Asynchronous Corona Training Protocols in Wireless Sensor and Actor Networks

F. Barsi[*] A.A. Bertossi[†] F. Betti Sorbelli[*] R. Ciotti[*] S. Olariu[‡]
M.C. Pinotti[*]

Abstract

Scalable energy-efficient training protocols are proposed for wireless networks consisting of sensors and a single actor, where the sensors are initially anonymous and unaware of their location. The protocols are based on an intuitive coordinate system imposed onto the deployment area which partitions the sensors into clusters. The protocols are asynchronous, in the sense that the sensors wake up for the first time at random, then alternate between sleep and awake periods both of fixed length, and no explicit synchronization is performed between them and the actor. Theoretical properties are stated under which the training of all the sensors is possible. Moreover, both a worst-case and an average case analysis of the performance, as well as an experimental evaluation, are presented showing that the protocols are lightweight and flexible.

Keywords: Wireless sensor networks, actors, corona training, localization, network protocols, design and analysis of algorithms

1 Introduction

Recent technological breakthroughs in ultra-high integration and low-power electronics have enabled the development of miniaturized battery-operated sensor nodes (*sensors*, for short) that integrate signal processing and wireless communications capabilities [2, 31]. Together with innovative and focused network design techniques that will make possible massive deployment [29] and sustained low power operation, the small size and cost of individual sensors are a key enabling factor for a large number of applications. Indeed, aggregating sensors into sophisticated computational and communication infrastructures, called *wireless sensor networks*, has a significant impact on a wide array of applications ranging from smart kindergarten [17, 23], to smart learning environments [5, 10, 19], to habitat monitoring [18, 27], to environment monitoring [13, 26], to greenhouse and vineyard experiments [6], to forest fire detection [7], to helping the elderly and the disabled [13, 24], among others.

Recently, it has been recognized that it would be beneficial to augment sensor networks by more powerful entities. This leads to a heterogeneous deployment including, alongside with the

[*]Department of Computer Science and Mathematics, University of Perugia, 06123 Perugia, Italy, {barsi,pinotti}@unipg.it
[†]Department of Computer Science, University of Bologna, Mura Anteo Zamboni 7, 40127 Bologna, Italy, bertossi@cs.unibo.it
[‡]Department of Computer Science, Old Dominion University, Norfolk, VA 23529-0162, USA, olariu@cs.odu.edu

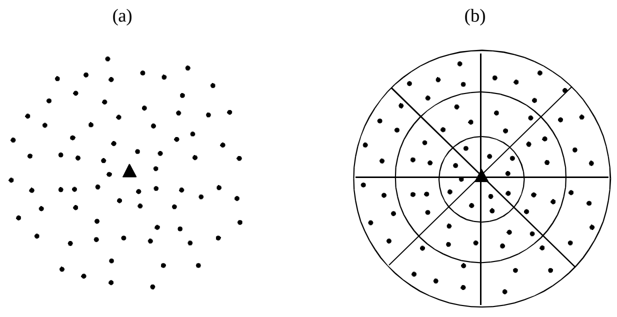

Figure 1: (a) *A sensor network with a central actor.* (b) *The trained sensor network.*

tiny sensors, some entities referred to as *actors* [1] or Aggregation and Forwarding Nodes (AFN) [16]. While the sensors are tasked mainly to sense their immediate neighbourhood, the actors collect, aggregate and fuse the data harvested by the sensors in order to act on the environment in a meaningful way. The resulting augmented version of sensor networks is commonly referred to as sensor and actor networks (SANETs). As pointed out by Olariu *et al.* [16], the typical mode of operation of an actor is to task the sensors in a disk of radius ρ centered at itself to produce data relevant to the mission at hand. Once these data have been aggregated, the actor has a good idea of what action to take. For instance, Figure 1(a) illustrates a disk around an actor. In this scenario, the actor is equipped with a special long-range radio and has a full range of computational capabilities, can send long-range directional broadcasts to the sensors at distance at most ρ, can receive messages from nearby sensors, and has a steady power supply.

The random deployment results in sensors initially unaware of their location and of the network topology. There are some applications requiring sensory data with exact geographical location, motivating the development of communication protocols that are location aware and perhaps location dependent. In some other applications, however, exact geographic location is not necessary, and all that the individual sensors need is only coarse-grain location awareness [17, 28]. One notable application is that of *clustering*, where the set of sensors deployed in an area is partitioned into clusters [2, 3, 8, 23], each corresponding to a small region of indistinguishable sensors. Of course, there is a trade-off, because coarse-grain location awareness is lightweight but the resulting accuracy is only a rough approximation of the exact geographic location.

The task of determining an exact geographic location is referred to as *localization* and has been extensively studied in the literature (see e.g. [11, 21] for surveys). The immediate approach to provide the exact geographic position to sensors is obviously based on GPS. Such an approach, however, is unsuitable for low-cost and small-sized sensors because GPS requires an extensive infrastructure (i.e. satellites). To reduce the infrastructure complexity and the sensor dependence on special hardware, prominent solutions assume the existence of several anchor nodes, that are aware of their location because they are the only GPS-equipped. Most solutions are distributed, that is they do not require centralized computation, and rely on each sensor determining its location with only limited communication with nearby sensors [15, 20, 22]. In general, a distributed protocol may follow up to three phases for determining the individual sensor positions [11]. First, the distances between sensors and anchor nodes are determined by flooding information into the network starting from the anchor nodes (e.g., by counting the number of hops). Then, when each sensor has located enough anchors in its neighborhood, it derives its position from the distances and the positions of its neighbour anchors (e.g., by applying multilateration or multiangulation techniques). Finally, the sensor position can be further refined by using information about the range to, and the position of, neighboring sensors. The main disadvantage of such distributed protocols is the fact that may incur in too much communication overhead and in a large energy consumption due to the lack of central control.

Instead, the task of acquiring a coarse-grain location is referred to as *training*. Such a task has been considered in several recent papers [4, 17, 28, 30] and it is also the topic of the present paper. The main characteristic of the training protocols studied so far relies on using a single actor node, which imposes a coordinate system on the area it covers. The process is centralized because it uses only asymmetric broadcasts (from the actor to the sensors) without multihop communications among the sensors. On the other hand the sensors, which act with the intent of being localized, cooperate by using the received information to deduce their coarse-grain location. Summarizing, using the taxonomy in [21], such training approach has cooperative targets and active infrastructure. Such an approach is more efficient and more effective than distributed localization because, being centralized, it allows the protocol design to be optimized for both the actor and the sensors. Moreover, with respect to the 3-phase approach outlined in [11], training corresponds to the first phase, which computes the distances from the anchor to the sensors, but it is performed in a fully centralized manner instead of being completely distributed. Note that, by asymmetric broadcasts, sensors learn unidirectional distances from the actor to them. Such distances do not necessarily coincide with the reverse multihop distances from them to the actor, which instead depend on the network connectivity.

The main contribution of this paper is to further study the task of training. Contrary to the synchronous model used in [4, 17, 28], an asynchronous model is assumed here as that defined in [30]. Such a model assumes that there is a single fixed actor and that the sensors are asynchronous in the sense that they wake up for the first time at random and then alternate between sleep and awake periods both of fixed length, while no explicit synchronization is performed between them and the actor. The present paper completes the work of [30] presenting three new *flat* protocols which are based on linear signal strength decrease. Moreover, three additional new protocols are exhibited which rely on a *two-level* approach, where jumps are initially made in the flat protocols and are filled later on. Novel theoretical properties on the parameters of the training protocols are stated under which the training of all the sensors in the network is possible. Moreover, a performance evaluation of the protocols is presented showing that they are lightweight in terms of both the number of wake/sleep transitions and the overall sensor awake time for training.

The remainder of this paper is organized as follows. Section 2 discusses the wireless sensor and actor network model and introduces the task of training. Training imposes a coordinate system which divides the sensor network area into equiangular wedges and concentric coronas centered at the actor, as first suggested in [28]. Section 3 offers a quick refresher of basic number theory. Section 4 is the backbone of the entire paper, presenting the theoretical underpinnings of a basic training protocol, called Flat–, along with its worst-case and average-case performance analysis. Section 5 shows two variants of the basic protocol, called Flat and Flat+, as well as a two-level approach, which improve the Flat– performance. In particular, it is shown that the

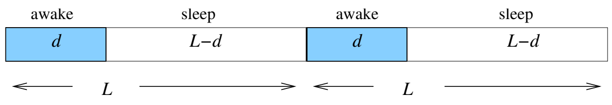

Figure 2: *The sensor sleep-awake cycle.*

two-level protocols outperform the flat ones, lowering the number of transitions from a linear down to, at most, a square-root function. Section 6 presents an experimental evaluation of the performance, tested on randomly generated instances, confirming the analytical results in both the worst and average cases, and showing a much better behaviour in practice. Finally, Section 7 offers concluding remarks.

2 The network model

In this work, a wireless sensor and actor network is assumed that consists of a single, fixed actor and a set of sensors randomly deployed in its broadcast range as illustrated in Figure 1(a). For simplicity, the actor is centrally placed, although this is not really necessary.

It is assumed that the time is ruled into slots. The sensors and the actor use equally long, in phase slots, but they do not necessarily start counting the time from the same slot.

A sensor is a device that possesses three basic capabilities: sensory, computation, and wireless communication, and operates subject to the following fundamental constraints:

a. Each sensor alternates between *sleep* periods and *awake* periods – the sensor sleep-awake cycle is of total length L time slots, out of which the sensor is in sleep mode for $L - d$ slots and in awake mode for d slots;

b. Each sensor is *asynchronous* – it wakes up for the first time according to its internal clock and is not engaging in an explicit synchronization protocol with either the actor or the other sensors;

c. Individual sensors must work *unattended* – once deployed it is either infeasible or impractical to devote attention to individual sensors;

d. Each sensor has no global information about the network topology, but can hear transmissions from the actor;

e. Sensors are *anonymous* – to assume the simplest sensor model, sensors do not need individually unique IDs;

f. Each sensor has a modest non-renewable energy budget and a limited transmission range.

As a result of training, a coordinate system is imposed onto the sensor network which involves establishing [28]:

1. *Coronas*: The deployment area is covered by k coronas $C_0, C_1, \ldots, C_{k-1}$ determined by k concentric circles, centered at the actor, whose radii are $0 < r_0 < r_1 < \cdots < r_{k-1} = \rho$;

2. *Wedges*: The deployment area is ruled into a number of equiangular wedges, centered at the actor, which are established by directional transmission [17].

For the sake of simplicity, in this paper, it is assumed that all coronas have the same width, although this is not strictly required. In a practical setting, the corona width might be equal to the sensor transmission range, say r, and hence the (outer) radius r_i of corona C_i might be equal to $(i+1)r$. In such a case, then, the corona number plus one gives the number of hops needed for a sensor-to-actor communication. Moreover, the length L of the sensor sleep-awake cycle is assumed to be no smaller than the number k of coronas. As illustrated in Figure 1(b), at the end of the training period each sensor has acquired two coordinates: the identity of the corona in which it lies, as well as the identity of the wedge to which it belongs. In particular, a cluster is the locus of all nodes having the same coordinates in the coordinate systems [17].

3 A refresher of basic number theory

Since several derivations in this paper employ number theoretic concepts, it is appropriate to offer the reader a quick refresher of the terminology and basic results used hereafter.

It is well known that for any two integers x and m, with $m \neq 0$, there exist two unique integers q and r such that

$$x = qm + r, \qquad 0 \leq r < |m| \tag{1}$$

where q (the integer quotient) is $\lfloor \frac{x}{m} \rfloor$ i.e., the largest integer smaller than or equal to $\frac{x}{m}$ and r is the nonnegative *remainder* of the division of x by m. Denoting the remainder r with $x \bmod m$, or equivalently $|x|_m$ as suggested in [25] for the sake of conciseness, Equation 1 can be rewritten as

$$x = \left\lfloor \frac{x}{m} \right\rfloor m + |x|_m \tag{2}$$

Two integers x and y are said to be *congruent modulo* m, with $m \neq 0$, and denoted by $x \equiv y \bmod m$, if and only if they have the same nonnegative remainder upon division by m, that is:

$$x \equiv y \bmod m \overset{\text{def}}{=} |x|_m = |y|_m \tag{3}$$

It is easy to see that two integers which differ by a multiple of m are congruent modulo m.

Let $\bullet$ indicate one of the three basic operations, *addition*, *subtraction*, and *multiplication*. With respect to the basic operations, the congruence relation has the following properties similar to those of equality.

Property 3.1. [9] *Given any integers x, y, z, w and $m \neq 0$, it holds:*

1. $x \equiv x \bmod m$
2. *If* $x \equiv y \bmod m$*, then* $y \equiv x \bmod m$
3. *If* $x \equiv y \bmod m$ *and* $z \equiv w \bmod m$*, then* $x \bullet z \equiv y \bullet w \bmod m$

The modulo operation distributes over such operations and hence the following property holds.

Property 3.2. [9] *Let x, y, and m be integers, with $m \neq 0$. Then,*

$$|x \bullet y|_m = ||x|_m \bullet y|_m = |x \bullet |y|_m|_m = ||x|_m \bullet |y|_m|_m$$

Property 3.2 can be easily extended, by induction, to a finite number of integers. Moreover, it is easy to prove that:

Property 3.3. *For any integers a, x, and m, with $a \neq 0$ and $m \neq 0$, it holds:*

$$|ax|_{am} = a|x|_m$$

Proof. By applying Equation 2, one has:

$$|ax|_{am} = ax - \left\lfloor \frac{ax}{am} \right\rfloor am = a\left(x - m\left\lfloor \frac{x}{m} \right\rfloor\right) = a|x|_m$$

□

Recall that the *greatest common divisor* of integers x and y is often denoted by $gcd(x, y)$, or simply (x, y) when no confusion is possible. Therefore, letting $x = x'(x, m)$, and $m = m'(x, m)$ and applying Property 3.3, one derives

$$|x|_m = (x, m)|x'|_{m'}$$

It is worthy to remind that the division of x by y modulo m is possible only when y and m are *coprime*, i.e. when $(m, y) = 1$. Indeed only in such a case there exists the *inverse multiplicative of y modulo m*, which is denoted by $\left|\frac{1}{x}\right|_m$ (as used in [25]) and is defined as that integer satisfying $x\left|\frac{1}{x}\right|_m \equiv 1 \bmod m$. The following properties, which are widely used in the paper, pertain to congruence and division.

Property 3.4. *Given any integers x, y, z, w, and $m \neq 0$, it holds:*

1. *If $ax \equiv ay \bmod m$ and $a \neq 0$ is such that $(a, m) = 1$, then $x \equiv y \bmod m$*
2. *If $ax \equiv ay \bmod m$ and $(a, m) = g$, then $x \equiv y \bmod m'$, where $m = m'g$*

Proof. When $(a, m) = 1$, by Property 3.1,

$$ax \equiv ay \bmod m \rightarrow ax\left|\frac{1}{a}\right|_m \equiv ay\left|\frac{1}{a}\right|_m \bmod m \rightarrow x \equiv y \bmod m.$$

In general, when $(a, m) = g$,

$$\begin{array}{rcll} ax \equiv ay \bmod m & \rightarrow & |ax|_m = |ay|_m & \text{by Definition 3} \\ & \rightarrow & |a'x|_{m'} = |a'y|_{m'} & \text{by Property 3.3} \\ & & \text{where } a = a'g \text{ and } m = m'g & \\ & \rightarrow & x \equiv y \bmod m' & \text{by Property 3.4.1} \end{array}$$

□

The next two properties show how the values generated by the expression $|ix|_m$ vary when i assumes any integer value. It is worthy to note that, since by Property 3.2 $|ix|_m = ||i|_m x|_m$, this is the same as to show how the values generated by the expression $|ix|_m$ vary when i assumes any integer value in $[0, \ldots, m-1]$.

Property 3.5. *Given two integers x and $m \neq 0$ such that $(x, m) = 1$, the congruence $ix \equiv y \bmod m$ has solution for any value y. Moreover, $|ix|_m$ generates all the values in $[0, 1, \ldots, m-1]$.*

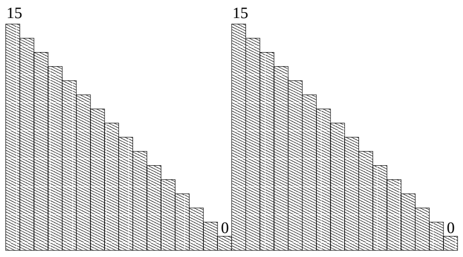

Figure 3: *The actor transmission cycle.*

Proof. Since $(x,m)=1$, for any y, the congruence $ix \equiv y \bmod m$ is verified by $i \equiv y \left|\frac{1}{x}\right|_m \bmod m$. Clearly, for two values of y_1 and y_2 such that $y_1 \equiv y_2 \bmod m$, the congruence has the same solution. Moreover, different values of the expression $|ix|_m$ correspond to different values of $|i|_m$. Indeed, suppose, by contradiction, that there are two integers $i_1 \not\equiv i_2 \bmod m$ such that $i_1 x \equiv i_2 x \bmod m$. Multiplying both sides of the congruence by $\left|\frac{1}{x}\right|_m \bmod m$, one gets $i_1 \equiv i_2 \bmod m$, which leads to a contradiction. Therefore, $|ix|_m$ generates a different value for each $|i|_m$, and hence $|ix|_m$ spans all the values in $[0,\ldots,m-1]$. □

As a particular case, the property above shows that, when i assumes all the m integer values in $[0,\ldots,m-1]$, the expression $|ix|_m$ generates all the m integer values in $[0,\ldots,m-1]$.

On the other hand, when $(x,m)=g$, Property 3.5 can be generalized as follows:

Property 3.6. *Given two integers x and $m \neq 0$ such that $(x,m)=g$, the congruence $ix \equiv y \bmod m$ has solution for any value $y = gy'$ with $y' \in [0,\ldots,m'-1]$, where $m' = \frac{m}{g}$. Moreover, $|ix|_m$ generates only the values multiple of g in $[0,1,\ldots,m-1]$, one for each different value of $|i|_{m'}$.*

Proof. When $(x,m)=g$, $|ix|_m$ is a multiple of g. Indeed, by Property 3.3, $|ix|_m = g|ix'|_{m'}$, where $x = x'g$ and $m = m'g$. Since $(x',m')=1$, by Property 3.5, the congruence $ix' \equiv y' \bmod m'$ has solution $i \equiv y' \left|\frac{1}{x'}\right|_{m'} \bmod m'$ for any value y' and $|ix'|_{m'}$ generates all the values in $[0,\ldots,m'-1]$. Therefore, since for each different value of $|i|_{m'}$ the expression $|ix'|_{m'}$ generates a different value $y' \in [0,\ldots,m'-1]$, the congruence $ix \equiv y \bmod m$ has solution for any value $y = gy'$.

Moreover, the congruence $ix \equiv y \bmod m$ has solution only if y is a multiple of g because both $|ix|_m$ and m are multiple of g. Indeed, the congruence holds if and only if $|ix|_m = |y|_m$, or equivalently by Equation 2, if and only if $g\left(|ix'|_{m'} - \left\lfloor \frac{|ix'|_{m'}}{m'} \right\rfloor m' + \lfloor \frac{y}{m} \rfloor m'\right) g = y$. □

4 The Flat– protocol

The main goal of this section is to present the details of the basic corona training protocol (the wedge training protocol is similar and will not be discussed), where each individual sensor has to learn the identity of the corona to which it belongs, regardless of the moment when it wakes up for the first time.

The idea of the protocol is illustrated in Figure 3. Immediately after deployment the actor cyclically repeats a transmission cycle which involves k broadcasts at successively lower power levels. Each broadcast lasts for a slot and transmits a beacon equal to the identity of the

```
Procedure Actor ($k, \tau_1$);
    for $\tau := 0$ to $\tau_1 - 1$ do
        transmit the beacon $|k-1-\tau|_k$ up to corona $C_{|k-1-\tau|_k}$;
```

Figure 4: *The protocol for the actor.*

outmost corona reached. Precisely, the actor starts out by transmitting the beacon $k-1$ at the highest power, sufficient to reach the sensors up to the outmost corona C_{k-1}; next, the actor transmits the beacon $k-2$ at a power level that can be received up to corona C_{k-2}, but not by the sensors in corona C_{k-1}. For the subsequent $k-2$ slots, the actor continues to transmit at decreasing power levels until it concludes its transmission cycle with a broadcast that can be received only by the sensors in corona C_0. In general, at time slot τ, with $\tau \geq 0$, the actor transmits the beacon $k-1-|\tau|_k$ with a power level that can reach all the sensors up to corona $C_{k-1-|\tau|_k}$, where $|a|_b$ stands for the non negative remainder of the integer division between a and b (i.e. $|a|_b$ is the same as a modulo b). The actor transmission cycle is repeated for a time τ_1 sufficient to accomplish the entire corona training protocol. The protocol for the actor is shown in Figure 4.

In order to describe the protocol for sensors, it is crucial to point out that each sensor is aware of the actor behaviour and of the total number k of coronas. Immediately after deployment, each sensor wakes up at random within the 0-th and the $(k-1)$-th time slot and starts listening to the actor for d time slots (that is, its awake period). Then, the sensor goes back to sleep for $L-d$ time slots (that is, its sleep period). Such a sleep/wake transition will be repeated until the sensor will learn the identity of the corona to which belongs, that is, until the sensor will be trained. Each sensor, during the training process, uses a k-bit register R to keep track of the beacons, i.e. corona identities, transmitted by the actor while the sensor is awake. As soon as the sensor hears an actor transmission for the first time, it starts to fill its register R and it is able to learn the actor global time t within the current actor transmission cycle, that is $t = |\tau|_k$. From now on, such a time will regularly increase so that the sensor can derive from t the beacon $|k-1-t|_k$ that the actor is transmitting. Then, in each time slot when the sensor is awake, one entry of R can be always set either to 0 or to 1. In fact, if the sensor hears beacon c, then it sets $R_c = 1$, while if the sensor hears nothing, it sets $R_{|k-1-t|_k} = 0$. Note that the awake sensors which belong to corona c, with $c > 0$, are able to receive any transmitted beacon from c up to $k-1$, whereas they cannot hear the beacons from 0 up to $c-1$. Hence, if a sensor sets $R_c = 0$ (resp., $R_c = 1$) then it belongs to a corona whose identity is higher than (resp., smaller than or equal to) c. Note that only the sensors in corona 0 can hear beacon 0 and thus they are the only ones which can set $R_0 = 1$. From the above discussion, the following *training condition* holds:

Lemma 4.1. [30] *A sensor which belongs to corona c, with $c > 0$, is trained as soon as the entries R_c and R_{c-1} of its register R are set to* 1 *and* 0*, respectively. A sensor which is in corona* 0 *is trained as soon as R_0 is set to* 1*.*

The resulting sensor protocol, called Flat–, is illustrated in Figure 5. Procedure Flat– mimics the behaviour of the sensor from its first wakeup until it is trained, that is when the identity of the corona to which it belongs is stored into *mycorona*. Each sensor counts the number ν of sleep/wake transitions needed to be trained (line 1), it keeps its local time t, which is initialized when the sensor receives a beacon for the first time from the actor (that is, when *heard* is set

```
    Procedure Flat− $(k, L, d)$;
1       heard := trained := false; $\nu := 0$;
2       while wakeup and ¬ trained do
3           $\nu := \nu + 1$;
4           for $i := 0$ to $d - 1$ do
5               if received beacon $c$ then
6                   if ¬ heard then
7                       heard := true, $t := k - 1 - c$;
8                   $R_c := 1$;
9                   if $c = 0$ or ($R_c = 1$ and $R_{c-1} = 0$) then
10                      mycorona := $c$, trained := true;
11                  $t := t + 1$;
12              else
14                  if heard then
15                      $c := k - 1 - |t|_k$;
16                      $R_c := 0$;
17                      if $R_{c+1} = 1$ then
18                          mycorona := $c$, trained := true;
19                      $t := t + 1$;
20          if heard then
21              alarm-clock := $t := t + L - d$;
22          else
23              alarm-clock := alarm-clock + $L$;
24          go to sleep until the alarm-clock rings;
```

Figure 5: *The Flat− protocol for a sensor.*

to `true` in line 7), and it stores in *alarm-clock* the time when the next sleep/wake transition is planned (line 21–23). After any entry of R is filled, the sensor checks the training condition stated in Lemma 4.1. Observe that lines 12–19 cannot be executed when $c = k-1$, because the beacon $k-1$ reaches the outmost corona C_{k-1}, all awake sensors hear, and thus they execute lines 6–11. In the procedure, each sensor executes $O(1)$ arithmetic/logic operations per time slot.

In the following, some conditions on the parameters k, L, and d will be investigated which guarantee that all the sensors are trained, independent of their first wakeup time and from the corona c they belong to.

Lemma 4.2. *Fixed L, d, and k, there are exactly $k' = \frac{k}{(L,k)}$ different corona identities that can be transmitted by the actor when the sensor starts any awake period. Assuming that the sensor wakes up for the first time at slot x, $0 \leq x \leq k-1$, then the corona identity transmitted when the sensor starts its i-th awake period is $\left|K_x - iL\right|_k = \left|K_x - i(L,k)|L'|_{k'}\right|_k$, where K_x is the corona identity transmitted at time x, that is $K_x = C_{|k-1-x|_k}$. Overall only k' different coronas can be transmitted by the actor when the sensor starts its awake periods, independent of how long the training process will be. Such k' coronas identities can be reindexed as $|K_x - s(L,k)|_k$, for $0 \leq s \leq k'-1$.*

Proof. Consider a sensor that wakes up for the first time at the global time slot $\tau = x$, while the actor is transmitting the beacon $K_x = |k-1-\tau|_k = |k-1-x|_k$. The i-th sleep-awake cycle of such a sensor starts at time $x + iL$ while the actor is transmitting the beacon $|k-1-x-iL|_k$ $= \left|K_x - i|L|_k\right|_k$, with $i \geq 0$. Observe that L and k can be rewritten as $L = gL'$ and $k = gk'$, where $g = (L,k)$. By Property 3.6, $|iL|_k$ generates only the k' multiple of g, one for each different value assumed by $i \bmod k'$, in $[0,\ldots,k]$. Then, by Properties 3.1.1 and 3.1.3, $K_x - iL \equiv K_x - g(|i|_{k'})|L'|_{k'} \bmod k$. In other words, in any two awake periods, say the i-th and the j-th ones, such that $i > j$ and $i - j < k'$, the coronas C_{x+iL} and C_{x+jL} are distinct and differ by a multiple of g. Whereas, in any two awake periods i and j such that $i \equiv j \bmod k'$ the same coronas are transmitted. Clearly, the k' different corona identities transmitted at the beginning of the awake periods can be rearranged so that, in the new order, two consecutive coronas differ exactly by g. Indeed the s-th corona in the new order, that is $|K_x - sg|_k$, with $0 \leq s \leq k'-1$, corresponds to the first beacon transmitted in the j-th awake period, with $j = \left|s|\frac{1}{L'}|_{k'}\right|_{k'}$. □

Therefore, after exactly k' sleep-awake cycles, that is after $k'L$ time slots, which correspond to $\frac{k'L}{k} = \frac{k'L}{gk'} = \frac{L}{g} = L'$ actor transmission cycles, the behaviour of the sensor and the actor will be cyclically repeated. In other words, at the beginning of the k'-th awake period, the sensor and the actor are in the same reciprocal state as they were at the beginning of the 0-th one, with the only difference that, if the sensor can be trained, it has heard the actor at least once. Thus:

Lemma 4.3. *Fixed L, d, and k, all the entries of R that the sensor can fill are set within the first $\frac{2k}{(L,k)}$ sleep-awake cycles.*

Proof. During the first $k' = \frac{k}{(L,k)}$ awake periods of any sensor, the actor transmits no more than $\frac{k}{(L,k)}d$ different corona identities. These corona identities will be cyclically transmitted during the training process of such a sensor. They correspond to exactly all the positions of R that the sensor can set and they include all the beacons that the sensor can hear from the actor. Hence, in the worst case, the sensor needs $k' = \frac{k}{(L,k)}$ awake periods to hear the actor for the first time and further $k' = \frac{k}{(L,k)}$ awake periods to fill R. □

Clearly, if the training condition of Lemma 4.1 cannot be verified by a sensor within its first $2k' = \frac{2k}{(L,k)}$ sleep-awake cycles, such a sensor will never be trained, independent of how long the training process will continue. The following result shows under which conditions for k, L and d all the sensors can be trained and also gives an upper bound on the number of sleep-awake cycles needed to accomplish the entire training process.

Theorem 4.4. *All the sensors are trained in at most $2k' = 2\frac{k}{(L,k)}$ sleep-awake cycles if and only if $d \geq (L,k)$.*

Proof. For brevity let $g = (L,k)$. By contradiction, suppose that all the sensors have been trained and let $d < g$. By Lemmas 4.2 and 4.3, in at most $\frac{2k}{g}$ sleep-awake periods, each sensor has filled at most $k'd$ entries of R. Since $d < g$, each sensor has filled less than k entries of R. Such filled entries depend on the time slot x when the sensor woke up for the first time. Consider now all the sensors that woke up at the same time x. Note that they have filled, although with different configurations, the same positions of R independent of the corona they belong. Let c be one unfilled entry of R. By the hypothesis of massive random deployment, there is at least one sensor that woke up at time x in each corona, and hence at least one sensor in corona c. Clearly, such a sensor will not be trained because the training condition in Lemma 4.1 will be never satisfied.

Conversely, if $d \geq g$, by Lemma 4.2, in k' consecutive sleep-awake cycles, the beacons transmitted by the actor in the first slot of such k' cycles are exactly g apart. Since an awake period lasts $d \geq g$ slots, at least g new corona identities are transmitted by the actor during an awake period of the sensor. Hence, after having heard the actor within the first k' awake periods, the sensor fills at least g entries of R in each awake period and completely fills R in at most other k' awake periods. Therefore, the sensor is trained in at most $2k'$ consecutive awake periods. Note that this happens for all the sensors, independent of their first wake-up time x and of the corona c to which they belong. $\square$

In the following, a better bound on the maximum number of sleep-awake periods required in the worst case to train a sensor is discussed for two particular cases, namely, $d = (L, k)$ and $d = |L|_k$. Note that, since $d = |L|_k = (L, k)|L'|_{k'}$, Theorem 4.4 holds in both cases and hence register R is completely filled within the first $\frac{2k}{(L,k)}$ sleep-awake periods. However, the training condition may be verified earlier because it is sufficient that the entries c and $c-1$ of R be filled. Precisely, Lemmas 4.5 and 4.6 specify, for $d = (L, k)$ and $d = |L|_k$ respectively, in which awake period of a sensor that wakes up for the first time at slot x the actor is transmitting an arbitrary beacon c.

Lemma 4.5. *Let c be any corona identity and assume $d = (L, k)$. The actor transmits the beacon c during the $i_{c,x}$-th awake period of a sensor that wakes up for the first time at slot x, where $i_{c,x} = \left| \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor \left|\frac{1}{L'}\right|_{k'} \right|_{k'}$, $L' = \frac{L}{d}$, and $k' = \frac{k}{d}$.*

Proof. When the sensor wakes up at time x the actor is transmitting the beacon K_x. Moreover, the beacon values decrease within a actor transmission cycle. Thus, the beacon c will be transmitted, starting from K_x, during the j-th group of d consecutive corona identities such that $j = \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor$. Such a j-th group of d consecutive corona identities will be transmitted during the $i_{c,x}$-th sensor awake period in which the actor transmits $\left| K_x - \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor d \right|_k$ as the first beacon. Hence, by Lemma 4.2, $i_{c,x}$ is derived by solving the equation $|K_x - i_{c,x}(L, k)|L'|_{k'}|_k = \left| K_x - \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor d \right|_k$. By Properties 3.1 and 3.4, it follows $i_{c,x} = \left| \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor \left|\frac{1}{L'}\right|_{k'} \right|_{k'}$ because $d = (L, k)$. $\square$

Lemma 4.6. *Let c be any corona identity and assume $d = |L|_k$. The actor transmits the beacon c during the $i_{c,x}$-th awake period of a sensor which wakes up for the first time at slot x, where $i_{c,x} = \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor$.*

Proof. The proof is similar to that of Lemma 4.5. Only observe that now, since $d = |L|_k = (L, k)|L'|_{k'}$ by Property 3.3, $i_{c,x}$ is derived by solving the equation $|K_x - i_{c,x}(L, k)|L'|_{k'}|_k = \left| K_x - \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor d \right|_k$, and hence $i_{c,x} = \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor$. $\square$

The following two lemmas determine when the training condition is satisfied by a sensor.

Lemma 4.7. *Let $d = (L, k)$. A sensor which wakes up for the first time at slot x and belongs to corona c, with $c > 0$, is trained during the i-th awake period where $i = i_{c-1,x}$, if $i_{c,x} \leq i_{c-1,x}$, or $i \leq i_{c,x} + \left|\frac{1}{L'}\right|_{k'}$, if $i_{c,x} > i_{c-1,x}$. If $c = 0$, then $i = i_{0,x}$.*

Proof. If $i_{c,x} \leq i_{c-1,x}$, during the $i_{c,x}$ awake period the sensor hears the beacon c and hence it sets $R_c = 1$. Moreover, during the $i_{c-1,x}$ awake period, the sensor sets $R_{c-1} = 0$ because it does not hear $c-1$ but, having already heard c, it knows what the actor is transmitting. If $i_{c,x} > i_{c-1,x}$, in the worst case the sensor hears for the first time during the $i_{c,x}$-th awake period and sets $R_c = 1$. Then, the beacon $c-1$ will be transmitted at the i-th awake period such that $|K_x - i(L,k)|L'|_{k'}|_k = |K_x - (j+1)d|_k$, where $j = \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor$. Solving the above equation, one has $i = \left| (j+1) \left| \frac{1}{L'} \right|_{k'} \right|_{k'}$, and hence $\left| \frac{1}{L'} \right|_{k'}$ awake periods after $i_{c,x}$. □

Lemma 4.8. *Let $d = |L|_k$. A sensor which wakes up for the first time at slot x and belongs to corona c, with $c > 0$, is trained during the i-th awake period where $i = i_{c-1,x}$, if $i_{c,x} \leq i_{c-1,x}$, or $i \leq i_{c,x} + 1$, if $i_{c,x} > i_{c-1,x}$. If $c = 0$, then $i = i_{0,x}$.*

Proof. The proof is similar to that of Lemma 4.7. For $d = |L|_k$, only observe that, when $i_{c,x} > i_{c-1,x}$, since $d = |L|_k = (L,k)|L'|_{k'}$, i is derived by solving the equation $|i(L,k)|L'|_{k'}|_k = |(j+1)d|_k$, and hence $i = j+1$, where $j = \left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor$. □

In order to analytically evaluate the performance of the Flat− training protocol, let us introduce the following notations. Let ν be the number of sleep/wake transitions required by a sensor to be trained, let ω be the overall sensor awake time, and τ be the total time for training. Recalling that a sleep-awake period has length L, a sensor is awake for d time slots per sleep-awake period, and wakes up at time $x < k$, one has $\omega = \nu d$ and $\tau = \nu L + k$. Moreover, let $\nu_{\max}$ and ν_{avg} denote the number of transitions in the worst and average case, respectively. Analogously, let $\omega_{\max}$ and ω_{avg} denote the worst and average overall sensor awake time. Note that τ is defined only in the worst case because it measures the time required to terminate the whole training process.

Thus, the worst case performance for the Flat− protocol can be summarized as follows:

Theorem 4.9. *Fixed L, d, and k, if $d < (L,k)$ then there are sensors which cannot be trained by the Flat− protocol; otherwise all the sensors are trained, and:*

1. *If $(L,k) \leq d < |L|_k$, then $\nu_{\max} \leq \frac{k}{(L,k)} + \left|\frac{1}{L'}\right|_{k'}$, where $k' = \frac{k}{(L,k)}$ and $L' = \frac{L}{(L,k)}$;*
2. *If $|L|_k \leq d < k$, then $\nu_{\max} \leq \left\lfloor \frac{k}{|L|_k} \right\rfloor + 1$;*
3. *If $d = k$, then $\nu_{\max} = 2$.*

Proof. When $(L,k) \leq d < |L|_k$, since by Lemma 4.2 the k' coronas transmitted by the actor when the sensor wakes up do not depend on d, the sensor cannot be trained later than in the case $d = (L,k)$, because the register R is filled faster. Hence by Lemma 5.3, observing that $\left\lfloor \frac{|K_x - c|_k}{d} \right\rfloor$ varies between 0 and $\frac{k}{d} - 1$, one has $\nu_{\max} \leq \frac{k}{(L,k)} + \left|\frac{1}{L'}\right|_{k'}$, where $k' = \frac{k}{(L,k)}$ and $L' = \frac{L}{(L,k)}$. Similarly, when $|L|_k \leq d < k$, the sensor cannot be trained later than in the case $d = |L|_k$. Hence, by Lemma 4.8, $\nu_{\max} \leq \left\lfloor \frac{k}{|L|_k} \right\rfloor + 1$. Note that, when k is a multiple of $|L|_k$, $\nu_{\max} = \left\lfloor \frac{k}{|L|_k} \right\rfloor + 1$ only for those sensors that wake up for the first time while the actor is transmitting corona $c-1$ and they belong to corona c. Finally, when $d = k$, two sleep-awake cycles are needed only by those sensors which wake up for the first time while the actor is transmitting corona $c-1$ and which belong to corona c. □

Note that, when $d = (L,k)$ or $d = |L|_k$, since $\nu_{\max}$ equals the upper bound stated in Theorem 4.9, $\tau = \left(\frac{k}{(L,k)} + \left|\frac{1}{L'}\right|_{k'}\right) L + k$ or $\tau = \left(\left\lfloor \frac{k}{|L|_k} \right\rfloor + 1\right) L + k$, respectively. Referring to Figure 4, it should be clear that τ_1 must be an upper bound on the total time for training, which is derived as $\tau_1 = \nu_{\max} L + k$ by choosing $\nu_{\max}$ according to the upper bounds given by Theorem 4.9.

Consider now the analysis of the average case performance of the Flat– protocol, where it is assumed that the sensor awake time x is a discrete random variable uniformly distributed in $[0, k-1]$. Let N be the total number of sensors, let N_c be the number of sensors that belong to corona c and, among them, let $N_{c,x}$ be those that wake up for the first time at slot x, with $0 \le c, x \le k-1$. Since x is uniformly distributed, it holds $N_{c,x} = \frac{N_c}{k}$ and, clearly, $\sum_{c=0}^{k-1} N_c = N$.

Theorem 4.10. *Fixed L, d, and k, if $d < (L,k)$ then there are sensors which cannot be trained by the Flat– protocol; otherwise all the sensors are trained, and:*

1. *If $(L,k) \le d < |L|_k$, then $\nu_{\text{avg}} \le \frac{k'+1}{2} + \frac{1}{(L,k)} \left|\frac{1}{L'}\right|_{k'}$, where $k' = \frac{k}{(L,k)}$ and $L' = \frac{L}{(L,k)}$;*
2. *If $|L|_k \le d < k$, then $\nu_{\text{avg}} \le \left(\left\lfloor \frac{k}{|L|_k} \right\rfloor + 1\right)\left(\frac{1}{2} + \frac{|k|_{|L|_k}+1}{k}\right) - \frac{1}{k}$;*
3. *If $d = k$, then $\nu_{\text{avg}} \le 1 + \frac{1}{k}$.*

Proof. In the following, the notation $\ell \le x \le r$ will denote either the values $\ell, \ell+1, \dots, r-1, r$ when $\ell \le r$ or the values $\ell, \ell+1, \dots, k-1, 0, \dots, r-1, r$ when $\ell > r$. Consider first the range $(L,k) \le d < |L|_k$. Assume $d \ge 2$. Fixed a corona c, the sensors that are trained during the 0-th sleep-awake period are those that wake up at $|k-1-(c+d-2)|_k \le x \le k-1-c$. Indeed, such sensors during the d slots of the first awake period hear the beacon c and set $R[c] = 1$ and $R[c-1] = 0$. In contrast, the sensors that wake up at $x = |k-1-(c+d-1)|_k$ can set $R[c]$ to 1, but cannot set $R[c-1]$ to 0 until the $\left|\frac{1}{L'}\right|_{k'}$-th sleep-awake period. In general, the sensors that wake up at $|k-1-(c+d-2+i(L,k)|L'|_{k'})|_k \le x \le |k-1-(c+i((L,k)|L'|_{k'})|_k$ are trained during the i-th sleep-awake period because they set both $R[c]$ and $R[c-1]$. In contrast, the sensors that wake up at $x = |k-1-(c+d-1+i(L,k)|L'|_{k'})|_k$ will set $R[c-1]$ at the $\left(i + \left|\frac{1}{L'}\right|_{k'}\right)$-th sleep-awake period. When $d = (L,k)$, the average number of transitions required to be trained by a sensor in corona c is bounded by

$$\begin{aligned}
\overline{\nu}_c \le\ & \frac{1}{N_c}\left(\sum_{x=|k-1-(c+d-2)|_k}^{k-1-c} N_{c,x} + \left(1 + \left|\frac{1}{L'}\right|_{k'}\right) N_{c,|k-1-(c+d-1)|_k} + \right. \\
& \left. \sum_{i=1}^{k'-1} \sum_{x=|k-1-(c+d-2+i(L,k)|L'|_{k'})|_k}^{|k-1-(c+i(L,k)|L'|_{k'})|_k} (i+1) N_{c,x} + \sum_{i=1}^{k'-1}\left(i+1+\left|\frac{1}{L'}\right|_{k'}\right) N_{c,|k-1-(c+d-1+i(L,k)|L'|_{k'})|_k}\right) \\
=\ & \frac{1}{N_c}\left(\sum_{i=0}^{k'-1} \sum_{x=|k-1-(c+d-1+i(L,k)|L'|_{k'})|_k}^{|k-1-(c+i(L,k)|L'|_{k'})|_k} (i+1) N_{c,x} + \sum_{i=0}^{k'-1} \left|\frac{1}{L'}\right|_{k'} N_{c,|k-1-(c+d-1+i(L,k)|L'|_{k'})|_k}\right) \\
=\ & \frac{1}{N_c}\left(\sum_{i=1}^{k'} i d \frac{N_c}{k} + k' \left|\frac{1}{L'}\right|_{k'} \frac{N_c}{k}\right) \\
=\ & \frac{1}{k}\left(\frac{(L,k)k'(k'+1)}{2} + k'\left|\frac{1}{L'}\right|_{k'}\right) \\
=\ & \frac{k'+1}{2} + \frac{1}{(L,k)}\left|\frac{1}{L'}\right|_{k'}
\end{aligned}$$

If $d = (L,k) = 1$, no sensor in corona c can be trained in a single awake period, independent of its first wake up time x, because it can hear at most one beacon. Thus, each sensor has to wait at least $1 + \left|\frac{1}{\overline{L}}\right|_k$ awake periods in order to set both $R[c]$ and $R[c-1]$. Therefore,

$$\begin{aligned}
\overline{\nu}_c \leq\ & \frac{1}{N_c}\left(\left(1+\left|\frac{1}{\overline{L}}\right|_k\right) N_{c,|k-1-c|_k} + \sum_{i=1}^{k-1}\left(i+1+\left|\frac{1}{\overline{L}}\right|_k\right) N_{c,|k-1-(c+i|L|_k)|_k}\right)\\
=\ & \frac{1}{N_c}\left(\sum_{i=0}^{k-1}\left|\frac{1}{\overline{L}}\right|_k N_{c,|k-1-(c+i|L|_k)|_k} + \sum_{i=0}^{k-1}(i+1)N_{c,|k-1-(c+i|L|_k)|_k}\right)\\
=\ & \frac{1}{N_c}\left(\sum_{i=0}^{k-1}\left|\frac{1}{\overline{L}}\right|_k \frac{N_c}{k} + \sum_{i=0}^{k-1}(i+1)\frac{N_c}{k}\right)\\
=\ & \left|\frac{1}{\overline{L}}\right|_k + \frac{k+1}{2}
\end{aligned}$$

Since $\overline{\nu}_c$ does not depend on c, summing up over all the coronas, one has $\nu_{\text{avg}} = \frac{1}{N}\sum_{c=0}^{k-1}\overline{\nu}_c N_c = \overline{\nu}_c$.

Consider now $1 \leq |L|_k \leq d < k$. Fixed a corona c, as before, the sensors that wake up at $|k-1-(c+d-2)|_k \leq x \leq k-1-c$ are trained during the 0-th sleep-awake period. In contrast, the sensors that wake up at $x = |k-1-(c+d-1)|_k$ are trained during the next sleep-awake period. In general, the sensors that wake up at $|k-1-(c+d-2+i|L|_k)|_k = |k-1-(c+(i+1)d-2)|_k \leq x \leq |k-1-(c+i|L|_k)|_k = |k-1-(c+id)|_k$ are trained during the i-th sleep-awake period, while those waking up at $x = |k-1-(c+(i+1)d-1)|_k$ are trained in the successive sleep-awake period. Therefore, the average number of transitions required to be trained by a sensor in corona c is

$$\begin{aligned}
\overline{\nu}_c =\ & \frac{1}{N_c}\left(\sum_{x=|k-c-d+1|_k}^{k-1-c} N_{c,x} + \sum_{i=1}^{\left\lfloor\frac{k}{|L|_k}\right\rfloor-1}\ \sum_{x=|k-c-(i+1)d+1|_k}^{|k-c-id|_k}(i+1)N_{c,x} + \right.\\
& \left.\sum_{x=|k-c-|k|_{|L|_k}|_k}^{k-c}\left(\left\lfloor\frac{k}{|L|_k}\right\rfloor+1\right)N_{c,x}\right)
\end{aligned}$$

where the last sum is due to the sensors that are trained in the $\lfloor\frac{k}{|L|_k}\rfloor$-th sleep-awake period. Note that in such a sum, when $|k|_{|L|_k} = 0$, only the sensors that belong to corona c and wake up at time slot $k-c$, while the actor is transmitting the beacon $c-1$, have to wait $\frac{k}{|L|_k}+1$ sleep-awake periods to be trained. With simple algebraic manipulations, when $d = |L|_k$, one gets:

$$\begin{aligned}
\overline{\nu}_c \leq\ & \frac{1}{N_c}\left((d-1)\frac{N_c}{k} + \sum_{i=2}^{\left\lfloor\frac{k}{|L|_k}\right\rfloor} id\frac{N_c}{k} + \left(\left\lfloor\frac{k}{|L|_k}\right\rfloor+1\right)\left(|k|_{|L|_k}+1\right)\frac{N_c}{k}\right)\\
=\ & \frac{d-1}{k} + \frac{d}{2k}\left(\left\lfloor\frac{k}{|L|_k}\right\rfloor\left(\left\lfloor\frac{k}{|L|_k}\right\rfloor+1\right)\right) - \frac{d}{k} + \frac{(|k|_{|L|_k}+1)}{k}\left(\left\lfloor\frac{k}{|L|_k}\right\rfloor+1\right)\\
=\ & \left(\left\lfloor\frac{k}{|L|_k}\right\rfloor+1\right)\left(\frac{1}{2}+\frac{|k|_{|L|_k}+1}{k}\right) - \frac{1}{k}
\end{aligned}$$

As before, summing up over all coronas, one has $\nu_{\text{avg}} = \overline{\nu}_c$.

```
6      if ¬ heard then
7          heard := true, $t := k - 1 - c$;
7.1        for $j := i - 1$ downto 0 do $R_{|c+1+j|_k} := 0$;
7.2        for $j := \nu - 1$ downto 1 do
7.3            for $h := d - 1$ downto 0 do $R_{|c+j|L|_k+i-h|_k} := 0$;
```

Figure 6: *The extra instructions for the Flat protocol.*

When $d = k$, only the sensors that belong to corona c and wake up at time slot $x = k - c$ are not trained in a single awake period, but they require one more period. Hence:

$$\nu_{\text{avg}} = 1 + \frac{1}{N}\sum_{c=0}^{k-1} N_{c,c-1} = 1 + \frac{1}{N}\sum_{c=0}^{k-1} \frac{N_c}{k} = 1 + \frac{1}{k}$$

As observed in the proof of Theorem 4.9, when either $(L,k) < d < |L|_k$ or $|L|_k < d < k$, more sensors can be trained in each sleep-awake period, and hence ν_{avg} cannot be greater than that for $d = (L,k)$ and $d = |L|_k$, respectively. □

5 Improvements

The Flat– protocol presented in the previous section can be improved in several ways so as to reduce the number ν of sleep/wake transitions, and hence also the overall sensor awake time as well as the total time for training.

5.1 The Flat protocol

As a first improvement of Flat–, recall that, as soon as a sensor hears the actor transmission for the first time, it learns from the beacon the actor global time modulo the actor transmission cycle. Therefore, it can immediately retrieve backwards the coronas which it did not hear and which were transmitted by the actor during its previous awake periods, setting to 0 the corresponding entries of R. The resulting improved protocol, which is called *Flat*, is derived from the Flat– protocol by modifying, as shown in Figure 6, the `if` instruction in lines 6-7 of Figure 5. As a drawback, a sensor may now execute as many as $O(\nu_{\max} d)$ arithmetic/logic operations per time slot. The time required to perform such arithmetic operations, however, should be negligible with respect to the time slot length, which instead depends on the characteristics of the radio broadcast equipment.

Observed that, when the sensor hears the actor for the first time, it fills R as it would have heard the actor since the first time it woke up, Lemma 4.3 and Theorem 4.4 can be restated as follows:

Lemma 5.1. *Fixed L, d, and k, all the entries of R the sensor can fill are set within the first $k' = \frac{k}{(L,k)}$ sleep-awake cycles.*

Theorem 5.2. *All the sensors are trained in at most $k' = \frac{k}{(L,k)}$ sleep-awake cycles if and only if $d \geq (L,k)$.*

In other words, the Flat protocol completes the training process in at most k' sleep/wake transitions. Such a bound is tight in the particular case that $d = (L,k)$, while it can be lowered when $d = |L|_k$. Indeed, since Lemmas 4.5 and 4.6 still hold, Lemmas 4.7 and 4.8 can be restated for the Flat protocol as follows:

Lemma 5.3. *When $d = (L,k)$ or $d = |L|_k$, a sensor which wakes up for the first time at slot x and belongs to corona c is trained during the i-th awake period where $i = max\{i_{c-1,x}, i_{c,x}\}$, if $c > 0$, or $i = i_{0,x}$, if $c = 0$.*

Proof. When $i_{c,x} \leq i_{c-1,x}$, the proof is the same as that in Lemmas 4.7 and 4.8 for $d = (L,k)$ and $d = |L|_k$, respectively. When $i_{c,x} > i_{c-1,x}$, although in the worst case the sensor hears for the first time during the $i_{c,x}$-th awake period, since R is set backwards, both $R_c = 1$ and $R_{c-1} = 0$ are set during such awake period. □

The worst case performance for the Flat protocol is summarized below:

Theorem 5.4. *Fixed L, d, and k, if $d < (L,k)$ then there are sensors which cannot be trained by the Flat protocol; otherwise all the sensors are trained, and:*

1. *If $(L,k) \leq d < |L|_k$, then $\nu_{\max} \leq \frac{k}{(L,k)}$;*
2. *If $|L|_k \leq d < k$, then $\nu_{\max} \leq \left\lceil \frac{k}{|L|_k} \right\rceil$;*
3. *If $d = k$, then $\nu_{\max} = 1$.*

Proof. When $d = (L,k)$, by Lemmas 4.5 and 5.3, a sensor is trained in at most $\frac{k}{(L,k)}$ sleep/wake transitions. Similarly, when $d = |L|_k$, the result derives from Lemmas 4.6 and 5.3. Moreover, when $(L,k) < d < |L|_k$ and $|L|_k < d < k$, the sensor cannot be trained later than in the case $d = (L,k)$ and $d = |L|_k$, respectively, because by Lemma 4.2 the k' coronas transmitted by the actor when the sensor wakes up are the same (such coronas depend only on L and k) and clearly the register R is filled faster. Finally, when $d = k$ at least one sleep/wake transition is needed. Note that, in both cases $d = k$ and k multiple of $|L|_k$, the sensors that belong to corona c and wake-up when the actor is transmitting corona $c-1$ will set R_{c-1} the first time they hear the actor, without waiting that the actor retransmits beacon $c-1$, saving one transition with respect to Flat–. □

Note that, when $d = (L,k)$ and $d = |L|_k$, $\tau = \frac{kL}{(L,k)} + k$ and $\tau = \left\lceil \frac{k}{|L|_k} \right\rceil L + k$, respectively, because $\nu_{\max}$ matches the upper bound given in Theorem 5.4.

With respect to the average case performance of the Flat protocol, one can prove:

Theorem 5.5. *Fixed L, d, and k, if $d < (L,k)$ then there are sensors which cannot be trained by the Flat protocol; otherwise all the sensors are trained, and:*

1. *If $(L,k) \leq d < |L|_k$, then $\nu_{\text{avg}} \leq \frac{k'+1}{2} + \frac{1}{(L,k)} \left|\frac{1}{L'}\right|_{k'} - \frac{1}{2k} \left|\frac{1}{L'}\right|_{k'} \left(\left|\frac{1}{L'}\right|_{k'} + 1\right)$, where $k' = \frac{k}{(L,k)}$ and $L' = \frac{L}{(L,k)}$;*
2. *If $|L|_k \leq d < k$, then $\nu_{\text{avg}} \leq \left(\left\lfloor \frac{k}{|L|_k} \right\rfloor + 1\right)\left(\frac{1}{2} + \frac{|k|_{|L|_k}+1}{k}\right) - \frac{1}{k}$;*
3. *If $d = k$, then $\nu_{\text{avg}} = 1$.*

Proof. The proof is similar to that of Theorem 4.10. Consider first the range $(L,k) \leq d < |L|_k$ with $d \geq 2$. From Theorem 5.4, $\nu_{\max} \leq k'$ for the Flat protocol. Hence, the sensor behaviour is the same as that for the Flat– protocol, except for those sensors which were trained by Flat– using more than k' transitions. Therefore, the number of transitions remains the same except for the sensors that wake up at slot $x = |k-1-(c+d-1+i(L,k)|L'|_{k'})|_k$, with $k' - \left|\frac{1}{L'}\right|_{k'} \leq i \leq k'-1$, which will now set $R[c-1]$ no later than during the $(k'-1)$-th sleep-awake period. Thus the formula for $\overline{\nu}_c$ becomes

$$\overline{\nu}_c \leq \frac{1}{N_c}\left(\sum_{x=|k-1-(c+d-2)|_k}^{k-1-c} N_{c,x} + \left(1+\left|\frac{1}{L'}\right|_{k'}\right) N_{c,|k-1-(c+d-1)|_k} + \right.$$
$$\sum_{i=1}^{k'-1}\ \sum_{x=|k-1-(c+d-2+i(L,k)|L'|_{k'})|_k}^{|k-1-(c+i(L,k)|L'|_{k'})|_k} (i+1)N_{c,x} +$$
$$\left.\sum_{i=1}^{k'-1-\left|\frac{1}{L'}\right|_{k'}} \left(i+1+\left|\frac{1}{L'}\right|_{k'}\right) N_{c,|k-1-(c+d-1+i(L,k)|L'|_{k'})|_k} + \sum_{i=k'-\left|\frac{1}{L'}\right|_{k'}}^{k'-1} k' N_{c,|k-1-(c+d-1+i(L,k)|L'|_{k'})|_k}\right)$$
$$= \frac{k'+1}{2} + \frac{1}{(L,k)}\left|\frac{1}{L'}\right|_{k'} - \frac{1}{2k}\left|\frac{1}{L'}\right|_{k'}\left(\left|\frac{1}{L'}\right|_{k'}+1\right)$$

If $d = (L,k) = 1$, since no sensor in corona c can be trained in a single awake period and $\nu_{\max} \leq k' = k$ by Theorem 5.4, one has:

$$\overline{\nu}_c \leq \frac{1}{N_c}\left(\left(1+\left|\frac{1}{L}\right|_k\right) N_{c,|k-1-c|_k} + \right.$$
$$\left.\sum_{i=1}^{k-1-\left|\frac{1}{L}\right|_k}\left(i+1+\left|\frac{1}{L}\right|_k\right) N_{c,|k-1-(c+i|L|_k)|_k} + \sum_{i=k-\left|\frac{1}{L}\right|_k}^{k-1} kN_{c,|k-1-(c+i|L|_k)|_k}\right)$$
$$= \frac{1}{N_c}\left(\sum_{i=0}^{k-1-\left|\frac{1}{L}\right|_k}\left(i+1+\left|\frac{1}{L}\right|_k\right)\frac{N_c}{k} + \sum_{i=k-\left|\frac{1}{L}\right|_k}^{k-1} k\frac{N_c}{k}\right)$$
$$= \frac{1}{N_c}\left(\frac{N_c}{k}\sum_{i=1+\left|\frac{1}{L}\right|_k}^{k} i + \left|\frac{1}{L}\right|_k N_c\right)$$
$$= \frac{k+1}{2} + \left|\frac{1}{L}\right|_k - \frac{1}{2k}\left|\frac{1}{L}\right|_k\left(\left|\frac{1}{L}\right|_k+1\right)$$

Summing up over all the coronas, one has $\nu_{\text{avg}} = \frac{1}{N}\sum_{c=0}^{k-1}\overline{\nu}_c N_c = \overline{\nu}_c$.

Consider now $1 \leq |L|_k \leq d < k$. The Flat protocol has the same behaviour as the Flat– one, except in the particular case that $d = |L|_k$ divides k. In such a case, the sensors that belong to corona c and wake up for the first time when the actor transmits the beacon $c-1$ save one transition. Indeed, such sensors set $R[c-1]$ the first time they hear any beacon, thus saving $\frac{1}{k}$ on the average number of transitions. However, in general, the same bound of ν_{avg} stated in Theorem 4.10 still holds.

Finally, when $d = k$, each sensor is trained in a single transition, and hence $\nu_{\text{avg}} = \nu_{\max} = 1$. □

5.2 The Flat+ protocol

A further improvement to the Flat protocol exploits the fact that when a sensor hears a beacon c, it knows that it will also hear all the beacons greater than c, and thus it can immediately set to 1 the entries from R_c up to R_{k-1}. Similarly, when a sensor sets an entry R_c to 0, it knows that it cannot hear any beacon smaller than c, and thus it can immediately set to 0 the entries from R_{c-1} down to R_0, too. In contrast to the previous protocols, the sensor now fills entries of R relative to beacons not yet transmitted during its awake periods. Therefore, it can look ahead to decide whether it is worthy or not to wake up in the next awake period. If the d entries of R that will be transmitted by the actor in the next awake period have already been filled, then the sensor can skip its next awake period, thus saving energy. The sensor repeats the look-ahead process above until at least one unfilled entry is detected among the d entries corresponding to a future awake period. The resulting protocol, called Flat+, is illustrated in Figure 7. Procedure Flat+ makes use of two variables, max0 and min1, which record the largest (smallest, resp.) index of R which has been filled to 0 (1, resp.). When a beacon c is heard, the sensor sets to 1 all the entries from R_c to R_{min1} (line 11). When an entry R_c has to be set to 0, then all the entries from R_{max0} to R_c are set to 0 (line 19). When the sensor hears the actor for the first time, it stores in max0 the largest entry of R which must be 0 due to its previous awake periods (lines 8-9), and thus it sets to 0 the entries from R_0 to R_{max0} (line 10). Finally, at the end of the awake period, the sensor performs the above mentioned look-ahead process, properly setting the alarm-clock (lines 24–29).

The worst case performance of the Flat+ protocol coincides with that of the Flat one, as stated below.

Theorem 5.6. *Fixed L, d, and k, if $d < (L,k)$ then there are sensors which cannot be trained by the Flat+ protocol; otherwise all the sensors are trained, and:*

1. *If $(L,k) \leq d < |L|_k$, then $\nu_{\max} \leq \frac{k}{(L,k)}$;*
2. *If $|L|_k \leq d < k$, then $\nu_{\max} \leq \left\lceil \frac{k}{|L|_k} \right\rceil$;*
3. *If $d = k$, then $\nu_{\max} = 1$.*

Proof. Clearly, the number ν of sleep/wake transitions of Flat+ cannot be larger than that of Flat. In fact, when $d = (L,k)$ or $d = |L|_k$, one can find bad instances, where $\nu_{\max}$ is the same for both Flat+ and Flat. When $d = (L,k)$, consider a sensor that belongs to corona $c = k-1$ and wakes up for the first time at $x = |d|L'|_{k'}|_k$, where, as usual, $k' = \frac{k}{(L,k)}$ and $L' = \frac{L}{(L,k)}$. Indeed, the sensor can hear only the beacon $k-1$ which is, according to Lemma 4.5, actually heard at the $(k'-1)$-th sleep-awake cycle. When $d = |L|_k$, a sensor which belongs to corona c and wakes up at time slot $x = k - c$, that is when the actor transmits $c-1$, has to wait until the actor transmits beacon c, which happens at the $(\lceil \frac{k}{|L|_k} \rceil - 1)$-th sleep-awake cycle. Thus, in both cases, the sensor cannot save transitions by means of its look-ahead process. □

Besides to the worst case performance, also the average case performance of both the Flat and Flat+ protocols coincide when $(L,k) = 1$, for for d ranging in $[|L|_k, k]$ because during subsequent awake periods of the sensors the actor transmits consecutive decreasing beacons, and hence Flat+ cannot take advantage of its look-ahead process. Similarly, when $(L,k) \neq 1$ and d is sufficiently large, both Flat+ and Flat have the same behaviour. Indeed, for $d \geq$

```
Procedure Flat+ $(k, L, d)$;
    heard := trained := false; $\nu := 0$; max0 := 0; min1 := $k-1$;
    while wakeup and $\neg$ trained do
        $\nu := \nu + 1$;
        for $i := 0$ to $d-1$ do
            if received beacon $c$ then
                if $\neg$ heard then
                    heard := true, $t := k - 1 - c$;
                    max0 := $max\{$max0, $|c+i|_k\}$;
                    for $j := \nu - 1$ downto 0 do max0 := $max\{$max0, $|c + j|L|_k + i|_k\}$;
                    for $h := 0$ to max0 do $R_h := 0$;
                for $h := c$ to min1 do $R_h := 1$;
                min1 := $c$;
                if $c = 0$ or $(R_c = 1$ and $R_{c-1} = 0)$ then
                    mycorona := $c$, trained := true;
                $t := t + 1$;
            else
                if heard then
                    $c := k - 1 - |t|_k$;
                    for $h :=$ max0 to $c$ do $R_h := 0$;
                    max0 := $c$;
                    if $R_{c+1} = 1$ then
                        mycorona := $c$, trained := true;
                    $t := t + 1$;
        if heard then
            filled:=true, alarm-clock := $t := t + L - d$;
            while filled do
                $s := |k - 1 - t|_k, z := 0$;
                while $z \leq d - 1$ and $R_{|s-z|_k}$ has been filled do $z := z + 1$;
                if $R_z$ is unfilled then filled := false else alarm-clock := $t := t + L$;
        else
            alarm-clock := alarm-clock + $L$;
        go to sleep until the alarm-clock rings;
```

Figure 7: *The Flat+ protocol for a sensor.*

```
Procedure Actor $(k_1, k_2, \tau_1, \tau_2)$;
    $\tau := 0$;
    for $z := 0$ to $\tau_1 - 1$ do
        transmit the beacon $|k_1 - 1 - z|_{k_1}$ up to corona $C_{k_2(|k_1-1-z|_{k_1}+1)-1}$;
        $\tau := \tau + 1$;
    for $m := 0$ to $k_1 - 1$ do
        for $z := 0$ to $\tau_2 - 1$ do
            transmit the beacon $|k_2 - 1 - z|_{k_2}$ up to corona $C_{mk_2+|k_2-1-z|_{k_2}}$;
            $\tau := \tau + 1$;
```

Figure 8: *The two-level protocol for the actor.*

$\max\{|L|_k + 1, k - |L|_k\}$, both protocols require the same number of transitions, precisely one or two transitions, for each sensor. In fact, for such values of d, all the k beacons are transmitted by the actor during at most two sensor awake periods. So, since Flat+ cannot skip the first awake period, it cannot gain from its look-ahead process.

Although the average case performance of both the Flat and Flat+ protocols coincide for large values of d, or when $(L, k) = 1$ and $d \geq |L|_k$, the practical behaviour of Flat+ is much better than that of Flat for small values of d, as it will be experimentally checked in Section 6.

5.3 The two-level approach

The protocols discussed so far can be further improved by following a nesting approach in which the k coronas are viewed as k_1 macrocoronas of k_2 adjacent coronas each. Precisely, each sensor first learns in which macrocorona it belongs and then refines its training by determining the microcorona inside its macrocorona. Once a sensor learns the index of its macrocorona, say m with $0 \leq m \leq k_1 - 1$, as well as that of its microcorona, say μ with $0 \leq \mu \leq k_2 - 1$, it obtains its actual corona identity as $c = k_2 m + \mu$, where of $0 \leq c \leq k_1 k_2 - 1$. For determining both the macrocorona and microcorona identities, any of the Flat protocol variants can be used.

The protocol for the actor is shown in Figure 8. The actor works in two levels and counts the total time in τ. In the first level, the actor cyclically repeats the *macrocorona transmission cycle*, that is a cycle of length k_1 using decreasing powers so as to distinguish different consecutive macrocoronas. In fact, at time slot $z = 0$, the actor starts out by transmitting the beacon $k_1 - 1$ to a power sufficient to reach the sensors up to the outmost macrocorona, that is up to corona C_{k-1}. Next, the actor transmits the beacon $k_1 - 2$ at a power that can be received up to the $(k_1 - 2)$-th macrocorona, that is corona C_{k-k_2-1}. For the subsequent $k_1 - 2$ slots, the actor continues to transmit at decreasing powers until it concludes its cycle at time slot $z = k_1 - 1$ with a broadcast that can be received only by the sensors in the 0-th macrocorona, that is, up to corona C_{k_2-1}. The first level lasts for τ_1 time slots, thus repeating $\frac{\tau_1}{k_1}$ times the macrocorona transmission cycle. The time τ_1 is properly chosen to allow all the sensors to be trained with respect to their macrocorona.

In the second level, for each macrocorona, the actor cyclically repeats a *microcorona transmission cycle*, that is one of length k_2 using decreasing powers so as to distinguish different consecutive coronas. Such a microcorona transmission cycle is repeated $\frac{\tau_2}{k_2}$ times, choosing τ_2 so as to allow all the sensors in each macrocorona m to be also trained with respect to their microcorona. Overall the second level of the protocol lasts $k_1\tau_2$ time slots.

As regard to the protocol for the sensors, it is assumed that each sensor is aware of the two-level actor behaviour and thus of the numbers k_1 and k_2 of macrocoronas and microcoronas,

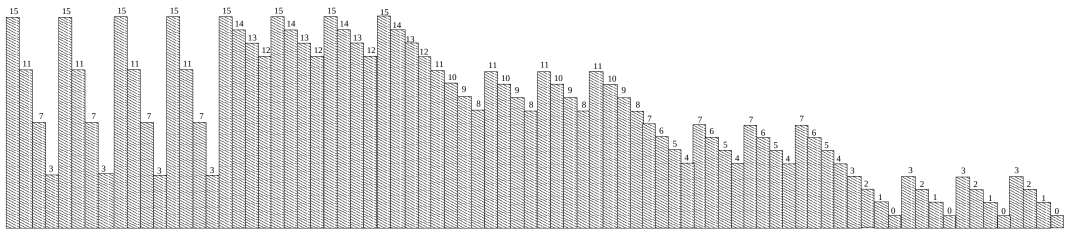

Figure 9: *The actor transmission cycle for the TwoLevel and TwoLevel+ protocols.*

respectively. Each sensor wakes up at time x, with $0 \leq x \leq \min\{k_1, k_2\}$, and repeats its sleep-awake cycle of length L such that $L \geq d \geq \max\{(L, k_1), (L, k_2)\}$. Each sensor uses a k_1-bit register P and a k_2-bit register Q to keep track of the macrocorona and microcorona identities, respectively. As soon as the sensor wakes up at time x, it performs one of the protocol variants, i.e. Flat–, Flat, and Flat+, using its register P to learn its macrocorona identity m. When it has been trained on its macrocorona, it sets its alarm clock to $\tau_1 + (k_1 - 1 - m)\tau_2 + x$ to be ready for the training on its microcorona, and goes to sleep. Reawakened, the sensor performs again the same protocol variant, but now filling its register Q to learn its microcorona identity μ. Clearly, as soon as it knows both m and μ, it derives its corona identity $c = k_2 m + \mu$, and thus it is trained.

Depending on which protocol, Flat–, Flat, and Flat+, is used to train the sensors on each macrocorona and microcorona level, three two-level protocols are achieved, denoted by TwoLevel–, TwoLevel, and TwoLevel+. In Figure 9, the macrocorona and microcorona actor transmission cycles are depicted for the TwoLevel and TwoLevel+ protocols in the case where $k = 16$, $k_1 = k_2 = 4$, $L = 6$, $d = |L|_4 = 2$. In this case, the number of sleep/wake transitions for the two levels, say ν_1 and ν_2, are both $\frac{4}{2}$, and hence $\tau_1 = 2 * 6 + 4$, $\tau_2 = 2 * 6 + 4$, and $\tau = 16 + 4 * 16 = 80$. Note that the actor performs $\frac{\tau_1}{k_1} = 4$ actor transmission cycles of length k_1 to train the sensors on their macrocorona, and additional $\frac{\tau_2}{k_2} = 4$ actor transmission cycles of length k_2 per macrocorona to train them on their microcorona.

In general, with respect to the performance of the two-level protocols, one has:

Theorem 5.7. *Fixed L, d, k, k_1, and k_2, with $k = k_1 k_2$ and $L \geq d \geq max\{(L, k_1), (L, k_2)\}$, letting ν_1 and ν_2 be, respectively, the numbers of sensor sleep/wake transitions required to train a sensor on k_1 macrocoronas and k_2 microcoronas, the two-level protocols require $\nu = \nu_1 + \nu_2$ sleep/wake transitions and $\omega = (\nu_1 + \nu_2)d$ overall sensor awake time. Moreover, the total time for training is $\tau = \tau_1 + \tau_2 k_1$, where τ_1 and τ_2 must be the upper bounds on the total time required by the training protocol adopted on each level.*

Note that, by Theorems 4.9 and 5.4, tight bounds on the values of τ_1 and τ_2 can be derived only when $d = (L, k)$, $d = |L|_k$, and $d = k$. In all other cases, the total time of each level is derived from $\tau = \nu L + k$, setting ν equal to the upper bound given in Theorems 4.9 and 5.4. For example, consider the TwoLevel– protocol and assume $d = |L|_{k_1}$, and $(L, k_2) \leq d < |L|_{k_2}$. Then, $\tau_1 = \left(\left\lfloor \frac{k_1}{|L|_{k_1}} \right\rfloor + 1 \right) L + k_1$ and $\tau_2 = \left(\frac{k_2}{(L,k_2)} + \left| \frac{1}{L'} \right|_{k'} \right) L + k_2$, where $k' = \frac{k_2}{(L,k_2)}$ and $L' = \frac{L}{(L,k_2)}$.

Next, the worst case performance of the Flat protocol is compared with that of the corresponding TwoLevel protocol when the same value of L and d are used, $\frac{k_1}{(L,k_1)} \neq 1$, $\frac{k_2}{(L,k_2)} \neq 1$, and $(L, k_2) \neq 1$. Note that to satisfy the constraints of both the Flat and TwoLevel protocols, d must vary between $(L, k) = (L, k_1)(L, k_2)$ and $\min\{k_1, k_2\}$, and L must be greater

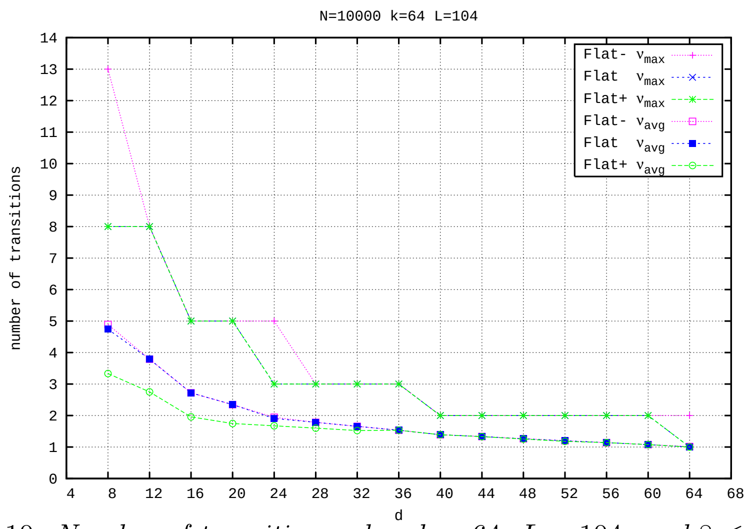

Figure 10: *Number of transitions when* $k = 64$*,* $L = 104$*, and* $8 \leq d \leq 64$*.*

than k. When $d = (L, k)$, by Lemma 5.1 and Theorem 5.4, the number of transitions is at most $\frac{k_1}{(L,k_1)} + \frac{k_2}{(L,k_2)}$ for TwoLevel and is at least $\frac{k}{(L,k)}$ for Flat. Since $(L, k) = (L, k_1)(L, k_2)$, one has $\frac{k_1}{(L,k_1)} + \frac{k_2}{(L,k_2)} < \frac{k_1}{(L,k_1)}\frac{k_2}{(L,k_2)} = \frac{k}{(L,k_1)(L,k_2)} = \frac{k}{d}$. Similarly, TwoLevel beats Flat when $d = \min\{k_1, k_2\}$. Indeed letting $d = k_1 = \min\{k_1, k_2\}$, TwoLevel requires at most $1 + \frac{k_2}{(L,k_2)} < k_2$ sleep/wake transitions, while Flat needs at least $\frac{k}{d} = k_2$ transitions. Since in both protocols the number of transitions decreases when d increases, TwoLevel beats Flat when $(L, k) \leq d \leq \min\{k_1, k_2\}$. Finally it is easy to see that both the overall sensor awake time and the total time for training of Flat are larger than those of TwoLevel.

6 Experimental tests

In this section, the worst and average performance of the corona training protocols are experimentally tested. The algorithms were written in C++ and the experiments were run on an AMD Athlon X2 4800+ with 2 GB RAM. In the simulation, each corona has a unit width. There are $N = 10000$ sensors uniformly distributed within a circle of radius $\rho = k$, centered at the actor and inscribed in a square. Precisely, the Cartesian coordinates of each sensor are uniformly generated choosing at random two real numbers in the range $[-k, k]$. The generation proceeds until N sensors are placed inside the circle, thus discarding those laying outside. In the experiments, both the worst and average number of transitions, denoted by $\nu_{\max}$ and ν_{avg}, as well as both the worst and average overall sensor awake time, $\omega_{\max}$ and ω_{avg}, are evaluated. Such average values are obtained by summing up the values for each single sensor and then dividing by the number of sensors. Moreover, the total time τ, which measures the time required to terminate the whole training process, is evaluated.

Consider first the experiments for the Flat–, Flat, and Flat+ protocols. In the simulations, the number k of coronas is fixed to 64. The length L of the sensor sleep-awake cycle assumes the values 104 and 168. Finally, the sensor awake period d is an integer that varies, with a step of 4, between the greatest common divisor $(L, k) = 8$ and $k = 64$, thus including $|L|_k = 40$. The results are reported only when all the sensors can be trained, that is for $d \geq 8$, and are averaged over 3 independent experiments.

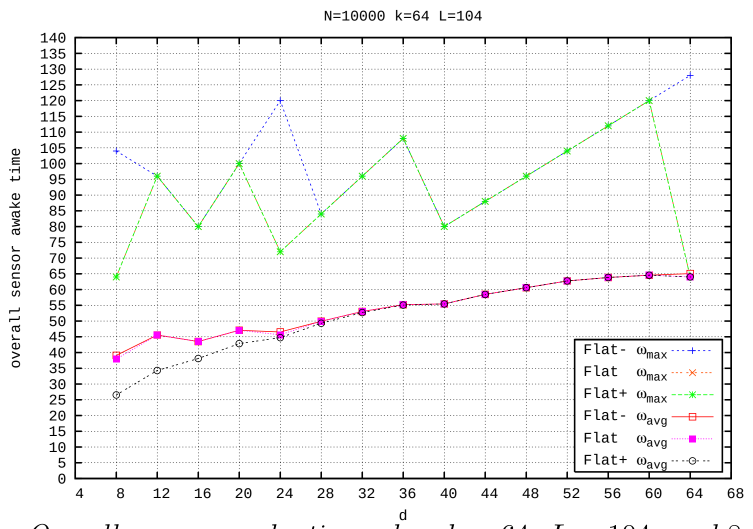

Figure 11: *Overall sensor awake time when* $k = 64$, $L = 104$, *and* $8 \le d \le 64$.

Figure 10 shows the number $\nu_{\max}$ and ν_{avg} of transitions for the different values of d. According to Theorems 4.9 and 5.4, when $d = 8$, Flat– has $\nu_{\max} = \frac{k}{(L,k)} + \left|\frac{1}{L'}\right|_{k'} = 8+5 = 13$, while both Flat and Flat+ have $\nu_{\max} = \frac{k}{(L,k)} = 8$. Similarly, when $d = 40$, all protocols take $\nu_{\max} = 2$ transitions. Except for the extreme values $d = 8$ and $d = 64$, the greatest percentage of gain for $\nu_{\max}$ is achieved when $d = 24$, where both Flat+ and Flat employ forty percent less transitions than Flat–. As regard to the average performance, one notes that ν_{avg} is considerable better than $\nu_{\max}$ for all three protocols. Flat and Flat– have almost the same average performances, while Flat+ always behaves better than them. In particular, its greatest percentage of gain for ν_{avg} is obtained in the range $8 \le d \le 20$, where Flat+ improves about twenty/thirty percent upon Flat–. For $d \ge \max\{|L|_k + 1, k - |L|_k\}$, both the worst and average results of Flat and Flat+ coincide.

Figure 11 shows the awake times $\omega_{\max} = \nu_{\max} d$ and $\omega_{\text{avg}} = \nu_{\text{avg}} d$, which measure the overall energy spent by each sensor to be trained. Although the number of transitions decreases as d increases, Figure 11 suggests to choose a small value of d from the sensor awake time perspective. The minimum $\omega_{\max}$ is achieved by Flat and Flat+ for $d = 8$ and $d = 64$, as expected by Theorems 4.9 and 5.4. However, when $d = 8$, ω_{avg} lowers to about two thirds of $\omega_{\max}$ for Flat– and Flat, and to about one third for Flat+. Note that Flat+ has the maximum gain when d is small. Indeed, it can fill the same entries of R just listening to the actor for a single slot or for d slots. Hence, small values of d save the same number of transitions as larger values, but allow sensors to reduce their energy consumption because they stay awake for smaller periods.

Figure 12 exhibits the total time τ required to accomplish the entire training task, for both $L = 104$ and $L = 168$. Since $|168|_{64} = |104|_{64} = 40$, by Lemma 4.2, each protocol maintains the same behaviour with respect to the number of transitions. Thus, the plots for $L = 168$ of $\nu_{\max}$ and ν_{avg}, and hence of $\omega_{\max}$ and ω_{avg}, are exactly the same as those shown in Figures 10 and 11. Recalling that $\tau = \nu_{\max} L + k$, the total time for $L = 168$ scales by a constant $\sim \frac{168}{104}$, as depicted in Figure 12. In general, all values of L such that $|L|_k$ is the same present the properties above, namely, ν and ω are identical, while τ scales. Therefore, the minimum total time τ is achieved for the smallest value of L. However, larger values of L could be also selected

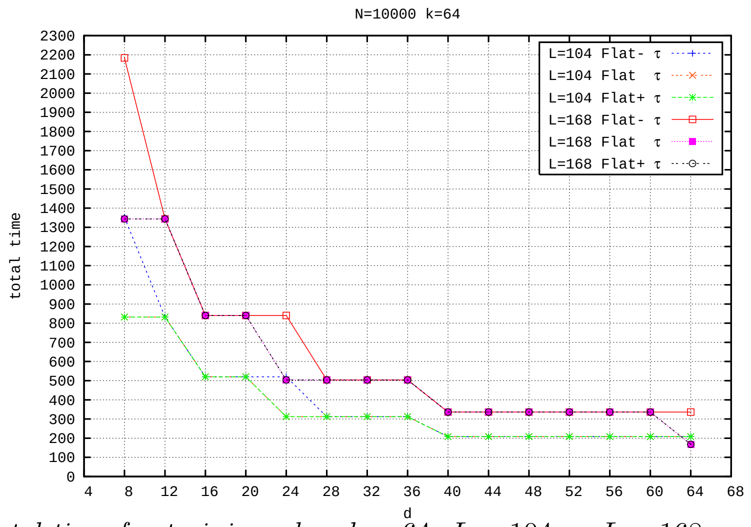

Figure 12: *Total time for training when $k = 64$, $L = 104$ or $L = 168$, and $8 \leq d \leq 64$.*

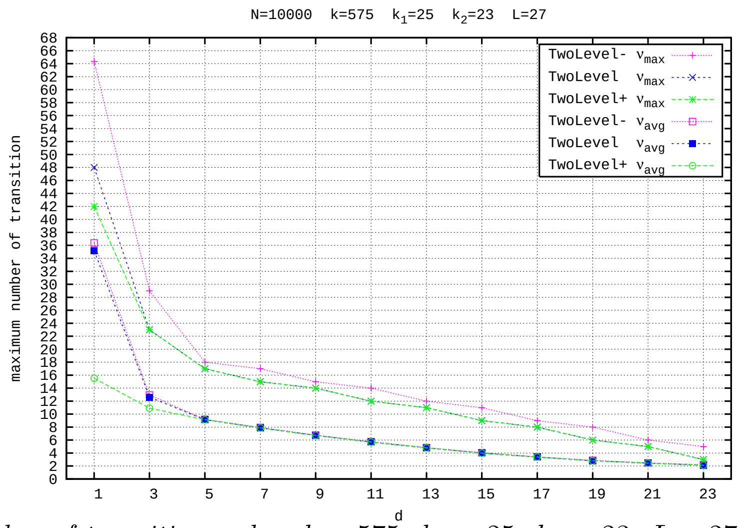

Figure 13: *Number of transitions when $k = 575$, $k_1 = 25$, $k_2 = 23$, $L = 27$, and $1 \leq d \leq 23$.*

in order to increase the longevity of the wireless sensor network. Fixed d, a longer L results in a longer life as the life of a sensor is measured in terms of the overall number of sleep-awake cycles until its energy is exhausted.

Consider now the experiments relative to the two-level approach. Recall that TwoLevel–, TwoLevel, and TwoLevel+ denote, respectively, the protocol when Flat–, Flat, and Flat+ are employed on each single level. In the simulations, the number k of coronas is fixed to 575 while k_1 and k_2 are fixed to 25 and 23, respectively. The length L of the sensor sleep-awake cycle is fixed to 27 and the sensor awake period d varies, with a step of 2, between $\max\{(L, k_1), (L, k_2)\} = 1$ and $\min\{k_1, k_2\} = 23$. The results are averaged over 3 independent experiments.

Figures 13, 14, and 15 plot both the average and worst case performance of ν, ω, and τ. As explained in the previous section, one can easily derive the worst case performance of the two-level protocols in Figure 13 from the worst case performance of the one-level protocols. For example, when $d = (L, k_1) = (L, k_2) = 1$, TwoLevel– requires $\nu = 67$ sleep/wake transitions because the Flat– protocol requires $\nu_1 = \frac{k_1}{(L,k_1)} + \left|\frac{1}{L}\right|_{k_1} = 25 + 13 = 38$ transitions when $k_1 = 25$

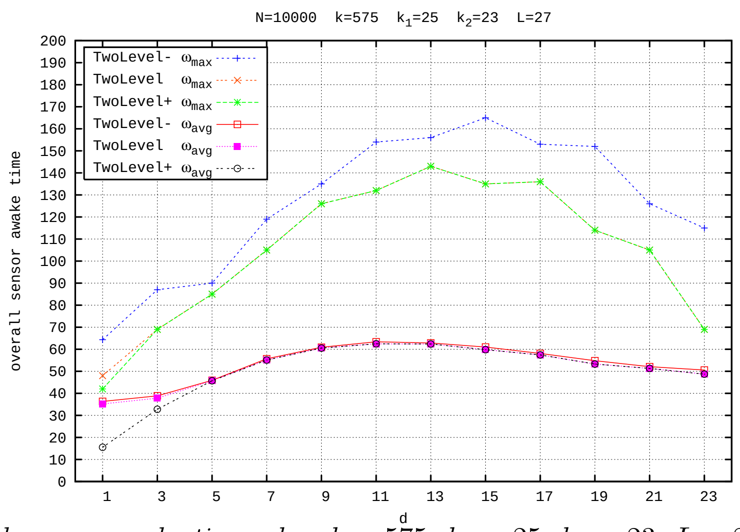

Figure 14: *Overall sensor awake time when* $k = 575$*,* $k_1 = 25$*,* $k_2 = 23$*,* $L = 27$*, and* $1 \leq d \leq 23$*.*

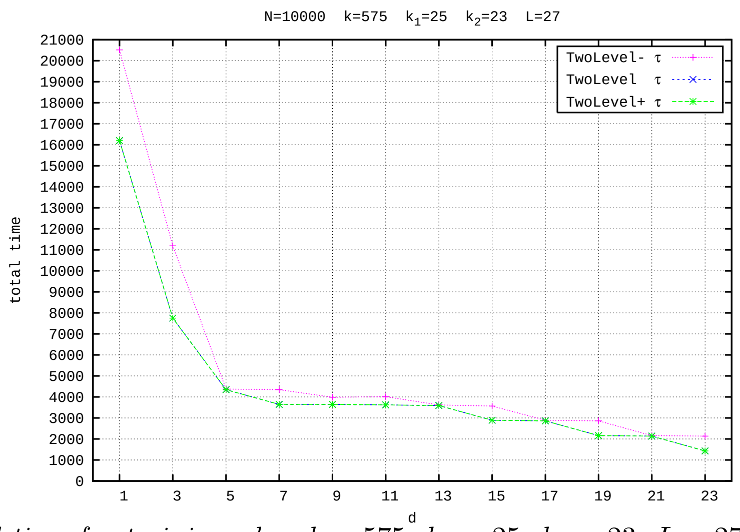

Figure 15: *Total time for training when* $k = 575$*,* $k_1 = 25$*,* $k_2 = 23$*,* $L = 27$*, and* $1 \leq d \leq 23$*.*

and $\nu_2 = \frac{k_2}{(L,k)} + \left|\frac{1}{L}\right|_{k_2} = 23 + 6 = 29$ transitions when $k_2 = 23$.

Figure 14 shows the awake times $\omega_{\max} = \nu_{\max} d$ and $\omega_{\text{avg}} = \nu_{\text{avg}} d$. The curves in Figure 14 smoothly change, without the abrupt peaks of Figure 11, because now the number of transitions monotonically decreases, as shown in Figure 13. Moreover, note that when $d = (L, k) = 1$ the overall awake time is minimum, although the number of transitions is maximum, and TwoLevel+ reaches the maximum gain with respect to the other algorithms, in both the worst and average cases.

Figures 16, 17, and 18 are devoted to compare the behaviour of the Flat and TwoLevel protocols. As before, $k = 575$, $k_1 = 25$, and $k_2 = 23$. The length L of the sensor sleep-awake cycle is fixed to 27 and 577 for TwoLevel and Flat, respectively. Note that both such values of L are the smallest possible choices for the two protocols, because Flat requires L larger than k and TwoLevel needs L larger than $\max\{k_1, k_2\}$. The sensor awake period d varies, with a step of 5, between $(L, k) = 1$ and 76. The results are averaged over 3 independent experiments.

As expected for $1 = (L, k) \leq d \leq \min\{k_1, k_2\} = 23$, the TwoLevel protocol always signif-

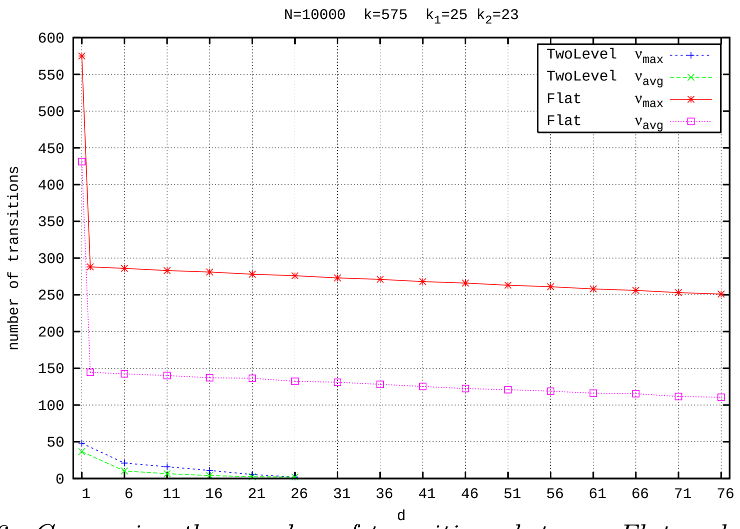

Figure 16: *Comparing the number of transitions between Flat and TwoLevel.*

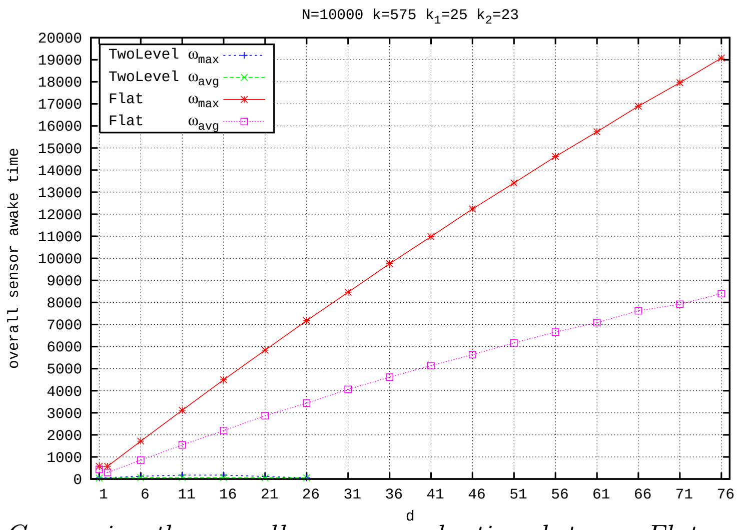

Figure 17: *Comparing the overall sensor awake time between Flat and TwoLevel.*

icantly beats the Flat protocol. Note that, in contrast to Flat, TwoLevel cannot be employed when $d \geq 23$. Observe that $\nu = O(1)$ can be achieved by both the Flat and TwoLevel protocols in correspondence of $d = \Theta(k)$ and $d = \Theta(\sqrt{k})$, respectively, leading therefore to a big difference in the values of ω and τ.

Although two different values of L are used in the experiments, $L = 577$ could be also used for the TwoLevel protocol. In such a case, since $|577|_{25} = |27|_{25}$, the number of transitions ν_1 on the macrocoronas would remain the same, whereas the number of transitions ν_2 on the microcoronas would become slightly larger because $|577|_{23} = 2 < |27|_{23} = 4$. Therefore, $\nu = \nu_1 + \nu_2$ would increase at most by $\lceil \frac{23}{2} \rceil - \lceil \frac{23}{4} \rceil = 6$ with respect to that shown in Figure 16. Obviously the total time τ for TwoLevel with $L = 577$ would dramatically increase with respect to that illustrated in Figure 18, but it would still remain below that of Flat.

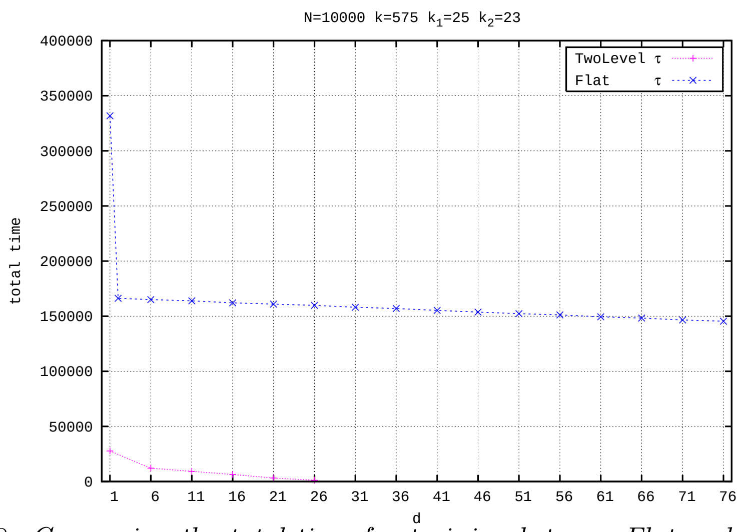

Figure 18: *Comparing the total time for training between Flat and TwoLevel.*

7 Concluding remarks

In this work new protocols have been proposed which employ the asynchronous model originally presented in [30] and are lightweight in terms of the number of sleep/wake transitions and overall sensor awake time for training. Among the various protocol variants and improvements, Flat– is the simplest one from the computational viewpoint because each sensor performs $O(1)$ operations per time slot. In contrast, TwoLevel+ has the best performance, but all the two-level protocols, as well as Flat+, cannot be used if the sensor is not allowed to skip one or more awake periods.

The results presented in this paper show that the protocols are flexible, in the sense that their parameters can be properly tuned. For instance, fixed the number k of coronas, one can decide the optimal values of d and L so as to minimize the number of sleep/wake transitions and/or the overall awake time per sensor. Conversely, one can fix the desired number of sleep/wake transitions, and then select suitable values of d and L.

However, several questions still remain open. In particular, a good idea for further work should be that of comparing the performance of the protocols proposed in the present paper with those devised in [4]. Indeed, the synchronous training protocols of [4] present an irregular toggling between sleep and wake periods, so as to optimize the total time for training, but they consume energy in the explicit synchronization between the sensors and the actor to handle such irregular sleep/wake toggling. In contrast, the asynchronous protocols proposed in the present paper assume sensors to periodically follow sleep-awake cycles, thus avoiding irregular toggling, but they take a larger total time for training. Finally, in this paper, a single fixed actor was assumed for training the sensors. As an extension, one could consider the case where there are several fixed actors, knowing about each other, and thus determine whether they can collaboratively train sensors better than using only a single actor. Moreover, one could study how the protocols presented in this paper can be adapted to the case of one or more mobile actors.

References

[1] I.F. Akyildiz and I. Kasimoglu. Wireless sensor and actor networks: research challenges. *Ad Hoc Networks*, 2, 351–367, 2004.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[3] S. Bandyopadhyay and E. Coyle. An efficient hierarchical clustering algorithm for wireless sensor networks. *Proc. IEEE INFOCOM 2003*, San Francisco, CA, April 2003.

[4] A. A. Bertossi, S. Olariu, and M.C. Pinotti. Efficient training of sensor networks. *Proc. 2nd International Workshop on Algorithmic Aspects of Wireless Sensor Networks (AlgoSensors 2006).* Lecture Notes in Computer Science 4240, 1–12, 2006.

[5] D. Culler, D. Estrin, and M. Srivastava. Overview of sensor networks. *IEEE Computer*, 37(8):41–49, 2004.

[6] K. A. Delin and S. P. Jackson. The sensor web: a new instrument concept. *Proc. SPIE Symposium on Integrated Optics*, San Jose, California, January 2001.

[7] D. M. Doolin and N. Sitar. Wireless sensors for wild remonitoring. *Proc. SPIE Symposium on Smart Structures and Materials (NDE 2005)*, San Diego, California, March 6-10, 2005.

[8] S. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh. Optimal energy-aware clustering in sensor networks. *Sensors*, 2:258–269, 2002.

[9] H. Griffin, *Elementary Theory of Numbers.* McGraw Hill, New York, 1954.

[10] B. Hemingway, W. Brunette, T. Anderal, and G. Boriello. The flock: Mote sensors sing in undergraduate curriculum. *IEEE Computer*, 37(8), 72-78, 2004.

[11] K. Langendoen and N. Reijers. Distributed localization algorithm. In *Embedded Systems Handbook*, R. Zurawski (Editor), CRC Press, Boca Raton, FL, 2004.

[12] J. J. Lee, B. Krishnamachari, and C. C. Jay Kuo. Impact of heterogeneous deployment on lifetime sensing coverage in sensor networks. *Proc. IEEE SECON*, 2004.

[13] K. Martinez, J.K. Hart, and R. Ong. Sensor network applications. *IEEE Computer*, 37(8):50–56, 2004.

[14] D. Nicolescu. Positioning in ad-hoc sensor networks. *IEEE Network*, 18(4):24–29, 2004.

[15] D. Nicolescu and B. Nath. Ad-hoc positioning system. In *IEEE GlobeCom*, November 2001.

[16] S. Olariu, M. Eltoweissy, and M. Younis. ANSWER: autonomous networks sensor systems. *Journal of Parallel and Distributed Computing*, 67, 114–126, 2007.

[17] S. Olariu, A. Waada, L. Wilson, and M. Eltoweissy. Wireless sensor networks leveraging the virtual infrastructure. *IEEE Network*, 18(4):51–56, 2004.

[18] J. Polastre, R. Szewcyk, A. Mainwaring, D. Culler, and J. Anderson. Analysis of wireless sensor networks for habitat monitoring. In *Wireless Sensor Networks*, Raghavendra, Sivalingam, and Znati, Eds., Kluwer Academic, 399-423, 2004.

[19] K. Ryokai and J. Cassell. StoryMat: A play space for collaborative storytelling. *Proc. CHI'99*, October 1999.

[20] C. Savarese, K. Langendoen, and J. Rabaey. Robust positioning algorithms for distributed ad hoc wireless sensor networks. In *USENIX Technical Annual Conference*, Monterey, CA, June 2002.

[21] A. Savvides, L. Girod, M. Srivastava, and D. Estrin. Localization in Sensor Network. In *Wireless Sensor Networks*, C.S. Raghavendra, K.M. Sivalingam, and T. Znati, Eds., Kluwer Academic, 2004.

[22] A. Savvides, H. Park, and M. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *First ACM Int'l Workshop on Wireless Sensor Networks and Application (WSNA)*, Atlanta, September 2002.

[23] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, 7(5), 16–27, 2000.

[24] M. Srivastava, R. Muntz, and M. Potkonjak. Smart Kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments. *Proc. ACM MOBICOM*, Rome, Italy, July 2001.

[25] N. S. Szabo and R. I. Tanaka. *Residue Arithmetic and its Applications to Computer Technology.* McGraw-Hill, New York, 1967.

[26] R. Szewczyk, E. Osterweil, J. Polatre, M. Hamilton, A. Mainwaring, and D. Estrin. Habitat monitoring with sensor networks. *Communications of the ACM*, 47(6), 34–40, 2004.

[27] R. Szewczyk, J. Polastre, A. Mainwaring, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. *Proc. 2nd ACM Conference on Embedded Networked Sensor Systems*, Nov. 2004.

[28] A. Waada, S. Olariu, L. Wilson, M. Eltoweissy, and K. Jones. Training a wireless sensor network. *Mobile Networks and Applications*, 10(1):151–168, 2005.

[29] B. Warneke, M. Last, B. Leibowitz, and K. Pister. SmartDust: communicating with a cubic-millimeter computer. *IEEE Computer*, 34(1):44–51, 2001.

[30] Q. Xu, R. Ishak, S. Olariu, and S. Salleh. On asynchronous training in sensor networks. *Proc. 3rd Intl. Conf. on Advances in Mobile Multimedia*, K. Lumpur, September 2005.

[31] V.V. Zhirnov and D.J.C. Herr. New frontiers: self-assembly and nano-electronics. *IEEE Computer*, 34(1):34–43, 2001.