Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 285396, 11 pages
http://dx.doi.org/10.1155/2015/285396

Research Article

Probabilistic Model Checking: One Step Forward in Wireless Sensor Networks Simulation

José A. Mateo, Hermenegilda Macià, M. Carmen Ruiz, Javier Calleja, and Fernando Royo

Instituto de Investigación en Informática, Avenida de España s/n. 02071 Albacete, Spain

Correspondence should be addressed to José A. Mateo; jmateo@dsi.uclm.es

Received 3 August 2014; Accepted 31 October 2014

Academic Editor: Shigeng Zhang

A novel collision resolution algorithm for wireless sensor networks is formally analysed via probabilistic model checking. The algorithm called *2CS-WSN* is specifically designed to be used during the contention phase of IEEE 802.15.4. Discrete time Markov chains (DTMCs) have been proposed as modelling formalism and the well-known probabilistic symbolic model checker PRISM is used to check some correctness properties and different operating modes and, furthermore, to collect some performance measures. Thus, all the benefits of formal verification and simulation are gathered. These correctness properties as well as practical and relevant scenarios for the real world have agreed with the algorithm designers.

1. Introduction

The joint efforts of the Zigbee Alliance and the IEEE 802.15.4 Task Group have produced a set of protocols that ensure the functionality of wireless personal area networks (WPANs). IEEE 802.15.4 standard [1] defines the specification of the physical and media access control (MAC) layers for low-rate wireless personal area networks (LR-WPANs). These networks are convenient in scenarios where the availability of resources is limited. IEEE 802.15.4 supports star and peer-to-peer network topologies and uses carrier sense multiple access with collision avoidance (CSMA/CA) as medium access mechanism. Moreover, it provides two operating modes that may be selected by a central node: nonbeacon-enabled and beacon-enabled. They are used in nonslotted CSMA/CA and slotted CSMA/CA, respectively. When a device wishes to transfer data to a coordinator in a beacon-enabled network, it first listens for the network beacon. When the beacon is found, the device synchronises to the superframe structure. At the appropriate point, the device transmits its data frame, using slotted CSMA-CA, to the coordinator. The coordinator acknowledges the successful reception of the data by transmitting an optional acknowledgement frame. When a device wishes to transfer data in a nonbeacon-enabled network, it simply transmits its data frame, using unslotted CSMA-CA, to the coordinator. The coordinator acknowledges the successful reception of the data by transmitting an optional acknowledgement frame. The performance of slotted CSMA/CA algorithm has been analysed previously (e.g., [2, 3]) concluding that the binary exponential backoff algorithm is not flexible enough to be used in large-scale sensor networks.

We focus here on 2CS-WSN (two cells sorted wireless sensor network) algorithm [4], a simple, fast, and effective collision resolution method specifically designed to be used during the contention phase of IEEE 802.15.4. It is intended to be used as alternative to CSMA/CA. As 2CS-WSN uses probabilities and sorted transmissions for quick collision resolution, there is a clear need to inspect how those parameters can be tuned so as to achieve performance improvements as well as to detect possible inconsistencies or issues (e.g., deadlocks).

From our experience, we advocate for the use of simulation and formal verification techniques to analyse protocols or algorithms since there is an eternal debate about the appropriateness of using simulation or formal verification in not only this area. On the one hand, simulation-based approaches study in a nonexhaustive way the behaviour of the system. On the other hand, formal verification is based on a systematic and exhaustive analysis of *all the possible paths* in the system, trying to find possible inconsistencies and/or errors not evaluated in the simulation. Obviously, each of them has its advantages and disadvantages and it is out of the scope of this paper to summarise them, but, from our experience, it is better to use formal verification up until the problem of "state explosion" arises and, then, use simulation to obtain results in bigger scenarios.

Here, we use probabilistic model checking (a formal method for the verification of probabilistic systems) since the use of probabilities can influence the behaviour of 2CS-WSN. In particular, we describe 2CS-WSN algorithm in terms of discrete time Markov chains (DTMCs) and, using the well-known probabilistic symbolic model checker PRISM [5], we verify some correctness properties, compare different operating modes of the algorithm, and analyse the performance and accuracy of different model abstractions. It is clear that the resolution of a collision in minimum time is a primary requirement in these kinds of algorithms and therefore our performance analysis is mainly focused on temporal aspects. By adding the *incurred time costs* during the execution of the system, we can evaluate the expected time to resolve the collision in different scenarios. In addition to this, we are able to study properties of great interest to designers such as "the probability that a certain number of nodes have successfully managed to transmit within a certain time" or "the probability that all nodes have transmitted before a certain time." Analysing such properties for a range of parameter values (e.g., retransmission probability) is often key to identify interesting or anomalous behaviour, and, probably the most important issue the designer can determine if the algorithm fulfils the timing requirements.

The rest of the paper is organised as follows. As usual, we first introduce some related works and compare it with our work. We continue by presenting some background required for a better understanding of this work. Thus, we describe the algorithm under study in Section 3, and some formal background in Section 4. After that, we show the PRISM model for 2CS-WSN in Section 5, and we study it in Section 6. Finally, we summarise some conclusions and discuss possible future work.

2. Related Works

Recently, the analysis of WSNs has attracted a lot of attention and, therefore, there are many ways to present these works. We divide such works in two main categories (simulation-based and formal verification-based) since both techniques are used here.

To begin with, we present those works that are based on simulation. It is worthwhile to mention here that some of them use simulation to demonstrate the correctness of their analytical approach; that is, an analytical model of the protocol/algorithm is introduced and, then, some experiments are conducted in a well-known (or ad hoc) simulator to validate the correctness of the analytical model. For instance, Bianchi defines in [6] an analytical evaluation of the saturation throughput of the 802.11 distributed coordination function. As in our work, the author uses a Markov chain to model the behaviour of a single node and assumes an ideal channel. In [7], Ye et al. introduce S-MAC, a medium access control protocol designed for wireless sensor networks, and validate it in a real testbed. Faridi et al. [8] characterise the key metrics in a beacon-enabled IEEE 802.15.4 system with retransmissions, and, in [9], Lee et al. propose an additional carrier sensing algorithm based on the IEEE 802.15.4 acknowledgement mode to detect the channel condition. Then, a Markov chain model is depicted, analysing the throughput of the algorithm by means of an ad hoc experimentation. Finally, Hoesel and Havinga [10] develop a novel lightweight medium access protocol (LMAC) for wireless sensor networks. In addition, they validate the algorithm in the simulation package OMNet++ enriched with a framework for WSNs.

On the other hand, one can use formal verification to verify the correctness of a system. A well-known problem when using formal verification is that it becomes intractable when the possible paths in the model are infinite. For example, in [11], it is modelled and analysed LMAC [10] by using timed automata and the popular model checker UPPAAL [12]. In [10], they are able to analyse networks with up to 5 nodes, whereas we are able to analyse bigger networks (up to 40 nodes). In [13], LMAC is studied as a case study to present a new version of UPPAAL, SMC-UPPAAL. The novelty here is that they apply statistical model checking to LMAC. Roughly speaking, the substantial difference between simulation and statistical model checking is that the latter one obtains the probability that the system behaves in such a manner. Again, small networks (up to 10 nodes) are studied. Next, we cite two works fairly related to the present one. On the one hand, Duflot et al. [14] evaluate CSMA/CD by using probabilistic timed automata and two well-known tools, PRISM and APMC [15]. With PRISM, they study the system using probabilistic model checking, whereas with APMC they approximate other properties. On the other hand, Kwiatkowska et al. [16] pose the automatic verification of a medium access control protocol of the IEEE 802.11 WLAN standard using probabilistic model checking. They use probabilistic timed automata as modelling formalism and PRISM as model checker. Finally, let us note that we have previous experience analysing wireless algorithms. For instance, we studied a recent role-based routing algorithm (NORA) for WSNs in [17], and its fuzzy-logic based version in [18]. Moreover, we would like to note that our paper is, to the best of our knowledge, the first work that achieves applying probabilistic model checking to networks up to 40 nodes in conflict since the related works presented in this section only success to model networks with at most 10 nodes.

3. 2CS-WSN: Random Access with Stack Protocols

Before we begin, let us remark that 2CS-WSN algorithm is partially derived from the definition of the stack algorithm described in [19]. In the following, we will refer to it as 2C algorithm. It is a fair, efficient, and simple algorithm to resolve the possible collision when sharing the same transmission channel and it is called a stack protocol because its time evolution can be easily visualised as a group of stations moving up and down in a two-cell stack; that is, stations may be either transmitting or waiting, and these two states can be

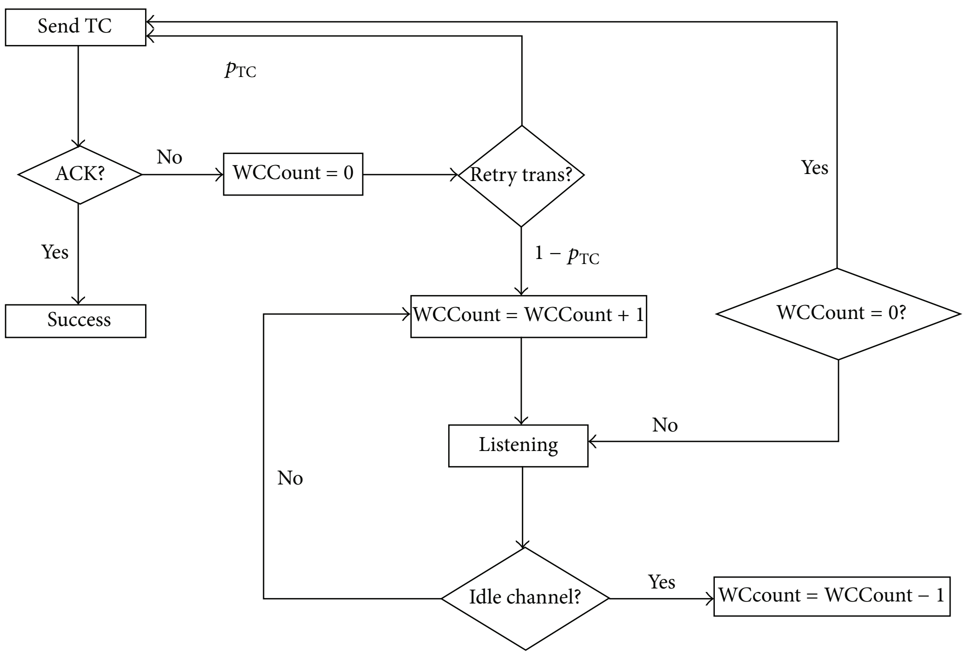


Figure 1: The 2CS-WSN algorithm.

represented using only two cells in a stack. The transmission cell (TC) represents the group of transmitting stations and the waiting cell (WC) the group of stations that have deferred transmission. Although 2C algorithm has many desirable features it may incur in significant access delays when a large number of stations contend for the channel since, with only two cells, it takes a long time to randomly distribute the stations. Thus, 2C algorithm was improved in [4], leading to the definition of 2CS-WSN algorithm, where wireless communication and several cells are considered. Moreover, there are two main features that share in common 2C and 2CS-WSN. First, collision resolution is performed by using probabilities and, second, time is slotted, allowing stations to transmit only at the beginning of a time slot. A time slot is normally considered as the time a station needs to transmit a packet and receive a feedback message from a central station. The feedback message is binary; that is, it is a C (collision) message when a collision was detected and a NC (no collision) message otherwise. If only one station transmitted, the corresponding packet will be successfully transmitted. On the other hand, if there were several transmission attempts in the same slot, there will be a collision and its resolution shall begin in the following slot. The collision resolution ends when all colliding stations can successfully transmit. This time interval is known as a collision resolution interval (CRI). A station that generates a new transmission request, when a CRI is in progress, has to wait until the current CRI ends before attempting to access the channel. Thus, 2C (and 2CS-WSN) are able to provide some fairness in the access to the channel since all the participants will eventually transmit.

As commented previously, 2CS-WSN is designed to be used with wireless communications although the original description of 2C algorithm is not tied to any specific transmission medium. Therefore, it has to be adapted to the particularities of the wireless medium. For instance, in 2C, it is assumed that there is a central station that is continuously monitoring the channel and providing feedback messages. However, in self-configuring wireless ad hoc networks, this assumption is unrealistic. In this case, the participants have to assume this role by monitoring the transmission medium and reacting accordingly. This leads to a second issue: how to detect a collision. In wired networks it is rather easy to detect a collision, but in wireless networks this is not a trivial matter. In 2CS-WSN, instead of detecting implicitly a collision, network nodes infer that a collision has happened. A wireless node can infer that its transmission has collided if the reply to its request does not arrive. In this case, the station has to randomly choose whether to retransmit (i.e., to remain in TC) or to join the waiting group (WC). We model this fact by using probabilities. Let us denote by p_{TC} the probability of remaining in TC and by p_{WC} the probability of moving to the first waiting group (with the obvious condition that $p_{WC} = 1 - p_{TC}$). We suppose here that all nodes are provided with an unbiased coin to make the decision; that is, they stay in TC with probability 0.5 or they move to WC with the same probability. Figure 1 shows the flow chart of the 2CS-WSN algorithm.

For instance, we show in Figure 2 how 2CS-WSN behaves in a five-node network, where all nodes want to transmit in the same slot and a collision occurs. To solve this collision, each node uses its unbiased coin to decide its following step. For instance, nodes 1 and 5 decide to enter the waiting group WC_1 whereas nodes 2, 3, and 4 decide to remain in the transmission group TC. Therefore, in the next slot, nodes 2, 3, and 4 attempt to transmit and collide again. Then, let us suppose that nodes 3 and 4 decide to enter the waiting group WC_1.

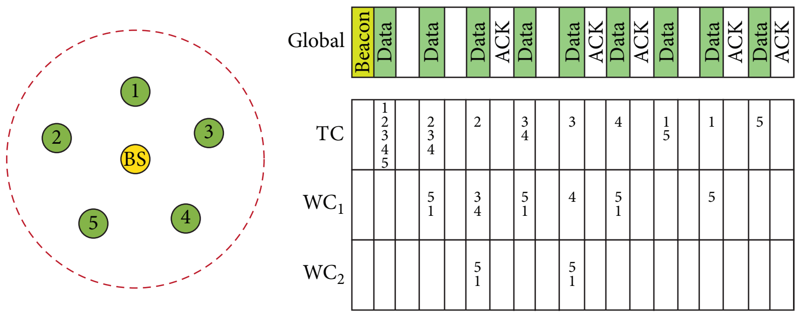


FIGURE 2: Collision resolution example using 2CS-WSN algorithm with a five-node network.

Nodes 1 and 5 move from WC_1 to WC_2. At this time only node 2 is in TC thus achieving a successful transmission. This successful transmission causes nodes in WC_1 (i.e., nodes 3 and 4) to move to TC and nodes in WC_2 (i.e., nodes 5 and 1) to move to WC_1. This process is repeated until all nodes that participated in the initial collision can successfully transmit.

4. Formal Background

Now, we introduce briefly some formal background. We start by defining discrete time Markov chains (DTMCs) as it has been used as modelling formalism. Next, we briefly introduce probabilistic model checking and PRISM.

4.1. Discrete Time Markov Chain. Basically, a Markov process is a special class of stochastic process that satisfies the Markov property (or memoryless), that is, given the state of the process at time t, the future behaviour after t is independent of the behaviour before t. When it is considered discrete state (sample) space, they are called Markov chains and if one considers only discrete time steps, they are called discrete time Markov chains (DTMC). Moreover, if the conditional probability is invariant with respect to the time origin, then the DTMC is said to be time-homogeneous. We only consider time-homogeneous DTMC in this paper. For more details see [20].

Definition 1. (i) A stochastic process $\{X_n,\ n = 0, 1, 2, \ldots\}$ on discrete state space S is said to be a discrete time Markov chain (DTMC) if, for all i_k in S,

$$\begin{aligned} &P\left(X_n = i_n \mid X_{n-1} = i_{n-1}, X_{n-2} = i_{n-2}, \ldots, X_0 = i_0\right) \\ &\quad = P\left(X_n = i_n \mid X_{n-1} = i_{n-1}\right) \end{aligned} \tag{1}$$

(memoryless property or Markov property).

(ii) A DTMC is said to be time-homogeneous if, for all $n = 0, 1, \ldots$ and for all i, j in S,

$$P\left(X_n = j \mid X_{n-1} = i\right\} = P\left(X_1 = j \mid X_0 = i\right) = p_{ij}. \tag{2}$$

In this way, p_{ij} represents the probability that the process will, when in state i, next make a transition into state j; that is, p_{ij} is the probability to move from the state i to the state j in one step.

(iii) The matrix of transitions probabilities P (stochastic matrix) of a time-homogeneous DTMC is defined as

$$P = \begin{pmatrix} p_{11} & p_{12} & p_{13} & \cdots \\ p_{21} & p_{22} & p_{23} & \cdots \\ \vdots & \vdots & \vdots & \cdots \\ p_{k1} & p_{k2} & p_{k3} & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}, \tag{3}$$

where each p_{ij} is the probability of moving from state i to state j. Since the probabilities are positive numbers and due to the fact that the process must take a transition into some state, we have that

$$\begin{aligned} &0 \le p_{ij} \le 1, \quad \forall i, j \in S, \\ &\sum_{j\in S} p_{ij} = 1 \quad \forall i \in S. \end{aligned} \tag{4}$$

Moreover, according to Chapman-Kolmogorov equations, the probability to reach the state j from the state i in n steps, denoted by $p_{ij}^{(n)}$, is the element (i, j) of the matrix P^n.

The behaviour of a DTMC is fully probabilistic; thus we can define a probability space over infinite paths through the model and it is possible to compute the probability of a particular event.

A DTMC can be also defined as a triple (S, s_0, P), where S is the set of states, s_0 is the initial state, and P is the stochastic matrix. And a DTMC can be also represented by a *state transition diagram*, which is a directed graph where each node is a state (number of nodes = number of states if S is finite), and there is an arc from i to j if and only if $p_{ij} > 0$. In this way, a state j is accessible from i_0 if there is a walk in the graph from i_0 to j, that is, an ordered string of nodes, $(i_0, i_1, \ldots, i_n, j)$, $n \ge 0$, in which there is a directed arc from i_k to i_{k+1} and from i_n to j.

4.2. Probabilistic Model Checking. Probabilistic model checking is a formal verification technique for the automatic analysis of systems that exhibit stochastic behaviour. It provides the likelihood of the occurrence of certain events. In conventional model checkers, it is used as input of the model of the system, represented in some formalism, and its specification, usually a formula in some temporal logic. After computing the formula in the model, one gets as output "yes" or "no," indicating whether or not the model satisfies it. Probabilistic model checking involves also reachability analysis in the state space, and the calculation of probabilities through appropriate numerical or analytical methods. The algorithms for probabilistic model checking are usually derived from conventional model checking, numerical linear algebra, and standard techniques for Markov chains. In this way, probabilistic model checking can be used to ascertain not only correctness, but also quantitative measures such as performance and reliability.

Probabilistic model checking can be applied to a range of probabilistic models, typically variants of Markov chains. The specification language is a probabilistic temporal logic, capable of expressing temporal relationships between events and likelihood of events. Probabilistic temporal logics are usually obtained from standard temporal logics by replacing the standard path quantifiers with a probabilistic quantifier. In this paper we use probabilistic computation tree logic (PCTL) [21] as probabilistic temporal logic, which is based on the well-known branching-time computation tree logic (CTL) [22]. It allows us to verify properties such as if the model "finishes or not properly" (all nodes have successfully transmitted) and/or to reason about quantitative measures such as "what is the probability that a certain number of nodes have successfully managed to transmit within a certain time" or "what is the probability that all nodes have transmitted before a certain bound" or "the expected time that all nodes have transmitted" and so on.

4.3. PRISM. PRISM [5] is an open source probabilistic model checker developed initially at the University of Birmingham and currently maintained and extended at the University of Oxford. PRISM supports several types of probabilistic models such as discrete time Markov chains (DTMCs), continuous time Markov chains (CTMCs), Markov decision processes (MDPs), probabilistic automata (PAs), and probabilistic timed automata (PTAs), considering also extensions of these models with costs and rewards.

Models are described using the PRISM modelling language, a state-based language based on *reactive modules*. The fundamental components of the PRISM language are modules and variables. A model is a set of modules which can interact with each other. Typically, a probabilistic model is constructed in PRISM as the parallel composition of a set of modules. In every state of the model, there is a set of commands (belonging to any of the modules) which are enabled, that is, whose guards are satisfied in that state. The choice between which command is performed (i.e., the scheduling) depends on the model type. PRISM includes also support for the specification and analysis of properties based on rewards (and costs). Thus, it is possible to assign different rewards to different states or transitions, depending on the values of model variables in each one.

PRISM provides also support for the automatic analysis of a wide range of quantitative properties. The property specification languages provided by PRISM are PCTL, CSL, LTL, and PCTL*, as well as extensions for quantitative specifications and costs/rewards. One of the key features of PRISM is its symbolic implementation technique. It uses data structures based on binary decision diagrams (BDDs), which allow compact representation and efficient manipulation of extremely large probabilistic models by exploiting structure and regularity derived from their high level description. As a proof of maturity, PRISM has been used to analyse systems from many different application domains, including communication and multimedia protocols (see the PRISM website [23] for multiple examples).

5. Modelling 2CS-WSN in PRISM

A model of 2CS-WSN in terms of DTMCs has been developed using PRISM language. In Section 3, we showed how 2CS-WSN behaves in a network with five colliding nodes and we will leverage this example to show how the algorithm has been modelled in PRISM. We recall that this does not mean that the network has only 5 nodes, but, among the nodes in the network, there are 5 trying to transmit at the same slot.

Box 1 shows the PRISM model for this scenario. A state consists of a triple (TC, WC_1, WC_2) where TC is the number of nodes in collision and WC_1, WC_2 are the number of nodes waiting to retransmit in each waiting cell. This initial state of the model is represented by the triple $(5, 0, 0)$, meaning that there are five nodes in TC and zero nodes in each one of the waiting cells. On the other hand, Figure 3 shows the state transition diagram associated to the DTMC described in Box 1. It consists of 42 states. Each state is represented as a rectangle with an identifier and a triple $(\mathrm{TC}, \mathrm{WC}_1, \mathrm{WC}_2)$. Each directed arc from the state i to the state j is labelled with the probability p_{ij}, which indicates the probability of moving from the state i to the state j according to the 2CS-WSN algorithm.

The initial state is set to the node number 41 labelled with the tuple $(5, 0, 0)$. The node number 0 is the final state and it is labelled with $(0, 0, 0)$, that is, no nodes either in TC or in WC_i, meaning that the five nodes have been able to transmit successfully and the initial collision has been resolved.

Carefully analysing the state transition diagram, it can be appreciated that the DTMC generates all possible alternatives to solve this collision. In particular, in Figure 3, the trace followed in Example 2 has been pointed out with dashed lines.

Hence, for instance, the probability to move in one step from the initial node $(5, 0, 0)$ (node number 41 in Figure 3) to the node $(3, 2, 0)$ (node number 37 in Figure 3) is obtained by considering that 3 of the 5 nodes in the transmission cell remain in it and the other 2 move to the first waiting cell. Let the probability to retransmit of each node be 0.5 (i.e., to remain in TC); then the probability to move is computed as $\binom{5}{2} \cdot 0.5^5 = 0.3125$. Notice that the model takes into account any 2 nodes taken from TC, and not only nodes 1 and 5 which were chosen in the example of Figure 2.

Once the model has been explained with a specific example, this model can be generalised as follows.

Let n be the number of nodes in collision and m the number of waiting cells. A state is defined as a tuple $(\mathrm{TC}, \mathrm{WC}_1, \mathrm{WC}_2, \ldots, \mathrm{WC}_m)$ where, initially, $\mathrm{WC}_j = 0, \forall j \in [1, \ldots, m]$. Therefore, the initial state is $(n, 0, 0, \ldots, 0)$. If $\mathrm{TC} = k$, with $k \in [2, \ldots, n]$, and there are $k + 1$ different

```
const nnodes = 5;
module T1
          TC: [0. . . nnodes] init nnodes;
          WC1: [0. . . nnodes] init 0;
          WC2: [0. . . nnodes] init 0;
          [] TC = 0 -> (TC′ = WC1) & (WC1′ = WC2) & (WC2′ = 0);
          [] TC = 1 -> (TC′ = WC1) & (WC1′ = WC2) & (WC2′ = 0);
          [] TC = 2 -> 0.25: (TC′ = 2) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 0) +
                        0.5: (TC′ = 1) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 1) +
                        0.25: (TC′ = 0) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 2);
          [] TC = 3 -> 0.125: (TC′ = 3) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 0) +
                        0.375: (TC′ = 2) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 1) +
                        0.375: (TC′ = 1) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 2) +
                        0.125: (TC′ = 0) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 3);
          [] TC = 4 -> 0.0625: (TC′ = 4) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 0) +
                        0.25: (TC′ = 3) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 1) +
                        0.375: (TC′ = 2) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 2) +
                        0.25: (TC′ = 1) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 3) +
                        0.0625: (TC′ = 0) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 4);
          [] TC = 5 -> 0.03125: (TC′ = 5) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 0) +
                        0.15625: (TC′ = 4) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 1) +
                        0.3125: (TC′ = 3) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 2) +
                        0.3125: (TC′ = 2) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 3) +
                        0.15625: (TC′ = 1) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 4) +
                        0.03125: (TC′ = 0) & (WC2′ = min (WC2 + WC1, nnodes)) & (WC1′ = 5);
```

Box 1: DTMC model for 5 nodes and 2 WCs.

possibilities for the number of nodes that remain in TC, then the behaviour of variable TC follows a binomial distribution $B = (k; p)$, where $p = p_{\mathrm{TC}}$. Whenever $k \geq 2$, nodes in WC_i move to WC_{i+1} with $i \in [2, \ldots, m-1]$ in the next step. Nodes in the last waiting cell make no movement. Nodes in TC choose to retransmit with probability p_{TC} or to move to WC_1 with probability $1 - p_{\mathrm{TC}}$. Let α be the number of nodes that remain in TC and β the number of nodes that move to WC_1, where $\alpha + \beta = k$. The probability that α nodes remain in TC is defined as $\binom{k}{\alpha} \cdot (p_{\mathrm{TC}})^{\alpha}(1 - p_{\mathrm{TC}})\beta$ with $\alpha \in [0, \ldots, k]$ and $\beta = k - \alpha$. If $p_{\mathrm{TC}} = 0.5$, the probability is $\binom{k}{\alpha} \cdot (0.5)^k$. Whenever $k \leq 1$ nodes in WC_i move to WC_{i-1} with $i \in [2, \ldots, m]$ and the last waiting cell must be empty. Therefore, next state z is defined as follows:

$$z = \begin{cases} (\mathrm{WC}_1, \mathrm{WC}_2, \ldots, \mathrm{WC}_m, 0) & \text{if } k \leq 1 \\ (\alpha, \beta, \mathrm{WC}_1, \ldots, \mathrm{WC}_{m-1}, \mathrm{WC}_{m-1} + \mathrm{WC}_m) & \text{if } k \geq 2, \end{cases} \tag{5}$$

where k is the number of nodes in TC and $k = \alpha + \beta$.

6. Verification and Performance Evaluation

To evaluate the DTMC model in PRISM, the following parameters have been considered. 40 nodes initially in collision ($n = 40$), up to 5 waiting cells ($m = 2, \ldots, 5$), and the probability of retransmit vary from 0.1 to 0.9 ($p_{\mathrm{TC}} = 0.1, \ldots, 0.9$). In particular, considering $m = 5$ and $p_{\mathrm{TC}} = 0.5$, the PRISM model consists of 8.996.429 states and 61.435.101 transitions. The time needed in PRISM to build this model is about 20 secs on a 3.2 GHz PC with 16 GB RAM.

As commented previously, we have used PCTL logic for expressing significant properties. First of all, we want to check if the model eventually finishes. To this end, we need to know how many nodes have successfully transmitted (finish). This is computed by using the following formula, where m is the number of the waiting cells:

$$\boxed{\text{formula finish} = n\text{nodes} - (\mathrm{TC} + \mathrm{WC}_1 + \cdots + \mathrm{WC}_m).} \tag{6}$$

This formula computes the number of finished nodes. Thus, *n*nodes is the number of nodes initially in collision (we consider 40 in this scenario), and TC, $\mathrm{WC}_1, \ldots, \mathrm{WC}_{m-1}$, or WC_m represent the number of nodes in such a cell.

By using this formula, we can evaluate the following property ϕ_1:

$$\phi_1 \equiv [P = 1\, [F\, (\text{finish} = 40)]]. \tag{7}$$

If this property holds, we can ensure that the algorithm eventually terminates successfully. The result of the evaluation was *true* and, therefore, we can conclude that the model eventually finishes. This property turns out to be of great interest for some scenarios (emergencies, control sensors, etc.) as it ensures that all the nodes can eventually transmit in contrast to CSMA-based protocols where a backoff period is used for channel contention and some threshold will decide if transmission is rejected. These protocols cannot guarantee channel access for all nodes [24].

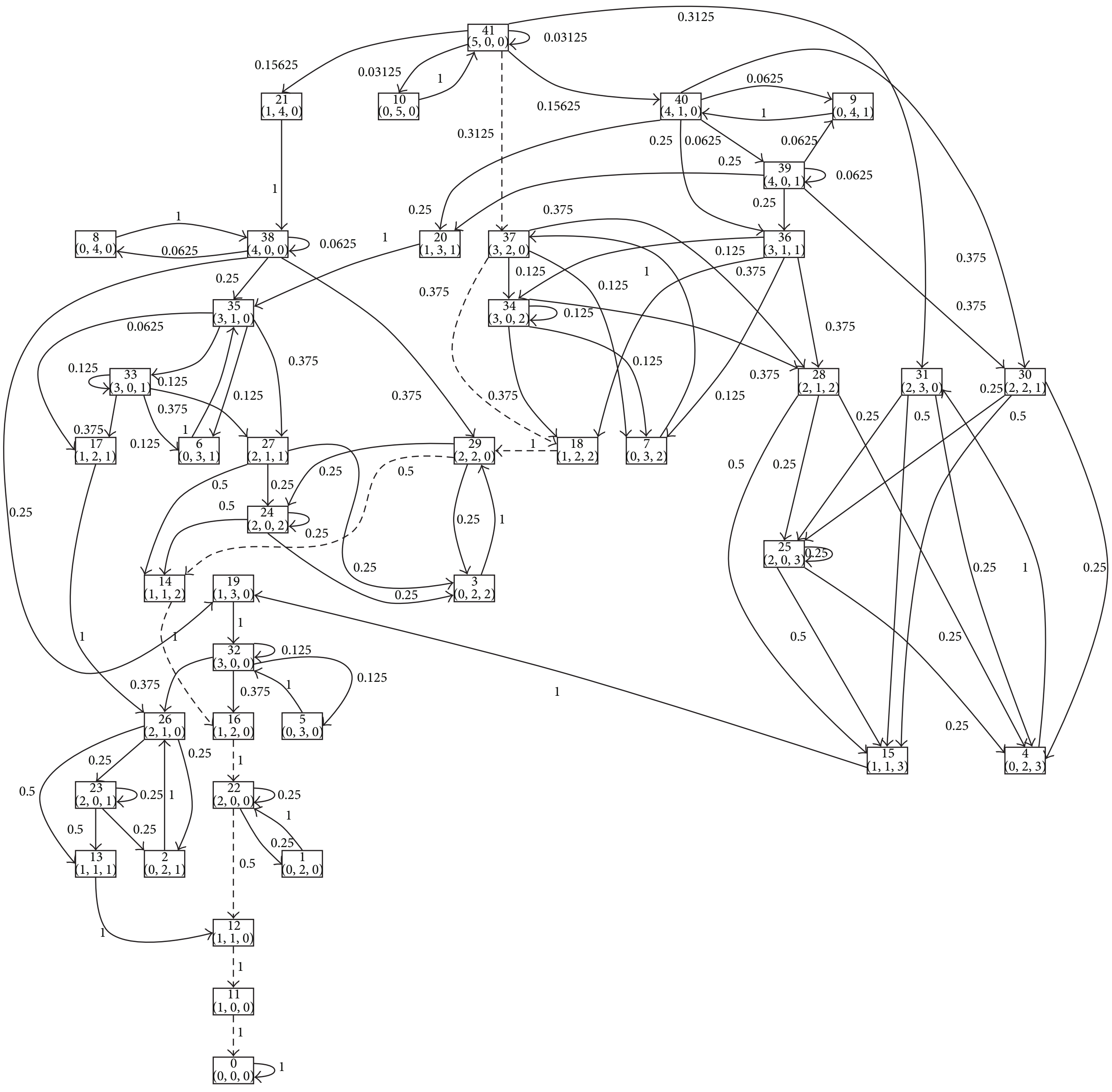
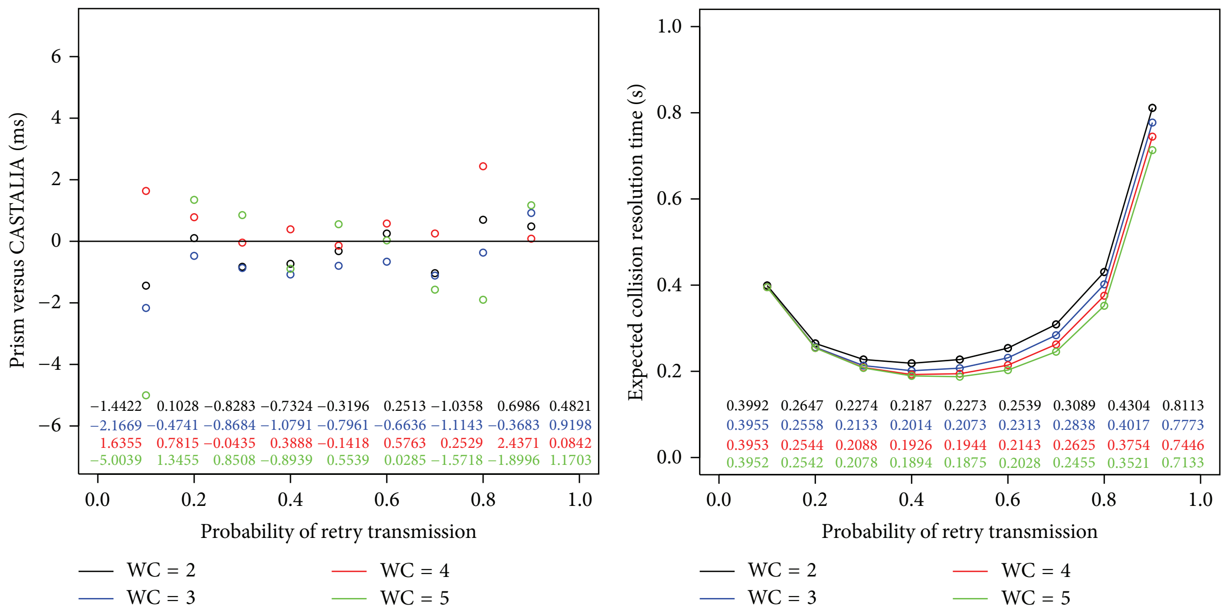


Figure 3: DTMC for 5 nodes in collision and 2 WCs.

Once we know that the algorithm eventually finishes, we turn our attention to collect performance results about collision resolution time. To this end, we extend first our model with rewards so that a real value (reward) is associated with certain transitions of the model. In this case, and to be in accordance with [4], we defined a time slot of 1.6 ms; that is, each transition in our model has a *reward* associated of 1.6. In PRISM, we can analyse properties about the expected values of these rewards. This is achieved using the R operator. In particular, we use "cumulative reward" property that also associates a reward with each path of a model (not only to states/transitions). In this case, we evaluate the expected conflict resolution time using ϕ_2. We ran then PRISM experiments on ϕ_2, with verification (we compute the whole state space),

$$\phi_2 \equiv [R\,\{\text{time}\} =?\,[F\,(\text{finish} = 40)]] \qquad (8)$$

varying the probability of transmission (p_{TC}) and/or the number of waiting cells according to Table 1. The field *Step value* represents how much we vary the parameter in each experiment. The goal of this experiment is to demonstrate what is the best configuration for the algorithm since, unfortunately, the designers of 2CS-WSN (and 2C) have never studied how these parameters could affect its performance. Figure 4(b) (or Table 2) shows that the best choice in order to



(a) Differences PRISM versus CASTALIA results (ms)

(b) Collision resolution time (s) for PRISM

FIGURE 4: Comparison CASTALIA versus PRISM.

TABLE 1: Parameter values.

Parameter	Initial value	Final value	Step value
p_{TC}	0.1	0.9	0.1
Number of waiting cells	2	5	1

TABLE 2: Expected collision resolution time (s) (Figure 4(b)).

Prob. trans.	2 WCs	3 WCs	4 WCs	5 WCs
0.1	0.3992	0.3955	0.3953	0.3952
0.2	0.2647	0.2558	0.2544	0.2542
0.3	0.2274	0.2133	0.2088	0.2078
0.4	0.2187	0.2014	0.1926	0.1894
0.5	0.2273	0.2073	0.1944	0.1875
0.6	0.2539	0.2313	0.2143	0.2028
0.7	0.3089	0.2838	0.2625	0.2455
0.8	0.4304	0.4017	0.3754	0.3521
0.9	0.8113	0.7773	0.7446	0.7133

minimise the collision resolution time is considering $p_{TC} = 0.5$ as the parameter of probability of retransmission and 5 as the number of waiting cells. Surprisingly, the designers of 2C and 2CS-WSN chose the best option without studying the possible configurations. In contrast, in 2CS-WSN, they assumed that the number of waiting cells is unlimited (this assumption is not realistic in the real world since the resources are limited). Moreover, the results obtained in [19] for 2C algorithm can be improved considering 4 waiting cells instead of 2 (see Table 2).

To validate the conformity of our proposed probabilistic model with the model proposed in [4], we compare the results obtained here with PRISM and the results obtained in

TABLE 3: Differences PRISM versus CASTALIA results (ms) (Figure 4(a)).

Prob. trans.	2 WCs	3 WCs	4 WCs	5 WCs
0.1	−1.4422	−2.1669	+1.6355	−5.0039
0.2	+0.1028	−0.4741	+0.7815	+1.3455
0.3	−0.8283	−0.8684	−0.0435	+0.8508
0.4	−0.7324	−1.0791	+0.3888	+0.8939
0.5	−0.3196	−0.7961	−0.1418	+0.5539
0.6	+0.2513	−0.6636	+0.5763	+0.0285
0.7	−1.0358	−1.1143	+0.2529	−1.5718
0.8	+0.6986	−0.3683	+2.4371	−1.8996
0.9	+0.4821	+0.9198	+0.0842	+1.1703

the simulator CASTALIA. In Figure 4(a) (or Table 3) we can observe that the differences between the collision resolution time obtained taking 1000 simulations in CASTALIA and the expected time obtained by using PRISM is, in the worst case, about 5 ms, and normally less than 2 ms. Therefore, we can conclude undoubtedly that our model fits adequately.

Furthermore, we can also use time-bounded probabilistic reachability properties. The main difference with other kind of properties is that one can associate a strict time deadline to relevant events. As it can be observed in Figure 4(b), the four best probabilities of retransmission are $p_{TC} = 0.3, 0.4, 0.5, 0.6$. Therefore, we ran PRISM experiments on ϕ_3 (using verification),

$$\phi_3 \equiv [P =? [F \leq T (\text{finish} = 40)]] \tag{9}$$

varying the retransmission probability parameter from 0.3 to 0.6 and the deadline time (T) from 0 to 400 (ms). Figure 5 (or Table 4) shows the results obtained. This property calculates

Table 4: Probability to solve all the collisions varying p_{TC} and the deadline time (Figure 5).

Finish time (ms)	$p_{TC} = 0.3$	$p_{TC} = 0.4$	$p_{TC} = 0.5$	$p_{TC} = 0.6$
≤128	0	0	0	0
144	0.0002	0.0026	0.0042	0.0005
160	0.0095	0.0592	0.0763	0.0187
176	0.0803	0.0279	0.3175	0.1229
192	0.2734	0.5958	0.6323	0.3542
208	0.5402	0.8334	0.8547	0.6274
224	0.7682	0.9474	0.9562	0.8319
240	0.9049	0.9867	0.9895	0.9396
256	0.9673	0.9972	0.9979	0.7133
272	0.9903	0.9994	0.9996	0.9823
288	0.9974	0.9999	0.9999	0.9957
304	0.9994	0.9999	0.9999	0.9991
≥320	0.9999	0.9999	0.9999	0.9999

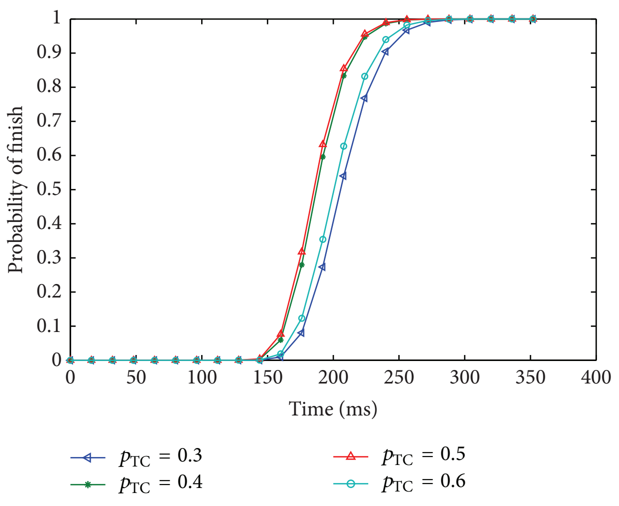


Figure 5: Success probability for $p_{TC} = 0.3, 0.4, 0.5, 0.6$.

the probability that all nodes have successfully within the deadline T. Observe that we have chosen 400 ms since after this time all the nodes have successfully transmitted. For example, the probability to solve all the collisions in less than 128 ms is nearly 0 and practically 1 considering 320 ms or more. Besides, if we fix $T = 240$ ms, we achieve to solve the collision with at least a probability of 0.98 ($p_{TC} = 0.4$ or $p_{TC} = 0.5$) and with at least a probability of 0.9 ($p_{TC} = 0.3$). Observe also that this probability is almost 0.94 considering $p_{TC} = 0.6$. Finally, note that this study was not conducted in [4, 19].

Finally, considering the best option ($p_{TC} = 0.5$ and the number of waiting cells as 5), we can ask about the probability that N nodes have transmitted within T ms. To answer this question we ran again PRISM experiments, using verification, on ϕ_4,

$$\phi_4 \equiv [P =? [F \leq T (\text{finish} = N)]], \tag{10}$$

where we change the number of nodes (N) from 5 to 40 and the deadline (T) from 0 to 300 ms. Figure 6 (or Table 5)

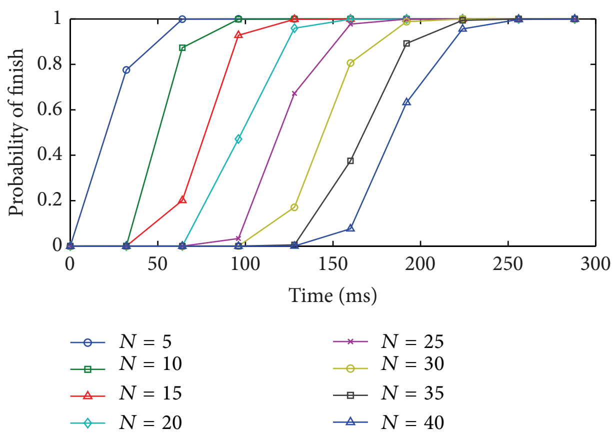


Figure 6: Partial success with $p_{TC} = 0.5$.

shows the results. For instance, considering 160 ms, the probability that at least 20 nodes had transmitted within 160 ms is more than 0.99, the probability that 25 nodes had transmitted within 160 ms is less than 0.9, and the probability that 35 nodes had transmitted within these 160 ms is less than 0.1. This kind of questions could help in critical situations, where a fixed number of nodes must transmit to inform, for instance, about an emergency.

7. Conclusions

This paper presents the formal modelling and initial validation of a novel collision resolution algorithm for wireless sensor networks. 2CS-WSN is specifically designed to be used during the contention phase of IEEE 802.15.4. In our study, we try to find any error/inconsistency presented in the specification of the algorithm, and we have evaluated some properties for nontrivial, practical, and relevant scenarios.

In detail, we present the specification of the 2CS-WSN algorithm in terms of DTMCs and perform probabilistic model checking by using the well-known tool PRISM.We have used PCTL to formulate relevant properties. Forinstance, we have checked the absence of deadlock and theconformity with the former implementation in CASTALIA. We have focused here on temporal parameters since they are of great interest for the algorithm designers. In particular, we have studied the expected collision resolution time and properties that cannot be evaluated with general simulators such as “the probability that a certain number of nodes have successfully transmitted within a certain time” or “the probability that all nodes have transmitted before a certain time.” Furthermore, we have found the best configuration ($p_{TC} = 0.5$ and $m = 5$) for the algorithm.

Now, our next step is aimed at finding possible improvements to the algorithm and, thus, we are collaborating with the designers in future versions of 2CS-WSN. For instance, we want to evaluate the effect of using adaptive probabilities (each node can use its own transmission probabilities regarding some parameter we are studying or we can use probabilities to move up in the stack instead of using only when

Table 5: Probability to solve N collisions varying the deadline time (Figure 6).

Finish time (ms)	$N = 5$	$N = 10$	$N = 15$	$N = 20$	$N = 25$	$N = 30$	$N = 35$	$N = 40$
0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0
32	0.7752	0	0	0	0	0	0	0
64	0.9779	0.4334	0.0002	0	0	0	0	0
80	0.9988	0.8733	0.2011	0.0001	0	0	0	0
96	0.9999	0.9834	0.6826	0.0851	0	0	0	0
112	0.9999	0.9985	0.9281	0.4715	0.0333	0	0	0
128	0.9999	0.9999	0.9890	0.8199	0.2946	0.0129	0	0
144	0.9999	0.9999	0.9987	0.9592	0.6725	0.1714	0.0052	0
160	0.9999	0.9999	0.9999	0.9932	0.8987	0.5147	0.0969	0.0042
176	0.9999	0.9999	0.9999	0.9991	0.9773	0.8065	0.3753	0.0763
192	0.9999	0.9999	0.9999	0.9999	0.9961	0.9444	0.6949	0.3175
208	0.9999	0.9999	0.9999	0.9999	0.9994	0.9878	0.8922	0.6323
224	0.9999	0.9999	0.9999	0.9999	0.9999	0.9978	0.9710	0.8547
240	0.9999	0.9999	0.9999	0.9999	0.9999	0.9996	0.9938	0.9562
256	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9989	0.9895
272	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9998	0.9979
288	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9996
304	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
≥320	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999

moving down). Moreover, we plan to expand our study to evaluate the energy consumption of 2CS-WSN.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work has been partially supported by CICYT Project TIN2012-36812-C02-02.

References

[1] IEEE 802.15 WPAN Task Group 4b (TG4b), "IEEE Std 802.15.4-2006 - Part 15.4: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LRWPANs)," September 2006.

[2] A. Koubaa, M. Alves, and E. Tovar, "A comprehensive simulation study of slotted CSMA/CA for IEEE 802.15.4 Wireless Sensor Networks," in *Proceedings of the IEEE International Workshop on Factory Communication Systems (WFCS '06)*, pp. 183–192, June 2006.

[3] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "Performance evaluation of the IEEE 802.15.4 MAC for low-rate low-power wireless networks," in *Proceedings of the Workshop on Energy-Efficient Wireless Communications and Networks (EWCN '04) Held in Conjunction with the IEEE International Performance Computing and Communications Conference (IPCCC '04)*, pp. 701–706, 2004.

[4] F. Royo, M. Lopez-Guerrero, L. Orozco-Barbosa, and T. Olivares, "2C-WSN: a configuration protocol based on TDMA communications over WSN," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '09)*, pp. 1–6, December 2009.

[5] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: verification of probabilistic real-time systems," in *Proceedings of the 23rd International Conference on Computer Aided Verification (CAV' 11)*, vol. 6806 of *Lecture Notes in Computer Science*, pp. 585–591, Snowbird, Utah, USA, 2011.

[6] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[7] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, pp. 1567–1576, June 2002.

[8] A. Faridi, M. R. Palattella, A. Lozano et al., "Comprehensive evaluation of the IEEE 802.15.4 MAC layer performance with retransmissions," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 8, pp. 3917–3932, 2010.

[9] B.-H. Lee, R.-L. Lai, H.-K. Wu, and C.-M. Wong, "Study on additional carrier sensing for IEEE 802.15.4 wireless sensor networks," *Sensors*, vol. 10, no. 7, pp. 6275–6289, 2010.

[10] L. F. W. V. Hoesel and P. J. M. Havinga, "A lightweight medium access protocol for wireless sensor networks," Tech. Rep., 2004.

[11] A. Fehnker, L. van Hoesel, and A. Mader, "Modelling and verification of the LMAC protocol for wireless sensor networks," in *Proceedings of the 6th International Conference on Integrated Formal Methods (IFM '07)*, pp. 253–272, 2007.

[12] K. G. Larsen, P. Pettersson, and W. Yi, "Uppaal in a nutshell," *International Journal on Software Tools for Technology Transfer*, vol. 1, no. 1-2, pp. 134–152, 1997.

[13] P. E. Bulychev, A. David, K. G. Larsen et al., "Statistical model checking for priced timed automata," in *Proceedings of the 10th*

Workshop on Quantitative Aspects of Programming Languages (QAPL '12), pp. 1–16, Tallinn, Estonia, March–April 2012.

[14] M. Duflot, L. Fribourg, T. Herault et al., "Probabilistic model checking of the CSMA/CD protocol using PRISM and APMC," *Electronic Notes in Theoretical Computer Science*, vol. 128, no. 6, pp. 195–214, 2005.

[15] T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet, "Approximate probabilistic model checking," in *Proceedings of the 5th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI '04)*, pp. 73–84, Venice, Italy, 2004.

[16] M. Kwiatkowska, G. Norman, and J. Sproston, "Probabilistic model checking of the IEEE 802.11 wireless local area network protocol," in *Proceedings of the 2nd Joint International Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification (PAPM-PROBMIV '02)*, vol. 2399, pp. 169–187, Springer, London, UK, 2002.

[17] M. C. Ruiz, J. A. Mateo, H. Macia, J. J. Pardo, and T. Olivares, "Formal modelling and performance evaluation of a novel role-based Routing Algorithm for wireless sensor networks," in *Proceedings of the 18th Annual International Conference on Advanced Computing and Communications (ADCOM '12)*, pp. 98–104, IEEE, Bangalore, India, December 2012.

[18] J. A. Mateo, H. Macià, M. C. Ruiz, J. J. Pardo, and A. M. Ortiz, "Formal study of a novel network role-based routing intelligent algorithm," in *Prooceedings of the 13th Annual International Conference on Computational Science (ICCS '13)*, pp. 2525–2528, June 2013.

[19] M. Paterakis and P. Papantoni-Kazakos, "Simple window random access algorithm with advantageous properties," *IEEE Transactions on Information Theory*, vol. 35, no. 5, pp. 1124–1130, 1989.

[20] S. M. Ross, *Stochastic Processes*, Wiley Series in Probability and Mathematical Statistics, 1983.

[21] H. Hansson and B. Jonsson, "A logic for reasoning about time and reliability," *Formal Aspects of Computing*, vol. 6, no. 5, pp. 512–535, 1994.

[22] E. M. Clarke and E. A. Emerson, "Design and synthesis of synchronization skeletons using branching time temporal logic," in *Logics of Programs*, pp. 52–71, 1981.

[23] PRISM, 2013, http://www.prismmodelchecker.org/.

[24] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, "MAC essentials for wireless sensor networks," *IEEE Communications Surveys and Tutorials*, vol. 12, no. 2, pp. 222–248, 2010.