Formal Modeling of Greedy Nodes in 802.15.4 WSN
Youcef Hammal¹, Jalel Ben-Othman², Lynda Mokdad³, and Abdelkrim Abdelli¹

¹LSI, Dept of Computer Science, USTHB University, Algeria
²L2TI, Galilee Institute, University of Paris 13, France
³LACL, University of Paris-Est, France

Abstract
This paper deals with formal specification of the non-slotted CSMA/CA protocol in wireless sensor networks (WSN) whose some nodes own a greedy behavior. This protocol requires sensor nodes to wait some time before initiating a transmission, whereas greedy nodes may try to reduce their waiting duration, which may penalize other nodes. To analyze their impact on WSN mode in operation, we use timed automata of the model-checker UPPAAL to capture the abstract behavior of communication medium, sane, and greedy nodes in WSN. This enables the use of analysis tools to check whether these models satisfy intended properties.

Index Terms: IEEE 802.15.4; CSMA/CA; Formal Modeling; Timed Automata.

1. Introduction
A Wireless Sensor Network (WSN) typically consists of a large set of sensor nodes scattered over a geographical area for gathering data on physical phenomena. Delivery of sensory data for processing and analysis to a control station, is based on a collaborative work of the nodes in a multi-hop fashion.

Traditionally, the largest challenge in wireless sensor networks is the energy saving due to limitations of power sources. Thus, communications shall abide to strict time-constraints in WSN as well as other Low-Rate Wireless Personal Area Networks (LR-WPANs). The IEEE 802.15.4 protocol [4] was just the solution to overcome such problems thanks to its medium access control (MAC) sublayer. Indeed, it employs carrier sense multiple access with collision avoidance (CSMA/CA) to reduce the possibility of collision, thereby, reducing energy consumption in nodes. Note that contention-based MAC can be either slotted or unslotted CSMA/CA, depending on the network operation behavior: beacon-enabled or non beacon-enabled modes, respectively.

Similarly to [3], we aim in this paper at using formal methods to model and analyze the non-slotted CSMA/CA protocol related to the MAC sublayer of the IEEE 802.15.4 standard where any station has to delay some amount of time referred to as backoff delay and then sense the medium as free before it could send its packet. Such an algorithm intends to reduce collisions and improve the transmission rate.

However, the IEEE 802.15.4 MAC layer is based upon a distributed algorithm that is executed locally on each node to determine the periods of access to the channel. Due to the execution of channel access on each device, some greedy nodes may have selfish behaviors without respecting the CSMA/CA rules. Actually, a greedy node modifies some parameters of its IEEE 802.15.4 MAC layer in order to increase its bandwidth, at the expense of other nodes. Simple changes of several protocol parameters in one or a set of nodes can have devastating effects on the overall network performance. These changes would even be similar to Denial of Service (DoS).

Therefore, in order to enable a rigorous analysis of WSN with compromised nodes, a formal specification is proposed in this study to capture the behavior of sane as well as greedy nodes along with the wireless medium in terms
of communicating timed automata that are used as inputs in model checking tools (e.g., UPPAAL[2]). Safety and liveness properties could be checked by means of this tool, such as absence of deadlock and successful termination of transmissions.

In our method, a greedy node can increase its access to the channel by different ways. Thus, it may change several parameters such as reduce the backoff delay during which the channel must be sensed as idle before it could be accessed, or/and increase the number of times the CSMA/CA algorithm could be required to backoff while trying to access the channel.

2. CSMA/CA Protocol Decription

![Diagram](image)

Fig 1: The non-slotted CSMA/CA algorithm.

The MAC protocol supports two operational modes: the non beacon-enabled mode which is mainly used for peer-to-peer topology, and the beaconenabled mode for other topologies. Accordingly, the IEEE 802.15.4 defines two versions of the CSMA/CA mechanism: (i) slotted CSMA/CA algorithm – used in the beacon-enabled mode and (ii) non-slotted CSMA/CA algorithm – used in the non beacon-enabled mode. In both cases, the CSMA/CA algorithm uses a basic time unit called Backoff Period (BP), which is equal to 20 Symbols (where a symbol corresponds to the time needed for sending 4 bits).

The non-slotted CSMA/CA procedure (given in Fig.1) depends on the following parameters: The backoff exponent (BE) enables the computation of the backoff delay, which is the time before performing a CCA (Clear Channel Assessment). The backoff delay is a random variable between 0 and (2BE-1) backoff periods. CCA represents the period (set to 8 symbols) during which the channel must be sensed idle before accessing the channel. The number of backoffs (NB) is the number of times the algorithm was called to backoff while attempting to access the channel. It is reset to zero before each new transmission attempt.

The previous parameters are used in each node to control the channel access. BE allows to generate a random backoff duration for which a node has to wait before attempting the carrier sensing. A node waits for a random number of backoff periods if it has data to send, where the backoff duration is between 0 and (2BE - 1) of period units. BE is initially set to the value of macMinBE = 3, and can be incremented up to the maximum value aMaxBE = 5, whenever the counter of backoff trials (NB) increases.

If the channel is sensed as idle during CCA, then data transmission starts. If the channel is assessed to be busy, NB and BE are incremented by 1. The backoff trial for a packet transmission can be repeated until NB (number of backoff trials) exceeds macMaxCSMABackoffs = 4.

3. Greedy Node

A greedy node aims to improve its channel access comparatively to non-compromised nodes. Consequently, this increases the amount of data to be sent at the expenses of non-compromised nodes which may suffer famine situations. For instance, the mean time observed by a greedy node before accessing the channel could be lesser than the CCA in a sane node and the number NB of backoff trials can increase greater than macMaxCSMABackoffs = 4.

It is obvious that reducing the CCA increases the chances to access the channel. On the other hand, this may also increase the probability to observe more collisions than in the normal case. Indeed, reducing the CCA to a value less than the RTT, will increase the probability of collision occurrences, as a signal transmitted by a sane node may reach the greedy node after the latter achieves its CCA. However, collisions may increase for a greedy node only when the number of other nodes accessing the channel at the same time is important. As these cases are unlikely to happen, the benefits in terms of increasing the probability to access the channel is prevailing on the risk to undergo a collision. Moreover, considering the greedy behavior, the risk of a collision is a secondary concern.

4. Formal Modeling Language

To assign a formal specification to the protocol, we use the modeling language of the model-checker UPPAAL [2]
where the system is viewed as a network of timed automata [1] evolving in parallel and each of which is related to at least one of the system components. A timed automaton is a flattened finite state machine extended with clock variables. It uses a dense-time model where a clock variable evaluates to a real number. All the clocks progress synchronously.

An untimed automaton is a tuple $A = (Q, \Sigma, \rightarrow, ; q_0)$ where $Q$ is the set of locations (depicted as graph nodes), $\Sigma$ is the set of interactions which this process could perform, $\rightarrow \subseteq Q \times \Sigma \times Q$ is the set of edges between locations, and $q_0$ is the initial location. However, designers of a component may add to its automaton timing constraints over interactions that may happen between the component and its environment. Formally, this approach consists in expressing time constraints by means of boolean formulas over logical clocks. Such variables express time progress but their values could be initialized and tested.

A Timed Automaton extends an untimed one with three mappings as follows: The first mapping $I$ assigns to each location a sojourn condition called invariant which may be true. The second mapping $G$ assigns to each edge a timing guard which should be true to let the edge be taken. Last, the mapping $Z$ associates with each edge a set of clock initializations which may be empty. The UPPAAL modeling language consists of timed automata extended further with discrete constants and bounded variables that are used as in programming languages:

```plaintext
const int macMinBE := 3, aMaxBE := 5;
const int maxMacCSMABackoffs := 5;
```

Automata interact with each other through binary synchronization channels (e.g., chan c). An edge labeled with $c!$ synchronizes with another labeled $c?$. A synchronization pair is chosen non-deterministically if several combinations are enabled. In a broadcast synchronization, one sender uses a broadcast channel collision! to synchronize with an arbitrary number of receivers collision??. Any receiver that can synchronize in the current state must do so. Arrays are also allowed for clocks, channels, and integer variables, e.g.

```plaintext
chan beginSend[NBR STATIONS];
```

A state of the system is defined by the locations of all automata, the clock constraints, and the values of the discrete variables. Every automaton may fire an edge separately or synchronize with another automaton, which leads to a new state.

### 5. CSMA/CA Specification

We aim at modeling a WSN made up of nodes communicating through a shared medium. We give below automata templates of these components: the wireless medium, normal and compromised nodes.

#### 5.1. Automaton of the medium

The role of the medium (see Fig. 2(a)) is to propagate packets between nodes. A transmission of any node may interfere with other transmissions leading to collisions. The process of a packet sending is modeled through two successive interactions between the medium and the sender whose identifier is $e$; that is, $(beginSend[e]?,?, endSend[e]?)$ modeling respectively the start and the successful termination of a transmission of $e$. The delay between these two events shall depend on the duration of the packet transmission. Such a time constraint is depicted by the invariant $x \leq \text{FtrDur}$ on the location Transmitting in the sender node automaton (where $x$ is a logical clock).

Any intervening transmission while control is still in the locations in-between, would raise a collision (see Fig. 2(a)). Furthermore, once the medium automaton receives the beginSend[$e$] event its status $wmStatus$ does not switch immediately from free to busy because it has to wait
The medium is sensed free by any inquiring node so that this increases the collision risks. The collision! event shall be raised whenever another transmission is launched (illustrated by endSend[a]? triggered by any station a) before the current transmission is finished (i.e., endSend[e]? event is not received yet).

5.2. Automaton of a sane node

The automaton template (see Fig.2(b)) depicts the sensing function of a node st. Note that its initial location is Idle from which the control can go to End location if all packets have been sent. Otherwise, the control goes to Testing location along with the backoff exponent BE initialized to macMinBE value if this location is entered for the first time during the transmission cycle of a same packet. Thereafter, the automaton checks whether the number of trials NB exceeds the threshold maxMacCSMABackoffs during the transmission of the current packet. In the former case, control shall be shifted to the Error location and then back to the initial location Idle. Otherwise, control moves to BackingOff where it stays therein for a duration equal to Backoff Delay. Such a time constraint is depicted by the invariant $\text{x} \leq \text{Backoff Delay}$ upon the location BackingOff. This delay is sampled from the adequate time interval (interval1, interval2, or interval3) depending on BE value (depicted by guards on the related arcs).

Once the backoff delay elapsed, the CCA procedure is performed where the medium is sensed for some amount of time. If it is sensed busy then control moves to Testing location along with control moves to Testing location along with increment of the number of trials NB and the backoff exponent BE modulo a threshold aMaxBE. However, if the medium is continually sensed free for CCA Delay the automaton triggers the event beginSend[st!] to the medium automation (modeling the start of a packet transmission) and control shifts to Transmitting location where it stays for a delay equal to RTT. If a collision occurs a relating event collision? is triggered incrementing thus the number of collisions NBRcol. This makes control move to the medium automation (modeling the start of a packet transmission) and control shifts to Transmitting location where it stays for a delay equal to RTT. If a collision occurs a relating event collision? is triggered incrementing thus the number of collisions NBRcol. This makes control move to Collision location and next to Idle location after the transmission delay is elapsed. However, if endSend[st!] event is triggered after CCA delay is elapsed then this would mean a successful termination and leads control to return back to Idle location for sending a next packet.

Here is the declaration of some parameters we have used in the node automaton (Fig.2(b)):

```plaintext
typedef int[0; 140] interval1;
typedef int[0; 300] interval2;
typedef int[0; 620] interval3;
```

<table>
<thead>
<tr>
<th>Constant Definition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>interval1</td>
<td>140</td>
</tr>
<tr>
<td>interval2</td>
<td>300</td>
</tr>
<tr>
<td>interval3</td>
<td>620</td>
</tr>
<tr>
<td>const int FtrDur</td>
<td>100</td>
</tr>
<tr>
<td>const int CCA Delay</td>
<td>8</td>
</tr>
<tr>
<td>const bool wmIdle</td>
<td>true</td>
</tr>
<tr>
<td>const bool wmStatus</td>
<td>true</td>
</tr>
</tbody>
</table>

5.3. Automaton of Greedy Node

Besides aforesaid behaviors of sane nodes, a compromised node can use false values about its NB, CCA Delay and Backoff Delay to increase its chances to firstly access the medium and send its packets according to a greedy behavior. The compromised node has the same automaton as that of the normal node (Fig.2(b)) but it does not behaves the same way because of the different delays and number of backoffs it uses. We give below the definition of the false parameters (i.e., time intervals and variables):

```plaintext
typedef int[0; 70] interval1;
typedef int[0; 150] interval2;
typedef int[0; 310] interval3;
```

<table>
<thead>
<tr>
<th>Constant Definition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>interval1</td>
<td>70</td>
</tr>
<tr>
<td>interval2</td>
<td>150</td>
</tr>
<tr>
<td>interval3</td>
<td>310</td>
</tr>
<tr>
<td>const int CCA Delay</td>
<td>4</td>
</tr>
<tr>
<td>const int maxMacCSMABackoffs</td>
<td>10</td>
</tr>
</tbody>
</table>

5.4. The Global Model

Once the templates are defined, we build the whole system by instantiating CompromisedNode and SaneNode templates as many times as necessary and then combining in parallel the produced processes with only one process WirelessMedium as follows:

```plaintext
node0=SaneNode(0); node1=SaneNode(1); ...
nodeC0=CompromisedNode(0); ...
```

<table>
<thead>
<tr>
<th>System Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>WirelessMedium</td>
</tr>
<tr>
<td>node0</td>
</tr>
<tr>
<td>node1</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>nodeC0</td>
</tr>
</tbody>
</table>

6. Conclusion

We have presented a timed automata-based approach to model the IEEE 802.15.4 non-slotted CSMA/CA protocol of WSN which some nodes may be greedy.

We intend to use in a future work model-checkers on our models to check expected properties and do performance evaluation once automata transitions are annotated with probabilities of transmission events.

References