Journal of Mobile Multimedia, Vol. 3, No. 1 (2007) 034–046
© Rinton Press

ON ASYNCHRONOUS TRAINING IN SENSOR NETWORKS[a]

QINGWEN XU
Department of Computer Science, Old Dominion University
Norfolk, Virginia 23529, USA
xu_q@cs.odu.edu

RUZANA ISHAK
Department of Mathematics, Universiti Technologi Malaysia
54100 Kuala Lumpur, Malaysia
ruzanaishak@yahoo.com

STEPHAN OLARIU
Department of Computer Science, Old Dominion University
Norfolk, Virginia 23529, USA
olariu@cs.odu.edu

SHAHARUDDIN SALLEH
Department of Mathematics, Universiti Technologi Malaysia
81310 Johor Bahru, Malaysia
ss@mel.fs.utm.my

Received November 30, 2005
Revised May 10, 2006

Due to their small form factor and modest energy budget it is infeasible to endow individual sensors with GPS capabilities. Yet, numerous applications require sensors to have a *coarse-grain* location awareness. The task of acquiring this coarse-grain location awareness is referred to as *training*. The main contribution of this work is to propose a fully asynchronous training protocol for massively-deployed sensor networks. The sensors wake up according to their internal clock and are not engaging in synchronization with the sink. Our protocol is lightweight and simple to implement. We show analytically that in spite of the lack of synchronization, individual sensors are trained energy-efficiently. The analytical results have been confirmed by simulation.

Keywords: wireless sensor networks, self-organization, coarse-grain location awareness, dynamic coordinate system, clustering, asynchronous protocols.
Communicated by: I. Ibrahim

1 Introduction

Recent technological advances have made it possible to develop a large variety of miniaturized low-power devices that integrate sensing, special-purpose computing and wireless communications capabilities [2, 17, 18, 19, 28, 39]. It is expected that these tiny devices, referred to as *sensors*, will be mass-produced and deployed, making their production cost negligible. Individual sensors have a small, non-renewable power supply and, once deployed, must work

[a]Address for correspondence olariu@cs.odu.edu

unattended. We envision a massive deployment of sensors, perhaps in the thousands or even tens of thousands [38].

Aggregating sensors into sophisticated computational and communication infrastructures, called wireless sensor networks (*sensor networks*, for short), are expected to have a significant impact on a wide array of applications ranging from military, to scientific, to industrial, to health-care, to domestic, establishing smart environments that will pervade society redefining the way in which we live and work [20, 28]. The novelty of sensor networks and the tremendous potential for a multitude of application domains has triggered a flurry of activity in both academia and industry.

Sensor network research as we know it can be traced to a DARPA sponsored program, SmartDust, originated in the late 1990's [2, 7, 14, 33, 38]. The title of the program creatively described its goal: to make machines with self-contained sensing, computing, transmitting, and powering capabilities so small and inexpensive that they could be released into the environment in massive numbers. DARPA defined a sensor network as follows: "*A sensor network is a deployment of massive numbers of small, inexpensive, self-powered devices that can sense, compute, and communicate with other devices for the purpose of gathering local information to make global decisions about a physical environment.*"

A few years later, the National Research Council's Committee on Networked Systems of Embedded Computers published a report [20] defining the concept of the embedded network, EmNet, as a network of heterogeneous computing devices pervasively embedded in the environment of interest. Since that time, the term wireless sensor network (sensor network, for short) has evolved to describe such systems and research abounds in academic and commercial environments. Improvements in Micro Electro-Mechanical Systems (MEMS) technology is producing, at rapidly decreasing cost, smaller and smaller self-powered devices that can sense, compute, calculate, and communicate [28]. These devices have come to be called *sensor motes* and serve as nodes in a sensor network. As these new motes are severely energy constrained, they cannot transmit over long distances. Much current research is concluded with forming multi-hop networks, to send information to and from the network motes where no single mote need transmit any farther than to nearby motes.

As technology improves, computing and communicating capabilities are expected to improve at an accelerating rate and new techniques for supplying energy will significantly reduce the low power constraint [1, 8, 34]. Increased capabilities will be possible and futurists predict that societies of machines will evolve to be autonomous, cooperative, fault-tolerant, self-regulating, and self-healing [28]. Due to the limited resources of available motes, most current implementations of sensor networks are composed of a small number of motes that collect simple sensory values and report these values to a central entity, sometime called the *end-user*. In the past few years we have witnessed the deployment of sensor networks in support of a growing array of applications ranging from smart kindergarten [35], to smart learning environments [4], to habitat monitoring [25, 36], to environment monitoring [16], to wildfire prevention and monitoring [6], to undergraduate education [11], among others. These prototypes provide solid evidence of the usefulness of sensor networks and suggest that the future will be populated by pervasive sensor networks that will redefine the way we live and work. It is, thus, to be expected that in the near future, in addition to the examples above, a myriad of other applications including battlefield command and control, disaster management and emergency response, will involve sensor networks as a key mission-critical component.

The massive deployment of sensors, combined with the lack of individual IDs, a modest energy budget, and – in many applications – a hostile environment, pose daunting challenges to the design of protocols for sensor networks. For one thing, the limited energy budget mandates the design of ultra-lightweight communication protocols. Likewise, issues concerning how the data collected by individual sensors may be queried and accessed and how concurrent sensing tasks can be performed internally are of particular significance. An important guideline in this direction is to perform as much local data processing at the sensor level as possible, avoiding the transmission of raw data through the sensor network. Indeed, it is known that it costs 3J of energy to transmit 1Kb of data a distance of 100 meters. Using the same amount of energy, a general-purpose processor with the modest specification of 100 million instructions/watt executes 300 million instructions [26].

As a consequence, the sensor network must be multi-hop and only a tiny minority of the sensors count the sink among their one-hop neighbors. For reasons of scalability, it is assumed that no sensor knows the topology of the network.

1.1 Interfacing sensor networks

There are several possible techniques available for interfacing sensor networks to the outside world and for harvesting the data they produce. Perhaps the simplest involves using one or several sinks – special long-range radios – deployed alongside with the sensors. Each sink has a full range of computational capabilities, can send long-range directional broadcasts to the sensors at distance at most R (see Figure 2), can receive messages from nearby sensors, and has a steady power supply. In this scenario, the raw data collected by individual sensors is fused, in stages, and forwarded to the nearest sink that provides the interface to the outside world. Such a scenario, involving three sinks A, B, and C, is depicted in Figure 1.

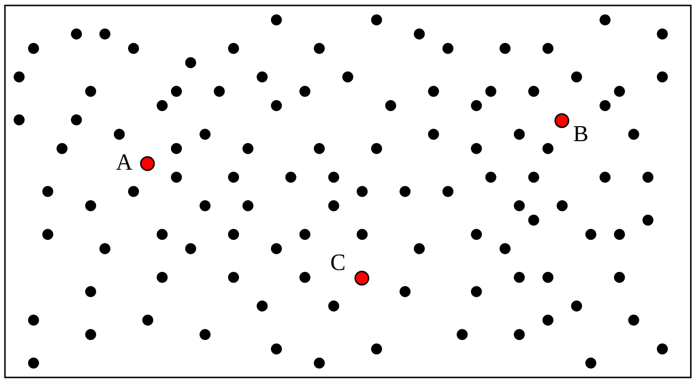

Fig. 1. Illustrating a multi-sink sensor network.

1.2 Our contribution

While some applications require sensory data with exact geographic location, motivating the development of communication protocols that are location aware and perhaps location dependent, in most applications, exact geographic location is not necessary: all that individual sensors need is coarse-grain location awareness. There is, of course, an obvious trade-off: coarse-grain location awareness is lightweight but the resulting accuracy is only a rough approximation of the exact geographic coordinates [22, 37].

The random deployment implies that the sensors are initially unaware of their exact location. Further, due to limitations in form factor, cost per unit and energy budget, individual sensors are not expected to be GPS-enabled; moreover, many application environments limit satellite access. It follows that the sensors have to learn either their exact geographic location (if specifically required by the application) or else a coarse-grain approximation thereof. The former task is referred to as *localization* and has received a good deal of well-deserved attention in the literature [15, 21]. The latter task, referred to as *training* was extensively discussed by Olariu *et al.* [22] and Wadaa *et al.* [37], who proposed elegant training protocols for sensor networks. However, these protocols assume that the sink and the sensors are somehow synchronized. While a number of synchronization protocols have been published in the literature [22, 30, 31], synchronization is an additional overhead that makes training unnecessarily complicated.

The main contribution of this work is to propose a fully asynchronous training protocol for massively-deployed sensor networks. The sensors wake up according to their internal clock and are not engaging in a synchronization protocol with the sink. Our protocol is truly lightweight and simple to implement. We show analytically that in spite of the lack of synchronization, individual sensors are trained very efficiently. Extensive simulation results have confirmed that our asynchronous training protocol is energy efficient, each sensor being awake for a total time interval that is proportional with its wake-to-sleep ratio.

The remainder of this paper is organized as follows. Section 2 discusses the sensor model used throughout the work. Section 3 introduces the task of training – that is, endowing individual sensors with coarse-grain location awareness. Section 4 is the backbone of the entire paper, presenting the theoretical underpinnings of the asynchronous training process. Section 5 presents the simulation model and the simulation results. Finally, Section 6 offers concluding remarks.

2 The sensor model

We assume a sensor to possess three basic capabilities: sensory, computation, and wireless communication. The sensory capability is necessary to acquire data from the environment; the computational capability is necessary for aggregating data, processing control information, and managing both sensory and communication activity. Finally, wireless communication is necessary for sending/receiving aggregated data and control information to (from) other sensors or the sink.

We assume that individual sensors are tiny, mass-produced devices that operate subject to the following fundamental constraints.

a. Sensors are *anonymous* – they do not have or need individually unique IDs,

b. Each sensor has a modest non-renewable energy budget,

c. In order to save energy, sensors are in *sleep* mode most of the time, waking up for short intervals,

d. Each sensor has a modest transmission range, perhaps a few meters. This implies that out-bound messages can reach only the sensors in its proximity, typically a tiny fraction of the sensors deployed,

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

It is worth mentioning that while the energy budget can support short-term applications, sensors dedicated to work over extended periods of time may need to scavenge energy from the specific environment they are placed into, employing light, temperature, vibration, kinetics, magnetic fields, etc. [27, 29].

3 Training a sensor network

Throughout this work, we assume an autonomous sensor network where a number of sinks provide local aggregation points for the data collected by the individual sensors. Each sink organizes the sensors in a disk of radius R into an autonomous, short-lived sensor network as illustrated in Figure 2.

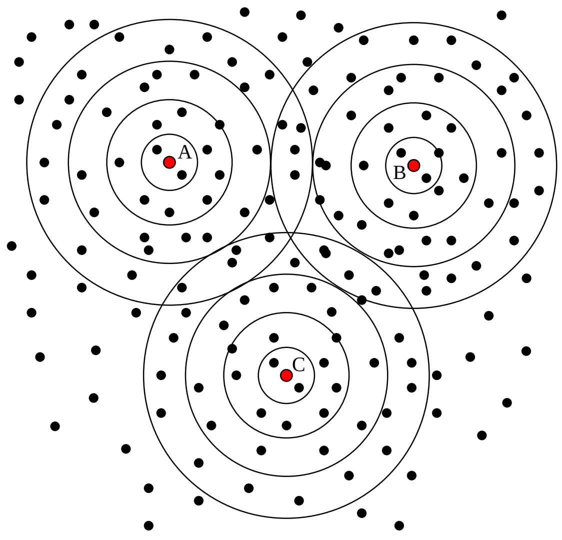

Fig. 2. Illustrating autonomous sensor networks, each trained by a sink.

For ease of understanding we shall present the training processed centered at one of the several sinks in the deployment area. It is important to understand that each sink is engaging in the training process either independently of other sinks or in a concerted effort. Figure 2 shows three sinks, each training independently the sensors in its neighborhood.

The task of endowing sensors with coarse-grain location awareness, task referred to as *training*, is essential in numerous applications. One example is clustering where the set of sensors deployed in an area is partitioned into *clusters* [2, 3, 10, 32]. As a result of training, we impose a coordinate system onto the sensor network in such a way that each sensor belongs to exactly one sector. The coordinate system divides the sensor network area into equiangular wedges. In turn, these wedges are divided into sectors by means of concentric circles or *coronas* centered at the sink and whose radii are determined to optimize the transmission efficiency of sensors-to-sink transmission as will be discussed later. Sensors in a given sector map to a cluster, the mapping between clusters and sectors is one-to-one.

Olariu *et al.* [22] argue that training is a very effective and lightweight clustering strategy. This follows immediately from the fact that sensors in a given sector map to a cluster, the mapping between clusters and sectors is one-to-one. A cluster is the locus of all the sensors that have the same coordinates in the dynamic coordinate system of Figure 3. It is also worth noting that as a result of training, the sensors acquire a rough location awareness (modulo the sector to which they belong) as well as a form of collective identity [22].

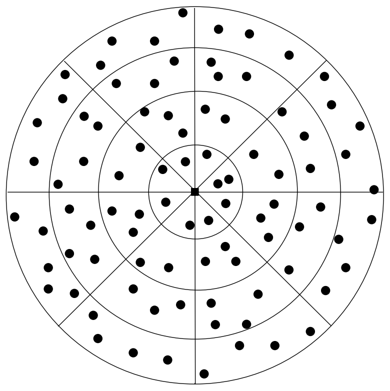

Fig. 3. Illustrating the dynamic coordinate system.

Referring to Figure 3, the task of training a sensor network involves establishing:

Coronas: The deployment area is covered by k coronas determined by k concentric circles of radii $0 < r_1 < r_2 < \cdots < r_k = R$ centered at the sink.

Wedges: The deployment area is ruled into a number of angular wedges centered at the sink. Wedges are established by directional transmission [22].

As illustrated in Figure 3, 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. Importantly, the locus of all the sensors that have the same coordinates determines a cluster.

4 The asynchronous training protocol

The main goal of this section is to discuss the details of our efficient asynchronous training protocol for a collection of sensors deployed uniformly at random in a disk of radius R centered at a local sink.

The idea of the protocol is as follows. Immediately after deployment, the sink repeats the transmission cycle illustrated in Figure 4. Assuming that k coronas $C_1, C_2, \ldots, C_k$ have to

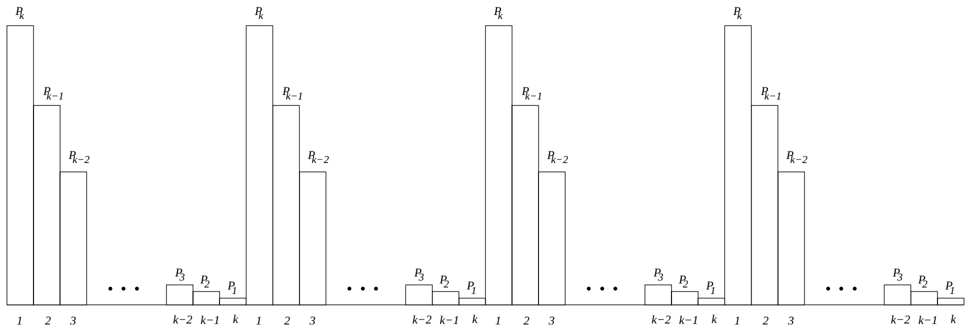

Fig. 4. Illustrating four sink transmission cycles.

be established, the sink *transmission cycle* involves k broadcasts at successively lower power levels. The sink starts out by transmitting at the highest power, sufficient to reach the sensors in the outmost corona C_k; next, the sink transmits at a power level that can be received in corona C_{k-1} but not C_k. This is, then, continued until in the sink transmits at a power level that can be received only by the sensors in corona C_1.

4.1 Selecting optimal sensor parameters

Each sensor alternates between sleep periods and awake periods. Referring to Figure 5, the sensor sleep-awake cycle is of total length L: the sensor is in sleep mode for $L-d$ time and in awake mode for d time units. Sink-cycles and sensor cycles will be referred to, respectively, as $k - cycles$ and $s - cycles$.

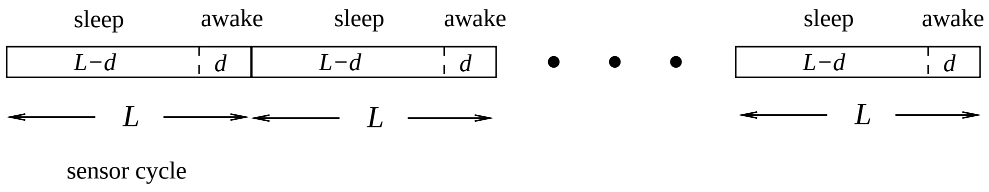

Fig. 5. Illustrating the sensor sleep-awake cycle.

Since wedge training is similar (in fact, simpler than corona training), in the remainder of this section we focus on corona training only. The goal is to guarantee that each sensor is corona-trained within n, $(n \geq 1)$, s-cycles *regardless* of the moment when it wakes up for the first time. We propose to derive mathematical expressions that allow a sensor to tailor both L and d (as functions of n and k) in such a way as to achieve this goal as efficiently as possible.

To begin, observe that since a sensor is awake for d time units per s-cycle, the smallest value of d that guarantees that the sensor can be trained in n s-cycles is

$$d = \left\lceil \frac{k}{n} \right\rceil . \tag{1}$$

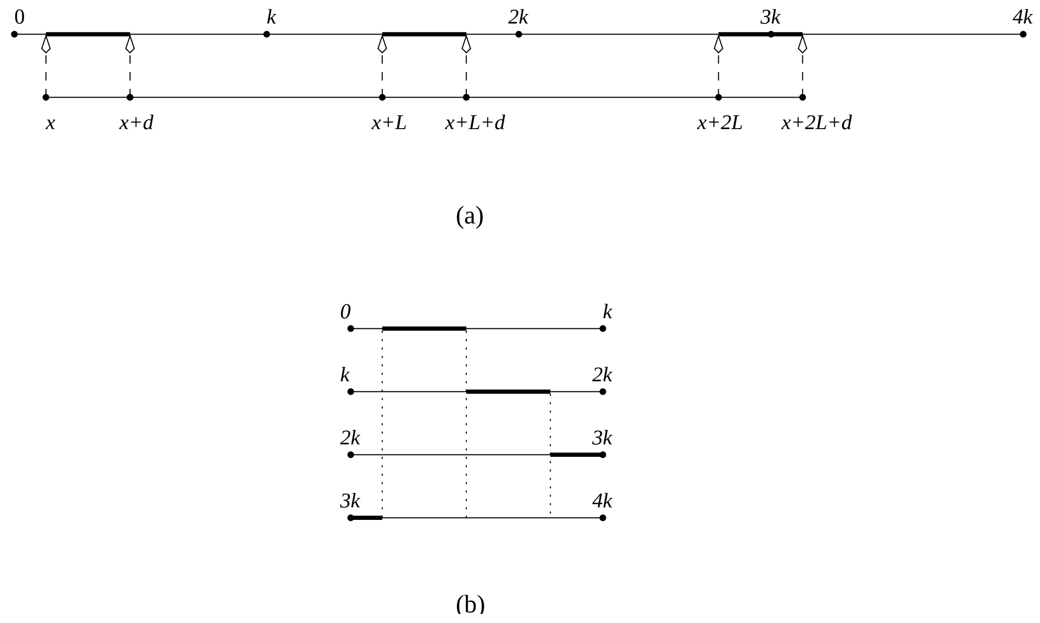

Fig. 6. Illustrating Theorem 1 for $n = 3$.

To simplify the notation, in the remainder of this section we assume that $\frac{k}{n}$ is an integer. Referring to Figure 6(a) assume, without loss of generality, that the first k-cycle begins at time 0 and that for some arbitrary x, the sensor wakes up x time units after the beginning of the k-cycle. With this assumption, it is easy to confirm that the n awake periods $A_1, A_2, \ldots A_n$ of the sensor are given for $1 \leq i \leq n$ by

$$A_i = [x + iL, x + iL + d]. \tag{2}$$

By projecting each of these intervals onto the generic interval $[0, k]$ of length k we obtain a new sequence $\overline{A_1}, \overline{A_2}, \ldots \overline{A_n}$ of intervals such that for $(0 \leq i \leq n-1)$

$$\overline{A_i} = [(x + iL) \bmod k, (x + iL + d) \bmod k]\,. \tag{3}$$

We note that the $\overline{A_i}$'s are *modulo* intervals in the sense that there may be a jump from the end of the generic interval to its beginning. Nonetheless, as we shall see this phenomenon does not create any difficulties.
In order to optimize coverage, we insist that the rightmost endpoint of the modulo interval $\overline{A_i}$ coincide with the left endpoint of the modulo interval $\overline{A_{i+1}}$. In other words, we are interested in determining L in such a way that

$$(x + iL + d) \bmod k = (x + (i+1)L) \bmod k. \tag{4}$$

In turn, (4) implies that

$$L - d \equiv 0 (\bmod k)$$

or, equivalently,

$$L = mk + d \tag{5}$$

for some natural number m.
Now, replacing the value of L suggested by (5) into (3) we obtain a new form for the modulo intervals $\overline{A_i}$. Specifically, we can write

$$\overline{A_i} = [(x + id) \bmod k, (x + (i+1)d) \bmod k] \,. \tag{6}$$

It is also easy to confirm that by virtue of (4) the union of the $\overline{A_i}$s covers the generic interval $[0, k]$ of length k. Thus, we have proved the following important result.

Theorem 1 *Given that $d = \frac{k}{n}$, a sufficient condition for training a sensor in n s-cycles regardless of the first wakeup time is that $L = mk + \frac{k}{n}$ for some arbitrary natural number m.*

To help the reader visualize what is going on, suppose that $n = 3$, and refer to Figure 6. The goal is to train the sensor in 3 s-cycles regardless of its first time x when it wakes up. Figure 6(a) illustrates the s-cycle as well as the wake-up periods in heavy lines. Figure 6(b) shows that the projections of $\overline{A_1}$, $\overline{A_2}$ and $\overline{A_3}$ cover the interval $[0, k]$.

5 Simulation results

Our simulation is implemented using PARSEC, a discrete-event simulator developed at UCLA. In our simulation, 1000 sensors are deployed uniformly at random in a field of size 640m x 640m. A sink node is placed in the middle of the field at (320, 320). The number k of coronas is 32 and each corona has a width of 10 meters. Hence, the length of a k-cycle is 32 time slots (a slot is 10 milliseconds). The sink transmits a beacon at a different power level repeatedly as shown in Figure 4. Each beacon contains the current corona number. The sensors wake up at random to simulate the asynchronous effect. For each sensor, the first wakeup time is generated uniformly at random in the interval $(0, T)$, where $T = 32$ (in fact, for the purpose of the simulation, T is taken to be 320,000 because the time unit is taken to be the microsecond, so T is equal to 32 time slots). Hence, each sensor nodes will wake up at a random time between between that first and the 32-nd time slot.

After wakeup, each sensor listens for d time slots and then goes back to sleep for $L - d$ time slots. The sensor is trained if it hears the beacon in some slot s but not in slot $s + 1$. If so, the corona number advertised in slot s is the corona number of the sensor and the sensor node is trained by the sink node. The only exception is a sensor in corona C_1 that will hear a transmission both in slot s and $s + 1$ except that the received power in $s + 1$ is larger than in s.

By equation (1) in Subsection 4.1, if we want all sensors to be trained in 3 s-cycles, i.e, $n = 3$, then d must equal 11. If we choose $m = 2$, then by Theorem 1, the length L of the s-cycle will be $L = mk + d = 75$ and the total time to train all the sensor nodes will be $2L + k + d = 150 + 32 + 11 = 193$ time slots. Our simulation confirms our theoretical prediction as showed in Figure 7.

From Figure 7, one can see that the nodes are evenly trained in the 3 s-cycles. More specifically, in the first training interval, approximately 33.3% of the nodes are trained. The same percentage of the nodes is trained in the second and the third training intervals. In the figure, the first training interval is from 11 to 43 because each sensor wakes up randomly at a time between 0 and 32 time slot and listens for 11 time slots. In the first training interval, the earliest time that a node is trained is at time 11 and the latest time is 43. In the second

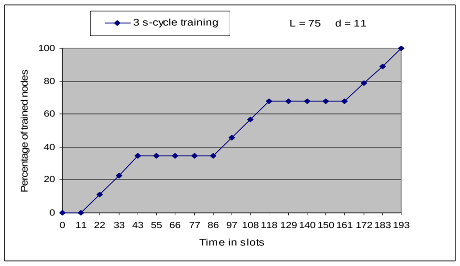

Fig. 7. Illustrating three s-cycle training.

training interval, the earliest time that a node is trained is at time 11+75 = 86, the latest time is 43+75 = 118, and so on.

Figures 8 and 9 show the total training time (in slots) with the different values of d (the value of L remain fixed at 75). It is clear that 11 is the optimal value for training the nodes. Indeed, if d is smaller than 11, then it takes much longer to train the sensors. The overall awake time is larger than 33 (33 is the overall awake time for 3 s-cycles when $d = 11$). For instance, in Figure 8 we can see that if $d = 5$, the total time slots needed to train the sensor nodes is around 1500 (20 s-cycles), which requires overall 100 (20*5) awake time slots to train the sensor nodes. On the other hand, if d is larger than 11 (see Figure 9), the overall awake time is still larger than 33 because a larger awake interval does not help much. For instance, when $d = 21$, the sensor nodes need 2 s-cycles to get trained, which makes the overall awake time equal to 42.

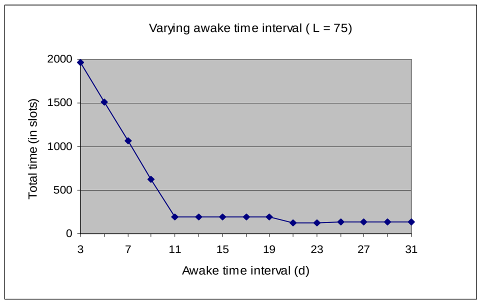

Fig. 8. Varying awake time interval – the overall picture.

Figure 10 shows that the length of s-cycle is also very important for node training. The larger is the greatest common divisor of L and k, the worse for training the sensor nodes. For example, if $L = 104$ and $k = 32$, the greatest common divisor of L and k is 8, which means the sensors has to be trained in 4 s-cycles. Any sensor is not trained in 4 s-cycles will never be trained. It is illustrated in the Figure 10 that only 37.5% and 62.5% nodes are trained

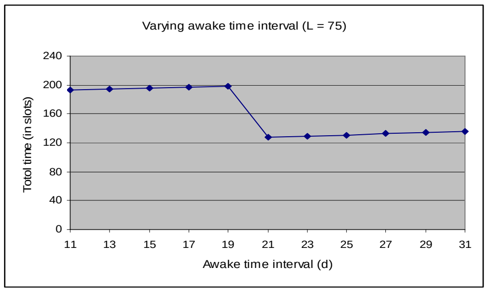

Fig. 9. Varying awake time interval – the details.

when $d = 3$ and $d = 5$, respectively. This makes sense because only 37.5% and 62.5% of 32 is covered in 4 s-cycles when $d = 3$ and $d = 5$, respectively. In this case, d has to be at least 8 to cover 32 in 4 s-cycles and all nodes will be trained.

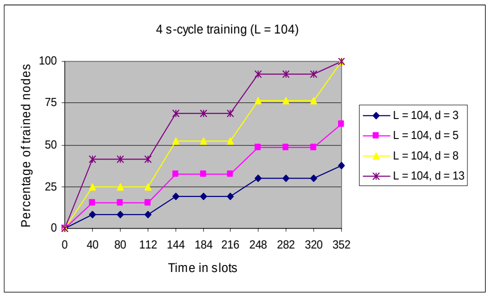

Fig. 10. The importance of the s-cycle length.

Our simulation results show that if k is known, we can decide the number n of s-cycles needed to train the sensors and we can compute the optimal values of d and L s a function of both n and k, using Theorem 1. However, if k is not known in advance, we are not able to compute the optimal value of d and L. One important conclusion that the simulation reveals is that the value of L should be a prime number in order to minimize the greatest common divisor of L and k.

6 Concluding remarks

The main contribution of this work was to propose an asynchronous training protocol for sensor networks. We showed analytically that in spite of the lack of synchronization, individual sensors are trained energy-efficiently. The analytical results have been confirmed by extensive simulation.

A number of problems remain open. First, the training protocol can be further improved by reducing the amount of time that a sensor needs to be awake. There is a trade-off here:

the synchronous training protocol of Wadaa *et al.* [37] requires individual sensor to be awake for only $\log k$ time (which may be shorted than d) but consumes a lot of energy in both synchronization and the toggling between sleep and wake periods [39]. Our protocol may force sensors to be awake for longer periods but avoids frequent transitions from sleep to wake periods. Striking the right balance between the two promises to be an exciting area of further work.

References

1. J. Agre and L. Clare, An integrated architecture for cooperative sensing networks, *IEEE Computer*, 33(5), 2000, 106–108.
2. F. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayirci, Wireless sensor networks: A survey, *Computer Networks*, 38(4), 2002, 393–422.
3. S. Bandyopadhyay and E. Coyle, An efficient hierarchical clustering algorithm for wireless sensor networks, *Proc. INFOCOM'2003*, San Francisco, California, April 2003.
4. D. Culler, D. Estrin, and M. Srivastava, Overview of sensor networks, *IEEE Computer*, 37(8), 2004, 41–49.
5. D. Culler and W. Hong, Wireless sensor networks, *Communications of the ACM*, 47(6), 2004, 30–33.
6. D. M. Doolin and N. Sitar, Wireless sensors for wild fire monitoring, *Proc. SPIE Symposium on Smart Structures & Materials*, (NDE 2005), San Diego, California, March 6-10, 2005
7. D. Estrin, R. Govindan, J. Heidemann and S. Kumar, Next century challenges: Scalable coordination in sensor networks, *Proc. MOBICOM*, Seattle, WA, August 1999.
8. D. Estrin, D. Culler, K. Pister and G. Sukhatme, Instrumenting the physical world with pervasive networks, *Pervasive Computing*, 1(1), 2002, 59-69.
9. C. C. Enz, A. El-Hoiydi, J.-D. Decotignie and V. Peiris, WiseNET: a ultralow power wireless sensor network solution, *IEEE Computer*, 37(8), 2004, 62–69.
10. S. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh, Optimal energy-aware clustering in sensor networks, *Sensors*, 2, 2002, 258–269.
11. B. Hemingway, W. Brunette, T. Anderl and G. Boriello, The flock: Mote sensors sing in undergraduate curriculum, *IEEE Computer*, 37(8), 2004, 72-78.
12. R.Ishak, S.Olariu, Q.Xu and S.Salleh, Dual-training for Massively-Deployed Sensor Networks, Submitted to 13th International Conference of Telecommunications, Portugal, May 2006.
13. J. Hill, M. Horton, R. Kling and L. Krishnamurthy, The platforms enabling wireless sensor networks, *Communications of the ACM*, 47(6), 2004, 41–46.
14. J. M. Kahn, R. H. Katz, and K. S. J. Pister, Next century challenges: Mobile support for Smart Dust, *Proc. ACM MOBICOM*, Seattle, WA, August 1999, 271–278.
15. K. Langendoen and N. Reijers, Distributed localization algorithm, in R. Zurawski (Ed.), *Embedded Systems handbook*, CRC Press, Boca Raton, FL, 2005.
16. K. Martinez, J. K. Hart and R. Ong, Sensor network applications, *IEEE Computer*, 37(8), 2004, 50–56.
17. http://www.darpa.mil/mto/mems/
18. http://www.stanford.edu/class/ee321/ho/MEMS-14-sensors.pdf
19. http://www.xs4all.nl/g̃answijk/chipdir/m/sensor.htm
20. National Research Council, Embedded, Everywhere: A Research Agenda for Systems of Embedded Computers, Committee on Networked Systems of Embedded Computers, for the Computer Science and Telecommunications Board, Division on Engineering and Physical Sciences, Washington, DC, 2001.
21. D. Niculescu, Positioning in ad hoc sensor networks, *IEEE Network*, 18(4), (2004), 24–29.
22. S. Olariu, A. Wadaa, L. Wilson and M. Eltoweissy, Wireless sensor networks: leveraging the virtual infrastructure, *IEEE Network*, 18(4), (2004), 51–56.

23. S. Olariu and Q. Xu, A simple self-organization protocol for massively deployed sensor networks, *Computer Communications*, 28, (2005), 1505-1516.
24. S. Olariu, M. Eltoweissy, and M. Younis, ANSWER: Autonomous Wireless Sensor Network, *Proc. ACM Q2SWinet*, Montreal, Canada, October 2005.
25. 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, 2004, 399-423.
26. G. J. Pottie and W. J. Kaiser, Wireless integrated sensor networks, *Communications of the ACM*, 43(5), 2000, 51–58.
27. S. Roundy, P. K. Wright, and J. Rabaey, Energy scavenging for wireless sensor networks with special focus on vibrations, Kluwer Academic Press, 2004.
28. P. Saffo, Sensors, the next wave of innovation, *Communications of the ACM*, 40(2), 1997, 93–97.
29. N. S. Shenck and J. A. Paradiso, Energy scavenging with shoe-mounter piezoelectrics, *IEEE Micro*, 21, (2001), 30–41.
30. M. Sichitiu and C. Veerarithiphan, Simple accurate synchronization for wireless sensor networks, *Proc. WCNC'2003*.
31. F. Sivrukaya and B. Yener, Time synchronization in sensor networks: a survey, *IEEE Network*, 18(4), 2004, 45–50.
32. K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie, Protocols for self-organization of a wireless sensor network, *IEEE Personal Communications*, October 2000, 16–27.
33. K. Sohrabi, J. Gao, V. Ailawadhi and G. Pottie, Protocols for self-organization of a wireless sensor network, *IEEE Personal Communications*, 7(5), 2000, 16-27.
34. K. Sohrabi, W. Merrill, J. Elson, L. Girod, F. Newberg and W. Kaiser, Methods for scalable self-assembly of ad hoc wireless sensor networks, *IEEE Transactions on Mobile Computing*, 3(4), 2004, 317-331.
35. 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.
36. R. Szewczyk, E. Osterweil, J. Polatre, M. Hamilton, A. Mainwaring, and D. Estrin, Habitat monitoring with sensor networks, *Communications of the ACM*, 47(6), (2004), 34–40.
37. A. Wadaa, S. Olariu, L. Wilson, K. Jones, and M. Eltoweissy, Training a sensor networks, *MONET*, January 2005.
38. B. Warneke, M. Last, B. Leibowitz, and K. Pister, SmartDust: communicating with a cubic-millimeter computer, *IEEE Computer*, 34(1), 2001, 44–51.
39. V. V. Zhirnov and D. J. C. Herr, New frontiers: self-assembly and nano-electronics, *IEEE Computer*, 34(1), 2001, 34–43.