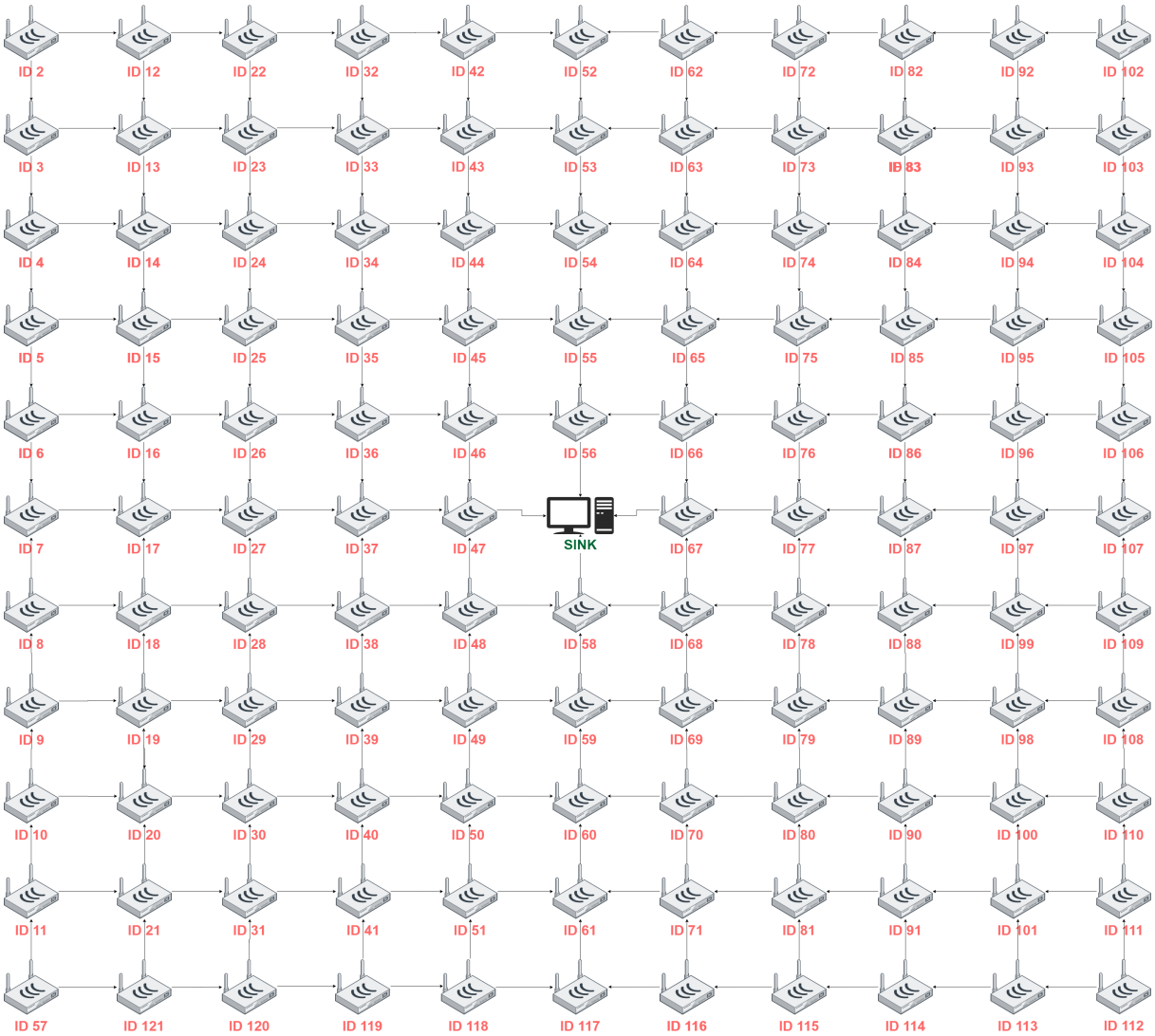


Figure 6.1: Network Topology

6.1.3 Neighbors List

The neighbors list was implemented in each node to get, at each moment, the energy level of all the nodes that it can reach. So, sensor nodes can decide what is the best path flow for the network at each moment to get the ENO, as already shown in Algorithm 2. Secondly, this neighbor list allows us to create a dynamic structure that provides the addition and removal of nodes at each moment, in run time. The system is still operating and can adjust dynamically - adding or removing the nodes for the corresponding neighbor list.

We use the neighbour discovery protocol to identify the neighbors and keep the range information, sink distance, battery level, neighbor address and save the wireless link indicators like Received Signal Strength Indicator (RSSI) and Link Quality Indicator (LQI).

This part of the implemented algorithm performs broadcast transmission in a periodic router advertisement multicast address. Sending this type of announcement is only performed periodi-

cally in order to conserve energy. This also contributes to the elimination of redundancy router-advertisement messages, due to the fact that we are working with battery-operated devices. The disadvantage is that if we need to add, remove or change some device, the network may take some time to calibrate all neighbor's list.

Each node can establish its distance from the sink through a hierarchical system. First, the sink establishes the distance to itself as 0 and starts sending its control messages to all nodes within its reach. In turn, these nodes that receive the message configure their range with +1 hop count. The receptors of this messages compare its distance (initially defined as infinite) with that of the message. If the hop message distance +1 is less than its current distance, then the current hop is set as hop message +1. The process continues between all the nodes until everyone knows their hop distance.

Thus, we were able to create a list of neighbors that provided important information, as already mentioned, and allow the network to be prepared for topology changes at any time, while its execution continues.

6.1.4 Time

Our modeled system corresponds to a time-slotted one with the finite-time operation. So, correctly defining the time for each time slot has great importance, since some other variables and constraints depend on this value. To do that, we first analyze some related works in this area to study the similarities and values used. Besides, choosing the time slot value depends on our finite horizon period time T. This time analysis was already performed in [3], and it serves as base for this dissertation. To meet what was already presented we used the same time slot duration - 5 minutes. Hence, as our goal was to achieve a degree of realism in this network's potential uses, 5 minutes per time slot is a suitable value. This value is consistent with the theoretical model's assumptions, where it is claimed that the operating states of the nodes need ample time to be fulfilled in their entirety before a new time slot is introduced. Because the active and most time-consuming states (i.e., sensing, receiving, and transmitting) represent operations that the built-in microcontroller will usually conduct in less than 2 seconds, 5 minutes offers a considerable operating period and is ideal for most IoT network applications with environmental sensing capabilities [3].

The time frame used in this heuristic development represents a time window superior to the one used in [3]. The dynamics of the network vary according to the harvesting available. Thus, since the day/night cycle is continuously repeating, it was important to analyze the network's behavior under different initial conditions. For this purpose, the duration of two day/night cycles was considered as a time frame. This allows assessing the network dynamics under different initial conditions and analyzing packets' management in the buffer and the communications between cycles, which helped us to conclude what were the effects of the first day/night cycle.

This study was also tested for simulations with five harvesting cycles, verifying that the network's behavior in the first cycle is different from all the others. Since this pattern was continuously repeated for cycles after the second one, we decided to represent only the first two cycles, in order to analyze the results.

6.1.5 Energy Profiles

All values, for energy consumption, used as the basis for this work, shown in Table 6.1, were calculated in the work presented in [3]. Therefore, these values were assumed to be constant in our model from the early beginning.

The solar-based harvesting values considered are taken from the 2016 Annual Average Value of Solar Radiation and its Variability in Portugal [58]. The measuring unit - kWh/m^2 - represents the amount of energy accumulated over one square meter for one year. Therefore, considering the areas where there is no sunlight each evening, we managed to draw a dynamic harvesting line to all the sensors.

6.1.6 Events

Events are considered a program input since our model is deterministic. They are distributed throughout time and follow a Weibull Distribution W(A; B). For the events distribution parameters inter-arrival times, we intend that the probability of event occurrences increases along the time, so B needs to be superior to 1. By fixing this value, we needed to find the value for A which depended on the finite-horizon T considered. In our model, we obtained Weibull-randomly distributed events in 36% of the time slots, using A = 3.11 and B = 3. These values are according to the previous theoretical model study presented in [3].

Events are guaranteed to be captured by at least one node and are detected if the sense operation is executed during the event occurrence. However, events are significant because they could contain critical information. Given this, whenever a packet is created due to an event's occurrence, a signal bit is placed to indicate that it should be given priority. Thus, whenever packet aggregation is performed, events are placed at the beginning of the buffer list to be sent first at the transmission time.

It is important to note that the model developed in [3] could not distinguish packets that corresponded to events with those that were created periodically, since the theoretical model corresponds to a real scenario simplification. Therefore, this is one of the significant advantages of the heuristic. We can ensure a lower delay while sending priority packets first and also providing a higher delivery rate.

Theoretical details about the events have been explained in Section 3.4.

6.1.7 Wireless Data Communication

In addition to the theoretical model validation, it is essential to create a network model that implements the heuristic and improves data circulation.

Data circulation on the network is made in the form of packets. These packet delivery between nodes involves network layers of the protocol stack responsible by packet routing, which decides the path of the data packet from a source to a destination. However, the default definition does not work for us, so we develop our routing protocol based on Algorithm 2 and in the neighbor list, as already explained.

Considering the Contiki communication stack, it is essential to emphasize that the object of study of this work was not optimizing the use of this stack. In this context, an analysis of all existing technologies was carried out and the one that best fits our work in each case was chosen.

Constrained Application Protocol (CoAP) is a protocol designed for internet devices with limited resources, offering support for multicast, very low overhead and simplicity, which is significant for EH-IoTN devices, given the poor memory and power supplies. 6LoWPAN makes it possible to use headers of the IPv6 protocol on IEEE 802.15.4 networks. It creates an adaptation layer between IEEE 802.15.4 and IPv6, with specific headers that can be added or removed as needed, allowing only what is useful to be sent.

One network layer, denominated as the MAC layer, is used to avoid packets collisions, i.e., whenever one or more sensor transmits data simultaneously for some network receptor, this layer avoids a crash. To ensure that nodes are ready to receive packets whenever they need, Radio Duty Cycle (RDC) is responsible for periodic waking-up for channel listen and to verify the existence of some packet transmission from their neighbors. If a node is transmitting a packet, it actuates as a source node that continuously sends the packet until it receives an acknowledgment from the receiver which means that the packet was successfully received. The physical layer is responsible for packet transmission and reception over the wireless medium.

6.1.7.A Rime Stack

Contrarily to the traditional layered communication architectures developed for a WSN, on Rime projects, each primitive-layer provides high-level abstractions, simplifying protocol implementations.

The Rime protocol stack offers a set of communication primitives like best-effort local neighbor broadcast or unicast, best-effort network flooding or hop-by-hop reliable multi-hop unicast. Due to such fact, this communication stack provides support for single and multi-hop communication primitives, which is very useful for our model development.

One main characteristic that we are looking for is implementing a routing protocol that specifies how packets are routed throughout the network. By doing this, we can control and

decide data flow according to node's energy level, maximizing the network utility, respecting our heuristics.

In Rime multi-hop primitive, the packet is sent across the network and the upper layer protocol decides at every node what is the next-hop neighbour. Given the ease of changing the protocols, provided by Contiki, we were able to change this layer according to Algorithm 2.

Therefore, Rime is designed for implementations in a WSN, providing primitive, as shown in Figure 6.2.

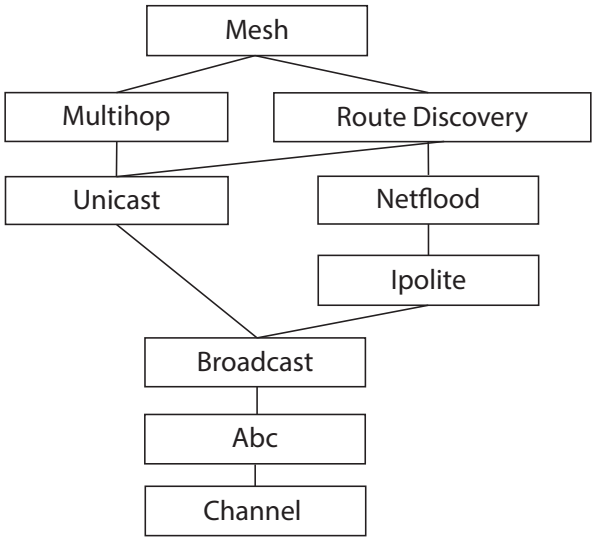


Figure 6.2: Rime Stack - Main communication modules

The presented primitives are projected as modules that can be reused and can easily be understood as follows:

- **Mesh -** Mesh routing protocol - Sends packets using multi-hop routing to a specified receiver somewhere in the network. Uses 3 channels: one for the multi-hop forwarding and two for route discovery;

- **Multi-hop -** Best-effort multi-hop forwarding - The multi-hop module implements a forwarding mechanism. Setting up routes is done with another Rime module, such as the route-discovery module;

- **Unicast -** Single-hop unicast - Sends a packet to an identified single-hop neighbor;

- **Route Discovery -** Route Discovery Protocol - Does route discovery for Rime through 2 channels: one for the flooded route request packets and one for the unicast route replies;

- **Netflood -** Best-effort network flooding - The netflood primitive sends a single packet to all nodes in the network. The basic netflood uses friendly broadcasts to minimize the number of redundant transmissions at any hop;
- **Ipolite -** Ipolite best effort local broadcast - Sends one local area broadcast packet within one-time interval. If a packet with the same header is received from a neighbor within the interval, it is not sent;
- **Broadcast -** Best-effort local area broadcast - The broadcast module sends packets to all local area neighbors with a header that identifies the sender;
- **Abc -** Anonymous best-effort local area broadcast - The Abc module sends packets to all local area neighbors;
- **Channel -** Rime's channel abstraction.

Therefore, Rime communication stack provides a set of lightweight communication primitives where the most complex protocols are created by joining other less complex ones, which fits what is intended for our model.

6.1.7.B MAC Layer - CSMA/CA

This layer is responsible for two main functions: Carrier data, i.e., verify if the channel is idle or not before transmitting data; Avoid collisions, which means that, before sending a packet, it confirms if some other node on the same transmission state is access the same channel transmission. If this happens, the node waits some time so that the other node can complete its transmission before the channel is available again.

This layer distinguishes broadcast and unicast packets, assigning different sequences to each of these types in the protocol layer. If the packet received in the MAC layer is of the broadcast type, it will be immediately transmitted without performing carrier sensing. However, suppose we receive a unicast packet. This packet is placed in a MAC layer buffer which decides what is the best time to continue transmission, depending on the medium's availability. This buffer has a limited capacity, so whenever a unicast packet has no space in the buffer, it is treated as a broadcast packet, so there is the risk that this packet not re-transmitted if there is any data corruption or collision.

In spite of this fact, our model needs to analyze all decision variables and buffers of the MAC layer. After that, it was essential to realize the limits of the data that we could send to avoid data collision, even in the extreme cases, where all the nodes wanted to send the maximum capacity of packets that they had hosted.

To improve the network utility and data transfer, it was necessary to change when a CSMA/CA MAC layer waits to perform the clear channel assessment. This layer uses Contiki's

callback timer (ctimer) mechanism to evoke the function responsible for carrying out the activities corresponding to packet transmission.

To make the CSMA/CA layer immediately perform clear channel assessment, we changed the referred callback timer's value to 0. Whenever there are packets in the MAC layer buffer waiting to be transmitted, Contiki immediately performs the channel evaluation, so that packet transmission occurs without any delay.

Moreover, as already mentioned, whenever the MAC layer buffer is completely occupied and a new unicast packet arrives, it is automatically re-transmitted as if it were a broadcast packet, with no availability check. This setting was causing a packet loss in the course of our work, so we decided to disable it and increase the buffer capacity.

As it is not the objective of this work to explore the MAC layer's best optimization for WSN communications, we only made the essential changes necessary to avoid packet loss. Therefore, our model does not evaluate metrics such as the average packet delay directly caused by the MAC layer.

6.1.7.C Radio Duty Cycle - ContikiMAC

RDC layer is designed for energy savings in communications, keeping sensors off whenever needed. ContikiMAC protocol is inspired by all the protocols that already exist for this network layer. This protocol is designed to be simple to implement, as it only uses asynchronous mechanisms and does not signal messages or add headers to the data transmitted. Thus, it uses radio service cycles to listen to transmissions made by neighbors periodically.

Clear Channel Assessment (CCA) time corresponds to the moment that a sensor checks if there is any packet to receive. Whenever a message is detected in this cycle's execution, the receiver is kept on to receive the data packet.

When the packet is completely received, it sends a confirmation signal called link-layer acknowledgment. Hence, the sender is sending the packet continuously until receiving the link-layer acknowledgment from the receiver.

That is, whenever a sender intends to send data packets, it does not know if the receiver is sleeping, awake or if it is busy receiving other data. Hence, transmission can only happen when the receiver is in the active state and realizes that there is a transmitter sending data, thus changing to the busy state to receive data from that transmitter. Finally, it sends a signal to the transmitter, informing that the reception was successful.

The protocol has a power-efficient mechanism for awakening the sensors. It uses fast sleep mechanisms, which allow data receivers to detect false-positive wake-ups and allows a transmission phase-lock optimization to allow better energy efficiency during communications. In conclusion, this protocol improves communications during the time they are not happening, effectively deactivating the sensors and improving how communication is carried out through

optimizations that operate in run-time during the transmission.

Concerning broadcast packets, they are not subject to the link-layer acknowledgment, as already mentioned. When a packet of this type is sent, sensors are sent throughout the wake-up interval to ensure that all neighbors can receive it. This principle of link-layer acknowledgment is shown in Figure 6.3.

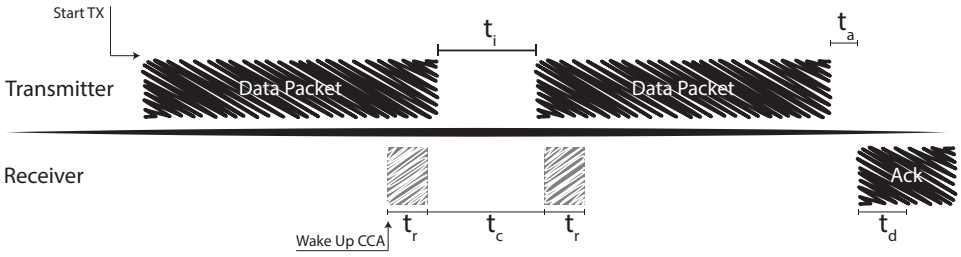


Figure 6.3: ContikiMAC transmission and link-layer acknowledgments

As we can see, the protocol works based on four different times:

- t_i**:** interval between transfer of each packet;
- t_r**:** safe link time needed for reliable CCA indication;
- t_c**:** intervals of CCA;
- t_a**:** period between packet reception and packet acknowledgment send;
- t_d**:** time needed to successfully detect an acknowledgement from the receiver.

These values must be defined according to the needs of the network. However, they must obey some restrictions. The interval between each packet transmission, t_i, must be less than t_c, which corresponds to each CCA interval. This happens so that the communication is guaranteed to be detected by one of the first two CCA signals. Also, t_i and t_c require the smallest packet that ContikiMAC can support. For only two CCAs to detect the transmission of a packet, it cannot be too short to happen between two different CCAs. This means the transmitting time of the shortest packet, t_s, must surpass $t_r + t_c + t_r$. Whenever a transmission happens, the receiver's radio is always on to receive the complete packet and, in the end, send the transmission recognition signal. The time until the acknowledgment packet reaches the original transmitter, t_d and the time it takes for this signal to be transmitted. It also establishes the lower limit of the verification interval [59].

In short, these times must obey the following restriction:

$$t_a + t_d < t_i < t_c < t_c + 2t_r < t_s. \tag{6.1}$$

6.1.7.D Physical - IEEE 802.15.4

WSN is used in a wide range of applications. The IEEE 802.15.4 standard is made for low data rate wireless network communications. When using this pattern, we have to consider the delay ratio of the packets, the rate of losses and the traffic capacity. If we use short IEEE 802.15.4 addressing mode, it is only possible to carry 102 bytes of data in a MAC layer frame using the Rime protocol stack of the Contiki operating system [60].

Therefore, in our work, we configured the maximum capacity of data aggregation per packet according to this value, respected the maximum transmit unit and avoided packets' fragmentation. The CSMA / CA protocol is directly linked to this standard, since the memo waits for a constant period after the transmission of a data frame to receive an ACK.

6.1.7.E Radio Driver - CC24240

While developing the entire project, the strategy was implemented in Contiki and compiled for the MSP430 processor microcontroller. Because of this, the hardware platform we have used is the sky mote devices with a CC2420 radio module for our simulations.

These wireless sensor nodes are entirely restricted in terms of memory capacity, computational capacity and energy. Since, in our model, the sensor nodes only need to collect environmental data and send it to the sink node to collect all the information from the network, the capacity of this type of sensors is more than sufficient. Also, these sensors have the advantage of having antennas for long-distance communications.

6.1.7.F Packet Buffer Management

One of the advantages of using Rime Stack is that it does not need to deal with packet header details, such as alignment or byte order. Instead, the header only has a list of attributes whose values are defined before the layer transitions.

To store a complete packet entering or leaving a node, a global structure, named packetbuff, is used.

Regard that each element of the packet queue has a maximum lifetime. In our model, we consider the lifetime infinite, so packets never time out from packet queue; they only can be discarded if the sensor node decides to do that for some reason, like packets overflow in the queue.

We express brief details about how packet queue works. This queue is the structure responsible for creating an outbound packet or store an inbound one. Only one packet at a time can be saved in a packet queue.

It is assumed that the packet headers are managed by a mechanism where all we need is to define the list of convenient attributes. However, in order to make the header not too long, we define the following attributes:

- The return address as an E-Sender. Since this address will be changed in MAC header while forwarding;
- The address of E-Receive, for the same reason as the previous one, we save the recipient and the sender's address as attributes to avoid changes to the MAC header.
- Timestamp, which corresponds to the moment when the packet was sent, to allow us to analyze the average delay of sending packets.
- Packet event ID. It is important to ensure that packets that match event detection are sent with priority and never dropped at the expense of packets called normal. To save space, we use only one byte to indicate and identify the packet as an event. An event packet is always sent before a packet that resulted from periodic sense, which allows the latency to be considerable reduced. Thus, events are aggregated in order to be the first in the sending queue.

Figure 6.4 represents the structure of each packet that could be saved in the queue.

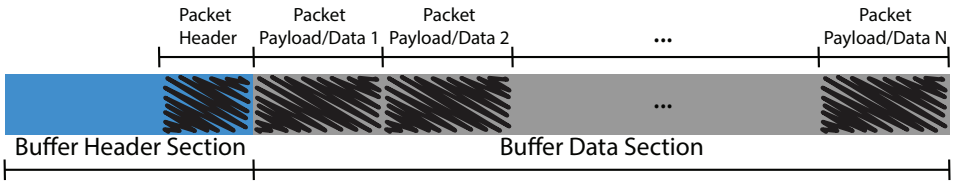


Figure 6.4: Packet Buffer structure

The represented structures are used for outbound packets and to store them in the queue. As we can see, we have two sections presented - the header section and payloads buffer data section.

For incoming packets, header and data payloads are temporarily saved in the packet buffer data section. Every time a node receives a new packet or sensing new data, he perform aggregation to complete the data section in one packet. When sensor nodes need to lead with a new data payload, it aggregates those with the previously added to the packets stored in the queue. This means that the packets saved in the queue should be completely filled until it creates new ones.

The header is only allocated on the outbound packets and it stores the communication protocol information and the packet attributes.

Given this, it is essential to note that all of these header attributes are defined at one time on the sender side so, in order to decode these parameters on the receiver side successfully, the entire header must remain intact until it is received correctly. This means that, the header

of the Rime stack must not contain any attributes that are changed from jump to jump, such as the number of hops the packet has already passed. In our model, this is not an issue, since the packets were always sent towards the sink. The decision to follow the packet along the network is made individually for each sensor on the path between the first packet emitter and the sink. Therefore, each intermediate node will decide to follow the packet according to its energy restrictions and Algorithms 1 and 2. Furthermore, when we send a packet over the network, it is always sent as a single packet transmitted like a unicast one.

Data aggregation should reduce the number of data transmissions. To maximize the aggregation level, when the payloads queue is full, nodes only send the packets that have a maximum length allowed - W_{max}, i.e., transmission node only sends multiples of that maximum. Therefore, a node will carry out one or more transmissions (each of a maximum length) and save the remaining data length, if it exists. In contrast, if the initial maximum length for transmission is lower than the maximum defined, all the data is sent, and the queue will stay empty. The transmissions with incomplete packets only happen if Tx has already been exceeded. Therefore, the sensors are obligated to transfer the information they have in order to minimize latency.

The difference between sensors using aggregation or not can be explained very succinctly. In simulations in which the sensors use aggregation, they can place more than one payload corresponding to the buffer data section, visible in Figure 6.4. When we do not use aggregation, only one payload can occupy this section, not taking advantage of all the space available for payloads. This causes the sensors' queue to fill more quickly. Each packet can only contain one payload in the buffer data section, leading to an increase in the number of transmissions carried out and a decrease in the battery level.

Whenever a sensor node exceeds its maximum packet capacity allowed in the buffer Q_{max}, it is obliged to discard the oldest payload to receive the new one. If some of the payloads needed to be discarded correspond to an event one, this can only be happen in the last case, which means that if all the remaining payloads present in the queue are also event ones. Otherwise, due to the importance of event information, they are always kept in the queue and they are the last ones to be discarded.

6.1.7.G Network Time Synchronization

A protocol responsible for synchronizing all nodes' clock in the network has been implemented to ensure that all nodes operate synchronously to avoid out of sync messages. Due we are using a multi-hop network, we implement a hierarchical tree synchronization.

This protocol works through implicit methods, which means that it is not necessary to exchange synchronization messages. Instead, it uses an underlying radio driver to timestamp all messages, incoming and outgoing. Each node has an authority level. The highest level nodes synchronize those of the lowest level. Our model defined the sink as unique, the highest author-

ity node, therefore the node with the reference clock. Based on this reference and according to the authority level, each sensor's time offset is calculated to establish synchronization. Therefore, the synchronization messages come from the sink and are transmitted hop by hop. In this way, the hierarchical tree can be implicitly constructed and dynamically adapted to changes in topology. Predictability can not be achieved without coordination, and time synchronization is necessary for the coordination of distributed entities and events. Also, time synchronization enables, among other things, ordered event logging. The engine used is named Timing-sync Protocol for Sensor Networks (TPSN). We chose to use this mechanism, as it is compatible with the same hardware that we are emulating in our network. It strikes a good compromise between overhead communication, precise synchronization and complexity of the computation.

6.1.8 System Architecture

As we already mentioned in the course of all the work, the idea was to make a dynamic architecture based on modules so that each of them could be used individually in different applications shortly. Figure 6.5 represents the main view of the architecture of the implemented system.

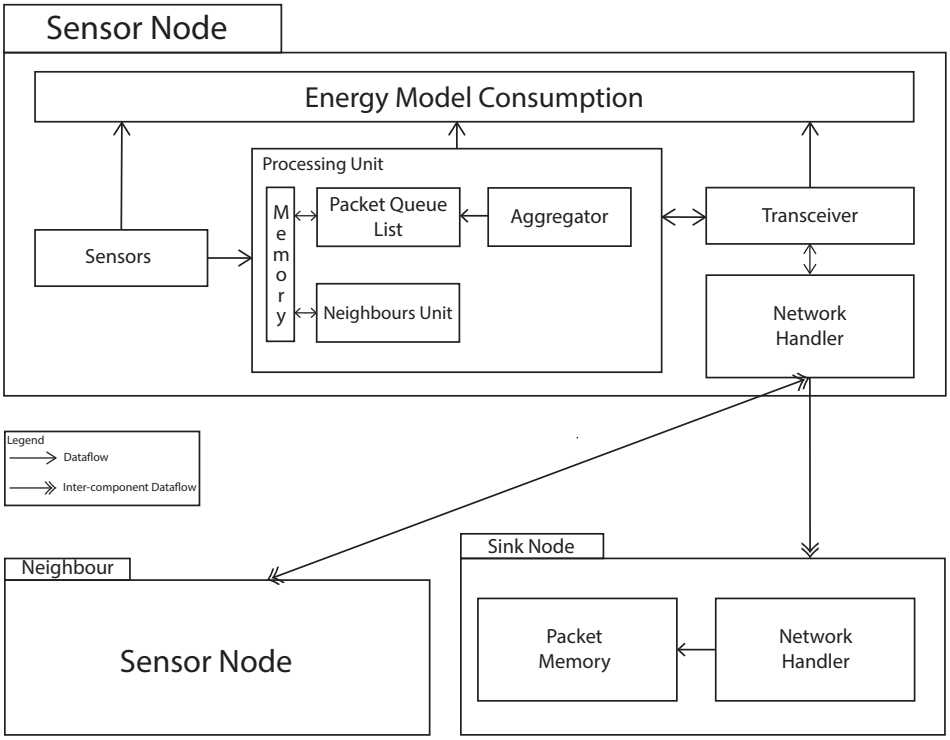


Figure 6.5: System nodes architecture

As we can see, concerning the node sensor, we have 5 main modules. The Energy Model Consumption module is responsible for calculating all energy costs related to communications

and data processing. It is also responsible for the Energy Harvesting module, which contains information on the available harvesting values. The sensor module represents the hardware that can be used according to the purpose of the network. If the objective is to measure the ambient temperature, then the sensors used will be the temperature sensors. The processing unit is responsible for all the memory management of the data from the aggregation of packets in sending/receiving to the management of the neighbors' list. The transceiver and the network handler are responsible for the communication between the nodes. They manage all the network protocols and decide the best moments so that the transactions are carried out successfully.

Each node sensor can communicate with a similar node, which is considered a neighbor or communicate directly with the sink, regardless of its range of action. In contrary, the sink has a much more simplistic architecture, since it only receives the data and saves it, doing any data processing whatsoever.

6.1.9 Memory Footprint

Given that the sensor nodes can have memory difficulties in the IoT content, since sometimes their physical size is relatively small, it is important to carry out the entire implementation of the heuristics, taking care of their size, so that the code can fit in the sensors. Thus, measuring the viability of the utility of the memory restrictions of the sensor is composed in two parts, the compiled code that will be stored in the on-chip ROM and the footprint memory that will be necessary to execute all the code. This size required for the code's execution varies according to the buffers' size, variables, and protocols used.

According to Adam Dunkels *et al.* [61], a simulation was performed with 25 nodes in the grid to determine a suitable configuration of the buffers. Each sensor node performs the sensing and transmitting operation every 10 seconds. Dunkels concludes that with this configuration, the maximum buffer capacity Q_{max} would be 10 packets. Our topology used 121 nodes, which only performed the sense operation every 5 minutes, so, taking these values into account and given the network's purpose, we concluded that 10 packets per queue buffer was more than enough to respect the memory rules and our heuristics.

6.2 Metrics

Some metrics have been defined to calculate the performance of the heuristic.

- **Battery (%):** as all the sensor nodes have harvesting devices and batteries with the same characteristics. Battery (%) is a performance metric that represents the remaining energy in the battery. This is calculated through Equation 3.15 and considers the energy consumed in any of the four states and the energy harvesting. Thus, it is possible to access the energy-neutral operation and examine the network's response to several harvesting

profiles. Due to this fact, it is conceivable to evaluate the energy-neutral operation and to monitor the network answer to harvesting profiles;

- **Percent of Unperformed Transmissions:** if there is a lack of energy, a node may not satisfy the conditions of Algorithm 1 concerning performing transmissions. When calculating this metric, the number of times the node does not help transmission conditions is divided by the sum of all the transmission attempts. It should be noted that a successful transmission is only performed if the receiver also meets the energy conditions imposed by the algorithm. This metric establishes the adopted data routes in variable energy conditions, so the network throughput is not affected;

- **Event-Traffic Delivered to the Sink:** according to Algorithm 1, a node must satisfy some conditions when performing sense. After carrying out this operation, the node must communicate the sensing readings. This metric is obtained by calculating the ratio between the number of times that at least one message, with information about a particular event, reaches the sink and the total number of events;

- **States Operation:** during the network execution, each node must know the best state operation at each moment. According to the battery level, packets queue state or event capture, nodes can operate in different states at each time slot. So, these metrics allow us to study the network state behavior across all the hops distance from the sink node;

- **Data aggregation:** this metric should vary according to sink distance. To study this, we analyze the aggregation as a function of sink distance to verify if it improves the energy waste.

Due to these metrics and due to Matlab's processing, we were able to extract all the necessary data and graphics to carry out the study of the network's behaviour.

6.3 Results

To validate all the proposed metrics, we evaluated the network's behavior following two harvesting profiles. This harvesting variation allows us to access the network's behavior in situations where there was a lack of energy collection and in cases where this energy was abundant. Thus, we used harvesting values that correspond to the day/night cycle that we could find in a real world situation, which brought these results closer to reality. However, as it was essential to create an analysis base where all variables were controllable, to facilitate the study, we carried out simulations with constant harvesting, making an analysis variable continuous to discern the results better. This continuous harvesting corresponds to the average of the dynamic harvesting used. All simulations were carried out over two harvesting cycles, corresponding to two

days/night cycles. The error associated with these simulations is shown in all bar graphs. In the images that correspond to the sensors' battery level, the error is not shown, as it would be visually unpleasant. Still, in all simulations, this was always less than 4%.

If we analyze Figures 6.6 and 6.7, we can observe the following data:

- On the left column, simulations that correspond to constant harvesting are shown, while on the right column, we observe those of dynamic harvesting, which corresponds to the solar pattern. It is possible to follow that initial and final harvesting values are very close to zero, with some harvesting peaks in the middle of the consideration cycle;

- In the first line of each figure, simulations are presented where the nodes took advantage of data aggregation, while in the second line, this resource was not used;

- The Figures 6.6 and 6.7, vary the parameter Tx and Sx is maintained in both simulations. That is, the timer that defines the maximum delay allowed between information collections is maintained; only the timer that corresponds to the maximum transmission delay is changed. This will allow us to evaluate if the network dynamics is different with different packet conditions present in each node's buffers.

These are the most critical graphics of the work, as they were the ones who brought the validation of the energy neutral operation network. The value for Sx and Tx are presented in minutes unit, throughout the entire chapter. Due to the combination of these two timers with the aggregation utilization method, together with the different harvesting profiles, we evaluated all the previous metrics according to 8 different standards.

Given this, Figures 6.6 and 6.7 show the energy neutral operation results, varying Tx, the harvesting and aggregation profiles. The left axis shows the battery level in all sensors, grouped by their distance to the sink. The individual battery levels for each sensor are not displayed, as the resulting figure would be confusing. Since the network is symmetrical, each line represents the nodes with the same characteristics. The battery average for the entire network also appears as a reference. The right axis is on a different scale, as the harvesting values are quite different from the battery values. This axis represents harvesting picked up by sensors over 60 time slots. Each time slot is equal to Sx.

Network aggregation is shown in Figure 6.8. The first Sub-figure, 6.8a, corresponds to the values with constant harvesting, while the Sub-figure 6.8b uses dynamic harvesting. In both graphics, the simulations for the two values of Tx used are presented. The aggregation value is shown on a scale from 0 to 1, where 1 means that the payload's data section's capacity was filled and 0 means that no data was transmitted. In the horizontal axis, nodes are grouped by their distance from the sink.

The percentage of average failed transmissions can be observed in Figure 6.9. Figure 6.10 represents the same data but is distributed throughout hops distance. These failed transmissions happen due to the lack of energy available in the sensors.

Figure 6.11 represents the percentage of events that were successfully detected during the simulations. For an event to be captured, at least one neighboring node must perform the sense operation when that event has been active. Furthermore, Figure 6.12 depicts the percentage of captured events that managed to be delivered to the sink. Keep in mind that the sending of these events has priority over the sending of periodic payloads.

The network utility is represented in the Figure 6.13. It is possible to observe the commutative packet reception by the sink over time. The graphics on the figure's first line, corresponding to the simulations where aggregations are used, represent a line for receiving payloads over time and another for receiving packets. As we are using aggregation, each packet can have one or more payloads. Naturally, these lines in the graphs that correspond to the simulations without aggregation are superimposed, since each packet only has one payload. The lines represented in each graph vary the value attributed to Tx.

Represented by the percentage of the time-slots, the distribution of the sensors' operational states is shown in Figures 6.14 and 6.15. Figure 6.14 is based on simulations in which the sense timer is the same as the transmission timer and Figure 6.15 corresponds to simulations where the transmission timer is double the sense. Thus, considering that in 120 time slots analyzed, a sensor carried out the transmission in 60 time slots, we can say that this sensor was in the transmission state in 50% of the analyzed time slots. The sensors have 4 operational states. However, the bar corresponding to the sleep state is not represented in the graphs because sensors are in this state at all time slots. This state has been removed from the chart for being constant and for simplification reasons. On the horizontal axis, the nodes are grouped by their distance from the sink.

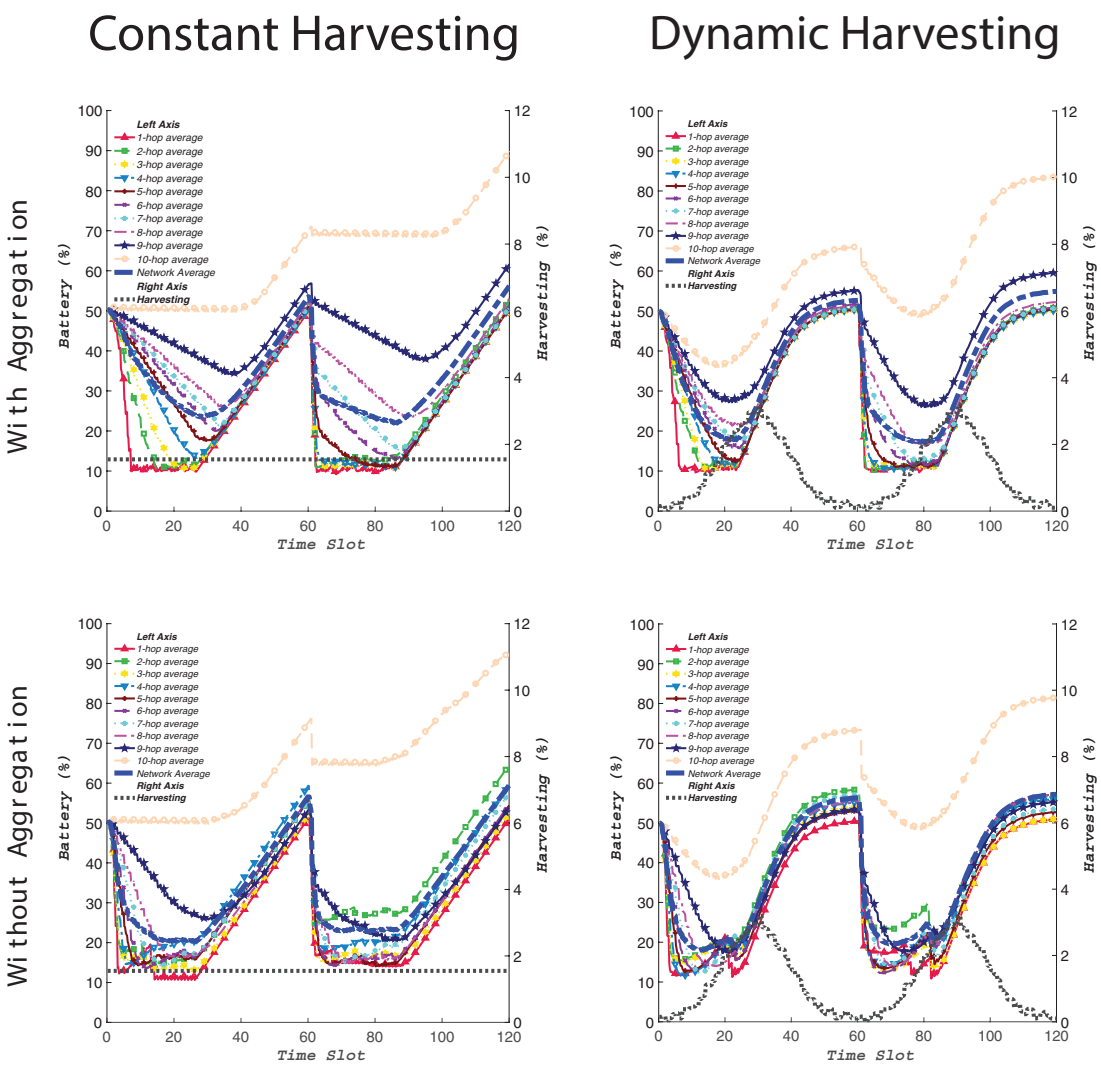


Figure 6.6: HENO-IoT - Battery of Nodes Over Time with $Sx = 5$m and $Tx = 5$m

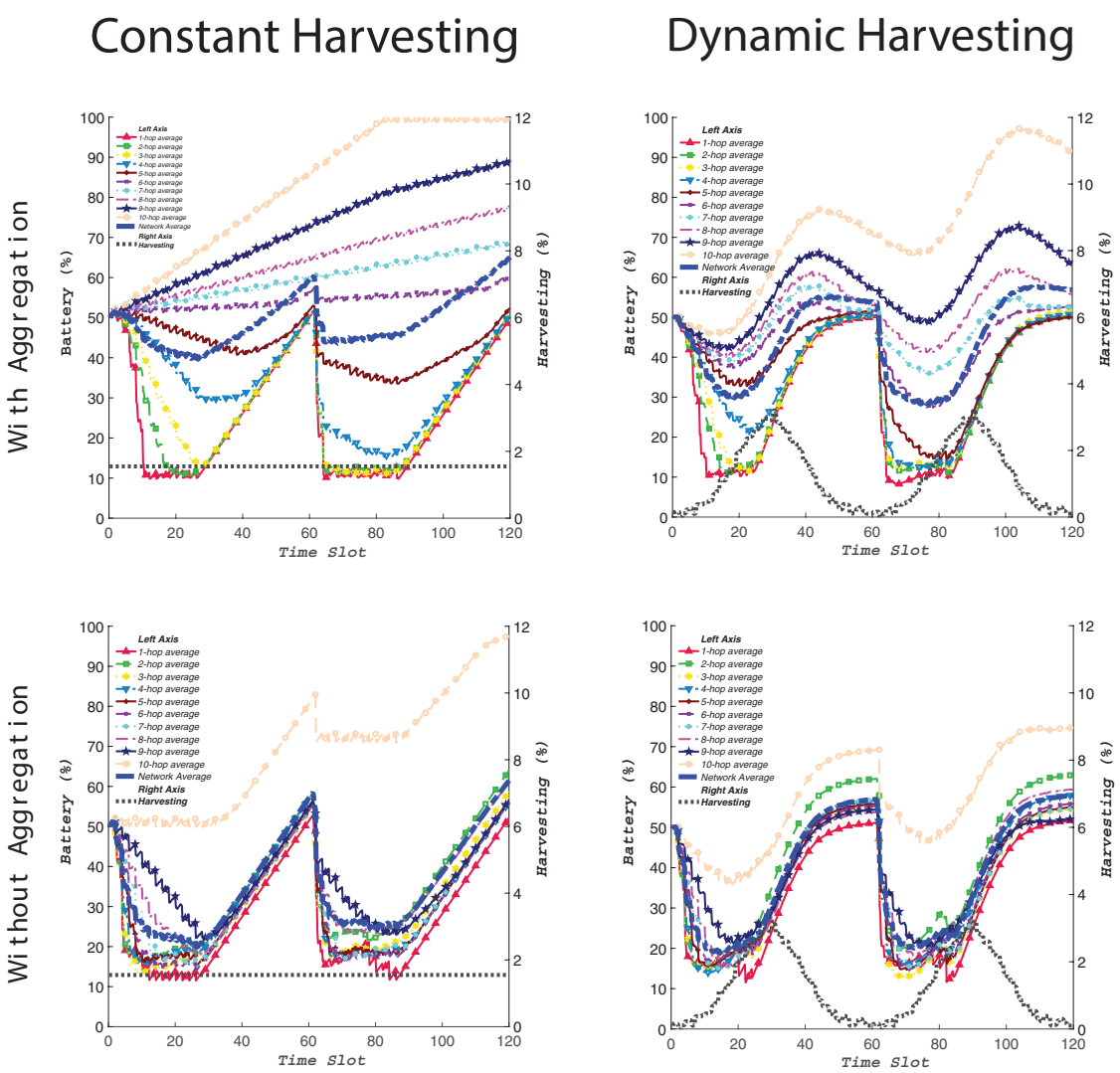
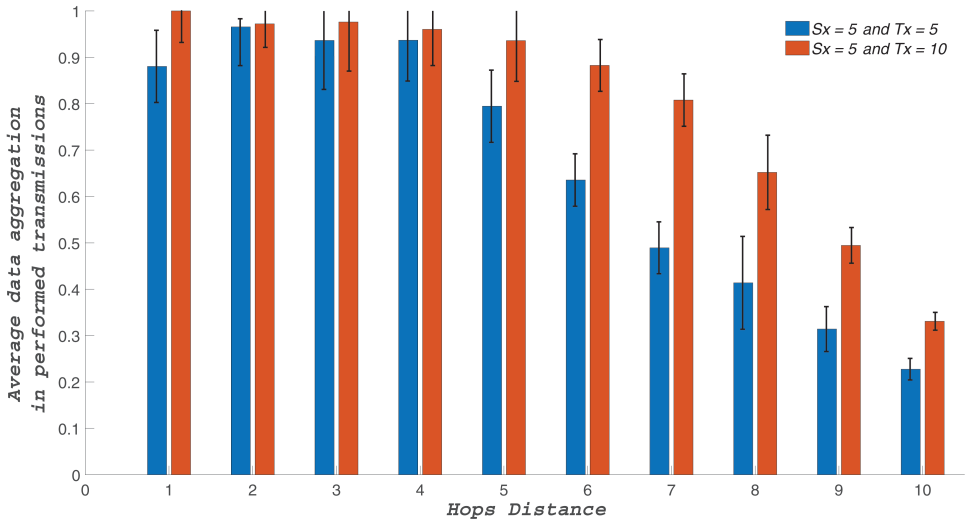
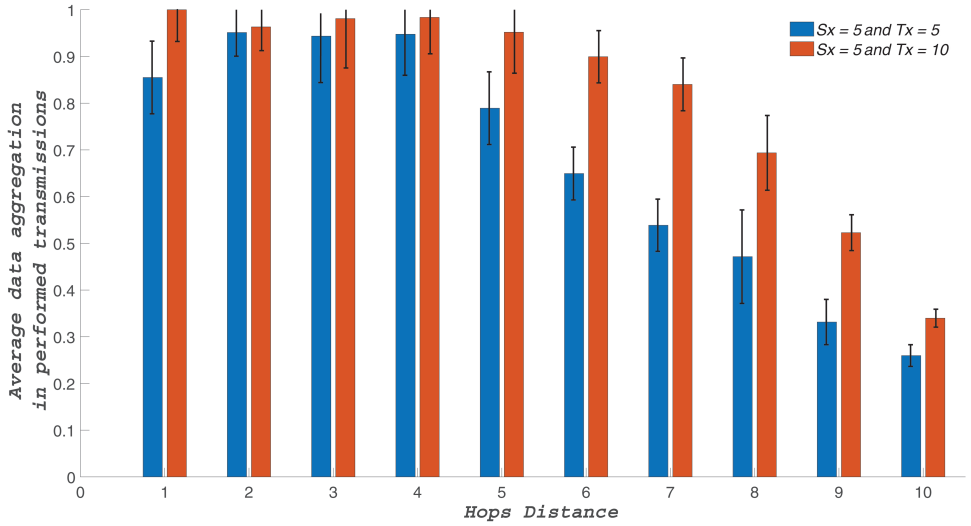


Figure 6.7: HENO-IoT - Battery of Nodes Over Time with Sx = 5m and Tx = 10m

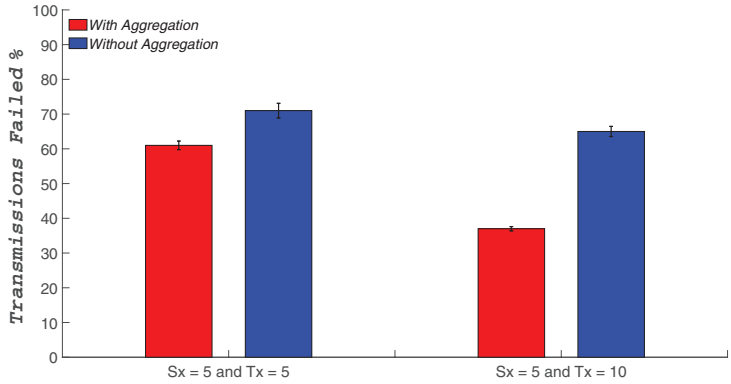


(a) Constant Harvesting

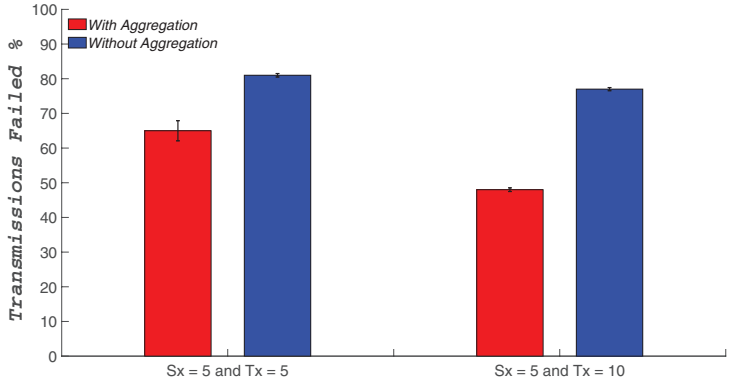


(b) Solar Harvesting

Figure 6.8: Network Data Aggregation.



(a) Constant Harvesting



(b) Solar Harvesting

Figure 6.9: Failed transmissions in different simulations.

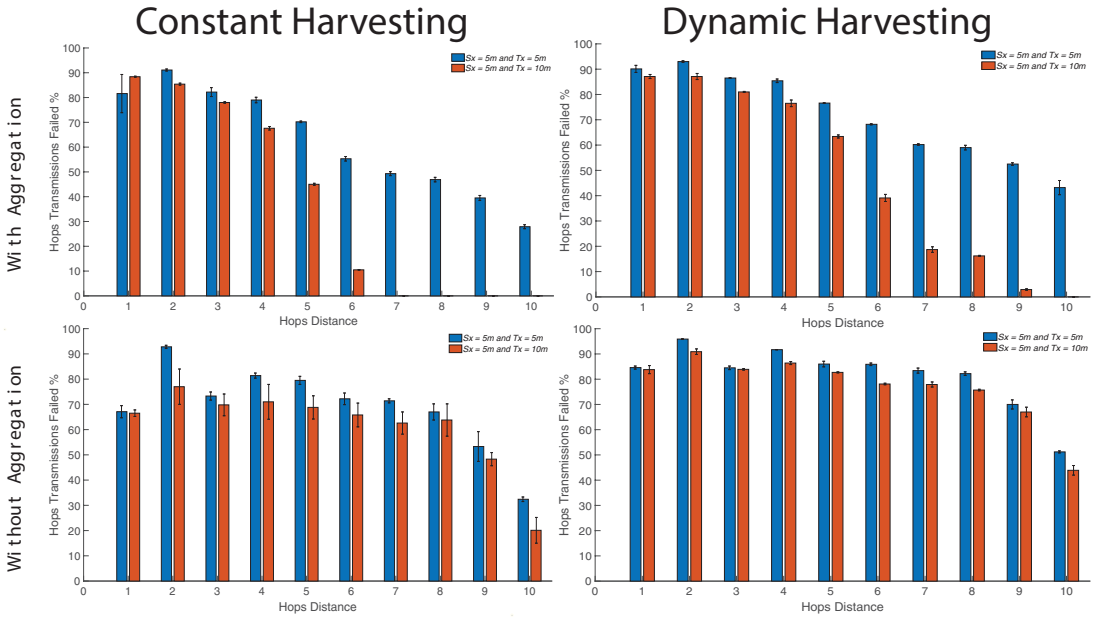
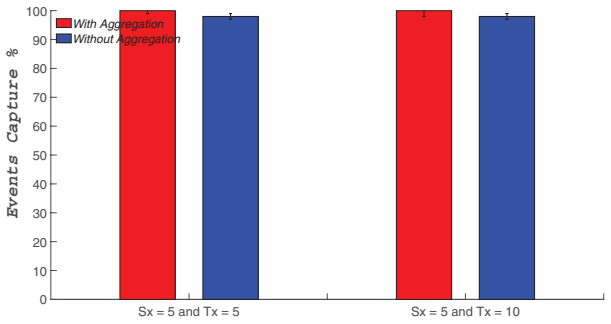
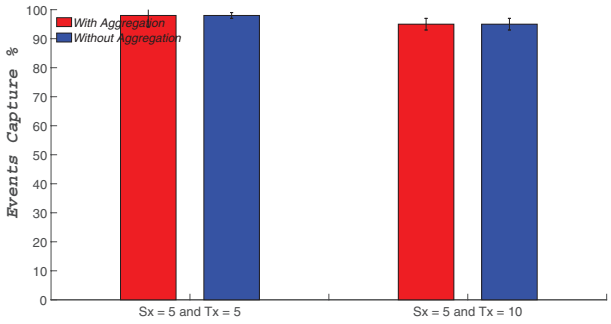


Figure 6.10: Failed transmissions per hop

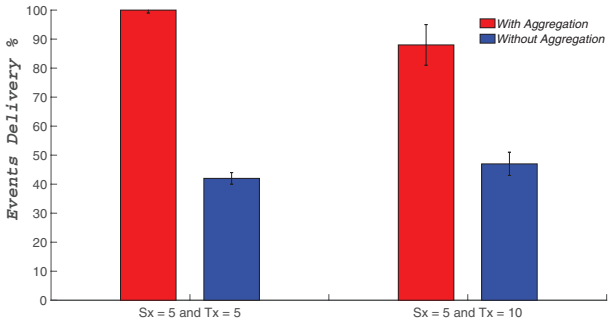


(a) Constant Harvesting

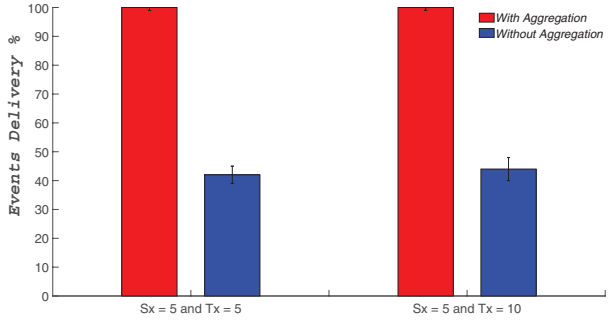


(b) Solar Harvesting

Figure 6.11: Events capture in different simulations.



(a) Constant Harvesting



(b) Solar Harvesting

Figure 6.12: Events delivery in different simulations.

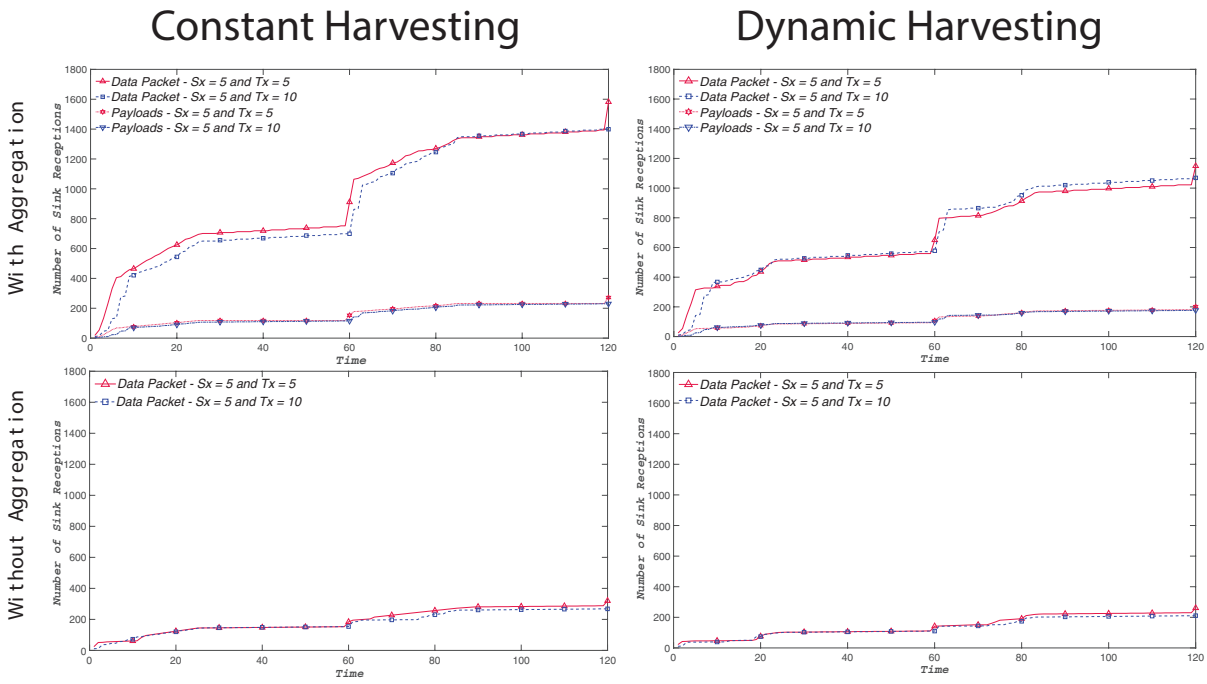


Figure 6.13: Network Utility

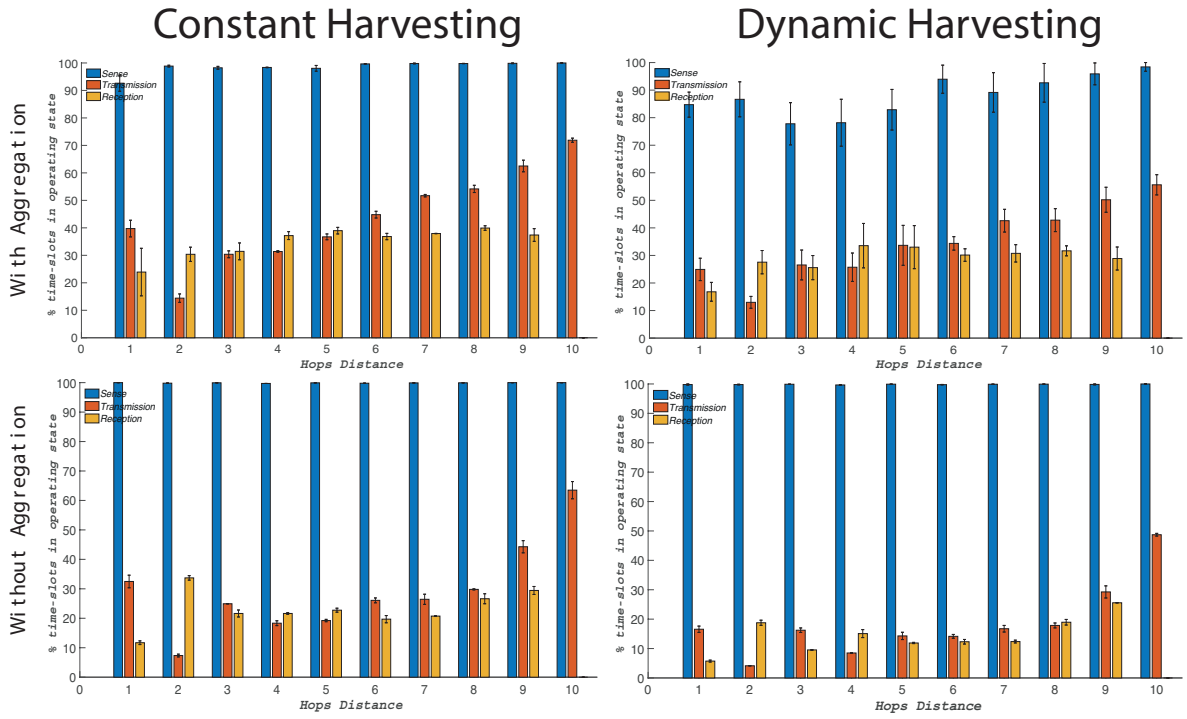


Figure 6.14: Time-slots in operating state - Sx = 5m and Tx = 5m

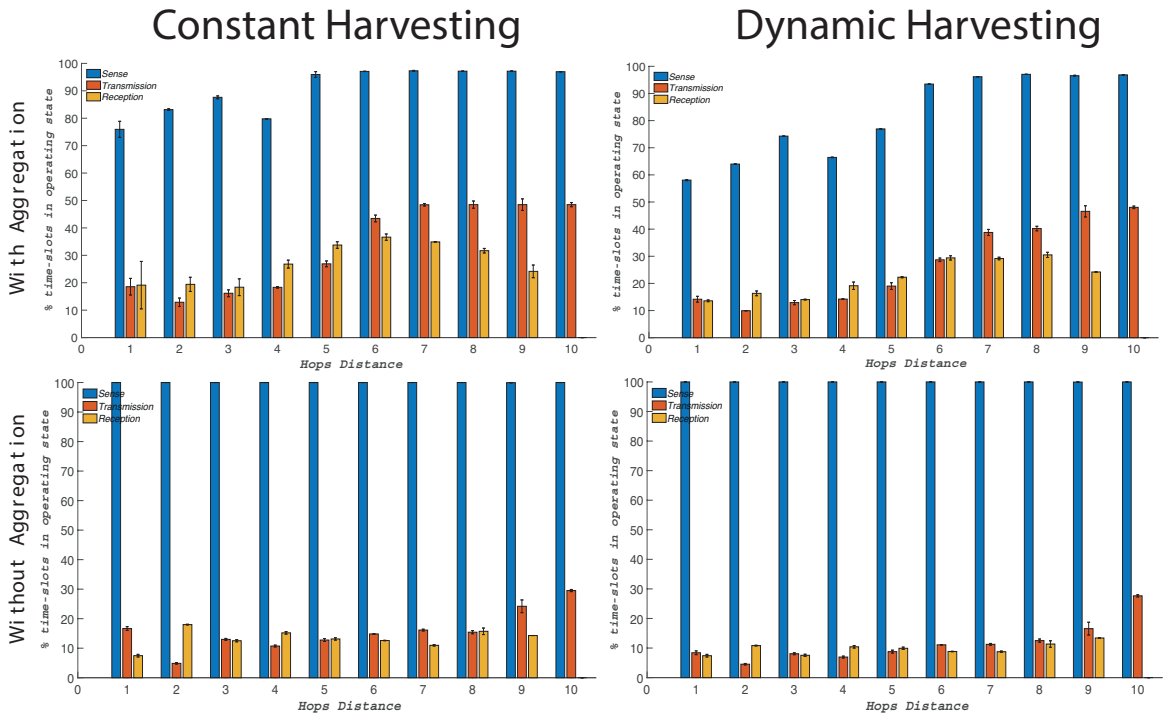


Figure 6.15: Time-slots in operating state - Sx = 5m and Tx = 10m

6.4 Results Analysis and Discussion

The graphics presented in the previous section intended to evaluate all the proposed metrics, showing that the mathematical model, can be applied in a practical context and close to reality. They are satisfied by all heuristics created with great computing capacity and by creating code that can be deployable. Thus, in this section, all graphics are examined with all the necessary detail, dividing the conclusions by the topics that we consider to be the key to this work.

6.4.1 Energy Neutral Operation

As shown in Figures 6.6 and 6.7, the network maintained a neutral energy operation, which can be seen in all the simulations.

Every node starts with 50% of the battery and ends with more battery than the one with which it begins in all the considered cases. This behavior helps the network nodes still begin with an appropriate battery level in the following operating timeframes. If comparable energy efficiency is used (which is again ensured by the ENO constraint), it is assured that the network will run permanently.

As expected, not every nodes have the same energetic behavior. Nodes that are 1 hop away from the sink have a much faster battery break than the rest. This happens due to the proximity of those nodes to the sink. They are the only ones capable of making a direct connection with the sink, which is responsible for delivering all the information created by the network. This undoubtedly leads to greater energy consumption at this level. However, if we observe the

behavior throughout the harvesting cycle, we find that these nodes can achieve neutral energy operation, even when operating in critical conditions.

When the batteries approach 10% (critical zone), the sensors reduce the number of transmissions they carry out in order to recover energy, using harvesting. At the end of the first period of finite-horizon - representing the end of the first day/night cycle - the queue has, on average, 70% payloads. Therefore, at the moment when the second day/night cycle begins, represented by the beginning of the second harvesting cycle, the nodes will perform several aggregations and much more transmissions than they had at the beginning of the first cycle. This leads to a decrease on the initial battery in the second cycle, which is different from what happened before and it can be observed in Figures 6.6 and 6.7.

Obviously, by ignoring the restriction of the operation energy neutral operation - Equation 3.16, we could maximize the number of packets delivered to the sink. Nevertheless, this would cause the network to lose its ability to operate perpetually.

The battery of 10 hop sensors is much higher than the rest due to the grid topology which unviable nodes to perform any reception. The destination of all communications is always towards the sink, so making this path go through the most distant nodes would only introduce a huge information delay on the network. Given that these nodes only need to transmit the information they created, the energy conditions become healthier. Contrary to these nodes, all the others are engaged in high energy operations, making these nodes more fragile.

6.4.2 Data Aggregation

Nodes perform less transmission when we increase the transmission timer, allowing that the data aggregation level in communications improves. The bar graphics confirm such fact in Figure 6.8. Aggregation rates increase when transmissions are performed nearby the sink node. This happens as we get closer to the sink, where the intermediate nodes start to be much more overloaded with information, since they need to follow up on all the information sent by the nodes that are further away. Consequently, from the same figure, it is possible to observe that the nodes that are furthest from the sink, affect a lower percentage of aggregation. This would be expected, as these nodes do not have complete information to send, which means that they carry out transmissions with incomplete packets, respecting the maximum delay allowed Tx_{Delay}.

However, a secret sequence may be deduced. We may expect an increase in accumulation in the cascade effect from the outside to the inside. Through running the nodes in a managed form, the network can make the most of the aggregation's energy savings through gradually storing payloads in closer nodes to sink over time.

The statement above further suggests that increasing transmission timer Tx increases the overall network data aggregation rates. This is not necessarily true. In Figure 6.8, the increase

of Tx value will not significantly benefit data aggregation rates since the network has limited physical resources, i.e., the number of nodes, RAM limitations, maximum transmission length, etc. This increase in overall network aggregation is more notable on the more distant nodes from the sink. This happens because the closest nodes operate in more critical conditions, making them reach their maximum processing capacity in the two simulations presented, not taking advantage of the Tx increase.

However, it is not easy to correlate the harvesting profiles (constant versus dynamic) with the data aggregation patterns. It seems that the aggregation behavior is equally distributed over time. Therefore, the different harvesting patterns do not influence the aggregation rate in the network, as it is continuously optimizing its transmissions.

When choosing a value for Tx, a trade-off must be considered. To a certain extent, increasing Tx will bring further data communication efficiencies due to the higher rates of data aggregation. Decreasing Tx will tend to lower those metrics, although the lifetime of the received data payloads in the sink is decreased. That means, they will reach the sink at faster rates.

6.4.3 Network Topology

As we have already discussed in the course of this section, the closest sink nodes are the ones that most frequently carry out communications, since they have to process the information of all nodes with a distance greater than its own. The information path between the periphery and the center of the network, where the sink is located, means that the sensors located most closest to sink are more likely to fail as we approach. If one of these central nodes ceases to function, it will have repercussions for the entire network, since one sensor's failure causes the overload of another. This can cause the remaining nodes to fail more quickly, causing a cascade effect after the first failure. From the analysis of Figure 6.10 we can confirm such thought. The figure shows a clear increase in transmission failures at the nodes that are closest to the sink. The mentioned cascade effect can be seen in the graphs corresponding to the simulations that use aggregation. Here, it is visible that, by increasing the sink node distance, we reduce the percentage of failed transmissions due to the decrease of the amount of information that the nodes need to process. It is also noticeable that, with an increase of Tx, the percentage of failed transmissions is null in some situations.

Thus, to improve this point in a real scenario, it is recommended that the number of sensors which communicate directly with the sink is significant to the network becoming more resilient the periphery and the center of the network, where the sink is located, means that the sensors located on these most critical routes are more likely to fail as we approach the sink. If one of these central nodes ceases to function, it will have repercussions for the entire network, since one sensor's failure causes the overload of another. This can cause the remaining nodes to fail

more quickly, causing a cascade effect after the first failure. Thus, to improve this point in a real scenario, it is recommended that the number of sensors which communicate directly with the sink is significant to the network becoming more resilient. The fact that we use a multi-hop model is exceptionally advantageous. It allows the nodes to adapt to these failures and always be able to find alternative routes, never forgetting the energy optimization at every moment. In contrary to the most information protocols, where the path is defined initially and is unique, our approach makes it possible to choose several communication routes to the sink. Therefore, a node does not need to send packets to the same neighbor at each time, which minimizes the probability of the sensors' failure, since the network always balances the shipments.

6.4.4 Failed transmissions

The number of failed transmissions can be observed in Figure 6.9, in which four different simulations are represented. It can be seen that data aggregation reduces the number of unperformed transmissions, since it allows more data in each message.

On the two left columns, corresponding to the simulation when transmission timer Tx is equal to 5 minutes - this reduction is 16%. However, when we increment this timer, there is a 29% improvement. The nodes reduce the number of transmissions if the stored energy in the battery approaches 10%. If there are less frequent transmissions, it means that more data is stored in the queue which causes data aggregation.

As explained earlier, the number of packets presented in each sensor's queue at the beginning of the second cycle is different from what happened at the beginning of the simulation. Thus, by the battery analysis graphics, we observed a substantial increase in battery consumption in the second cycle, reflected in the other metrics. This causes the need of the second cycle sensors to recover more energy, compared to the ones on the first harvesting cycle. In Figure 6.10, we can see that the percentage of failed transmissions increased at the closest nodes to the sink. This proves the increase in payload on the queue as already mentioned. When the nodes need to stop transmitting to guarantee ENO, they start accumulating packets in their queue. As the nodes closest to the sink are the ones that process a more significant amount of information, they are more potential to complete the queue quickly. This happens due to the fact that these sensors are the most predisposed to failure in their communications, as we can see in the Figure 6.10, since they are the ones that communicate with all the network information to the sink.

When choosing a value for Tx, there must be a trade-off. The network utility increases with Tx, at the expenses of higher latency. However, a sharp increase in the value of Tx will not benefit data aggregation or successfully transmission rates significantly, since the network has limited physical resources. This is why by increasing Tx to twice the time of Sx, we were able to reduce the number of unsuccessful transmission rates, by means of aggregation, by 17%. However, this reduction is only of 4% when compared with the case without aggregation. These

results showed that there were no significant benefits from the increase in transmission time, in this type of aggregation.

6.4.5 Network Utility

When comparing the network usefulness in simulations where constant harvesting was carried out versus dynamic harvesting, we can not note any significant difference between them. The plot in Figures 6.6 and 6.7 helps prove this point, where it can be shown that each pair of lines with the same degree of harvesting and hop follows a very similar network utility scheme. The difference is that the dynamic harvesting line is increasing and decreasing smoothly. The energy collection goes up and down, i.e., with dynamic harvesting, the battery lines follow a curve pattern, contrary to what happens in constant harvesting that follows a straightway.

Let us study the complex trends of harvesting. The network is configured by only using the correct amount of energy without operating exploiting the nodes when it has the most energy available. This is done to prevent future power outages and operating with caution, in cases of less access to energy, knowing that power can be recover later when it is more accessible. Thus, whenever there is a significant amount of harvesting until the end of the cycle, the nodes can lower their energy level, as the heuristic guarantees an ENO operation. This means that, even if the amount of harvesting is increasing, heuristics can prevent the nodes from making communications, in order to not lower their energy level and guarantee an ENO operation at the end of the cycle. This behavior is the network's reaction to save its resources, as long as it retains the minimum services imposed by the restrictions of the model.

It should be noted that we are solving a deterministic model. That is, each sensor knows, at all times, the harvesting available to receive until the end of the simulation and can act by following this data.

The Figure 6.13 is used to prove some of the conclusions we already obtained at this stage of the work. This shows the sum of the packets received by the sink over time. When analyzing the simulations with the two harvesting profiles, we observe that there is practically no difference between them, which proves the idea that the network always operates optimally at all times. The nodes carry out more sense operations when the availability of energy is more generous. However, it maintains the same communication ratio throughout the simulation. The harvesting profile is not the only action interfering in the network's capacity to deliver the packets created in the sink. On the other hand, when comparing the simulations with and without aggregation, the apparent increase in packets' rate delivered in simulations when using aggregation is noticeable. This is because, with the same number of communications, we managed to send a larger number of payloads, making the battery more profitable, as we can observe in the Figures 6.6 and 6.7. Thus, when we compare the number of packets received by the sink in Figure 6.13 with the number of failed transmissions in Figure 6.9, we can infer that the packets sent, which is the

number of communications made, are very similar in the two harvesting profiles, if we equate simulations with and without aggregation. Naturally, simulations with aggregation substantially increase the network's usefulness - Figure 6.13, as they reduce failed transmissions - Figure 6.9 and maximize the battery of the nodes - Figures 6.6 and 6.7.

Figure 6.13 prove the idea already introduced in the analysis of the Energy Neutral Operation. Here you can see the increase in aggregation at the beginning of the second harvesting cycle - see time slot 60. When looking at this moment, it is possible to observe that the difference between the number of packets and the number of payloads received in the sink increases faster at the beginning of the second cycle. This proves the theory that during the time that the nodes are recovering energy, their packet queue buffer starts to fill, causing an increase in aggregation when the energy conditions allow data transmission.

Furthermore, another essential conclusion derives from Figures 6.14 and 6.15. In these figures, it is possible to observe that the percentage of senses made is lower when we use data aggregation. However, given that the percentage of communication failures is also inferior - Figures 6.9 and 6.10, these simulations can deliver more packets to the sink, contributing to better network utility - Figure 6.13.

Thus, it is natural to conclude that the nodes closest to the sink are the ones that perform the sense operation less often. It is a normal thing to happen as the nearest nodes are continually being used for transmissions, so it is necessary to save battery to transmit it. With aggregation, nodes can be more often in transmission and reception because they make better battery management. This metric is lower for nodes at 1 hop because the percentage of time that nodes are in transmission decreases with the proximity to the sink for a reason already mentioned. It is also possible to conclude that dynamic harvesting significantly impacts the percentage of times the nodes are sensing, which is justified by the need for more rigorous battery management.

6.4.6 Events

Events are guaranteed to be captured by at least one node. They are detected if the sense operation is executed during the event occurrence. If we look at Figure 6.11, we can observe that the events capture are successful in all simulations.

Due to the more significant lack of battery in simulations that do not use aggregation, we can significantly improve the delivery of the events to the sink node while using aggregation. We obtained more 52% events detected - see Figure 6.11. This happens because, if we do not use aggregation, the battery performance decreases substantially, so the sense and transmission operation may fail more times to recover more battery.

Similar to what happens on the data aggregation, it is not easy to correlate the harvesting profiles (constant versus dynamic), which supports the idea of data acquisition and aggregation equally distributed over time.

7

Conclusions

Energy Neutral Operation is a challenge for IoT networks. Many factors can cause nodes to run out of energy in the long-term. However, this is a challenge that is worth tackling since it represents a greener method of powering IoT devices and also reduces maintenance costs.

This work describes a Mixed Integer Linear Programming model and introduces a distributed heuristic to achieve ENO in IoT networks. Both solutions consider periodic and event-based traffic and use data aggregation to regulate network traffic, which adapts the energy consumption.

The distributed heuristic has been evaluated in a simulation environment. The solutions' improvements are set out in terms of residual battery, successful transmission and the event-traffic that is effectively delivered to the sink.

More importantly, this work demonstrates that the entire mathematical model developed can be applied in a physical configuration, close to reality. Thus, all developed heuristics are involved in a real environment, validating all the conclusions drawn from the theoretical model and proving this model's veracity in an entire IoT context.

From the state of the art analysis, we can conclude that no work presented can bring together the four main topics: ENO, Data Aggregation, Event Capture and Multi-hop Network. Similar to this work, when we analyzed all these factors in the same simulation and managed to cross the conclusions between them, we can say that this work brings a clear improvement and an advance, regarding the state of the art.

Future Work

As future investigation, we propose creating an online platform that provides the harvesting values available in the most updated way possible. This would allow the sensors, taking into account the history, to make a more realistic forecast of the amount of harvesting that they would have available by the end of each cycle.

Another future work introduces a layer capable of predicting the natural wear and tear of the sensors' physical batteries. As a result, it can adjust all the necessary calculations to keep the energy operation neutral for a longer time.

In order to maximize the energy available to all sensors as much as possible, balancing the network regardless of topology, we propose the study and implementation of Wireless Energy Transfer between the nodes, making the nodes with abundant energy transfer energy to the most critical nodes.

Other future work is about studying different network topologies, such as a random topology, given that the results are already validated with this thesis. Finally, an exciting work would be the study on optimizing the parameters for the delay of the Sx sense and Tx transmissions, analyzing which are the best values taking into account the utility of the network.

Bibliography

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[2] K. Mekkaoui and A. Rahmoun, *Short-hops vs. long-hops-energy efficiency analysis in wireless sensor networks*. 1, 2011, vol. 825.

[3] J. Torrado, "Modeling And Analysis of Energy Harvesting IoT Networks," Ph.D. dissertation, Universidade de Coimbra, 2018.

[4] R. Jin, X. Li, B. Yan, X. Li, W. Luo, M. Ma, J. Guo, J. Kang, Z. Zhu, and S. Zhao, "A nested ecohydrological wireless sensor network for capturing the surface heterogeneity in the midstream areas of the Heihe River Basin, China," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 11, pp. 2015–2019, 2014.

[5] K. S. Adu-Manu, N. Adam, C. Tapparello, H. Ayatollahi, and W. Heinzelman, "Energy-harvesting wireless sensor networks (EH-WSNs): A review," *ACM Transactions on Sensor Networks*, vol. 14, no. 2, 2018.

[6] W. K. Seah, A. E. Zhi, and H. P. Tan, "Wireless Sensor Networks Powered by Ambient Energy Harvesting (WSN-HEAP) - Survey and challenges," in *Proceedings of the 2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace and Electronic Systems Technology, Wireless VITAE 2009*, 2009, pp. 1–5.

[7] C. Wang, J. Li, Y. Yang, and F. Ye, "Combining Solar Energy Harvesting with Wireless Charging for Hybrid Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 3, pp. 560–576, 2018.

[8] T. Sanislav, S. Zeadally, G. D. Mois, and S. C. Folea, "Wireless energy harvesting: Empirical results and practical considerations for Internet of Things," *Journal of Network and Computer Applications*, vol. 121, pp. 149–158, 2018. [Online]. Available: https://doi.org/10.1016/j.jnca.2018.08.002

[9] N. John and A. Jyotsna, "A Survey on Energy Efficient Tree-Based Data Aggregation Techniques in Wireless Sensor Networks," *Proceedings of the International Conference on Inventive Research in Computing Applications, ICIRCA 2018*, no. Icirca, pp. 461–465, 2018.

[10] A. He, J. Long, and J. Zhang, "An Energy-Efficient Multi-Ring-Based Routing Scheme for WSNs," *IEEE Access*, vol. 7, pp. 181 257–181 272, 2019.

[11] M. Kochláň, S. Žák, J. Miček, and J. Milanová, "Control unit for power subsystem of a wireless sensor node," in *Proceedings of the 2015 Federated Conference on Computer Science and Information Systems, FedCSIS 2015*, 2015, pp. 1239–1246.

[12] S. Bhattacharya, S. Sridevi, and R. Pitchiah, "Indoor air quality monitoring using wireless sensor network," in *Proceedings of the International Conference on Sensing Technology, ICST*, 2012, pp. 422–427.

[13] K. K. Khedo, R. Perseedoss, and A. Mungur, "A Wireless Sensor Network Air Pollution Monitoring System," *International Journal of Wireless & Mobile Networks*, vol. 2, no. 2, pp. 31–45, 2010.

[14] M. Suzuki, S. Saruwatari, N. Kurata, and H. Morikawa, "Demo abstract: A high-density earthquake monitoring system using wireless sensor networks," *SenSys'07 - Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems*, pp. 373–374, 2007.

[15] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," *OSDI 2006 - 7th USENIX Symposium on Operating Systems Design and Implementation*, pp. 381–396, 2006.

[16] L. Yu, N. Wang, and X. Meng, "Real-time forest fire detection with wireless sensor networks," *Proceedings - 2005 International Conference on Wireless Communications, Networking and Mobile Computing, WCNM 2005*, vol. 2, pp. 1214–1217, 2005.

[17] R. G. Lee, K. C. Chen, S. S. Chiang, C. C. Lai, H. S. Liu, and M. S. Wei, "A backup routing with wireless sensor network for bridge monitoring system," in *Proceedings - 4th Annual Communication Networks and Services Research Conference, CNSR 2006*, 2006, pp. 157–161.

[18] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128601003024

[19] S. C. Ergen and P. Varaiya, "On multi-hop routing for energy efficiency," *IEEE Communications Letters*, vol. 9, no. 10, pp. 880–881, 2005.

[20] A. Chehri and H. T. Mouftah, "Energy efficiency adaptation for multihop routing in wireless sensor networks," *Journal of Computer Networks and Communications*, vol. 2012, p. 767920, 2012. [Online]. Available: https://doi.org/10.1155/2012/767920

[21] M. Sana and L. Noureddine, "Multi-hop energy-efficient routing protocol based on Minimum Spanning Tree for anisotropic Wireless Sensor Networks," in *Proceedings of International Conference on Advanced Systems and Emergent Technologies, IC_ASET 2019*, 2019, pp. 209–214.

[22] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Proceedings of the Hawaii International Conference on System Sciences*, vol. 00, no. c, p. 223, 2000.

[23] G. E. Zhihui, L. Anzhong, and L. I. Taoshen, "Energy Conservation and Harvesting for Green Communications EEFA: Energy Efficiency Frame Aggregation Scheduling Algorithm for IEEE 802 . 11n Wireless Network," *China Communications*, no. March, pp. 19–26, 201401.

[24] M. B. Krishna and N. Vashishta, "Energy efficient data aggregation techniques in wireless sensor networks," *Proceedings - 5th International Conference on Computational Intelligence and Communication Networks, CICN 2013*, pp. 160–165, 2013.

[25] Y. Xiao, X. Zhao, H. Wang, and C. H. Hsu, "Energy-aware multipath routing for data aggregation in wireless sensor networks," *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, vol. 2015-April, pp. 829–832, 2014.

[26] S. Ulukus, A. Yener, E. Erkip, O. Simeone, M. Zorzi, P. Grover, and K. Huang, "Energy Harvesting Wireless Communications: A Review of Recent Advances," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 3, pp. 360–381, 2015.

[27] N. Rajathi and L. S. Jayashree, "Energy efficient grid clustering based data aggregation in Wireless Sensor Networks," *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, pp. 488–492, 2017.

[28] M. Zareei, C. Vargas-Rosales, R. Villalpando-Hernandez, L. Azpilicueta, M. H. Anisi, and M. H. Rehmani, "The effects of an Adaptive and Distributed Transmission Power Control on the performance of energy harvesting sensor networks," *Computer Networks*, vol. 137, pp. 69–82, 2018.

[29] S. Yang, Y. Tahir, P. Y. Chen, A. Marshall, and J. McCann, "Distributed optimization in energy harvesting sensor networks with dynamic in-network data processing," in *Proceedings - IEEE INFOCOM*, vol. 2016-July. IEEE, 2016, pp. 1–9.

[30] G. Jackson, Z. Qin, and J. A. McCann, "Long Term Sensing via Battery Health Adaptation," in *Proceedings - International Conference on Distributed Computing Systems*. IEEE, 2017, pp. 2240–2245.

[31] T. N. Le, A. Pegatoquet, O. Berder, and O. Sentieys, "Energy-Efficient Power Manager and MAC Protocol for Multi-Hop Wireless Sensor Networks Powered by Periodic Energy Harvesting Sources," *IEEE Sensors Journal*, vol. 15, no. 12, pp. 7208–7220, 2015.

[32] A. Castagnetti, A. Pegatoquet, C. Belleudy, and M. Auguin, "A framework for modeling and simulating energy harvesting WSN nodes with efficient power management policies," *Eurasip Journal on Embedded Systems*, vol. 2012, no. 1, p. 8, 2012.

[33] D. Gao, Y. Liu, F. Zhang, and J. Song, "Data Aggregation Routing for Rechargeable Wireless Sensor Networks in Forest Monitoring," *Wireless Personal Communications*, vol. 79, no. 1, pp. 773–788, 2014.

[34] S. Jeong, H. Kim, D. K. Noh, and I. Yoon, "Energy-aware data aggregation scheme for energy-harvesting wireless sensor networks," in *Computer Communication and the Internet (ICCCI), 2016 IEEE International Conference on*. IEEE, 2016, pp. 140–143.

[35] A. Gaglione, D. Rodenas-Herraiz, Y. Jia, S. Nawaz, E. Arroyo, C. Mascolo, K. Soga, and A. A. Seshia, "Energy Neutral Operation of Vibration Energy-Harvesting Sensor Networks for Bridge Applications," *International Conference on Embedded Wireless Systems and Networks (EWSN)*, pp. 1–12, 2018.

[36] A. H. Dehwah, J. S. Shamma, and C. G. Claudel, "A distributed routing scheme for energy management in solar powered sensor networks," *Ad Hoc Networks*, vol. 67, pp. 11–23, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870517301749

[37] Y. Zhu, Y. Liu, and L. M. Ni, "Optimizing event detection in low duty-cycled sensor networks," *Wireless Networks*, vol. 18, no. 3, pp. 241–255, 2012.

[38] Z. Ren, P. Cheng, J. Chen, D. K. Yau, and Y. Sun, "Dynamic activation policies for event capture in rechargeable sensor network," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3124–3134, 2014.

[39] N. Correia, D. Sacramento, and G. Schutz, "Dynamic Aggregation and Scheduling in CoAP/Observe-Based Wireless Sensor Networks," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 923–936, 2016.

[40] D. Sacramento, G. Schutz, and N. Correia, "Aggregation and scheduling in CoAP/Observe based wireless sensor networks," in *IEEE International Conference on Communications*, vol. 2015-Septe. IEEE, 2015, pp. 654–660.

[41] W. Colitti, K. Steenhaut, N. De Caro, B. Buta, and V. Dobrota, "Evaluation of constrained application protocol for wireless sensor networks," in *2011 18th IEEE Workshop on Local Metropolitan Area Networks (LANMAN)*, 2011, pp. 1–6.

[42] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "A first look at cellular machine-to-machine traffic - Large scale measurement and characterization," *Performance Evaluation Review*, vol. 40, no. 1 SPEC. ISS., pp. 65–76, 2012.

[43] K. Pahlavan and A. H. Levesque, "Wireless Data Communications," *Proceedings of the IEEE*, vol. 82, no. 9, pp. 1398–1430, 1994.

[44] Z. Ren, P. Cheng, J. Chen, D. K. Yau, and Y. Sun, "Dynamic activation policies for event capture in rechargeable sensor network," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3124–3134, 2014.

[45] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and Classification of Acoustic Scenes and Events," *IEEE Transactions on Multimedia*, vol. 17, no. 10, pp. 1733–1746, 2015.

[46] G. Feltrin and N. Popovic, "Monitoring of strain cycles on a railway bridge with a wireless sensor network," *IABSE Conference, Geneva 2015: Structural Engineering: Providing Solutions to Global Challenges - Report*, vol. 1911, no. July, pp. 1311–1318, 2015.

[47] C. F. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: Random & coordinated sleep algorithms," in *Third International Symposium on Information Processing in Sensor Networks, IPSN 2004*, 2004, pp. 433–442.

[48] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki - A lightweight and flexible operating system for tiny networked sensors," in *Proceedings - Conference on Local Computer Networks, LCN*, 11 2004, pp. 455–462.

[49] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with COOJA," *Proceedings - Conference on Local Computer Networks, LCN*, pp. 641–648, 2006.

[50] MATLAB, *9.7.0.1190202 (R2019b)*. Natick, Massachusetts: The MathWorks Inc., 2018. [Online]. Available: https://www.mathworks.com/products/matlab.html

[51] N. Reijers and K. Langendoen, "Efficient Code Distribution in Wireless Sensor Networks," *Proceedings of the Second ACM International Workshop on Wireless Sensor networks and Applications, WSNA 2003*, pp. 60–67, 2003.

[52] P. Levis and D. Culler, "Maté: A tiny virtual machine for sensor networks," in *Operating Systems Review (ACM)*, vol. 36, no. 5. Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems, 2002, pp. 85–94.

[53] A. Boulis, C. C. Han, and M. B. Srivastava, "Design and implementation of a framework for efficient and programmable sensor networks," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services, MobiSys 2003*, 2003, pp. 187–200.

[54] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler, "The emergence of networking abstractions and techniques in TinyOS," *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation - Volume 1*, p. 1, 2004. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251175.1251176

[55] M. Meister, "Data Gathering in Wireless Sensor Networks using IPFIX under Contiki," in *University of Zurich*, 2014, pp. 1–73.

[56] A. Dunkels, O. Schmidt, T. Voigt, and M. Ali, "Protothreads: Simplifying event-driven programming of memory-constrained embedded systems," in *SenSys'06: Proceedings of the Fourth International Conference on Embedded Networked Sensor Systems*. Proceedings of the Fourth International Conference on Embedded Networked Sensor Systems, 2006, pp. 29–42.

[57] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim," *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, p. 126, 2003.

[58] A. Cavaco, H. Silva, P. Canhoto, S. Neves, J. Neto, and M. Collares Pereira, "Global Solar Radiation in Portugal and its variability, monthly and yearly," *WES 2016 - Workshop on Earth Sciences, Institute of Earth Sciences*, p. 32, 2016.

[59] A. Dunkels, "The ContikiMAC Radio Duty Cycling Protocol," *SICS Technical Report T2011:13 , ISSN 1100-3154*, pp. 1–11, 2011. [Online]. Available: http://dunkels.com/adam/dunkels11contikimac.pdf

[60] M. O. Farooq and T. Kunz, "Contiki-based IEEE 802.15.4 channel capacity estimation and suitability of its CSMA-CA MAC layer protocol for real-time multimedia applications," *Mobile Information Systems*, vol. 2015, 2015.

[61] A. Dunkels, F. Österlind, and Z. He, "An adaptive communication architecture for wireless sensor networks," *SenSys'07 - Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems*, pp. 335–349, 2007.

[62] M. Kodialam and T. Nandagopal, "Characterizing Achievable Rates in Multi-hop Wireless Networks: The Joint Routing and Scheduling Problem," in *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*. ACM, 2003, pp. 42–54.

[63] L. Chen, S. H. Low, M. Chiangs, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Proceedings - IEEE INFOCOM*, 2006.

[64] R. Rajagopalan and P. K. Varshney, "Data-aggregation techniques in sensor networks: A survey," *IEEE Communications Surveys and Tutorials*, vol. 8, no. 4, pp. 48–63, 2006.

[65] S. He, J. Chen, D. K. Yau, H. Shao, and Y. Sun, "Energy-efficient capture of stochastic events under periodic network coverage and coordinated sleep," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1090–1102, 2012.

[66] Y. Yi and S. Shakkottai, "Hop-by-hop congestion control over a wireless multi-hop network," *Proceedings - IEEE INFOCOM*, vol. 4, no. 1, pp. 2548–2558, 2004.

[67] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power Management in Energy Harvesting Sensor Networks," *ACM Transactions on Embedded Computing Systems*, vol. 6, no. 4, p. 32, 2007. [Online]. Available: https://doi.org/10.1145/1274858.1274870

[68] V. Raghunathan, S. Ganeriwal, and M. Srivastava, "Emerging techniques for long lived wireless sensor networks," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 108–114, 2006.

[69] D. Niyato, E. Hossain, and A. Fallahi, "Sleep and wakeup strategies in solar-powered wireless sensor/mesh networks: Performance analysis and optimization," *IEEE Transactions on Mobile Computing*, vol. 6, no. 2, pp. 221–236, 2007.

[70] S. Rhee, D. Seetharam, and S. Liu, "Techniques for minimizing power consumption in low data-rate wireless sensor networks," *2004 IEEE Wireless Communications and Networking Conference, WCNC 2004*, vol. 3, pp. 1727–1731, 2004.

Appendix I

Extending Energy Neutral Operation in Internet-of-Things

Marco Silva, Joel Torrado, Marília Curado, *Centre for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, Coimbra, Portugal,* André Riker, *Faculty of Computing, Federal University of Pará, Belém, Brazil* and José Santos, *University of Coimbra, Centre for Mathematics of the University of Coimbra, Department of Mathematics, Coimbra, Portugal*

***Abstract*—With the advances in sustainable energy technology to power small Internet-of-Things (IoT) devices, emerges the challenge to prolong the Energy Neutral Operation (ENO), in which, no node depletes its battery and all nodes avoid battery overflow. To power the IoT networks using environmental-friend sources (*e.g. solar, thermal, and vibration*) is a first stage towards energy sustainable communications, but it is not enough if nodes over or under use their harvested energy. Therefore, communication protocols should be modified to keep ENO as long as possible. This paper presents an optimization model and a distributed heuristic able to achieve and prolong ENO in IoT networks, supporting periodic and event-based traffic. Besides, the proposed solutions achieve ENO by controlling the produced traffic, the transmissions and regulating traffic applying data aggregation. The conducted simulation shows the benefits of the proposed distributed heuristics study in terms of time in ENO and amount of event-traffic delivered to the sink node..**

I. Introduction

BATTERY-POWERED Internet-of-Things (IoT) devices are largely deployed as enablers for buildings monitoring, smart-homes, and smart-cities. The advances in battery technology and related hardware have allowed these IoT devices to efficiently capture and store energy from the environment. Thus, they can replenish their batteries, as long as there is energy to be harvested from sustainable sources, such as solar, thermal, and vibration.

Energy Neutral Operation (ENO) and Sustainable Network Operation are terms used to mean no node runs out of energy during a long period, so the network lifetime prolongs indefinitely. ENO IoT networks can be defined as the network that keep the state that avoids battery overflow and depletion. In ENO, both cases must be avoided as much as possible, but depletion is worse than battery overflow, since it causes operation outage.

From a general perspective, to maintain long-term ENO is the first requirement for the current IoT networks. Long-term ENO means that IoT nodes must be able to maintain a sustainable operation not only for some instants, but throughout days. Enabling long-term ENO reduces the maintenance costs in comparison with the non-rechargeable battery-powered IoT networks.

Regarding the second requirement in ENO scenario, IoT networks must be able to transport data traffic with heterogeneous characteristics. In general, this traffic can be divided in periodic and event-based traffic. Periodic-based traffic is a low-priority data, which is generated following a fixed time interval pattern and it is a permanent communication for a long time window. In opposite, event-based traffic is a high priority data, which is triggered according to a stochastic occurrence and has a limited duration. To deal with these traffics is an important aspect to keep the IoT networks in ENO because they coexist and compete for resources inside the network. For instance, a build-monitoring application might be implemented to receive data report regularly regarding a facility structure and also to gather data when some critical event occur, such as smoke detection. Therefore, to achieve ENO in IoT network it is a requirement to efficiently use resources, manage the communications aspects, for both periodic and event-based traffic.

Other noticeable aspect is that IoT networks produces small payloads, containing short information [1]. However, due to the number of IoT devices, the amount of generated traffic can represent a challenge for the network in ENO. Thus, the third requirement for ENO IoT is to apply mechanisms to control the amount of traffic flowing over the wireless connections. Otherwise, the network resources will become scarce at some time, which would break the ENO. Traffic control to prolong the ENO can be achieved by controlling efficiently the sensing operation and regulating, using data aggregation, the number of forwarded messages.

Table I presents a summary of the three IoT requirements to achieve ENO introduced so far.

TABLE I: Requirements Summary

	Requirements
1.	To enable long-term ENO to achieve a sustainable network operation for a long period of time.
2.	To efficiently manage network's resources to support both periodic and event-based traffic in ENO IoT networks.
3.	To control the amount of traffic generated and communicated by the IoT network, extending ENO.

Many optimization models have been proposed to fulfill requirement 1 using deterministic and stochastic models. However, most of these works do not consider requirements 2 and 3. It means that the network does not support periodic and event-based traffic and also it does not optimizes the traffic production, communication and aggregation to prolong ENO.

This work presents an optimization model and a distributed heuristic that seeks to fulfill the requirements 1 - 3. This work has the following main contributions:

1) It introduces an Integer Linear Programming (ILP) model and a distributed heuristic to achieve ENO in IoT networks.
2) It manages communication aspects to support periodic and event-based traffic.
3) It regulates the production and communication of network traffic using energy-aware data aggregationn.

The paper is structured as follows. Section II presents the state-of-the-art. Section III and IV introduce the proposed optimum model and the distributed heuristic, respectively. Section V presents the evaluation and the results. Section VI introduces the conclusions.

II. Related Works

Regarding the works that aims to achieve ENO, there are some approaches that use techniques to balance energy generation and storage with consumption. Zareei et al. [2] propose an adaptive scheme for transmission power to make this balancing possible. The main idea is to increase the transmission range of the devices that have abundant energy in their reserves in comparison to their neighbor nodes. On the other hand, nodes in critical energy conditions avoids to perform any communication.

Yang et al [3] also present solutions for ENO. Authors consider the nodes can execute one of the following actions during a time-slot: collect raw data readings, process data packets, transmit or receive data packets. This work formulates ENO as a finite-horizon stochastic optimization problem. At each time-slot the decision variables are sensing rate, wireless transmission rate, and data aggregation. The sensing rate controls the amount of network traffic will be produced. The wireless transmission rate is associated with the energy communication cost. The data aggregation is a mean to perform in-network control on the network traffic. In addition to the formulation, the authors also presents a distributed and sub-optimal algorithm, where each node decides its sensing rate, data aggregation, and computes a scheduler for wireless transmission based on the lightweight Longest Queue First (LQF). This solution was implemented on a testbed, showing improvements in terms of energy sustainability.

Another optimization formulation for ENO is proposed by Jackson et al [4]. However, the objective of this work is to maximize the duty-cycle of the nodes while maintain an Energy Neutral Operation of all the network nodes. One differential of this work is that it considers a battery model in which its capacity suffer degradation over time.

The work proposed by Le et al [5], focused on adapting the duty-cycling to achieve ENO. The proposed solution is a protocol that was developed for periodically powered indoor IoT devices. The protocol applies the Zero Energy Interval predictor (ZEI) proposed in [6] to estimate the harvesting and non-harvesting periods. After estimation, it adapts the node's duty cycle according only to its residual energy in the non-harvesting periods. In addition, the protocol saves portion of the harvested energy for non-harvesting periods.

Seeking ENO, some works are focused on the tradeoff between event-detection rate and energy consumption. This tradeoff exists because spending more time sensing the environment enables the nodes to detect the event, decreasing the statistics of not detected events. Zhu et al [7] proposed to control the wake up and sleep intervals to seek high rate of event detection and low energy consumption, considering nodes with non-replenishable batteries. Yau et al [8] addressed a similar problem, but considering rechargeable batteries. Correia et al [9] and Sacramento et al [10] seek to optimize mainly the energy consumption by applying cache and aggregation schemes on the observe CoAP traffic.

Some other approaches applies data regulation, such as data-aggregation, to achieve ENO. This is the case for the work proposed by Gao et al. [11]. In the presented data-aggregation solution, each node decides its aggregation percent considering the energy in its own reserve. A node always begins its operation performing the lowest aggregation level, and changes it gradually if the energy in the battery increases.

Jeong et al [12] propose a solution to achieve ENO by controlling the data production and managing the transmissions. In this solution, before the message transmission is executed, the devices in the network estimate their remaining energy. If the energy reserve is low, which means the nodes may deplete energy, the node avoid data production and transmissions. When the estimated remaining energy is high, then the node transmits data. Doing this, the devices also try to avoid wasting energy, decreasing the occurrence of battery overflow.

TABLE II: Related Works

Work	Optimization Formulation	Avoids Overflow	Data Aggregation	Event Capture
Zareei et al [2]	No	No	No	Yes
Yang et al [3]	Yes	Yes	Yes	No
Jackson et al [4]	Yes	Yes	No	No
Le et al [5]	No	Yes	No	No
Gao et al [11]	No	No	Yes	No
Jeong et al [12]	No	Yes	No	No

One of the most usual approaches to achieve validity on proposed solutions encompassing the topics mentioned above is to formulate an optimization problem, and next propose an algorithm which can solve that problem. Similarly to this work, some of these solutions (see Table II) propose routing mechanisms and operations scheduling such as to maximize aggregated network utility while dealing with performance constraints. Energy Harvesting IoT Networks are the next step of the technological evolution in this field: they combine the almost-perpetual sensing abilities of EH-WSNs with the capacity of actuating on the environment.

III. Optimal Neutral Operation

We study the problem of composing an optimal scheduling algorithm in a discrete finite-horizon network. Finite capacity batteries power the nodes; still, they are capable of harvesting energy. It is possible to infer from the overall description above (and from Table II) that this work studies jointly these four important IoT techniques: Energy Neutral Operation of a multi-hop network with data aggregation and environmental events capture. To the best of our knowledge, this quartet optimization has not been studied yet.

The considered network consists of a set of statically-deployed nodes $\mathcal{N}$ that includes sensor nodes $\mathcal{S}$ and only one IoT gateway (sink node) $\odot$ such that $\mathcal{N} = \mathcal{S} \cup \odot$. The network can be expressed as a graph $\mathbf{G}(\mathcal{N}, \mathcal{L})$ with arbitrary topologies, where $\mathcal{L}$ represents the set of all possible wireless links. These links are directed, though connectivity is symmetric, i.e., link $(j, i) \in \mathcal{L}$ if and only if $(i, j) \in \mathcal{L}$. Let $N \geq 2$ be the order of the graph (its number of nodes), i.e., $N = |\mathcal{N}|$; and let $L \geq 1$ be the size of the graph (its number of links), i.e., $L = |\mathcal{L}|$. Let $\mathcal{N}_i \subseteq \mathcal{N}$ be the set of all one-hop neighbors of each node $i \in \mathcal{N}$.

The network operates in a finite-horizon period consisting of discrete time slots $t \in \{0, 1, 2, \ldots, T\}, T < \infty$, where $t = 0$ is the initial state of the network. Each time slot has a fixed time span Δ, where all the operations related to a state in entirely executed.

In a particular time slot, there are four possible states for a sensor node: staying idle (sleeping) $\mathtt{z}$, collecting raw sensor readings (sensing) $\mathtt{s}$, transmitting $\mathtt{Tx}$ or receiving $\mathtt{Rx}$ data packets over a wireless link. Thus, the set of states is defined as $\mathcal{M} = \{\mathtt{z}, \mathtt{s}, \mathtt{Tx}, \mathtt{Rx}\}$. A binary variable $S_i^{\mathbf{m}}(t)$ indicates if the sensor node i at time slot t is in the state $\mathbf{m} \in \mathcal{M}$ or not, i.e., $S_i^{\mathbf{m}}(t) = 1$ and $S_i^{\mathbf{m}}(t) = 0$, respectively. Thus, a node can only be in one operational state at each time slot, as showed in the following equation.

$$\sum_{\mathbf{m} \in \mathcal{M}} S_i^{\mathbf{m}}(t) = S_i^{\mathtt{z}}(t) + S_i^{\mathtt{s}}(t) + S_i^{\mathtt{Tx}}(t) + S_i^{\mathtt{Rx}}(t) = 1, \quad (1)$$
$$\forall i \in \mathcal{N}, 1 \leq t \leq T$$

Table III summarizes the symbols used in the basic model of the system presented so far.

TABLE III: Symbols Descriptions: Basic Model

Symbol	Description
$\mathcal{N}$	Set of all nodes (sensor nodes + *sink node* $\odot$).
$\odot$	The *sink node* (or just *sink*) is the only IoT gateway in $\mathcal{N}$
$\mathcal{L}$	Set of all possible wireless links (i, j) between the nodes $\mathcal{N}$ where all data is transferred.
T	Finite-horizon period limit, or operational *time-frame*. $t = 0$ coincide with the initial state of the network.
$\mathcal{M}$	Set of all the operational states of a certain node $i \in \mathcal{N}$. $\mathcal{M} = \{\mathtt{z}, \mathtt{s}, \mathtt{Tx}, \mathtt{Rx}\}$. They are: sleeping $\mathtt{z}$, sensing $\mathtt{s}$, transmitting $\mathtt{Tx}$, or receiving $\mathtt{Rx}$.
$S_i^{\mathbf{m}}(t)$	Binary variable that is 1 if the node $i \in \mathcal{N}$ at *time slot* t is in the state $\mathbf{m} \in \mathcal{M}$. Otherwise, is 0.

A. Traffic Production, Aggregation and Communication

At each time slot t, every node i that is in sensing state ($S_i^{\mathtt{s}}(t)$) collects raw data readings from a hardware sensor. This activity produces a payload of information on that node.

We consider that the network has primary interference: links that share a common node cannot transmit and receive simultaneously, but links that do not share nodes can do so. An identical interference model has been used in [13], [14], [15].

We use a binary variable $a_{i,j}(t)$ to indicate if a link $(i, j) \in \mathcal{L}$ is active ($a_{i,j}(t) = 1$) at time slot t. This activity exists only when node i is in $\mathtt{Tx}$ state and j is in $\mathtt{Rx}$ state during that time slot, i.e.,

$$S_i^{\mathtt{Tx}}(t) + S_j^{\mathtt{Rx}}(t) \geq 2 \cdot a_{i,j}(t), \quad (2)$$
$$\forall (i, j) \in \mathcal{L}, 1 \leq t \leq T$$

We can see the influence of the considered interference model on (2), where the restriction is formulated as inequality and not equality. Thus, this model then allows the following in a particular time slot t: a node in transmitting state can send data to multiple neighbor nodes if those are in receiving state; a node in receiving state can receive data from various neighbor nodes if those are in transmitting state.

A data packet is constructed by two parts: one header and a variable number of payloads. The header contains the communication protocol information, so the header size is constant; a payload carries sensed data. In our model, the payload's size is smaller than the header's. A data packet must have at least one payload and can have a maximum of P_{max} payloads.

Let $p_{i,j}(t)$ and $h_{i,j}(t)$ be the integer number of payloads and headers (data packets) respectively sent, for a successful transmission over the wireless link $(i, j) \in \mathcal{L}$ at time slot t.

It is possible to model the relationship between the number of data packets and payloads involved in that transmission as follows.

$$h_{i,j}(t) = \left\lceil \frac{p_{i,j}(t)}{P_{max}} \right\rceil, \quad (3)$$
$$\forall (i, j) \in \mathcal{L} : a_{i,j}(t) = 1, 1 \leq t \leq T$$

Equation 3 computes the minimum number of necessary headers to send the number of payloads $p_{i,j}(t)$.

Let W_{max} be the constant maximum capacity of a wireless link $(i, j) \in \mathcal{L}$ for any t, i.e., the maximum integer number of data packets that can be successfully transmitted from i to j during any time slot t. Thus,

$$h_{i,j}(t) \leq W_{max}, \quad (4)$$
$$\forall (i, j) \in \mathcal{L} : a_{i,j}(t) = 1, 1 \leq t \leq T$$

Each sensor node $i \in \mathcal{S}$ maintains a *data buffer* $\mathcal{Q}_i(t)$ to store its own sensed data payloads and data payloads received from its neighbors in $\mathcal{N}_i$. Therefore, the buffer $\mathcal{Q}_i(t)$ represents the set of data payloads in node i's *data buffer* at time slot t. We assume that the headers do not make part of this buffer because they are attached and detached only at the instants of sending or receiving data packets, respectively.

Let $Q_i(t) \geq 0$ be the size of buffer $\mathcal{Q}_i(t)$, i.e., $Q_i(t) = |\mathcal{Q}_i(t)|$. Since sensor nodes normally have limited RAM resources, we consider a finite buffer size Q_{max} in our model, i.e., $\forall i \in \mathcal{S}, \forall t, Q_i(t) \leq Q_{max}$.

We implicitly assume that exists some underlying mechanism that perform operations with the existing data inside each nodes' data buffer. These operation allows that data from different sources can be opportunistically aggregated at intermediate nodes before reaching the sink node.

Considering the sensing, transmitting and receiving operations, it can be seen that the buffer size dynamics of a node $i \in \mathcal{S}$ can be described as bounded buffer capacity, which the theoretical description can be formulated as follows.

$$0 \leq Q_i(t) \leq Q_{max}, \quad \forall i \in \mathcal{S}, 1 \leq t \leq T \tag{5}$$

$$Q_i(t) = Q_i(t-1) - p_i^{out}(t) + p_i^{in}(t) + S_i^{\mathtt{s}}(t), \quad \forall i \in \mathcal{S}, 1 \leq t \leq T \tag{6}$$

Here, (6) represents the filling and clearing process of the node buffer; and (5) highlights the bounded buffer capacity.

Besides, the following equations

$$p_i^{out}(t) = \sum_{j:(i,j)\in\mathcal{L}} p_{i,j}(t) \quad \text{and} \quad p_i^{in}(t) = \sum_{j:(j,i)\in\mathcal{L}} p_{j,i}(t)$$

represent the total number of transmitted and received data payloads by that node i at time slot t respectively.

TABLE IV: Symbols Descriptions: Data and Communication

Symbol	Description
$Q_i(0)$	*Buffer* size of a node $i \in \mathcal{N}$ at the start of the simulation, i.e., how many *data payloads* it has in the *buffer* at $t = 0$.
$Q_i(t)$	*Buffer* size of node $i \in \mathcal{N}$ at *time slot* t, i.e., how many *payloads* it has in the *buffer*.
Q_{max}	Maximum number of *data payloads* that a sensor node can have in its *buffer* at any time.
P_{max}	Maximum number of *data payloads* that a transmitted *data packet* can contain.
$h_{i,j}(t)$	Number of *data packets* (*headers*) being transmitted over wireless link $(i,j) \in \mathcal{L}$ at *time slot* t.
$p_{i,j}(t)$	Number of *data payloads* being transmitted over wireless link $(i,j) \in \mathcal{L}$ at *time slot* t.
$a_{i,j}(t)$	Binary variable that is 1 if the link $(i,j) \in \mathcal{L}$ is active during *time slot* t, i.e., there is information flowing from i to j. Otherwise, is 0.
W_{max}	Maximum capacity of a wireless link.

B. Energy Consumption and Harvesting

In our model, the sink node $\odot$ does not have any battery limitations and has virtually infinite buffer size.

Additionally, all the sensor nodes have harvesting devices and batteries with the same characteristics. This implies that all the nodes are in the same initial technical conditions.

We denote the total energy cost for the sensing operation of $i \in \mathcal{S}$ as a constant $C^{\mathtt{s}}$ as we consider that all sensor nodes in the network have the same hardware sensor.

Denote by $C_p^{\mathtt{Tx}}$ and by $C_h^{\mathtt{Tx}}$ a fixed energy cost for successfully transmit a payload and a header respectively. In the same fashion, designate by $C_p^{\mathtt{Rx}}$ and by $C_h^{\mathtt{Rx}}$ the corresponding cost for a node to successfully receive a payload and a header. We can then define the overall energy cost of transmitting p payloads together with h headers over a wireless link $(i,j) \in \mathcal{L}$ at a time slot t as

$$C_p^{\mathtt{Tx}} \cdot p_{i,j}(t) + C_h^{\mathtt{Tx}} \cdot h_{i,j}(t)$$

and

$$C_p^{\mathtt{Rx}} \cdot p_{i,j}(t) + C_h^{\mathtt{Rx}} \cdot h_{i,j}(t)$$

be the total energy cost of receiving that same amount of information.

The amount of deterministic harvested energy from the environment is $H_i(t) \geq 0$.

The residual battery level, with finite-capacity, i.e., $\forall i, t,\ B_i(t) \leq B_{max}$ is $B_i(t) \geq 0$. The total energy consumption is $C_i(t) \geq 0$. Each of the four possible states in $\mathcal{M}$ have a different energy consumption:

- if the node is sleeping ($\mathtt{z}$), the energy consumption is very low and it is constant, i.e., $C_i(t) = S_i^{\mathtt{z}}(t) \cdot C^{\mathtt{z}}$;
- if the node is sensing ($\mathtt{s}$), the energy consumption is also constant, i.e., $C_i(t) = S_i^{\mathtt{s}}(t) \cdot C^{\mathtt{s}}$ and usually $C^{\mathtt{s}} \gg C^{\mathtt{z}}$;
- if the node is transmitting ($\mathtt{Tx}$) or receiving ($\mathtt{Rx}$) data packets, its energy consumption will have a constant component and a component that is a function of how many headers and payloads it is transmitting/receiving.

In time slots when a node is in transmitting or receiving state, the energy consumption can be modeled as follows.

$$C_i(t) = S_i^{\mathtt{Tx}}(t) \cdot C^{\mathtt{Tx}} + C_p^{\mathtt{Tx}} \cdot p_i^{out}(t) + C_h^{\mathtt{Tx}} \cdot h_i^{out}(t) \tag{7}$$

$$C_i(t) = S_i^{\mathtt{Rx}}(t) \cdot C^{\mathtt{Rx}} + C_p^{\mathtt{Rx}} \cdot p_i^{in}(t) + C_h^{\mathtt{Rx}} \cdot h_i^{in}(t) \tag{8}$$

Transmission and reception is considered in eq. 7 and 8, respectively, where

$$h_i^{out}(t) = \sum_{j:(i,j)\in\mathcal{L}} h_{i,j}(t) \ \text{ and } \ h_i^{in}(t) = \sum_{j:(j,i)\in\mathcal{L}} h_{j,i}(t) \tag{9}$$

represent the total number of transmitted and received data packets, respectively. Here, the definition of $p_i^{out}(t)$ and $p_i^{in}(t)$ is the same as in the preceding section.

Considering the hardware of real sensor nodes, the total per-slot energy consumption $C_i(t)$ should be upper bounded by a finite value C_{max} (i.e., $\forall i, t,\ C_i(t) \leq C_{max}$), which depends on the maximum total power consumption of sensor nodes and the duration of a time slot. Practically, $C_{max} \ll B_{max}$, because C_{max} (typically in mJ) is multiple orders of magnitude smaller than B_{max} (typically in kJ).

The dynamic energy system of each sensor node can be modelled as:

$$B_{min} \leq B_i(t) \leq B_{max}, \quad \forall i \in \mathcal{S}, 1 \leq t \leq T \tag{10}$$

$$B_i(t) = B_i(t-1) - C_i(t) + H_i(t), \quad \forall i \in \mathcal{S}, 1 \leq t \leq T \tag{11}$$

$$B_i(T) \geq B_i(0), \quad \forall i \in \mathcal{S} \tag{12}$$

Here, (10) represents the highlights the bounded battery capacity; and (11) represents the recharging and discharging process of the battery. More importantly, the constraint (12) ensures *neutral operation*, which is expected to be achieved by every sensor node in the network, i.e., a node must consume only the energy it harvested – its residual battery level at the end of operation must be always greater or equal than at the beginning of operation.

TABLE V: Symbols Descriptions: Energy Consumption and Harvesting

Symbol	Description
$B_i(0)$	Battery level of node $i \in \mathcal{N} \setminus \odot$ at the start of operation. This is also the value of battery that the node must have at the end of the operational *time-frame*.
B_{min}	Minimum level of battery that any node can reach during operation.
B_{max}	Maximum level of battery that any node can reach during operation.
$H_i(t)$	Amount of increased battery level of a sensor node $i \in \mathcal{N} \setminus \odot$ in a certain *time slot* t. This amount comes from harvested energy from the environment.
$C_i(t)$	Total amount of decreased battery level of a sensor node $i \in \mathcal{N} \setminus \odot$ at *time slot* t.
$C^{\mathtt{z}}$	decreased battery level when in *sleeping* (z) state
$C^{\mathtt{s}}$	decreased battery level when in *sensing* (s) state
$C^{\mathtt{Tx}}$	decreased battery level when in *data transmitting* (Tx) state
$C^{\mathtt{Rx}}$	decreased battery level when in *data receiving* (Rx) state
$C_p^{\mathtt{Tx}}$	decreased battery level when transmitting one *data payload*
$C_h^{\mathtt{Tx}}$	decreased battery level when transmitting one *header* (*data packet*)
$C_p^{\mathtt{Rx}}$	decreased battery level when receiving one *data payload*
$C_h^{\mathtt{Rx}}$	decreased battery level when receiving one *header* (*data packet*)

C. Events

We can consider that an event is a notable occurrence at a particular point in time. In our model, we do not define particularly what an event is, as it depends on the general purpose of the wireless network and the kind of hardware sensor mounted on the set of sensor nodes $\mathcal{S}$. For example, if the deployed nodes are monitoring the fluid pressure at specific points of an industrial piping system, sudden rise or drop of pressure (events) in the neighborhood of those points could be interpreted as pipe obstructions or leakage in those areas, respectively.

Whatever is the case, the general approach is to ensure that every point of the deployment area is covered all the time by at least one active sensing sensor, provided that there is an available sensor in the random placement [16]. Some algorithms, e.g. [17], are designed to minimize the probability that any given point is not covered by an active sensor, provided that it could be covered. Doing so will also maximize the detection probability of any event and ensure instantaneous detection, if the event is within range of at least one sensor that is sensing. In some applications for transient events (e.g., an animal which arrives and then leaves), the goal may be to maximize the detection probability of the events before they disappear, and some delay in the detection is acceptable.

Furthermore, it is widely accepted that real-world events can be modelled as stochastic processes [18]. So, for this model, we consider that the starting time slots and locations of the occurring events are known beforehand, as they follow a predictable distribution.

Thus, let $E_i(t)$ be a binary variable indicating that at time slot t the node $i \in \mathcal{S}$ is close enough to detect an event ($E_i(t) = 1$) or not ($E_i(t) = 0$) before it disappears, as all events have a fixed duration δ. This suggests that is possible that E_i can be 1 to more than one node at the same time, and can be 1 several time slots in a row for the same subset of sensor nodes.

For a certain event be declared as detected or captured, there must be at least one sensor node $i \in \mathcal{S}$ in its neighborhood that is in sensing state while that event exists, i.e.,

$$\exists i \in \mathcal{S}, \exists t \in [\delta_{start}, \delta_{end}] : (S_i^{\mathtt{s}}(t) = 1 \wedge E_i(t) = 1) \quad (13)$$

where $\delta_{start} \in \{1, ..., T - \delta\}$ and $\delta_{end} = \delta_{start} + \delta$ are the starting and ending time slots of the event, respectively.

TABLE VI: Symbols Descriptions: Events

Symbol	Description
$E_i(t)$	Binary variable that is 1 if node $i \in \mathcal{N} \setminus \odot$ is close enough to detect an *event* at *time slot* t. Otherwise, is 0.
$v(t)$	Binary variable that is 1 if an *event* has started at *time slot* t. Otherwise, is 0. This means that $v(t) = \max\{E_i(t) : i \in \mathcal{N} \setminus \odot\}$.
δ	Fixed duration, in *time slots*, of all emerging events.

D. Additional Constraints

It is necessary to impose new restrictions in order to ensure the nodes can only be in a state of transmission or reception mode if they are involved in some communication (eq. 14-15).

$$S_{\odot}^{\mathtt{Rx}}(t) = 1, 1 \leq t \leq T \quad (14)$$

$$S_i^{\mathtt{Tx}}(t) \leq \sum_{j:(i,j)\in\mathcal{L}} a_{i,j}(t), \quad (15)$$

Equation 16 has been included to consider data aggregation.

$$P_{max}\left(h_{i,j}(t) - \bar{f}_{i,j}(t)\right) + 0.1 \cdot \bar{f}_{i,j}(t) \leq p_{i,j}(t) \leq P_{max} \cdot h_{i,j}(t), \quad \forall (i,j) \in \mathcal{L} \quad (16)$$

In addition, to impose a minimum amount of information (β) to be received periodically (α) in the sink node (eq. 17) as well as to be transmitted and measured in each node (eq. 18, 19). Finally, it is necessary to impose a limit (γ) for the portion of information generated in a period that can continue to circulate in the network, penalizing the nodes further away from sink (eq. 20).

$$\sum_{t=(k-1)\alpha+1}^{k\alpha} \sum_{j:(j,\odot)\in\mathcal{L}} p_{j,\odot}(t) \geq (N-1)\cdot\beta, 1 \leq k \leq \frac{T}{\alpha} \quad (17)$$

$$\sum_{t=(k-1)\alpha+1}^{k\alpha} S_i^{\mathtt{Tx}}(t) \geq \mathtt{Tx}_{min}, \forall i \in \mathcal{N} \setminus \odot, 1 \leq k \leq \frac{T}{\alpha} \quad (18)$$

$$\sum_{t=(k-1)\alpha+1}^{k\alpha} S_i^{\mathtt{s}}(t) \geq \mathtt{s}_{min}, \forall i \in \mathcal{N} \setminus \odot, \quad 1 \leq k \leq \frac{T}{\alpha} \quad (19)$$

$$\frac{\sum_{k=1}^{T/\alpha} \left(\sum_{i \in \mathcal{N} \setminus \odot} (D_i \cdot Q_i(k \cdot \alpha)) \right)}{T/\alpha} \leq \gamma \cdot Q_{\odot}(T), 0 \leq \gamma \leq 1 \quad (20)$$

E. Linerization and Final Model

In order to write the problem as a linear program, Eq. 3 and 13 have to be rewritten as Eq. 21 and 22, respectively. Besides, Eq. 5 and 6 and were written 23 and 24.

$$S_i^{\mathtt{Tx}}(t) + S_j^{\mathtt{Rx}}(t) \geq 2 \cdot a_{i,j}(t) \forall (i,j) \in \mathcal{L} \quad (21)$$

$$\sum_{i \in \mathcal{N} \setminus \odot} \left(E_i(t) \cdot \sum_{\tau=t}^{\min\left\{t+\frac{\delta}{2}, T-t\right\}} S_i^{\mathtt{s}}(\tau) \right) \geq v(t) \quad (22)$$

$$a_{i,j}(t) \leq p_{i,j}(t) \leq \infty \cdot a_{i,j}(t), \forall (i,j) \in \mathcal{L} \quad (23)$$

$$Q_i(0) + \sum_{\tau=1}^{t-1} p_i^{in}(\tau) + \sum_{\tau=1}^{t-1} S_i^{\mathtt{s}}(\tau) \geq \sum_{\tau=1}^{t} p_i^{out}(\tau) \quad (24)$$

It is not the purpose of this work to make any probabilistic (e.g. specific distribution) and stochastic assumptions of the dynamic network states, including harvested energy, energy costs (for sensing, sleeping, transmitting, and receiving), as well as transmission and data buffering capacities, or events space-time location. The objective of this model is to seek an algorithm that can solve the following finite-horizon optimization problem, by finding the most suitable state $S_i^{\mathbf{m}}(t) \in \mathcal{M}$ and wireless payload transmission $p_{i,j}(t)$ for each sensor node in $\mathcal{S}$ at each time slot $1 \leq t \leq T$, so as to maximize the number of data payloads that arrive at the sink node $\odot$ at the end of the horizon, i.e., at $t = T$ (25).

maximize

$$Q_{\odot}(T) \quad (25)$$

subject to

Constraints 1, 2, 4, 7 - 12, 14 - 22

TABLE VII: Symbols Descriptions: Additional for Symbols for Full Model Implementation.

Symbol	Description
D_i	Distance to the *sink node*, i.e., the length of the shortest path between node $i \in \mathcal{N}$ and *sink node* $\odot$. Distance of *sink node* to itself is 0.
α	Duration of the period, in *time slots*, in which a minimum amount of *data payloads* will be required to enter the *sink node* $\odot$. This means that from α to α *time slots* a certain minimum amount of *data payloads* (see β) need to reach the *sink node*, with $1 \leq k \leq \frac{T}{\alpha}$, being k the current period.
β	Minimum amount of *data payloads per sensor node* that the *sink node* must receive in each of the time periods k (see α).
$\mathtt{Tx}_{min}$	Minimum amount of *data transmit* (Tx) states that a sensor node must achieve in each of the time periods k (see α).
$\mathtt{s}_{min}$	Minimum amount of *sensing* (s) states that a sensor node must achieve in each of the time periods k (see α).
γ	Value between 0 and 1 that sets what fraction of the total number of *payloads* that enter the *sink node* throughout all the operational *time-frame* can remain outside in the sensor nodes *data buffer*.
$\bar{f}_{i,j}(t)$	Binary variable that is 1 if there is a *data packet* being transmitted over a wireless link $(i,j) \in \mathcal{L}$ at *time slot* t that is not full, i.e., the number of *data payloads* in it is less than P_{max}. Otherwise, is 0.

IV. Distributed Heuristic for Neutral Operation

As the computational costs to compute the optimal solution can be prohibitive, this section presents a heuristic able to approximate the optimal solution within reasonable time.

The proposed heuristic, named Heuristic for Energy Neutral Operation in Internet-of-Things (HENO-IoT), is designed as a time-sloted solution, where for each time-slot the nodes are allowed to performed sleeping, sensing, transmission, and reception. Differently from the optimization model presented in Section III, the heuristic is able to perform more than one operation per time-slot. This change brings our heuristics closer to the real world, where sensor nodes may need to do more than one operation in the same time slot. For example, this is because they may need to capture important information from the network and send it right away, minimizing latency and making the information reach its destination quickly, making the system more reliable.

Table VIII presents the additional symbols used for the HENO-IoT design.

Algorithm 1 State of Operation - Decision

1: **Input:** *time-slot* ;
2: **Output:** Operation state of node i;
3: */* Each node i runs distributively */*
4: */* j represents a neighbor of i */*
5: *time-slot* $= 0$;
6: **while** (*time-slot* $< end$) **do**
7: **if** $AtualTime - TimeLastSx \geq Sx_{Delay}$ **then**
8: **if** SxConditions( ) **then**
9: Sense;
10: **end if**
11: **end if**
12: **if** $AtualTime - TimeLastTx \geq Tx_{Delay}$ **then**
13: **if** TxConditions( ) **then**
14: j $\leftarrow$ Select Neighbor(i); /* Call Algorithm 2 */
15: **if** RxConditions( ) **then**
16: Transmit(i, j);
17: **end if**
18: **end if**
19: Sleep;
20: **else**
21: Sleep;
22: **end if**
23: *time-slot* ++;
24: **end while**
25: ---
26: **procedure** TXCONDITIONS( )
27: TxBtt1 = $B_i(t) + H_i(t) - C_i^{Tx}(t)$;
28: TxBtt2 = $B_i(t) + HT_i(T) - C_i^{Tx}(t)$;
29: **if** (TxBtt1 $\geq 10\% * B_{max}$ **and** TxBtt2 $\geq B_i(0)$) **then**
30: $E_i^{T_x} \leftarrow$ TRUE;
31: **end if**
32: **Return** $E_i^{T_x}$;
33: **end procedure**
34: ---
35: **procedure** RXCONDITIONS( )
36: RxBtt1 = $B_j(t) + H_j(t) - C_j^{Rx}(t)$;
37: RxBtt2 = $B_j(t) + HT_j(T) - C_j^{Rx}(t)$;
38: **if** (RxBtt1 $\geq 10\% * B_{max}$ **and** RxBtt2$\geq B_j(0)$) **then**
39: $E_j^{R_x} \leftarrow$ TRUE;
40: **end if**
41: **Return** $E_j^{R_x}$;
42: **end procedure**
43: ---
44: **procedure** SXCONDITIONS( )
45: SxBtt1 = $B_i(t) + H_i(t) - C_i^{Sx}(t)$;
46: SxBtt2 = $B_i(t) + HT_i(T) - C_i^{Sx}(t)$;
47: **if** (SxBtt1 $\geq 10\% * B_{max}$ **and** SxBtt2$\geq B_i(0)$) **then**
48: $E_i^{S_x} \leftarrow$ TRUE;
49: **end if**
50: **Return** $E_i^{S_x}$;
51: **end procedure**

Algorithm 1 presents the main HENO-IoT pseudo-code, which runs distributively in each node. This algorithm decides the operation state of each node in each time-slot. It is possible to observe that for each slot of time this algorithm checks if the conditions for transmission, reception, and sensing is satisfied. These conditions are checked in the procedures *TxConditions()*, *RxConditions()*, and *SxConditions()*, presented in lines 8, 13, and 15, respectively. The binary variables E_i^{Tx}, E_i^{Rx}, and E_i^{Rx} store true to indicate the conditions are meet for these states for a node i. If the variable E_i^{Sx} is true, then the algorithm assigns a sensing operation for that time-slot (see line 9). If the variable E_i^{Tx} is true, it calls the Algorithm 2 to select a neighbor able to receive data from i.

The neighbor selection (see Algorithm 2) is a choice considering all neighbors. If the sink node is among the candidates, then it is selected. Otherwise, the algorithm selected the node with shorter distance to the sink and with more energy in the battery. The list of neighbors is obtained through broadcast messages from the sink. First, the sink establishes the distance to itself as 0 and starts sending its control messages to all nodes within its reach. In turn, these nodes that receive the message configure their distance with +1 hop count. The receptors of this messages compare its distance (initially defined as infinite) with that of the message. If the hop message distance +1 is less than your current distance, then the current hop is set as hop message +1. The process continues between all the nodes until everyone knows their hop distance.

In order to minimize the transmissions the amount of data transmitted each time must be increased due data aggregation. If all the payloads in the queue have a length longer than the one defined as maximum length, then the node only sends multiples of that value. Otherwise, the node sends the remaining length. Therefore, if we have a queue length longer than the defined as maximum length, we can have the case that a node realizes one or more transmissions (each with the maximum length) and saves the remaining data length, if it exists, because it is not enough to fill the maximum again. Differently, if the initial maximum length to transmit is lower than the maximum defined, all of this data that does not correspond a full transmission is sent, and the queue will stay empty.

It is important to notice that the proposed heuristic does not have any specific procedure that guarantee event detection. However, the events are detected if the sense operation is executed during the event occurrence. Therefore, if the sense operation is performed in every time-slot, all events are detected.

V. EVALUATION AND RESULTS

This section introduces the environment configuration used to test the proposed distributed heuristic and discusses the obtained results. The evaluation was carried out by simulations

TABLE VIII: Symbols Descriptions: Additional Energy Symbols for HENO-IoT

Symbol	Description
$B_i(T)$	Percentage of battery that a node should have at the end of the simulation
$HT_i(T)$	[l]Amount of increased battery level of a sensor node i until the end the simulation time-frame
Tx_{Delay}	Maximum time that a node can retard the transmission
Sx_{Delay}	Maximum time that a node can retard the sensing
$E_i^{R_x}$	Binary variable that is true if the conditions to receive packets is satisfied
$E_i^{T_x}$	Binary variable that is true if the conditions to transmit packets is satisfied
$E_i^{S_x}$	Binary variable that is true if the conditions to sense is satisfied

Algorithm 2 Select Neighbor

```
1: Input: Neighbors[ ] - Neighbors List of node i;
2: Output: bestN - Best Neighbor to send the information;
3: bestN ← Neighbors[0];
4: for N = 0; N < length(Neighbors[ ]);N++ do
5:     if (Neighbors[N] == SinkNode) then
6:         bestN ← Neighbors[N];
7:         Break;
8:     end if
9:     if (SinkDistance(Neighbors[N]) < SinkDistante(bestN)) then
10:        bestN ← Neighbors[N];
11:    end if
12:    if (SinkDistance(Neighbors[N]) == SinkDistante(bestN)) and
   (BttLevel(Neighbors[N])>BttLevel(bestN)) then
13:        bestN ← Neighbors[N];
14:    end if
15: end for
```

to demonstrate the benefits of the proposed solution in terms of neutral operation.

The environment setup is presented in Section V-A. The performance metrics and the results are presented in Sections V-B and V-C, respectively.

A. Environment Setup

The proposed solutions were implemented in the Contiki operational system [19]. Cooja framework [20] was used to embed the Contiki firmware in emulated IoT nodes and to simulate the network. The settings in Table IX were used for Contiki and Cooja in order to conduct the simulations.

TABLE IX: Simulation Settings.

Parameter	Value
Number of IoT devices	121
Simulation Area	200 m x 200 m
Event Duration (time slots)	4
Protocol Stack	RIME
Sensor Mote	Sky mote
Wireless Range	50 m
Node Carrier Sensing Range	50 m
Data Buffer Size	10 Packets
Mac and Physical	IEEE 802.15.4
MAC layer	CSMA-CA
Radio Duty Cycling	ContikiMAC

The simulated network is based on a grid topology, where a particular node is 50m from the neighbors. The data produced by the most distant nodes reaches the sink after performing 10 wireless hops. This topology allows each node to have between 2 to 4 neighbors.

As shown in Equation 5, nodes have limited storage capabilities. For this evaluation, the maximum data in the buffer is 10 packets. Since we are using short IEEE 802.15.4 addressing mode, 102 bytes are maximum transported in a single MAC layer when used with the Rime Protocol Stack [21]. Inevitably, and to take advantage of the maximum capacity of the packets, we set 102 bytes the maximum size of data that each package can carry when transmitted. So, the length of aggregated payloads can not exceed this maximum transmitted unit (MTU).

In order to obtain realistic results, experiments were carried out with 2 day/night cycles where these results can be confirmed since, after the second cycle, the nodes always had a queue with initial packages, and the behavior still maintained the same pattern.

Hence, as our goal is to attain a certain sense of realism in the potential uses for this sort of networks, 5 minutes per time slot is a suitable value. This value respects the assumptions made on the theoretical model, where it is declared that the nodes' operational states require sufficient time to be performed in their entirety before a new time slot initiates. Considering that the active and more consuming states (i.e., sensing, receiving and transmitting) represent operations that an embedded micro-controller can ordinarily perform in less than 2 seconds, 5 minutes gives a substantial operational margin and is proper for most applications of an IoT network with environment-sensing capabilities

The proposed HENO-IoT, which perform data aggregation, has been tested under the two traffic patterns and also compared to a version of itself that does not perform data aggregation.

Regarding the harvesting rates, the tests considered constant energy harvesting and a solar-based trace.

The primary goal of this aspect is to verify that the network is able to operate with diverse external harvesting situations and how its modus operandi depends on the harvesting outline.

The solar-based harvesting values considered are taken from the 2016 Annual Average Value of Solar Radiation and its Variability in Portugal [22]. The measuring unit - kWh/m^2 - represents the amount of energy accumulated over one square meter for one year. Therefore, taking into account the areas where there is no sunlight each evening, we managed to draw a dynamic harvesting line to all the sensors.

The events in the simulations were implemented according to the information described in Section III-C.

B. Performance Metrics

Some metrics have been defined to compute the performance of proposed heuristic.

- **Battery (%)**: As shown in Section III, all the sensor nodes have harvesting devices and batteries with the same characteristics. The nodes have four possible states in each time-slot: idle (sleeping) z, collecting raw sensor readings (sensing) s, transmitting Tx or receiving Rx data packets over a wireless link. Battery (%) is a performance metric that represents the remaining energy in the battery. It is computed using Equation (11) and takes into account the consumed energy, caused by any of the four states, and the energy harvesting. Therefore, is possible to appraise the energy-neutral operation and to examine the network response to various harvesting profiles.
- **Percent of Not Performed Transmissions**: Due to lack of energy, a node may not satisfy the conditions of Algorithm 1 to perform transmissions. To compute this metric, the number of times the node does not satisfy conditions for transmission is divided by the sum of all transmission tentative. It is important to notice that a succeeded transmission is performed only if the receiver also meets the energy conditions imposed by Algorithm

1. This metric allows us to understand the adopted data routes when in variable energy conditions so the network throughput is not compromised.
- **Event-Traffic Delivered on Sink**: According to Algorithm 1, a node must satisfy some conditions to perform sensing. After the execution of a sensing, the node must communicate the sensing readings. This metric is computed by ratio between the number of times at least one message carrying information about a particular event reaches the sink and the total number of events.

C. Results

The results presented are the result of an average of 5 simulation rounds. The conducted simulations produced four different scenarios of results combining simulation with or without aggregation, and simulation with solar or constant harvesting. To summarize the most crucial information, in this article, we will only address the results related to dynamic harvesting, as it is the closest to reality.

Therefore, since the sensors are within a distance of 10 wireless hops to facilitate the reading of the battery graphs, we only consider the lines of the nodes with the shortest and longest distances (1 and 10 hops, respectively), as well as the average energy line of all nodes and the harvesting information - see figure 1. All the complete graphics, related to the constant harvesting and the battery level information of all wireless links, can be consulted in the following link.

In Figure 1, the heuristic considered solar trace harvesting and the Figures 1a and 1b, represents the comportment with and without aggregation, respectively. In this cases, the nodes execute 1 sensing every 5 minutes and 1 transmission every 10 minutes. Results with sensing and transmission every 5 minutes can be consulted on the link described above. Figures 1a and 1b does not present the variation bar, since the margin of variation was not greater than 4%.

As depicted in Figure 1, when the batteries approach 10% (critical zone), the sensors reduce the number of transmissions they carry out in order to be able to recover energy using harvesting. This causes the sensors to start accumulating packets in their queue. At the end of the first period of finite-horizon - representing the end of the first day/night cycle - the queue has on average 70% payloads, which is much higher than they had at the beginning of the same cycle when the queue was empty. Therefore, at the moment when the second day/night cycle begins, represented by the beginning of the second harvesting cycle, the nodes will perform a number of aggregations and transmissions much higher than they had at the beginning of the first cycle. This leads to a decrease in the initial battery in the second cycle, which is different from what happened. This behavior can be easily minimized using aggregation because, even though the sensors at the beginning of the second cycle have more payloads to send, using aggregation will always carry the packages as compact as possible - the energy used is much less than when aggregation is not used. Even so, the behavior of the second cycle is identical to the first in the rest of the time.

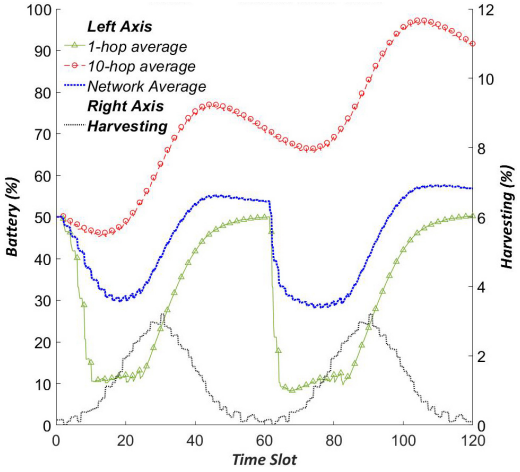
In all cases, the network maintains a neutral energy operation. Every node starts with 50% of battery and ends with more battery than the one with which it starts in all the considered cases then it is guaranteed that the network can operate perpetually recovering drop battery. Note that we could increase the amount of data that can reach the sink bypassing the ENO constraint 12, but it would prevent the ability to keep the network running continuously. In general, the nodes that have more battery, on average, are the 10-hops nodes since this nodes enjoy a null rate of reception in comparison with all other nodes. Despite critical battery situations in the inner nodes (1-9 hops) may arise, the 10-hops nodes will almost never be used to route information – all the information is route to reach sink node.

Sensors more distant are able to use other communication channels if any of the neighbors fail, however will be repercussions on all the utility of the network if any of the coren nodes cease to function, since all the information will have to be routed by the remaining 1-hop nodes. This situation will, in turn, make these nodes extraordinarily vulnerable and to likely fail too. Thus, in a real network setting, it is reasonable that the number of nodes with direct access to the sink should always be more significant to make the network more robust. The fact that the network has multi-hop capabilities is convenient in some cases where a node needs to find an alternative route to send the data.

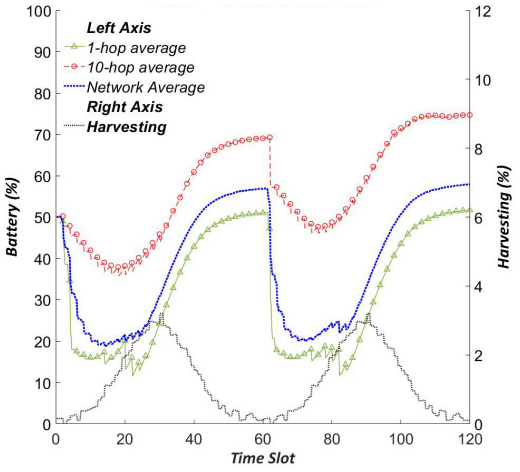
By operating the nodes in such controlled way, the network will make the most use of the aggregation energy savings by increasingly accumulating payloads in closer nodes to sink over the time period duration.

Since events are considered to be extremely unpredictable activities of extreme importance, the network is not only prepared to detect them - through sense operation - but also assigns them a priority rate at aggregation and transmission times. So, they are the first to arrive at the base station. Besides, sensors generally have limited storage capabilities, since all the information generated cannot be stored. Discarding or storing the data is decoded by the node. However, an event packet only can be discarded if the entire queue is occupied with packages of this type. This means that normal packages will always be discarded first.

The nodes decrease the number of transmission if the stored energy in the battery approaches 10%. Less frequent transmissions means that more data is stored in the queue and it causes data aggregation. The number of failed transmissions can can be observed in Figure 2. In the figure, we represent two types of simulation where Sx and Tx represent sensing and transmission timer used (Sx_{Delay} and Tx_{Delay}), respectively. It is possible to observe that data aggregation reduces the number of not performed transmission since data aggregation allows more data in each message. This reduction is in the order of 16% in case of Tx its equal to 5 minutes, and reduce 29% in 10 minutes transmission timer simulation. When choosing a value for Tx, a trade-off must be considered. Increasing Tx to a certain extent will bring further data communication efficiencies due to the higher rates of data aggregation. Moreover, the network utility will also increase. However, increasing substantially the value of Tx will not benefit data aggregation or successful transmission rates significantly, since the network has limited physical resources, i.e., the number of nodes,



(a) With Aggregation



(b) Without Aggregation

Fig. 1: HENO-IoT - Battery of Nodes Over Time.

payloads storage capabilities on packets and transmissions rates between nodes. This is why by increasing the Tx to twice the time of Sx, we reduced unsuccessful transmission rates using aggregation by 17%. However, this reduction is only 4% if we consider the case when we are without aggregation. Without aggregation, the sensor node's physical limitations are easily reached. Therefore, in this type of aggregation, we do not benefit from the increase in transmission time because we have more critical physical limitations that overlap.

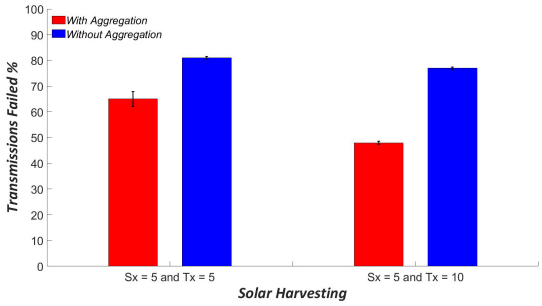


Fig. 2: Transmissions failed in different simulations.

As we can see in the Figure 3, the delivery of events type packages is 100% in all simulations in which aggregation is used, which demonstrates the importance of this factor for the functioning of the network. Aggregation provides a 52% better rating on this evaluation metric. To achieve a high rate of delivery of events, giving priority to the delivery rate of normal packages may drop slightly, which is not a problem since the events are much more critical, and the success in their communications overlaps the success of the remaining packages.

VI. Conclusions

Energy Neutral Operation is a challenge for IoT networks. Many aspects contributes for nodes running out of energy in the long-term. However, this is valuable challenge to be tackle,

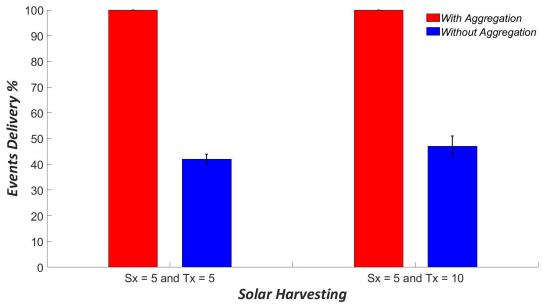


Fig. 3: Events Delivered on Sink.

since it represents a greener option of power IoT devices and reduces the maintenance costs.

This article introduces an Integer Linear Programming model and a distributed heuristics to achieve ENO in IoT networks. Both solutions considers periodic and event-based traffic, and uses data aggregation to regulate the network traffic, reducing the energy consumption.

The distributed heuristic has been evaluated in a simulation environment. The improvements of the proposed solution are presented in terms of residual battery, transmissions, and event-traffic effectively delivered to the sink.

Acknowledgements

The work presented in this paper was partially carried out in the scope of the MobiWise project: From mobile sensing to mobility advising (P2020 SAICTPAC/0011/2015), co-financed by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the project CISUC - UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020 and was partially supported by the Centre for Mathematics of the University of Coimbra – UID/MAT/00324/2019, funded by the Portuguese Government through FCT/MEC and co-funded by the European Regional Development Fund through the Partnership Agreement PT2020.

REFERENCES

[1] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "A first look at cellular machine-to-machine traffic: large scale measurement and characterization," *ACM SIGMETRICS performance evaluation review*, vol. 40, no. 1, pp. 65–76, 2012.

[2] M. Zareei, C. Vargas-Rosales, R. Villalpando-Hernandez, L. Azpilicueta, M. H. Anisi, and M. H. Rehmani, "The effects of an adaptive and distributed transmission power control on the performance of energy harvesting sensor networks," *Computer Networks*, vol. 137, pp. 69–82, 2018.

[3] S. Yang, Y. Tahir, P.-y. Chen, A. Marshall, and J. McCann, "Distributed optimization in energy harvesting sensor networks with dynamic in-network data processing," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[4] G. Jackson, Z. Qin, and J. A. McCann, "Long term sensing via battery health adaptation," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2240–2245.

[5] T. N. Le, A. Pegatoquet, O. Berder, and O. Sentieys, "Energy-efficient power manager and mac protocol for multi-hop wireless sensor networks powered by periodic energy harvesting sources," *IEEE Sensors Journal*, vol. 15, no. 12, pp. 7208–7220, 2015.

[6] A. Castagnetti, A. Pegatoquet, C. Belleudy, and M. Auguin, "A framework for modeling and simulating energy harvesting wsn nodes with efficient power management policies," *EURASIP Journal on Embedded Systems*, vol. 2012, no. 1, p. 8, 2012.

[7] Y. Zhu, Y. Liu, and L. M. Ni, "Optimizing event detection in low duty-cycled sensor networks," *Wireless Networks*, vol. 18, no. 3, pp. 241–255, 2012.

[8] R. Zhu, C. Peng, C. Jiming, D. Yau, and S. Youxian, "Dynamic activation policies for event capture in rechargeable sensor network," *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, vol. 25, no. 12, pp. 3124–3134, 2014.

[9] N. Correia, D. Sacramento, and G. Schütz, "Dynamic aggregation and scheduling in coap/observe-based wireless sensor networks," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 923–936, 2016.

[10] D. Sacramento, G. Schütz, and N. Correia, "Aggregation and scheduling in coap/observe based wireless sensor networks," in *2015 IEEE International Conference on Communications (ICC)*. IEEE, 2015, pp. 654–660.

[11] D. Gao, Y. Liu, F. Zhang, and J. Song, "Data aggregation routing for rechargeable wireless sensor networks in forest monitoring," *Wireless Personal Communications*, vol. 79, no. 1, pp. 773–788, 2014.

[12] S. Jeong, H. Kim, D. K. Noh, and I. Yoon, "Energy-aware data aggregation scheme for energy-harvesting wireless sensor networks," in *Computer Communication and the Internet (ICCCI), 2016 IEEE International Conference on*. IEEE, 2016, pp. 140–143.

[13] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer Congestion Control, Routing and Scheduling Design in Ad Hoc Wireless Networks," in *IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, 2006.

[14] M. Kodialam and T. Nandagopal, "Characterizing Achievable Rates in Multi-hop Wireless Networks: The Joint Routing and Scheduling Problem," in *MobiCom '03*. ACM, 2003, pp. 42–54.

[15] Y. Yi and S. Shakkottai, "Hop-by-Hop Congestion Control Over a Wireless Multi-Hop Network," *IEEE/ACM Transactions on Networking*, vol. 15, no. 1, pp. 133–144, 2007.

[16] S. He, J. Chen, D. K. Yau, H. Shao, and Y. Sun, "Energy-Efficient Capture of Stochastic Events under Periodic Network Coverage and Coordinated Sleep," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1090–1102, 2012.

[17] C.-f. H. C.-f. Hsin and M. L. M. Liu, "Network Coverage Using Low Duty-Cycled Sensors: Random & Coordinated Sleep Algorithms," in *Third International Symposium on Information Processing in Sensor Networks 2004 (IPSN 2004)*, 2004, pp. 433–442.

[18] Z. Ren, P. Cheng, J. Chen, D. K. Yau, and Y. Sun, "Dynamic Activation Policies for Event Capture in Rechargeable Sensor Network," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3124–3134, 2014.

[19] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *29th annual IEEE international conference on local computer networks*. IEEE, 2004, pp. 455–462.

[20] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*. IEEE, 2006, pp. 641–648.

[21] M. O. Farooq and T. Kunz, "Contiki-based IEEE 802.15.4 channel capacity estimation and suitability of its CSMA-CA MAC layer protocol for real-time multimedia applications," *Mobile Information Systems*, vol. 2015, 2015.

[22] A. Cavaco, H. Silva, P. Canhoto, S. Neves, J. Neto, and M. Collares Pereira, "Global Solar Radiation in Portugal and its variability, monthly and yearly," p. 32, 2016.

B

Appendix II

Installation Guidelines

This guideline provides necessary instructions for building a sensor platform in the IoT context, creating a Wireless Sensor Network based on Contiki OS.

The development of the entire model was carried out through the Cooja framework. For the execution of this guideline, it is necessary to install Linux distribution – Ubuntu.

With this document, a simulation platform bases on all the restriction mentioned during the dissertation are created.

The goal of the repository https://bitbucket.org/marcosantosilva/mobiwise/ is to create a functional example that replicates the findings in MobiWise work (Heuristic and Simulation of Energy Harvesting IoT Nertworks). For that, source code for the Sink Node and Sensor Node is developed, as well some changes to the underlying network modules are necessary.

The developed model allows the easy adjustment of some network parameters, which can be configured, considering the network's utility.

A.1 Warning Note

This user manual refers to a beta version of the entire application provided. This means that there may be future changes. Thus, some features can be added or removed until the final product.

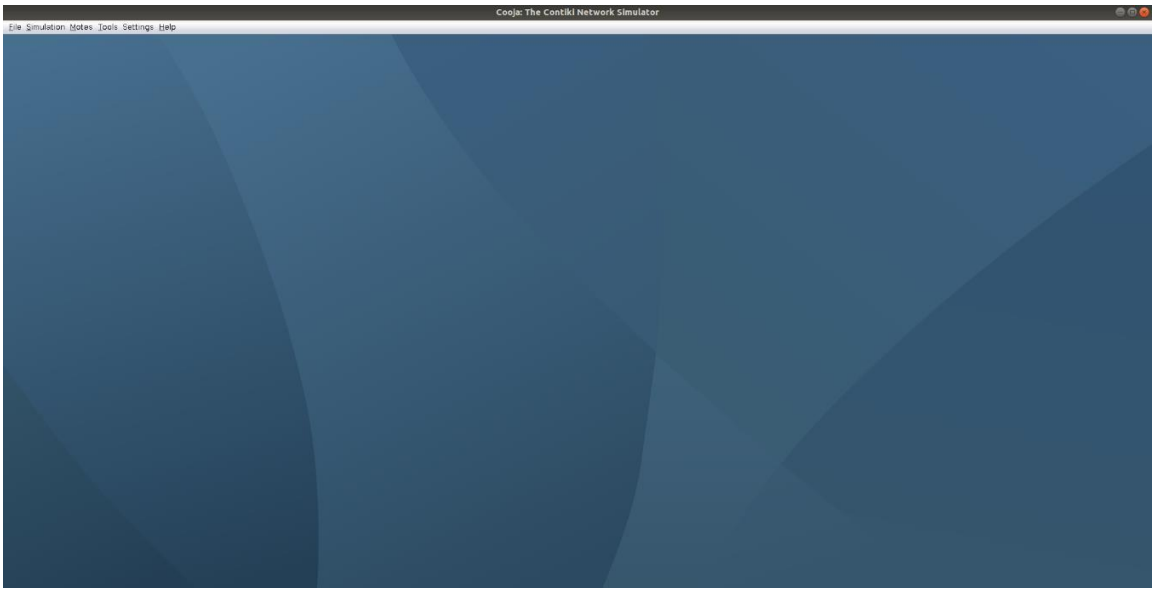
At the time of launching the final version of this application, this manual will be updated.

A.2 Getting Started

1. Contiki official page: http://www.contiki-os.org
2. Some tutorials and examples at http://anrg.usc.edu/contiki/index.php/Contiki_tutorials and https://senstools.gforge.inria.fr/doku.php?id=contiki%3Aexamples
3. Contiki timers: https://github.com/contiki-os/contiki/wiki/Timers
4. Instruction manual: https://www.researchgate.net/file.PostFileLoader.html?id=5912cfa3f7b67ef5d630d3bc&assetKey=AS%3A492384527097856%401494405027722
5. Important exercises: https://team.inria.fr/fun/files/2014/04/exercise_partII.pdf

A.3 Installing Contiki OS and Cooja Simulator

1. Go to https://bitbucket.org/marcosantosilva/mobiwise/ and download Contiki with all the files developed. Unzip and open the folder.
2. Build Cooja with extra memory: `cd contiki/tools/cooja && ant run_bigmem`. If exists some git error: `git submodule update - -init && ant run`

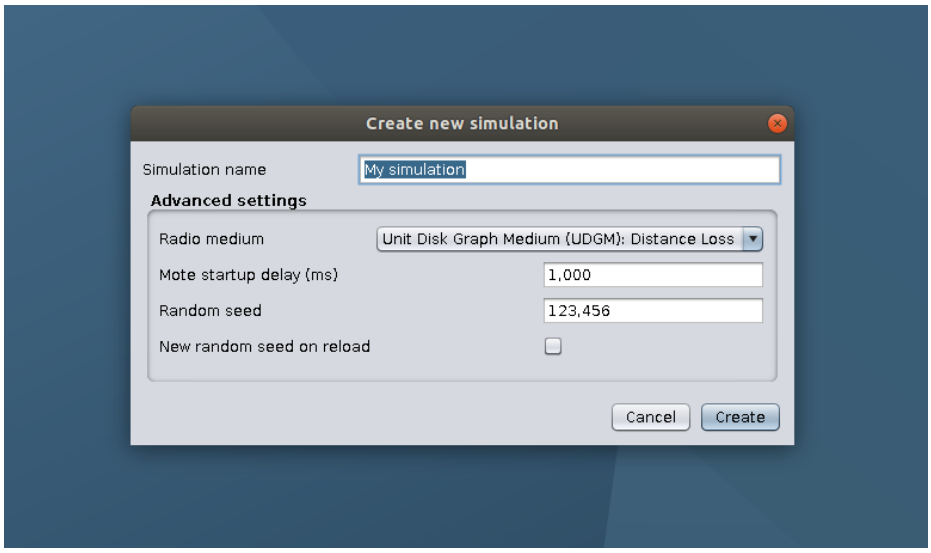


A.4 Create Simulation

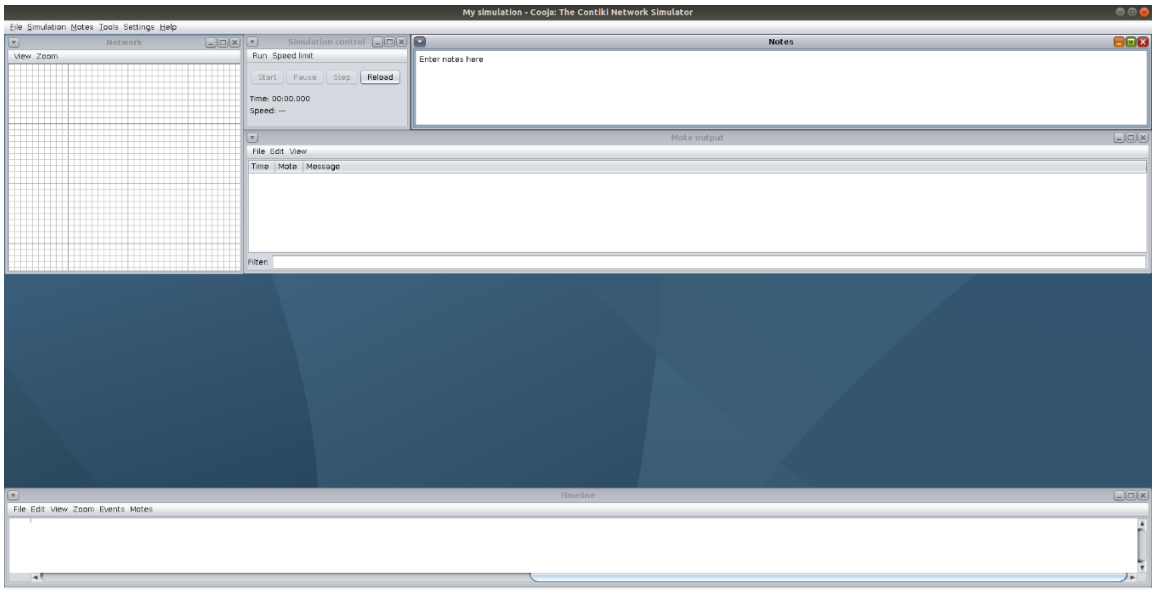
In the topology of our model, we use two types of sensors. The "sensor_nodes" responsible for capturing information from the environment and forwarding this information throughout the entire network. This information is destined for the "sink_node", where all the data is stored.

So, to compile the simulation used in this work, we will create a grid topology with 120 nodes of the type "sensor_nodes" and only one "sink_node". These values and the topology can be changed according to personal preferences.

1. Go to File -> Create new simulation.
2. Choose a Simulation Name; Radio Medium: Unit Disk Graph Medium (UDGM): Distance Loss; Mote startup delay (ms): 1.000; Random Seed: 123.456; New random seed on reload: No.



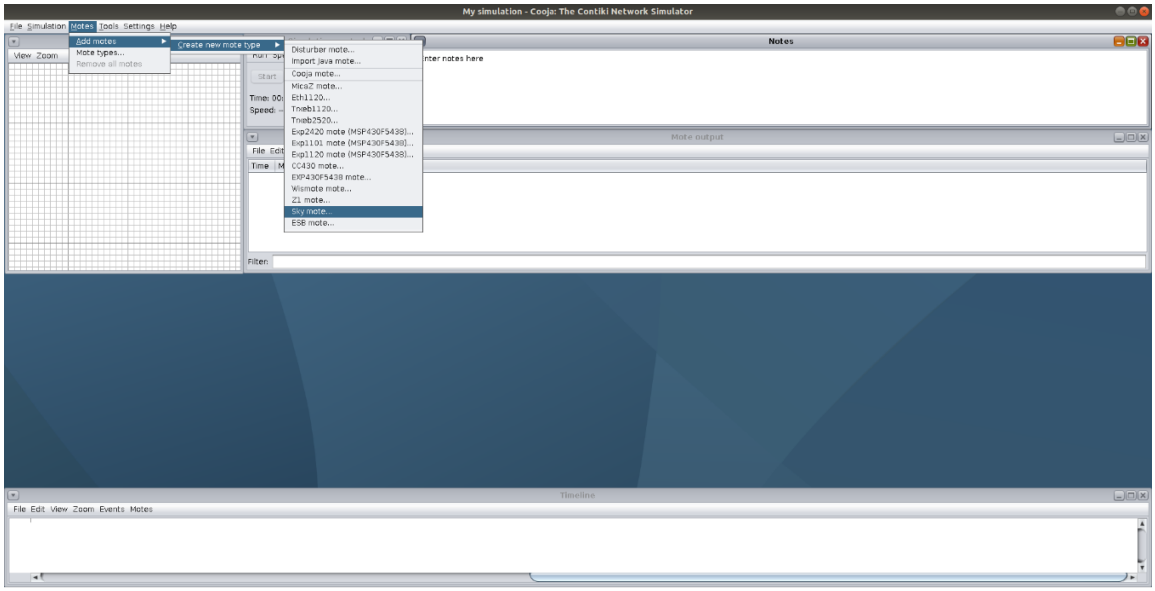
3. After everything is configured, several windows are opened: Simulation Window, Simulation Control Window, Notes, Mote Output and Timeline. Note that these windows can all be changed. These are just a few examples of the numerous plugins that Cooja has at its disposal.

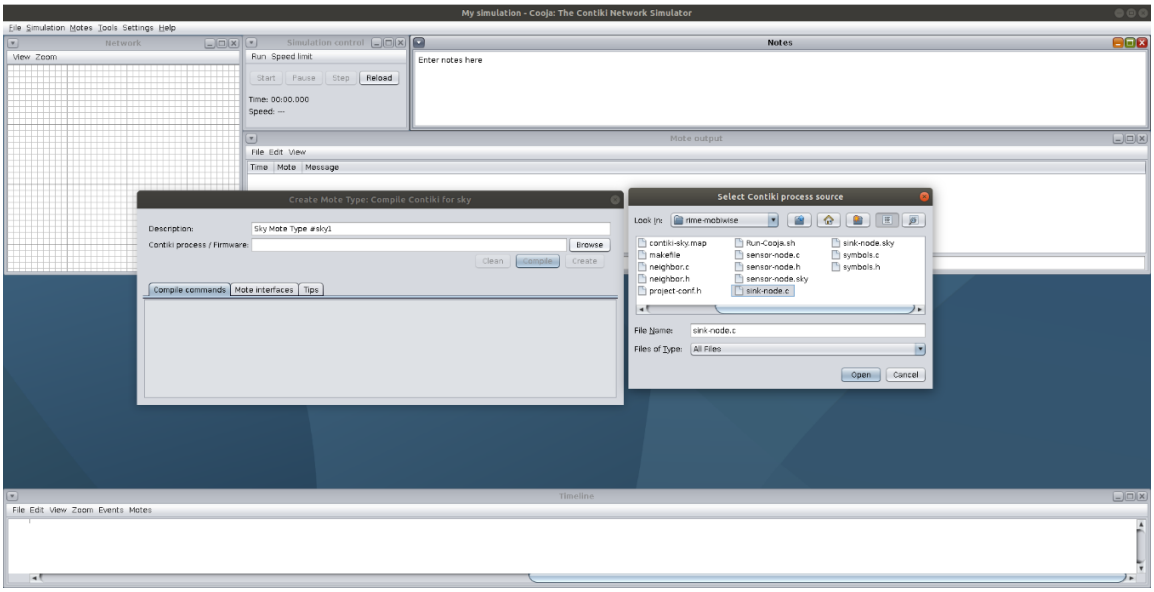


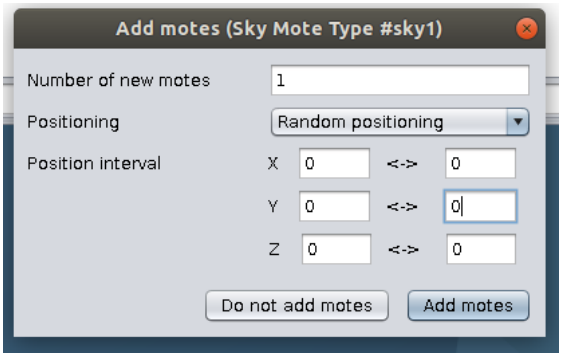
A.5 Start Simulation

A.5.1 Sink Node

1. Go to Mote -> Create New Mote Type -> Sky Mote
2. Click on Browse and open: Firmware -> contiki/examples/contiki-mobiwise/sink-node.c
3. Compile and create.
4. Select number of new motes to 1. Select Random positioning and 0 in all the axis range. Add motes.


My simulation - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Add motes
Mote types...
Remove all motes
Create new mote type
Disturbar mote...
Import Java mote...
Cooja mote...
MicaZ mote...
Eth1120...
Tmeb1120...
Tmeb2520...
Exp2420 mote (MSP430F5438)...
Exp1101 mote (MSP430F5438)...
Exp1120 mote (MSP430F5438)...
CC430 mote...
EXP430F5438 mote...
Wismote mote...
Z1 mote...
Sky mote...
ESB mote...
Notes
Enter notes here
Mote output
Timeline
File Edit View Zoom Events Motes


My simulation - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Network
Simulation control
Start Pause Step Reload
Time: 00:00.000
Speed: ---
Notes
Enter notes here
Mote output
File Edit View
Time Mote Message
Create Mote Type: Compile Contiki for sky
Description: Sky Mote Type #sky1
Contiki process / Firmware:
Browse
Clean Compile Create
Compile commands Mote interfaces Tips
Select Contiki process source
Look in: rime-mobiwise
contiki-sky.map
makefile
neighbor.c
neighbor.h
project-conf.h
Run-Cooja.sh
sensor-node.c
sensor-node.h
sensor-node.sky
sink-node.c
sink-node.sky
symbols.c
symbols.h
File Name: sink-node.c
Files of Type: All Files
Open Cancel
Timeline
File Edit View Zoom Events Motes

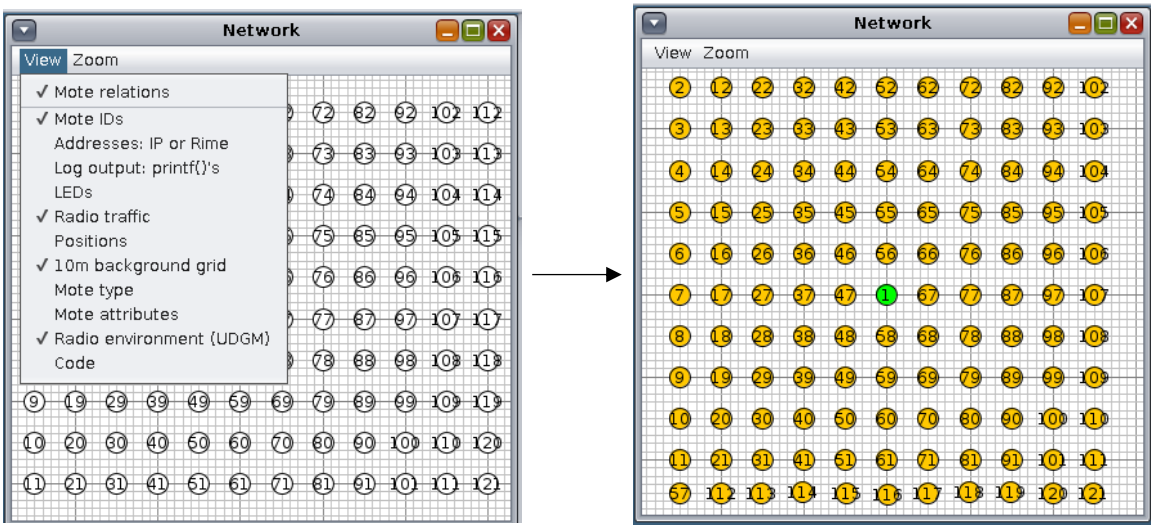

Add motes (Sky Mote Type #sky1)
Number of new motes
1
Positioning
Random positioning
Position interval
X
0
<->
0
Y
0
<->
0
Z
0
<->
0
Do not add motes
Add motes

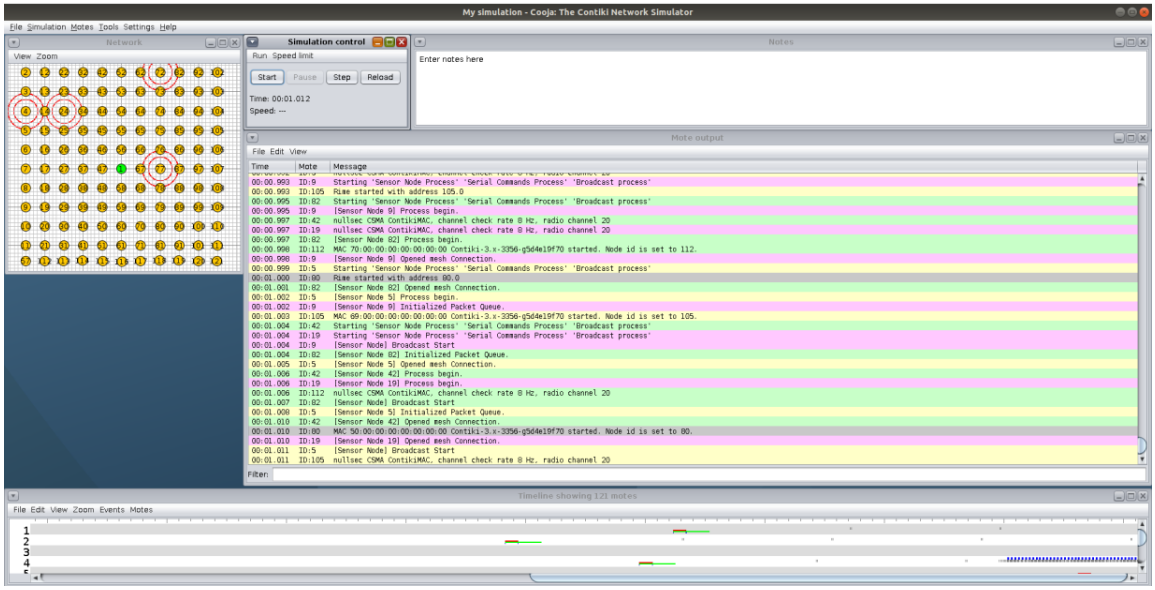
A.5.2 Sensor Node

1. Go to Mote -> Create New Mote Type -> Sky Mote
2. Click on Browse and open: Firmware -> Contiki/examples/Contiki-mobiwise/sensor-node.c
3. Compile and create.
4. Select number of new motes to 120. Select Linear positioning. X and y must be between -250 and 250. Z selects as default.

A.6 Configure Network

1. In the Network window: View -> Mote IDs; View -> Mote type; View-> Radio traffic; Sensor nodes and sink node are differentiated with a different color. If there is an overlapping sensor, it can be repositioned with the mouse cursor, until reaching the following topology.
2. To configure motes communication ranges: right button click on mote -> change transmission ranges.
3. Go to Simulation Control -> Start. All sensor messages are shown in the Mote Output.





A.7 Source code Overview

All files present in folder examples/rime-mobiwise were created from scratch to implement all models. The sensor-node.c file and the sink-node.c contains all the control primitives for the sensor functions, applying the defined heuristics. The neighbor.c file has the broadcast code responsible for creating the neighbors' list for each node with all the necessary information.

Find Rime code at contiki/core/net/rime, some underlying modules at contiki/core/net.

Network protocols can be explored at contiki/os/net, contiki/core/net and contiki/dev.

Energy module, that controls each sensor's battery discharge and harvesting, can be find at contiki/apps/powertrace.

Throughout the work, and as explained in this dissertation's main body, changes were made to the existing network protocols. In the file "examples / rime-mobiwize / project-conf.h, you can see all the control variables related to the settings of the network simulation.

```
ifndef __PROJECT_CONF_H__
```

```
#define __PROJECT_CONF_H__
// Automatic Operations
#define AUTO 1
// Define if nodes do aggregation
#define AGGREGATION 0
// Operation Timeslot -> sense time
#define SENSE_DELAY_MAX 60 //seconds
// Limit delay to transmit
#define TRANSMIT_DELAY_MAX 120 //seconds
// Simulation time
#define SIMU_TIME 3600 // seconds
// Period to Refresh battery level
#define POWER_PERIOD 12 // Seconds
//seconds to stabilize nodes connections when the network starts
#define NETWORK_STABILIZE 60
//initial Seconds to repeat the broadcast message
#define ROUTE_REQUEST 5
#define PAYLOAD_MAX_LEN 102
#define MAX_QUEUEBUF 10
#endif /* __PROJECT_CONF_H__ */
```

File contiki-conf.h, represent all the configurations of Contiki and Network Protocols provided. This file only represents the control variables. Given the complexity of files used and each one's length, we chose not to include it. These files can be consulted from the Contiki config, accessing the file where each variable is used.

This page intentionally left blank.