People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
Mohamed Khider University - Biskra

Order Number: ....
Series: ....

Faculty of Exact Sciences and Sciences of Nature and Life
Computer Science Department

THESIS

In Candidacy for the Degree of

DOCTOR 3^{rd} CYCLE IN COMPUTER SCIENCE

Option: Artificial Intelligence

By

SIHAM ZROUG

TITLE

Modelling, verification and performance evaluation of the CSMA/CA protocol in WSNs, by Coloured Petri Nets

Defended on: 15/07/2021

In front of the jury composed of:

Mr. Hammadi Bennoui	Professor at the University of Biskra	**President**
Mr. Laid Kahloul	Professor at the University of Biskra	**Supervisor**
Mr. Karim Djouani	Professor at the University of Creteil	**Co-supervisor**
Mr. Soheyb Ayad	Associate Professor at the University of Biskra	**Examiner**
Mr. Rachid Seghir	Professor at the University of Batna 2	**Examiner**
Mr. Saber Benharzallah	Professor at the University of Batna 2	**Guest**

Academic year: 2020/2021

Abstract

This thesis tackles the problem of formal modelling and verification of Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol in wireless sensor networks. Indeed, our research focuses on the modelling, the verification and performance evaluation of CSMA/CA using Coloured Petri Nets (CPNs). Medium Access Control (MAC) protocol plays a vital role in performance of wireless sensor networks due to its crucial purpose to emulate successful communication. The CSMA/CA protocol was one of the predominant protocols in WSNs for its contention mechanism and channel access.

This thesis provides three basic contributions. In the first contribution, Hierarchical Timed Coloured Petri Nets (HTCPNs) formalism is used to model the CSMA/CA protocol, and then the CPN-Tools is exploited to analyse the obtained models. Indeed, the timed aspect in HTCPN allows us to deal with temporal constraints in CSMA/CA. The hierarchical aspect of HTCPN makes the CSMA/CA model "manageable", despite the complexity of the protocol.

The limited energy associated with WSNs is a major drawback of WSN technologies. To overcome this major limitation, the design and development of efficient and high performance energy harvesting (EH) systems for WSNs environments are being explored. Hence, as a second contribution, a hierarchical timed coloured Petri net model for CSMA/CA in EH-WSNs is proposed. In order to evaluate the proposed model, a set of qualitative and quantitative properties are verified. Linear Temporal Logic (LTL) is used to formalise and verify qualitative properties. The quantitative verification is done by using extracted files from monitors results defined in CPN-Tools.

As a third contribution, a Machine Learning (ML) is used to discuss the scalability of the first proposed model when the number of sensor nodes is increased. An effective neuronal network is exploited for the prediction of the throughput metric of the network based on the number of nodes to prove the scalability of the proposed formal approach. Indeed, a set of performance metrics have been predicted to show the evolution of the network when the number of nodes increases.

Key words: *Medium Access Control, Wireless Sensor Networks, CSMA/CA, EHWSNs, Formal Modelling, Formal Analysing, Performance Evaluation, Coloured Petri nets, Linear Temporal Logic. Machine Learning.*

Résumé

Cette thèse aborde le problème de la modélisation/vérification formelles du protocole CSMA/CA (Sense Multiple Access with Collision Avoidance) dans les réseaux de capteurs sans fil (WSNs). En effet, nos recherches concentrent sur la modélisation, la vérification et l'évaluation des performances de protocol CSMA/CA à l'aide de Réseaux de Petri colorés (RdPCs). Le protocole MAC (Medium Access Control) joue un rôle essentiel dans les performances des réseaux de capteurs sans fil en raison de son objectif crucial d'émuler une communication réussie. Le protocole CSMA/CA était l'un des protocoles prédominants dans les WSNs pour son mécanisme de contention et son accès au cannal.

Cette thèse fournit trois contributions de base. Dans la première contribution, le formalisme HTCPNs (Réseaux de Petri Colorés Temporisés et Hiérarchiques) est utilisé pour modéliser le protocole CSMA/CA, puis l'outil CPN-Tools est exploité pour analyser les modèles obtenus. En effet, l'aspect temporisé en HTCPN permet de prendre en considération les contraintes temporelles du protocole CSMA/CA. L'aspect hiérarchique du HTCPN rend le modèle proposé gérable, malgré la complexité du protocole.

L'énergie limitée inhérente aux WSN est un inconvénient majeur des technologies WSN. Pour surmonter cette limitation majeure, la conception et le développement de systèmes de récupération d'énergie (EH) efficaces et à haute performance pour les environnements WSN sont explorés. Par conséquent, comme deuxième contribution, un modèle de réseau de Petri coloré hiérarchique pour le protocole CSMA/CA dans les EH-WSNs est proposé. Afin d'évaluer le modèle proposé, un ensemble des propriétés qualitatives et quantitatives est vérifié. La logique temporelle linéaire (LTL) est utilisée pour formaliser et vérifier les propriétés qualitatives. La vérification quantitative est effectuée en utilisant les fichiers extraits des résultats des moniteurs définis dans l'outil CPN-Tools. La troisième contribution consiste à utiliser la Machine Learning pour renforcer la scalabilité du premier modèle proposé lorsque le nombre de nœuds capteur est augmenté. Un réseau de neurone efficace est exploité pour la prédiction de la métrique de débit du réseau en fonction du nombre de nœuds. En plus, un ensemble de metriques de performance ont été prédits pour voir l'évolution du réseau quand le nombre de nœuds est augmenté.

Mots clés: *Contrôle d'accès au support, Réseaux de capteurs sans fil, CSMA/CA, Modélisation/Vérification Formelle, Evaluation de performance, Réseaux de Petri Colorés, Apprentissage automatique.*

Acknowledgement

First and foremost, heartfelt gratitude and praises go to the ***Almighty Allah*** *who guided me through and through.*

I would like to extend my sincere thanks and appreciations to my honourable supervisor ***Prof. Laid KAHLOUL****. He has painstakingly corrected and recorrected draft after draft. Moreover, This work could not have reached fruition without the unflagging assistance and participation of my supervisor. I would never thank him enough for the huge contribution that made this work what it is now. Once again, I would like to express my deep gratitude to my supervisor because I owe thanks to him.*

Many thanks go to ***Prof. Karim Djouani*** *for his advice.*
I also thank those who agreed to be the referees of this thesis and allocated their valuable time in order to evaluate the quality of this work: ***Prof. Hammadi BENNOUI****,* ***Dr. Soheyb Ayad****,* ***Prof. Rachid Seghir****, and* ***Prof. Saber Benharzallah*** *for their examination of the report and their valuable comments.*
Most warm gratitude goes to ***Prof. Okba KAZAR****, head of the Intelligent Computer Science Laboratory.*
Last but not least, I wish to thank ***Prof. Abdelhamid Djeffal****, head of the Computer Science department.*

Contents

List of Figures

List of Tables

List of Algorithms

List of abbreviations

ACK	Acknowledgement
ADC	Analog to Digital Converter
AI	Artificial Intelligence
BC	Backoff Counter
BS	Base Station
CAP	Contention Access Period
CPN	Coloured Petri Net
CSMA/CA	Carrier Sense Multiple Access Collision Avoidance
CW	Contention Window
DCF	Distributed Coordination Function
DIFS	Distributed Interframe Space
DL	Deep Learning
DP	Delay Performance
EH-WSNs	Energy Harvesting Wireless Sensor Networks
EIFS	extended interframe space
HTCPN	Hierarchical Temporal Coloured Petri Net
LTL	Linear Temporal Logic
MAC	Medium Access Control
MLP	Multi Layer Perception
ML	Machine Learning
NN	Neuronal Network
OOPN	Object Oriented Petri Net
PN	Petri Net
PTA	Priced Timed Automata
QoS	Quality of Service
S-MAC	Sensor Medium Access Control
SIFS	Short Interframe Space
SPNs	Stochastic Petri Nets
TA	Timed Automata
WLANs	wireless local area networks
WSNs	Wireless Sensor Networks
WT	Waiting Time

INTRODUCTION

Context and aims

A Wireless Sensor Network (WSN) [1] is a distributed system of cooperating devices that perform distributed monitoring applications in a physical environment over a self-organised wireless network topology. WSNs market has grown significantly over the two last decades. This growth is due to the industry's increasing demand for WSNs technology. Indeed, a large number of researches confirm that the growth of WSNs allows a new way to perform unusual tasks [2]. WSNs are a precious technology to encourage the evolution of innovative applications in many domains, such as environmental and habitat monitoring, surveillance, indoor climate control, structural monitoring, medical diagnosis, disaster management, and so on [3], [4].

Moreover, WSNs are multi-hop self-organizing networks composed of sensor nodes that communicate wirelessly and mostly supply vital support for collecting, processing, and forwarding the real-time information in mission-critical applications. Although these networks confront several challenges owing to the fact that the sensor nodes are characterised by limited energy, limited memory, and weak computing power [1]. The role of wireless sensor nodes is to detect and gather information from a sensor field or area of interest, computes the information and forwards it through a radio module back to a central point or destination [5, 6, 7, 8, 9, 10].

In WSNs, the communication between sensors and base station is an essential axis in the success of the WSN. This communication requires both the appropriate hardware as well as the appropriate software layer. The software layer is generally concerned with a set of protocols and algorithms that are developed to allow nodes to share the communication channel, route messages, and collaborate to perform a certain task.

The Medium Access Control (MAC) layer is responsible for regulating channel access. MAC protocols have been developed to assist each node in deciding when and how to access the channel. Carrier Sense Multiple Access with Collision Avoidance [11] (CSMA/CA) is a widespread MAC protocol used in modern wireless communication systems to allow multiple users to share the same channel. Common examples of wireless systems using CSMA/CA are Wi-Fi networks (IEEE 802.11 standard) [12] and IEEE 802.15.4 [13] wireless sensor networks. A CSMA/CA protocol must sense the channel before transmitting. If the medium is not busy, the transmission begins. Else, after a random back-off period, another attempt is performed. Therefore, the MAC protocol of CSMA/CA for carrier transmission in WSNs avoids collisions by transmitting only when the channel is idle. In the following paragraphs, we describe the problematic and the contributions of this thesis through three challenges that we tackle by providing three solutions which are basically the three contributions of this thesis.

First Challenge

Hence, CSMA/CA is the most common medium access control method. The primary task of any MAC protocol is to control the access of the nodes to a shared medium. A MAC protocol is a set of established rules that dictate how to format, transmit and receive data so computer network devices can communicate regardless of the differences in their underlying infrastructures, designs, or standards. A network using such protocol must satisfy this set of rules in all its possible scenarios. When the CSMA/CA method is used in wireless systems, the set of constraints are specified. In a formal specification of the CSMA/CA protocol, one must specify all the constraints that the system must satisfy to guarantee the access to the channel. Then the formal verification consists in proving that these constraints are satisfied by the behaviour of the network in all its scenarios. Thus, formal modelling/analysis approach of such MAC protocol is needed to prove its consistency within a network. For that reason, designing effective medium access control (MAC) protocols is one of WSNs crucial challenges in aiming to determine the channel access control capabilities and the energy consumption properties of WSNs.

Formal verification requires a system (a protocol in this case) and a constraint or property to be checked. Often the formal verification works on a model of the system (abstract representation). This is obligatory due to the system's complexity; thus, such property must be formalised to be checked over time using a specification language. The response of a qualitative property to a verification request is either YES (property checked) or NO (property not verified). However, in a real study, we are interested instead in quantitative properties, for example, the number of arrived packets in the network, the time it takes for a packet to go from a transmitter to a receiver, the time that a node must wait before having to access to the medium for sending its packet, the throughput of the network at time T, etc.

The proposed solution to deal with the first challenge

The implementation of WSNs protocols must respect the standards and satisfy the expected good properties. Formal methods can check this last constraint. These methods provide techniques to certify if a system satisfies a given property by its modelling and specifying its properties. Actually, formal methods are based on mathematics and they are used to develop computer tools to facilitate the verification process. As a formal method, Petri Nets [14] are well suitable to describe discrete processes and to analyse the system concurrency and synchronism. A Petri Net has a graphical representation and a rigorous semantics. These characteristics make Petri nets a good formalism to specify and to analyse MAC protocols in WSNs. Colored Petri Nets (CPN) [15] are an extension of Petri Nets which is more suitable to describe more complex and typed data. Then, Timed CPN extends CPN with a set of time stamps. Hierarchical CPN and Hierarchical TCPN are extensions for CPN and TCPN, where the structure of the model can be organized hierarchically.

As a consequence, the use of formal methods allows us to prove that the designed protocol is consistent. For that reason, we present a formal modelling /analysis approach of CSMA/CA protocol in WSNs which is our first contribution that will be detailed in chapter two. This approach uses Hierarchical Timed Coloured Petri Nets (HTCPN) formalism to model CSMA/CA protocol, and the CPN-Tools [16] to analyse the generated models. The timed aspect, in HTCPN, facilitates the consideration of temporal constraints introduced in this protocol. The hierarchical aspect of HTCPN makes the model "manageable", despite the complexity of CSMA/CA protocol specification. In the HTCPN model, we define the events that can occur in the system and their preconditions and post-conditions. These preconditions and post-conditions specify CSMA/CA protocol constraints that should be satisfied. After the specification by using the Hierarchical Timed CPN formalism, the CPN-Tools will be used to ensure an analysis of the proposed model and to verify some properties.

Second Challenge

Although, the limited energy associated with WSNs is a major challenge of WSN technologies. To overcome this major limitation, the design and development of efficient and high performance energy harvesting [17] systems for WSN environments are being explored. The MAC protocols for energy-harvesting WSNs [18] may be different from conventional battery-powered WSNs and need some different design criteria, such as environmental adaptability, backlog estimation, and frame length selection. Energy harvesting provides new opportunities and challenges for the design of the protocols and algorithms to support environmental monitoring. In particular, instead of focusing on reducing the energy consumed by nodes to prolong the network lifetime, as it is the critical design metric in traditional WSNs, in EH-WSNs, it is essential to re-consider the impact of performance metrics such as energy efficiency, scalability, and latency in the presence of energy flow into the network. Similarly, MAC protocols need to be re-designed to optimize the rate at which the energy is used, rather than simply minimizing the consumed total energy. The MAC protocol of CSMA/CA may improve access control and serve to reduce collisions in carrier transmission of energy harvesting WSNs.

The proposed solution to deal with the second challenge

The modelling and the verification of a communication protocol needs a formal tool that provides: (i) modelling of complex and typed data; (ii) modelling of discrete process; (iii) modelling time constraints; (iv) modelling characteristics of an event driven system; (v) analysing of synchronisation; (vi) analysing of system concurrency. Indeed, the graphical representation and the rigorous semantics of HTCPNs makes it one of the efficient formal tools to model the complex and typed data. Further, HTCPNs are able to capture many characteristics of an event driven system, namely concurrency, asynchronous operations, deadlocks, conflicts, etc. Moreover, HTCPNs are well suitable to describe discrete process and to analyse synchronisation and system concurrency. Furthermore, the hierarchical aspect makes the model more manageable and more comprehensible. In addition to these features, HTCPNs allow researchers to model the time aspect.

To more accurately verify the CSMA/CA protocol in energy-harvesting WSNs, in this thesis as a second contribution, a new formal approach of CSMA/CA is proposed in chapter three using Hierarchical Timed Coloured Petri Nets. The analyse procedure of the proposed HTCPN model focuses on qualitative and quantitative verification, which are realised using CPN-Tools.

Third challenge

Performance evaluation of WSNs MAC protocols is an important process that helps designers to obtain numerical results from their proposed models and to prove the correctness of the designed protocols. Scalability is an important issue which concerns the study of the WSN performance when the number of nodes dramatically increases. When scalability factor is considered in wireless sensor networks design, the protocol should work effectively in wide WSNs. When using a formal model like HTCPNs, the major challenge is that the formal model size increases with the increase of the number of nodes. The formal analysis of HTCPN models can be very hard in terms of time and effort when the number of nodes in the WSNs is very high. This number of nodes, when considered in the modelling process, makes the model very huge and complicated and the analysis process will be no more feasible.

The proposed solution to deal with the third challenge

For that, researchers adopt Machine Learning (ML) [19] techniques as an effective alternative solution for the performance evaluation estimation of designed protocols and algorithms in WSNs. In particular, computer systems can use machine learning algorithms and techniques to learn from prior data, allowing these systems to classify and predict new results [20, 21]. Recently, Machine learning (ML) attracts considerable attention due to its ability to predict a wide variety of complex phenomena accurately [22]. Applying ML in WSNs lies in exploiting historical data to improve the performance of sensor networks on given tasks without reprogramming.

As a third contribution in this thesis, an effective artificial neuronal network [23] is investigated to predict the performance evaluation of the proposed HTCPN formal approach presented in chapter two. The scalability of the proposed approach is discussed by estimating the evolution of throughput metric over the number of nodes increase in the network. Indeed, a set of temporal performance metrics including, *the average waiting time before transmitting, the average delay performance and the average waiting time* *for an acknowledgement*, have been predicted when the number of nodes increases in the network.

Manuscript organization

The rest of the thesis is organized as follows:

- **Chapter II** introduces the basic concepts of wireless sensor networks and MAC protocols. In this chapter, we present the informal/formal definition and some necessary concepts of Hierarchical Timed Coloured Petri Nets and Linear Temporal Logic. Moreover, we discuss the choice of HTCPN as a formal method to prove CSMA/CA protocol consistency. After that, Machine Learning definition and techniques are detailed.
- **Chapter III** introduces the first contribution, which consists of exploiting HTCPN to model and analyse the CSMA/CA method in WSNs.
- **Chapter IV** details the second contribution which proposes the performance evaluation of new formal HTCPN approach of CSMA/CA protocol in EH-WSNs.
- **Chapter V** presents the use of Machine Learning to prove the proposed model scalability presented in the third chapter and illustrates the predicted results concerning a set of critical performance metrics.
- Finally, in **Chapter VI**, the thesis is concluded with a summary of contributions and open perspectives.

Scientific Production

As a result of this thesis, the following publication was produced:

- ZROUG, Siham, KAHLOUL, Laid, BENHARZALLAH, Saber, et al. A hierarchical formal method for performance evaluation of WSNs protocol. Computing, 2021, vol. 103, no 6, p. 1183-1208.

And the following accepted conference paper:

- Siham Zroug, Ikram Remadna, Laid Kahloul, Sadek labib Terrissa, and Saber Benharzallah. Leveraging the power of machine learning for performance evaluation prediction in wireless sensor networks. In *The 10th International Conference on Information Technology (ICIT 2021)*. IEEE, 2021.

Chapter I

WSNs MAC Protocols, Formal Methods and Machine Learning

I.1 Introduction

Wireless Sensor Networks (WSNs) are used in many different domains and applications. They have gained significant popularity due to their flexibility in solving problems in different application areas and enhancing our lives in several manners. WSNs face many challenges due to the fact that the sensor nodes are characterized by limited energy, limited memory, and weak computing power. Therefore, designing an effective Medium Access Control (MAC) protocol is a challenging task with the aim to determine the channel access control capabilities and the energy consumption properties of WSNs. The use of formal methods has proved their efficiency to ensure such analysis in MAC protocols. As a formal method, Petri Nets are well suitable to prove the consistency and the correctness of MAC protocols. The scalability in WSNs is a crucial factor in proving the effectiveness of protocols. When the number of sensors increases in WSNs, the formal methods become unable to analyse large models with a huge number of nodes in the model. Machine learning can be exploited to predict the performance evaluation of such a protocol.

Hence, what is a sensor node? What is a wireless sensor network? What is a medium access control protocol? Why are Petri Nets well suited to describe and analyse a medium access control protocol in WSNs? How can machine learning be used to predict the scalability of the formal method proposed models? This chapter answers all of these questions in Sections I.2, I.3, and I.4, respectively. Finally, Section I.5 gives a brief conclusion from what this chapter presents.

I.2 Wireless Sensor Network Protocols

In this section, the background concerning wireless sensor network protocols is presented. Starting with the definition of sensor node and wireless sensor network introduced by researchers, after that, components of WSNs are presented.

I.2.1 (Wireless) Sensor Node

A sensor device, also known as a node or mote, performs operations such as sensing, detecting, or responding to analogical inputs of its physical environment [24]. These sensed inputs could be the light, the temperature, the humidity, the motion, the sound, the pressure, the vibration, or any other environmental phenomena. Therefore, such a device embeds an Analog-to-Digital Converter (ADC) that digitizes the continual analog signal produced by the sensors. Other main components include an external memory and a power source, usually a battery.

A sensor node is made wireless once a micro-controller (e.g., data processing and protocol signalling) and a wireless radio transceiver are added. Figure I.1 represents a typical sensor node that can be seen as a wireless device with a tiny size, and it provides a lightweight and portable detection station that can gather data and communicate with other wireless nodes in order to transmit its readings.

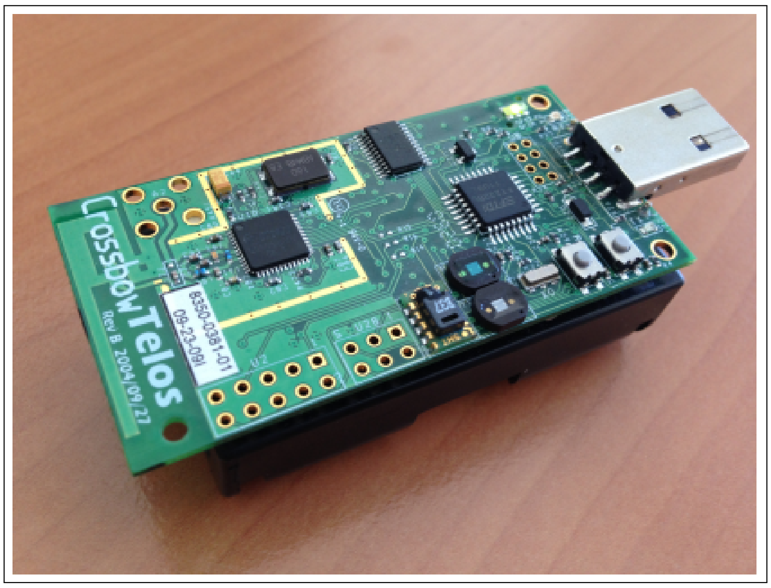


Figure I.1: A typical sensor node

A sensor node comprises four essential components as shown in I.2 a power unit, a transceiver unit, a processing unit, and a sensing unit. They may also have application-dependent additional components such as a location finding system, a power generator, and a mobilizer. Sensing units usually include two subunits: sensors and analog to digital converters (ADCs). The analog signals generated by the sensors based on the observed phenomenon are converted to digital signals by the ADC.
One of the more critical components of a sensor node is the power unit. Power units may be supported by a power scavenging unit such as solar cells. The processing unit manages the procedures that make the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. This unit is generally associated with a small storage unit. The objective of a transceiver unit is to connect the node to the network. There are also other subunits, which are application-dependent.

I.2.2 Wireless Sensor Networks Definition

In the literature, several definitions of wireless sensor network exist [1, 6, 25]. The definition of *Sohraby et al.* stated in [6] is adopted for its generality.

A sensor network is an infrastructure comprised of sensing (measuring), computing, and communication elements that gives an administrator the ability to instrument, ob-

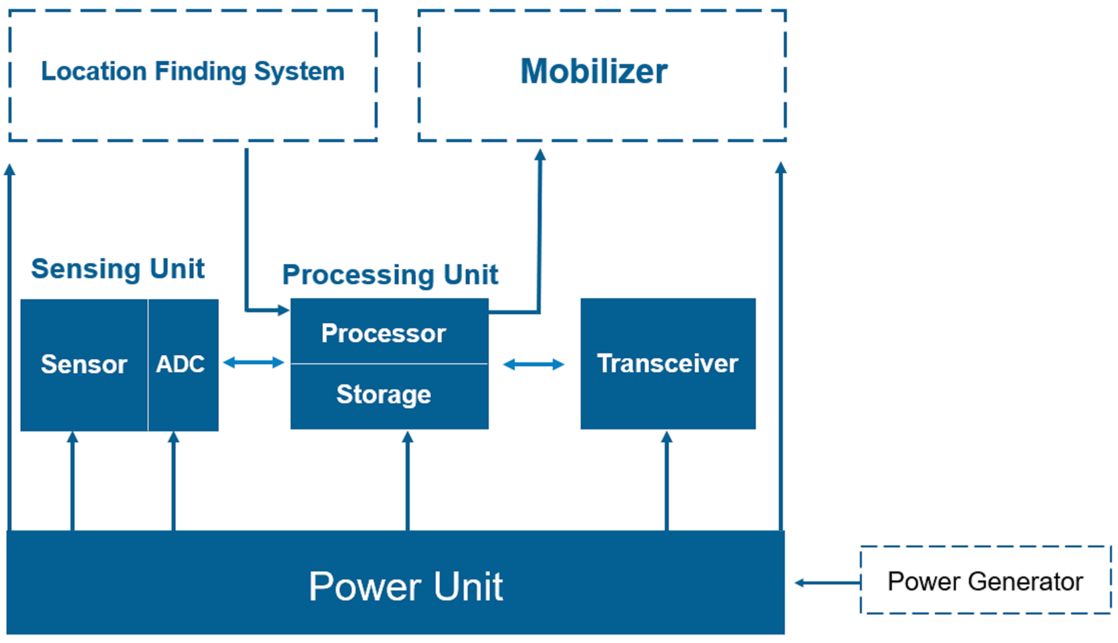


Figure I.2: The components of a sensor node

serve, and react to events and phenomena in a specified environment [6].

A wireless sensor network consists of four components:

- A set of nodes containing sensors.
- A wireless interconnection network.
- A base station for gathering information and acting as an interface between nodes and the user.
- Base station-level processing resources to handle data correlation, data mining, and status queries.

The base station has unlimited energy and significant computing power. It connects the nodes, through the Internet or by satellite, to the user's computer (see Figure I.3). We are only interested in the base station, the nodes, and the interconnection network (dotted box in Figure I.3) because it is the most complex part (communication between nodes) and the most constrained (limited resources of nodes, heterogeneity, etc.). The four components of a wireless sensor network allow an administrator to instrument it to capture, process, or communicate data.

Each application has specific requirements. The specificity of wireless sensor networks is that they are dedicated to the application (application-specific). However, we have isolated the core requirements common to all of these applications.

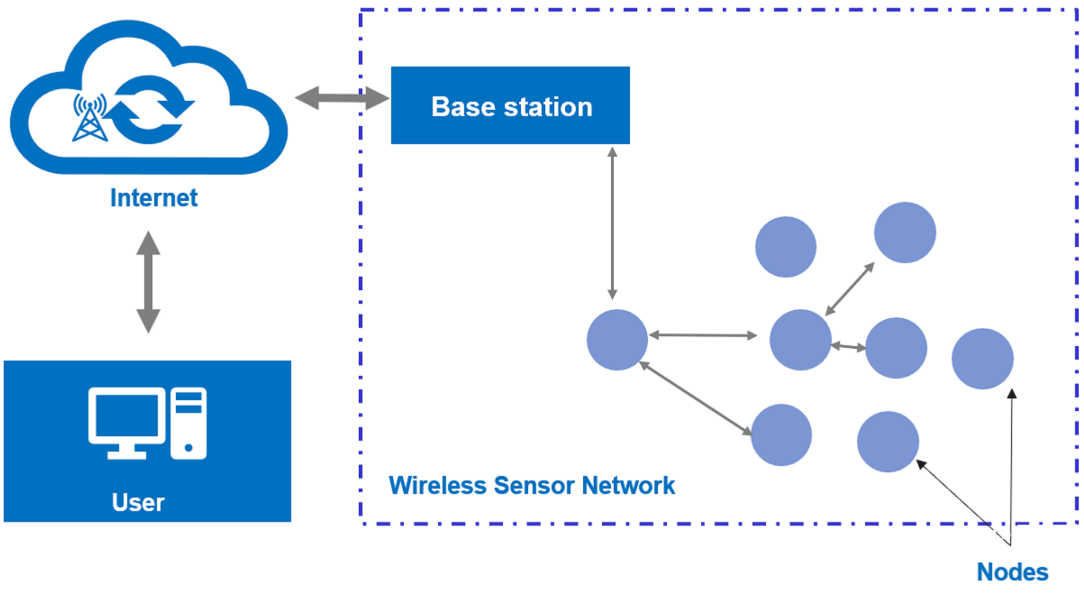


Figure I.3: Wireless Sensor Network Architecture

I.2.3 Main requirements of Wireless Sensor Networks

The two basic requirements of wireless sensor networks are:

- **Node cost:** A low cost is required to allow the deployment of high-density networks.
- **Operating autonomy:** The network must operate for the required duration, despite node failures, and without the possibility of re-supplying energy. Indeed, generally the nodes are placed in hostile environments (e.g., near a volcano [26]) where humans cannot access.

From the cost and operating autonomy requirements, the following requirements can be deduced:

- **Low energy consumption:** Nodes contain a limited amount of energy, which generally cannot be renewed. Therefore, the specificity of wireless sensor networks compared to other types of networks lies in the priority of the energy constraint over the quality of service (QoS) constraints [1]. Quality of service being the ability to provide different priorities to different applications, users, or data streams, or to guarantee a certain level of performance of a data stream [27] (e.g. rate of packet loss between source and recipient, transmission rate).
- **Fault-tolerance:** Fault tolerance is the ability to sustain sensor network functionalities without any interruption due to sensor node failures [1].

- **Auto-configuration:** The ability of nodes to detect the presence of other nodes and organize themselves into a structured and functional network without human intervention [28].
- **Auto-reconfiguration:** The ability of the network to change its configuration to achieve a given goal. The main objectives are to reconstruct the connection paths between living nodes after a node failure or reduce the power consumption of a group of nodes. Auto-reconfiguration for a wireless sensor network, therefore, includes fault tolerance [28].

I.2.4 WSNs Medium Access Control Protocols

In the wireless sensor network, the Medium Access Control (MAC) protocols have to reach two goals. The first goal is the creation of the sensor network infrastructure. A vast number of sensor nodes are deployed, and the MAC protocol must build the communication link between the sensor nodes. The second goal is to share the communication medium reasonably and efficiently.

WSNs MAC protocols can be categorized into two classes: Contention based and Schedule based. The schedule-based protocol can avoid collisions, overhearing, and idle listening by scheduling transmit and listen periods but have strict time synchronization requirements. On the other hand, the contention-based protocols can easily adjust to the topology variations as some new nodes may join, and others may fail few years after deployment and relax time synchronization requirements. The contention-based protocols are based on Carrier Sense Multiple Access (CSMA) method and have higher costs for message collisions, overhearing, and idle listening.

I.2.4.1 CSMA/CA Protocol Description

CSMA/CA is currently used in a variety of networks for computing devices to access the communication medium. Before transmitting data, a device determines whether the medium is not being used to start transmitting data. Only devices that require to transmit data try to access the medium [29]. In this subsection, the mechanism used by the protocol CSMA/CA to achieve a successful transmission of a packet is described as depicted in Figure I.4.
The primary medium access control (MAC) technique of 802.11 is called *distributed coordination function (DCF)* [30]. It represents a decentralized mechanism where stations can share a wireless medium in the network and coordinate the use of this medium to avoid collisions. The DCF is based on the *carrier sense multiple access with collision avoidance (CSMA/CA)* protocol developed for collision avoidance when a shared medium employs a random backoff algorithm.
Any station that wishes to transmit a packet data unit (PDU) will sense the medium before

initiating a transmission. If the medium is sensed free for a period of time equals to a distributed interframe space (DIFS), the station is allowed to begin its transmission. Otherwise, if the medium is sensed as busy, the station remains in sensing until the medium will be idle for a sufficient length of time DIFS. In this situation, a station enters in a "random backoff interval" of time before transmitting. This backoff behaviour aims to decrease the possibility of collisions when stations want to transmit packets at the same time. After receiving a data packet, the destination station must sense when the wireless medium will be free for a short interframe space (SIFS) amount of time to send an acknowledgement packet (ACK) back to the sender. In wireless networks, the acknowledgement has significant importance because a sending station cannot listen to its transmission. By using the ACK, sending station confirms that its transmission was successful.
A sending station can recognise a collision if either: a wireless medium is sensed busy on termination of the transmission by another station or if an ACK packet is not received within a period of time equals to extended interframe space (EIFS). In order to avoid collisions, the MAC protocol obliges the stations to enter in a backoff procedure before sending if either [31]:

- the channel is not sensed idle for a DIFS;
- the channel is sensed busy after the station terminates a data transmission;
- a positive acknowledgement of successful transmission is not received from the target station before a Timeout; or
- the station has received an acknowledgement and wants to send another packet.

Once the backoff condition becomes true, the sending station generates a Backofftime composed of a uniformly chosen random BackoffValue of slot times, where each slot has the size aSlotTime.

$$BackoffTime = Backoffvalue * aSlotTime$$

The $backoffvalue$ can be a value in the range $[0; CW - 1]$, where CW is called the contention window or backoff window. If the wireless medium is sensed as idle, the $backoffvalue$ is decremented by 1. This decrementing procedure is interrupted when a transmission is discovered, and in this situation, the $backoffvalue$ is frozen and is resumed just after the wireless medium is sensed free for DIFS time. When the backoff timer reaches 0, the sending station can initiate its transmission. The Contention Window CW has an initial value CWmin. If two or many stations decrease their backoff counter to reach 0 at the exact moment, a collision comes. In this last situation, the CW is doubled for each retransmission to reach its maximum value $CWmax$. CW contention window is an integer equal to $CW = (CWmin + 1) * 2^{bc} - 1$, where bc is the $BackoffCounter$. Thus CW increases with the number of consecutive unsuccessful transmissions.

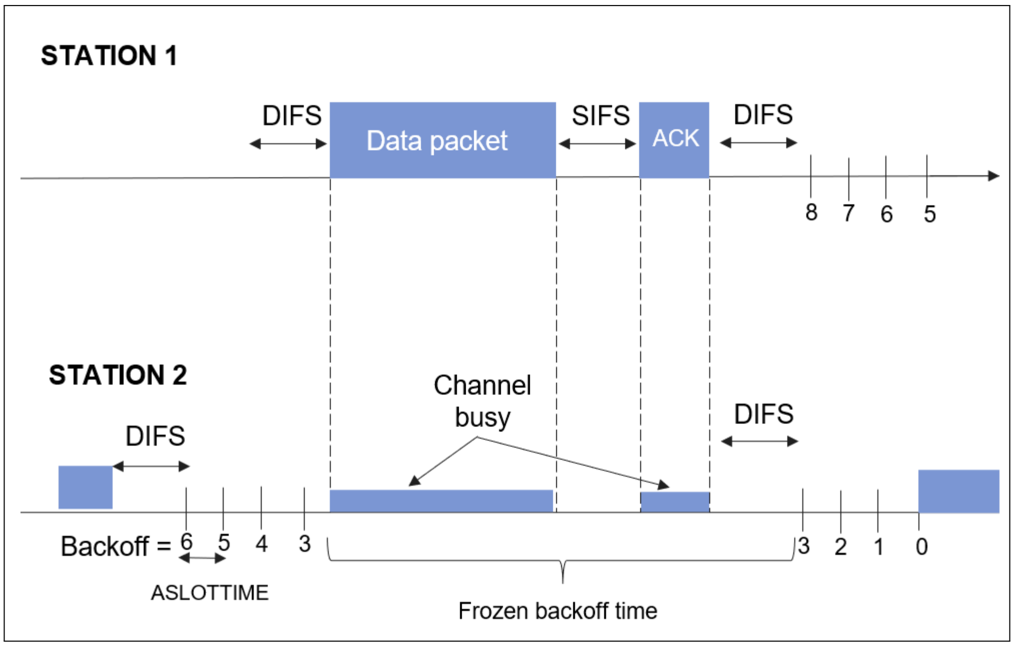


Figure I.4: An example of the backoff procedure

Figure I.4, adapted from [30], illustrates this process. Two stations, 1 and 2, shared the same wireless channel. After the end of a previous transmission, station 2 waits for a $DIFS$ time and then chooses a $backoff$ counter equal to 6 before sending the consecutive packet. Meanwhile, a new frame is generated at station 1. Station 1 senses the channel idle for a $DIFS$ and transmits the frame. Note that the transmission of packet 1 occurs in the middle of the Slot Time corresponding to a $backoffvalue$, for station 2, equal to 3. When the channel was sensed busy, the $backoff$ time is frozen to its value 3, and the $backoff$ counter decrements again only when the channel is sensed idle for a $DIFS$.

I.2.4.2 Algorithmic presentation

CSMA/CA is based upon a distributed algorithm that is executed locally on each node to determine the periods of access to the channel, this algorithm is presented by Algorithm 1. Algorithm 2 concerns the Bcakoff procedure.

I.2.4.3 MAC Protocols Performance Metrics

In wireless sensor networks, most of the applications have strict requirements in terms of quality of Service (QoS) such as throughput and delay [32]. The researchers have defined a set of performance metrics in order to evaluate and compare the performance of conscious MAC protocols.

1. **Average Delivery Ratio:** The average packet delivery ratio is the number of packets received to the number of packets sent averaged over all the nodes.

Algorithm 1: CSMA/CA Algorithm

```
1: while $Time < DIFS$ and $Free$ do
2:    $Free := SenseMedium()$;
3:    $Time := Time + Slot$;
4: end while
5: if $!Free$ then
6:    $BackOff(bc)$;
7:    Goto( 14);
8: else
9:    if $Retry < RETRY\_MAX$ then
10:       $Retry := Retry + 1$;
11:    else
12:       $Retry := 0; bc := 0$;
13:    end if
14:    TransmetPDU
15:    $Free := SenseMedium$;
16:    if $!Free$ then
17:       if $bc < bc\_MAX$ then
18:          $bc := bc + 1; BackOff(bc)$; Goto( 14);
19:       else
20:          $BackOff(bc)$; Goto( 14);
21:       end if
22:    else
23:       $Time := 0; ACK := false$;
24:       while $Time < TimeOut$ and $!ACK$ do
25:          $ACK := SenseMedium(); Time := Time + Slot$;
26:       end while
27:       if $Time > TimeOut$ then
28:          $BackOff(bc)$; Goto( 14);
29:       else
30:          $bc := 0$;
31:          $BackOff(bc)$;
32:          $SendPDU$;
33:       end if
34:    end if
35: end if
```

Algorithm 2: BackOff Algorithm

```
 2: $Free := false$;
while $!Free$ do
    $Free := SenseMedium()$;
end while
$Time := 0$;
while $Free$ and $Time < DIFS$ do
    $Free := SenseMedium()$;
    $Time := Time + Slot$;
end while
if $!Free$ then
    $Goto$( 2);
else
    $CW = (CWmin + 1) * 2^{bc} - 1$;
    $BOF := Uniform[0, CW]$;
    while $BOF! = 0$ do
         2: $Free := SenseMedium()$;
        if $Free$ then
            $BOF := BOF - 1$;
        else
            while $!Free$ do
                $Free := SenseMedium()$;
            end while
            $Time := 0$
            while $Free$ and $Time < DIFS$ do
                $Free := SenseMedium()$;
                $Time := Time + 1$;
            end while
            $Goto$( 2);
        end if
    end while
end if
```

2. **Average Packet Latency:** The average packet latency is the average time taken by the packets to reach to the sink node.

3. **Network Throughput:** The network throughput is defined as the total number of packets delivered at the sink node per time unit.

4. **Energy Consumption per bit:** The energy efficiency of the sensor nodes can be defined as the total energy consumed / total bits transmitted. The unit of energy efficiency is joules/bit. The lesser the number, the better is the efficiency of a protocol in transmitting the information in the network. This performance matrices gets affected by all the major sources of energy waste in wireless sensor network such as idle listening, collisions, control packet overhead and overhearing.

I.2.5 Energy Harvesting WSNs

The common need in most WSNs for long lifetimes and small form factors does not match up well with the power density of available battery technology [33]. This problem can limit the use of WSNs due to the need for large batteries. Therefore, researchers have integrated energy harvesting technology to make WSNs autonomous and enable widespread use of these systems in many applications. Energy harvesting allows using energy from the environment to power embedded devices and nodes of WSNs. By scavenging energy from their ambient environment, energy-harvesting wireless sensor nodes can significantly improve their typical lifetime: If the harvested energy is efficiently utilised, low-power devices can last virtually forever. Energy harvesting (EH) technology [34] is very promising due to the unlimited energy supply provided by power sources such as solar power and thermal power. By adding an EH module to the existing node structure, a novel EH-WSN was proposed in [33], which substantially changed the design of the network by eliminating the major constraint: the battery capacity. The main objective of adding a energy harvesting module in the sensor node is to make WSNs able to perform their sensing functions and wireless communication without any supervision, configuration, or maintenance. CSMA/CA my improve access control and serve to reduce collisions in carrier transmission of energy-harvesting WSNs. EHWSNs are composed of individual nodes that in addition to sensing and wireless communications are capable of extracting energy from multiple sources and converting it into usable electrical power as shown in Figure I.5.

As mentioned, MAC protocol performs a significant role in the design of WSNs as significant energy consumption is due to the sensing, reception, and transmission process. Accordingly, special attention has been paid to MAC protocol design in EH-WSNs. Therefore, MAC design becomes even more challenging for EH-WSN because the pattern of energy harvested from the environment is not easily predictable in advance.

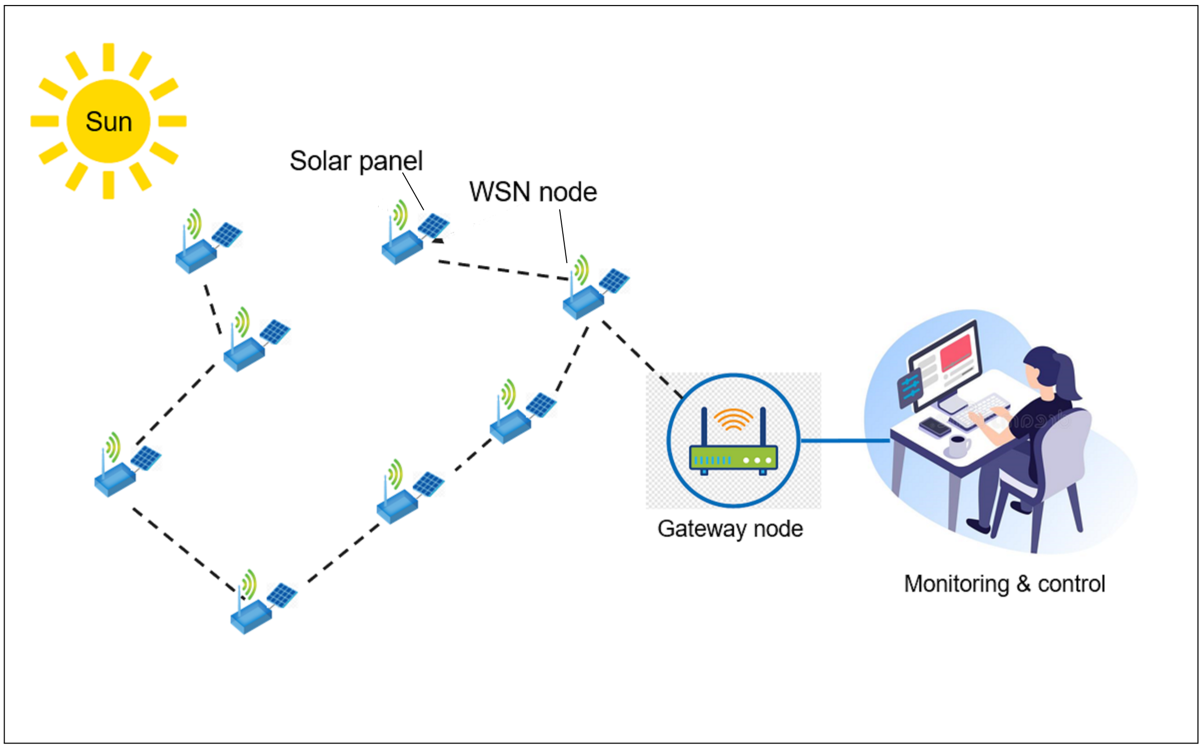


Figure I.5: Example of Solar energy harvesting wireless sensor network

I.3 Formal Analysing and Verification methods

Formal methods allow researchers to prove or disprove the correctness of a system with respect to a particular formal specification or property. Thus, they are naturally based on formal and mathematical roots. The NASA Langley Formal Methods Group [35] given the definition of formal methods as mathematically rigorous techniques and tools for the specification, design, and verification of software and hardware systems. The phrase "mathematically rigorous" means that the specifications used in formal methods are well-formed statements in mathematical logic and that the formal verifications are rigorous deductions in that logic (i.e., each step follows from a rule of inference and hence can be checked by a mechanical process.). Hence, there is a need for formal specification and verification in order to analyse and verify medium access control protocols. In a formal specification of a medium access control protocol, one must specify the protocol constraints to be satisfied in all states of the network. After the specification is done, the formal verification consists on proving that all reachable states during the execution of the system are consistent with respect to the set of predefined constraints.

As a formal method, Petri Nets are well suited to describing discrete processes and analysing the system concurrency and synchronism. A Petri Net has a graphical representation and rigorous semantics. These characteristics make Petri nets a suitable formalism to specify and analyse MAC protocols in WSNs. In the literature, one can find some works which have applied Petri net, and its extension Coloured Petri net to model/analyse protocols in Wireless Networks. In [36, 37, 38, 39, 40], Petri Nets are used as a formalism for the modelling and verification of wireless network protocols. The authors in work [36] have proposed a new model for the 1-wire protocol using timed coloured Petri net (TCPN) and CPN-Tools in the analysis phase. In [37], the authors used coloured Petri nets (CPNs) to present a modelling method for wireless network protocol. In [39], a Petri net model is presented for a medium access control protocol in WSNs named sensor medium access control protocol (S-MAC). The proposed model is based on Hierarchical Coloured Petri Nets (HCPNs). The authors considered a topology of a two-hop network with two sources and two sinks. They described the main part of the model hierarchy then sub-models have been described. For constructing and analysing the model, CPN-Tools is used. The following subsections give an informal and formal definition of Petri net and its extension hierarchical timed coloured Petri net.

I.3.1 Hierarchical Timed Coloured Petri Nets

Informally, a Petri Nets (PN) [14] is a graph composed of two kinds of nodes places and transitions. A set of arcs link places to transitions and transitions to places. The places can be marked with tokens (modelling non-typed data). These tokens are called marking of the PN. Transitions can be enabled, and if this is the case they can be fired. Firing a transition update the marking of the places in the net.

Many extensions of Petri Nets have been proposed in the literature [15, 41]. Coloured Petri Nets (CPN) [15] is an extension of Petri Nets. In CPN, each place has a type (a colour). So the tokens can be more complex and typed data. The arcs are labelled by expressions that belong to the types of their incoming places. The transition can have some guards. A guard is a Boolean expression. In a guard expression, variables that are used in the input arcs or the output arcs of the transition can be used.

Timed CPN extends CPN with a set of stamps of time. These stamps can be associated with tokens or to transitions. A stamp s associated with a token will make this token ready to be used only after the system's time will be more than s. When a stamp s is associated with a transition, all the stamps associated with tokens that are generated, when this transition is fired, are incremented with the stamp s. The following definition presented in the following paragraphs is inspired by the definition of CPN [15]. The definition of Timed CPN updated the definition of CPN with the concept of stamps, as implemented in the CPN-Tools [16].

Hierarchical CPN and Hierarchical TCPN [42] are extensions for CPN and TCPN, where the structure of the model can be organized hierarchically. A hierarchical representation of the specification is a modular representation of the model. In this modular representation, the model comprises a set of sub-models composed together to build one big model. Therefore, the specification is represented as an abstract principle model that can be refined towards an elaborated model at any time. Hierarchical Timed CPN uses *hyper transitions* in the abstract model to hold in the sub-models. The refinement of the abstract model passes through the unfolding of these hyper-transitions.

In the following subsection, the formal definition of a Hierarchical Timed CPN is presented, and the dynamic behaviour of this formalism is shown.

I.3.1.1 Formal Definition

Firstly, some necessary concepts (timed sets and timed multi-sets) which will be used in the formal definition of the Hierarchical Timed CPN are presented. Let $\mathbb{N}$ denote the set of non-negative integers. These definitions are extensions for sets and multi-sets concepts, used in CPN [15].

Definition 1. A Time set Γ is a set of non-negative integers. $\Gamma = \{\tau \in \mathbb{N}\}$.

Definition 2. A time multi-set τm, over a non-empty set X, is a function $\tau m \in [X \rightarrow \mathbb{N} \times \Gamma]$, for each $x \in X$, $\tau m(x) = (O(x), S(x))$,where $O(x) \in [X \rightarrow \mathbb{N}]$, is the number of occurrences of x and $S(x) \in [X \rightarrow T]$ is a stamp (from a Time set, in our case a set of positive integers). $\tau m(x)$ is represented as a formal sum: $\Sigma_{x \in X}(O(x)'x@+S(x))$. In this expression, the symbol @ distinguishes between the O values and the S values.

By $X_{\tau MS}$, we denote the set of all timed multi-sets over X. The non-negative integers $\{O(x)|x \in X\}$ are the coefficients of the multi-set, and the non-negative integers $\{S(x)|x \in X\}$ are the stamps of the multiset.

For example, if we take the set $X = \{2, 5\}$. A time multi-set on this set can be $(2'2@+2)+ +(1'5@+4)$. This time multi-set represents the set that contains two occurrences of 2 stamped with the stamp 2 (each occurrence has the stamp=2) and one occurrence of 5 (with a stamp=4). $\{1, 2\}_{MS}$ is the set of all timed multi-sets of the set $\{1, 2\}$, which is an infinite set.

In the following definitions, some notations are used to facilitate the presentation. We use $Type(E)$ to denote the type of the expression E. We use $Var(E)$ to extract the set of variables used in the expression E.

Definition 3. A Timed CP-net is a tuple $TCPN = (\Sigma, P, T, A, C, G, Exp, I, \tau)$, where:

1. Σ is a finite set of non-empty **types**, also called colour sets,
2. P is a finite set of **places**,
3. T is a finite set of **transitions**,
4. A is a finite set of **arcs** such that: $A \subseteq (P \times T) \cup (T \times P)$ and $P \cap T = P \cap A = T \cap A = \phi$. If $a = (p, t)$ is an arc in A, we say that p is an input place for t, we denote this: $p \in^{\circ} t$. If $a = (t, p)$ is an arc in A, we say that p is an output place for t, we denote this: $p \in t^{\circ}$.

5. C is a **colour** function (it defines the type of each place in the TCPN). It is defined from P into Σ.

6. G is a **guard** function. It is defined from T into expressions such that:
   if $tr \in T$ then $Type(G(tr)) =$ Boolean
   and $Type(Var(G(tr)) \subseteq \Sigma$

7. Exp is an arc **expression** function. It is defined from A into expressions such that:
   $a \in A : Type(Exp(a)) = C(p)_{\tau MS}$ and
   $Type(Var(Exp(a))) \in \Sigma$, where p is the place component in a.

8. I is an initialisation function (or an initial marking of the set of places). It is defined from P into closed expressions such that: for each $p \in P : Type(I(p)) = C(p)_{\tau MS}$.

9. τ: is a **time function** that associates with each transition a stamp. $\tau : T \rightarrow \Gamma$, ($\Gamma$ is a time set).

Definition 4. A Hierarchical TCPN is a finite set of nets $H = \{N_1, N_2, N_3, \dots\}$. Each net N in H is a tuple $N = (\Sigma, P, T, A, C, G, Exp, I, \tau, h)$, where:

1. Σ, A, C, Exp, I: are defined as in TCPN,

2. $P = OP \cup IP$, where: OP is a set of ordinary places as defined in TCPN, and IP is a set of interface places. An interface place is a place that is shared between more than one net. An interface place is used in communication between nets in H.

3. $T = OT \cup HT$, where: OT is a set of ordinary transitions as defined in TCPN, and HT: a set of hyper-transitions (can be empty).

4. G: is a **guard** function. It is defined from T into expressions such that:
   if $tr \in T$ then $Type(G(tr)) \in$ Boolean
   and $Type(Var(G(tr)) \subseteq \Sigma$

5. τ: is a **time function** that associates with each ordinary transition a stamp. $\tau : OT \rightarrow \Gamma$. ($\Gamma$ is a time set).

6. h: is a function that maps each hyper transition to a net. $h : HT \rightarrow H$. We require that if ht is a hyper transition in N, so $h(ht)$ must not be N and must not lead to N indirectly. This means recursion is not allowed in the model. The input-places of ht ($^\circ ht$) and the out-places of ht (ht°) are places in the net $h(ht)$.

I.3.1.2 Dynamic Behaviour and Semantics of Hierarchical Timed CPN

The dynamic behaviour of the net is obtained when the transitions are fired. A transition can be fired if it is enabled. To be enabled, a transition requires some preconditions. These preconditions depend on the marking of its input places, the expressions labelling its input arcs, and its associated guards. Once the transition is fired, some post-conditions will be satisfied. Firing a transition will update the marking of its input and output places. The new marking depends on the labels of the input-arcs and output-arcs of this transition. Some necessary concepts are tackled first to present the preconditions of firing a transition and how the marking is updated.

The function $Var(tr)$ is adopted to extract the set of variables used in the guards associated with the transition tr, or used in the expressions labelling input-arcs or output-arcs of tr.

Definition 5 (**binding**). A binding of a transition tr is a function b defined on $Var(tr)$, such that:

(i) For each $v \in Var(tr)$: $b(v) \in Type(v)$,

(ii) The binding of tr satisfies the guard function of tr. Formally, this is written: $G(tr)[b] = true$, or $G(tr)[b]$.

Definition 6 (**timed-binding**). A timed-binding of a transition tr is a couple $< b, t >$, where b is a binding defined on $Var(tr)$, t is a time, and at the time t, we have :

(i) For each $v \in Var(tr) : b(v) \in Type(v)$,

(ii) $G(tr)[b]$.

Definition 7 (**binding element**). A binding element be is a pair (tr, b), such that tr is a transition and b is a binding of tr. The set of all binding elements of a transition tr is denoted by $BE(tr)$. The set BE denotes the set of all binding elements for a CPN.

Definition 8 (**timed binding element**). A timed binding element tbe is a pair $(tr, < b, t >)$, such that tr is a transition and $< b, t >$ is a timed-binding of tr. The set of all timed binding elements of a transition tr is denoted by $tBE(tr)$. The set tBE denotes the set of all timed binding elements for a Timed CPN.

Definition 9 (**timed-marking**). Let L be the set of tokens in the place p at the time t. Then the timed-marking of the place p at the time t, denoted $M_t(p)$, is defined as the multi-set of tokens x in p with a stamp less than or equal to t: $M_t(p) = \Sigma_{x \in L \ and \ S(x) \leq t}(O(x)'x@ + S(x))$.

The initial timed-marking, denoted M_0, is the timed-marking of the net at the time 0.

Definition 10 (**time-enabled**). A transition tr is time-enabled at time t in a marking M if and only if there is a timed-binding $[b, t]$ which satisfies the property: for each $p \in^{\circ} tr$: $Exp(p, tr)[b] \leq M_t(p)$. Where $^{\circ}tr$ is the set of input-places of the transition tr. We write that $(tr, [b, t])$ is time-enabled at the time t.

Definition 11 (**stamped-expression**). Let X be a timed multi-set, exp be an expression defined over X, and t be a stamp, the stamped-expression $exp@+t$ denotes the expression exp in which the stamps of all its operands are incremented with t. for example, if $exp = (2'1@+2){+}{+}(1'2@+4)$, then $exp@+2 = (2'1@+4){+}{+}(1'2@+6)$.

Definition 12 (**firing precondition and reachable marking**). When a transition tr is time-enabled at time t (with a time-binding $[b, t]$), in a timed-marking M_t, it can be fired. Firing tr is an event that can take a duration Δt. Firing tr changes the marking M_t to another marking $M_{t+\triangle t}$, such that:

$$\text{for each } p \in P, M_{t+\Delta t}(p) = (M_t(p)\check{\ } Exp(p, tr)[b]) + (Exp(tr, p)[b]@+(tr)).$$

We say that $M_{t+\Delta t}$ is directly reachable from M_t, and this is written: $M_t[tr > M_{t+\Delta t}$.

Definition 13 (**firing a hyper transition**). Preconditions to fire a hyper transition ht defined in a net N (in a HTCPN H), are the same as well as for an ordinary transition. When ht is fired, ht is unfolded to the net $h(ht)$. Firing ht changes the marking of N as well as the marking of $h(ht)$.

I.3.1.3 Reachability Graph of a Timed CPN

The analysis of Timed CPN models can be done through computing reachability graphs [15]. A reachability graph is a graph with a node for each timed reachable marking and an arc for each occurring timed binding element. Let M_1 and M_2 be two reachable timed markings and tb is a timed binding element enabled in M_1. If the occurring of tb transforms M_1 into M_2, then we denote this by: $M_1[tb > M_2$. The reachability graph is a directed graph considered as a couple (V, A), such that:

- V: a set of reachable timed markings. $V = \{M_1, M_2, \ldots\}$.
- A: a set of arcs linking nodes of V. $A \subseteq V \times tBE \times V$.

Given a Timed CPN, one can use the algorithm proposed in [15], to construct the reachability graph. When using CPN-Tools, this construction is semi-automatic.

I.3.1.4 An example of a Timed CPN

In order to clarify the previous formal definitions and aspects related to timed CPN, we consider a simple example. The example illustrates the concepts of marking, timed-marking, timed-binding, timed-enabled, and stamped-expression. The example is realised using the CPN-Tools. Let us consider a simple CPN (see Figure I.6) composed of three places P1, P2, P3 and one transition T. Initial markings of places are shown next to each place in green colour. Place P1 has the type: *Timed_INT* (i.e., timed integer), hence it contains a timed-marking $1'1@2$. This timed-marking means that place P1 contains an integer token equals, which will be ready at a time equals to 2 time-units. P2 has the type *INT* (i.e., integer) and its initial marking is $2'2{+}{+}1'4$, which means that P2 contains three integers (two occurrences of value 2 and a unique occurrence of value 4). $2'2{+}{+}1'4$ is an element in the integer multi-set. Transition T has two input places P1 and P2, one output place P3 and a guard $[x > 2]$. Arcs (P1, T) and (P2, T) are labelled with expressions y, x which belong to the types *Timed_INT* and *INT*, respectively. P3 has the type *INT timed* and its initial timed-marking is empty. Arc (T, P3) is labelled with a stamped expression $x@{+}2$, which means that firing transition T will add a stamped token which value is x and which stamp is 2. As shown in Figure I.6, transition T has a green colour, which means that T is time-enabled at a time equals to 2 with timed-binding $[y = 1, x = 2]$. After firing T, the marking of the CPN is updated as shown in Figure I.7. The new marking of P3 is $1'4@4$, which means a token with a value 4 and a stamp equals to 4. This new stamp is the sum of the enabling time which was 2 and the stamp of the stamped expression $x@+2$.

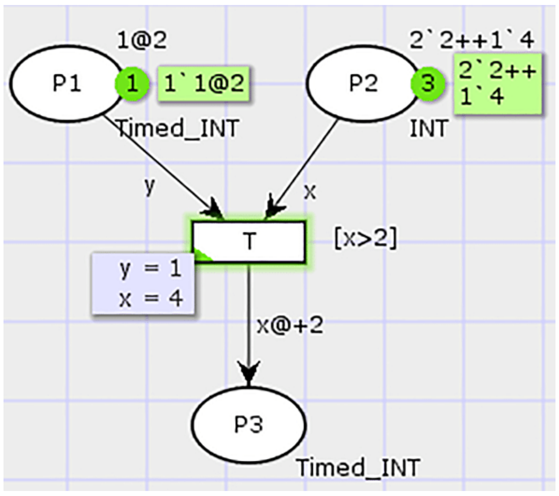


Figure I.6: A simple example of a CPN: before firing the transition

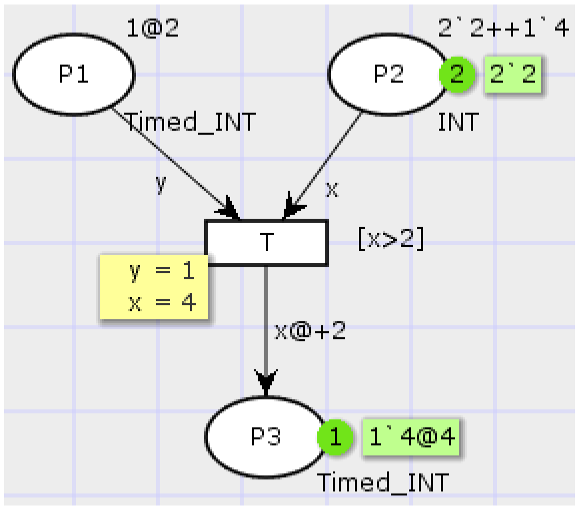


Figure I.7: A simple example of a CPN: after firing the transition

I.3.2 Linear Temporal Logic

Linear Temporal Logic (LTL) over infinite traces was originally proposed in Computer Science as a specification language for concurrent programs [43]. Temporal Logic is a particular type of Modal Logic; it provides a formal system for qualitatively describing and reasoning how the truth values of assertions change over time [44]. In a system of Temporal Logic, various temporal operators or \modalities" are provided to describe and reason about how the truth values of assertions vary with time. Typical temporal operators include sometimes F, which is true now if there is a future moment at which F becomes true, and always G, which is true now if G is true at all future moments. LTL syntax and semantics are defined briefly as follows:

I.3.2.1 Syntax:

The basic LTL formula element is an atomic proposition α ($\alpha \in AP$, where AP is the set of atomic propositions). Hence, the LTL formulae over AP is constructed with logical operators (such as $\neg$, $\wedge$ and $\vee$) and temporal modal operators. The temporal modal operators are:

- $\bigcirc$ or **X**: is a unary prefix basic modal operator and it is read as next.
- **U**: is a binary infix basic modal operator and it is read as until.
- $\square$ or **G**: is a unary prefix additional modal operator and it is read as always or Globally (for now on forever).
- $\lozenge$ Or **F**: is a unary prefix additional modal operator and it is read as eventually or in the future (sometimes in the future).

Hence, over the set AP where $\alpha \in AP$, an LTL formula can be written according to the following grammar:

$$\psi ::= true \mid \alpha \mid \neg\psi_1 \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \mid \mathbf{X}\,\psi_1 \mid \psi_1\,\mathbf{U}\,\psi_2 \mid \mathbf{F}\,\psi_1 \mid \mathbf{G}\,\psi_1$$

I.3.2.2 Semantic:

LTL formulae stand for paths trace. Hence, an infinite sequence of truth evaluations of variables in AP (a path) can satisfy an LTL formula or cannot. This sequence can be considered as an ω-word over alphabet 2^{AP} (Power AP is the set of sets of atomic propositions and words formed by using letters from this set). Let the set of infinite words over 2^{AP} denoted by $AP - inf$ and $w \in AP - inf$ where $w = w_0\ w_1\ w_2 \ldots w_n$. The satisfaction relation between w and an LTL formula is defined as follows:

- $w \models \alpha$ iff $\alpha \in w_0$
- $w \models \neg\psi$ iff $w \not\models \psi$
- $w \models \psi_1 \vee \psi_2$ iff $w \models \psi_1$ or $w \models \psi_2$
- $w \models \psi_1 \wedge \psi_2$ iff $w \models \psi_1$ and $w \models \psi_2$
- $w \models \mathbf{X}\,\psi$ iff $\psi \in w_1$
- $w \models \mathbf{G}\,\psi$ iff $\forall\, i \geq 0 : \psi \in w_i$
- $w \models \mathbf{F}\,\psi$ iff $\exists\, i \geq 0 : \psi \in w_i$
- $w \models \psi_1\,\mathbf{U}\,\psi_2$ iff $\exists i \geq 0$ such that $\psi_2 \in w_i$ and $\forall j \leq i : \psi_1 \in w_j$

I.4 Machine Learning for Prediction

Machine learning (ML) has recently attracted considerable attention for its ability to predict a wide variety of complex phenomena accurately [22]. ML ultimately aims at developing computer algorithms that improve automatically through experience [45]. Various researchers have shown the success of ML algorithms in many fields such as machine health monitoring [46] and diagnostics in healthcare [47]. Applying ML in WSNs lies in exploiting historical data to improve the performance of sensor networks on given tasks without the need for re-programming. Researchers have categorized ML systems into three kinds of learning systems: (i) supervised learning, (ii) unsupervised learning, and (iii) reinforcement learning. In this section, ML definition and algorithms are tackled.

I.4.1 Machine Learning Definition

Machine learning [48]is a part of artificial intelligence focused on constructing algorithms that make predictions based on data without programming to perform a task. Many definitions of ML are adopted in the literature. Researchers in [49] define ML as the development of computer models for learning processes that provide solutions to the problem of knowledge acquisition and enhance the performance of developed systems. Other definition is presented in [50] where authors capture the essence of ML as the adoption of computational methods for improving machine performance by detecting and describing consistencies and patterns in training data. ML aims to identify a function $f : X \rightarrow Y$ that maps the input X into output Y [19]. Function f is chosen from different function classes, dependent on the type of learning algorithm used.

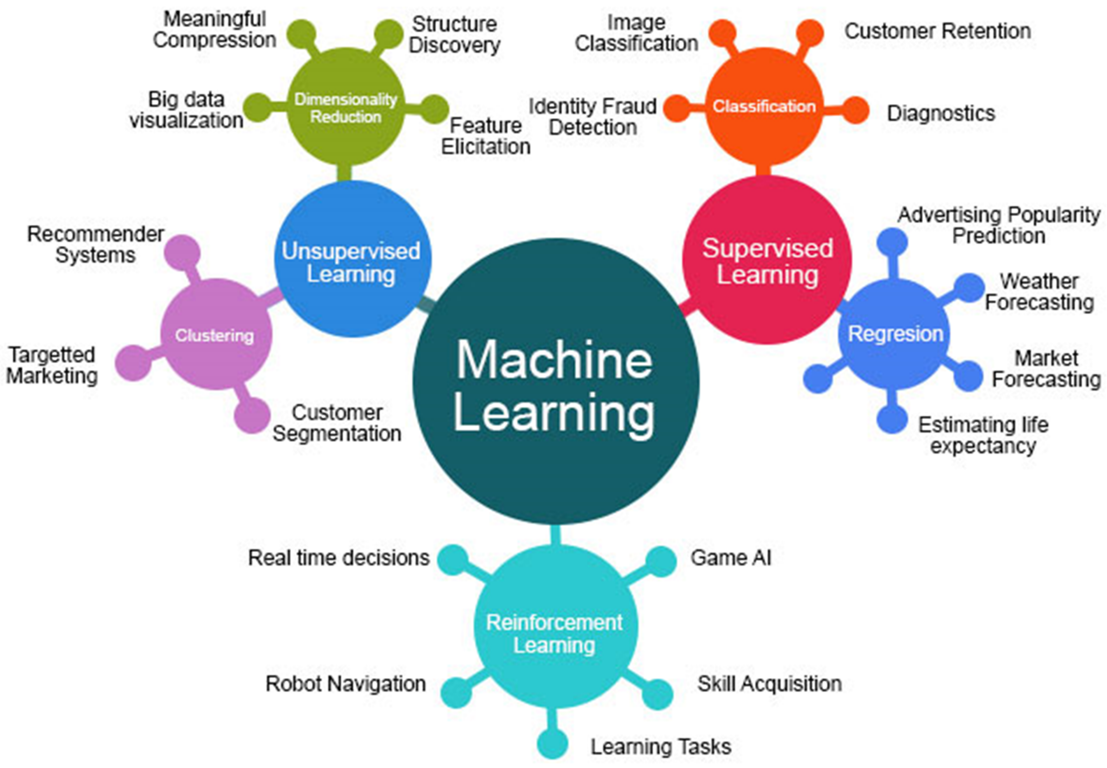


Figure I.8: Machine learning techniques.

I.4.2 Machine Learning Algorithms

Existing machine learning algorithms can be categorized by the intended structure of the model. For that, researchers classify ML algorithms mainly into three categories by the type of datasets used as experience. These categories are supervised learning, unsupervised learning, and reinforcement learning [51] as shown in Figure I.9. Other learning systems combine two categories, such as semi-supervised learning that use labelled and unlabelled data, more details in the followings subsections.

I.4.2.1 Supervised Learning

Supervised learning systems make use of labelled datasets, where x represents a data point and y the corresponding true prediction for x. This training set of input-output pairs is used to find a deterministic function that maps any input to an output, predicting future input-output observations while minimizing errors as much as possible [52]. Supervised learning problems can be further grouped into regression and classification problems:

- Classification: a classification problem is when the output variable is a category, such as “disease” and “no disease”.Classes can be called as targets/labels or categories (See Figure I.9).
- Regression: a regression problem is when the output variable is a real value, such as "dollars" or "weight".

Some popular examples of supervised learning algorithms are: Linear regression, random forest, support vector machines, decision tree, neural network (Multiple layer perceptron), K-Nearest neighbours and naïve Bayes.

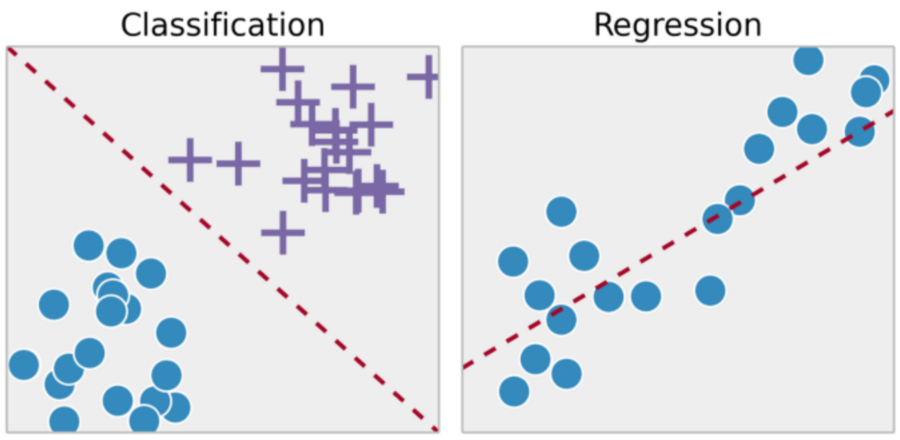


Figure I.9: Supervised learning

I.4.2.2 Unsupervised Learning

Unsupervised learning systems use unlabelled datasets to train the system. The objective of unsupervised learning is to derive structure from unlabelled data by investigating the similarity between pairs of objects, and is usually associated with density estimation or data clustering [53]. Unsupervised learning problems can be further grouped into clustering and association problems (See Figure I.10).

- Clustering: a clustering is a way of grouping the data points into different n clusters, consisting of similar data points. The objects with the possible similarities remain

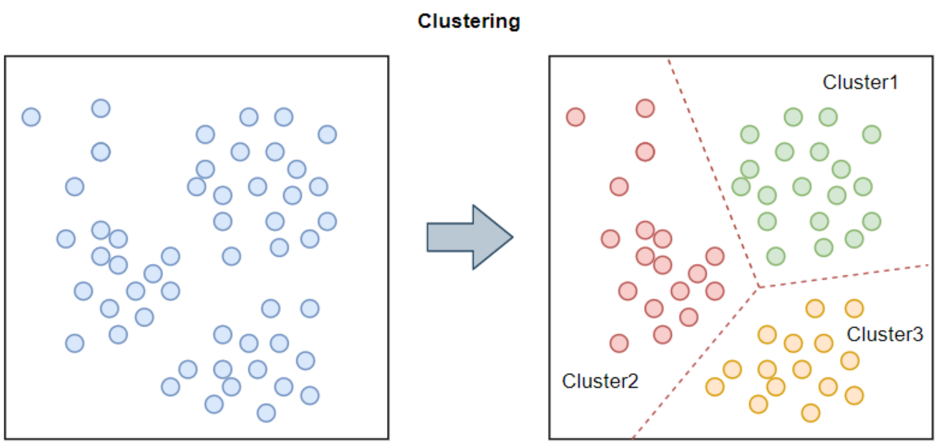


Figure I.10: Unsupervised learning

in a group that has less or no similarities with another group. It does it by finding some similar patterns in the unlabelled dataset such as shape, size, color, behaviour, etc., and divides them as per the presence and absence of those similar patterns.

- Association: checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable [54]. It tries to find some interesting relations or associations among the variables of dataset. It is based on different rules to discover the interesting relations between variables in the database.

Some popular examples of unsupervised learning algorithms are: k-means for clustering problems, apriori algorithm for association rule learning problems.

I.4.2.3 Reinforcement Learning

Reinforcement learning systems do not experience a fixed dataset but a feedback loop between the system and its experiences [55]. A dynamic environment is considered as shown in Figure I.12, state-action-reward triples are observed as the data. The objective of reinforcement learning is mapping situations to actions to maximize rewards.

I.4.3 Machine learning process

Machine Learning is a data-driven process that starts by pre-processing the collected data until the model construction spits out predictions and insights. The method of performing machine learning usually requires many steps that are explained in the following sections.

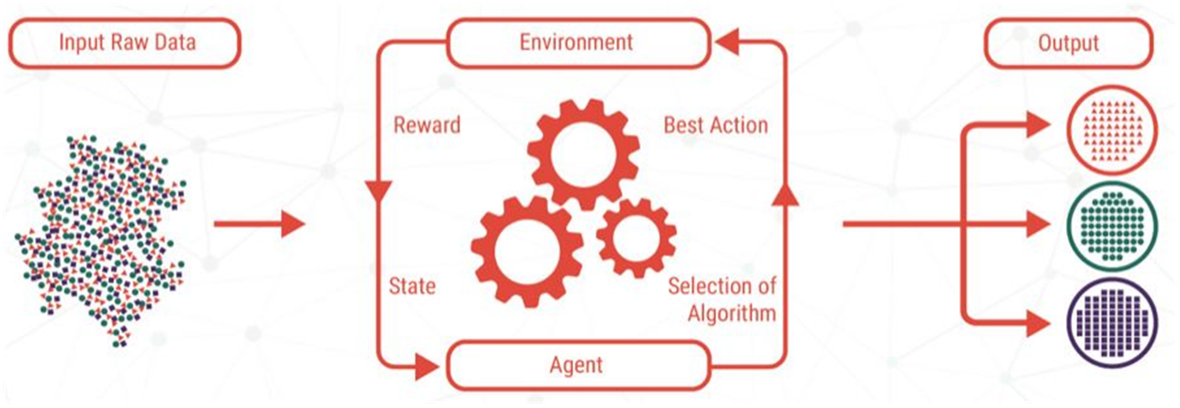


Figure I.11: Reinforcement Learning

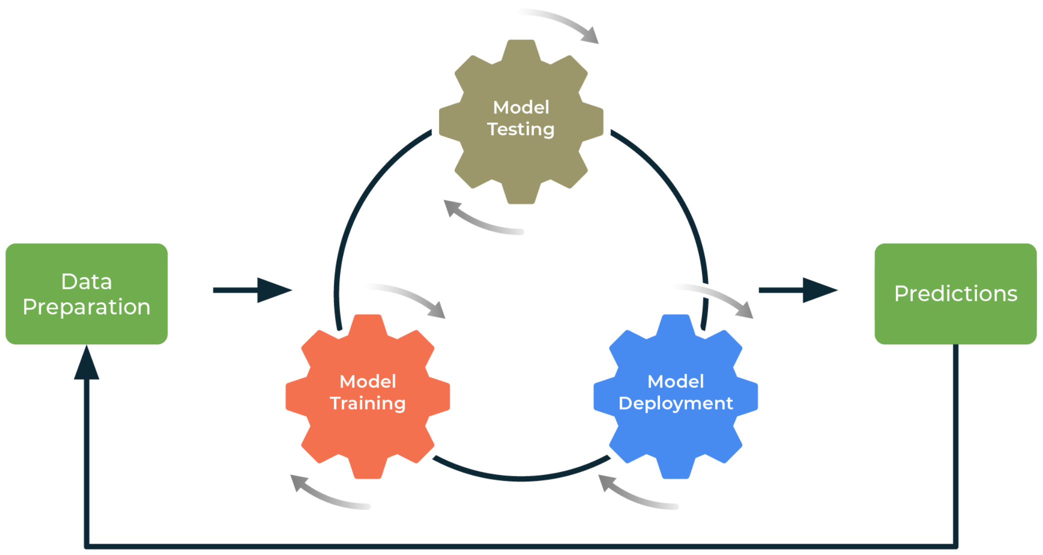


Figure I.12: Machine learning process

I.4.3.1 Data collection

This is the first and the most important step for any ML process because the quality and quantity of data gathered will determine how good our predictive model will be.

I.4.3.2 Data preprocessing

Data preprocessing is the crucial step while creating a machine learning model. It is a process of preparing the raw data and making it suitable for a machine learning model.

I.4.3.3 Choosing a model

There are various existing models developed by data scientists which can be used for different purposes. These models are designed with different goals. For instance, some models are more suited to dealing with texts, while another model may be better provided to handle images. We need to make the choice that meets our expected outcome. The options for machine learning models can be explored across three broad categories shown in Figure I.9.

I.4.3.4 Model Training

Training a model means providing the Machine Learning Algorithm along with the training data. This process then repeats. Each iteration or cycle of updating the weights and biases is called one training "step".

I.4.3.5 Model evaluation

Evaluating a model is a crucial step throughout the development of the model. Evaluation metrics have a correlation with machine learning tasks (classification, regression, etc.).

I.5 Conclusion

The use of formal methods has proved their efficiency to ensure such analysis, in medium access control protocols. As a formal method, Petri nets are well suitable to describe discrete processes and to analyse the system concurrency and synchronism. The analysis of Petri net models allows the designer to prove the consistency of the protocol. One can use the reachability graph of Petri net model to analyse and to verify the protocol. Coloured Petri Nets (CPNs) represent an extension of Petri nets with more expressive power. The modelling of MAC protocols using CPNs is more practical than using classical Petri Nets.

On the other hand, the complexity of the analysis process in large-scale networks makes the assurance of their consistency and correctness a challenging problem. Moreover, artificial intelligence can be exploited to solve this problem. As a part of artificial intelligence, machine learning is a powerful technique to predict the evolution of performance metrics defined in proposed models. The following chapter aims to present the first contribution of this thesis, which is a formal approach of CSMA/CA MAC protocol based on Hierarchical Timed Coloured Petri Nets (HTCPN).

Chapter II

First Contribution: Using Hierarchical Timed Coloured Petri Nets in the Formal Study of CSMA/CA Protocol

II.1 Introduction

Formal modelling and formal analysis are efficient techniques to model and to evaluate networking protocols and algorithms in Wireless Sensor Networks (WSNs). WSNs face many challenges due to the fact that the sensor nodes are characterized by limited energy, limited memory, and weak computing power. Therefore, designing an effective medium access control (MAC) protocol is a challenging task in aim to determine the channel access control capabilities and the energy consumption properties of WSNs [56]. Many approaches have been proposed such as simulation [57], testing and formal verification to improve the reliability of WSNs and their protocols. Several formalisms have been used to model and to analyse the behaviour of wireless networks [58, 59, 60, 61, 62].

Among these formalisms, Petri nets formalism are the most exploited formalisms for WSNs modelling/analysis. Coloured Petri nets (CPNs) have become a useful formalism for researches in computer science mainly to model wireless communications [39, 38, 63, 64, 37, 36]. Modelling and analysis results are obtained by using automatic tools such as CPN-Tools.

In the current chapter, a new approach for modelling and analysing CSMA/CA protocol in WSNs is proposed [65]. In addition, the proposed model can be used to perform the qualitative and quantitative verification of the protocol. In this approach, the arriving packet in the network is random and the number of nodes is variable. A set of properties are evaluated using the capabilities of hierarchical timed coloured Petri nets which provide the performance evaluation using CPN-Tools. The qualitative properties are formalised using Linear Temporal Logic (LTL).

The rest of this chapter is organized as follows: Section II.2 presents some related works. Section II.3 describes the new HTCPN model for the CSMA/CA protocol in WSNs. Section II.4 reports our protocol performance analysis results. Finally, a conclusion is made in Section II.5.

II.2 Related Work

In literature, several work have proposed the use of formal methods to model, verify and evaluate WSNs protocols [66, 67, 68, 58, 59, 60, 61]. In [66], the authors present a performance study of the Distributed Coordination Function (DCF), which is the fundamental contention based access mechanism of 802.11 standard [12] for wireless local area networks(WLANs). The authors used Stochastic Petri Nets (SPNs) as a modelling formalism to describe the IEEE 802.11 MAC protocol. They modelled all stations in an IEEE 802.11 in one SPN model. They developed two compact and analytically tractable models which can capture most of relevant system aspects but they failed to model some aspects of IEEE 802.11 DCF. The proposed model does not exactly capture freezing of the backoff counter at a station when some other station captures the channel. The authors did not validate their work using network simulations.

In [67], a simulation study of the performance aspects of IEEE 802.11 Wireless Local Area Networks has been presented. The authors have developed a Stochastic Petri nets model to describe the protocol. A numerical result is computed by means of the SPNL [69] simulation component of the software tool TimeNET [70]. Results have also been obtained from a compact analytical model which was derived from the detailed model based on the insight due to the simulation data. The authors were interested in two performance indices, throughput and mean waiting time.

The work [40] presents a formal model of Carrier Sense Multiple Access Collision Avoidance [11] (CSMA/CA) mechanism. Hierarchical coloured Petri nets formalism is used to construct this model. The authors have used CPN-Tools as modelling and simulation environment. The proposed HCPN model of CSMA-CA mechanism is composed of three pages. The authors have verified two qualitative properties: boundness and liveness.

The authors in [71] have proposed a detailed 802.11b model based on Object-oriented Petri Nets that precise backoff procedure and time synchronization. They have proposed a modular model that enabled them to evaluate the impact of network performances on the performances of distributed discrete event systems. Their approach considers two basic modules to model IEEE 802.11 network: workstation based module and medium based module.

The work in [72] presents a deterministic Petri-Net model of the IEEE 802.15.4 CSMA/CA process, that is timer driven and operates within the bounds of the contention access period (CAP). Using this model, the authors have analysed the performance characteristics of the CSMA/CA process, especially in terms of channel throughput and energy consumption.

A formal approach is presented in [58] where authors have proposed a new variant of the IEEE 802.11 CSMA/CA protocol with DCF mode (Distributed Coordination Function). In [58], each station must disconnect every time when its signal-to-noise ratio is below a specific threshold. These disconnections are intended to minimize the number of collisions and to enhance the transmission rate. In [58], the authors have used timed automata and UPPAAL [] for modelling the CSMA/CA protocol. Based on their model, the authors have verified a set of qualitative properties: safety and liveness, such as deadlock-freedom and the successful termination of a transmission.

For modelling and verification of the MAC level protocols, the presented work in [59] describes a statistical model checking based approach to analyse and evaluate the CSMA/CA MAC protocol in WSNs. In [59], the authors have modelled stochastic uncertainties which are relatively common in WSN systems behaviour, as disconnections and failures using probabilistic timed automata in the UPPAAL. Based on their model, the authors have realised a verification of a set of qualitative properties such as reachability, safety liveness and fairness and also a set of quantitative properties like: sending rate, receiving rate and collisions rate.

In [37], authors present a modelling method for wireless network protocol using coloured Petri nets. They have applied this method to model and analyse IEEE 802.15.4 wireless protocol by exploiting the hierarchical features of CPN Tools. They develop a compact and scalable CPN model by using hierarchical and symmetrical modelling techniques. The authors have simulated their CPN model in terms of throughput, delivery ratio, delay, and energy cost for three, six and nine nodes in a typical star topology, in which all devices are transmitting to the network coordinator. To validate their results, they compared them with the ones obtained using the ns-2 simulation tool [73].

In [61], the authors have proposed a stochastic generic model of CSMA/CA protocol in WSNs. They use Priced Timed Automata (an extension of Probabilistic Timed Automata) to model the protocol. The authors have described how model analysis can be performed through statistical model checking implemented in the UPPAAL tool. The proposed model is composed of two synchronized stochastic timed automata: wireless station (WS) model, and medium model. Work in [61] performs a qualitative evaluation that proves the correctness of the model, and a quantitative evaluation using the statistical model checking to measure the probabilistic performance of the protocol. With regard to the work of [60], the authors have studied CSMA/CA of energy-harvesting WSNs using timed automata at the modelling level and UPPAAL at the analysis level.

In order to compare the above works with the proposed work in this chapter, a set of comparison criteria have been proposed in the formal specification (i.e. formalism, modularity, backoff procedure, arriving packets, number of nodes and network topology). Indeed, in the formal verification, a set of other specific comparison criteria is defined (i.e. modelling/verification tool, qualitative verification and quantitative verification). Table II.1 shows the achieved comparison. This table aims to illustrate the differences between the proposed work presented in this chapter and the other related work existing in the literature.

II.3 Modelling CSMA/CA with Hierarchical Timed Coloured Petri Nets

The modelling process is the first step in which the designer must investigate a set of events that occur in a system, the events dynamic can easily modify the state of the system. In medium access control using CSMA/CA protocol (our case study), we identify four types of events: sending a data packet, receiving acknowledgement, receiving a data packet, and sending acknowledgement. In the Petri Net model, these events are modelled by transitions. In CSMA/CA protocol, the occurred events are constrained by several conditions which are presented by a set of pre-conditions and post-conditions. It is important to mention that when pre-conditions are satisfied the post-conditions will be also satisfied. The pre-conditions are modelled by input-places, expressions on input-arcs, and associated guards. The post-conditions are modelled by output-places and expressions on

Attributes / Works			[66]	[67]	[40]	[71]	[72]	[58]	[60]	[37]	[61]	Our model
Formal specification		Formalism	SPN	SPN	HCPN	OOPN	TPN	TA	TA	HCPN	PTA	HTCPN
		Modularity	No	Yes	Yes	Yes	No	No	No	Yes	No	Yes
		Backoff Procedure	Not detailed	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes
		Arriving Packets	Yes	Yes	No	No	Yes	No	No	Yes	No	Yes
		Number of nodes	10	10	20	12	9	-	-	9	Variable	Variable
		A Network topology	No	No	No	No	No	No	No	Yes	Yes	Yes
Formal verification		Modelling/Verification Tool	TimeNET	TimeNET	CPN-Tools	Renew 2.1	-	UP-PAAL	UP-PAAL	CPN-Tools	UPPAAL	CPN-Tools
		Qualitative verification	✗	✗	✓	✗	✗	✓	✓	✗	✓	✓
	Quantitative verification	Throughput	✓	✓	✗	✓	✓	✗	✗	✓	✗	✓
		Delivery ratio	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓
		Delay	✗	✗	✗	✓	✗	✗	✗	✓	✗	✓
		Waiting time	✓	✓	✗	✗	✗	✗	✗	✗	✗	✓
		Waiting time for ACK	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

Table II.1: CSMA/CA protocol analysis related works.

SPNs: Stochastic Petri Nets, **OOPs:** Oriented Object Petri Nets, **TA:** Timed Automata, **PTA:** Priced Timed Automata
(✓): verified (✗): not verified (-): not mentioned

output-arcs.

In this section, the proposed Hierarchical Timed Coloured Petri Net(HTCPN) model for CSMA/CA protocol in WSNs is described. The global model is constructed taking the advantage of the hierarchical capabilities of HCPNs [74]. Hierarchical CPNs facilitate the construction of large models by the possibility of dividing the model into smaller models connected to each other using well-defined interfaces (i.e. substitution transitions and interface places). Based on this, more details can be added to the proposed model. In the following subsections, we use often the syntax of CPN-Tools specification language. The CPN-Tools represents an implementation of the Hierarchical Timed CPNs, this tool allows the graphical representation of HTCPN models and their analysis. Starting by describing the abstract model followed by the sub-models presentation.

II.3.1 The global HTCPN model

Using the hierarchical principle in HTCPNs, the global model can be constructed as an abstract model as shown in Figure II.1. In the abstract model, the sensors of the network and the base station are represented as hyper transitions. The interaction among these components of the network is modelled through interfaces places. This model gives us an overview of the mechanism used by CSMA/CA MAC protocol to access the channel for sending a data packet, where all the events are modelled. The folding technique [75] is used to construct a compact and scalable model. In the current work, the proposed model is described as follow: it includes five sensors and one base station. A sensor node can send a data packet to the base station and the last one can respond with an acknowledgement. The communication is made through a shared channel. The model hierarchy is composed of six substitution transitions representing network sensors and the base station as depicted in Figure II.1. Transitions: A, B, C, D, E and $Base\_Station$ are hyper-transitions and each transition represents one of the former sub-models. Since, all the sensor nodes in the network have the same features, we model them with one sub-model that we call $Sensor$ and we create five instances of this sub-model (sub-page): A, B, C, D and E. These substitution transitions represent the events of sending data and receiving acknowledgement realized by sensor nodes. Receiving data and sending acknowledgement events are represented by substitution transition $Base\_Station$.

II.3.2 CPN model for Sensors

In this subsection, CPN sub-model is described for the sensor nodes and the network nodes sub-model includes the processes of sending Data, receiving acknowledgement (ACK) and the Backoff Procedure. Figure II.2 depicts these events. CPN allows using the same Petri net structure to model multiple nodes. Table II.2 and Table II.3 are an overview tables presenting the most important transitions and places (respectively) that are shown in figure II.2 and their description.

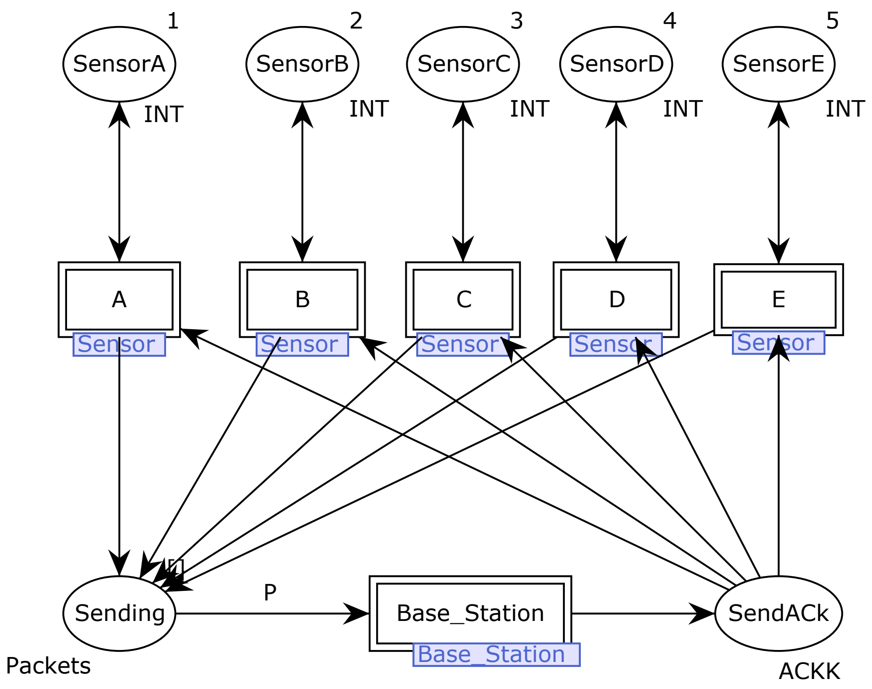


Figure II.1: HTCPN abstract model for CSMA/CA protocol

Transition	**Description**
Init	Initiating of arriving packets
Arrive	Arriving of a packet
Sense	Sensing for the medium if it is free or no
DIFS	Blocking for amount of time that a node waiting before sending a packet
Send	Sending a packet from the sensor node to the base station
SenseB	Sensing in Backoff procedure
TimeOut	The acknowledgement time out
ReceiveACK	Receiving an ACK from the base station
Successed	Sending successes
Backoffing	The node enters in Backoff procedure

Table II.2: An overview table with all important transitions and their description

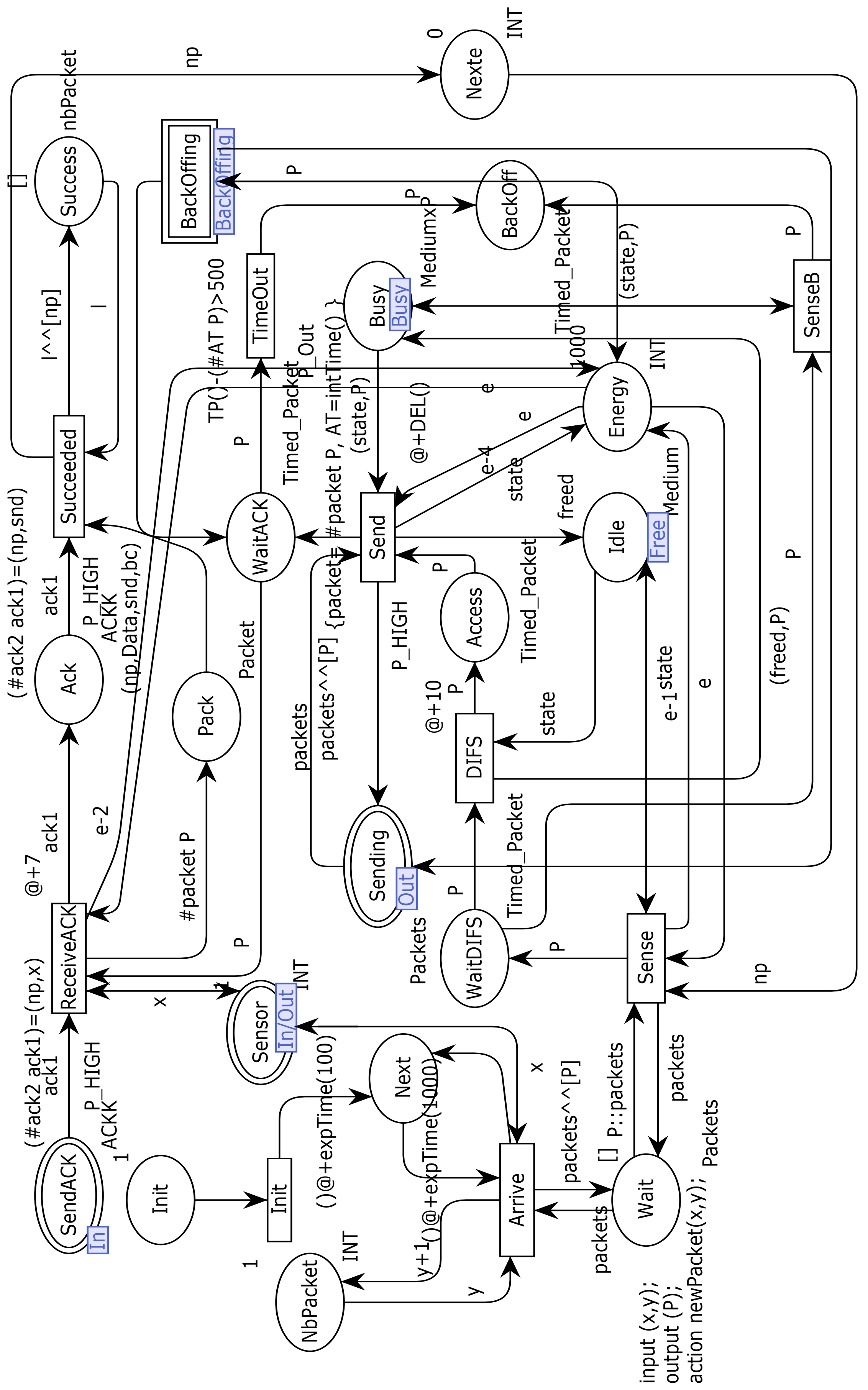


Figure II.2: HTCPN model for Sensor Node

Place	**Description**
Init	Initialisation
Next	Next Packet of the sensor node
NbPacket	Packet number
Wait	Packets waiting before sensing
WaitDIFS	Packets waiting DIFS time before sending
Access	Node accesses to the channel for sending
Idle	Channel idle
Busy	Channel busy
Sending	Sending packets from the sensors
Energy	Energy of sensor
WaitACK	The packet waiting for an ACK
BackOff	The packet enters in the backoff procedure
Success	Packets have a successful sending
Ack	Receiving acknowledgement
Nexte	Next sending

Table II.3: An overview table with all important places and their description

II.3.2.1 Backoff Procedure

The node can enter in this procedure if:

- The channel is not sensed idle for a DIFS time. In the sub-model, this situation is represented by a token in the place $WaitDIFS$ and a token in the place $Busy$. The transition $SenseB$ fires and provides a token in the place $BackOff$;
- A packet acknowledgement has not been arrived before a Timeout. In the sub-model, this situation is represented by the firing of transition $TimeOut$ that provides a token in the place $BackOff$.

Figure II.3 illustrates the sub-model for Backoff procedure. Components of this sub-model is detailed in table II.4.

II.3.3 HTCPN model for Base Station

In this subsection, a description of the HTCPNs sub-model for the base station is given. This model describes the behaviour of the base station, how a packet from the nodes is received and how an acknowledgement is sent. Figure II.4 illustrates these two events: receiving packet and sending acknowledgement. Table II.5 describes a set of places and transitions that are defined in the previous figure.

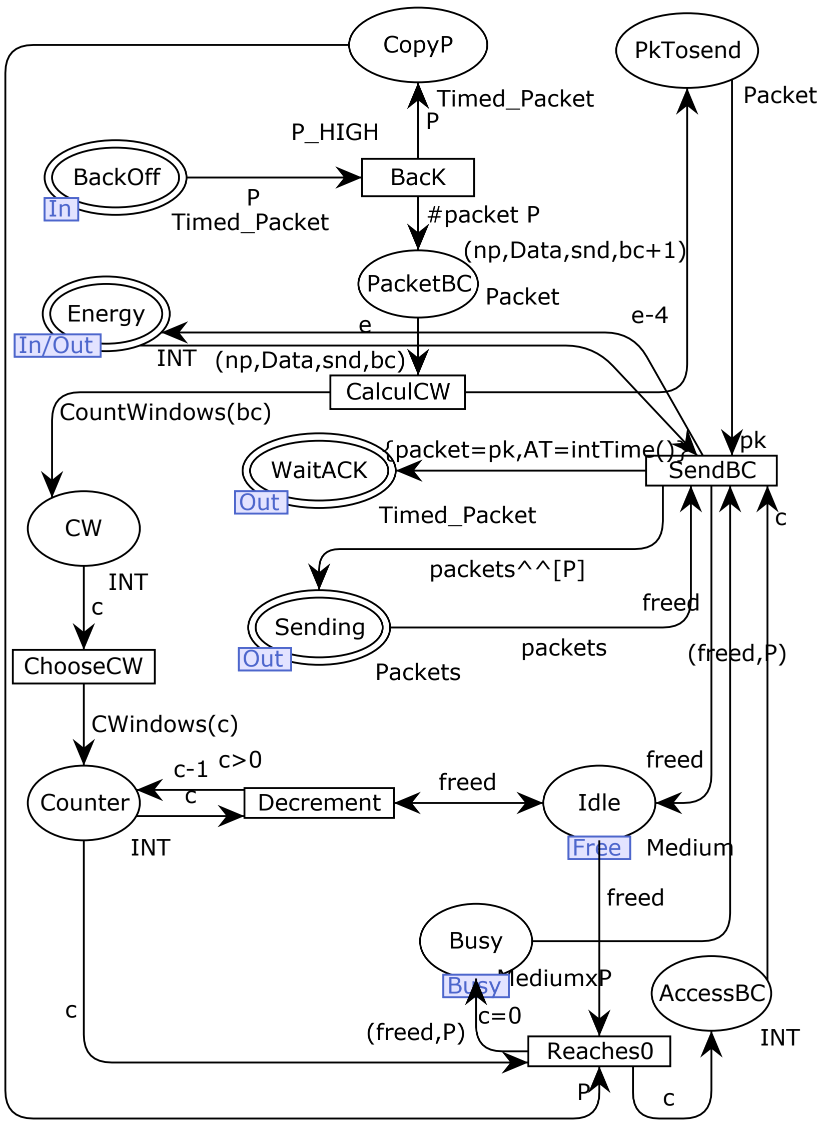


Figure II.3: HTCPN model for Backoffing

Element	**Type**	**Description**
BackOff	Place	The packet in Backoff procedure
CW	Place	Contention Windows
CopyP	Place	Contains copy of the packet
PacketBC	Place	Contains the Packet which is in Backoff procedure
PKTosend	Place	Contains the Packet to send
Counter	Place	Backoff Counter
AccessBC	Place	Packet accesses to the channel after Backoff procedure
Sending	Place	The sent packet
WaitACK	Place	Packet Waits for acknowledgement
Back	Transition	Gives the packet
CalculCW	Transition	Calculating the contention window
ChooseCW	Transition	Choosing a contention window
Reaches0	Transition	Backoff counter reaches 0
Decrement	Transition	Decrementing of backoff counter
SendBC	Transition	Sending the Packet

Table II.4: An overview table with all important places and transitions with their description in sub-model Backoff

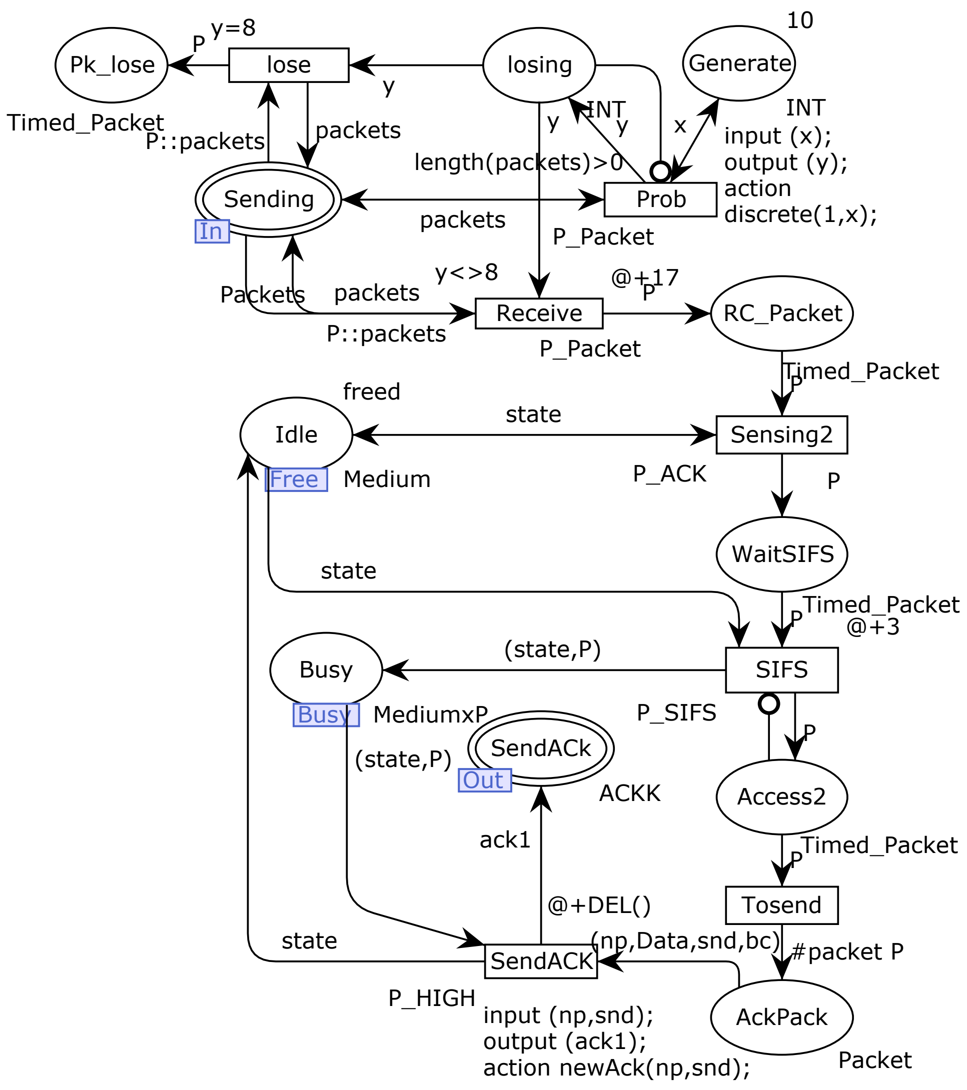


Figure II.4: HTCPN model for Base Station

II.4 Analysis

The performance analysis of CSMA/CA model is realized using the CPN-Tools. CPN-Tools allows the simulation of the model and the verification of several properties. In the case of the HTCPN model of CSMA/CA protocol, the initial marking of the model represents the initial state of the network. In our model, the communication between nodes and the base station through a star topology is highlighted . The simulated network is composed of five nodes and one base station. New packets arrive periodically in the network and each node will send the arrival packets to the base station through the shared medium. We search to assess our model by verifying some properties which influence the protocol. In CPN-Tools, the monitor mechanism is used to examine the markings and occurring binding elements, and to periodically extract information from the constructed

Element	**Type**	**Description**
RC_Packet	Place	Received Packets from the nodes
WaitSIFS	Place	Packet waiting for amount of time SIFS before sending a ACK
Access2	Place	The base station accesses to the channel for sending a ACK
WaitACK	Place	The packet waiting for an ACK
Generate	Place	Set of numbers
losing	Place	Number of lost packet
Receive	Transition	Receiving a packet from sensor nodes
Sensing2	Transition	Sensing the medium if it is free
SIFS	Transition	Blocking for SIFS time before sending ACK
SendACK	Transition	Sending an ACK
Prob	Transition	Giving a random number
lose	Transition	Losing packet

Table II.5: An overview table with all important places and transitions with their description in sub-model Base_Station

model. In our study of CSMA/CA MAC protocol, performance measures includes a set of properties to be verified. A number of monitors are defined to extract these performance measures.

The correctness verification of the design of the protocol can be performed with qualitative properties, and the performance evaluation can be done with quantitative properties.

II.4.1 Verification of Qualitative Properties

In this section, we are interested to formalise a set of properties in order to check them using CPN-Tools. The formalization is done using the Linear Temporal Logic (LTL) [76]. LTL formalism allows us to describe mathematically the relative order of events.

II.4.1.1 Properties Verification:

Three properties are identified that we consider the most important to be checked in the protocol. In the following, these properties are firstly informally described, then formalised in the LTL logic, and finally they are checked in the previous model of the protocol.

- *Packet sending:* For each sensor S_i, each data packet P_j sensed by S_i is transmitted to the base station BS. In the model of Figure II.2, this property is interpreted as: each token which is in place $Wait$ must reach place $Wait\_ACK$. Thus, the property is formalized as:

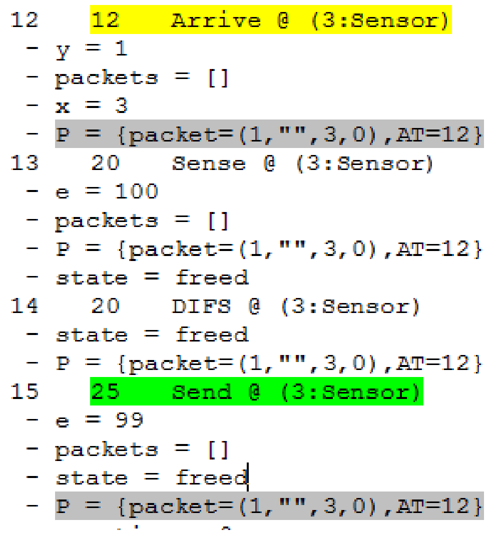
$$AG((Pi \in Wait) \implies XF(Pi \in Wait\_ACK)) \tag{II.1}$$

This property is checked using the simulation report, each packet in *Arrive* transition binding (i.e.,in $Wait$ place marking) must be in *Send* transition binding (i.e., in $Wait\_ACK$ place marking). For example, Figure II.5 illustrates that packet $P = (1, "", 3, 0)$ which is sensed by sensor 3, is sent to the base station.

- *Packet receiving:* For each sensor S_i, each data packet P_j sent by S_i is received by the base station BS. In the model of Figure II.2 and Figure II.4, this property is interpreted as: each token which is in place $Sending$ must reach place RC_Packet. Thus, the property is formalized as:

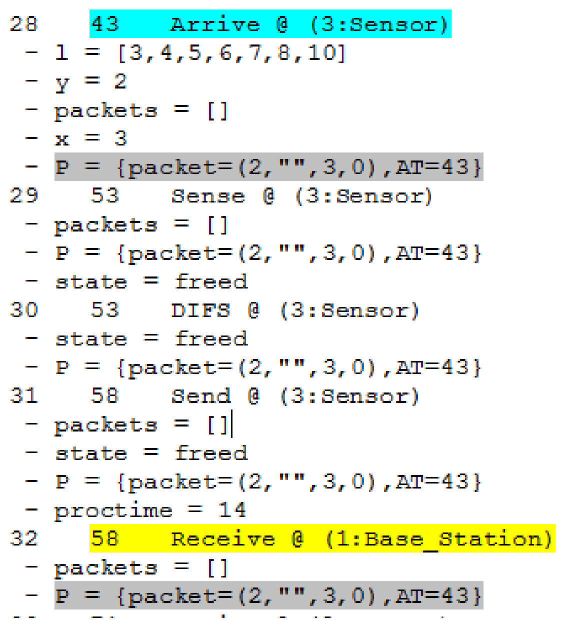
$$AG((Pi \in Sending) \implies XF(Pi \in RC\_Packet)) \tag{II.2}$$

This property is checked using the simulation report, each packet in *Send* transition binding (i.e., in $Sending$ place marking) must be in *Receive* transition binding (i.e., in RC_Packet place marking). For example, Figure II.6 illustrates that packet $P = (2, "", 3, 0)$ which is sent by sensor 3 it is received by the base station.



```
12    12    Arrive @ (3:Sensor)
 - y = 1
 - packets = []
 - x = 3
 - P = {packet=(1,"",3,0),AT=12}
13    20    Sense @ (3:Sensor)
 - e = 100
 - packets = []
 - P = {packet=(1,"",3,0),AT=12}
 - state = freed
14    20    DIFS @ (3:Sensor)
 - state = freed
 - P = {packet=(1,"",3,0),AT=12}
15    25    Send @ (3:Sensor)
 - e = 99
 - packets = []
 - state = freed
 - P = {packet=(1,"",3,0),AT=12}
```

Figure II.5: Packet sending property



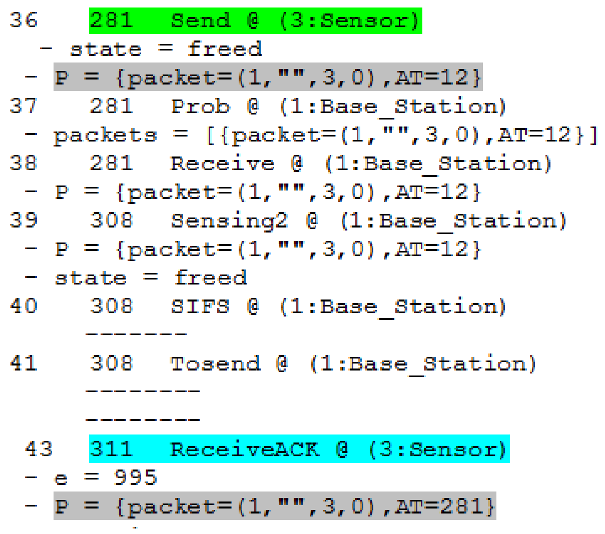
```
28    43    Arrive @ (3:Sensor)
 - l = [3,4,5,6,7,8,10]
 - y = 2
 - packets = []
 - x = 3
 - P = {packet=(2,"",3,0),AT=43}
29    53    Sense @ (3:Sensor)
 - packets = []
 - P = {packet=(2,"",3,0),AT=43}
 - state = freed
30    53    DIFS @ (3:Sensor)
 - state = freed
 - P = {packet=(2,"",3,0),AT=43}
31    58    Send @ (3:Sensor)
 - packets = []
 - state = freed
 - P = {packet=(2,"",3,0),AT=43}
 - proctime = 14
32    58    Receive @ (1:Base_Station)
 - packets = []
 - P = {packet=(2,"",3,0),AT=43}
```

Figure II.6: Packet receiving property

- *Packet acknowledgement:* For each received data packet P_i by base station BS, this last must send an acknowledgement for P_i to the sensor S_j. In the model of Figure II.2 and Figure II.4, this property is interpreted as: each token which is in place RC_Packet, must have its correspondent acknowledgement in place $SendACK$. Thus, the property is formalized as:

$$AG((P_i \in RC\_Packet) \Longrightarrow XF(Ack_i \in SendACK)) \tag{II.3}$$

This property is checked using the simulation report. Thus, each packet in *Send* transition binding (i.e., in RC_Packet place marking) must have an acknowledgement in *ReceiveACK* transition binding (i.e., in $SendACK$ place marking). For example, Figure II.7 illustrates that packet $P = (1, ””, 3, 0)$, which is sent by sensor 3, causes the reception of an ACK from the base station.



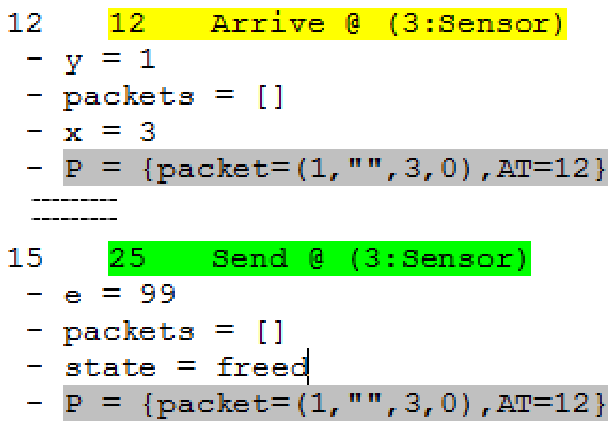
```
36    281   Send @ (3:Sensor)
   - state = freed
  - P = {packet=(1,"",3,0),AT=12}
37    281   Prob @ (1:Base_Station)
  - packets = [{packet=(1,"",3,0),AT=12}]
38    281   Receive @ (1:Base_Station)
  - P = {packet=(1,"",3,0),AT=12}
39    308   Sensing2 @ (1:Base_Station)
  - P = {packet=(1,"",3,0),AT=12}
  - state = freed
40    308   SIFS @ (1:Base_Station)
     -------
41    308   Tosend @ (1:Base_Station)
     --------
     --------
 43   311   ReceiveACK @ (3:Sensor)
  - e = 995
  - P = {packet=(1,"",3,0),AT=281}
```

Figure II.7: Packet ACK property

II.4.2 Verification of Quantitative Properties

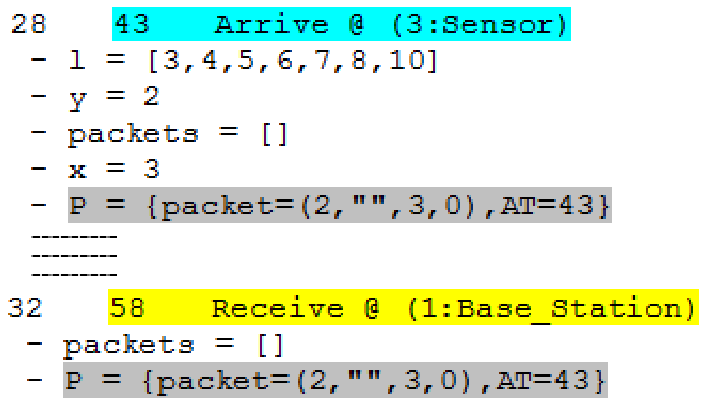
On the first hand, three temporal metrics are defined to evaluate the model as follow:

- **Metric 1:** *Waiting time* WT before a transmission. This performance metric is given by: the amount of time that a packet spend in the $Wait$ place before entering into the transmission process. Hence, at each sensor X we have to define for each packet P_i the time of bindings $< Arrive, Packets =?, P = P\_i, l =?, y =?, x = ?, ss = packet = P\_i, AT =? >$ and $< Send, Packets = list, P = P\_i, state = x >$ from paths in the reachability graph. Then, its waiting time is calculated by the subtraction of the second binding time from the first one (see Figure II.8). Finally, the waiting time average of all packets within the network is computed.



```
12    12    Arrive @ (3:Sensor)
 - y = 1
 - packets = []
 - x = 3
 - P = {packet=(1,"",3,0),AT=12}
---------
---------
15    25    Send @ (3:Sensor)
 - e = 99
 - packets = []
 - state = freed
 - P = {packet=(1,"",3,0),AT=12}
```

Figure II.8: Waiting Time metric



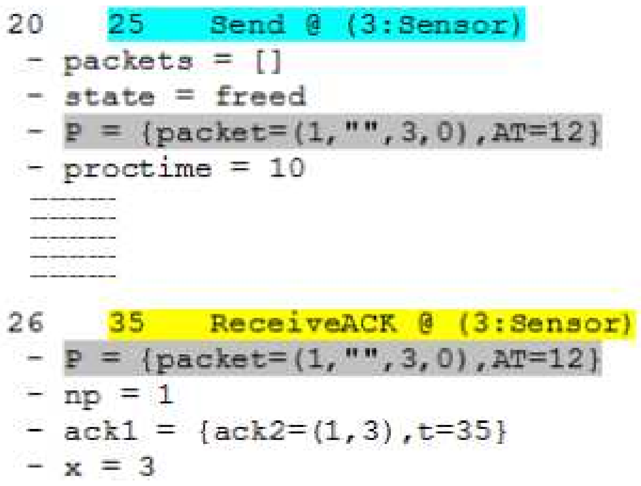
```
28    43    Arrive @ (3:Sensor)
 - l = [3,4,5,6,7,8,10]
 - y = 2
 - packets = []
 - x = 3
 - P = {packet=(2,"",3,0),AT=43}
 ---------
 ---------
 ---------
32    58    Receive @ (1:Base_Station)
 - packets = []
 - P = {packet=(2,"",3,0),AT=43}
```

Figure II.9: Delay Performance metric

- **Metric 2:** *The delay performance DP*, which is the duration from the moment that a packet enters the transmitter side to the moment that the packet is successfully received at the receiver side. This performance metric is given by: the amount of time that a packet entered in the place *Wait* until it reaches the place *RC_Packet* in the base station. Hence, at each sensor X we have to define for each packet P_i the time of bindings $< Arrive, Packets =?, P = P\_i, l =?, y =?, x =?, ss = \{packet = P\_i, AT =?\} >$ and $< Receive, Packets = list, P = P\_i >$ from paths in the reachability graph. Then, its delay performance is calculated by the subtraction of the second binding time from the first one (see Figure II.9). Finally, the delay performance average of all packets within the network is computed.

- **Metric 3:** *Node time waiting for an acknowledgement.* After the transmission of the data packet, the node must enter in a waiting time for a positive acknowledgement. This performance metric is given by: the amount of time that a packet entered in the place *WaitAck* until it receives an acknowledgement in the place *Ack*. Hence, at each sensor X we have to define for each packet P_i the time of bindings $< Send, Packets =?, P = P\_i >$ and $< ReceiveACK, P = P\_i, np =?, ack1 = ?, x =? >$ from paths in the reachability graph. Then, its waiting time for an acknowledgement is calculated by the subtraction of the second binding time from the first one (see Figure II.10). Finally, the average waiting time for an acknowledgement of all packets within the network is computed.

A set of data collector monitor is defined that extracts numerical data concerning the three former temporal metrics from our HTCPN model during simulations. The extracted files are used to plot the waiting time, delay performance and waiting time for an ACK versus number of simulations that are shown in Figures II.11, II.12 and II.13. Indeed, the average waiting time, performance delay and waiting time for an ACK that are shown in the performance report (Tables II.6, II.7 and II.8).



```
20    25    Send @ (3:Sensor)
  - packets = []
  - state = freed
  - P = {packet=(1,"",3,0),AT=12}
  - proctime = 10
  -------
  -------
  -------
  -------
  -------

26    35    ReceiveACK @ (3:Sensor)
  - P = {packet=(1,"",3,0),AT=12}
  - np = 1
  - ack1 = {ack2=(1,3),t=35}
  - x = 3
```

Figure II.10: Waiting Time for an Acknowledgement metric

Name	Count	Sum	Avg	Min	Max
Waiting_Time	100	2907	29.0700000	10	129

Table II.6: Waiting Time Average

Name	Count	Sum	Avg	Min	Max
Performance_Delay	99	11422	115.373737	10	510

Table II.7: Delay Performance Average

Name	Count	Sum	Avg	Min	Max
Waiting_Time_for_an_ACK	99	5304	53.575758	28	78

Table II.8: Waiting Time for an ACK Average

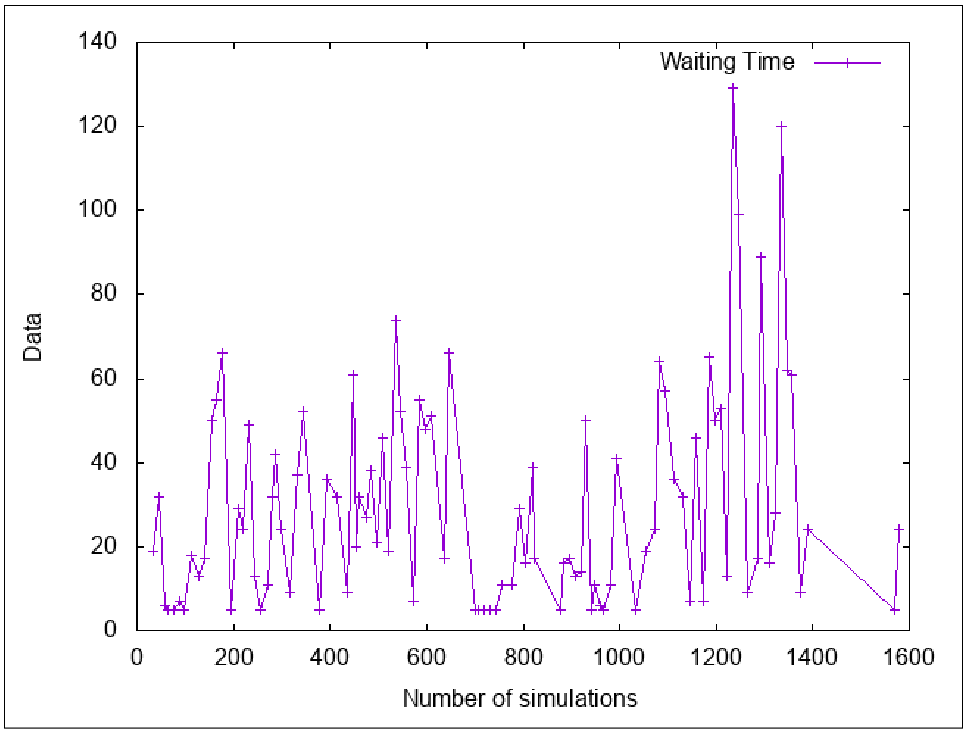


Figure II.11: Waiting Time metric by simulations

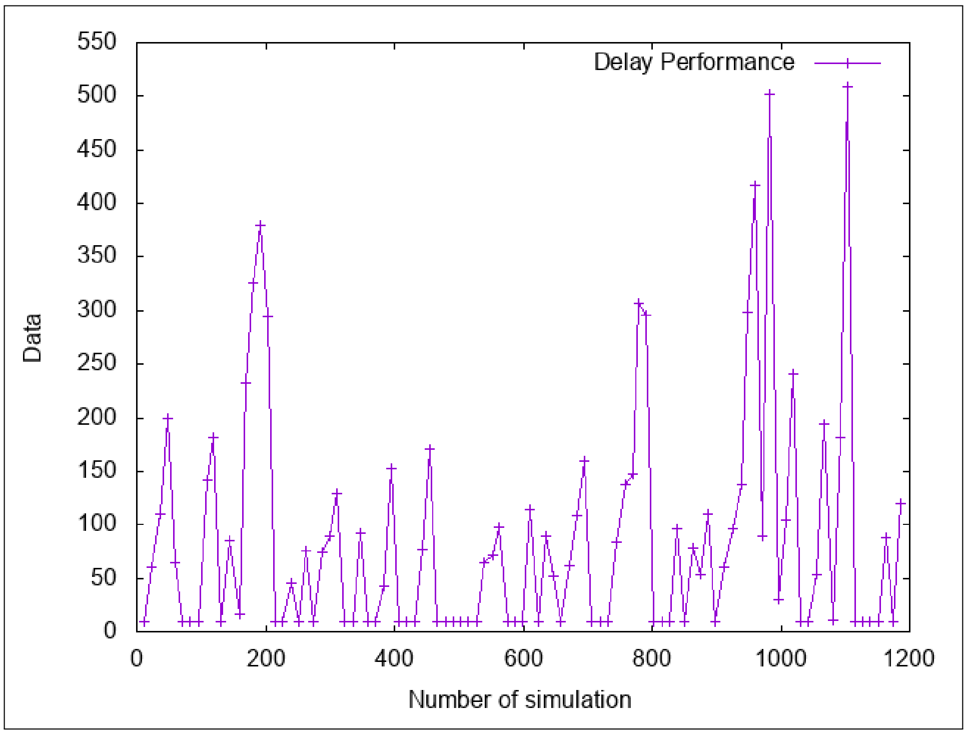


Figure II.12: Delay Performance metric by simulations

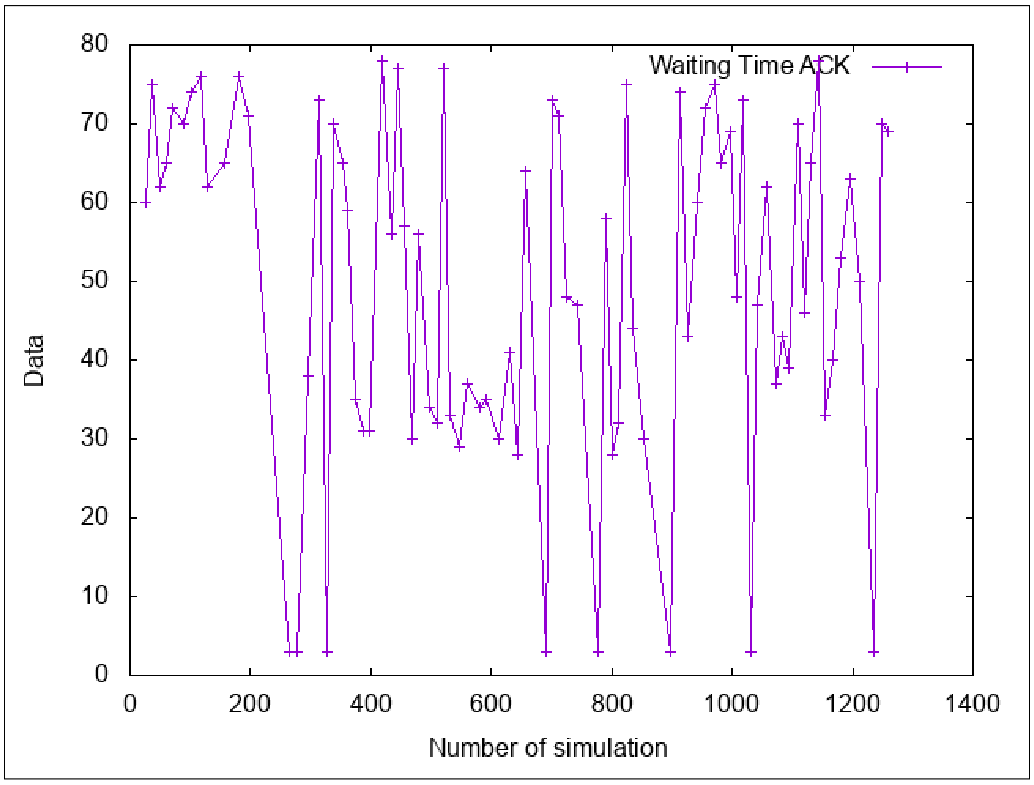


Figure II.13: Waiting Time for an ACK metric by simulations

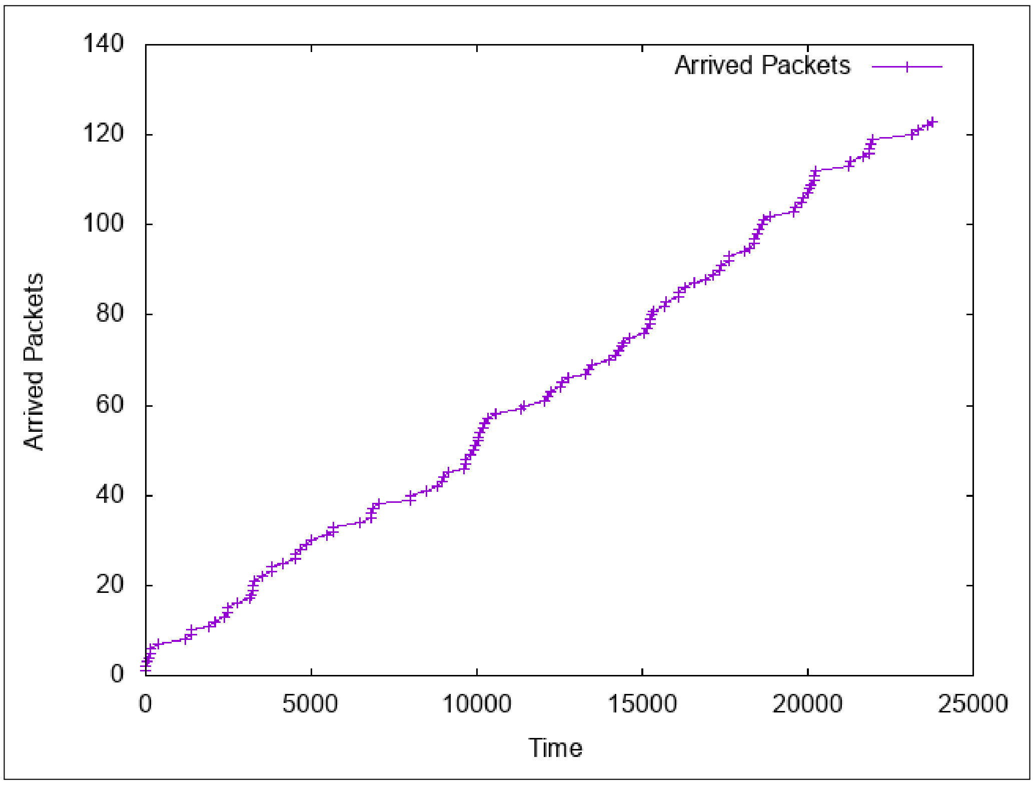


Figure II.14: Arrived packets in the network by time

Name	Count	Sum	Avg	Min	Max
Count_trans_occur_Sensor'Arrive_1	7	7	1.000000	1	1
Count_trans_occur_Sensor'Arrive_2	8	8	1.000000	1	1
Count_trans_occur_Sensor'Arrive_3	5	5	1.000000	1	1
Count_trans_occur_Sensor'Arrive_4	4	4	1.000000	1	1
Count_trans_occur_Sensor'Arrive_5	5	5	1.000000	1	1

Table II.9: The number of arrived packets by sensor

On the other hand, the following non-temporal metrics can also be considered to evaluate the protocol:

- **Metric 4:** *Arriving of packets*, in our CPN model we propose that the number of packets is not fixed beforehand. Therefore, we model the coming of packets by transition *Arrive* which guarantee a periodical arriving of packets in the network. Indeed, to calculate the number of arrived packets, a write in file monitor is defined. Then, the extracting file is used to plot the number of arriving packets versus time that is shown in figure II.14.

- **Metric 5:** *throughput at time* T, this metric is computed as the rate of the total number of packets transmitted by nodes to the total number of received acknowledgements at time T. To verify this property, a set of data collector monitors are defined which extract the number of arrived packets and the number of received acknowledgements by sensor. The result is shown in performance report (Table II.9 and Table II.10). At time $T = 5822$, the throughput is equal to the sum of received acknowledgements dividing by the sum of arrived packets:

$$Throughput = \frac{\sum receivedacknowledgements}{\sum arrivedpackets} = \frac{7+8+4+3+1}{7+8+5+4+5} = 0,79$$

- **Metric 6:** *Channel loading*, this metric is given by the percentage of time that the medium is busy. A data collector monitor is defined that examines the marking size of *Busy* place during simulations. This property is shown in the performance report (Table II.11).

II.5 Conclusion

A Hierarchical Time Petri Net based modelling that characterizes the behaviour of sensor nodes, is first presented. Both qualitative and quantitative verification are done

Name	Count	Avg	Min	Max
List_length_dc_Sensor'Success_1	9	3.688767	0	7
List_length_dc_Sensor'Success_2	10	4.236517	0	8
List_length_dc_Sensor'Success_3	6	2.449502	0	4
List_length_dc_Sensor'Success_4	5	2.253178	0	3
List_length_dc_Sensor'Success_5	3	0.928375	0	1

Table II.10: The number of received acknowledgements by sensor

Name	Count	Avg	Min	Max	First Time	Last Time
Marking_size_Sensor'Busy_4	8	04.00000	0	1	0	25

Table II.11: Channel loading average

using LTL logic to verify a set of qualitative properties and CPN-Tools to obtain numerical result concerning a set of performance metrics.

In this chapter, a formal approach is proposed to model and analyse the CSMA/CA protocol. This approach allows researchers to evaluate the performance of this protocol in WSNs. Using the proposed approach, we can simulate the behaviour of the network with random packets and whatever the number of sensors. Moreover, we can verify some properties which influence CSMA/CA protocol. Indeed, the main goal of the performance evaluation is to obtain a MAC protocol with high quality of service and efficient communications between sensor nodes and base station. Hierarchical CPNs allow us to construct a large network model as smaller models connected to each other. Furthermore, HTCPNs provide a timed aspect that facilitates the consideration of time constraints of the CSMA/CA protocol. To solve the problem of energy consumption in WSNs, a new approach for modelling and analysing CSMA/CA in EHWSNs will be presented in the following chapter.

Chapter III

Second Contribution: Performance Evaluation of CSMA/CA Protocol in Energy Harvesting Wireless Sensor Networks

III.1 Introduction

Sensor nodes must remain operational to collect and transfer data to a base station. Hence, WSNs have many challenges because of devices or sensor nodes which have specific characteristics of limited energy, memory, and computing power. Indeed, if the sensor node remains operational over considerable time then its energy will be drained. Therefore, sensor's battery must be autonomously rechargeable by gathering energy from the ambient environment (magnetic fields, mechanical vibrations, wind, solar power, temperature, variations,etc.). This concept is known as Energy Harvesting-based WSNs (EHWSNs) which was proposed in [33].

In this chapter, a modelling approach is proposed for the performance evaluation of CSMA/CA protocol in EHWSNs. Specifically, Hierarchical Timed Coloured Petri Nets is used to model this protocol. The analysis of this model has been realised by CPN-Tools. The generated models are simulated in various scenarios by varying the number of sensor nodes. The LTL logic is used to formalised qualitative properties. The quantitative properties are verified by exploiting the generated files during simulations.

The rest of the chapter is organized as follows. After introduction, Section III.2 presents related work overview. Section III.3 describes the proposed HTCPN model for the CSMA/CA protocol in EHWSNs. Section III.4 reports the proposed model performance analysis results. Finally, a conclusion is made in section III.5.

III.2 Related Work

Formal methods are considered as an important solution in the performance evaluation of protocols. This solution ensures the complying with standards and the verification of required properties. In this section, some related works are provided which have used coloured Petri nets as a formalism for the modelling and verification of wireless network protocols [36, 37, 38, 39, 40].

The authors in the work [36] have proposed a new model for the 1-wire protocol using timed coloured Petri net (TCPN). This model contains one master and several slaves. Exploiting the hierarchy capabilities of CPN Tools, their TCPN model has been constructed in two parts (pages). The authors used the first page to implement the master TCPN model and the second page to implement the slaves TCPN model. They have verified a set of qualitative and quantitative properties using the CPN-Tools. The authors have analysed the absence of deadlocks and the quantitative evaluation of the protocol. In [37], the authors used coloured Petri nets (CPNs) to present a modelling method for wireless network protocol. By exploiting the hierarchical features of CPN-Tools, they have applying this method to model and analyse IEEE 802.15.4 [13] wireless protocol. The authors have simulated the proposed CPN model in terms of delivery ratio, throughput, energy cost, and delay for three, six and nine nodes in a star topology. They compared their results with the ones obtained using the ns-2 simulation tool [73] to validate the proposed model. In [38], authors have exploited CPNs to perform the performance evaluation of the EQ-MAC protocol. They have verified two metrics, average delay and packet delivery ratio using CPNs in the modelling and performance evaluation. The GreatSPN [77] software system is used to validate the correctness of the developed models and evaluate the performance. The authors propose that the network is composed of 26 nodes including the base station. In [39], a Petri net model is presented for a medium access control protocol in WSNs named sensor medium access control protocol (S-MAC). The proposed model is based on Hierarchical Coloured Petri Nets (HCPNs). The authors considered a topology of a two-hop network with two sources and two sinks. They described the main part of the model hierarchy then sub-models have been described. For constructing and analysing the model, CPN-Tools is used. Some results of certain performance measures are evaluated include energy consumption and packet delivery delay. The authors considered that the collision can occur only in the control packets and data packets do not collide. They assumed that the size of the buffer is one packet. The work in [40] presents a formal model of Carrier Sense Multiple Access Collision Avoidance [11] (CSMA/CA) mechanism. Hierarchical coloured Petri nets formalism is used to construct this model. The authors have used CPN-Tools as modelling and simulation environment. The proposed HCPN model of CSMA/CA mechanism is composed of three pages. The authors have verified two qualitative properties: boundness and liveness.

Besides the use of coloured Petri nets in the above works, there are some other works which have tackled the modelling and verification of CSMA/CA protocol in wireless networks by the using other formalisms. Works [66, 67, 71, 72, 58, 59, 60, 61] proposed approaches to model and to analyse CSMA/CA MAC protocol in wireless networks. The authors in [66] and in [67] have used Stochastic Petri Nets [78] to model and to analyse the Distributed Coordination Function (DCF), which is the fundamental contention based access mechanism of 802.11 standard [12] for wireless local area networks(WLANs). In work [71], the authors have proposed a detailed 802.11b model based on Object-oriented Petri Nets (OOPNs) [79] that precise backoff procedure and time synchronization. Formal approaches are presented by [58, 59, 60, 61] where authors used timed automata (TA) [80] and UPPAAL [81] for modelling and analysing the CSMA/CA protocol.

In fact, all the above works do not deal with all aspects of CSMA/CA protocol such as network topology, number of nodes, node behaviour, arriving of packets, and the Backoff procedure. Moreover, they do not verify all important properties such as waiting time, delay performance, waiting time for an acknowledgement, throughput and harvesting energy feature. In the current chapter, a new approach is proposed for modelling and analysing CSMA/CA protocol in harvesting wireless sensor networks. Although, the proposed model allows to perform the qualitative/quantitative verification of the protocol. The arriving of packets is considered random in the network and the number of nodes is variable. Indeed, a set of properties is evaluated using the capabilities of hierarchical timed coloured Petri nets which offer the performance evaluation in CPN-Tools.

III.3 Modelling CSMA/CA using Hierarchical Timed-CPNs

The modelling process is the first step in which the designer must investigate a set of events that occur in a system, the events dynamic can easily modify the state of the system.

In medium access control using the CSMA/CA protocol (our case study), four major events are identified. These events modify the states defined in the system. The events are: Sending a data packet, receiving acknowledgement, receiving a data packet, and sending acknowledgement. In the proposed HTCPN model, these events are modelled by transitions. In CSMA/CA protocol, the occurred events are constrained by several conditions which are presented by a set of pre-conditions and post-conditions in the proposed HTCPN model. It is important to mention that when pre-conditions are satisfied the post-conditions will be also satisfied. The pre-conditions are modelled by input-places, expressions on input-arcs, and associated guards. The post-conditions are modelled by output-places and expressions on output-arcs.

In this section, the proposed HTCPN model for CSMA/CA protocol in EHWSNs is described. The global model is constructed taking the advantage of the hierarchical capabilities of HCPNs [74]. Hierarchical CPNs facilitate the construction of large models by the possibility of dividing the model into smaller models connected to each other using well-defined interfaces (i.e. substitution transitions and interface places). Based on this, more details can be added to the proposed model. In the proposed model, the syntax of CPN-Tools specification language is used. The CPN-Tools represents an implementation of the Hierarchical Timed CPNs, this tool allows the graphical representation of HTCPN models and their analysis. In the following a description of the abstract model is presented as well as a presentation of the sub-models.

III.3.1 The Global HTCPN Model

Using the hierarchical principle in HTCPNs, the global model can be constructed as an abstract model as shown in Figure III.1. In the abstract model, the sensors of the network and the base station are represented as hyper-transitions. The interaction among these components of the network is modelled through interfaces places. This model gives an overview of the mechanism used by CSMA/CA MAC protocol to access the channel for sending a data packet, where all the events are modelled. The folding technique [75] is used to construct a compact and scalable model. In the current contribution, the proposed model is described as follow: it includes five sensors and one base station. A sensor node can send a data packet to the base station and the last one can respond with an acknowledgement. The communication is made through a shared channel. The model

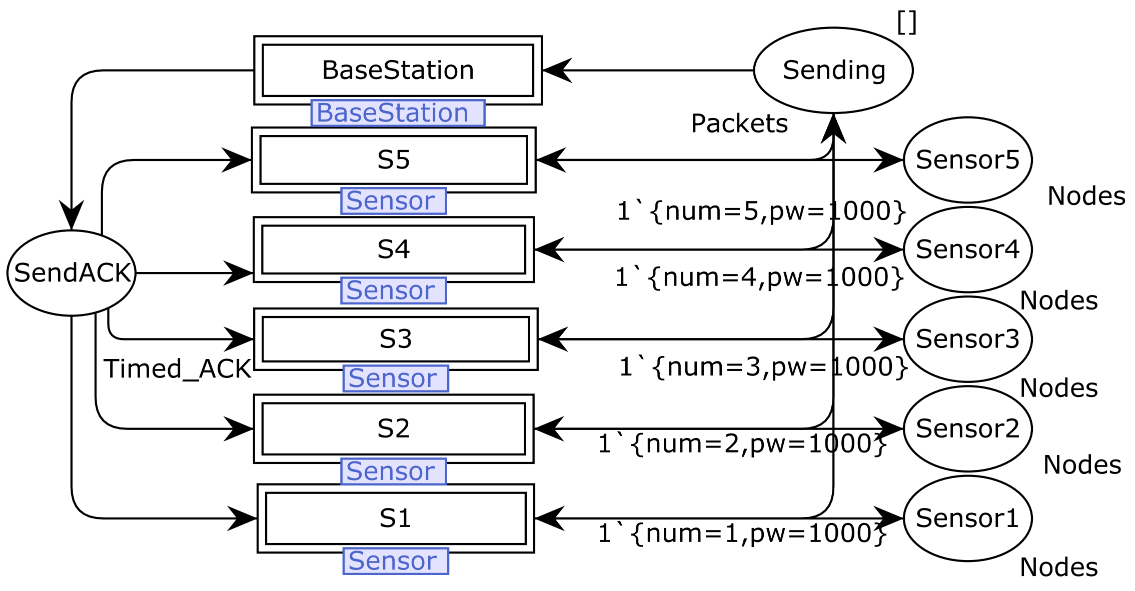


Figure III.1: HTCPN abstract model for CSMA/CA protocol

hierarchy is composed of six substitution transitions representing network sensors and the base station as depicted in Figure III.1. Transitions: $S1$; $S2$; $S3$; $S4$; $S5$ and $BaseStation$ are hyper-transitions and each transition represents one of the former sub-models. Since all the sensor nodes in the network have the same features, they are modelled with one sub-model called $Sensor$, and five instances of this sub-model (sub-page) are created: $S1$; $S2$; $S3$; $S4$; and $S5$. These substitution transitions represent the events of sending data and receiving acknowledgement realised by sensor nodes. Receiving data and sending acknowledgement events are represented by $BaseStation$ substitution transition.

In our proposed Hierarchical Timed CPN model, a set of types is defined. Table III.1 illustrates these types and their description.

III.3.2 CPN model for a sensor node

In this subsection, the HTCPN sub-model for the sensor nodes is described. The network nodes sub-model includes the processes of: sending data, receiving acknowledgement(ACK), harvesting energy process and the Backoff procedure. Figure III.2 depicts these events. Table III.2 and Table III.3 present the most important transitions and places, respectively, that are used in Figure III.2 and their description. HTCPN allows using the same Petri net structure to model multiple nodes, thus one sub-model is created to illustrates the behaviour of one sensor node called: $Sensor$ which can be instantiated for the other nodes.

Transition $Active$ is enabled when a token is in $Sensor$ place and the energy of the sensor is equal to $1000\ units$. A token is put in $Sensors$ place in the queue of the list of active sensors, and a token in $ASensor$ place. Transition $Discharge$ is enabled when a

Element	**Type**	**Description**
Nodes	record $\{num, pw\}$	This color set models a *sensor* as a record consisting of two fields: num, an $integer$ which contains the number of sensor and pw, an $integer$ which contains the energy of sensor.
Packet	product $(Npk \times Data \times Snd \times BC)$	This type models the set of packets. A packet is specified by: Npk, Snd, BC are $integers$ which contain the packet number, the sender number, the Backoff counter respectively and $Data$ is a $string$ which represents the data of packet.
Timed_Packet	record {Packet, AT}	This color set models a *packet* as a record consisting of two fields: $Packet$ contains the data packet and AT contains its sending time.
ACK	product ($Npk \times snd$)	This type models the set of acknowledgements. An acknowledgement is specified by: Npk, an $Integer$ which contains the packet number and snd, an $integer$ which represents the sender number.
Timed_ACK	record {ACK, AT}	This color set models an *acknowledgement* as a record consisting of two fields: ACK and AT. AT contains the time of ACK sending.
Medium	with $free$ timed	This timed small color set models the medium when it is free.
MediumxP	product (Medium $\times$ Timed_Packet)	This timed color set models the medium when it is busy.

Table III.1: Token types' table.

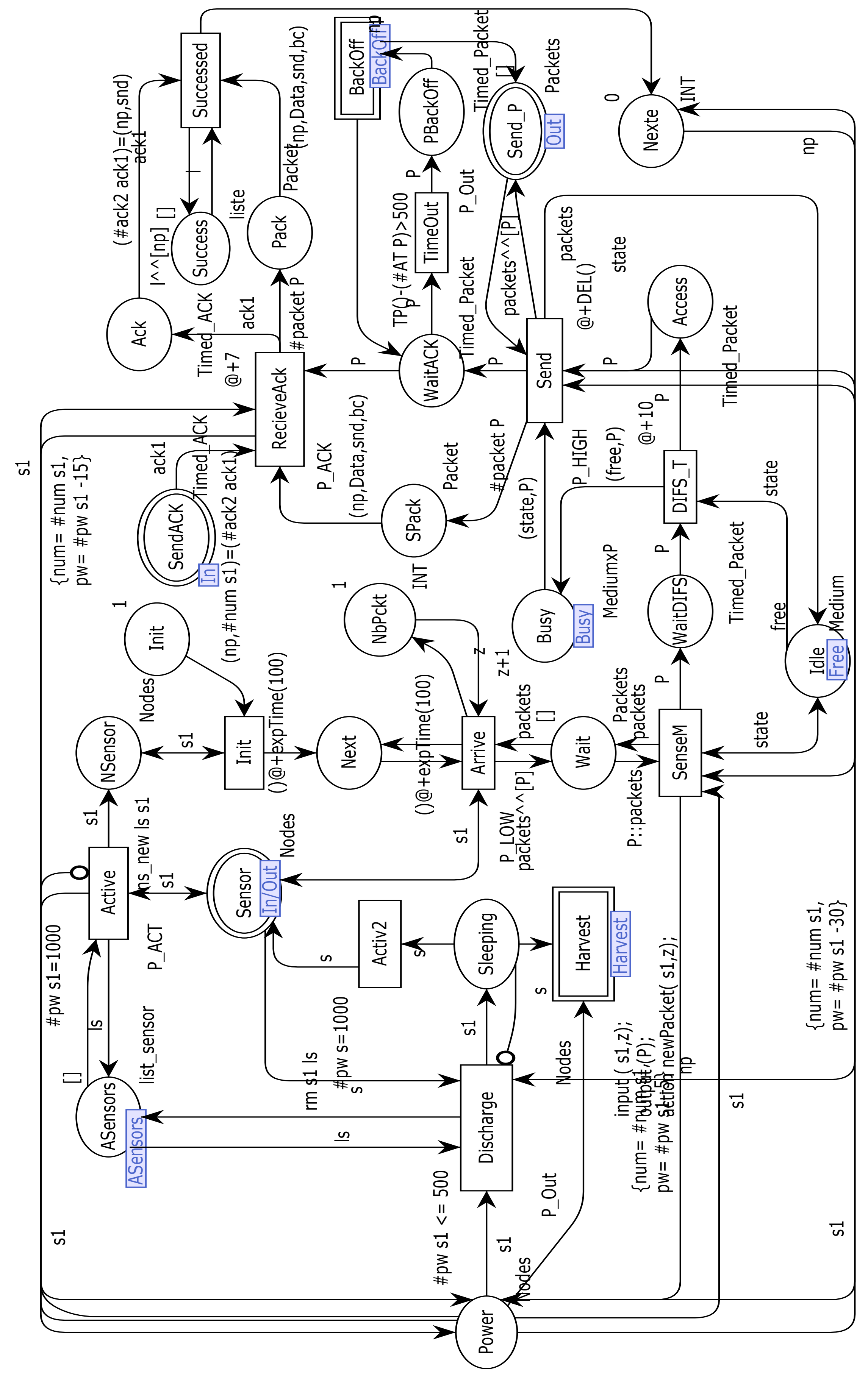


Figure III.2: HTCPN model for a sensor node

Transition	**Description**
Active	Activating the sensor node
Discharge	Passing to the harvest process
Activee	Activating the sensor node after the harvest process
Harvest	Harvesting energy from environment
Init	Initiating the first arriving packet
Arrive	Arriving of a packet
SenseM	Sensing the medium if it is free
DIFS_ T	Blocking the packet for DIFS time before reaching the channel
Send	Sending a packet from the sensor node to the base station
SenseB	Sensing the medium in Backoff procedure
TimeOut	The acknowledgement timeout
ReceiveACK	Receiving an ACK from the base station
Successed	Sending successes
Backoffing	The node coming to the Backoff procedure

Table III.2: An overview table with all important transitions and their description

Place	**Type**	**Description**
Power	Node	Sensors whose their power must be reduced
ASensors	list_sensor	List of active sensors
NSensor	Integer	Sensor number
Sleeping	Node	Sensor in harvest process
Next	Integer	Next packet of the sensor node
NbPacket	Integer	Packet number
Wait	Packet	Packets waiting before sensing
WaitDIFS	Timed_Packet	Packets waiting for DIFS time before sending
Access	Packet	Arrived packet to the channel for sending
Idle	Medium	Represents that the channel is free
Busy	MediumxP	Represents that the channel is busy
Send_ P	Packet	Packets to be sent to the base station
WaitACK	Packet	The waiting packet for an ACK
PBackOff	Packet	The coming packet to the backoff procedure
Success	List of integer	Number of Packets with successful sending
Ack	ACK	Received acknowledgement

Table III.3: An overview table with all important places and their description

token is in $Power$ place and the energy of the sensor is equal or less than 500 $units$. A token is put in $Sleeping$ place. A token in $Sleeping$ place enables $Harvest$ transition. If a token is in $Sleeping$ place and if the energy of the sensor is equal to 1000 $units$, then transition $Activ2$ fires and a token is put in $Sensor$ place. A token in *Next* place is used to determine when new packets arrive. The time stamp of the token in the place will determine when *Arrive* transition can occur. Transition *Init* is the only transition that is enabled in the initial marking. This transition is used to put a token in *Next* place. The time inscription $@ + expTime(100)$ on the arc that links transition *Init* and *Next* place is used to create a time stamp of *Next* place tokens. The $expTime$ function will return a value from the exponential random distribution function. This time stamp is used for ensure that the first packet will never arrive at time 0 in different simulations. Transition $Arrive$ will occur when the time stamp of the token in $Next$ place is equal or greater than the current model time. When the last transition occurs, a new packet is created and bounded it to the variable P through the code segment of the transition. Then, the new packet is added to the end of the queue of $packets$, as determined by the inscription on the arc that links $Arrive$ transition and $Wait$ place. The time inscription $@ + expTime(100)$ on the arc that links $Next$ place and $Arrive$ transition is used to determine the time at which the next packet will arrive. The inter-arrival times are exponentially distributed with a mean of 100 time units. In the sub-model, place $NbPacket$ contains the counter of packets, each enabling of $Arrive$ transition will increment this counter and a taken is put in $Wait$ place. A token in place $Wait$ means that a packet is waiting for sending and it will sense a medium, by enabling of $SenseM$ transition. If the medium is free, the packet enters in the waiting for a $DIFS$ time in $WaitDIFS$ place, otherwise, it stays in the waiting until the medium will be free. A token is in $Idle$ place means that the medium is free. If $DIFS\_T$ transition is enabled, it puts a token in place $Access$ and a token in place $Busy$. A token in $Access$ means that the packet will be sent and transition $Send$ will be enabled. When $Send$ is enabled, a token is put in $WaitACK$ place and in $Sending$ fusion place. If acknowledgement arrives in time, then $ReceiveACK$ transition can occur and will put a token in Ack place, otherwise the transition $TimeOut$ will occur and puts a token in $BackOff$ place. A token in $Success$ place means that this packet has been sent successfully.

III.3.2.1 Harvest Process

In this sub-model, a token in $Sleeping$ place means that the sensor is in the sleep mode. Transition $Harvest$ fires while the energy of sensor has not reached 1000 $units$. Transition $Charged$ is enabled when the energy of the sensor reaches 1000 $units$. A token is put in $Power$ place and $Sensors$ place, in the queue of the list of active sensors. Figure III.3 illustrates the sub-model of Harvest process. The components of this sub-model is detailed in Table III.4.

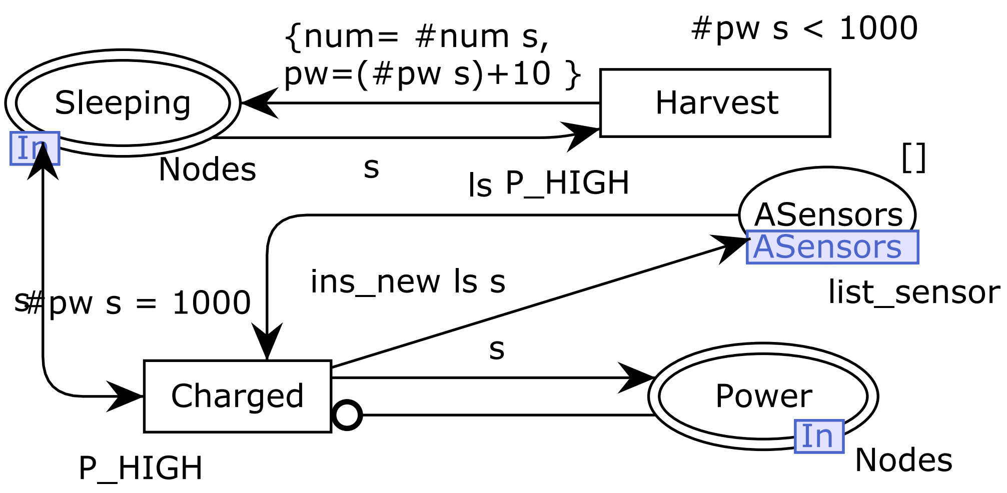


Figure III.3: HTCPN model for Harvesting energy process

Element	Place/Transition	Description
Sleeping	Place	Sensor in sleep period
Power	Place	Sensor that its power reaches 1000 *unit*
ASensors	Place	List of active sensors
Harvest	Transition	The sensor harvests energy
Charged	Transition	The sensor is charged

Table III.4: An overview table with all important places and transitions with their description in sub-model Harvest

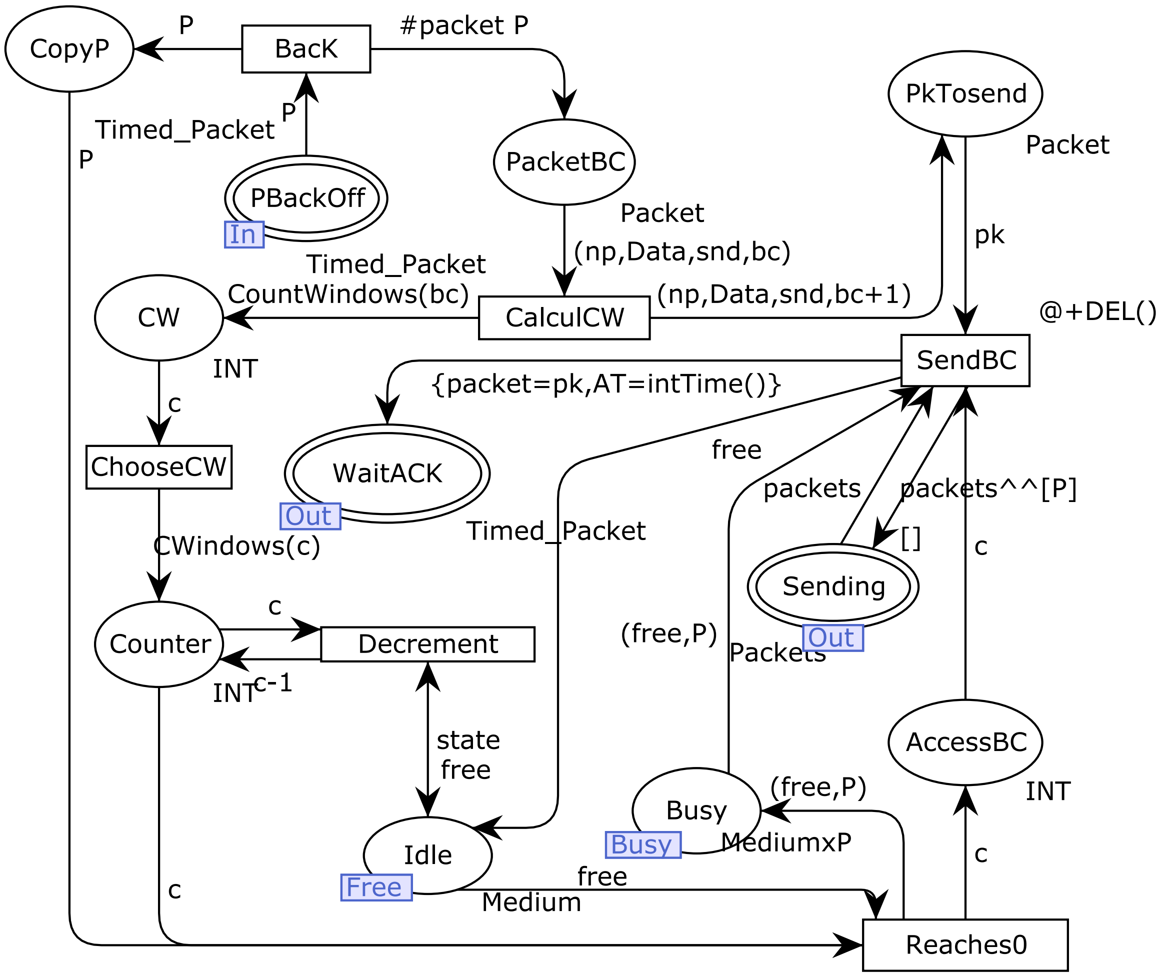


Figure III.4: HTCPN model for Backoff

III.3.2.2 Backoff Procedure

A node can enter in this procedure if:

- The channel is not sensed idle for a DIFS time. In the sub-model, that is represented by a token in $WaitDIFS$ place and a token in $Busy$ place. Transition $SenseB$ fires and a token is placed in $BackOff$ place;
- A packet acknowledgement has not been arrived before a Timeout. Transition $TimeOut$ has been fired and a token is put in $BackOff$ place.

Figure III.4 illustrates the sub-model of the Backoff procedure. The components of this sub-model is detailed in Table III.5.

A token in place $BackOff$ means that the node enters in the BackOff procedure, and transition $back$ fires. The firing of $back$ transition puts a token in $CopyP$ and a token in $PacketBC$. Transition $CountCW$ can occur and calculates the contention window CW. Transition $ChooseCW$ fires and puts a token in place $Counter$ which represents the backoff counter. If the medium is idle this counter will be decremented with $decrement$

Element	**Place/Transition**	**Description**
BackOff	Place	Packet in Backoff procedure
CW	Place	Contention Windows
CopyP	Place	Contains copy of the packet
PacketBC	Place	Contains the Packet which is in Backoff procedure
PKTosend	Place	Contains the Packet to send
Counter	Place	Backoff Counter
AccessBC	Place	Packet accesses to the channel after Backoff procedure
Sending	Place	The sent packet
WaitACK	Place	Packet Waits for acknowledgement
Back	Transition	Gives the packet
CalculCW	Transition	Calculating the contention window
ChooseCW	Transition	Choosing a contention window.
Reaches0	Transition	Backoff counter reaches 0.
Decrement	Transition	Decrementing of backoff counter.
SendBC	Transition	Sending the Packet.

Table III.5: An overview table with all important places and transitions with their description in sub-model Backoff

transition until it reaches 0. When it reaches 0 the packet can be transmitted by firing $SendBC$ which will put a token in place $Sending$ and a token in place $WaitACK$.

III.3.3 HTCPN model for Base Station

In this subsection, the HTCPN sub-model for the base station is described. This model demonstrates the behaviour of the base station, how to receive a packet from the nodes and how to send back an acknowledgement. Figure III.5 illustrates these two events: receiving packet and sending acknowledgement. A set of places and transitions is defined and described in Table III.6.

In this sub-model, a packet in $Sending$ place means that a node has sent a packet to the base station. Transition $Receive$ fires and puts a token in RC_Packet place. The base station sensed for the medium when it will be idle for a SIFS time by firing $Sensing2$ transition that puts a token in $WaitSIFS$ place. After that, the base station accesses to the channel and sends an acknowledgement by firing $SendACK$ transition.
Transition $lose$ can handle the lost of the packet if occurred and a probability of losing packets in this case is equal to 0.1.

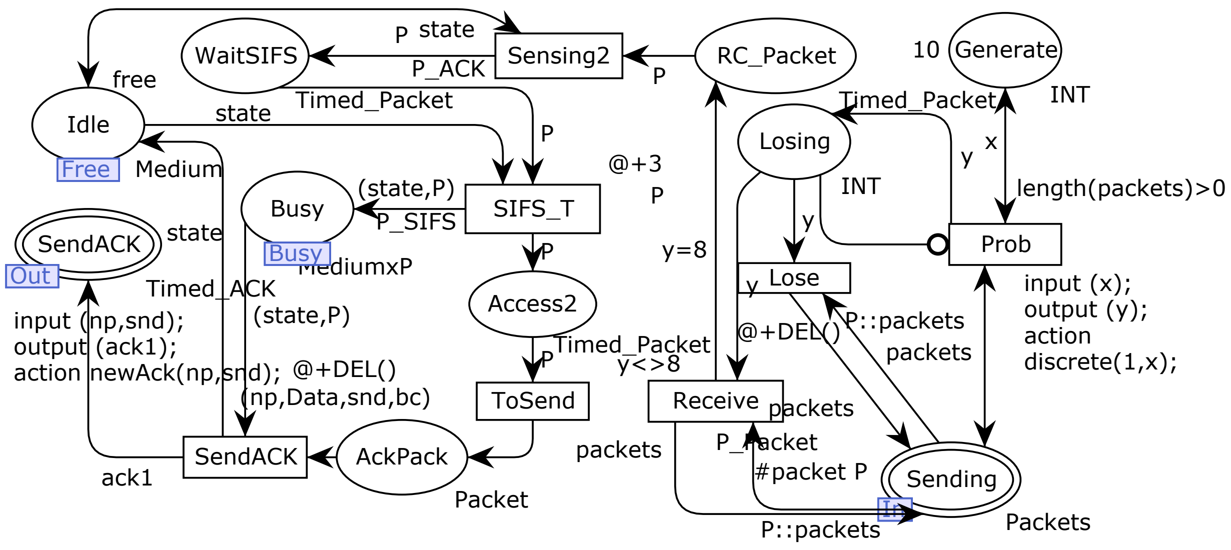


Figure III.5: HTCPN model for Base Station

Element	Place/Transition	Description
RC_Packet	Place	Received Packets from nodes
WaitSIFS	Place	Waiting packet for amount of time SIFS before sending an ACK
Access2	Place	The base station accesses to the channel for sending an ACK
WaitACK	Place	The packet waits for an ACK
Receive	Transition	Receiving a packet from sensor nodes
Sensing2	Transition	Sensing the medium if it is free
SIFS_T	Transition	Blocking for SIFS time before sending an ACK
SendACK	Transition	Sending an ACK

Table III.6: An overview table with all important places and transitions with their description in sub-model Base_Station

III.4 Analysis

Performance analysis using HTCPNs relies on simulation to calculate performance measures for the model. During a simulation of a HTCPN, the CPN-Tools generates quite a bit of quantitative information about the performance of a system, such as queue length, response time, throughput, etc. These information can be extracted from the model by examining the markings of places and the occurring binding elements during simulations.

The initial marking of the model represents the initial state of the CSMA/CA protocol. In our model, we are interested in the communication between nodes and the base station through a star topology. The simulated network is composed of five nodes and one base station. New packets arrive periodically in the network. Each node will send its arrival packets to the base station through the shared medium. We aim to assess our model by verifying some properties which influence protocol.

To examine the markings and occurring binding elements, and to periodically extract information from the constructed model, CPN Tools uses the monitor mechanism. A monitor is used to observe, inspect, control, or modify a simulation of a CPN model and the desired measures. In our study of CSMA/CA MAC protocol, performance measures include a set of properties that will be verified. A number of monitors to extract the performance measures are defined.

The correctness verification of the design of the protocol can be performed with qualitative properties, and the performance evaluation can be done with quantitative properties.

III.4.1 Verification of qualitative properties

In this section we are interested to formalize a set of properties in order to check them using CPN-Tools. The formalization is done in the Linear Temporal Logic(LTL). LTL formalism allows us to describe mathematically the relative order of events.

Four properties have been identified that are considered the most important to be checked in the protocol. In the following, these properties are described, then formalized in the LTL logic, and finally these properties are checked in the previous model of the protocol.

- *Packet sending:*

  For each sensor S_i, each data packet P_j sensed by S_i is transmitted to the base station BS. In the model of Figure III.2, this property is interpreted as: each token which is in place $Wait$ must reach place $Wait\_ACK$. Thus, the property is formalized as:

  $$AG((Pi \in Wait) \Longrightarrow XF(Pi \in Wait\_ACK)) \tag{III.1}$$

  This property is checked using of the simulation report, each packet in *Arrive* transition binding (i.e.,in $Wait$ place marking) must be in *Send* transition binding (i.e., in $Wait\_ACK$ place marking). For example, Figure III.6 illustrates that packet $P = (3, "", 5, 0)$ which is sensed by sensor 5, is sent to the base station.

- *Packet receiving:* For each sensor S_i, each data packet P_j sent by S_i is received by the base station BS. In the model of Figure III.2 and Figure III.5, this property is interpreted as: each token which is in place $Sending$ must reach place RC_Packet. Thus, the property is formalized as:

$$AG((Pi \in Sending) \implies XF(Pi \in RC\_Packet)) \tag{III.2}$$

This property is checked using of the simulation report, each packet in *Send* transition binding (i.e., in $Sending$ place marking) must be in *Receive* transition binding (i.e., in RC_Packet place marking). For example, Figure III.7 illustrates that packet $P = (5, "", 3, 0)$ which is sent by sensor 5 it is received by the base station.

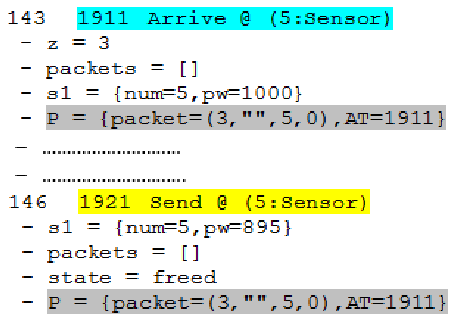

```
143  1911 Arrive @ (5:Sensor)
 - z = 3
 - packets = []
 - s1 = {num=5,pw=1000}
 - P = {packet=(3,"",5,0),AT=1911}
 - ...........................
 - ...........................
146  1921 Send @ (5:Sensor)
 - s1 = {num=5,pw=895}
 - packets = []
 - state = freed
 - P = {packet=(3,"",5,0),AT=1911}
```

Figure III.6: Packet sending property

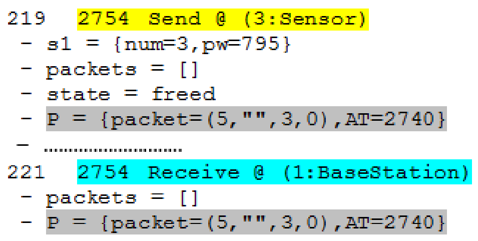
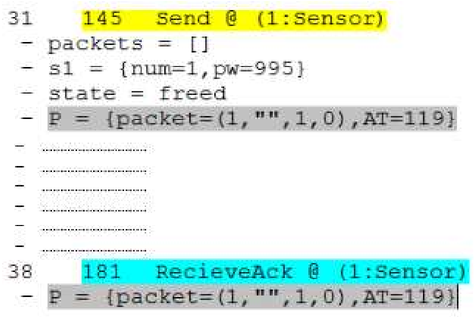

```
219  2754 Send @ (3:Sensor)
 - s1 = {num=3,pw=795}
 - packets = []
 - state = freed
 - P = {packet=(5,"",3,0),AT=2740}
 - ...........................
221  2754 Receive @ (1:BaseStation)
 - packets = []
 - P = {packet=(5,"",3,0),AT=2740}
```

Figure III.7: Packet receiving property

- *Packet acknowledgement:* For each received data packet P_i by the base station BS, this last must send an correspondent acknowledgement for P_i to the sensor S_j. In the model of Figure III.2 and Figure III.5, this property is interpreted as: each token which is in place RC_Packet, we must find its correspondent acknowledgement in place $SendACK$. Thus, the property is formalized as:

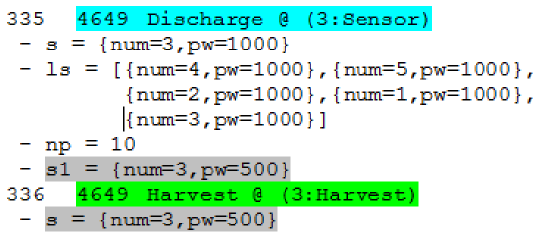
$$AG((P_i \in RC\_Packet) \implies XF(Ack_i \in SendACK)) \tag{III.3}$$

This property is checked using of the simulation report, each packet in *Send* transition binding (i.e., in RC_Packet place marking) must have an acknowledgement in *ReceiveACK* transition binding (i.e., in $SendACK$ place marking). For example, Figure III.8 illustrates for that packet $P = (5, "", 3, 0)$ which is sent by sensor 5, this last received ACK from the base station.



```
31    145   Send @ (1:Sensor)
 - packets = []
 - s1 = {num=1,pw=995}
 - state = freed
 - P = {packet=(1,"",1,0),AT=119}
 - ..........
 - ..........
 - ..........
 - ..........
 - ..........
 - ..........
38    181   RecieveAck @ (1:Sensor)
 - P = {packet=(1,"",1,0),AT=119}
```

Figure III.8: Packet ACK property



```
335   4649 Discharge @ (3:Sensor)
 - s = {num=3,pw=1000}
 - ls = [{num=4,pw=1000},{num=5,pw=1000},
         {num=2,pw=1000},{num=1,pw=1000},
         {num=3,pw=1000}]
 - np = 10
 - s1 = {num=3,pw=500}
336   4649 Harvest @ (3:Harvest)
 - s = {num=3,pw=500}
```

Figure III.9: Harvesting property

- *Harvesting feature:* each sensor S_i which energy is equal or less than 500 $unit$, must enter into the harvesting process. In the model of Figure III.2 and Figure III.3, this property is interpreted as: each token which is in place $Power$ and the energy of sensor is less than 500 $unit$, must reach place $Sleeping$. Thus, the property is formalized as:

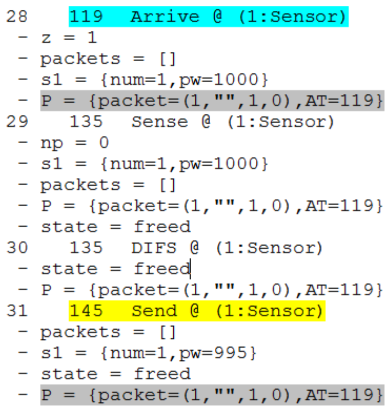
$$AG((S_i \in Power) \implies XF(S_i \in Sleeping)) \tag{III.4}$$

This property is checked using of the simulation report, each packet in *Discharge* transition binding (i.e., in $Power$ place marking) must be in *Harvest* transition binding (i.e., in $Sleeping$ place marking). For example, Figure III.9 illustrates that sensor $s1 = \{num = 3, pw = 500\}$ is entered in the Harvest process.

III.4.2 Verification of quantitative properties

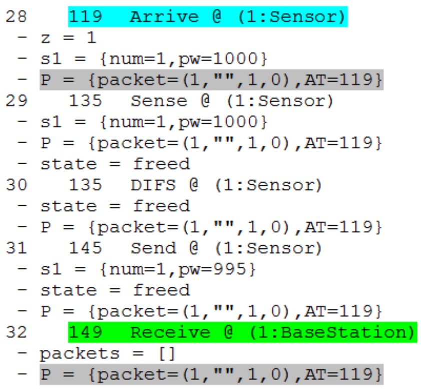
On the first hand, three temporal metrics are defined to evaluate the proposed model as follow:

- **Metric 1:** Waiting time WT before a transmission. This metric is given by: the amount of time that a packet spends in place *Wait* before entering in the transmission process. Hence, at each sensor X we have to define for each packet P_i the time of bindings $< Arrive, z =?, Packets = list, s1 =?, P = \{packet = P\_i, AT = ?\} >$ and $< Send, s1 =?, Packets = list, state = x, P = \{packet = P\_i, AT = ?\} >$ from paths in the reachability graph. Thus, the waiting time is calculated by the subtraction of the second binding time from the first one (see Figure III.10). Finally, the waiting time average of all packets within the system is computed.



```
28    119   Arrive @ (1:Sensor)
 - z = 1
 - packets = []
 - s1 = {num=1,pw=1000}
 - P = {packet=(1,"",1,0),AT=119}
29    135   Sense @ (1:Sensor)
 - np = 0
 - s1 = {num=1,pw=1000}
 - packets = []
 - P = {packet=(1,"",1,0),AT=119}
 - state = freed
30    135   DIFS @ (1:Sensor)
 - state = freed
 - P = {packet=(1,"",1,0),AT=119}
31    145   Send @ (1:Sensor)
 - packets = []
 - s1 = {num=1,pw=995}
 - state = freed
 - P = {packet=(1,"",1,0),AT=119}
```

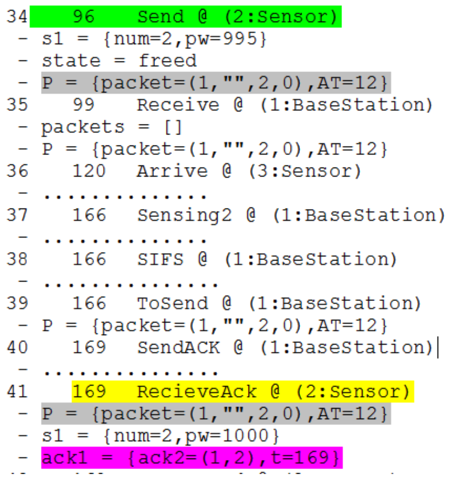
Figure III.10: Waiting Time Metric



```
28    119   Arrive @ (1:Sensor)
 - z = 1
 - s1 = {num=1,pw=1000}
 - P = {packet=(1,"",1,0),AT=119}
29    135   Sense @ (1:Sensor)
 - s1 = {num=1,pw=1000}
 - P = {packet=(1,"",1,0),AT=119}
 - state = freed
30    135   DIFS @ (1:Sensor)
 - state = freed
 - P = {packet=(1,"",1,0),AT=119}
31    145   Send @ (1:Sensor)
 - s1 = {num=1,pw=995}
 - state = freed
 - P = {packet=(1,"",1,0),AT=119}
32    149   Receive @ (1:BaseStation)
 - packets = []
 - P = {packet=(1,"",1,0),AT=119}
```

Figure III.11: Delay Performance Metric

- **Metric 2:** The delay performance DP, which is the duration from the moment that a packet enters the transmitter side to the moment that the packet is successfully received at the receiver side. This performance metric is given by: the amount of time that a packet spends in place *Wait* before it reaches place *RC_Packet* in the base station. Hence, at each sensor X we have to define for each packet P_i the time of bindings $< Arrive, z =?, Packets = list, s1 =?, P = \{packet = P\_i, AT = ?\} >$ and $< Receive, Packets = list, P = P = \{packet = P\_i, AT =?\} >$



```
34    96    Send @ (2:Sensor)
 - s1 = {num=2,pw=995}
 - state = freed
 - P = {packet=(1,"",2,0),AT=12}
35    99    Receive @ (1:BaseStation)
 - packets = []
 - P = {packet=(1,"",2,0),AT=12}
36    120   Arrive @ (3:Sensor)
 - ..............
37    166   Sensing2 @ (1:BaseStation)
 - ..............
38    166   SIFS @ (1:BaseStation)
 - ...............
39    166   ToSend @ (1:BaseStation)
 - P = {packet=(1,"",2,0),AT=12}
40    169   SendACK @ (1:BaseStation)
 - ...............
41    169   RecieveAck @ (2:Sensor)
 - P = {packet=(1,"",2,0),AT=12}
 - s1 = {num=2,pw=1000}
 - ack1 = {ack2=(1,2),t=169}
```

Figure III.12: Waiting Time for an ACK Metric

from paths in the reachability graph. Then, the delay performance is calculated by the subtraction of the second binding time from the first one. Finally, the delay performance average DP of all packets within the system is computed (see Figure III.11).

- **Metric 3:** The waiting time for an acknowledgement. After the transmission of the data packet, the sensor must enter in the waiting for a positive acknowledgement. This performance metric is given by: the amount of time that a packet spends in place *WaitAck* before it receives an acknowledgement in the place *Ack*. Hence, at each sensor X we have to define for each packet P_i the time of bindings $< Send, s1 =?, Packets = list, state = x, P = \{packet = P\_i, AT =?\} >$ and $< ReceiveACK, P = \{packet = P\_i, AT =?\}, s1 =?, ack1 =? >$ from paths in the reachability graph. Then, the waiting time for an ACK is calculated by the subtraction of the second binding time from the first one (see Figure III.12). Finally, the waiting time for an acknowledgement average of all packets within the system is computed.

  A set of data collector monitors is defined that extract numerical data concerning the waiting time, the delay performance and the waiting time for an ACK metrics from our HTCPN model during simulations. The extracting files are used to plot this metrics versus number of simulations that are shown in Figures III.13, III.14 and III.15. Indeed, the average waiting time, the average delay performance and the average waiting time for an ACK are shown in the performance report (Tables III.7, III.8 and III.9).

  On the other hand, the following non-temporal metrics can also be considered to evaluate the protocol:

- **Metric 4:** Energy consumption, in the proposed model each operation realised by the sensor consumes energy. After consuming energy, the sensor which its energy is

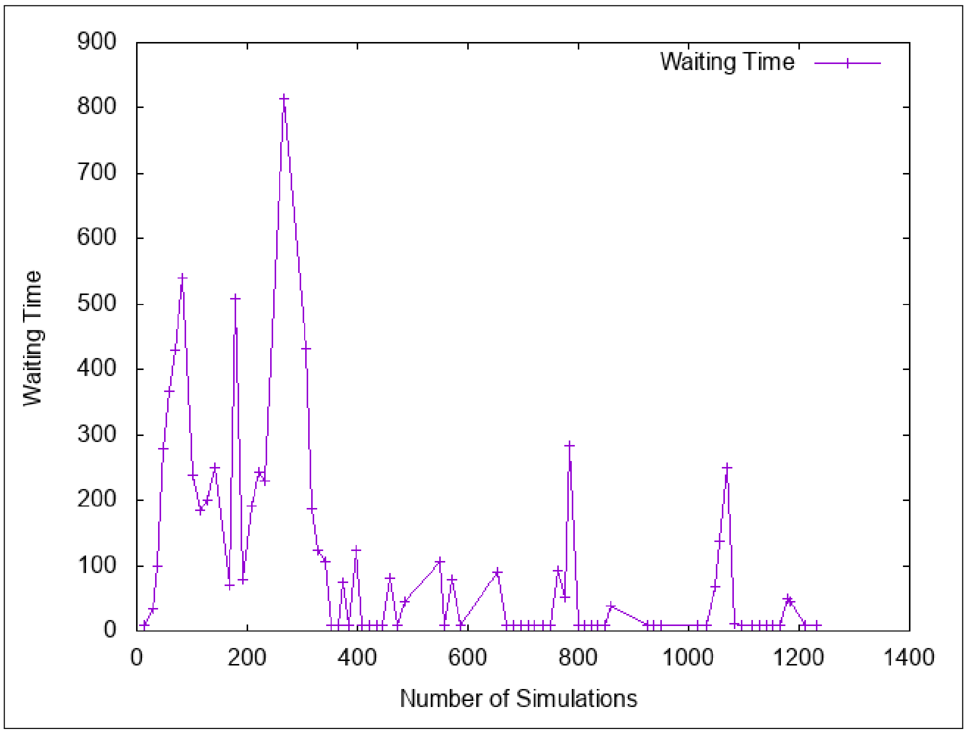


Figure III.13: Waiting Time Metric

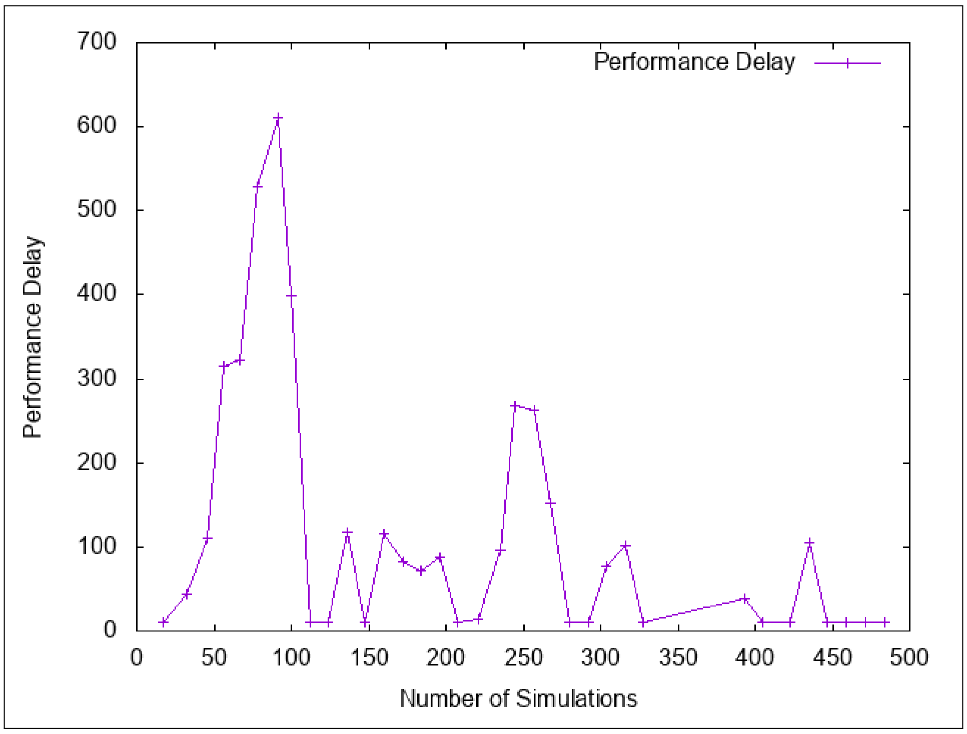


Figure III.14: Delay Performance Metric

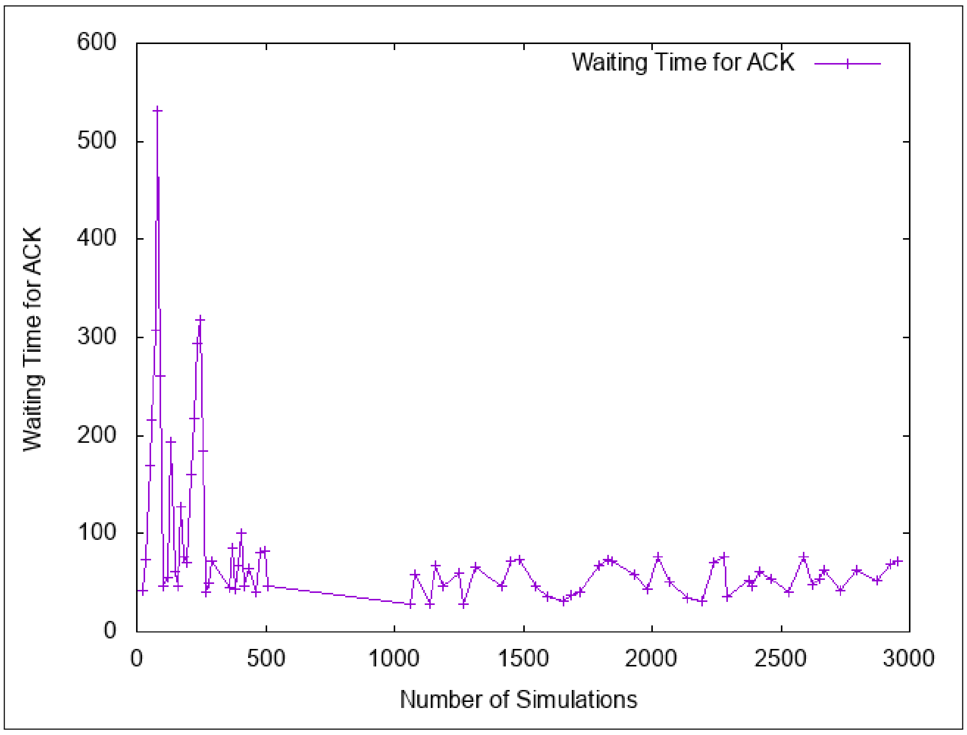


Figure III.15: Waiting Time for an ACK Metric

Name	**Count**	**Sum**	**Avg**	**Min**	**Max**
Waiting_Time	74	7588	102.540541	10	815

Table III.7: Delay Performance Average

Name	**Count**	**Sum**	**Avg**	**Min**	**Max**
Performance_Delay	41	2351	57.341463	10	315

Table III.8: Waiting Time Average

Name	**Count**	**Sum**	**Avg**	**Min**	**Max**
Waiting_Time_for_an_ACK	76	6569	86.434211	28	531

Table III.9: Waiting Time for an ACK Average

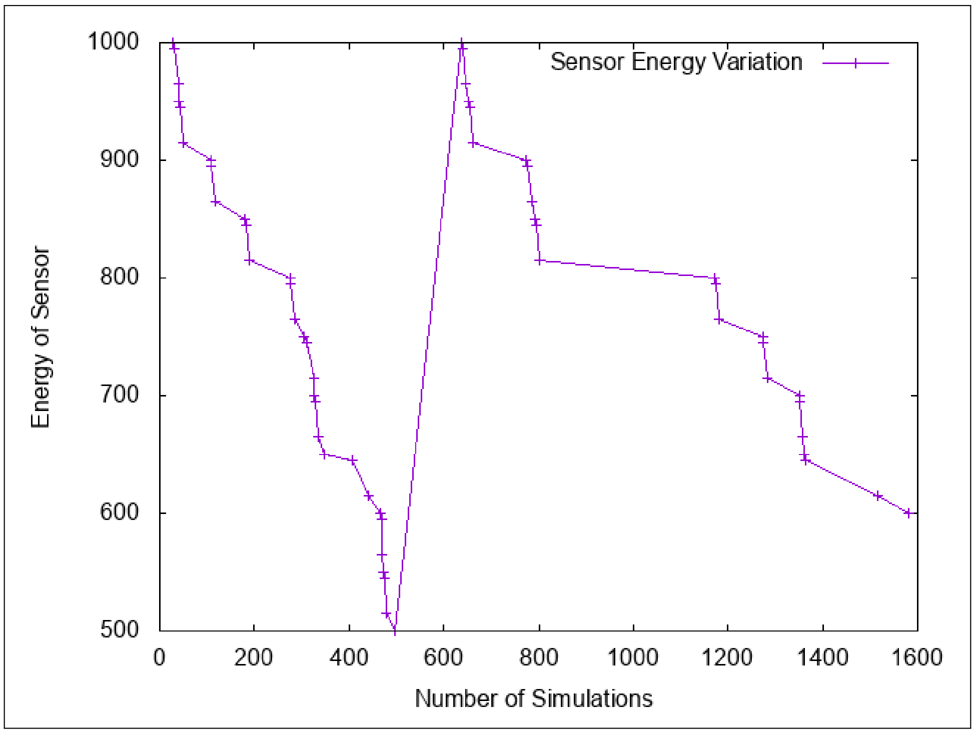


Figure III.16: Energy of sensor

less then $500\ units$, will harvest energy from environment. A data collector monitor is defined to extract the energy variation of such sensor during simulation. Then, the extracting file is used to plot the amount of energy of sensor versus number of simulation that is shown in Figure III.16.

- **Metric 5:** Arriving of packets, in the proposed HTCPN model the number of packets is not fixed beforehand. Therefore, the coming of packets is modelled by *Arrive* transition which guarantee the periodical arriving of packets in the network. Indeed, to calculate the number of arriving packets, a write in file monitor is defined. Then, the extracting file is used to plot the number of arriving packets versus time that is shown in Figure III.17.

- **Metric 6:** Throughput at time T, this metric is computed as the rate of the total number of received acknowledgements to the total number of packets transmitted by nodes at time T. To calculate this metric, a set of data collector monitors is defined which extract the number of arrived packets and the number of received acknowledgements. The result is shown in performance report (Table III.10 and Table III.11). At time $T = 5822$, the throughput is equal to the sum of received acknowledgements divided by the sum of arrived packets.

$$Throughput = \frac{\sum receivedacknowledgements}{\sum arrivedpackets} = \frac{15+11+13+9+10}{21+21+13+13+14} = 0.7$$

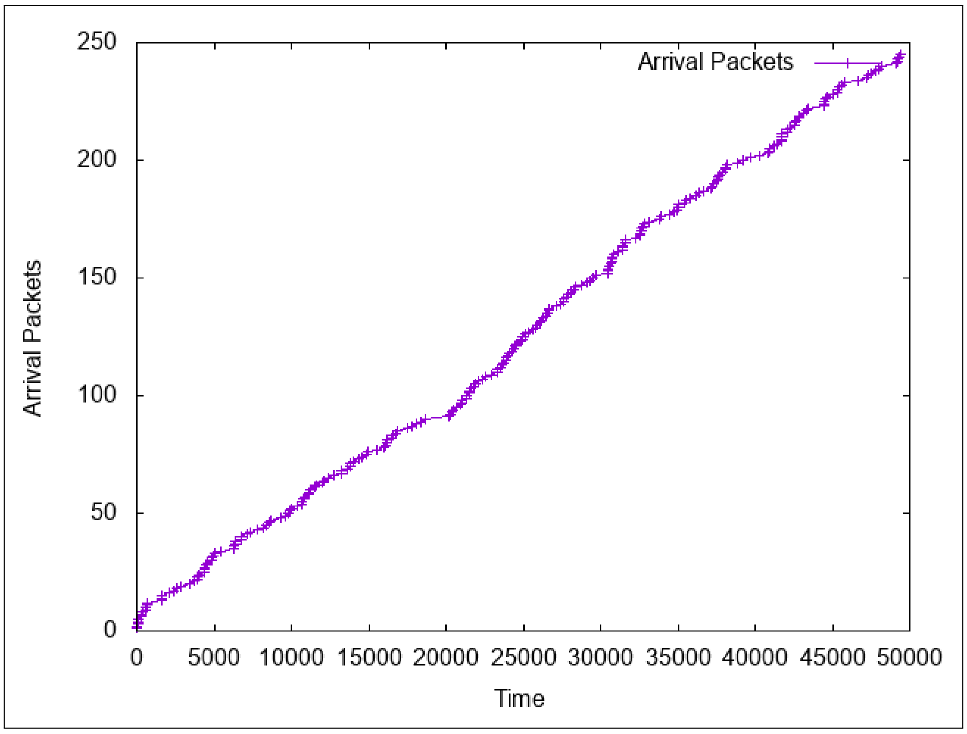


Figure III.17: Arrived packets in the network

Name	Count	Sum	Avg	Min	Max
Arrived_packets	82	749	9.134146	1	21
Count_trans_occur_Sensor’Arrive_1	21	21	1.000000	1	1
Count_trans_occur_Sensor’Arrive_2	21	21	1.000000	1	1
Count_trans_occur_Sensor’Arrive_3	13	13	1.000000	1	1
Count_trans_occur_Sensor’Arrive_4	13	13	1.000000	1	1
Count_trans_occur_Sensor’Arrive_5	14	14	1.000000	1	1

Table III.10: The number of arrived packets by sensor

Name	Count	Avg	Min	Max
List_length_dc_Sensor’Success_1	17	6.558406	0	15
List_length_dc_Sensor’Success_2	13	5.876236	0	11
List_length_dc_Sensor’Success_3	15	8.302367	0	13
List_length_dc_Sensor’Success_4	11	4.320947	0	9
List_length_dc_Sensor’Success_5	12	6.261972	0	10

Table III.11: The number of received acknowledgements by sensor

III.5 Conclusion

In this chapter, a formal approach is proposed to model and analyse CSMA/CA MAC protocol in EHWSNs. This approach allows researchers to evaluate the performance of this protocol in EHWSNs. Using the proposed approach, we can simulate the behaviour of the network with random packets and whatever the number of sensors. Moreover, we can verify some properties which have an influence on CSMA/CA MAC protocol. Indeed, the main goal of the performance evaluation is to obtain a MAC protocol with high quality of service and efficient communications between sensor nodes and base station.

Hierarchical CPNs allow us to construct the large network model as smaller models connected to each other. This last advantage makes the model more readable. Furthermore, HTCPNs provide a timed aspect that facilitates the consideration of time constraints of the CSMA/CA protocol.

In the following chapter, machine learning will be exploited to prove the scalability of the proposed HTCPNs model presented in chapter two. An effective artificial neuronal network will be applied to predict the evolution of the verified performance metrics when the number of nodes increases in the network.

Chapter IV

Third Contribution: Using Machine Learning in the Study of Scalability

IV.1 Introduction

The formal analysis of designed models is a crucial process. However, it can be very hard in terms of time and effort. For that, researchers adopt Machine Learning (ML) [19] techniques as an effective alternative for the performance evaluation estimation of designed protocols and algorithms in WSNs. In particular, computer systems can use machine learning algorithms and techniques to learn from data allowing these systems to classify and predict [20, 21].

Machine learning (ML) has recently attracted considerable attention for its ability to predict a wide variety of complex phenomena accurately [22]. Researchers provide that network performance improves by using ML techniques for network design and management [82]. Applying ML in WSNs lies in exploiting historical data to improve the performance of sensor networks on given tasks without the need for re-programming. Artificial Neural Networks (ANNs), or simply Neural Networks (NNs) [83] constitute one of the most popular ML models applied in WSNs. Using these learning systems, NNs can predict outputs for a set of given inputs [21]. The structure of NNs is composed of a set of artificial neurons interconnected together with an aim to mimic the neural processing (organization and learning)of biological neurons and their behaviour.

In this chapter, an effective ANN is proposed to predict the evolution of throughput metric and some temporal performance metrics in CSMA/CA MAC protocol. This prediction uses prior data obtained by a formal model which is a Hierarchical Timed Coloured Petri Net (HTCPN) model of CSMA/CA protocol proposed in chapter two. Indeed, the throughput metric and three crucial temporal metrics have been predicted to discuss their evolution and the performance of the WSNs when the number of nodes increases. The studied temporal metrics include average packet waiting time, average packet delay performance, and average packet waiting time for acknowledgement. The rest of the chapter is organised as follows. In Section IV.2, related works using ML techniques in WSNs are expressed. Section IV.3 discusses the scalability of the proposed model in two phases modelling and analysis. Section IV.4 presents the predicted result concerning temporal performance metrics. Finally, Section IV.5 gives a brief conclusion from what this chapter presents.

IV.2 Related Work

Usually, sensor network designers characterize machine learning as a collection of tools and algorithms that are used to create prediction models. However, machine learning experts recognize it as a rich field with very large themes and patterns. Understanding such themes will be beneficial to those who wish to apply machine learning to WSNs. Applied to numerous WSNs applications, machine learning algorithms provide tremendous flexibility benefits.

In the literature, many works have investigated Artificial Intelligence (AI) techniques to improve wireless sensor networks performance. Researchers in [84] present an extensive review of works applied Machine Learning (ML) to address common issues in WSNs. In this section, some works related to the use of machine learning approaches in WSNs are presented.

The authors in [85] present a delay prediction scheme for mobile wireless networks using two types of neural networks: Multi-Layer Perceptron (MLP) network and Radial Basis Function (RBF) network. Two types of inputs for prediction are used the mean delay time series itself only and the mean delay time series together with the corresponding traffic loads. Moreover, the traffic pattern generated in experiment by authors is not as complex as the traffic pattern in the real network.

The work [86] compares statistical models based on time series applied to predict half daily values of global solar irradiance with a temporal horizon of 3 days. The models tested are autoregressive, neural networks, and fuzzy logic models. The authors incorporate an artificial neural network model to propose a scheme that predicts irradiance values over a time horizon of half a day. The scheme is shown to outperform the autoregressive and fuzzy logic models by achieving increased accuracy.

The presented work in [87] describes fuzzy Hopfield neural network (FHNN) technique to solve the TDMA (Time Division Multiple Access) broadcast scheduling problem in WSN. To find the TDMA schedule for nodes in a communication network the authors have been formulated the problem as discrete energy minimization and mapped it into a Hopfield neural network with the fuzzy c-means strategy. The FHNN reduces the processing time and increases the convergence rate for Broadcast Scheduling Problem. Simulation results show that the FHNN improves performance substantially through solving well-known benchmark problems.

Research published in [88] discusses an application of a neural network in wireless sensor network security. The work presents a MultiLayer Perceptron (MLP) based media access control protocol (MAC) to secure a CSMA-based wireless sensor network against the denial of service attacks launched by adversaries. The authors have been implemented the MLP-guarded secure WSN using the Vanderbilt Prowler simulator [89]. The obtained simulation results show that the MLP helps in extending the lifetime of the WSN.

The research in [90] proposed algorithms that use a three-layer neural network. Input layer neurons are located in each cluster's members, while the hidden layer neurons are located in cluster heads, and output layer neurons are located in the base station. Simulation results show that data aggregation performance is improved, and energy consumption of the network is reduced.

ML techniques are widely used in the Wireless Sensor Networks(WSNs) field [84] and in Software Defined Networking (SDN) area [91] to improve WSNs performance. Moreover, no research before has been done on the prediction of performance metrics of WSNs MAC protocols using ML. In this chapter, neuronal network machine learning is exploited to predict the performance evaluation of the CSMA/CA protocol. The proposed ANN is used to predict the evolution of throughput metric and a set of temporal performance metrics based on increasing the number of nodes in the network. To the best of our knowledge, no one before has been applied ML to predict the formal analysis of HTCPNs models, which was our third contribution.

IV.3 Scalability of the Proposed Approach

Scalability is an important issue which concerns the study of the WSN performance when the number of nodes increases. When the scalability factor is considered in wireless sensor network design, the protocol should work effectively in wide WSNs. The scalability of the proposed approach is discussed in this section. When using a formal model like HTCPNs, the major challenge is that the formal model size increases with the increase of the number of nodes. This challenge affects the two levels in the formal approach: modelling and verification. In the following, we discuss how this challenge is addressed in the two levels.

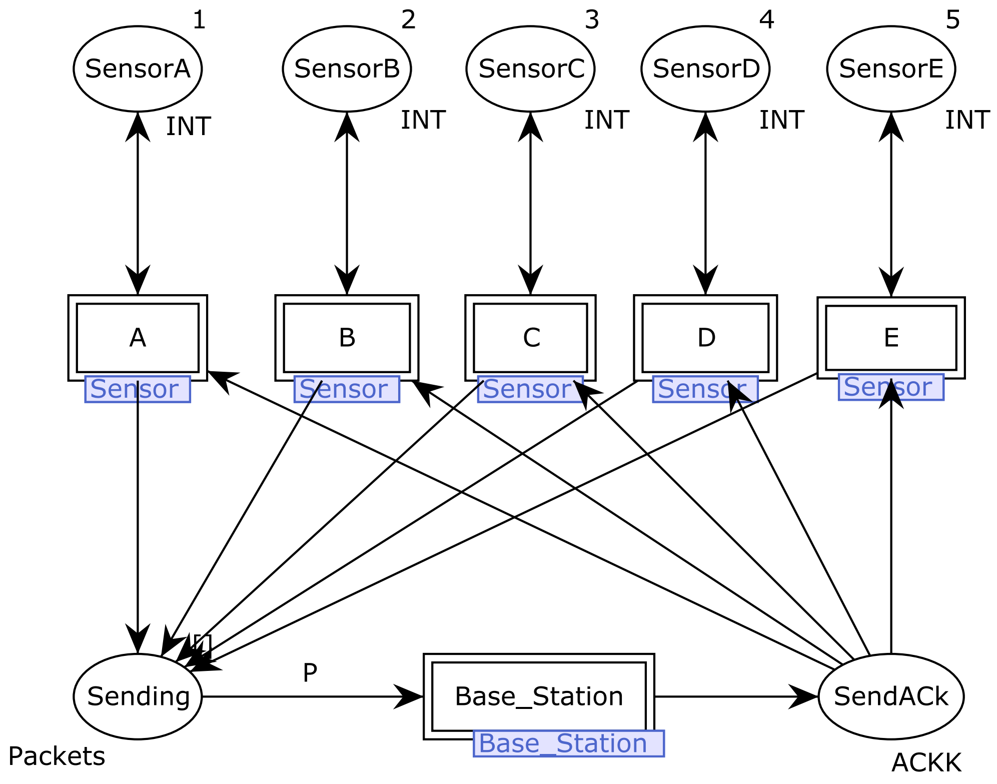


Figure IV.1: HTCPN abstract model for CSMA/CA protocol

IV.3.1 Modelling Phase and Scalability Study

Hierarchical Timed CPNs use hyper-transitions in the abstract model to hold in the sub-models. In the proposed HTCPN model, increasing the number of sensor nodes directly increases the number of sub-models. Thus, to generate a new sub-model for a new node, easily a copy of a hyper-transition $Sensor$ (Figure IV.1) has to be created. This modularity reduces the effort and time in the modelling process. Indeed, exploiting HTCPNs modularity helps the designer to study the scalability of the studied protocol with less complexity than standard Petri nets or Coloured Petri nets. In our study, the HTCPN model was extended to specify a network including 50 nodes. This number can be augmented easily by adding other sub-models. However, continuing this augmentation makes the verification process costly in computing time.

IV.3.2 Analysis Phase and Scalability Study

In this subsection, the scalability of the proposed HTCPN model will be discussed by increasing the number of nodes in the analysis phase. In the first hand, the applied machine learning steps will be presented.

IV.3.2.1 Machine Learning Process

The machine learning process is divided into four steps as listed below (see Section I.4).

1. **Data collection:** the first step before applying machine learning is collecting data. In our case, the data is tabular data that contains results obtained by simulations in CPN-Tools. A set of monitors are defined to extract performance measurements during simulations.

2. **Choosing a model:** after collecting data, we must choose a suitable model for the regression task. A Multi-Layer Perception (MLP) regressor model is chosen for our HTCPN model, a supervised learning algorithm. To built the MLP model, hyper-parameters must be defined, including the number of hidden layers, the number of neurons, batch size, number of epochs, learning rate, and other parameters. The difficulty is how to adjust the hyper-parameters manually by the expert; therefore, a grid search algorithm [92] is used for an automated hyper-parameters selection within the defined space and the parallel computing.

3. **Training the model:** the training process aims to adjust the weights of the MLP with its calculated gradient, which involves three cascaded stages: forward propagation, backward propagation, and the gradients applied [93].

4. **Test and evaluation of the model:** The trained model needs to be tested to see if it would operate well in real obtained analysis results.

Parameters	**Values**
Number of hidden layers	2
Number of neurons	{5, 15}
learning rate	0.02
Batch size	5
Epoch	300

Table IV.1: The best hyper-parameters of NN obtained by Grid search algorithm

On the other hand, before applying these ML steps, we must prepare the data. The gathered data (Excel file) is obtained by extracting information about throughput metric using monitors defined in CPN-Tools. These monitors extract information during simulation of the proposed HTCPN model. The obtained simulation results concerning the throughput metric will be presented in the next subsection.

IV.3.2.2 Simulation Results

Using CPN-Tools, we have increased the number of nodes till fifty nodes and we have checked the evolution of the throughput metric according to the nodes number increase. After 1000 simulations for each model, the throughput metric is calculated from monitors showing the number of arrived packets in the network and the succeeded sent packets. The generated simulation report is used to plot the evolution of the throughput according to the number of nodes; Figure IV.2 illustrated this evolution. In this Figure, we observe that the throughput metric decreased in the first twenty nodes. After that, it becomes almost steady with minimal changes (decreasing and increasing).

IV.3.2.3 Throughput Metric Estimation

Applying Machine Learning steps defined above, and based on the results obtained for HTCPN model composed of 1 node to 50 nodes (50 sub-models), we have exploited a neuronal approach to investigate how the evolution will be when the number of nodes evolves from 51 nodes until 100 nodes. An effective neuronal network is exploited for the prediction of the throughput metric based on the number of nodes. Hyper-parameters are defined using an automated grid search algorithm. These Hyper-parameters include the number of hidden layers, the number of neurons, batch size, number of epochs, learning rate, etc. The best hyper-parameters of MLP obtained using the grid search method is shown in Table IV.1. The proposed NN architecture illustrated in Figure IV.3 was trained using back-propagation [94] algorithm and we applied the sigmoid activation function in the output layer, which ranges from 0 to 1 as throughput values. Figure IV.4 shows that the predicted throughput values stabilised as the number of nodes increases.

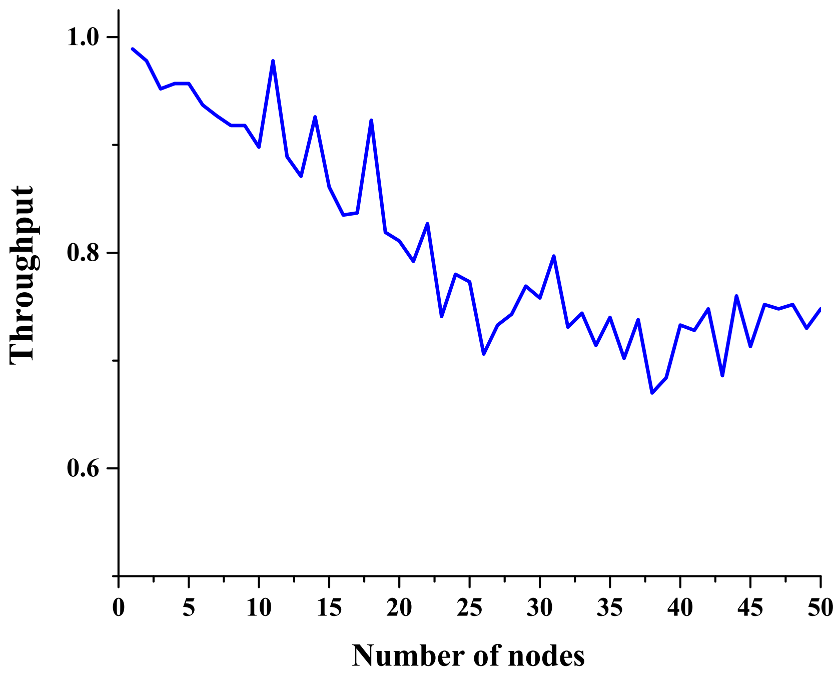


Figure IV.2: Throughput values by number of nodes computed using CPN-tools simulation report.

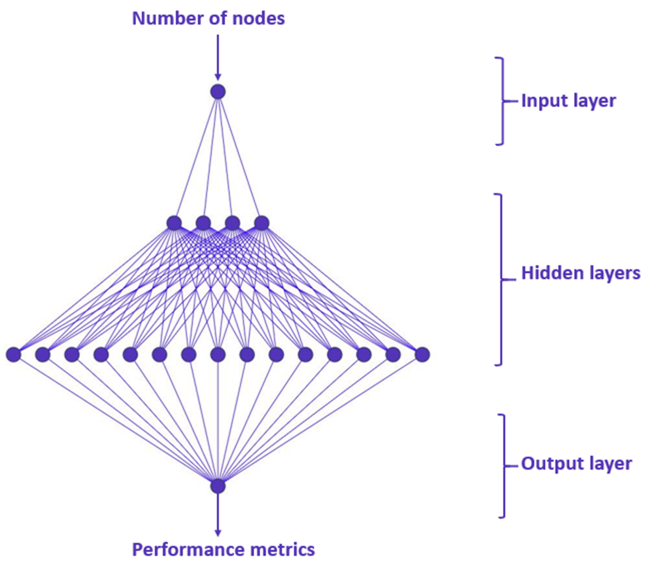


Figure IV.3: Neuronal network architecture.

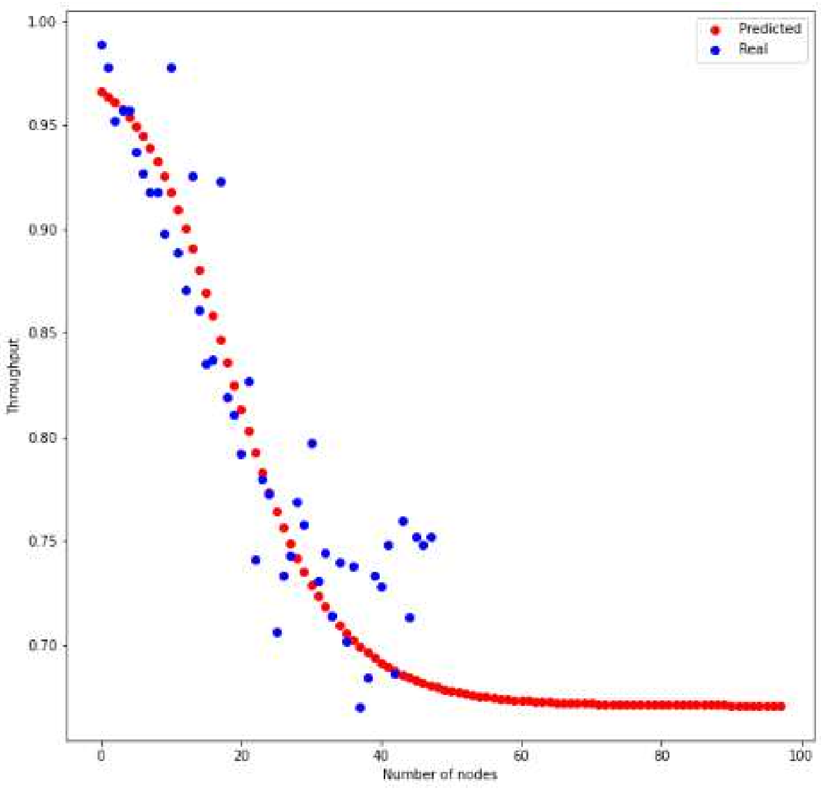


Figure IV.4: Predicted throughput values vs. number of nodes computed by NN.

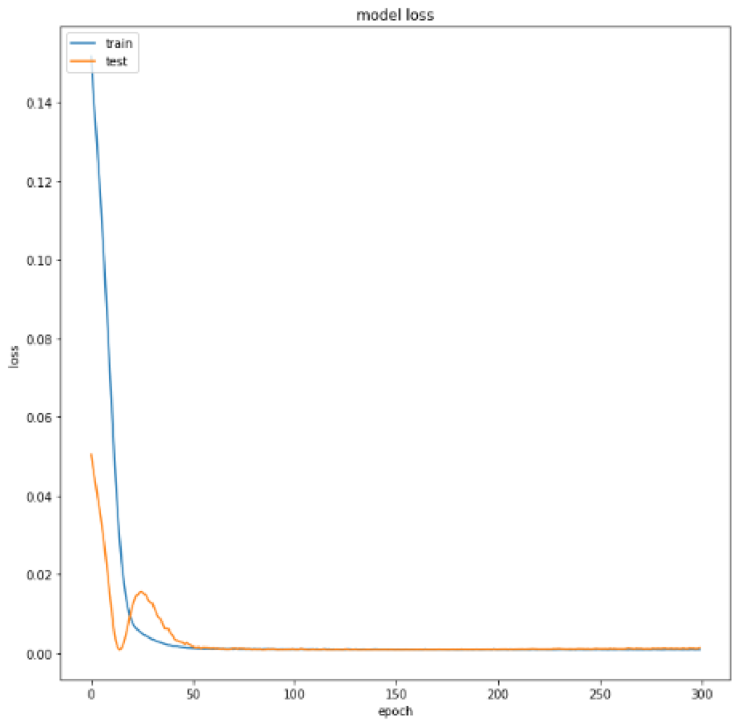


Figure IV.5: Loss function.

IV.4 Prediction of Temporal Performance Metrics

Performance evaluation of WSNs MAC protocols is a necessary process that helps designers to obtain numerical result from their proposed models and to prove the correctness of each designed protocol. The proposed Hierarchical Timed Coloured Petri Net (HTCPN) model illustrated in Figure IV.1, was presented in the second chapter in this thesis and was simulated in CPN-Tools to obtain numerical results about a set of performance measures. A collection of monitors are defined in CPN-Tools to extract these performance measures. In CPN-Tools, the monitor mechanism is used to examine the markings and occurring binding elements and to extract information from the constructed model periodically. The performance measures include average packet waiting time, average packet delay performance, and average packet waiting time for acknowledgement. The last metrics are considered the most influential metrics. The previous metrics are calculated as follows:

- **Average packet Waiting Time:** the waiting time of a packet is calculated by subtracting the packet's arrival time from the sending time of this packet. The average packet waiting time is automatically computed by the defined monitor and shown in the generated performance report.
- **Average packet Delay Performance:** the delay performance of a packet is calculated by subtracting the packet sending time from the receiving time of this packet by the base station. The average packet delay performance is automatically computed by the defined monitor and shown in the generated performance report.
- **Average packet Waiting Time for Acknowledgement:** the waiting time for an acknowledgement (ACK) of a packet is calculated by subtracting receiving ACK time of the packet from the sending time of this packet. The average packet waiting time for acknowledgement is automatically computed by the defined monitor and shown in the generated performance report.

IV.4.1 Simulation Results

To observe the evolution of performance metrics defined in the previous section, we have increased the number of nodes by increasing the number of HTCPN models. After 1000 simulations of each proposed model in CPN-Tools, the obtained numerical results are used to plot the evaluation of previous metrics based on the number of nodes in the network. Using CPN-Tools,we have increased the number of nodes to fifty nodes, and we have checked the evolution of the average waiting time, average delay performance, and average waiting time for ACK metrics according to the nodes number increase. After 1000 simulations for each model, the three metrics are calculated from monitors showing the arrival time, sending time, and receiving time of all packets in the network. The generated simulation reports are exploited to plot the evolution of the previous metrics according to the number of nodes. Figure IV.6, Figure IV.7 and Figure IV.8 present average waiting time, average delay performance, and average waiting time for ACK metrics by number of nodes increase.

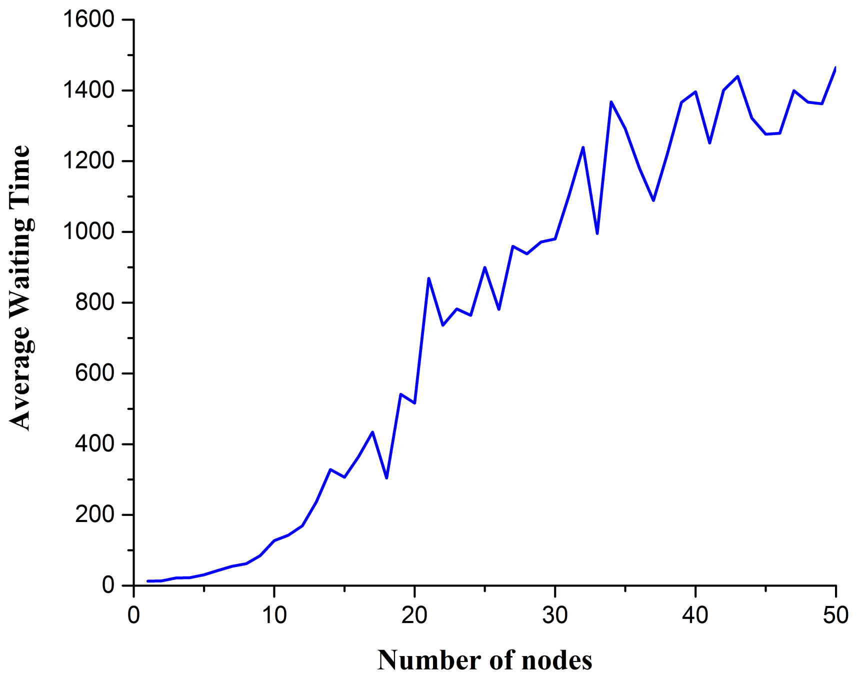


Figure IV.6: Average waiting time by number of nodes.

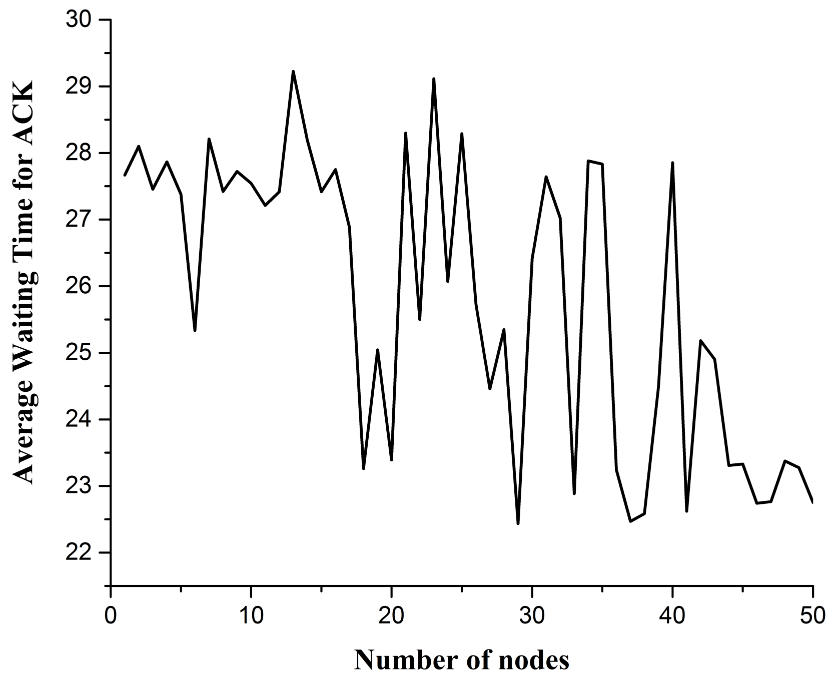


Figure IV.7: Average waiting time for ACK by number of nodes.

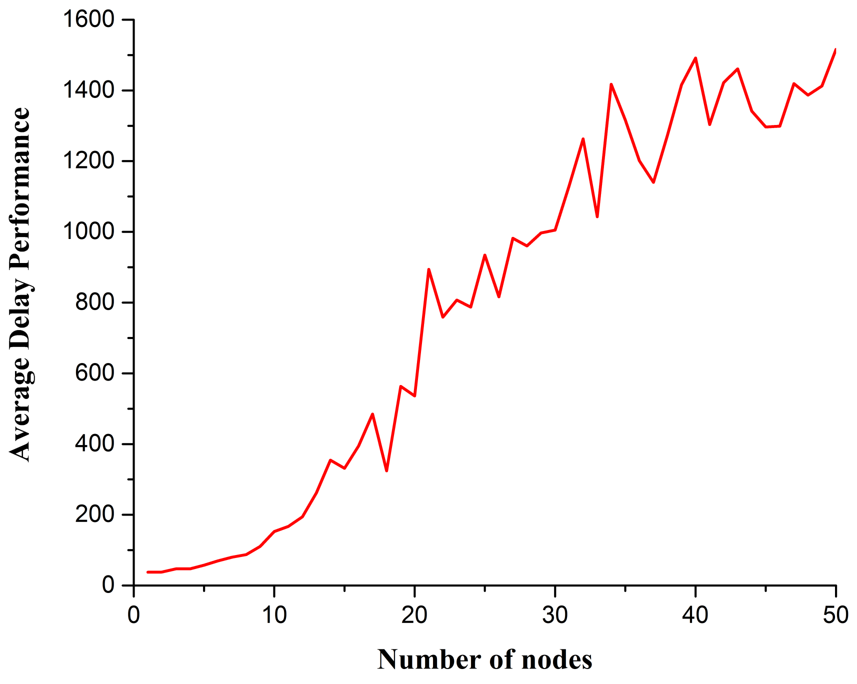


Figure IV.8: Average delay performance by number of nodes.

IV.4.2 Performance Evaluation Prediction

Based on the results obtained from simulations of the proposed HTCPN models composed of one node to 50 nodes (50 sub-models), a neuronal approach is exploited to investigate how the evolution will be when the number of nodes evolves from 51 nodes until 100 nodes. An effective neuronal network is applied to predict *the average waiting time, average delay performance time, and average waiting time for ACK* metrics based on the number of nodes. The same ML steps are appealed to give the estimated evolution of these performance metrics. The proposed MLP architecture illustrated in Figure IV.3 was trained using backpropagation algorithm, and we have applied the RELU (Rectified Linear Unit) activation function in the output layer, which ranges from 0 to infinity as the performance metrics values. Figure IV.9, Figure IV.10 and Figure IV.10 show the predicted metrics values of average waiting time (WT), average waiting time for ACK (WTA) and average delay performance (DP), respectively.

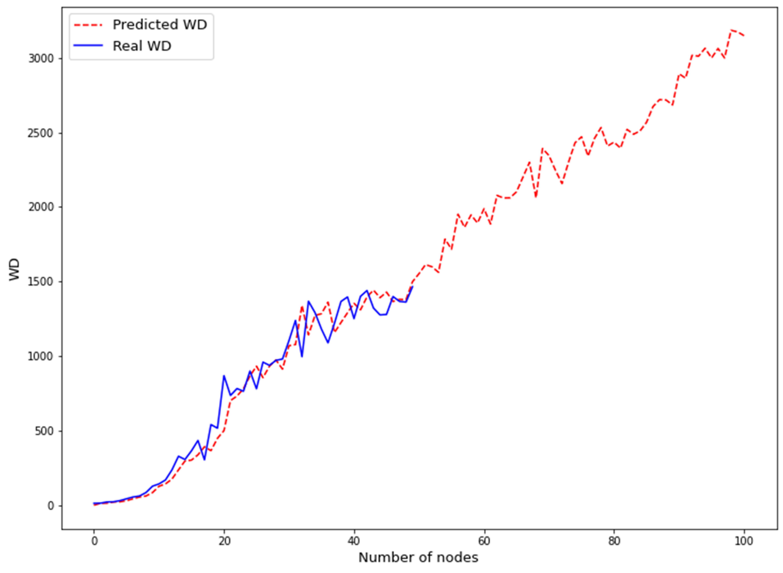


Figure IV.9: Predicted average waiting time based on number of nodes.

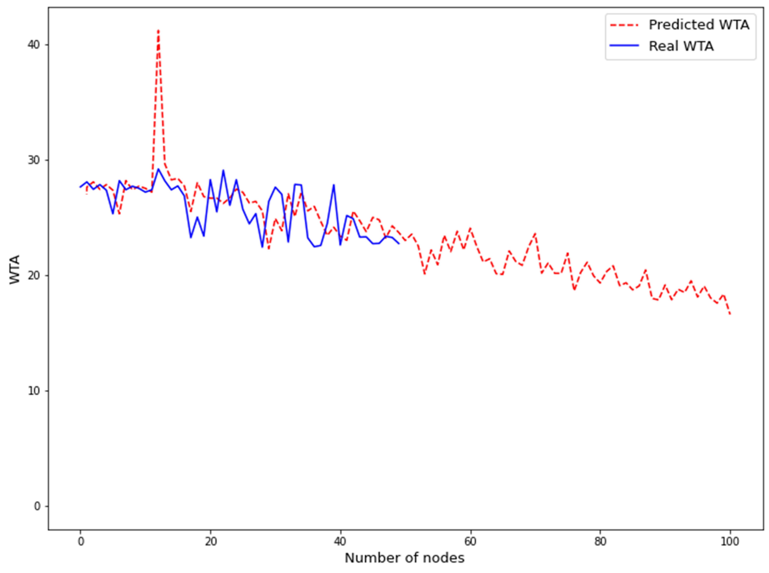


Figure IV.10: Predicted average waiting time for ACK based on number of nodes.

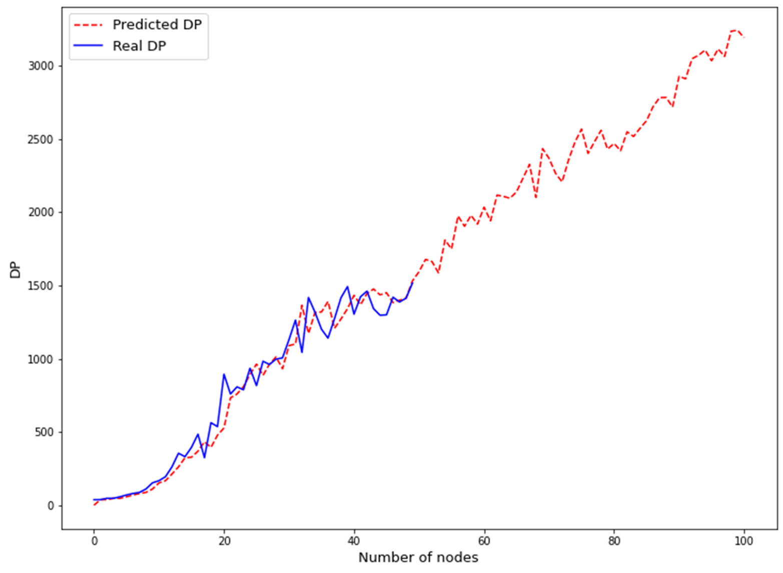


Figure IV.11: Predicted average delay performance based on number of nodes.

IV.5 Conclusion

In this chapter, the third contribution of this thesis is presented. This contribution concerns the application of Machine Learning to prove the scalability of the proposed Hierarchical Coloured Petri Net model and to predict the evolution of some temporal performance metrics. By increasing the number of sensor nodes, the throughput metric of the network is estimated and a set of performance metrics also is predicted using the obtained numerical results by simulations in CPN-Tools. The main advantage of using Machine Learning is allowing the designer to reduce time and complexity in the analysis phase of the proposed models. The predicted values of the throughput metric can demonstrate that the proposed HTCPN is scalable when the number of nodes increases in the network. Indeed, the estimated values of temporal performance metrics show that the relation between these metrics and the number of nodes is linear.

CONCLUSION

Thesis aims

The design and implementation of communication protocols are crucial processes in wireless sensor networks and energy harvesting WSNs. The verification, analysis, and validation processes of communication protocols are essential components of high-quality performance in such WSNs. These three steps are used to prove the consistency of a communication protocol and help designers to verify and evaluate their proposed protocol models. For those reasons, the main goal of this thesis is to model, analyse and formally verify CSMA/CA MAC protocol.

We have chosen CSMA/CA protocol to be study because the medium access control is the most predominant protocol for specifying channel access and energy consumption in WSNs. However, the major issue of WSNs is the limited energy of sensor nodes. The energy harvesting technologies are appeared to respond to this issue. The specification of the sensor node in EHWSNs includes the harvesting process. CSMA/CA is one of the most important MAC protocols, which is proposed to deal with the temporal aspects, and it defines necessary temporal constraints to access to the channel. Moreover, the CSMA/CA verification approach can be used as a base for other MAC protocols verification and validation. On other hand, the high density of nodes in WSNs is a crucial challenge, to solve the problem of huge formal models analysis for protocols and algorithms artificial intelligence method can be investigated. An artificial neuronal network as part of supervised Machine Learning techniques is used in performance evaluation prediction of the proposed formal approach of CSMA/CA in wide WSNs.

Finally, the formal approaches proposed in this thesis are an essential action toward leveraging a collection of techniques and tools that help with the design of communication protocols. Indeed, the exploitation of Machine Learning to estimate and predict the performance evaluation of such protocol in aim to solve the problem of complicated analysis of formal approaches is a new step to combine intelligence methods with formal methods.

Thesis Contributions

The main contributions of this thesis can be summarized as follows:

First Contribution

The first contribution was the use of Hierarchical Timed Coloured Petri Nets in the formal study of CSMA/CA MAC protocol in WSNs. This approach uses HTCPN formalism to model the protocol, and the CPN-Tools to analyse the generated models. The time aspect, in HTCPN, facilitates the consideration of temporal constraints introduced in CSMA/CA protocol. The hierarchical aspect of HTCPN makes the model "manageable", despite the complexity of CSMA/CA protocol specification. In the HTCPN model, the events that can occur in the network and their preconditions and post-conditions are defined. These preconditions and post-conditions specify CSMA/CA protocol constraints that should be satisfied. After the specification by using the Hierarchical Timed CPN formalism, the CPN-Tools was used to ensure an analysis of the protocol and to verify some inherent properties. We used the Hierarchical Timed Petri Nets (HTCPNs) in a down-top approach (from an abstract model to sub-models) to model CSMA/CA. Besides the use of HTCPNs, we presented the analysis phase that focuses on the verification process of several temporal properties and the linear temporal logic is used to formalise a set of qualitative properties.

Second Contribution

The second contribution was the modelling and the verification of CSMA/CA protocol in energy harvesting WSNs. A modelling approach is proposed for the performance evaluation of CSMA/CA protocol. Specifically, Hierarchical Timed Coloured Petri Nets is used to model this protocol. Qualitative and quantitative properties have been verified. The analysis of the proposed model has been realised by CPN-Tools. The generated models are simulated in various scenarios by varying the number of sensor nodes. The LTL logic is used to formalised qualitative properties. The quantitative properties are verified by exploiting the generated files during simulations.

Third Contribution

The third Contribution was the discuss of the scalability of the the proposed model in the first contribution. In the modelling phase, the advantage of the hierarchical aspect of hierarchical timed coloured Petri nets can reduces the effort and time in the modelling process by easily create a copy of a each hyper-transition in the proposed model to generate a new sub-model. In our study, the HTCPN model was extended to specify a network including 50 nodes. This number can be augmented easily by adding other sub-models, however continuing this augmentation makes the verification process costly in computing time. However, in the analysis phase, Machine Learning was proposed to predict the throughput of the network by using the obtained throughput metrics. Based on the results obtained for networks composed from 1 node to 50 nodes, we have exploited a neuronal approach to investigate how the evolution will be when the number of nodes evolves from 51 nodes until 100 nodes. An effective neuronal network is exploited for the prediction of the throughput metric based on the number of nodes.

Thesis drawbacks and future directions

To resolve the thesis's drawbacks, the future aims and our suggestions for future works are:

- Generalizing the formal verification approach of CSMA/CA to be suitable to other MAC protocols.
- Using other algorithms of Machine Learning to discuss the scalability of the proposed approach, comparing the results and choose the best ML method.
- Using Machine Learning to prove the scalability of the proposed approach presented in second contribution.

Bibliography

[1] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422, 2002.

[2] Bandar Alghamdi and Hacene Fouchal. A mobile wireless body area network platform. *Journal of Computational Science*, 5(4):664–674, 2014.

[3] Th Arampatzis, John Lygeros, and Stamatis Manesis. A survey of applications of wireless sensors and wireless sensor networks. In *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, pages 719–724. IEEE, 2005.

[4] Ali Alemdar and Mohamed Ibnkahla. Wireless sensor networks: Applications and challenges. In *2007 9th International Symposium on Signal Processing and Its Applications*, pages 1–6. IEEE, 2007.

[5] Lucia Keleadile Ketshabetswe, Adamu Murtala Zungeru, Mmoloki Mangwala, Joseph M Chuma, and Boyce Sigweni. Communication protocols for wireless sensor networks: A survey and comparison. *Heliyon*, 5(5):e01591, 2019.

[6] Kazem Sohraby, Daniel Minoli, and Taieb Znati. *Wireless sensor networks: technology, protocols, and applications*. John wiley & sons, 2007.

[7] Adamu Murtala Zungeru, Li-Minn Ang, and Kah Phooi Seng. Classical and swarm intelligence based routing protocols for wireless sensor networks: A survey and comparison. *Journal of Network and Computer Applications*, 35(5):1508–1536, 2012.

[8] Muhammad Faheem, Muhammad Zahid Abbas, Gurkan Tuna, and Vehbi Cagri Gungor. Edhrp: Energy efficient event driven hybrid routing protocol for densely deployed wireless sensor networks. *Journal of network and computer applications*, 58:309–326, 2015.

[9] Muhammad Faheem and Vehbi C Gungor. Energy efficient and qos-aware routing protocol for wireless sensor network-based smart grid applications in the context of industry 4.0. *Applied Soft Computing*, 68:910–922, 2018.

[10] Muhammad Faheem and Vehbi Cagri Gungor. Mqrp: Mobile sinks-based qos-aware data gathering protocol for wireless sensor networks-based smart grid applications in the context of industry 4.0-based on internet of things. *Future Generation Computer Systems*, 82:358–374, 2018.

[11] Leonard Kleinrock and Fouad Tobagi. Packet switching in radio channels: Part i-carrier sense multiple-access modes and their throughput-delay characteristics. *IEEE transactions on Communications*, 23(12):1400–1416, 1975.

[12] IEEE 802.11 Working Group et al. Part 11: wireless lan medium access control (mac) and physical layer (phy) specifications: higher-speed physical layer extension in the 2.4 ghz band. *ANSI/IEEE Std 802.11*, 1999.

[13] LAN/MAN Standards Committee et al. Part 15.4: wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (lr-wpans). *IEEE Computer Society*, 2003.

[14] Tadao Murata. *Petri Nets and their Application an Introduction*, pages 351–368. Springer US, Boston, MA, 1984.

[15] Kurt Jensen. *An introduction to the theoretical aspects of Coloured Petri Nets*, pages 230–272. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[16] Cpn-tool can be downloaded (free for academics) from:. `http://wiki.daimi.au.dk/cpntools/cpntools.wikim`. Accessed: january 3rd 2021.

[17] Shashank Priya and Daniel J Inman. *Energy harvesting technologies*, volume 21. Springer, 2009.

[18] Kofi Sarpong Adu-Manu, Nadir Adam, Cristiano Tapparello, Hoda Ayatollahi, and Wendi Heinzelman. Energy-harvesting wireless sensor networks (eh-wsns) a review. *ACM Transactions on Sensor Networks (TOSN)*, 14(2):1–50, 2018.

[19] Tom M Mitchell et al. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45(37):870–877, 1997.

[20] Tom Michael Mitchell. *The discipline of machine learning*, volume 9. Carnegie Mellon University, School of Computer Science, Machine Learning . . . , 2006.

[21] Nauman Ahad, Junaid Qadir, and Nasir Ahsan. Neural networks in wireless networks: Techniques, applications and guidelines. *Journal of network and computer applications*, 68:1–27, 2016.

[22] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, 2019.

[23] Sun-Chong Wang. Artificial neural network. In *Interdisciplinary computing in java programming*, pages 81–100. Springer, 2003.

[24] Georgios Papadopoulos. *Improving medium access for dynamic wireless sensor networks*. PhD thesis, Strasbourg, 2015.

[25] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.

[26] Geoffrey Werner-Allen, Konrad Lorincz, Mario Ruiz, Omar Marcillo, Jeff Johnson, Jonathan Lees, and Matt Welsh. Deploying a wireless sensor network on an active volcano. *IEEE internet computing*, 10(2):18–25, 2006.

[27] Ajith Kumar and S Ganesh Thorenoor. Analysis of ip network for different quality of service. In *International Symposium on Computing, Communication, and Control (ISCCC 2009) Proc. of CSIT*, volume 1, 2011.

[28] Edgar H Callaway. *Wireless sensor networks: architectures and protocols*. CRC Press, Inc., 2003.

[29] Gayoung Kim and Junho Jeong. Csma/ca channel hopping in iot environment toward intelligent multi-user diversity. *The Journal of Supercomputing*, pages 1–16, 2021.

[30] Giuseppe Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on selected areas in communications*, 18(3):535–547, 2000.

[31] Marta Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of the ieee 802.11 wireless local area network protocol. In *Process Algebra and Probabilistic Methods: Performance Modeling and Verification*, pages 169–187. Springer, 2002.

[32] Atul Kumar Pandey and Nisha Gupta. An energy efficient adaptive wake-up radio mac (eeawur-mac) protocol for iot wireless body area networks. *Wireless Personal Communications*, pages 1–25, 2021.

[33] Rudd JM Vullers, Rob Van Schaijk, Hubregt J Visser, Julien Penders, and Chris Van Hoof. Energy harvesting for autonomous wireless sensor networks. *IEEE Solid-State Circuits Magazine*, 2(2):29–38, 2010.

[34] Chin Keong Ho and Rui Zhang. Optimal energy allocation for wireless communications with energy harvesting constraints. *IEEE Transactions on Signal Processing*, 60(9):4808–4818, 2012.

[35] Ricky W Butler. What is formal methods? *NASA LaRC Formal Methods Program*, 2001.

[36] María Emilia Cambronero, Hermenegilda Macià, Valentín Valero, and Luis Orozco-Barbosa. Modeling and analysis of the 1-wire communication protocol using timed colored petri nets. *IEEE Access*, 6:27356–27372, 2018.

[37] Xiang Hu and Li Jiao. Efficient modeling and performance analysis for ieee 802.15. 4 with coloured petri nets. In *Quality of Service (IWQoS), 2017 IEEE/ACM 25th International Symposium on*, pages 1–6. IEEE, 2017.

[38] Jalel Ben-Othman, Serigne Diagne, Lynda Mokdad, and Bashir Yahya. Performance evaluation of a hybrid mac protocol for wireless sensor networks. In *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, pages 327–334. ACM, 2010.

[39] Mohammad Abdollahi Azgomi and Ali Khalili. Performance evaluation of sensor medium access control protocol using coloured petri nets. *Electronic Notes in Theoretical Computer Science*, 242(2):31–42, 2009.

[40] José Mauricio Neto, José Sérgio da Rocha Neto, Kyller Gorgônio, and Angelo Perkusich. Modeling csma-ca protocol with coloured petri nets for wireless sensor networks applications.

[41] Hammadi Bennoui. Interacting behavioral petri nets analysis for distributed causal model-based diagnosis. *Autonomous agents and multi-agent systems*, 28(2):155–181, 2014.

[42] Hasiba Ben Attia, Laid Kahloul, Saber Benhazrallah, and Samir Bourekkache. Using hierarchical timed coloured petri nets in the formal study of trbac security policies. *International Journal of Information Security*, 19(2):163–187, 2020.

[43] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 46–57. IEEE, 1977.

[44] E Allen Emerson. Temporal and modal logic. In *Formal Models and Semantics*, pages 995–1072. Elsevier, 1990.

[45] Lorenzo Giambagli, Lorenzo Buffoni, Timoteo Carletti, Walter Nocentini, and Duccio Fanelli. Machine learning in spectral domain. *Nature communications*, 12(1):1–9, 2021.

[46] Ikram Remadna, Labib Sadek Terrissa, Soheyb Ayad, and Noureddine Zerhouni. Rul estimation enhancement using hybrid deep learning methods. *International Journal of Prognostics and Health Management*, 12(1), 2021.

[47] Abir Belaala, Labib Sadek Terrissa, Noureddine Zerhouni, and Christine Devalland. Computer-aided diagnosis for spitzoid lesions classification using artificial intelligence techniques. *International Journal of Healthcare Information Systems and Informatics (IJHISI)*, 16(1):16–37, 2021.

[48] Zubair Md Fadlullah, Fengxiao Tang, Bomin Mao, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani. State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems. *IEEE Communications Surveys & Tutorials*, 19(4):2432–2455, 2017.

[49] Alex HB Duffy. The" what" and" how" of learning in design. *IEEE Expert*, 12(3):71–76, 1997.

[50] Pat Langley and Herbert A Simon. Applications of machine learning and rule induction. *Communications of the ACM*, 38(11):54–64, 1995.

[51] Yaser S Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning from data*, volume 4. AMLBook New York, NY, USA:, 2012.

[52] Laura Maruster. *A machine learning approach to understand business processes*. Citeseer, 2003.

[53] Reza Mehmood and Arvind Selwal. Fingerprint biometric template security schemes: attacks and countermeasures. In *Proceedings of ICRIC 2019*, pages 455–467. Springer, 2020.

[54] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

[55] Khadija El Bouchefry and Rafael S de Souza. Learning in big data: Introduction to machine learning. In *Knowledge Discovery in Big Data from Astronomy and Earth Observation*, pages 225–249. Elsevier, 2020.

[56] Mounib Khanafer, Mouhcine Guennoun, and Hussein T Mouftah. A survey of beacon-enabled ieee 802.15. 4 mac protocols in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 16(2):856–876, 2013.

[57] Faiza Nawaz and Varun Jeoti. Performance assessment of wirelesshart technology for its implementation in dense reader environment. *Computing*, 98(3):257–277, 2016.

[58] Youcef Hammal, Jalel Ben-Othman, Lynda Mokdad, and Abdelkrim Abdelli. Formal modeling and verification of an enhanced variant of the ieee 802.11 csma/ca protocol. *Journal of Communications and Networks*, 16(4):385–396, 2014.

[59] Zohra Hmidi, Laïd Kahloul, Saber Benharzallah, and Cherifa Othmane. Statistical model checking of csma/ca in wsns. In *VECoS*, pages 27–42, 2016.

[60] Zhi Chen, Ya Peng, and Wenjing Yue. Modeling and analyzing csma/ca protocol for energy-harvesting wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2015.

[61] Hmidi Zohra, Laid Kahloul, and Saber Benharzallah. Using priced timed automata for the specification and verification of csma/ca in wsns. *International Journal of Information and Communication Technology*, 17(2):129–145, 2020.

[62] Mouloud Atmani, Djamil Aïssani, and Yassine Hadjadj-Aoul. Towards bandwidth and energy optimization in ieee 802.15. 4 wireless sensor networks. *Computing*, 100(6):597–620, 2018.

[63] Manel Houimli, Laid Kahloul, and Sihem Benaoun. Formal specification, verification and evaluation of the mqtt protocol in the internet of things. In *2017 International Conference on Mathematics and Information Technology (ICMIT)*, pages 214–221. IEEE, 2017.

[64] Manel Houimli and Laid Kahloul. Modeling and performance evaluation of protocols in mobile wireless sensor networks. In *International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 328–339. Springer, 2017.

[65] Siham Zroug, Laid Kahloul, Saber Benharzallah, and Karim Djouani. A hierarchical formal method for performance evaluation of wsns protocol. *Computing*, 103(6):1183–1208, 2021.

[66] Reinhard German and Armin Heindl. Performance evaluation of ieee 802.11 wireless lans with stochastic petri nets. In *Petri Nets and Performance Models, 1999. Proceedings. The 8th International Workshop on*, pages 44–53. IEEE, 1999.

[67] Armin Heindl and Reinhard German. Performance modeling of ieee 802.11 wireless lans with stochastic petri nets. *Performance Evaluation*, 44(1-4):139–164, 2001.

[68] Kejie Lu, Jianfeng Wang, Dapeng Wu, and Yuguang Fang. Performance of a burst-frame-based csma/ca protocol: Analysis and enhancement. *Wireless Networks*, 15(1):87–98, 2009.

[69] Reinhard German. Spnl: Processes as language-oriented building blocks of stochastic petri nets. In *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 123–134. Springer, 1997.

[70] Armin Zimmermann, Jörn Freiheit, Reinhard German, and Günter Hommel. Petri net modelling and performability evaluation with timenet 3.0. In *International conference on modelling techniques and tools for computer performance evaluation*, pages 188–202. Springer, 2000.

[71] Aladdin Masri, Thomas Bourdeaud'Huy, and Armand Toguyeni. Performance analysis of ieee 802.11 b wireless networks with object oriented petri nets. *Electronic Notes in Theoretical Computer Science*, 242(2):73–85, 2009.

[72] A Haffiz Shuaib, Toktam Mahmoodi, and A Hamid Aghvami. A timed petri net model for the ieee 802.15. 4 csma-ca process. In *2009 IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1204–1210. IEEE, 2009.

[73] Teerawat Issariyakul and Ekram Hossain. Introduction to network simulator 2 (ns2). In *Introduction to network simulator NS2*, pages 1–18. Springer, 2009.

[74] Kurt Jensen. *Coloured Petri nets: basic concepts, analysis methods and practical use*, volume 1. Springer Science & Business Media, 2013.

[75] Luís Gomes and Joao Paulo Barros. Structuring and composability issues in petri nets modeling. *IEEE Transactions on Industrial Informatics*, 1(2):112–123, 2005.

[76] Max Kanovich and Takayasu Ito. Temporal linear logic specifications for concurrent processes. In *Proceedings of Twelfth Annual IEEE Symposium on Logic in Computer Science*, pages 48–57. IEEE, 1997.

[77] Giovanni Chiola, Giuliana Franceschinis, Rossano Gaeta, and Marina Ribaudo. Greatspn 1.7: graphical editor and analyzer for timed and stochastic petri nets. *Performance evaluation*, 24(1-2):47–68, 1995.

[78] M Ajmone Marsan, Gianfranco Balbo, Gianni Conte, Susanna Donatelli, and Giuliana Franceschinis. *Modelling with generalized stochastic Petri nets*, volume 292. Wiley New York, 1995.

[79] Zhenhua Yu and Yuanli Cai. Object-oriented petri nets based architecture description language for multi-agent systems. *IJCSNS*, 6(1):123–131, 2006.

[80] Rajeev Alur and David L Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.

[81] Gerd Behrmann, Alexandre David, and Kim G Larsen. A tutorial on uppaal. In *Formal methods for the design of real-time systems*, pages 200–236. Springer, 2004.

[82] Deepa Naik and Tanmay De. Machine learning application in the hybrid optical wireless networks. In *Machine Intelligence and Soft Computing*, pages 491–502. Springer, 2021.

[83] Simon Haykin and N Network. A comprehensive foundation. *Neural networks*, 2(2004):41, 2004.

[84] Mohammad Abu Alsheikh, Shaowei Lin, Dusit Niyato, and Hwee-Pink Tan. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials*, 16(4):1996–2018, 2014.

[85] Zhihao Guo and Behnam Malakooti. Delay prediction for intelligent routing in wireless networks using neural networks. In *2006 IEEE International Conference on Networking, Sensing and Control*, pages 625–630. IEEE, 2006.

[86] Luis Martín, Luis F Zarzalejo, Jesús Polo, Ana Navarro, Ruth Marchante, and Marco Cony. Prediction of global solar irradiance based on time series analysis: Application to solar thermal power plants energy production planning. *Solar Energy*, 84(10):1772–1781, 2010.

[87] Yu-Ju Shen and Ming-Shi Wang. Broadcast scheduling in wireless sensor networks using fuzzy hopfield neural network. *Expert systems with applications*, 34(2):900–907, 2008.

[88] Raghavendra V Kulkarni and Ganesh K Venayagamoorthy. Neural network based secure media access control protocol for wireless sensor networks. In *2009 international joint conference on neural networks*, pages 1680–1687. IEEE, 2009.

[89] Prowler: Probabilistic wireless network simulator. `http:https://www.isis.vanderbilt.edu/projects/nest/prowler/`. Accessed: Mars 3rd 2021.

[90] Fereshteh Khorasani and Hamid Reza Naji. Energy efficient data aggregation in wireless sensor networks using neural networks. *International Journal of Sensor Networks*, 24(1):26–42, 2017.

[91] Junfeng Xie, F Richard Yu, Tao Huang, Renchao Xie, Jiang Liu, Chenmeng Wang, and Yunjie Liu. A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(1):393–430, 2018.

[92] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *25th annual conference on neural information processing systems (NIPS 2011)*, volume 24. Neural Information Processing Systems Foundation, 2011.

[93] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International conference on database systems for advanced applications*, pages 214–228. Springer, 2016.

[94] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.