A Localized Slot Allocation Algorithm for Wireless Sensor Networks

Domenico De Guglielmo
Dept. of Information Engineering
University of Pisa, Italy
domenico.deguglielmo@iet.unipi.it

Giuseppe Anastasi
Dept. of Information Engineering
University of Pisa, Italy
giuseppe.anastasi@iet.unipi.it

Marco Conti
IIT-CNR
National Research Council, Italy
marco.conti@iit.cnr.it

Abstract — **While energy efficiency is typically considered the major concern in wireless sensor networks (WSNs), many real-life applications also require reliability, timeliness, and scalability. In such scenarios, Time Division Multiple Access (TDMA) is typically used for data communication, as it avoids collisions and provides predictable latency and minimum energy consumption. TDMA requires a slot scheduling algorithm to allocate transmission slots to sensor nodes. In this paper, we propose a decentralized slot allocation algorithm which is *localized* and *self adaptive*, i.e., each node selects its slot(s) and adapts its behavior only basing on locally-available information. We derive analytically the time taken by the algorithm and the average energy consumed by the network to achieve a complete schedule. We also show that our solution performs significantly better than another previous similar algorithm.**

Keywords: TDMA scheduling, Energy Efficiency, Reliability, Scalability, Markov Chain.

I. INTRODUCTION

Wireless sensor networks (WSNs) are currently used for a large number of applications in different domains, ranging from environmental monitoring to healthcare, from logistics to industrial applications, from location and tracking to automatic building management, and so on. Usually, *energy efficiency* is the major constraint in the design of WSN-based systems [1] since sensor nodes are typically powered by batteries, with limited power budget, that cannot be replaced or recharged, due to environmental and/or cost constraints. However, in many application scenarios, additional requirements need to be considered, such as *reliability*, *timeliness,* and *scalability* [2]. In such scenarios contention-based MAC (Medium Access Control) protocols – like IEEE 802.15.4 [3], or B-MAC [4] – are not suitable for data collection as they are not able to provide any of the above-mentioned requirements. Hence, Time Division Multiple Access (TDMA) is typically used for communication among sensor nodes.

In TDMA, time is divided into transmission slots that are assigned to sensor nodes, according to a slot scheduling algorithm. Each sensor node activates only during its own slot and sleeps for the rest of the time, thus saving energy. Therefore, TDMA provides guaranteed bandwidth, high energy efficiency, absence of collisions (i.e., high reliability), low and predictable latency. Due to these nice properties, the *Time Synchronized Channel Hopping* (*TSCH*) protocol, which is part of the recently approved IEEE 802.15.4e standard [5], relies on a TDMA scheme. Specifically, TSCH is based on a *slotframe* structure, consisting of a number of transmission slots, and each sensor node follows a schedule to know which slots have been assigned to it for transmissions/receptions. However, the standard just describes the mechanisms for executing a given schedule, while it does not specify how such a schedule can be built.

Slot scheduling algorithms for TDMA networks have been studied extensively in the past, and a number of solutions have been proposed also in the context of WSNs. A detailed survey of scheduling algorithms for tree-based sensor networks can be found in [6]. Both centralized (e.g., RAND [7]) and decentralized solutions (e.g., D-RAND [8], PEDAMACS [9], TIGRA [10]) have been proposed. Centralized algorithms perform better than decentralized ones when the network is (quasi) static, while they are not suitable for dynamic networks where sensor nodes can join and leave frequently. However, most of the previous decentralized algorithms are not localized, as sensor nodes typically need to exchange special control messages with their neighbors to acquire the right to use one or more slots in a dedicated way. This may be a problem in many real environments where packet loss is relevant. In addition, sensor nodes must remain active during the entire slot negotiation phase, to exchange information with other nodes, thus consuming energy.

In this paper we propose a *localized* slot allocation (LOCALL) algorithm which allows sensor nodes to select their own transmission slot(s) in an *autonomous* way, just relying on *local* information. This reduces energy consumption and makes the algorithm suitable for environments where packets can be easily corrupted or missed. LOCALL takes an approach similar to the *Collision Detection with Memory* (CDM) algorithm presented in [11]. However, our algorithm uses a different and more efficient strategy for slot selection, which results in a shorter time for completing the slot allocation phase. In addition, while the authors of [11] just provide bounds on the duration of the slot allocation phase, we derive its probability distribution. In this paper we refer to a single-hop sensor network, however we plan to extend LOCALL to multi-hop topologies as a further study, following an approach similar to that in [11].

To analyze the performance of the proposed algorithm we develop and solve an analytical model based on a Discrete Time Markov Chain (DTMC). We validate our model through simulation, and evaluate LOCALL in terms of *convergence time* (i.e., time required by the algorithm to achieve a complete TDMA schedule with a pre-defined probability) and *energy consumption.* We show that our algorithm has a lower convergence time than CDM. In addition, the energy consumed by LocAll to achieve a complete schedule is quite limited.

The rest of this paper is organized as follows. Section II presents the LocAll algorithm. Section III describes the analytical model, while section IV discusses the obtained results. Finally, Section V concludes the paper.

II. LocAll ALGORITHM

In this section, we provide a description of our LocAll algorithm. We consider a star sensor network, where each sensor node has to report data periodically to the sink node (which is assumed to be always active). Without losing in generality, we can assume that the reporting period, *T*, is fixed and common to all sensor nodes in the network. We also assume that the reporting period is divided into N_s slots, of equal duration, and sensor nodes have to report one data packet per period. Hence, each sensor node requires one dedicated slot per reporting period *T*.

In order to obtain a collision-free schedule, different sensor nodes must select a different slot in the period. This can be easily achieved if sensor nodes can exploit some centralized information (e.g., an allocation pattern sent by the sink node), or exchange information with other nodes. Instead, LocAll takes a different approach as it provides a *simple*, *localized,* and *self adaptive* mechanism through which sensor nodes can *autonomously* decide their transmission slot, within the period *T*.

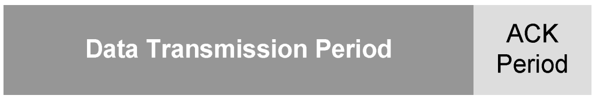

Figure 1. Slot access pattern during the data reporting phase.

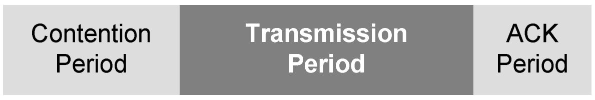

Figure 2. Slot access pattern during the slot acquisition phase.

As shown in Figure 1 the size of a slot is such that it can accommodate the transmission of one data packet and the related acknowledgement (ACK). Before using a slot for data transmission, however, a node has to achieve the right to use it. To this end, a preliminary *slot acquisition* phase is carried out where sensor nodes contend for acquiring the *exclusive* use of a slot. At the end of this phase, *each* node has acquired the right to use *one* slot. To implement contention, during the slot acquisition phase slots are accessed by sensor nodes with a different access pattern, shown in Figure 2 (the slot size is the same in both phases). Basically, sensor nodes contending for a generic slot σ, try to transmit a *fake* packet in that slot, using the contention period. If a sensor node wins the contention (i.e., it transmits successfully), it achieves a priority on that slot over all the other sensor nodes, and is authorized to access slot σ in all subsequent periods, *without* contention, to transmit its data packets. A complete collision-free schedule is achieved as soon as all sensor nodes in the network have acquired their own slot.

Ideally, we can assume that a contention is always solved (i.e., there is always one winner) irrespective of the number of competing nodes. Under this assumption, the quickest way to achieve a collision-free schedule is to allow *all* sensor nodes to contend for any slot. As a result of contention for a slot, one node is accommodated in that slot, while the remaining ones will contend for the next slots. Hence, the slot acquisition phase takes just one period.

In practice, the previous (ideal) scheme is unfeasible and we can only approximate it. In our algorithm we use a *random backoff time* to solve contentions, and we assume that it can take a number of discrete values (hence, collisions can still occur). After waiting for the chosen backoff time, a competing sensor node *i* checks the status of the channel and, if idle, tries to transmit a (fake) packet. Three possible outcomes can occur, namely (**i**) *successful transmission*, (**ii**) *busy channel*, or (**iii**) *collision*. In case of a successful transmission, node *i* is the winner of the contention and, hence, it acquires the right to use slot σ in all subsequent periods. To get priority over all the other nodes, in the next periods node *i* will access slot σ with a backoff time equal to 0 (i.e., hereafter, slot σ will be viewed by node i as its own data slot). This also reduces energy consumption and packet latency at node i. If the channel is found busy after the backoff time (case **ii**), it means that one or more sensor nodes have generated a *shorter* backoff time. Thus, node i has to try the next slot (i.e., $\sigma+1$) in the current period. Finally, when a collision is experienced by node i (case **iii**), it means that one or more other nodes have selected the *same* backoff time for contention in slot σ. In principle, node *i* could either retry the next slot (i.e., $\sigma+1$) in the current period, or retry the same slot (i.e., σ *)* in the next period. The rationale behind the latter option is that, if the number of colliding nodes is limited, the contention will be very likely solved at the next period. Another option for a colliding node i, would be re-trying slot $\sigma+1$ in the current period with probability p_r and defer contention to slot σ in the next period with probability ($1-p_r$). Also, this is the most general case (the previous ones can be derived from it, by using a value of p_r equal to 1 or 0).

Algorithm 1 shows the specific actions performed by a generic sensor node *i*. Initially, node i selects a random slot σ, within the current period, to try contention. This random choice is aimed at spreading contention trials within the whole period *T*, thus reducing the number of competitors for each single slot and increasing the success probability. However, this initial randomization very rarely provides a collision-free schedule. Hence, the slot acquisition phase follows, as described above. Specifically, node i contends for slot σ using a random backoff time *B*, and waits for the corresponding ACK message (lines 2-3). Depending on the received notification (line 4), node i either acquires the right to use slot σ (lines 5-6), or realizes that a failure has occurred. In the latter case it behaves in a different way, depending on whether a

CHANNEL-BUSY (lines 7-8), or COLLISION (lines 9-11) notification has been received.

Algorithm 1: LOCALL Algorithm

```
1   Choose a slot $\sigma$ in [1, $N_s$ ] randomly;
2   Try slot $\sigma$ (using a random backoff B); // contention slot
3   Wait (Notification);
4   Switch (Notification);
5     Case SUCCESS:
6       Use slot $\sigma$ in all subsequent periods
        with backoff B=0;                    //data slot
7     Case CHANNEL-BUSY:
8       Re-try slot $(\sigma+1)\bmod N_s$ (with random backoff B);
9     Case: COLLISION
10      Re-try slot $\sigma+1$ in the current period (with random
        backoff B) with probability $p_r$
11      Defer contention to slot $\sigma$ in the next period (with
        random backoff B) with probability (1- $p_r$);
```

A. *Slot Allocation in IEEE 802.15.4 sensor networks*

In the previous description we made no assumption about the sensor platform where LOCALL is supposed to run. Actually, it can be implemented on any sensor platform. However, if the considered sensor platform includes a contention-based MAC protocol (e.g., 802.15.4 MAC [3], B-MAC [4]), the algorithm can be customized to take advantage of the specific contention mechanisms provided by the underlying MAC layer. Below we briefly describe how LOCALL behaves when it operates on top of the IEEE 802.15.4 MAC layer.

The 802.15.4 MAC protocol [3] implements an un-slotted CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance) algorithm, whose behavior can be controlled through a set of parameters (see [15] for details). The following parameters are used by LOCALL.

- *macMaxCSMABackoffs* defines the maximum number of carrier sensing operations (**C**lear **C**hannel **A**ssessment (CCA)), per packet, that can be performed by sensor nodes. It is set to 0, i.e. just one carrier sensing is performed. If the channel is busy a failure notification is received by LOCALL from the underlying MAC.
- *macMaxFrameRetries* defines the maximum number of packet retransmission attempts allowed for a packet. It is set to 0, so as to force a failure notification if a collision occurs.
- *macMinBE* defines the initial backoff-window size. Sensor nodes randomly select a backoff time in the range $[0, N_B - 1]\cdot BP$, where $N_B = 2^{macMinBE}$, and $BP = 320\,\mu s$. During the slot acquisition phase *macMinBE* is set to a value larger than 0 (e.g., 3). As soon as a success notification is received (i.e., the slot has been acquired), LOCALL sets *macMinBE=0.* This gives priority to that sensor node over all the other sensor nodes in the network.

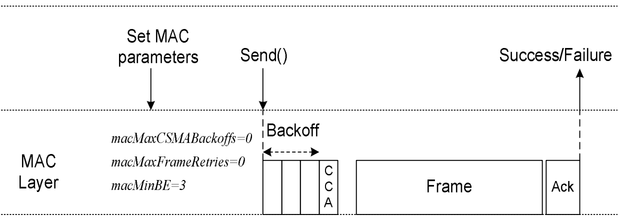

Figure 3. Interaction between the LOCALL algorithm and the underlying MAC layer.

Figure 3 shows the interactions between LOCALL and the underlying 802.15.4 MAC protocol. Protocol parameters are set in advance with respect to packet transmission. At the beginning of a slot, LOCALL makes a *send()* operation, thus activating the underlying MAC protocol. Then, it receives a success/failure notification. A failure may be caused by either a collision or a busy channel condition. The notification message received from the MAC layer contains enough information for LOCALL discriminating between the two events and reacting accordingly (as specified in lines 4-11 of Algorithm 1).

III. ANALYSIS

In this section we derive a Discrete-Time Markov-Chain (DTMC) model of the proposed LOCALL algorithm, under the hypothesis that it is operating on top of the 802.15.4 MAC protocol. To simplify the analysis, throughout we will make the following assumptions.

- *All* sensor nodes start competing for the first slot in a period, i.e., we do not consider the initial randomization (line 1 in Algorithm 1). This maximizes the contention and, hence, we model a "worst-case" condition.
- The number of contending nodes is equal to the number of available slots, i.e., $N = N_s$ (for the scheduling process to converge it must be $N \le N_s$).
- The retry probability p_r (line 10 in Algorithm 1) is assumed equal to 0, i.e., after a collision, *all* colliding sensor nodes defer their transmission to the next period.

TABLE I. MAIN SYMBOLS USED IN THE ANALYSIS.

Symbol	Description
N_B	Number of backoff periods.
BP	Length of a backoff period
$P_{succ}(x)$	Probability of successful transmission, given x contending nodes.
$P_{coll}(c\|x)$	Probability that c nodes collide, given x contending nodes.
$P_{busy}(b\|x)$	Probability that b nodes find the channel busy, given x contending nodes.
$P^{A}_{coll}(c\|x)$	Probability that c nodes collide on a slot already acquired, given x contending nodes.
s_i, n_i	Status of slot i (F/A), Number of nodes that have scheduled a transmission on slot i in the current period
Ω	State space of the Markov chain
P	Transition probability matrix
$\mathbf{v}_k$	State probability vector

Our analysis is split into three parts. In the first part, we consider all the possible events that can occur during the slot acquisition phase and, for each of them, we derive the probability to occur. In the second part, we will use these probabilities to model the slot acquisition process, through a DTMC, and derive the probability distribution of the convergence time. Finally, in the third part, we will derive the average energy consumed by LOCALL during the slot acquisition phase and in steady-state conditions. Table I summarizes the main symbols used in our analysis.

A. Event Probabilities

Let us consider a sensor network with N nodes, and assume that, at a given time, M (with $M \leq N$) sensor nodes are contending for the same slot. Contention can result in one of the following outcomes.

(a) SUCCESS. One sensor node, out of the M contending ones, successfully transmits in the slot.
(b) COLLISION. k ($2 \leq k \leq M$) sensor nodes experience a collision.
(c) BUSY CHANNEL. h ($0 \leq h \leq M-2$) sensor nodes find the channel busy and must retry at the next slot.

We now analyze the different cases individually and, for each of them, we derive analytically the probability to occur. A *successful* transmission (i.e., case (**a**)) occurs when one sensor node generates a backoff time shorter than that of all the other $M-1$ contending nodes. Let $P_{succ}(M)$ denote the probability that a successful transmission occurs, given that M nodes are contending for the same slot. Since each node can extract a backoff time with a discrete value in the range $[0, N_B - 1] \cdot BP$, the following equation holds.

$$P_{succ}(M) = M \cdot \sum_{b=0}^{N_B-1}\left[\left(\frac{1}{N_B}\right)\cdot\left(\frac{N_B-1-b}{N_B}\right)^{M-1}\right] \qquad (1)$$

Equation (1) can be explained as follows. For each potential backoff time b that can be generated by any sensor node (which can occur with probability $1/N_B$), the second term inside the sum gives the probability that the remaining $M-1$ sensor nodes extract a backoff time larger than b. Finally, all the M possible combinations, corresponding to the different sensor nodes, are considered.

Let us now consider case (**b**), where two or more sensor nodes (*i*) generate the same backoff time, (*ii*) start transmitting at the same time and, (*iii*) experience a collision. Let $P_{coll}(k \mid M)$ denote the probability that k out of the M contending nodes (with $2 \leq k \leq M$) collide. The following equation holds.

$$P_{coll}(k \mid M) = \sum_{b=0}^{N_B-1}\left[\binom{M}{k}\cdot\left(\frac{1}{N_B}\right)^{k}\cdot\left(\frac{N_B-1-b}{N_B}\right)^{M-k}\right] \qquad (2)$$

In equation (2), for each potential backoff time b, the term inside the sum gives the probability that k (out of M) nodes randomly pick up a value equal to b, and no other node chooses a value shorter than b. Of course, all $\binom{M}{k}$ possible combinations are considered.

In case (**c**), h ($0 \leq h \leq M-2$) sensor nodes experience a busy-channel condition. Let $P_{busy}(h \mid M)$ denote the probability that h nodes find the channel busy during a transmission attempt, given that there are M contending nodes. If h ($0 \leq h \leq M-2$) sensor nodes find the channel busy, it means that the remaining $M-h$ nodes have extracted the same backoff time and have collided. Hence,

$$P_{busy}(h \mid M) = P_{coll}(M-h \mid M) \qquad (3)$$

Finally, let us consider another event that may occur after a slot has been assigned. As described above, when a sensor node successfully transmits in a slot, it acquires the right to use that slot in all subsequent periods, and sets *macMinBE*=0. This gives it a priority, for that slot, over all the other nodes. However, collisions can still occur. This is because in the 802.15.4 CSMA algorithm, backoff times range in $[0, N_B - 1] \cdot BP$ and, thus, a collision occurs whenever any other node randomly generates a backoff time equal to 0. Let $P_{coll}^{A}(k \mid M)$ denote the probability that a collision involving k sensor nodes occurs on a slot already acquired by a node, given that there are M contending nodes, overall. This probability can be expressed as

$$P_{coll}^{A}(k \mid M) = \binom{M-1}{k-1}\cdot\left(\frac{1}{N_B}\right)^{k-1}\cdot\left(\frac{N_B-1}{N_B}\right)^{M-k} \qquad (4)$$

In equation (4), the second term gives the probability that $k-1$ nodes – in addition to the slot owner – extract a backoff time equal to 0. The third term gives the probability that all the remaining $M-k$ nodes pick up a backoff time larger than 0. Finally, the first term considers all possible combinations.

B. Convergence Time Distribution

In this section we develop a DTMC model of the LOCALL algorithm and use it to derive the probability distribution of the convergence time, i.e., the time required to achieve a complete schedule. To this end, we will use the event probabilities derived in the previous section.

We observe the system (i.e., sensor network) at the beginning of each period *T*. Since each period consists of $N_s = N$ slots, the system state at the beginning of period p ($p = 1,2,\ldots,n$) can be represented by a vector $\mathbf{X}_p = [(s_1,n_1),(s_2,n_2),\ldots,(s_N,n_N)]$, whose generic element $\mathbf{X}_p[i] = (s_i, n_i)$, $i = 1,2,\ldots,N$, refers to the corresponding slot in that period. Specifically, s_i indicates the status of slot i – either *Free* (*F*) or *Acquired* (*A*) by a sensor node – and n_i denotes the number of sensor nodes that have scheduled a packet transmission in slot i of period p.

Given the problem constraints, not all vectors in the form defined above represent possible states for the system.

The state space Ω can be defined as the set of vectors $\mathbf{X}_p = [(s_1, n_1), (s_2, n_2), ..., (s_N, n_N)]$ such that the following conditions hold.

C_1: $\sum_{i=1}^{N} n_i = N$

C_2: $\forall i \in \{1,2,...,N\}, n_i = 0 \Rightarrow s_i = F$

C_3: $\forall i \in \{1,2,...,N\}, n_i = 1 \Rightarrow s_i = A$

C_4: $\exists j \in \{1,2,...,N\} : n_j > 0 \Rightarrow \forall i < j,\ n_i > 0$

Condition C_1 states that the total number of transmissions scheduled for all the slots, in any period *p*, cannot exceed the number of sensor nodes. It follows from assuming that each sensor node can transmit, at most, one packet per period. Condition C_2 is almost obvious as well. It states that, if there is no scheduled transmission for slot *i* ($i = 1,2,...,N$) in period *p*, then slot *i* is necessarily free (i.e., $s_i = F$). Instead, Condition C_3 states that, if there is exactly one scheduled transmission for slot *i* ($i = 1,2,...,N$) in period *p*, then slot *i* has been acquired by some sensor node in one of the previous periods (i.e., its state must be $s_i = A$). This statement follows from the definition of *acquired* slot. Condition $n_i = 1$ implies that one sensor node has successfully transmitted on slot *i* in one of the previous periods. Hence, the slot has been acquired. Finally, condition C_4 states that if there is a slot $j\,(j = 1,2,...,N)$ for which the number of scheduled transmissions is larger than zero, then all previous slots must have, at least, one scheduled transmission. This follows from assuming that sensor nodes start contending for the first slot and, then, try all the other slots in sequence. As an example, Figure 4 shows all possible states of the Markov chain when the number of sensor nodes (and available slots) is equal to three. It can be easily verified that all states satisfy conditions $C_1 - C_4$.

Now, we need to derive the transition probabilities for the Markov chain, i.e., the probability of passing from a state ***X*** to another state ***Y***, for any ***X*** and ***Y*** in Ω. As a preliminary step, let us focus on a generic slot i, with $i = 1,2,...,N$. Let $\mathbf{X}[i] = (s_i, n_i)$ and $\mathbf{Y}[i] = (s_i', n_i')$ denote the condition of slot i at period p and $p+1$, respectively. Then, the probability of passing from $\mathbf{X}[i] = (s_i, n_i)$ to $\mathbf{Y}[i] = (s_i', n_i')$ can be expressed as follows.

$$P(\mathbf{Y}[i] \mid \mathbf{X}[i]) = \begin{cases} 1 & if\, n_i' = n_i = 0 & (T_1) \\ 1 & if\, n_i' = n_i = 1 & (T_2) \\ P_{succ}(n_i) & if\, (s_i' = A) \wedge (n_i' = 1) \wedge (s_i = F) \wedge (n_i > 1) & (T_3) \\ [(N_B - 1)/N_B]^{n_i - 1} & if\, (s_i' = A) \wedge (n_i' = 1) \wedge (s_i = A) \wedge (n_i > 1) & (T_4) \\ P_{busy}(n_i - n_i' \mid n_i) & if\, (s_i' = F) \wedge (1 < n_i' \le n_i) \wedge (s_i = F) & (T_5) \\ P_{coll}^{A}(n_i' \mid n_i) & if\, (s_i' = A) \wedge (1 < n_i' \le n_i) \wedge (s_i = A) & (T_6) \\ 0 & otherwise & (T_7) \end{cases}$$

The previous probabilities can be justified as follows.

In transition T_1 we assume $n_i = 0$, i.e., there are no sensor nodes contending for slot i in period p. Hence, the status of that slot at period $p+1$ will be unchanged. Similarly, in transition T_2 it is supposed that $n_i' = n_i = 1$. Since there is just one contending node, $P(\mathbf{Y}[i] \mid \mathbf{X}[i]) = 1$ in this case. In transition T_3, we suppose to start with $n_i > 1$ contending sensor nodes (and $s_i = F$). For the final state being $(A, 1)$ – meaning that the slot has been acquired by a sensor node – a successful transmission must occur during period *p*. Thus, the corresponding probability is $P_{succ}(n_i)$. Similarly, in transition T_4, we assume that there are $n_i > 1$ contending sensor nodes but, now, $s_i = A$. The final state $(s_i' = A, n_i' = 1)$ can occur only when all nodes – but the one that has already acquired the slot (and, hence, uses a null backoff time) – pick up a random backoff time larger than zero. The associated probability is thus $[(N_B - 1)/N_B]^{n_i - 1}$. In transition T_5 we suppose to start from state $(s_i = F, n_i > 1)$ but, now, we move to state $(s_i = F, n_i' > 1)$. This can occur when $n_i - n_i'$ (with $n_i - n_i' \ge 0$) find the channel busy during period *p*. Hence, the corresponding probability is $P_{busy}(n_i - n_i' \mid n_i)$. Finally, in transition T_6, it is assumed that the initial state is $(s_i = A, n_i > 1)$, while the final state is $(s_i = A, n_i' > 1)$. This transition occurs when n_i' sensor nodes experience a collision during period *p*. According to the notation introduced in section III-A, the corresponding probability is $P_{coll}^{A}(n_i' \mid n_i)$.

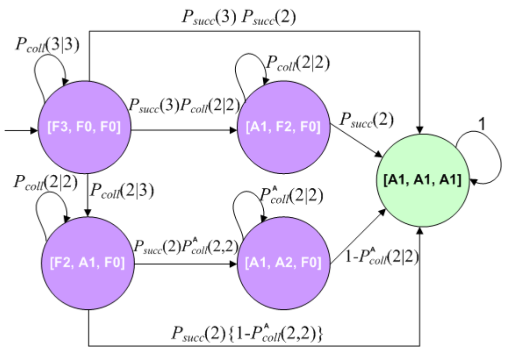
$P_{succ}(3)\ P_{succ}(2)$
$P_{coll}(3|3)$
$P_{coll}(2|2)$
$P_{succ}(3)P_{coll}(2|2)$
[F3, F0, F0]
[A1, F2, F0]
$P_{succ}(2)$
1
[A1, A1, A1]
$P_{coll}(2|2)$
$P_{coll}(2|3)$
$P^{A}_{coll}(2|2)$
$P_{succ}(2)P^{A}_{coll}(2,2)$
[F2, A1, F0]
[A1, A2, F0]
$1\text{-}P^{A}_{coll}(2|2)$
$P_{succ}(2)\{1\text{-}P^{A}_{coll}(2,2)\}$

Figure 4. Markov chain when N=N_s=3.

We are now in the position to derive the transition probability P_{XY} for any ***X*** and ***Y*** in Ω, starting from $P(\mathbf{Y}[i] \mid \mathbf{X}[i])$, $i = 1,2,...,N$. To this end, let us observe that the transition probability P_{XY} is given by the joint probability that each slot $i\,(i = 1,2,...,N)$ passes from $\mathbf{X}[i] = (s_i, n_i)$ to $\mathbf{Y}[i] = (s_i', n_i')$. Hence,

$$P_{XY} = P(Y[1] \mid X[1]) \cdot \prod_{i=2}^{N} P\{Y[i] \mid [s_i, n_i + n_i^r]\} \qquad (5)$$

where $n_i^r = \sum_{j=1}^{i-1} (n_j - n_j')$. In equation (5), for slots with index $i \ge 2$, the term inside the product is not

$P[Y[i] \mid (s_i, n_i)]$, but $P\{Y[i] \mid [s_i, n_i + n_i^r]\}$ because we need also to take into consideration possible nodes that experience a busy channel at slot $i-1$ and, thus, retry at slot i. Figure 4 shows the transition probabilities for the simple case when $N = N_s = 3$.

Once we have derived the transition probability P_{XY} for any $\mathbf{X}$ and $\mathbf{Y}$ in Ω, we can obtain the transition probability matrix $\mathbf{P}$ of the Markov chain. To this end, let us sort the states of the Markov chain in such a way that the initial state $\mathbf{I} = [(F,N),(F,0),...,(F,0)]$ and the absorbing state $\mathbf{Z} = [(A,1),(A,1),...,(A,1)]$ are, respectively, the first and last state in the sequence. Let $\mathbf{v}_0$ denote the initial probability vector, and $\mathbf{v}_k$ the vector probability after k periods $(k = 1,2,...)$. Without losing in generality we can assume $\mathbf{v}_0 = [1,0,0,...,0]$. Hence, $\mathbf{v}_k = \mathbf{v}_0\mathbf{P}^k$. The probability that the slot allocation process is over after k periods, $P_{schedule}(k)$, corresponds to the probability of being in state $\mathbf{Z}$ at step k. Let $|\Omega|$ denote the cardinality of Ω (i.e., the total number of states). Since $\mathbf{Z}$ is the last state in the sequence (according to the selected order), it follows

$$P_{schedule}(k) = \mathbf{v}_k[|\Omega|] \qquad (6)$$

C. Energy Consumption Analysis

In this section we derive formulas for computing the average energy consumed by the network both in a single period and during the entire slot acquisition phase. In the following we will assume that (**i**) the power consumption of sensor nodes in idle state is negligible, i.e., $P_{idle} = 0$; and (**ii**) the power consumption during the turnaround time is $P_{tat} = (P_{rx} + P_{tx})/2$, where P_{rx} (P_{tx}) is the power consumed in receive (transmit) mode. As a preliminary step, we first derive the energy E_{XY} consumed by the network to pass from state $\mathbf{X}$ to state $\mathbf{Y}$.

Let us assume there are M nodes ($M \le N$), all contending for the same slot. The total energy consumed by all nodes depends on the specific outcome following the contention. If one of the M nodes succeeds in transmitting its packet, the energy consumption $E_{succ}(M)$ can be calculated as follows

$$E_{succ}(M) = M \cdot P_{rx}D_{cca} + 2 \cdot P_{tat}D_{tat} + P_{tx}D_{tx} + P_{rx}D_{ack} \qquad (7)$$

where $D_{cca}, D_{tat}, D_{tx}, D_{ack}$ denote the duration of CCA, turnaround time, packet transmission time, and ACK reception time, respectively. The first term in Equation (7) accounts for the energy consumed by all the M nodes to perform their CCA, while the other terms account for the additional energy consumed by the winner node (all the other nodes find the channel busy and give up). The latter energy is spent for switching the radio from receive to transmit mode ($D_{tat} \cdot P_{tat}$), transmitting the packet ($D_{tx} \cdot P_{tx}$), switching again to receive mode ($D_{tat} \cdot P_{tat}$), and receiving the ACK ($D_{ack} \cdot P_{rx}$). Following the same line of reasoning, the total energy $E_{coll}(k \mid M)$ consumed when k out of the M competing nodes experience a collision, with $2 \le k \le M$, can be expressed as follows

$$E_{coll}(k \mid M) = M \cdot P_{rx}D_{cca} + k \cdot [2P_{tat}D_{tat} + P_{tx}D_{tx} + P_{rx}D_{to}]$$

where D_{to} is the timeout interval.

We can now calculate the energy consumed by the network when the state of a certain slot i ($i = 1,2,...,N$) passes from $\mathbf{X}[i] = (s_i, n_i)$ to $\mathbf{Y}[i] = (s_i', n_i')$, i.e., $E(\mathbf{Y}[i] \mid \mathbf{X}[i])$.

$$E(\mathbf{Y}[i] \mid \mathbf{X}[i]) = \begin{cases} E_{succ}(n_i) & n_i' = 1 \wedge n_i' \le n_i \\ E_{coll}(k \mid n_i) & n_i' = k \wedge n_i' \le n_i \\ 0 & otherwise \end{cases} \qquad (8)$$

Equation (8) can be justified as follows. In the first case, we assume to start with $n_i \ge 1$ competing nodes, one of which will be the winner ($n_i' = 1$). Hence, the energy consumption is $E_{succ}(n_i)$. In the second case, there are $n_i \ge n_i'$ nodes in the initial state and k ($2 \le k \le M$) nodes in the final state. Hence, k out of the n_i nodes experience a collision, and the total energy consumption is given by $E_{coll}(k \mid n_i)$. In all other cases, since there are no nodes accessing the slot, or the transition from $\mathbf{X}[i]$ to $\mathbf{Y}[i]$ is not possible, the related energy consumption is 0.

The total energy E_{XY} consumed by the network to pass from state $\mathbf{X}$ to state $\mathbf{Y}$ can be derived by considering the energy spent when the state of slot i passes from $\mathbf{X}[i] = (s_i, n_i)$ to $\mathbf{Y}[i] = (s_i', n_i')$, for any slot $i = 1,2,...,N$, i.e.,

$$E_{XY} = E(Y[1] \mid X[1]) + \sum_{i=2}^{N} E\{Y[i] \mid [s_i, n_i + n_i^r]\} \qquad (9)$$

where $n_i^r = \sum_{j=1}^{i-1}(n_j - n_j')$. In Equation (9), for slots with index $i \ge 2$, the term inside the sum is not $E[Y[i] \mid (s_i, n_i)]$, but $E\{Y[i] \mid [s_i, n_i + n_i^r]\}$. This accounts for nodes that found the channel busy at slot $i-1$ and, thus, retry at slot i.

We are now in the position to derive the average energy $\overline{E_k}$ consumed by the network during a period k ($k = 1,2,...$) for transmitting either a contention packet (during the slot acquisition phase) or a data packet (after acquiring a slot). Let $P_{\mathbf{X}}^k$ denote the probability that the system is in state $\mathbf{X}$ at period k for any $\mathbf{X} \in \Omega$ ($P_{\mathbf{X}}^k$ is the component of $\mathbf{v}_k$ associated with state $\mathbf{X}$). The following equation holds.

$$\overline{E_k} = \sum_{\mathbf{X} \in \Omega} P_{\mathbf{X}}^{k-1} \sum_{\mathbf{Y} \in \Omega} E_{\mathbf{XY}} P_{\mathbf{XY}} \qquad (10)$$

Equation (10) can be justified as follows. To calculate $\overline{E_k}$ we need to consider all the possible state changes that can occur from period $k-1$ to period k. Hence, the outer sum considers any possible state $\mathbf{X} \in \Omega$ where the system can be at period $k-1$, which occurs with probability $P_{\mathbf{X}}^{k-1}$. For each state $\mathbf{X} \in \Omega$ the inner sum considers all the possible states $\mathbf{Y} \in \Omega$ where the system can transit to – which occurs with probability $P_{\mathbf{XY}}$ – and the energy consumption $E_{\mathbf{XY}}$ associated with such a transition.

Finally, we derive the average energy spent by the network to achieve a complete schedule, i.e., the total amount of energy consumed by all nodes to contend and acquire slots. This measures the energy overhead due to slot scheduling. A complete schedule is achieved when the system enters the absorbing state $\mathbf{Z} = [(A,1),(A,1),...,(A,1)]$. Hence, according to the methodology in [12], the average energy $\mu_{\mathbf{X}}$ consumed by the network to reach state $\mathbf{Z}$, starting from any state $\mathbf{X} \in \Omega$, can be obtained by solving the following system of linear equations with $\mu_{\mathbf{X}}$ as unknowns

$$\mu_{\mathbf{X}} = \sum_{Y \in \Omega} P_{\mathbf{XY}} (E'_{\mathbf{XY}} + \mu_{\mathbf{Y}}),\ \forall \mathbf{X} \in \Omega \qquad (11)$$

with $\mu_{\mathbf{Z}} = 0$. $E'_{\mathbf{XY}}$ differs from $E_{\mathbf{XY}}$ in that it only considers transitions associated with the transmission of fake packets (its derivation is omitted for the sake of space). Since we always start from the initial state $\mathbf{I}$, we only need to calculate $\mu_{\mathbf{I}}$.

IV. RESULTS

In this section we show the results obtained from the analytical model derived in the previous section. To validate our analytical results, as well as for comparing the performance of LOCALL with that of the CDM algorithm [11], we also used simulation. CDM takes a lightweight vertex-coloring approach. Specifically, sensor nodes are supposed to choose a different color from a set of available colors (in our terminology, colors correspond to slots). Initially, all sensor nodes are in *search* mode. At each round (period) *p*, a generic node *v* (**i**) picks up randomly a color from the set of available colors, and (**ii**) checks whether it has a conflict with any other node. If there is no conflict, node *v* enters *permanent* mode, selects *c* as its permanent color, and stops. Otherwise, it waits for a new round and performs the same actions again. The algorithm ends when *all* sensor nodes are in permanent mode.

We implemented both LOCALL and CDM using the ns-2 simulation tool [13]. We considered a single-hop network, where sensor nodes are located at a fixed distance (10 m) from the sink node. The transmission range was set to 15 m (according to the settings in [14]), while the carrier sensing range was set to 30 m, as in [15]. Unless stated otherwise, the other parameter values are as shown in Table II (power consumptions were derived from the CC2420 data sheet [16]). In each experiment, we performed ten independent replications, each of which consisted of 500 different slot acquisition processes. The results shown below are averaged over all the replications. We also derived confidence intervals by using the *independent replication* method and 99% confidence level.

TABLE II. PARAMETERS USED IN OUR ANALYSIS

Parameter	Value
N, N_s	10
Bit Rate	250 Kbps
Data Frame (Payload) Size	127(118) bytes
ACK Frame Size	11 bytes
N_B (slot acquistion phase)	8
Power Consumption in RX mode (P_{rx})	35.46 mW
Power Consumption in TX mode (P_{tx})	31.32 mW
Power Consumption in Idle mode (P_{idle})	0 mW

Figure 5 shows the probability distribution of the convergence time, for different number of sensor nodes, derived both from analysis (i.e., using Equation (6)) and simulation. In order to validate the analytical model, the initial randomization performed by the algorithm has not been considered in simulations. As it can be observed, analytical and simulation results almost overlap.

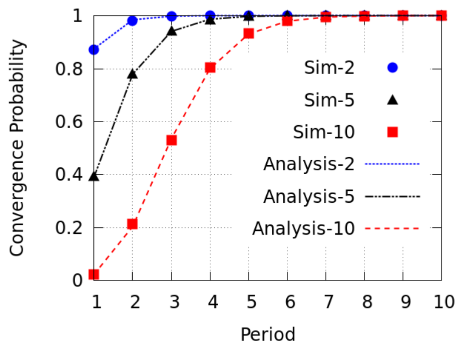

Figure 5. Convergence time of LOCALL for different number of nodes.

TABLE III. 95% CONVERGENCE TIME (# OF PERIODS).

# of nodes	LOCALL Simulation	CDM Simulation
2	2.00 (±0.00)	4.8 (±0.34)
5	3.80 (±0.43)	16.3 (±0.77)
10	5.10 (±0.32)	34.3 (±1.59)
20	8.00 (±0.41)	71.1 (±2.53)
30	10.50 (±0.54)	113.1 (±5.92)
40	12.70 (±0.50)	150.4 (±7.51)
50	14.80 (±0.43)	178.1(±9.63)

We also compared our algorithm with CDM in terms of convergence time. Since an analytical model for CDM is not available, we used simulation for comparison (for LOCALL we also considered the initial randomization). Table III reports the 95-th percentile of the convergence time distribution, i.e., the number of periods required to provide a complete schedule with a probability of, at least, 0.95. Our algorithm converges in a significantly shorter time due to its more efficient slot (color) selection strategy. In CDM, at each round (period) any sensor node selects a color (slot) at random, and checks for possible conflicts with other nodes. If a conflict is detected, the node tries a new color (slot) at random in the next round. Hence, a complete schedule is reached only after a number of rounds, which increases dramatically with the number of sensor nodes. Conversely, in LOCALL the contention is performed sequentially, for each single slot (color). The initial randomization reduces the number of potential competitors for each slot. Then, if a conflict is detected, the node tries the next slot in the same period, or the same slot in the next period (round). Table III shows that, for a sensor network with 50 nodes, with LOCALL a complete schedule is reached in about 15 periods (with a probability of 0.95). It should be emphasized that this is the time taken to obtain a *complete* schedule. However, individual sensor nodes may achieve their own slot in considerably less time.

For LOCALL we also investigated analytically the impact of the Contention Period (i.e., the N_B value) on the convergence time (Eqn. (6)). The obtained results are not shown here for the sake of space. We found that, as expected, increasing the Contention Period reduces the convergence time. However, to avoid undetected collisions (during slot allocation phase) the Contention Period must be shorter than the Transmission Period (see Figure 2). For 802.15.4, it can be shown that the maximum allowed value for N_B is 8.

Figure 6 shows the average energy consumed by the network in each period (derived from Eqn. (10)), when there are 10 nodes. As above, analytical and simulation results overlap, when there is no initial randomization. The latter reduces the average energy consumption as it drastically reduces the number of competitors per slot. The consumption in steady-state conditions is the same in both cases.

Finally, we computed (through Eqn. (11)), the total average energy consumed by the network due to slot allocation (i.e., without considering the transmission of data packets). The results are shown in Table IV which also includes simulation results. With 10 nodes, the energy overhead due to slot allocation is 3.32 mJ, which corresponds to the total energy consumed by the network during the data transmission phase in 2.2 periods (1.5 periods with initial randomization).

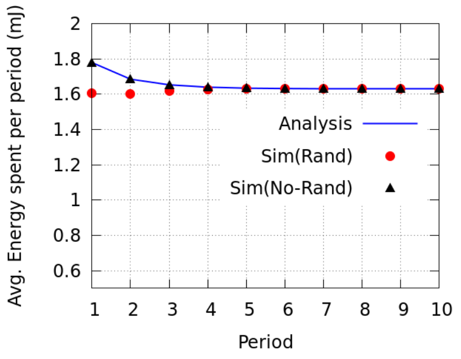

Figure 6. Average energy consumed by the sensor network when *N*=10.

TABLE IV. AVG. ENERGY SPENT FOR SLOT SCHEDULING (mJ)

# of nodes	Model	Simulation without Rand.	Simulation with Rand.
2	0.38	0.38 (±0.01)	0.38 (±0.00)
5	1.21	1.21 (±0.01)	1.02 (±0.01)
10	3.32	3.32 (±0.03)	2.28 (±0.02)

V. CONCLUSIONS

In this paper we have proposed a slot allocation algorithm for WSNs. Unlike many previous decentralized solutions, the proposed algorithm is localized, i.e., sensor nodes select their slot basing on local information only. We have developed an analytical model of the proposed algorithm, based on a Discrete Time Markov Chain, and derived the probability distribution of the *convergence time* (i.e., the time to obtain a complete schedule) and the average *energy consumption*. Our results show that (**i**) the proposed algorithm is able to converge much faster than another similar algorithm used for comparison, (**ii**) the schedule time increases with the number of nodes with a slope less than linear, (**iii**) the energy overhead due to slot scheduling is very limited. In this paper we have referred to a single-hop network. As a further step, we plan to extend our algorithm to multi-hop topologies.

REFERENCES

[1] G. Anastasi, M. Conti, M. Di Francesco, A. Passarella, "Energy Conservation in Wireless Sensor Networks: a Survey", *Ad Hoc Networks*, Vol. 7, N. 3, pp. 537-568, May 2009.

[2] R. Zurawski, "Networked Embedded Systems: An Overview" Chapter 1 in Networked Embedded Systems (R. Zurawski, Editor), pp. 1.11-1.16, CRC Press, 2009

[3] IEEE Standard for Information technology, Part 15.4; Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), IEEE Computer Society, 2006.

[4] J. Polastre, J. Hill and D. Culler, "Versatile Low Power Media Access for Sensor Networks", Proc. *ACM Conference on Embedded Networked Sensor Systems (SenSys 2004),* Baltimore, USA, 2004.

[5] IEEE std. 802.15.4e, Part. 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendament 1: MAC sublayer, IEEE standard for Information Technology, April 2012.

[6] O. Incel, A. Ghosh, B. Krishnamachari, "Scheduling Algorithms for Tree-Based Data Collection in Wireless Sensor Networks", Chapter 14 in *Theoretical Aspects of Distributed Computing in Sensor Networks* (S. Nicoletseas, J.D.P. Rolim, Eds.), 2011. Springer.

[7] S. Ramanathan, "A unified framework and algorithms for (T/F/C)DMA channel assignment in wireless networks", Proc. *IEEE INFOCOM* 1997, pp. 900-907.

[8] I. Rhee, A. Warrier, J. Min, L. Xu, "DRAND: Distributed Randomized TDMA Scheduling for Wireless Ad Hoc Networks", *IEEE Trans. on Mobile Computing*, vol. 8, no. 10, 2009.

[9] S. Coleri-Ergen and P. Varaiya, "PEDAMACS: Power Efficient and Delay Aware Medium ACcess protocol for Sensor networks," *IEEE Trans. on Mobile Computing*, vol. 5, no. 7, pp. 920–930, 2006.

[10] L. Paradis, Q. Han, A Data Collection Protocol for Real-time Sensor Applications, *Pervasive and Mobile Computing (PMC*), Vol. 5, No. 1, 2009.

[11] A. Motskin, T. Roughgarden, P. Skraba, L. Guibas, "Lightweight Coloring and Desynchronization for Networks", Proc. *IEEE INFOCOM 2009*, Rio de Janeiro, Brazil, April 19-25, 2009.

[12] T. Verhoeff, "Reward Variance in Markov Chains : a Calculational Approach", Proc. *Eindhoven FASTAR Days 2004.*

[13] Network Simulator Ns2, http://www.isu.edu/nsnam/ns.

[14] J. Zheng and M. J. Lee, "A comprehensive performance study of IEEE 802.15.4", *IEEE Press Book*, 2004.

[15] G. Anastasi, M. Conti, M. Di Francesco, "A Comprehensive Analysis of the MAC Unreliability Problem in 802.15.4 Wireless Sensor Networks", *IEEE Transactions on Industrial Informatics*, Vol.7, N.1, Feb. 2011

[16] Chipcon CC2420 Website, http://www.ti.com/product/cc2420.