A Timed Petri Net Model For The IEEE 802.15.4 CSMA-CA Process

A.Haffiz Shuaib, Toktam Mahmoodi, A.Hamid Aghvami.
Center for Telecommunications Research
King’s College London, Strand, London, UK.
WC2R 2LS.
Email: {abdul.shuaib, toktam.mahmoodi, hamid.aghvami}@kcl.ac.uk

Abstract—The IEEE 802.15.4 specification has generated a lot of interest in recent times, especially within the Wireless Sensor Network (WSN) research community, primarily because energy efficiency is one of the specifications’ design cornerstone.

As this specification is relatively new, it is incumbent that its operational mechanisms are well understood, to allow for efficient cross layer interactions between the different processes that make up the specification and existing or new higher layer protocols.

In this paper, we present 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, we are able to analyze the performance characteristics of the CSMA-CA process, especially in terms of channel throughput and energy consumption. We also verify the extracted system indices by comparing them to those gotten from a full model of the specification, implemented using the OPNET network simulation platform.

Keywords: Wireless Sensor Networks, IEEE 802.15.4, CSMA-CA, Modelling, Petri-Net.

I. INTRODUCTION AND MOTIVATION

The IEEE 802.15.4 Low-Rate Wireless Personal Area Network (LR-WPAN) specification [1] is fast becoming to Wireless Sensor Networks (WSN), at least for now, what the Internet Protocol (IP) is to the internet i.e. a protocol specification on which other related protocols can be built around. For instance, several network protocols including [2] [3] and [4] have been designed to run atop the specification, [5] attempts to retrofit the Ad-hoc On-demand Distance Vector routing (AODV) protocol for the specification and [6] [7] and [8] for topology control. Several digital receiver architectures [9], security mechanisms [10], ranging applications [11] and even protocols that will support voice transmissions [12] have been analyzed within the context of the specification. The above underscores a need for a proper understanding of the specification in terms of its limitation, and areas that will allow for efficient cross layer interactions.

From an implementation perspective, the IEEE 802.15.4 Medium Access Control (MAC) layer is made up of several processes. These processes include; the MAC Common Part Sub-layer (MCPS), the MAC sub-Layer Management Entity (MLME), the Contention Free Period Process and the CSMA-CA process. Of all these, the CSMA-CA process, which is the primary channel access process for the specification is the most complicated to implement or model in relative terms.

Several methods exist for modelling the CSMA-CA process analytically e.g. Markov chains, Petri-net and Finite State Machines (FSM). In this paper, we chose the Petri-net method due to its intuitiveness, simplicity, ability to efficiently couple a group of processes or systems together and the availability of a large body of analytical techniques to investigate the model [13]. Comparatively, modelling with Petri-net or FSM, the CSMA-CA process can be represented using just four states. A similar model using Markov chains requires approximately 17–19 states [14][15][16]. In terms of pictorial representation, the FSM and the Petri-net graphs are the most intuitive; this is particularly important when a need exists to transform these analytical models into simulation models or hardware implementations. On the other hand, in terms of mathematical representation, the Markov chain and the Petri-Net methods are comparatively better than the FSM method. Although, the mathematical representation of a protocol using Petri-Nets is by far simpler than when Markov chains are used, this is especially true for deterministic protocols.

To that end, in this paper, we present a Petri-Net model of the CSMA-CA process that advances with real time and is coupled with a channel contention mechanism, all of which allows our model to take cognizance of saturated and unsaturated traffic conditions.

The rest of the paper is organized as follows; In section II, we present a quick overview of the CSMA-CA specification. In III, we formally describe the CSMA-CA process as a timed Petri-Net graph with its associated state equations. In this section, we also derive a channel contention model. In section IV, we do a comparative analysis on the results extracted from the analytical platform and simulation platform. The Conclusion comes after that.

II. AN OVERVIEW OF IEEE 802.15.4 CSMA-CA PROCESS

The standard defines the active period for each node as its super frame duration (SD). This active period begins at the reception of a beacon. The interval between two consecutive beacons is referred to as the beacon interval (BI). Within the scope of this paper, we assume that each of the IEEE 802.15.4 nodes operates on the 2.4 GHz band with a bit rate of 250 Kbps. With this bit rate, equations (1) and (2) give the length of the BI and SD in seconds.
\[ BI = 0.01536 \times 2^{BO} \]  
\[ SD = 0.01536 \times 2^{SO} \]

Where \( BO \) and \( SO \) are the beacon order and super frame order respectively and \( 0 \leq SO \leq BO \leq 15 \).

The difference between the BI and the SD is the time portion in any BI for which a node is inactive and could go to sleep. The SD can be further divided into two time periods, namely the Contention Access Period (CAP) and the Contention Free Period (CFP). In the beacon enabled mode, the operation of the IEEE 802.15.4 CSMA-CA process is confined to the Contention Access Period (CAP) of the super frame duration.

The standard specifies a module called the MAC Common Part Sub-layer (MCPS) which handles framing and channel access activities for data packets. As soon as an upper layer data packet requiring contention channel access arrives at the MCPS, it frames this packet and sets the required fields in the MAC header. If the CSMA-CA process is idle, the MCPS invokes it and requests that it contends for and acquire the channel for frame transmission. The MCPS passes the length of the frame, \( T_{\text{frame}} \) (+ physical layer header) in seconds to the CSMA-CA process with each invocation. The CSMA-CA process then locates the next back off boundary, \( T_{\text{bo}} \) of the current super frame and goes into the back off state at the start of the back off boundary. In the back off state, the back off delay, \( T_{\text{bd}} \) is computed based on the selected Back off Exponent (BE) value. The CSMA-CA process computes the transaction duration, \( T_{\text{rd}} \) as follows:

\[ T_{\text{rd}} = T_{\text{bd}} + T_{\text{frame}} + T_{\text{iat}} + T_{\text{dack}} + T_{\text{curr}} + 2T_{\text{cca}} \]

where \( T_{\text{dack}} \) is the length of the acknowledgement frame in seconds, it is set to zero if none is required. \( T_{\text{iat}} \) is the transmitter-receiver turn around time in seconds, \( T_{\text{cca}} \) is the duration for a single Clear Channel Assessment (CCA) and \( T_{\text{curr}} \) is the current time. The CSMA-CA process checks to see if the frame transmission transaction which is measured by its duration \( T_{\text{rd}} \), can be completed before the current CAP ends. If the transaction duration exceeds the remaining period of the current CAP, then the CSMA-CA process pauses its operation and resumes at the beginning of the next CAP. If the transaction can be completed before the CAP ends, the CSMA proceeds to the CCA state. In this state, it queries the receiver at most twice for the status of the channel. If on any of these two queries a busy channel is indicated, the process immediately goes back to the back off state, where the back off process is repeated. The back off process can be repeated multiple times, as long as the maximum number of back offs set by the implementer is not exceeded. If the maximum number of back offs has been exceeded, a channel access failure is reported to the MCPS. Typically, the MCPS discards the frame and the source of the data frame is notified. On the other hand, if both CCAs indicate a free channel, the MCPS is given the go ahead to transmit the frame.

Within the scope of this work, we define a CSMA-CA cycle as the time from when a request is put forth to the CSMA-CA process to the time the result is returned (whether it is a channel access success or failure).

A Petri net graph is a directed weighted bipartite graph that can be used to capture the structure of a system. In a Petri Net graph, a state is represented by a combination of two types of vertices, referred to as transitions and places. A transition represents an event that changes the state of the system, while places are associated with those conditions that enable transitions to fire. Places and transition are connected by directed arcs.

From the structural information discussed in section II, we build a 7-tuple Petri net graph \( N \) shown in Figure.1.

\[ N = (P, T, W, w, K, M_0, x) \]

where

- \( P = \{p1, p2, p3, p4\} \) is a finite set of places.
- \( T = \{t1, t2, ..., t9\} \) is a finite set of transitions.
- \( K = \{k1, k2, ..., k9\} \) is the set of real time clock sequences associated with each transition.
- \( W \subseteq (PXT) \cup (TXP) \) the set of arcs from transitions to places and places to transitions.
- \( w : W \rightarrow \{1, 2, ..., 9\} \) is the weight functions on the arcs.
- \( M_0 \) the initial marking.
- \( x \) the marking of the set places.

In order to invoke the CSMA-CA module, the invoking module must pass a token to place \( p1 \). The CSMA-CA module is invoked, if transition \( t1 \) fires. At the end of a CSMA-CA cycle, the token is returned to the invoking module. This implies that graph \( N \) exhibits the property of non conservation. This property assures that our CSMA-CA Petri-Net model works in step with all its calling processes. Note that, if the CSMA-CA process currently has a token, its status is set to
busy and therefore no other token will be accepted from any other calling module.

Several parameters exist within the CSMA-CA process that must be updated several times in each CSMA-CA cycle. Three of such variables are the Number of Back Offs (NB), the back off exponent, BE and the Contention Window (CW). NB is set to zero and BE is set to \(-1\) in place \(p1\). NB is incremented by 1 when we transition to place \(p4\) and BE is incremented by 1 when we transition to place \(p2\). CW is set to zero in place \(p2\) and incremented by 1 every time we enter place \(p4\).

Notice that multiple transitions can be enabled simultaneously out of places \(p2\) and \(p4\). In order to make the transitions of the graph \(N\) deterministic, we place further conditions on the graph that enables just one transition out of place \(p2\) or \(p4\) at any given time, and these conditions are shown in Equation (5).

\[
\begin{align*}
t_3 & \text{ enabled iff } NB > MaxNB \\
t_5 & \text{ enabled iff } T_{rd} > T_{cap-end} \\
t_6 & \text{ enabled iff } T_{rd} \leq T_{cap-end} \\
t_9 & \text{ enabled iff } CW = 0 \text{ and } CH = idle \\
t_7 & \text{ enabled iff } CW = 1 \text{ and } CH = idle \\
t_8 & \text{ enabled iff } CH = busy
\end{align*}
\]

where MaxNB is the maximum number of back offs in one CSMA-CA cycle, CH is the channel busy status and will be governed by the channel contention model presented in the section after the next.

In order to advance in real time, we define two functions \(\Theta\) and \(\Delta\); \(\Theta(T_{displacement})\) advances the current time by \(T_{displacement}\) and defined by Algorithm 1. \(\Delta\) updates the CAP times i.e. takes us to the next CAP and is defined by Algorithm 2.

\[\text{Algorithm 1 update current time, } \Theta(T_{displacement})\]

\[
\text{if } T_{displacement} < T_{curr} \\
\text{ } T_{curr} = T_{curr} + T_{displacement} \\
\text{else} \\
\text{ } T_{curr} = T_{displacement}
\]

\[
\text{if } T_{curr} > T_{cap-end} \\
\text{update cap time, } \Delta
\]

\[\text{Algorithm 2 update cap time, } \Delta\]

\[
\text{do} \\
\text{ } T_{cap-start} = T_{cap-end} + T_{beacon-intervat} - T_{cap-length} \\
\text{ } T_{cap-end} = T_{cap-start} + T_{cap-length} \\
\text{while } T_{curr} >= T_{cap-end} \\
\text{ } T_{curr} = T_{cap-start}
\]

In (6), we show how the real time clock sequences advances the system time. For instance, \(k_1\) advances the current time by the absolute arrival time of the packet, \(T_{arr-time}\), while \(k_4\) advances the system time to the start of the next CAP, \(k_3, k_5, k_8\, \text{and } k_9\) do not advance the system time.

\[
\begin{align*}
k_1 &= \Theta(T_{arr-time}) \\
k_2 &= \Theta(T_{beacon-intervat}) \\
k_4 &= \Delta \\
k_5 &= \Theta(T_{beacon-intervat} + T_{beacon-intervat}) \\
k_6 &= \Theta(T_{beacon-intervat}) \\
k_3 &= k_5 = k_8 = k_9 = \varnothing
\end{align*}
\]

\[\text{B. State Equations For The CSMA-CA Process}\]

In order to obtain the equations that govern each state, we must first derive the incidence matrix, \(A\) of the graph as well as the row firing vector, \(U_i\).

Let \(n\) represent the number of places and \(m\) the number of transitions in a graph, then \(A\) which is the incidence matrix of the graph is an \(m \times n\) matrix populated using equation (8).

\[
A = \begin{pmatrix}
1 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 \\
1 & -1 & 0 & 0 \\
0 & 1 & -1 & 0 \\
0 & -1 & 1 & 0 \\
0 & -1 & 0 & 1 \\
0 & 0 & 0 & 0 \\
0 & 1 & 0 & -1 \\
1 & 0 & 0 & -1
\end{pmatrix}
\]

\[a_{ij} = \omega(t_i, p_j) - \omega(p_j, t_i)\]

where \(\omega(t_i, p_i) = 1\), if there is an arc going from transition \(t_i\) to place \(p_i\) and equal to zero other wise. Similarly, \(\omega(p_i, t_j) = 1\) if an arc exist that goes from \(p_i\) to \(t_j\).

The firing row vector \(U_i\) of the graph is as shown in Equation (9).

\[
U_i = \{0, ..., 0, 1, 0, ..., 0\}
\]

where \(U_i\) is a vector of dimension \(m\), that has 1 in its \(i^{th}\) position and 0 everywhere else, to indicate that the \(i^{th}\) transition is firing with \(i \in \{1, 2, ..., m\}\).

With \(A\) and \(U_i\), the state equations can be derived using the transition function shown in Equation (10).

\[
X' = f(X, t_i) = X + U_iA
\]

where \(t_i\) is the transition that fires to move the system from state \(X\) to \(X'\).

Using Equation (10) we compute the state Equations for the Petri-Net graph shown below:

\[
X_0 = f(X_1, t_3, k_3) = f(X_3, t_9, k_9) = [1 0 0 0]
\]

\[
X_1 = f(X_0, t_2, k_2) = f(X_2, t_4, k_4) = f(X_3, t_8, k_8) = [0100]
\]

\[
X_2 = f(X_1, t_5, k_5) = [0 0 1 0]
\]
\[ X_3 = f(X_1, t_6, k_0) = f(X_3, t_7, k_7) = [0 \ 0 \ 0 \ 1] \] (14)

From the state equations, we can infer the following: all states are reachable from the idle state \( X_0 \), the operation of the graph \( N \) is deterministic, as only one place has the token at any given time. The equations also show that the graph \( N \) is safe i.e. it can be contained within that of another graph e.g. a graph describing the MCPS or even the IEEE 802.15.4 node. This “safe” property is important especially when modelling cross layer interactions where these interacting layers share a resource. A simple example of how cross layer interaction can be achieved, is by having a higher layer interact with the back off state \( X_1 \), to infer the channel load by looking at the number of back offs, or by looking at state \( X_0 \) when \( t_0 \) and \( t_3 \) fire to determine the ratio of transmission failures to successes.

To compute the average throughput, we summed the number of bits transmitted every time transition \( t_0 \) fired. At the end of all transmissions, we divided this sum by the current time \( T_{curr} \), which gave the average throughput.

Recall, that in section II, we mentioned that the CSMA-CA process “pauses” its operation, if it is unable to complete a transaction within the remaining CAP. We will refer to the difference between the time at which the process enters the pause state and the end of the current CAP as the wasted CAP time. The wasted CAP time represents the opportunity cost of not contending for the medium due to the size of the frame that requires contention channel access. To compute the wasted CAP time we simply get the difference between \( T_{curr} \) and \( T_{cap-end} \), whenever we enter the pause state \( (X_2) \) i.e. when \( t_4 \) fires.

C. Modelling Channel Contention

The probability of a node capturing the medium for transmission is naturally affected by, among other things, the number of nodes contesting for the channel and the traffic generation intensity of the contending nodes.

If we denote the number of nodes contesting for the channel as \( n_c \), then the probability of a node finding the channel idle in a given contention interval/slot is given as \( f(k; n_c, p) \) with \( k = 1 \).

\[ f(k; n_c, p) = \binom{n_c}{k} p^k (1 - p)^{n_c - k} \] (15)

1 \( \leq n_c \leq N_{total} \), \( p = \frac{1}{n_c} \)

where \( N_{total} \) is the total number of nodes in the network.

For example, if \( n_c = 3 \) and \( k = 1 \), we see that each of the 3 contending nodes have a 44 % chance of sensing an idle channel. This translates to \( \sim 2 \) successful CCAs in every 5.\(^2\) This means that, on average each frame expands at least 5 CCAs before it is transmitted, as the standard mandates that two consecutive CCAs indicating a free channel is required before a frame is transmitted in the beacon enabled mode.

\(^2\)For analytical tractability, we assume that these two successful CCAs are consecutive.

Intuitively, we know that there will be increased contention for the medium within a certain contention slot if the offered load is high. The offered load is directly proportional to the packet inter-arrival times of the \( N_{total} \) nodes in the network. In order to accurately model the channel contention behavior, in both the saturated and unsaturated scenario, a relationship between the number of contending nodes and the packet inter arrival time at each node is required. To be able to do this, we will assume that as soon as a packet arrives from the upper layer, the CSMA-CA process, if idle, begins to contend for the medium. This implies that the contention rate on the channel follows the arrival rate of the packets.

Since the medium can handle \( N_{total} \) number of independent contensions, then with a contention rate of \( \alpha \), the probability of exactly \( k \) number of contention in any contention slot (interval), \( \beta \) is given by \( f(k, \lambda) \).

\[ f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \] (16)

Where \( \lambda = \beta \times \alpha \).

The value of \( k \in 1, 2, 3, \ldots, N_{total} \), for which \( f(k, \lambda) \) is maximum, was chosen as the number of contending nodes \( n_c \) within any contention slot given \( \alpha \) and \( \beta \).

For instance, let us assume that \( N_{total} = 8 \), \( \alpha = 1 \) packet or contention per unit time for all \( N_{total} \) nodes and \( \beta = 24 \) units of time, we will find that \( f(k, \lambda) \) is maximum when \( k = 8 \), meaning that within any contention slot all \( N_{total} \) nodes will contend for the medium. Now, if we set \( \alpha \) to 0.25 packet per unit time or 1 packet every 4 unit of time, \( f(k, \lambda) \) is maximum at \( k = 6 \). As expected, we see a reduction in the number of nodes contesting for the channel in any contention slot at a lower arrival rate, which in effect increases the probability of transmission (see Equation (15)).

There are two ways to verify the accuracy of the proposed channel contention model. We could either steadily increase the number of contending nodes or steadily increase the traffic generation intensity of all the pre existing nodes. For simulation efficiency, we chose the later and kept the number of nodes constant. The accuracy of the above is shown in section IV-B.

IV. SIMULATION PARAMETERS AND COMPARATIVE RESULT ANALYSIS

In this section, we compare the results of the analytical model to those of the simulation model.

In the first set of evaluations, we studied the impact of the frame inter arrival times (and by extension channel contention) on the throughput and the associated energy consumption. We also looked at how varying the value of the SO affected the throughput. In the second set of evaluations, we studied the impact of the frame size and the SO value on the wasted CAP time.

A. Simulation Setup and Parameters

Our simulation network scenario was made up of nine nodes, with one set as the network coordinator. Each non-coordinator nodes initiate the association process on receiving
the coordinator’s beacon. After association, the application layer of each node sends 1000 packets to the MAC sub-layer for onward transmission to the coordinator i.e. the coordinator acts as the network sink.

The network and CSMA parameters are set to the default values of the specification and are shown in Table I. The parameter values that relate to the analytical model are exactly as they were for the simulation model. For the analytical model, we make the following assumptions; the channel contention follows that defined in section III-C with \( \beta = 24 \text{ seconds} \).

\footnote{We chose 24 seconds as the contention interval, as it is a multiple of the number of nodes in the network and as you will see later, it also assures that given equation (15) and (16) and the chosen packet inter arrival time, at least two nodes will contend for the medium at any given point in time.}

**TABLE I**

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmit Power</td>
<td>0 dBm</td>
</tr>
<tr>
<td>Receiver sensitivity</td>
<td>-85 dBm</td>
</tr>
<tr>
<td>Bit Rate</td>
<td>250 Kbps</td>
</tr>
<tr>
<td>Modulation Scheme</td>
<td>QPSK</td>
</tr>
<tr>
<td>Physical Layer Header size</td>
<td>48 bits</td>
</tr>
<tr>
<td>MAC layer header size</td>
<td>104 bits</td>
</tr>
<tr>
<td>Acknowledgment frame size</td>
<td>104 bits</td>
</tr>
<tr>
<td>CCA Duration</td>
<td>( 1.25 \times 10^{-4} ) seconds</td>
</tr>
<tr>
<td>Back off Exponent (BE)</td>
<td>0 - 5</td>
</tr>
<tr>
<td>Maximum Number of Back offs</td>
<td>4</td>
</tr>
<tr>
<td>A unit Back off period (BP)</td>
<td>20 symbols</td>
</tr>
<tr>
<td>Acknowledgement enabled</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Fig. 2. Average Throughput

(a) BO = 5, SO = 1

(b) BO = 5, Packet Inter Arrival Time = 1 milliseconds

**Fig. 3. Average Number of CCAs and Energy expended to transmit a Frame**

(a) Average CCA per Frame

(b) Average Energy per Frame (\( E_{frame} \))

**B. Comparative Analysis A**

In this section, the size of the upper layer packets is a random variable that follows a poisson distribution and with mean length of 70 bytes. In the plots presented in Figure 2(a), we show the throughput as a function of the frame inter-arrival time. As expected, we see that as we increase the inter arrival time the throughput drops. While the trend in the figure is similar for both plots, we see that the throughput for the simulated scenario is lower than that of the analytical scenario. There are several reasons for this. The first reason has to do with the fact that packets are sent from the application layer to the MAC using interrupts in the simulation scenario. These interrupts take time to be serviced. And as a result delay is added, which impact on the throughput. Secondly, the time to encapsulate an upper layer packet is not negligible; the analytical model does not take cognizance of this. Thirdly, recall that we assumed that the whole of the SD is used for the CAP. In reality, some part of it is used to receive and process the network beacons. The effect of this processing delay is much more obvious in Figure 2(b), where the inter arrival time is much lower. To mitigate this effect, we can factor in the associated processing delay into the analytical model, if it is known.
In order to study the effect of the SO value on the throughput, we set the packet inter arrival time to 1 millisecond and the results are shown in Figure 2(b). As expected, we see that the throughput is directly proportional to the SO value, primarily because with a higher SO value (or a larger CAP) there is more time for contention and transmission and less time to sleep, as the BO is held constant.

In Figure 3(a), we show the average number of CCA expended to transmit a single frame as a function of the packet inter arrival time. Observe, that as we increased the inter arrival time the average CCA count drops for the node. This is as a direct result of a lower contention for the medium, given the interval between packet arrival. Notice that the trend for both the simulation and analytical model are similar, this helps to give some weight to our channel contention model introduced previously. Note that as the channel gets more and more saturated, the expected number of CCAs expended in an attempt to transmit a frame never exceeds 6.75. See Lamma 1 in the appendix for the proof of this.

We use Equation (17) to compute the energy in mili-joules consumed for the transmission, medium contention and acknowledgement reception for a single frame. The energy utilized for internal computation in the nodes is not considered in our observations. This same equation is also implemented in our simulation model.

$$E_{frame} = T_{dframe} \times P_{tx} + T_{dack} \times P_{rx} + T_{avg-cca} \times P_{tx} \quad (17)$$

where $T_{avg-cca}$ is the transmission time in seconds utilized by the average number of CCAs. $P_{rx}$ and $P_{tx}$ are the amount of power consumed for reception and transmission, respectively. The value of $P_{rx}$ and $P_{tx}$ are set to 81.0 and 74.1 milli-watts, respectively. This is based on the TI CC2430 radio chip [17] that implements the IEEE 802.15.4 physical layer specification and supports the CSMA-CA process.

In Figure 3(b), we show the energy expended to transmit frames as a function of the frame inter arrival time. If we were to pose a question on how much gains in terms of throughput can be made by using 1 second as the inter arrival time as opposed to 10 seconds, and what the opportunity cost will be in terms of energy consumption. Figures 2(a) and 3(b) are able to give us an idea. For instance, by setting the inter arrival time to 1 second, when compared to setting it to 10 seconds, we gain 89% more in terms of throughput for the analytical scenario. For the simulation scenario these gains translate to 85%. The opportunity cost for setting the inter arrival time to 1 second instead of 10 seconds, in terms of the energy consumed per frame is marginal i.e. the analytical scenario expands 3.6% more energy, when we set the inter arrival time to 1 second, when compared to setting the inter arrival time to 10 seconds. For the simulation scenario this opportunity cost is 5.7%.

C. Comparative Analysis B

Another question we can pose using the CSMA-CA model is ‘Given the pause mechanism of the IEEE 802.15.4 CSMA-CA protocol, how much CAP time is wasted on the average when we transmit data using a particular frame size?’ We can also ask ‘if increasing the SO value increases the efficiency of using the CAP?’

For the evaluation contained in this section, we had three sub-scenarios for both the analytical and simulation scenarios. In each of the scenarios we assumed that the application layer had 1000 bytes of data to send. This data was then fragmented into 25, 50 and 100 bytes frames for each of the three sub-scenarios, respectively.

Figure 4 shows that the time wasted (as a function of the frame size), in the CAP due to the pause mechanism of the IEEE 802.15.4 CSMA-CA process is not negligible, as was assumed in [14]. For instance, the amount of CAP time that goes unused in the 100 byte scenario is enough time to contend for the channel and possibly transmit a frame that is between 51 and 127 bytes long. If we picture the CAP as a container and the frames as items to be put into this container, intuitively, we know that smaller items will be packed more efficiently and this is what is depicted in the figure. It can also be seen from the figure that increasing the SO value does nothing to reduce the amount of CAP time wasted i.e. regardless of how big the container is, it is still more efficient to pack smaller items into it.

V. CONCLUSION

In this paper, we have demonstrated the simplicity and the strength of using the Petri Net modelling language to model the CSMA-CA protocol of the IEEE 802.15.4 standard. We suggested ways in which the model can be extended to address cross layer interactions. We validated the Petri Net model using a simulation platform and we have shown that the extracted performance indices in terms of energy consumption, throughput and the wasted CAP time, of both the simulation and analytical model align. In this work, we have shown that while increasing the inter arrival times of frames at the nodes increases the throughput, it does not necessary lead to higher energy consumption. We have also shown that in terms of utilizing the CAP period, smaller frames are better.

ACKNOWLEDGEMENT

This research was funded by AMYN Investments Nigeria Limited.
Lemma 1: As the channel gets more and more saturated the expected number of CCAs expended in an attempt to transmit a frame never exceeds 6.75.

Proof: The maximum number of back off periods in any CSMA-CA cycle is 5, as set by the IEEE 802.15.4 standard, with at most two CCAs after every back off period. There are three possible CCAs outcomes after every back off period. The first outcome is a success on the first CCA and a success on the second CCA (SS). The second outcome is a success on the first CCA and a failure on the second (SF). And the third is a failure on the first CCA (F), which means there will be no need for a second CCA. Remember that the condition necessary for a channel access failure after a back off period is that one of the two CCAs returns a busy signal. We will denote outcome a channel access failure after a back off period is that one of for a second CCA. Let us denote the set of outcomes that will lead to the minimum amount of CCAs expended to transmit a frame as $CS_{min}$ and the maximum as $CS_{max}$. Let us also denote the minimum amount of CCAs expended to transmit a frame as $|CS_{min}|$ and the maximum as $|CS_{max}|$. Correspondingly, let us denote the set of outcomes that will lead to the minimum number of CCAs expended before a channel access failure is reported to the CSMA-CA calling process as $CF_{min}$ and the maximum as $CF_{max}$. The minimum amount of CCAs expended for failure is $|CF_{min}|$ and the maximum is $|CF_{max}|$. Note that if the CSMA-CA process reports a channel access failure to its calling process, the frame is discarded as discussed previously.

\[ CS_{min} = [SS, \emptyset, \emptyset, \emptyset, \emptyset], |CS_{min}| = 2 \]  
\[ CS_{max} = [SF, SF, SF, SF, SF], |CS_{max}| = 10 \]  
\[ CF_{min} = [F, F, F, F, F], |CF_{min}| = 5 \]  
\[ CF_{max} = [SF, SF, SF, SF, SF], |CF_{max}| = 10 \]  

For $CS_{min}$, the CSMA-CA process might find the channel idle on the two CCAs after the first back off period. If this happens the frame is transmitted. There is therefore no need to go into the other back of periods, hence $|CS_{min}| = 2$. The possible outcomes that will cause the maximum number of CCAs to be expended on a frame before a successful channel capture, is shown in equation (19). In equation (19), a successful channel capture is obtained in the fifth back off period. To get to the fifth back of period, the previous four back off periods must have been unsuccessful in sensing an idle channel, therefore $|CS_{max}|$ is 10. Equations (20) and (21) can be interpreted accordingly.

The expected number of CCAs expended to capture the medium in order to transmit a frame never exceeds 6.75 i.e. $|CS_{min}|+|CS_{max}|+|CF_{min}|+|CF_{max}|$, irrespective of channel saturation.

\[ \text{References} \]