# Anti-collision Protocols for Single-Reader RFID Systems: Temporal Analysis and Optimization

Thomas F. La Porta, Gaia Maselli, and Chiara Petrioli,

**Abstract**—One of the major challenges in the use of Radio Frequency-based IDentification (RFID) on a large scale is the ability to read a large number of tags quickly. Central to solving this problem is resolving collisions that occur when multiple tags reply to the query of a reader. To this purpose several MAC protocols for passive RFID systems have been proposed. These typically build on traditional MAC schemes such as aloha and tree based protocols. In this paper we propose a new performance metric by which to judge these anti-collision protocols: time system efficiency. This metric provides a direct measure of the time taken to read a group of tags. We then evaluate a set of well-known RFID MAC protocols in light of this metric. Based on the insights gained we propose a new anti-collision protocol and show that it significantly outperforms previously proposed mechanisms.

**Index Terms**—RFID, passive tags, single reader, anti-collision, MAC protocols.

## 1 Introduction

Radio-Frequency IDentification (RFID) is considered a key technology for item identification and efficient object tracking, and enables fundamental operations such as automatic inventory and management. An RFID system consists of radio frequency identification devices, named *tags*, that are able to communicate wirelessly to one or more *readers*. Tags are attached to objects that need to be identified and answer with their ID when inquired by a reader. Typical applications require cheap and unobtrusive identification tags to be attached to various different items (T-shirts, cereal boxes, books, animals, etc.). To this aim tags should be small, light, low cost and able to operate independently of batteries or external sources of energy. *Passive tags* fulfill this purpose by receiving the energy needed for communication by the RF carrier of the inquiring reader. The request message from a reader gives the tags enough energy to remodulate the signal so that their ID information is *backscattered* to the reader.

One of the major challenges for passive RFID systems is that of avoiding or solving collisions due to interference that might occur among readers and/or tags. Collisions occur when two or more tags simultaneously transmit to the same reader (tag-tag collisions). This is a serious problem when the density of RFID tags is high and for applications in which many RFID tags in the same area must be read. For example, if an RFID reader is used to take inventory in a warehouse with a high density of RFID tags placed on items on pallets, many collisions will occur increasing the amount of time required to complete the inventory. Likewise, if RFID is deployed to scan shopping carts at a store checkout counter, collisions amongst tags in the cart will delay the checkout process.

In this paper, we specifically address this problem in a *single-reader* scenario in which the reader has no knowledge of the tag population. Note that the presence of multiple readers in the same area may also cause reader-reader collisions [1], [2], [3], but we do not address this problem here.

Since the system is highly asymmetric (the reader is resource-rich while tags have very limited storage and computing capabilities, and are unable to hear the signal transmitted by the other tags and to detect collisions) channel access must be arbitrated by the reader. This is the basis of the many *anti-collision* protocols that have been proposed in the literature [4][5].

There are two main classes of anti-collision protocols so far proposed: *Aloha-based* and *tree-based*. *Aloha-based* protocols consider the channel to be slotted into intervals of time [6]. The start of each slot is signaled by the reader with a short message. Slots are issued in groups, or *frames* (from which the name Framed Slotted Aloha), whose size is set and communicated by the reader at the beginning of each frame. Each tag randomly and uniformly selects one slot, and transmits its ID during that slot time. At the end of the frame, tags that are successfully identified become silent, while tags that generated collisions keep trying in the following frames. The performance of aloha-based protocols is highly affected by the frame size. Because the cardinality of tag population to be identified is not known, frame sizing is a big issue for aloha-based protocols.

The second group of anti-collision protocols build on serial *tree* algorithms [7] (also known as walking tree algorithms). *Tree-based* protocols proceed more determin-

- *Thomas F. La Porta is with is with Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802.E-mail: tlp@cse.psu.edu*
- *Gaia Maselli and Chiara Petrioli are with Department of Computer Science, University of Rome "La Sapienza", Italy. E-mail: maselli@di.uniroma1.it (Gaia Maselli), petrioli@di.uniroma1.it (Chiara Petrioli)*

*This work was supported in part by National Science Foundation under grant CNS-0916171 and the European ARTEMIS CHIRON Project.*

Digital Object Indentifier 10.1109/TMC.2010.58 

istically: They iteratively query a subset of tags which match a given property until all tags are identified. These protocols are called tree-based because the identification process can be represented as a tree where the root is the set of tags to be identified, intermediate nodes represent groups of colliding tags answering the same reader request, and the leaves correspond to single-tag responses. Tree-based protocols differ in the way tags are queried (e.g., based on a counter stored in the tags, or on the binary structure of tag ID's).

Despite the wide range of anti-collision protocols proposed in the literature, an in-depth understanding of their relative performance is still lacking. In particular preliminary comparison between different solutions have failed to consider the ultimate performance metric – the time needed to complete the identification process.

In this paper we make the following contributions:

- We define a new performance metric by which to judge RFID anti-collision protocols: Time System Efficiency ($Time\_SE$). The $Time\_SE$ is a measure of the percentage of time successfully spent in identifying tags. Specifically, it is the ratio between the time taken to read all tags if there were no collisions and the total actual time taken to read the tags. Time_SE gives a clear idea of how much time is devoted to tag identification and how much is wasted in collisions or in idle rounds. A protocol that has Time_SE = 0.6 spends the $60\%$ of time in identifying tags, while $40\%$ of time is wasted in collision and idle rounds. The $Time\_SE$ is complemented with latency, which is a measure of the total time taken to resolve queries, and reflects the performance experienced by the user.
- Given the new $Time\_SE$ metric, we evaluate a set of the major anti-collision protocols previously proposed and characterize their behavior. This provides insight into the strengths and weaknesses of each protocol.
- Given the new $Time\_SE$ metric, we determine the optimal parameter settings to tune existing protocols. Using these settings maximizes their $Time\_SE$ (and minimizes the time taken to resolve queries).
- With the insight into the performance of existing protocols with respect to $Time\_SE$, we define a new anti-collision protocol, called Binary Splitting Tree Slotted Aloha (BSTSA), that is designed to provide the maximum $Time\_SE$. We show through extensive analysis and simulation that this protocol achieves very close to the maximum $Time\_SE$ and outperforms the other anti-collision protocols.

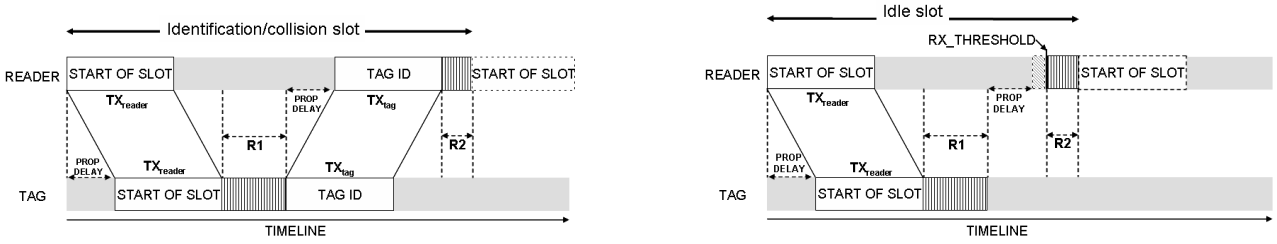
BSTSA uses a novel combination of several of the existing anti-collision protocols. In addition to achieving a high $Time\_SE$, BSTSA has the attractive property of being robust. While we show other protocols may perform well in networks of specific sizes, or when the number of tags to be read is known in advance, BSTSA works well across the full range of network sizes without any knowledge of the number of tags in advance. BSTSA also resolves a large percentage of queries quickly. Therefore it is ideal in applications in which mobile readers, perhaps being used by people or attached to forklifts in a warehouse, continuously update inventories as they move.

The rest of the paper is organized as follows. Section 2 presents the transmission model used in the paper. Section 3 presents an overview of the major anti-collision protocols proposed and presents a new analysis of each in terms of their $Time\_SE$. In this Section we also tune some of the protocols to overcome obvious bottlenecks in their performance. Section 4 presents the derivation of the optimal frame size for the aloha-based protocols considering $Time\_SE$, while Section 5 describes our new protocol BSTSA. In Section 6 we discuss the results of a ns2-based performance evaluation of BSTSA and other major representatives of the aloha-based and tree-based protocols. Finally, section 7 concludes the paper.

## 2 TRANSMISSION TIME MODEL

To perform a realistic evaluation of the temporal aspects of protocols, it is necessary to define a reference model that specifies time requirements for reader-to-tag ($R => T$) and tag-to-reader ($T => R$) communications. The transmission model we consider has been derived by the EPCglobal Specification Class-1 Gen-2 [8], that defines the physical and logical requirements for a passive-backscatter, interrogator-talks-first, radio-frequency identification system, based on Framed Slotted Aloha. From this standard, we drew out two important aspects that must be considered when evaluating transmission time. First of all, physical overhead has to be considered, as both $R => T$ and $T => R$ transmissions begin with a *preamble*. Because the $R => T$ preamble is not transmitted in each slot, but only in the first request issued at the beginning of a frame (when the reader signals the frame size and the start of the first slot), its effect is negligible. Conversely, because the $T => R$ preamble is sent at the beginning of each tag transmission, it has a high impact on protocol performance. The size of this preamble depends on data encoding. In the case of FM0 which is usually employed for single-reader scenarios, the preample is 6 bits long.

The second aspect to take into account is *link timing*: when estimating transmission duration propagation delay, transmission delay, and the reaction time of the tags must be considered. Propagation delay in both directions ($R => T$ and $T => R$) is a fixed time, estimated at $1/30$ $\mu s$ based on a reference distance of $10$m between the reader and the tags and on the propagation speed (i.e., the speed of light). Transmission delay depends on the transmission datarate and on the amount of bits to be transmitted. Reaction time depends on device characteristics and measures the reaction time from the end of a message reception to the start of a message transmission. On the tag side, it is estimated as $R1 =$




(a) Identification or collision slot.

(b) Idle slot.

Fig. 1. Link timing for reader-to-tag and tag-to-reader transmission during a slot.

$10/datarate$, while on the reader side it is estimated as $R2 = 1/datarate$ (according to EPC specification (see pag. 34 of [8])).

Figures 1(a) and 1(b) show the link timing for a message exchange between a reader and a tag in the case of a collision/identification slot or an idle slot, respectively[1]. The reader starts by transmitting a *start of slot* to the tags. This message is received by the tags after the transmission time, $TX\_reader$, and the propagation delay. After receiving the message, the reader takes time $R1$ to react and reply. The reader receives the response after the transmission time, $TX\_tag$, and the propagation delay. After receiving the tag response, the reader needs a reaction time $R2$ before being able to issue a new *start of slot*. If no tag responds to the reader, and idle slot results, shown in Figure 1(b). The reader realizes that no transmission is coming back from the tags after time $RX\_threshold$ which is the time at which the reader should receive the first bit of tag transmissions. In this case the slot ends when the $RX\_threshold$ elapses, and the reader issues a new *start of slot* after a reaction time $R2$.

We apply a similar model to tree-based protocols, in which the message sent by the reader is a query, and tags respond with their ID (we call this message exchange a *round*). The only difference between a slot and a round in terms of duration is the time taken by the reader message; in aloha-based protocols this is a *start of slot*, while in a tree-based protocol it is a *query*.

We use this model in our investigation and as the reference time model for our simulations. This allows us to have a common time reference that depends only on datarate and physical device characteristics. The model highlights that the duration of a slot or round strictly depends on the amount of bits transmitted by the reader (in the slot signaling or query message) and by the tag (in the response message). As a consequence, the duration of an idle slot or round that does not involve any tag response is shorter than the time taken by an identification or collision slot or round that involves the transmission of tag's ID. This characteristic plays an important role when comparing protocols from a temporal point of view.

## 3 Time System Efficiency and Analysis of Aloha Protocols

In this section we formally define our new performance metric, time system efficiency. We then analyze and evaluate the time system efficiency of a representative set of the major tree-based and aloha-based anti-collision protocols for RFID networks. This analysis is the first to show the efficiency of these protocols in terms of the *time* required to resolve queries of groups of tags. As such it provides insight into how these existing protocols may be optimized and how a new protocol may be designed to reduce inefficiencies in terms of time to resolve group queries.

### 3.1 System efficiency vs. time system efficiency

So far, system efficiency ($SE$) has been the most common metric used to evaluate anti-collision protocol performance. $SE$ is a measure of efficiency of the protocol in terms of its use of slots. Formally, $SE$ is the ratio between the minimum number of rounds[2] in which all tags could possibly be read, which is equal to the number of tags within range of the reader, and the actual number of rounds taken by a specific protocol:

$$SE = \frac{R_{ident}}{R_{tot}} = \frac{R_{ident}}{R_{idle} + R_{ident} + R_{coll}} \quad (1)$$

where $R_{ident}$ is the number of tags identified and $R_{tot}$ is the total number of rounds required to identify them, which is given by the sum of identification ($R_{ident}$), collision ($R_{coll}$), and idle ($R_{idle}$) rounds.

System efficiency does not consider the *time* taken by different rounds. For this reason we introduce the time system efficiency, $Time\_SE$, that accounts for the different duration of each round. Because idle rounds do not involve the transmission of tag IDs, they are shorter than identification and collision rounds, and consequently have a significantly lower impact on protocol performance. For this reason, when evaluating $Time\_SE$, idle rounds are normalized to the length of identification

1. For the purpose of our analysis, collision slots have the same duration of identification slots, as tags respond always sending their IDs.

2. From now on we consider a round the exchange of a reader message (i.e., time slot signaling in aloha-based protocols and query in tree-based protocols) and relative tag response.

and collision rounds (which have the same duration), through a multiplicative factor $\beta$, which represents their weight. Specifically, the time system efficiency $Time\_SE$ is defined by eq. 2.

$$Time\_SE = \frac{R_{ident}}{\beta R_{idle} + R_{ident} + R_{coll}} = \frac{R_{ident}}{R_{tot} + (\beta - 1) \cdot R_{idle}} \quad (2)$$

When protocol designers aim to maximize $SE$, they attempt to read all tags in the minimum total number of rounds without regard to whether the additional rounds required are a result of collisions or idle rounds. When aiming to maximize $Time\_SE$ a tradeoff may be made in which more idle slots are acceptable if there is a large reduction in collision slots.

In the following we go through the main anti-collision protocols and analyze their efficiency in terms of both $SE$ and $Time\_SE$.

### 3.2 Tree-based protocols

Tree-based protocols draw on tree algorithms for a packet broadcast channel [7][9]. The tag identification process is deterministic, and is based on iteratively querying a subset of tags which match a given property until all tags are identified. These protocols are called tree-based because the identification process can be represented as a tree where the root is the set of tags to be identified, intermediate nodes represent groups of colliding tags answering the same request from the reader, and the leaves correspond to single-tag responses. Protocols in this class differ in the way tags are queried: according to the generation of a binary random number in Binary Splitting [10], and the binary structure of tag IDs in Query Tree [11].

#### 3.2.1 Binary Splitting (BS) [10]

BS recursively splits answering tags into two sub-groups until obtaining single-tag groups. Each tag maintains a counter (initially set to zero). Tags with the counter value equal to zero answer the reader query, while others remain silent until their counter decreases to zero. The value of the tag counter is modified depending on whether the query results in a collision, identification, or no-answer. In case of collision, colliding tags add a random binary value to their counter, i.e., they are split into two subsets: those whose counter value is zero and those whose counter value is one. The tags which were not involved in the collisions increase their counters by one. In case of identification or no answer, all the tags decrease their counter by one, so that those with the counter at one will answer the next query.

It has been shown that for a network of $n$ tags the expected number $R_{tot}$ of rounds (i.e., queries) needed to identify all tags is $2.881n - 1 \leq R_{tot} \leq 2.887n - 1$ [9]. The system efficiency is thus obtained by eq. 3

$$SE_{BS} = \frac{n}{2.88 \cdot n} = 0.34 \quad (3)$$

To estimate the time system efficiency it is necessary to evaluate the number of idle rounds. We refer to the analysis performed in [12] for an $r$-ary tree to derive the number of idle rounds in a binary splitting tree, as in eq. 4

$$R_{idle} = \frac{R_{tot}}{2} - n + \frac{1}{2} \quad (4)$$

while the sum of identification and collision rounds is estimated as in eq. 5.

$$R_{ident} + R_{coll} \simeq 2.44n - \frac{1}{2} \quad (5)$$

It follows that time system efficiency is estimated as in eq. 6, considering $\beta = 0.13$ (which is the ratio among idle and collision rounds obtained for 96-bit IDs and 40Kbps channel datarate, according to the EPC global standard specification [8]).

$$Time\_SE_{BS} = \frac{n}{\beta(\frac{R_{tot}}{2} - n + \frac{1}{2}) + 2.44n - \frac{1}{2}} \simeq 0.40 \quad (6)$$

#### 3.2.2 Query Tree (QT) [11]

QT queries tags according to the binary structure of their ID. The reader interrogates tags by sending them a string, and only those tags whose IDs have a prefix matching that string respond to the query. At the beginning, the reader queries all tags: this is implemented by including a NULL string in the query. If a collision occurs, then the string length is increased by one bit until the collision is solved and a tag is identified. The reader then starts a new query with a different string. In particular if tag identification occurs with a string $q0$ the reader will query for string $q1$. The resulting binary tree has nodes at the $i-$th level labeled with all the possible values of a prefix of length $i$ (e.g. nodes at level 1 contain prefixes 0 and 1, nodes at level 2 prefixes 00, 01, 10, 11 and so on). The exploration of a subtree is skipped in case there is only one tag matching the prefix stored in the subtree root (i.e. if a tag identification occurs when the reader queries with the subtree root prefix).

In case of uniform ID distribution, the tree induced by the query tree is analogous to the tree induced by the BS protocol. This is because a set of uniformly distributed tags splits approximately in equal parts at each query, like in the BS protocol. Hence the QT protocol presents the same system efficiency and time system efficiency of BS protocol, estimated respectively as in eq. 3 (i.e., $34\%$) and eq. 6 (i.e., $40\%$).

#### 3.2.3 Query Tree Improved (QTI) [11]

QTI is an enhancement of the QT protocol. QTI optimizes the number of queries by avoiding those that will certainly produce collisions. As an example, consider the case in which prefix "$p$" produces a collision, while prefix "$p0$" results in no tag answers. Then the reader skips prefix "$p1$" that will certainly produce collision and queries directly for "$p10$" and "$p11$". The expected

TABLE 1
System efficiency vs Time system efficiency for tree-based protocols.

| | SE | Time_SE |
|---|---|---|
| BS | 0.34 | 0.40 |
| QT | 0.34 | 0.40 |
| QTI | 0.39 | 0.41 |

number of rounds needed by QTI to identify all tags has been estimated as $2.6607n \leq R_{tot} \leq 2.665n - 1$ [9]. Consequently, the QTI system efficiency is given by eq. 7

$$SE_{QTI} = \frac{n}{2.66 \cdot n} = 0.39 \quad (7)$$

As only collision rounds are reduced, the number of idle rounds do not change with respect to the BS or QT tree. Thus, the time system efficiency can be stimated as eq. 8

$$Time\_SE_{QTI} = \frac{n}{\beta(\frac{R_{tot}}{2} - n + \frac{1}{2}) + 2.44n - \frac{1}{2}} \simeq 0.41 \quad (8)$$

Table 1 summarizes the system efficiency and time system efficiency values for the tree-based protocols.

### 3.3 Aloha-based protocols

Protocols in this class consider the channel to be slotted into intervals of time, whose duration is equal to the tag's ID transmission time. When a reader issues a *start of frame* it includes the number of slots in a frame. The tags then randomly pick a slot in which to reply. Collisions occur if two or more tags pick the same slot. The process repeats itself until all tags are identified. Once a tag is identified it no longer responds to the *start of frame*.

The performance of these protocols is dominated by the selection of the number of slots in a frame. If there are too few slots, many collisions occur and very few tags are identified in each frame. If there are too many slots, many tags may be identified, but many idle slots are wasted. Compounding this problem is the fact that a reader typically does not know a priori how many tags are to be read, so it has no basis on which to set its frame size. For this reason, a reader typically issues a first frame of a predefined size.

Thus there are two important and strictly related issues for aloha-based protocols: 1) estimating the number of tags; and 2) properly tuning the frame sizes.

A previous study [13] has shown that it is reasonable to fix the initial frame size to a value such as 128, which represents a trade-off for both small and large networks. In cases of small networks (e.g. few hundreds of tags), this allows fast identification of all tags, without incurring too many idle slots. In cases of bigger networks (e.g. 1000-2000 tags), a frame size of 128 slots does not incur many more collisions than those that would occur with bigger frame sizes (e.g., 256 slots). Using even bigger initial frames (i.e., thousands of slots) would result in poor performance in cases of small networks.

Intuitively, minimizing the number of wasted slots in a slotted aloha system is the same as maximizing the throughput. Thus the maximum throughput will result in the highest $SE$. It is well known that for slotted aloha system throughput is maximized when the normalized offered load is equal to one, i.e., in our case the number of slots is made equal to the number $n$ of tags to be read. Therefore, to maximize $SE$ the optimal size of a frame is $N = n$. This result is verified in [14].

Still, the problem of estimating the number of tags to be read must be solved. Several methods have been proposed to estimate tag cardinality [15]. The most accurate and widely adopted method exploits the knowledge on the number of idle, identification, and collision slots that occur in the immediately preceding frame. This estimation, based on Chebyshev's inequality, is computed by searching for the number of tags $n$ such that the distance between the triple of estimated values $\langle a_0^{N,n}, a_1^{N,n}, a_k^{N,n} \rangle$ and the triple of observed values $\langle c_0, c_1, c_k \rangle$ is minimum, as defined by Equation 9 [16].

$$\epsilon(N, c_0, c_1, c_k) = \min_n \left| \begin{pmatrix} a_0^{N,n} \\ a_1^{N,n} \\ a_{k\geqslant 2}^{N,n} \end{pmatrix} - \begin{pmatrix} c_0 \\ c_1 \\ c_k \end{pmatrix} \right| \quad (9)$$

Here $N$ denotes the size of the completed frame, $a_0^{N,n}$ is the expected number of empty slots (*idle slots*), $a_1^{N,n}$ is the expected number of slots with one responding tag (*identification slots*), and $a_k^{N,n}$ is the expected number of slots in which multiple tags reply (*collision slots*). The values $a_0^{N,n}$,$a_1^{N,n}$, and $a_k^{N,n}$ depend on the frame size $N$ and on the number of tags, $n$, and are estimated according to Equation 10.

$$a_r^{N,n} = N \times \binom{n}{r} \left(\frac{1}{N}\right)^r \left(1 - \frac{1}{N}\right)^{n-r} \quad (10)$$

The condition $N = n$ coupled with Chebyshev's estimation of tags cardinality represents the approach most widely taken by current aloha-based protocols. In the following we go through the main aloha-based solutions, highlighting protocol differences and studying their efficiency in terms of both the number of rounds (i.e., $SE$) and the time (i.e., $Time\_SE$).

#### 3.3.1 Framed Slotted Aloha (FSA) [16], [17]

FSA represents the starting step for the aloha mechanism in the RFID environment. This protocol issues a first frame, and then sizes the subsequent frame according to the outcomes of the first. All tags that have not been identified in the first frame participate in the following one. The process is repeated on subsequent frames, until all tags are identified.

To evaluate the performance of FSA, it is necessary to estimate the total number of slots, $R_{tot}$, issued during

protocol execution. $R_{tot}$ is given by the sum of slots executed in each frame, which varies in each frame. The size of a frame depends on the number of tags that are not identified in the previous frame. Given $n$ tags and $N$ slots in a frame, the success probability $P_s(n,N)$ of identifying one tag is given by $P_s(n,N) = n/N(1-1/N)^{n-1}$. It follows that the expected number of tags, $T_i$, that are identified is given by $T_i = N \cdot P_s(n,N) = n(1-1/N)^{n-1}$, while the expected number of tags, $T_c$, that collide, and hence are unidentified, is $T_c = n - T_i = n - n(1-1/N)^{n-1}$.

We now distinguish two cases: i) the number $n$ of tags to identify is known and hence the initial frame can be properly sized ($N = n$); ii) the number of tags to identify, $n$, is unknown and a predefined initial frame size $N$ is used ($N \neq n$).

In the first case ($N = n$) the percentage of identified tags is given by $P_s(n) = (1-1/n)^{n-1}$, where for brevity we do not show in the notation the frame size that is equal to the number of tags ($N = n$). The percentage of unidentified tags (failure) is given by $P_f(n) = 1 - P_s(n) = 1 - (1-1/n)^{n-1}$. It follows that the expected numbers of identified and unidentified tags during the first frame are given, respectively, by $P_s(n) \cdot n$ and $P_f(n) \cdot n$. The latter provides the size for the following frame. Looking at the complete protocol execution, the frame size varies in the following way: at the first frame ($i = 0$) the frame size is $n_0 = n$, at the second frame ($i = 1$) the frame size is $n_1 = P_f(n_0) \cdot n_0 \quad (= P_f(n) \cdot n)$, at the third frame ($i = 2$) frame size is $n_2 = P_f(n_1) \cdot n_1 = P_f(n_1) \cdot P_f(n_0) \cdot n_0$, and so on. Hence the total number of slots, $R_{tot}$, is given by eq. 11:

$$R_{tot} = n_0 + \sum_{i=1}^{F} n_i = n + \sum_{i=1}^{F} (\prod_{j=0}^{i-1} P_f(n_j)) \cdot n \qquad (11)$$

where $F$ represents the expected number of executed frames, and the product $(\prod_{j=0}^{i-1} P_f(n_j)) \cdot n$ gives the number of slots executed at frame $i$. When only one tag has to be identified, meaning that $(\prod_{j=0}^{i-1} P_f(n_j)) \cdot n = 1$, then a single slot is executed and protocol execution ends.

Given the total number of slots, $R_{tot}$, executed by the FSA algorithm to identify $n$ tags, we can estimate the system efficiency $SE = n/R_{tot}$. Figure 2 shows the system efficiency function by varying the number of tags. For very small networks (i.e., less than 10 tags) system efficiency is over $40\%$, while for larger networks it tends to $36.8\%$, that is the classical value for Framed Slotted Aloha throughput in multi-access networks [6].

To estimate $Time\_SE$, we need to estimate the number $R_{idle}$ of idle slots to evaluate eq. 2. The expected number of idle slots $R_{idle}$ can be estimated as follows (eq. 12).

$$R_{idle} = \sum_{i=0}^{F} P_{idle}(n_i) \cdot n_i \qquad (12)$$

where $P_{idle}(n_i) = (1-1/n_i)^{n_i}$. The $Time\_SE$ function is plotted in Figure 2. We set $\beta = 0.03$ which corresponds

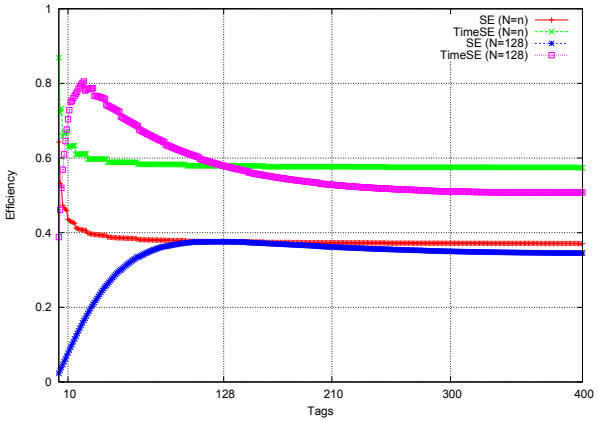



Fig. 2. System efficiency and time system efficiency for FSA protocol.

to the ratio among idle and collision rounds obtained for 96-bit IDs and a 40 Kbps channel datarate, according to the EPC global standard specification [8], and N=n. $Time\_SE$ shows the same behavior of $SE$ but is about $50\%$ higher. For a small number of tags, $Time\_SE$ ranges from near $80\%$ (for 2 tags) to $40\%$ (in the case of 20-30 tags). For larger numbers of tags $Time\_SE_{FSA} \approx 0.57$.

The comparison between the (round-based) system efficiency and time system efficiency clearly shows that considering real temporal aspects, protocol efficiency increases because of the low impact of idle slots.

We now analyze the case in which the number of tags to identify, $n$, is not known and the initial frame size $N$ is set to a predefined value ($N \neq n$). The main difference is in the size of the first frame. When the first frame is completed, it is possible to estimate the number of tags and tune the size of the following frame according to this estimation, falling in the case $N = n$. The percentage of tags that are identified in the first frame is given by $P_s(n,N) = n/N(1-1/N)^{n-1}$, and the expected number of identified tags is $T_i = N \cdot P_s(n,N) = n(1-1/N)^{n-1}$. The expected number of unidentified tags is then given by $T_c = n - T_i = n - n(1-1/N)^{n-1}$. From the second frame, thanks to tag estimation we fall in the case $N = n$. It follows that the number of slots required by the FSA protocol to identify $n$ tags with and initial frame of $N$ slots is: $n_0 = N$ at the first frame, $n_1 = n - n(1-1/N)^{n-1}$ in the second frame, $n_2 = P_f(n_1) \cdot n_1 = P_f(n_1) \cdot P_f(n,N) \cdot n$ in the third frame, and so on. Then the total number of slots is given by eq. 13:

$$R_{tot} = N + n_1 + \sum_{i=1}^{H} (\prod_{j=1}^{i-1} P_f(n_j)) \cdot n_1 \qquad (13)$$

where H represents the expected number of executed frames after the first two.

The number of idle slots, $R_{idle}$, can be estimated considering that an estimation of the number of tags is possible only after completing the first frame, whose length is $N$, and ratio of idle slots is $P_{idle}(n,N) \cdot N$. After that, the size of the following frame is estimated

as $n_1 = P_f(n, N) \cdot n$ and the percentage of idle slots is calculated as $P_{idle}(n_1) \cdot n_1 = P_{idle}(n_1) \cdot P_f(n, N) \cdot n$. Summing the idle slots on all frames we get:

$$R_{idle} = N \cdot P_{idle}(n, N) + \sum_{j=1}^{H+1} P_{idle}(n_j) \cdot n_j \qquad (14)$$

The system efficiency $SE = n/R_{tot}$ and the $Time\_SE = \frac{n}{R_{tot}+(\beta-1)\cdot R_{idle}}$ for $N = 128$ are shown in Figure 2. System efficiency is very low when the number of tags is much smaller than the frame size (from a few to about 50 tags). It reaches the maximum (i.e., $36.8\%$) when the number of tags equals the frame size ($n = N = 128$), and then slightly decreases due to the under-estimation of the size of the first frame. As the initial frame size is lower than the number of tags, it produces a system efficiency on the first frame that is below the optimum obtained for 128 tags. On subsequent frames it is possible to exploit tag estimation to properly tune the frame size, yielding a higher (possibly maximum) system efficiency. Hence for $n > 128$, the difference between the values of the system efficiency in the case of $n = N$ and in the case of $N = 128$ is due to the impact of the error on the first frame.

When considering temporal aspects, the impact of different types of rounds becomes clearly visible. Because idle rounds are much shorter than collision and identification rounds, time system efficiency achieves its maximum when the number of tags is between 20 and 30. In that case, almost all tags are identified during the previous frame, collisions are minimum and idle rounds have minor impact on time system efficiency, which achieves $80\%$. Then, as collisions increase, the time system efficiency decreases and crosses the curve for the case of $N = n$ when the number of tags equals the frame size ($n = N = 128$). For a higher number of tags ($n > 128$) the time system efficiency decreases due to the impact of the higher number of collisions in the first frame, with respect to the case of $n = N$. These results highlight the need for solutions that overcome the limitation due to the use of a fixed initial frame size, and allow for a better frame tuning, enabling higher efficiency.

A subsequent proposal, the Enhanced Dynamic Framed Slotted Aloha (EDFSA) [14] slightly improves efficiency over the classical Framed Slotted Aloha by reducing the number of collisions when there are many tags. EDFSA's main advantage is that only a subset of tags is queried to estimate the whole set, reducing collisions. However, it does not solve the problem on the initial frame size, showing performance comparable to FSA.

### 3.3.2 Tree Slotted Aloha (TSA) [13]

TSA deals with collisions more efficiently by using a different way of grouping tags that are queried in the same frame. After the first frame, a new set of "child" frames is allocated, each devoted to solving the collisions which have occurred in a given slot of the first frame. Only the (few) tags which transmitted in that slot participate in the corresponding frame. The approach is repeated: If collisions occur in one of the frames allocated to solve collisions (say, frame $i$), new frames are allocated to solve such collisions (one for each collision slot in frame $i$). This is possible by estimating the number of tags colliding in each slot, and then allocating a properly sized frame to solve the collisions which have occurred in such slot. More precisely, if $n_i$ is the estimated number of transmitting tags in reading cycle $i$ (computed using Chebyshev's inequality), $c_1^i$ is the number of identified tags during the $i-$th cycle, and $c_k^i$ the number of slots with collisions, then the frame size of the $(i+1)-$th reading cycle is given by $l_{i+1} = \lfloor (n_i - c_1^i)/c_k^i \rfloor$.

TSA's strength lies in generating a new frame for each colliding slot, which avoids any re-mix of colliding tags. However, tag estimation remains a weak point. In TSA the estimated number of tags $n$ (Equation 9) is searched in the range $[c_1 + 2c_k, 2(c_1 + 2c_k)]$. While the lower limit is correct, as at least two tags transmit in a colliding slot, the upper limit is too low to reflect realistic situations comprising a large number of tags. This number is upper bounded by $4$ times the initial frame size $128$, i.e., $512$. As a result, the size of the following frame is set to a value that is too small for networks with a large number of tags. For example, considering a scenario with $5000$ tags and an initial frame size of $128$ slots, it is highly likely that $c_1 = 0$, i.e., that no tag is identified in the first frame. In this case, $2(c_1 + 2c_k) = 4c_k$ and therefore the size of the frame at level 1 is $\left\lfloor \frac{4c_k}{c_k} \right\rfloor = 4$. This is a large underestimation as the expected number of tags colliding in a slot is around $40$. Therefore, many collisions happen at level 1 and in the following round, in which the frame size will be around $4$ and the number of tags per slot reduces to around $10$. This shows that the upper bound $2(c_1 + 2c_k)$ is not suitable for large networks (with a small initial frame size), and a better bound is needed. However, in the case of a frame with all collision slots, it is not possible to give a feasible interval in which to search for the $n$ value, as the number of tags that may generate collisions on all slots may be highly variable. Table 2 shows the triple of estimated values $\langle a_0, a_1, a_k \rangle$, and their distance from the the observed values $\langle c_0, c_1, c_k \rangle = \langle 0, 0, 128 \rangle$ (when there are all collisions), by varying the number $n$ of tags. The vectorial distance between the two triples become extremely small (close to 0) when $n \geq 1000$, and it keeps close to zero (with very small decrease) by (largely) increasing the number of tags. The number of tags that generates collisions on all slots can be 1000 as well as 2000, or even more. Hence, it is not possible to give an upper bound to the number of tags. For this reason, we propose the *unbounded* search, according to which the smaller number $n$ of tags for which the vectorial distance is next to zero (in this case $n = 1000$) is selected. This

TABLE 2
Estimation accuracy in the case $N = 128$, and $c_0, c_1 = 0$, $c_k = N$

| n | vect. distance | $a_0$ | $a_1$ | $a_k$ |
|---|---|---|---|---|
| 256 | 64.671 | 17.187 | 34.645 | 76.167 |
| 500 | 16.211 | 2.536 | 9.983 | 115.482 |
| 700 | 4.537 | 0.528 | 2.912 | 124.560 |
| 800 | 2.337 | 0.241 | 1.519 | 126.240 |
| 900 | 1.188 | 0.110 | 0.780 | 127.110 |
| 1000 | 0.598 | 0.050 | 0.396 | 127.554 |
| 1500 | 0.017 | 0.001 | 0.012 | 127.987 |
| 2000 | 0.0005 | 0.00002 | 0.0003 | 127.9997 |

method improves the tag estimation with respect to the *bounded* search, that stops at $2(c_1 + 2c_k)$, but it is still limited.

The cost of the TSA protocol can be estimated by referring to random hash trees [18], a data structure that is built in the following way. Given $X_1, \cdots, X_n$, $n > 1$, independent uniform $[0; 1]$ random numbers, we partition $[0; 1]$ into $n$ equal intervals of length $1/n$ each, and place all points in the intervals. Let $N_1, \cdots, N_n$ be the cardinalities of the intervals (thus, $\sum_i N_i = n$). The data partitioning process is repeated recursively for every interval that contains two or more points until all intervals contain either zero or one point. The root node represents $[0; 1]$, and each node represents a given interval. All internal nodes have at least two of the $X_i$'s, while all leaf nodes have one or zero of the $X_i$'s.

The size $s_n$ of a random hash tree, defined as the number of nodes in the tree, corresponds to the total number of rounds required by TSA protocol to identify $n$ tags (idle + identification + collision). It can be estimated as in eq. 15

$$s_n = \begin{cases} 1, & 0 \leq n \leq 1 \\ 1 + n \sum_{i=0}^{n} P\{B(n, 1/n) = i\} \cdot s_i & n \geq 2 \end{cases} \quad (15)$$

which gives $2.3020238 \cdot n$. Hence, when $n = N$, TSA system efficiency is given by eq. 16

$$SE_{TSA} = \frac{R_{ident}}{2.3020 \cdot R_{ident}} = \frac{1}{2.3020} = 0.43 \quad (16)$$

The value $2.3020 \cdot n$ corresponds to $2.3020 \cdot R_{ident}$ and represents the total number of rounds needed to complete the identification process. To estimate the time system efficiency it is necessary to estimate the number of idle rounds, $R_{idle}$. To this end, we estimate the number $t_n$ of slots filled with at least one tag transmission (i.e., not idle) and then we find the estimated number of idle slots as $R_{idle} = s_n - t_n$. The estimated number of slots with transmission is given by:

$$t_n = \begin{cases} 1, & 0 \leq n \leq 1 \\ 1 + n \sum_{i=1}^{n} P\{B(n, 1/n) = i\} \cdot s_i & n \geq 2 \end{cases} \quad (17)$$

which differs from $s_n$ only for the starting index of the summation (we do not consider empty slots). Numerical computation shows that $t_n \approx 1.59 \cdot n$. It follows that $R_{idle} = 0.71 \cdot n$ which corresponds to about $0.31$ of the total number of slots. Hence the time system efficiency for TSA protocol, in the case the number of tags $n$ is known, is given by eq. 18

$$\begin{aligned} Time\_SE_{TSA} &= \frac{R_{ident}}{(\beta - 1) \cdot R_{idle} + s_n} \\ &= \frac{n}{2.30 \cdot n + (\beta - 1) \cdot 0.71 \cdot n} = 0.62 \end{aligned} \quad (18)$$

### 3.3.3 Dynamic Tree Slotted Aloha

(Dy_TSA) [19]: Dy_TSA represents an enhancement of the TSA protocol coping with inaccurate tag estimation, due to large numbers of tags and a small initial frame size. The idea is to exploit the knowledge gained during previously completed frames to estimate the number of unread tags. The protocol is based on the assumption that the allocation of a tag in a slot is completely independent of the behavior of other tags. Therefore, tags are uniformly distributed in available slots, and the number of tags that fall in a slot is given by the binomial distribution. The expected value of the number of tags in a slot is $E[X] = \frac{n}{N}$.

Let us consider a frame of size $N$, with a population of $n$ tags. At the end of the frame, in case there were colliding slots (i.e., $c_k > 0$), then Chebyshev's estimation is used to estimate the number of tags that participated in the frame, and $c_k$ new child frames are issued, one for each colliding slot in the parent frame. The participants in each new frame are only the tags that collided in the corresponding slot in the parent frame. The new frames to be executed are estimated to have a size given by Equation 9. However, when the first of these sibling frames is completed, the knowledge on the number of tags that were found in the frame can be exploited to refine tag estimation and accordingly adapt the size of sibling frames to be executed. When the second frame is completed, the knowledge on the first two frames can be useful to refine the size of the remaining frames, and so on. The more frames are completed in a group of sibling frames, the higher the accuracy achieved in estimating the number of tags that are going to participate in the following frames.

More generally, let us consider the execution of the $i$th frame at level $l$, $1 < i \leq c_k^{l-1}$, where $c_k^{l-1}$ represents the number of colliding slots in the parent frame at the previous level. The size of the $i$th frame $S_i$ at level $l$ is estimated as the mean value of the tags that have been identified in the sibling frames previous to $i$. Specifically, if $t_j$ is the number of tags that participated in the frame $j$ at level $l$, then

$$S_i = \frac{1}{i-1} \sum_{j=1}^{i-1} t_j. \quad (19)$$

As TSA proceeds in a depth-first order, in cases in which a frame experiences collisions, they are resolved going down in the tree, and only when all collisions in a frame have been resolved, the next sibling frames at the same level can be executed. This allows us not only to exploit knowledge of previous frames, but also to recursively apply the estimation method on deeper levels of the tree, whenever multiple collision slots are present in a frame.

The convergence towards a proper frame size tuning brings Dy_TSA system efficiency closer to the TSA theoretical optimum. Hence Dy_TSA efficiency is upper bounded by TSA efficiency (estimated for the case in which tag population is known), which is estimated at about $43\%$ system efficiency, and about $62\%$ time system efficiency.

## 4 Optimal Frame Sizing

So far, the optimal frame tuning in aloha protocols has ignored that different types of slots have different durations. Idle slots are much shorter than identification and collision slots. Intuitively, to maximize $Time\_SE$ the protocol should *over-allocate* frames. This will trade collision slots with idle slots which have a much lower cost. In fact, given the large discrepancy in the relative weights between idle and collision slots in actual systems (recall that $\beta$ in eq. (2) is 0.03), it is worth experiencing a large number of additional idle slots to eliminate a small number of collision slots. The open question we discuss here is how large should the overestimation be.

Specifically if we consider the $Time\_SE$ as defined in eq. (2), we can obtain the optimal frame sizing in the following way. Let $a_1$, $a_k$, and $a_0$ be the number of identification, collision and idle slots in a frame, respectively, as defined in eq. (10) (for brevity we omit the superscripts $N, n$). Consequently we get $a_0 = N \times (1-1/N)^n$, $a_1 = n \times (1-1/N)^{n-1}$, and $a_k = N - a_0 - a_1$.

The $Time\_SE$ is thus defined by Eq. 20

$$Time\_SE = \frac{a_1}{\beta a_0 - a_0 + N} = \frac{n\left(1-\frac{1}{N}\right)^{n-1}}{(\beta-1)N(1-\frac{1}{N})^n + N} \tag{20}$$

To obtain the optimal frame size $N$ for a given number of tags, we compute the maximum value of $Time\_SE$ by deriving Eq. 20, and posing $\frac{\partial Time\_SE}{\partial N} = 0$. The maximum is achieved when

$$(\beta-1)\left(1-\frac{1}{N}\right)^n + 1 = \frac{n}{N} \tag{21}$$

We have plotted the functions on the left and right sides of Eq. 21 in Fig 3 for varying values of $n$ and $\beta = 0.03$ (which is the ratio among idle and collision rounds obtained for 96-bit IDs and 40Kbps channel datarate, according to the EPC global standard specification [8]). The crossing points in Fig. 3 represents the values for which the left and right side of Eq. 21 are equal when $n = 1000, 2000, 3000$. To obtain a relation between the number of tags $n$ and the optimal frame size $N$ we can apply linear interpolation, which gives

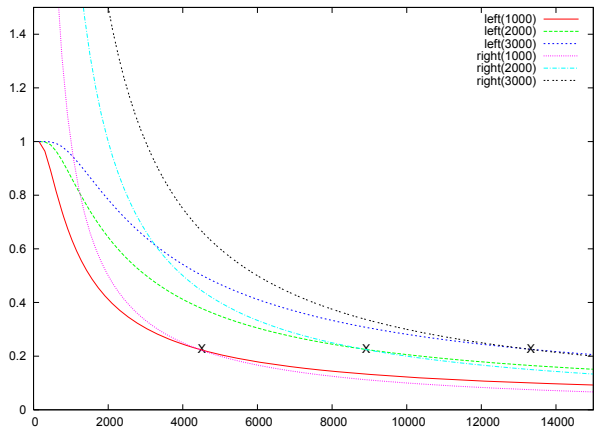



Fig. 3. Left and right side functions of eq. 21.

$$N = 4.406 \cdot n - 1 \tag{22}$$

The above equation tells us that when the frame size is about $4.4$ times the number of tags, the time system efficiency is maximum, and it is near $80\%$ for a single frame.

The $\beta$ value plays an important role when finding the overestimation factor that maximizes time system efficiency. Table 3 shows how frame overestimation and relative time system efficiency change by varying the $\beta$ value.

The optimal frame sizing improves the performance of framed slotted aloha protocols. As a proof we estimate the $Time\_SE$ of the best performing aloha protocol, TSA. Analogous to the analysis performed for TSA when the number of tags $n$ is known and the frame size is set equal to the number of unidentified tags ($N = n$), we evaluate the number of slots executed by the TSA protocol when the optimal overestimation is applied ($N = 4.406 \cdot n - 1$) as in eq. 23

$$s_n = \begin{cases} 1, & 0 \le n \le 1 \\ 1 + N\sum_{i=0}^{n} P\{B(n,1/N) = i\} \cdot s_i & n \ge 2 \end{cases} \tag{23}$$

which gives $5.252 \cdot n$. Substituting this value is eq. 1 we obtain the expected $SE$ of TSA with optimal frame size to be $SE_{TSA} = \frac{1}{5.252} = 0.19$

To estimate the $Time\_SE$ we calculate the expected number $t_n$ of slots filled with at least one tag transmission (i.e., not idle), so that we can obtain the number of idle slots as $R_{idle} = s_n - t_n$. The estimated number of slots with transmission is given by:

$$t_n = \begin{cases} 1, & 0 \le n \le 1 \\ 1 + N\sum_{i=1}^{n} P\{B(n,1/N) = i\} \cdot s_i & n \ge 2 \end{cases} \tag{24}$$

which gives $1.086 \cdot n$. It follows that the number of idle rounds is $s_n - t_n = 4.164 \cdot n$. Substituting in eq. 2 we get

TABLE 3
Optimal frame size and maximum time system efficiency for different $\beta$ values.

| $\beta$ | 1 | 0.5 | 0.33 | 0.25 | 0.20 | 0.10 | 0.06 | 0.05 | 0.04 | 0.03 |
|---|---|---|---|---|---|---|---|---|---|---|
| **N** | $1 \cdot n$ | $1.3 \cdot n$ | $1.53 \cdot n$ | $1.72 \cdot n$ | $1.89 \cdot n$ | $2.55 \cdot n$ | $3.20 \cdot n$ | $3.48 \cdot n$ | $3.85 \cdot n$ | $4.40 \cdot n$ |
| **TSE** | 0.36 | 0.46 | 0.52 | 0.56 | 0.59 | 0.67 | 0.73 | 0.75 | 0.77 | 0.80 |

TABLE 4
TSA performance by optimizing SE (i.e., $N = n$) and Time_SE (i.e., $N = 4.4 * n - 1$)

| Tags | Opt. SE | | | Opt. Time_SE | | |
|---|---|---|---|---|---|---|
| | SE | Time_SE | Latency | SE | Time_SE | Latency |
| 1000 | 0.38 | 0.52 | 6.91 | 0.28 | 0.64 | 5.30 |
| 3000 | 0.37 | 0.51 | 21.0 | 0.20 | 0.68 | 14.8 |
| 5000 | 0.37 | 0.51 | 35.3 | 0.20 | 0.71 | 23.5 |

that the expected $Time\_SE$ for the TSA with optimal frame sizing is $TSE_{TSA} = 0.82$.

To show the importance of the temporal efficiency we compare the results achieved by optimizing respectively the SE and the Time_SE (simulation details are given in Section 6). Table 4 summarizes the results for these metrics and absolute latency (measured in seconds), for a few scenarios ($n$=1000,2000,3000), when the TSA protocol is optimally tuned for SE and Time_SE. Results show that when we optimize for SE (i.e, $N = n$), Time_SE and absolute latency are not good. When we optimize on Time_SE (i.e., $N = 4.4 * n - 1$), the Time_SE increases (up to 40% in the case of 5000 tags) and latency drastically decreases (33% for 5000 tags).

The trend of Time_SE by varying the number of tags is shown in figure 4, where the basic TSA and Dy_TSA protocols are compared with their optimal versions (i.e., TSA OPT and Dy_TSA OPT), which apply optimal frame sizing, and implement the *unbounded* search of the number $n$ of tags. The TSA *bounded* refers to the basic TSA protocol, in which tag estimation is performed searching for the $n$ value that minimizes the error between the observed values on frame outcome and the estimated values, varying $n$ in the interval $[c_1+2c_k, 2(c_1+2c_k)]$. The *unbounded* version removes the upper bound $2(c_1 + 2c_k)$ and stops searching when the error is next to 0 (feasible when there are all collisions).

Results show that TSA and Dy_TSA with optimal frame sizing coincide and have better performance than their basic counterparts. The reason behind their similar behavior is the joint effect of optimal frame sizing and the unbounded search of the number of tags. This greatly reduces the number of collisions, obviating the improvements provided of Dy_TSA which cope with inaccurate tag estimation in situations in which many collisions occur in a single frame. The time system efficiency of the optimal TSA and Dy_TSA improves over the basic protocol versions, achieving 80% for very small networks (i.e., 100-200 tags), and decreasing to 65% for larger networks (around 1000 tags). Then it slightly increases again, due to a more effective tag estimation in the case of large number of tags (i.e., 5000 tags).

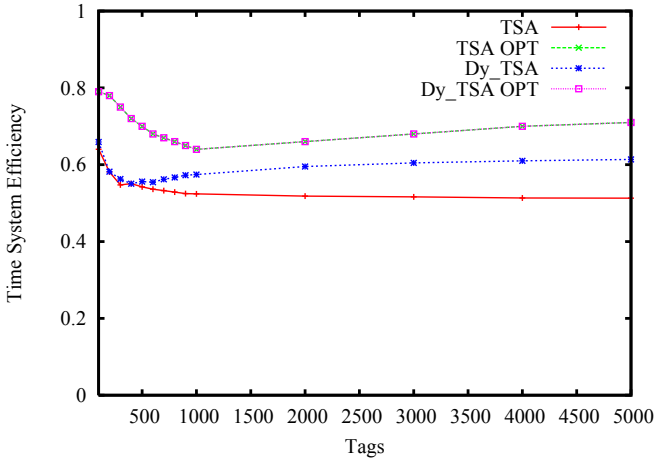
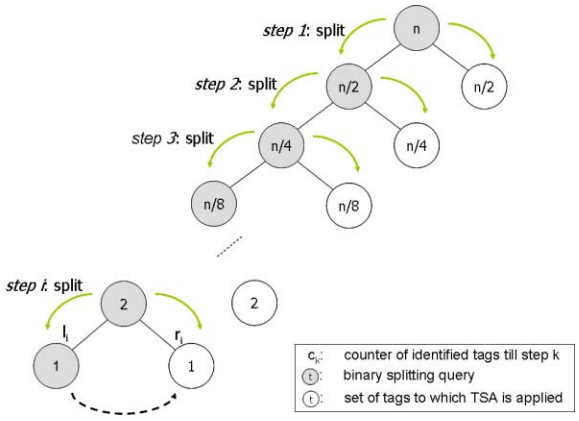



Fig. 4. Time system efficiency for different variations of aloha protocols.

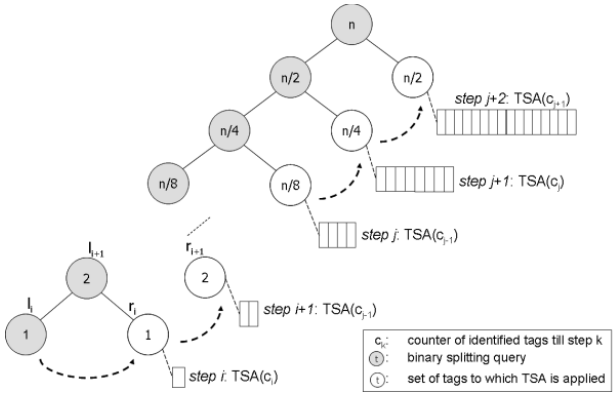
However, the performance of these protocols is lower than expected (i.e., theoretically estimated as $\approx 0.82$). This is because TSA time system efficiency is calculated for the case that the tags population $n$ is known. In practice, the $n$ value is unknown and the initial frame cannot be tuned to the optimal value. A predefined value, such as 128, is considered for the tag set, and optimal frame sizing is applied to that value, issuing an initial frame of $4.406 * 128 - 1 = 562$ slots. This initial frame size allows for good performance in cases of a small number of tags (as long as it provides an overestimation), but penalizes larger networks, making the Time_SE decrease significantly. This aspect motivates our search for an effective and efficient tag estimation method that allows to execute the TSA protocol with knowledge of tag population, achieving almost maximum system efficiency for any network size.

## 5 A NEW PROTOCOL: BSTSA

We propose a new protocol, called Binary Splitting Tree Slotted Aloha (BSTSA), that provides accurate tag estimation and fast tag identification. The protocol combines aspects of the BS and TSA protocols, leveraging the strengths of each. We use the BS approach to accurately estimate the number of tags. We then follow by using the TSA approach to actually identify the tags because of its strength in overcoming collisions.




(a) Binary Splitting phase.




(b) Tree Slotted Aloha phase.

Fig. 5. BSTSA protocol tree.

### 5.1 BSTSA protocol description

The protocol starts by splitting tags into groups through binary splitting (see Figure 5(a)), and proceeds by applying the Tree Slotted Aloha to each group separately (see Figure 5(b)). The binary splitting phase hands over to the TSA protocol when it identifies the first tag.[3] In terms of tree representation, the first tag identification corresponds to visiting the leftmost leaf of the tree (see Figure 5(a)). At this point of protocol execution, tags are split into groups of different sizes (the size decreases as we descend on the tree). In particular, at each query (represented by intermediate nodes) the BS process splits tags into two smaller subsets (whose size is nearly half of the original group), and consequently each node at a given level of the tree contains approximately half of the tags of its parent node. When the splitting process reaches a single-tag group (i.e., the left leaf on the tree), the protocol starts identifying the right siblings on the tree. We start from the right sibling $r_i$ of the leftmost leaf $l_i$. We expect $r_i$ to contain a group of tags whose cardinality is similar to the number of tags found on its left sibling $l_i$. We can use this information to properly size the TSA frame to identify such group of tags (see Figure 5(b)). When we go up one layer (say to layer $i+1$), the sum of the number of tags that have been identified in the two sibling child nodes $r_i$ and $l_i$ gives the number of tags we expect to identify on the right sibling of the parent node (i.e., on $r_{i+1}$). This information is again used to apply TSA with a proper initial frame size to identify the tags of $r_{i+1}$.

As more groups are identified, the group size estimation becomes more accurate: going up on the tree tag groups are bigger and hence the probability that they are divided into two equal parts is higher[4]

3. Please note that the binary splitting is applied only to reach the first single-tag group, and not to identify all tags (i.e., obtaining all single tag-groups) as in the classical BS protocol.

4. Large number of tags (thus large frame sizes) are envisioned in RFID systems. The EPC global standard [8] specifies that a frame length can be as large as 32768 slots.

### 5.2 Analysis of the BSTSA Protocol

To evaluate the BSTSA complexity, we have to consider that the protocol includes two different mechanisms: the splitting process and the tag identification, both of which involve a cost in terms of rounds. Let $R_P[n]$ be the expected number of rounds needed to identify a set of $n$ tags by applying the protocol $P$. Then, $R_{BSTSA}[n]$ can be calculated as in eq. 25.

$$R_{BSTSA}[n] = 1+\sum_{k=0}^{n} P\{B_{n,1/2}(k)\}\times(R_{BSTSA}[k]+R_{TSA}[n-k]) \tag{25}$$

where the left term (i.e., 1) represents the initial splitting and $P\{B_{n,1/2}(k)\} = \binom{n}{k}2^{-k}$ gives the probability of dividing a set of $n$ tags into two groups, one of $k$ tags, and the other one of $n-k$ tags. $R_{TSA}[i]$ is the expected number of rounds taken by the TSA protocol with optimal frame size executed on $i$ tags and is estimated as in eq. 23.

Considering the two phases, we have:

$$\begin{aligned} R_{BSTSA}[n] &= 1+\sum_{k=0}^{n} P\{B_{n,1/2}(k)\} \times R_{BSTSA}[k] \\ &\quad +\sum_{k=0}^{n} P\{B_{n,1/2}(k)\} \times R_{TSA}[n-k] \end{aligned} \tag{26}$$

where the first two terms give the splitting cost while the right most term gives the cost of the several TSA executions. By referring to the TSA cost already evaluated in Section 3 and considering the length of different rounds in the BS and TSA phases we calculate the time system efficiency for the BSTSA protocol as:

$$Time\_SE_{BSTSA} = \frac{R_{ident} * l_{TSA}}{R_{BS} * l_{BS} + R_{TSA} * l_{TSA}} \approx 0.80 \tag{27}$$

where the numerator indicates the time taken by tag identification. This is approximated as if all identifications are performed through the TSA protocol, even if the first tag identification happens during the BS phase.

The denominator shows that we make a distinction between BS rounds and TSA slots, that have different duration. Furthermore, for each protocol we distinguish also between idle and collision rounds. Specifically, in $R_{BS} * l_{BS}$ and $R_{TSA} * l_{TSA}$ we consider that the length of an idle round divided by the length of a collision or identification round gives a factor $0.13$ for the BS protocol, and a factor $0.03$ for the TSA protocol.

The Time_SE of BSTSA protocol (i.e., 0.80) is slighlty lower than the Time_SE of TSA OPT (i.e., 0.82). This is due to the tag estimation method, based on binary splitting, that introduces some cost, but allows the optimal tuning of the initial frame size so that TSA OPT may be applied with knowledge of the number of tags to identify. BSTSA is scalable, as it achieves 0.80 Time_SE for any network size. It also overcomes the need to know the number of tags $n$ a priori to achieve maximum time system efficiency, which is a serious limitation of TSA OPT.

## 6 Performance Evaluation

This section reports the results of a thorough ns2 based comparative performance evaluation among several of the major schemes proposed for single reader–tag communications. We have implemented an RFID extension of the Network Simulator $ns2$ (v. 2.30) [20] accounting for all the unique features of reader-tag communications. We simulated the QT, QTI, BS, TSA, Dy_TSA, and EDFSA anti-collision protocols. We do not show the results for QT and EDFSA because they always perform worse than QTI and TSA, respectively, as shown in previous work [21]. Because TSA and Dy_TSA with optimal frame sizing perform better than their basic versions (see Section 4), we compare BSTSA directly with them.

### 6.1 Simulation setup

To analyze the protocol performance we focus on two metrics: (i) *latency*, defined as the time measured in seconds for identifying all tags, and (ii) *time system efficiency* as defined in Section 3.

We consider an RFID system with a single reader that has a transmission range of 10m. We generate results while varying the number of tags between $n = 100 - 5000$. The channel data rate is 40 Kbps and reader–tag communications occur at a frequency 866 Mhz as specified by the EPCglobal standard [8]. Tag IDs are $k = 96$ bits long which is the most commonly used ID length [8]. The ID values are uniformly distributed. We experiment with varying initial frame sizes as specified below. Results have been obtained by averaging over $150$ runs.

### 6.2 Results

Results on time system efficiency for all protocols are shown in Fig. 6(a) for network sizes ranging from n = 100 - 5000 nodes. BSTSA outperforms all other protocols and achieves nearly the optimal $80\%$ time system efficiency (see Section 5.2). Results for networks between 100 and 500 nodes show around $75\%$ efficiency, due to the cost of the initial splitting. For larger networks (from 500 to 5000 nodes) this cost becomes negligible compared to the identification time and BSTSA achieves $79\%$ efficiency. BSTSA improves on the optimal versions of TSA and Dy_TSA by up to $20\%$, while compared to tree-based protocols, BSTSA is nearly $100\%$ more efficient.

As explained in Section 4, the efficiency of TSA and Dy_TSA coincide because the optimal frame sizing together with unbounded tag estimation greatly reduces the chances of all tags colliding in a single frame. Small networks (100-200 nodes) achieve 80% efficiency because the initial frame size $N$ is fixed to $N = 128 * 4.4 - 1 = 562$, providing the optimal frame size for such networks. As the networks increase in size, the initial frame size becomes an underestimation and time system efficiency decreases to around $65\%$ for 1000 tags. For larger networks, $n > 1000$ nodes, the TSA and Dy_TSA efficiency slightly increases because the error incurred in the initial frame is amortized over the longer identification times.

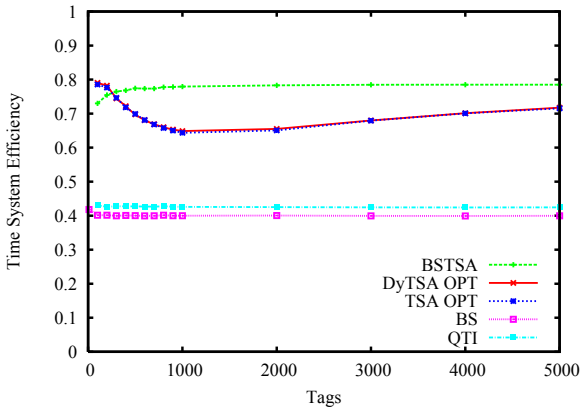
The time system efficiency of BS and QTI remains constant independent of $n$ because of the deterministic operation of tree based-protocols. The BS protocol splits a group of colliding tags into two groups of similar cardinality at each step. The QTI protocol behaves similarly, as the IDs are uniformly distributed and hence the identification tree is similar to the BS tree. QTI is slighlty better than BS because it avoids certain collisions.

The rational behind these trends is shown by results on protocol latencies (see Fig. 6(b)). The bars in the figure show the execution time of each protocol classified into the time spent in collision rounds, idle rounds, and identification rounds.

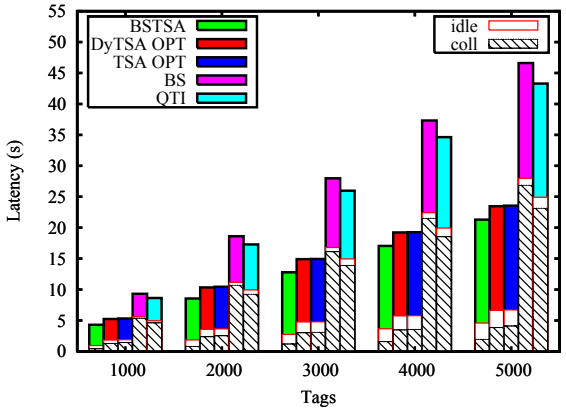
For tree-based protocols, idle rounds have a minor impact on overall protocol execution time, ranging from $2\%$ to $4\%$ of protocol execution time. The negligible percentage of idle time in tree-based protocols is due to the protocol design that causes many more collisions than idle rounds. Eq. 4 and 5 denote the number of idle and collision rounds for BS, which are $0.44 * n$ and $1.44 * n$, respectively. The small number of idle rounds and their short duration explain the very low percentage of idle time, compared to collision and identification time.

On the other hand, BSTSA, TSA, and Dy_TSA show higher idle percentage (up to $15\%$ in the case of 5000 nodes), but significantly reduce the collision percentage (by up to $90\%$ with respect to BS collision percentage in the case of 5000 tags), thanks to the optimal frame sizing. Furthermore, BSTSA outperforms TSA and Dy_TSA (reducing the latency up to $12\%$), as it is able to accurately predict the number of tags being read and sets the correct frame size because of the application of binary tree splitting. This capability allows BSTSA to perform an effective trade-off between idle and collision slots starting with the first frame.

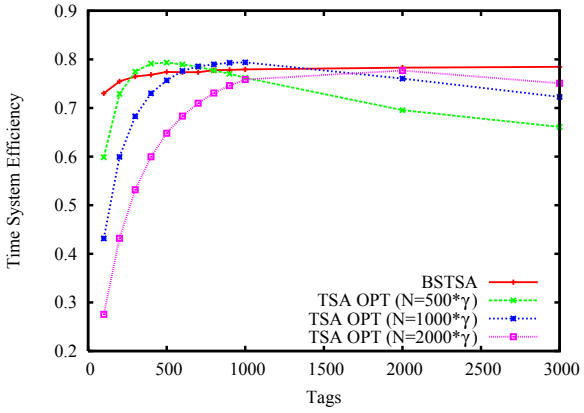
To confirm BSTSA predominance over TSA and




(a) Time_SE comparison versus the number of tags. TSA and Dy_TSA apply optimal frame sizing (the initial frame size, to which overestimation is applied, is set to 128.




(b) Execution time versus the number of tags: 1000-5000 nodes. TSA and Dy_TSA apply optimal frame sizing (the initial frame size, to which overestimation is applied, is set to 128.




(c) Time_SE comparison among BSTSA and TSA by varying the initial frame sizes N = 500, 1000, 2000 (multiplied by the overestimation factor $\gamma = 4.4$).

Fig. 6. Time system efficiency and latency.

Dy_TSA, which are shown to have good performance in their optimal versions, we also investigated their time system efficiency for different initial frame sizes. As they perform similarly we report the results only for optimal TSA. Figure 6(c) shows the time system efficiency of the optimal TSA by varying the initial frame size, to

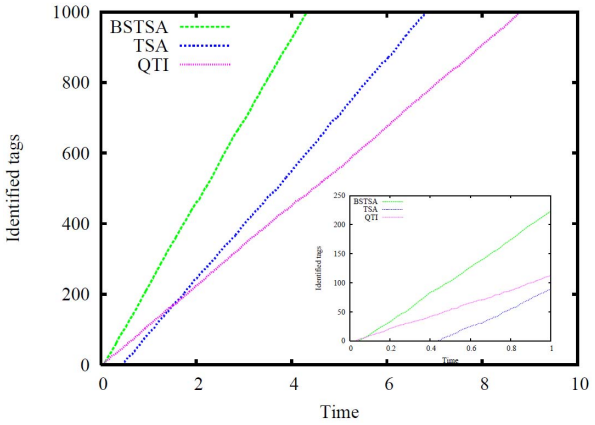



Fig. 7. Tags identification versus time.

which overestimation is applied ($N = n * 4.4 - 1$). Compared with BSTSA, whose efficiency is 79% for any network size except very small networks (100-300 tags), TSA achieves optimal time system efficiency when the initial frame size effectively provides the optimal overestimation (e.g. $N = 500 * 4.4$ in the case of 500 tags). However, outside an interval around the optimal point (where the time system efficiency reaches 80%), the error introduced by the use of a fixed initial frame size causes the time system efficiency to decrease. When the number of actual tags is lower than the basis of the initial frame size, (e.g., 100-300 tags in case of $N = 500 * 4.4 - 1$) the overestimation produces too many idle slots. When the number of actual tags is larger than the basis of the initial frame size (e.g., 1000-5000 tags in case of $N = 500 * 4.4 - 1$), the initial frame results in an underestimation, consequently increasing the number of collisions, that in turn reduces the time system efficiency.

The BSTSA protocol improvement over TSA OPT and Dy_TSA OPT is not high because BSTSA executes a TSA OPT over tag groups. However, BSTSA has the additional feature of efficiently estimating the cardinality of tag groups before starting TSA OPT, so that optimal frame tuning is applied since the first frame. This aspect highlights an important characteristic of BSTSA – its robustness. While TSA and Dy_TSA may perform well for networks of specific sizes, or when the number of tags in the system is known a priori, BSTSA performs well across the full range of network sizes even when no information about the number of tags to be read is known in advance. This makes it a powerful protocol in environments in which the tag density may vary or is unknown.

Figure 7 show how the protocols perform with respect to tags read versus time when 1,000 tags are to be read. A snapshot of the first second shows that BSTSA has read over 20% of the tags after 1 second, while TSA and QTI have read about 10%. After a little more than 4 seconds BSTSA has read all 1,000 tags, while TSA and QTI have read approximately 60% and 50%, respectively. This may have a large impact in systems in which

readers are active for a limited time (for example, when moving on a mobile platform that is carrying inventory into a warehouse), and the system is attempting to perform inventorying functions during normal operations. These results show that much more information may be learned using BSTSA in a small fixed amount of time than with the other protocols.

## 7 Conclusions

In this paper we addressed collision resolution protocols for RFID systems. We defined a new metric for evaluating such protocols, the time system efficiency, which provides a direct measure of the latency incurred by such systems to read a group of tags. Based on this metric we evaluated the performance of existing proposed protocols and derived new optimal frame sizes for specific protocols. Based on the insights gained, we designed a new collision resolution protocol, BSTSA, that combines the strengths of both tree-based and aloha-based protocols. BSTSA not only achieves higher time system efficiency than the other protocols, but is robust to networks ranging from small sizes to very large sizes without a priori knowledge of tag population.

## References


[1] D. W. ENGELS, "The Reader Collision Problem," *AUTO-ID CENTER*, 2001.

[2] D. W. Engels and S. E. Sarma, "The reader collision problem," in *IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 2002.

[3] S. Zhou, Z. Luo, E. Wong, C. Tan, and J. Luo, "Interconnected RFID Reader Collision Model and its Application in Reader Anticollision," in *IEEE International Conference on RFID, 2007*, Mar. 2007.

[4] D.-H. Shih, P.-L. Sun, D. C. Yen, and S.-M. Huang, "Taxonomy and survey of RFID anti-collision protocols," *Elsevier Computer Communications*, vol. 29, pp. 2150–2166, 2006.

[5] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in rfid systems," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2006, pp. 322–333.

[6] N. Abramson, "The aloha system-another alternative for computer communications," in *Proceedings of Full Joint Comp. Conf. AFIPS*, vol. 37, 1970.

[7] J. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Transactions on Information Theory*, vol. 25, no. 5, pp. 505 – 515, 1979.

[8] "EPC Radio-Frequency Identification Protocols Class-1 Gen-2 UHF RFID Protocol for Communications at 860MHz-960MHz, EPCglobal," http://www.epcglobalinc.org/standards/uhfc1g2, Dec. 2004.

[9] J. Massey, "Collision resolution algorithms and random-access communication," *Multi-Users Communication Networks, CISM Courses and Lectures*, no. 256, pp. 73–137, 1981.

[10] J. Myung and W. Lee, "Adaptive binary splitting: a rfid tag collision arbitration protocol for tag identification," *Mob. Netw. Appl.*, vol. 11, no. 5, pp. 711–722, 2006.

[11] C. Law, K. Lee, and K.-Y. Siu, "Efficient memoryless protocol for tag identification (extended abstract)," in *ACM DIALM '00*, New York, NY, USA, 2000, pp. 75–84.

[12] F. B. M. and Cesana, "Arpa: An arbitration protocol based on advanced channel feedback for radio frequency identification," in *WIMOB '08: Proceedings of the 2008 IEEE International Conference on Wireless & Mobile Computing, Networking & Communication*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 178–183.

[13] M. Bonuccelli, F. Lonetti, and F. Martelli, "Instant collision resolution for tag identification in RFID networks," *Elsevier, Ad Hoc Networks*, no. 5, pp. 1220–1232, 2007.

[14] S.-R. Lee, S.-D. Joo, and C.-W. Lee, "An enhanced dynamic framed slotted aloha algorithm for rfid tag identification," in *IEEE MOBIQUITOUS '05*, Washington, DC, USA, 2005, pp. 166–174.

[15] D. Klair, K.-W. Chin, and R. Raad, "On the accuracy of rfid tag estimation functions," in *International Symposium on Communications and Information Technologies, (ISCIT '07)*, Sydney, Australia, 2007, pp. 1401 – 1406.

[16] H. Vogt, "Efficient object identification with passive rfid tags," in *Pervasive '02*. London, UK: Springer-Verlag, 2002, pp. 98–113.

[17] J.-R. Cha and J.-H. Kim, "Dynamic framed slotted aloha algorithms using fast tag estimation method for rfid system," in *IEEE CCNC 2006* vol. 2, Jan. 2006, pp. 768–772.

[18] L. Devroye, "The height and size of random hash trees and random pebbled hash trees," *SIAM J. Comput.*, vol. 28, no. 4, pp. 1215–1224, 1999.

[19] G. Maselli, C. Petrioli, and C. Vicari, "Dynamic tag estimation for optimizing tree slotted aloha in rfid networks," in *MSWiM '08: Proceedings of the 11th symposium on Modeling, analysis and simulation of wireless and mobile systems*. New York, NY, USA: ACM, 2008, pp. 315–322.

[20] "The Network Simulator - ns-2," http://www.isi.edu/nsnam/ns/.

[21] G. Bagnato, G. Maselli, C. Petrioli, and C. Vicari, "Performance analysis of anti-collision protocols for rfid systems," in *IEEE 69th Vehicular Technology Conference, 2009. VTC Spring 2009.*, April 2009, pp. 1–5.


**Thomas F. La Porta** Thomas F. La Porta is a Distinguished Professor in the computer science and engineering department at Penn State. He is the Director of the Networking and Security Research Center at Penn State. He received his B.S.E.E. and M.S.E.E. degrees from The Cooper Union, New York, NY, and his Ph.D. degree in Electrical Engineering from Columbia University, New York, NY. Prior to joining Penn State, he was the Director of the Mobile Networking Research Department in Bell Laboratories where he led various projects in wireless and mobile networking. He is a Bell Labs Fellow. Dr. La Porta was the founding Editor-in-Chief of the IEEE Transactions on Mobile Computing and is currently the Chair of its Steering Committee. He served on the Board of Governors of the IEEE Communications Society. He is co-inventor on over 30 patents.

**Gaia Maselli** Gaia Maselli is a Postdoc Researcher at the computer science department, University of Rome "La Sapienza" . She received her M.S. and Ph.D. in Computer Science from University of Pisa, Italy. Prior to joining "La Sapienza", she worked with the IIT Institute of the Italian National Research Council (CNR). Her research interests include MAC protocols for RFID networks and networking aspects of mobile ad hoc and sensor networks.

**Chiara Petrioli** Chiara Petrioli is Associate Professor in the department of Computer Science of the University of Roma "La Sapienza." She is director of SENSES, a lab of the Technology incubator of "La Sapienza." She holds a Ph.D. in computer engineering from the same university (1998). Her current work focuses on WSNs, RFID systems, cognitive networks, green wireless networking and mobile communications. In these areas she contributed over eighty papers published in international journals and conferences. Dr. Petrioli serves in the steering committee of IEEE Trans. on Mobile Computing. She has been a member of the ACM SIGMOBILE Executive Committee (2005–2009). She is member of the steering committee of ACM Sensys and of the ICST Mobiquitous. She is also member of the the Editorial Board of IEEE Trans. on Mobile Computing, of ACM/Springer Wireless Networks, of Wiley's WCMC and of Elsevier's Ad Hoc Networks. Dr. Petrioli is a Fulbright scholar, a senior member of IEEE and a member of ACM.