
Yiannis Chatzigiannakis and Thanasis Kinalis and Sotiris Nikoletseas

2004

Ioannis Chatzigiannakis ∗† Athanassios Kinalis ∗† Sotiris Nikoletseas ∗†

Abstract

We propose, implement and evaluate new energy conservation schemes for efficient data propagation in wireless sensor networks. Our protocols are adaptive, i.e. locally monitor the network conditions and accordingly adjust towards optimal operation choices. This dynamic feature is particularly beneficial in heterogeneous settings and in cases of re-deployment of sensor devices in the network area.

We implement our protocols and evaluate their performance through a detailed simulation study using our extended version of ns-2. In particular we combine our schemes with known communication paradigms. The simulation findings demonstrate significant gains and good trade-offs in terms of delivery success, delay and energy dissipation.

1 Introduction

Wireless Sensor Networks are very large collections of tiny sensor nodes that form ad hoc distributed sensing and data propagation wireless networks that collect quite detailed information about the physical environment. In a usual scenario, these networks are largely deployed in areas of interest (such as inaccessible terrains or disaster places) for fine grained monitoring in various classes of applications [1].

Efficient and robust realization of such large, highly-dynamic, complex, non-conventional networking environments is a challenging algorithmic and technological task. Features including the huge number of sensor devices involved, the severe power, computational and memory limitations, their dense deployment and frequent failures, pose new design and implementation aspects which are essentially different not only with respect to distributed computing and systems approaches but also to ad-hoc networking techniques.

∗1. Computer Technology Institute (CTI), P. O. Box 1122, 261 10 Patras, Greece. Tel: ++30 2610 960324, Fax: ++30 2610 960442. Email: {ichatz,nikolet}@cti.gr
†2. Department of Computer Engineering and Informatics, University of Patras, 26500 Patras, Greece. Email: kinalis@ceid.upatras.gr
A usual model of wireless sensor networks assumes a *homogeneous* set of sensor devices; i.e. sensor networks that are comprised of *identical sensors* with the *same capabilities* in terms of computing, sensing, communication and power. The assumption of having *identical sensors* is quite realistic and captures the idea of simultaneous deployment of a large number of devices [1]. In the setting of *homogeneous* sensor networks, many efficient solutions have been proposed, e.g. [9, 12, 18].

A critical aspect in the design and efficient implementation of wireless sensor networks is to save energy and keep the network functional for as long as possible. However, given the limited capabilities of the power supplies, regardless of the energy efficiency of the protocols, the power of the sensor devices will (inevitably) be exhausted. The disconnection of a certain number of sensor devices causes topology changes in the overall sensor network. Such topology changes may reduce the overall efficiency of the network and even its expected effectiveness, i.e. in cases where connectivity across the network can not be maintained.

A possible scenario to overcome such a situation, in order to extend the lifetime of the sensor network, is to re-deploy additional sensor devices to “replace” the malfunctioning nodes while the network is in operation [1]. In such a case, it is expected that the re-deployed devices will have higher levels of total energy available than the devices initially deployed in the area.

Recently, another scenario was presented by [19] in which sensor devices with different capabilities are used to form sensor networks. The concept is simple: a network comprised of small number of high-end nodes with improved communication capabilities is overlaid on a sensor network. A hierarchical structure is formed and sensors forward data under propagation to the high-end nodes that can propagate data at higher distances. Theoretically, this should enable faster trips across the network and result in improved performance [19]. An alternative is to avoid adding any high-end nodes to the network but instead introduce sensor nodes that have larger energy resources available or even sensor nodes that are *plugged into the wall*. How would this improve the networks performance? Would it help the rest of the network to survive longer, by shifting the burden of forwarding packets to these nodes and away from the battery-powered motes? How such an improvement compares to that of deploying from the start a larger number of homogeneous sensors?

In the light of the above possible scenarios, we study the emergence of networks where the assumption of a *homogeneous* set of sensor devices may not always hold. We identify possible situations where a collection of devices that have different technical characteristics may improve the performance of the network compared to “typical” (homogeneous) sensor networks made up from identical devices. Note that related ideas of *heterogeneous* sensor networks have been mentioned in [26] and also studied in [13, 21].

### 1.1 Our Contribution

In this paper we propose a family of power conservation schemes that can be used to improve the energy-efficiency of sensor networks. More specifically, we apply our schemes to the communication paradigm for
sensor networks called Directed Diffusion (see [18]), an approach to attribute-based data communication for wireless sensor networks. The goal of Directed Diffusion is to establish communication between sources and sinks. Data is named using attribute-valued pairs. A sensing task is disseminated throughout the sensor network as an interest for named data as indicated by a (fixed or mobile) control center. This dissemination sets up gradients within the network designed to “draw” events (i.e. data matching the interest). Events start flowing towards the originators of interests along multiple paths. Note that the control center may be some human authorities responsible of taking action upon the realization of a particular, crucial, event.

The family of protocols we present here are based on (and significantly extend) the “sleep-awake” scheme for energy conservation presented in [10]. The main idea of our protocols here is that nodes go through alternating periods of “sleeping” and “awake” modes; each protocol defines a different way to decide when and how each node will go to “sleep” and when it will be “awake”. Since alternating sensors between on and off (active and sleep) states inevitably disrupts the network operation, e.g., coverage and connectivity, our protocols monitor the local conditions of the network and suitably adjust their sleep-awake schedules dynamically according to a given strategy. This dynamic adaptation of sleep-awake schedules is an important new contribution with respect to related ideas in [10] and the relevant literature. We study two main adaptation strategies: a) with respect to the current density in the network, thus coping with physical failures or re-deployment of sensors, power depletion of certain sensors etc. and b) adaptation with respect to the available energy, thus explicitly aiming at balancing the energy load and prolong the system’s lifetime. Our protocols do not use (control) messages that provide means for synchronizing the sensors, but on the contrary, operate on a set of simple rules that are based only on locally sensed conditions.

Implementing and deploying these protocols, we investigate their impact on the performance of dynamic wireless sensor networks that are made up from nodes with heterogeneous resources and operated using the Directed Diffusion protocol [18]. In our attempt to model the dynamic aspects of wireless sensor networks we consider (i) temporary topology changes as sensor nodes try to reduce energy consumption, (ii) permanent disconnections due to power failure of individual sensors and (iii) introduction of new nodes as additional sensors are redeployed while the network is operating. The particular use of adaptive techniques in this heterogeneous setting is another novelty of our work. We indeed show that, using adaptive schemes, redeployment of few resources may be more efficient compared to static deployment of many resources from the start.

We implement our protocols and carry out a large scale simulation and detailed experimental evaluation of various important measures of their performance. In particular we examine (i) the success rate of the Directed Diffusion protocol (i.e. the final delivery percentage of the disseminated data to the human authorities), (ii) the propagation delay of the sensed data to the control center and (iii) the energy dissipated by the sensor devices for the duration of the network operation. The detailed investigation shows the efficiency of our
schemes and highlights the advantages and disadvantages of each approach and its suitability for a certain network and specific task dynamics. Our findings focus on the impact of (i) the network density and (ii) the presence of sensor devices with heterogeneous characteristics, on the performance measures studied.

We note that, to the best of our knowledge, this is the first time that the possibility of one or more re-deployment phases is quantitatively investigated.

1.2 Discussion of Selected Related Work and Comparison

In a recent paper [21], the Resource Oriented Protocol (ROP) is presented that takes specific characteristics of sensors into account and implements a network topology accordingly. The authors model the heterogeneity of the sensor devices according to their energy resources. Their protocol adopts a hierarchical approach, in a way similar to LEACH [16], and entails two phases: topology formation phase and topology update phase. A similar cluster based approach is used in [13], that studies the problem of continuous monitoring in sensor networks with a heterogeneous organization. The authors formulate the energy consumption and study the estimated network lifetime based on parameters related to the sensing field, e.g. size and distance. In contrast to these approaches, we avoid any topology exploration and develop protocols that do not require any knowledge on the topology of the network. We look into systems that achieve global cooperation in a distributed manner by applying a set of simple rules at local level.

In [22], heterogeneous sensor networks are studied by considering two groups of sensor devices with different energy resources and communication capabilities that are dropped over the area of interest using two-dimensional homogeneous Poison point processes. The authors investigate the optimum node densities and initial energy resources, for each group of sensors, so that a minimum network lifetime is guaranteed while ensuring connectivity and coverage of the surveillance area with a high probability. Our model does not limit the heterogeneity of the sensing devices to just two groups, but allows the possibility of having a set of sensor devices with highly diverse energy resources.

The problem of providing network coverage using wireless sensors that alternate between active and sleep states to conserve energy is investigated in [17]. Two types of mechanisms are considered: the random sleep type where each sensor keeps an active-sleep schedule independent of another, and the coordinated sleep type where sensors coordinate with each other in reaching an active-sleep schedule. Both types are studied within the context of providing network coverage. The authors examine the fundamental relationship between the reduction in sensor duty cycle and the required level of redundancy for a fixed performance measure, and explore the design of good sensor sleep schedules. It is shown with either type of sleep schedule the benefit of added redundancy saturates at some point in that the reduction in duty cycles starts to diminish beyond a certain threshold in deployment redundancy. It is also shown that at the expense of extra control overhead, a coordinated sleep schedule is more robust and can achieve higher duty cycle reduction with the same amount
of redundancy compared to a random sleep schedule. To the contrary, our schemes try to avoid the use of control messages and synchronization between sensors.

In [15] similar energy conservation schemes are considered for a different problem (target tracking). The authors consider pre-scheduled independent sleeping schemes and neighborhood cooperative sleeping schemes and evaluate their performance via a simulation study for various network densities. For the same problem, in [20] a random sleep-awake scheme is presented that guarantees k-coverage in sensor networks. The authors relate the need for a particular/minimum number of sensor nodes in order to succeed in target tracking. In their protocol, each sensor decides to be active in a given period with probability \( p \), independently from the other sensors. The authors investigate the probability \( p \) in order to achieve k-coverage of every point in the network. No online/adaptive techniques are used.

For a different approach to balancing the energy dissipation, see [14]. For a combination of energy balance and obstacle avoidance see [2].

Recently, Boukerche et. al [3] propose a novel and efficient energy-aware distributed heuristic, which they refer to as EAD, to build a special rooted broadcast tree with many leaves that is used to facilitate data-centric routing in wireless microsensor networks. EAD algorithm makes no assumption on local network topology, and is based on residual power. It makes use of a neighboring broadcast scheduling and distributed competition among neighboring nodes.

EAD basically computes a tree with many leaves. With the transceivers of all leaf nodes being turned off, the network lifetime can be greatly extended.

For a survey of efficient data propagation protocols, see [5]. For a focused study of energy efficiency aspects, see [4].

2 The Model

Sensor networks are comprised of a vast number of ultra-small homogeneous sensors, which we here call “grain” particles (see also [10, 11]). Each grain particle is a fully-autonomous computing and communication device, characterized mainly by its available power supply (battery) and the energy cost of computation and transmission of data. Such particles (in our model here) do not move. Let \( n \) be the number of smart dust particles spread in an area \( A \).

There is a single point in the network area, which we call the sink \( S \), and represents a control center where data should be propagated to.

The particles are equipped with a set of monitors (sensors) for light, pressure, temperature etc. Each particle has a broadcast (digital radio) beacon mode of fixed transmission range \( R \). Each particle is equipped with a power supply (e.g. battery), that offers a mechanism that provides measurements of its remaining energy supplies. Let \( E_{(i)} \) be the available energy supplies of particle \( i \) at a given time instance. At any given
time, each particle can be in one of four different modes, regarding the energy consumption: (a) transmission of a message, (b) reception of a message, (c) sensing of events and (d) sleeping. During the sleeping mode, particle ceases any communication with the environment, thus it is unable to receive any message or sense an event. In this sense, we assume that the energy consumption of a sleeping particle is negligible.

In our model, for the case of transmitting and receiving a message, we assume that the radio module dissipates an amount of energy proportional to the message’s size. To transmit a $k$-bit message, the radio expends $E_T(k) = \epsilon_{\text{trans}} \cdot k$ and to receive a $k$-bit message, the radio expends $E_R(k) = \epsilon_{\text{recv}} \cdot k$ where $\epsilon_{\text{trans}}, \epsilon_{\text{recv}}$ are constants that depend on the radio module and the transmission range of the particles.

Concluding, there are three different kinds of energy dissipation: (a) $E_T$, the energy dissipation for transmission, (b) $E_R$, the energy dissipation for receiving and (c) $E_{\text{idle}}$, the energy dissipation for idle state. For the idle state, we assume that the energy consumed for the circuity is constant for each time unit and equals $E_{\text{elec}}$ (the time unit is 1 second). We note that in our simulations we explicitly measure the above energy costs; the exact values of $\epsilon_{\text{trans}}, \epsilon_{\text{recv}}$ and $E_{\text{idle}}$ were set to match as close as possible the specifications of the mica mote platform [23].

Finally, we assume that a specific, high-level, application is executed by the particles that form the network. Applications for smart dust networks fall in three major categories [6]: (i) Periodic Sensing (the particles are always monitoring the physical environment and continuously report their sensors’ measurements to the control center $S$), (ii) Event driven (to reduce energy consumption, particles operate in a silent monitoring state and are “programmed” to notify about events) and (iii) Query based (queries can be propagated to the particles arbitrarily by the control center $S$, according to the application and/or user’s will). We model in an abstract way the kind of high-level application by the message generation rate in the network.

Events take place all over the network and messages are generated accordingly. Let $I$ the global message generation rate of the smart dust network, measured in number of messages per time period.

## 3 A Family of Protocols for Power Conservation

### 3.1 The Sleep-Awake Protocol (SWP)

The family of protocols that we present in this section are based on the “Sleep-Awake” protocol (SWP) that first appeared in [10] and was further extended and studied in [24]. According to the SWP protocol, each particle goes though alternating periods of “sleeping” and “awake”. During a sleeping period, the particles cease any communication with the environment, thus the power consumption is assumed to be minimal and practically insignificant, whereas when a particle is awake, it consumes the regular amount of energy.

We assume that the sleeping/awake time periods alternate stochastically independently in each particle
and have durations $T_s$, $T_w$ respectively (this can be easily achieved if during the start-up phase, each particle maintains a timer, which is initially set at a random time point chosen from the sleep-awake time frame $T$).

The sleep-awake scheme is basically achieved by setting a global ratio, which defines the proportion between the durations of the sleep and awake periods. The timer’s expiration marks a switch over to the alternate mode. The duration of $T$ is defined as the sum of the duration of the sleep and awake periods. Therefore, it is ensured that the sleep-awake transitions do not occur simultaneously, but rather in a random manner. Let now $\gamma = \frac{T_s}{T_w}$, i.e. $\gamma$ represents the energy saving specifications of the smart dust particles (a typical value for $\gamma$ may be 10). Then,

**Definition 1** The energy saving specification is: $en = \frac{T_s}{T_s + T_w} = 1 - \frac{1}{1+\gamma}$

Propagation protocols for such energy-restricted systems should at least guarantee that the control center eventually receives the messages that report a crucial event. The success of such protocols depends on the density of sensor devices on the area of deployment $A$ and their distribution, the distribution of sleeping and awake time periods and, of course, the broadcast range $R$.

In [10], some first analytic results on the interplay between these parameters are given. In particular, the analysis conducted focuses on the relation between the maximum sleeping time period and the other parameters that affect the network performance (e.g. particle density, event generation rate), thus allowing to program the sensor network energy saving specifications accordingly. However we note that so far SWP has been studied under the assumption of no collisions (i.e. under an ideal MAC layer protocol). Furthermore, the scheme parameters are static, i.e. do not adapt to network changes.

SWP protocol is using a similar approach to the work of [27], where a new technique called Sparse Topology and Energy Management (STEM) is proposed that aggressively puts nodes to sleep. Interestingly, the analysis and experiments show improvements of nearly two orders of magnitude compared to sensor networks without topology management.

### 3.2 The Density Adaptive Sleep-Awake Protocol (DA-SWP)

Assuming that the particles are random uniformly distributed on the smart dust plane, the particle density can be calculated according to [7] as

$$\mu(R) = \frac{(n \pi R^2)}{A},$$

where $n$ is the total number of particles deployed in the area $A$ and $R$, the radio transmission range (for more details see Sec. 2). Basically, $\mu(R)$ gives the number of particles within the transmission radius of each particle in region $A$. However, since SWP forces each particle to alternate between periods of “sleeping” and “awake”, $\mu(R)$ is in fact an upper bound for the number of particles that are “awake”. Let $\mu_a(R)$ be the number of active particles in the area $A$. Then, since the length of each “sleep” and “awake” period is adjusted by the energy saving specification $en$, it is $\frac{\mu_a(R)}{\mu(R)} \propto en$.\n
7
Clearly if the parameters that affect the performance of propagation protocols (e.g. density of the network, the broadcast range, see Section 3.1) are (more or less) known in advance, the energy saving specification $en$ can be adjusted by the network operator to maximize the energy-efficiency and keep the network functional for as long as possible. For example, in sparse networks (i.e. low $\mu(R)$) the $en$ should be small so that enough particles participate in the propagation of messages. On the other hand, in dense networks (i.e. high $\mu(R)$) a larger number of particles can be set to “sleep” in order to save more energy. In other words, $en$ can be adjusted in a way such that a target $\mu_a(R)$ is achieved given the expected $\mu(R)$.

However, in real environments, measuring the density of the smart dust plane may be a highly non-trivial task (if possible at all), especially if we consider cases where the particles are dropped randomly on the area of interest. Moreover, the network density continuously changes, as the network evolves over time, since (because of the limited power supplies) the power of the sensor devices will (inevitably) be exhausted. It is also possible for particles to stop functioning due to physical damage (i.e. destruction by external factors) or failure on the (low-cost) equipment. Because of this fact, we expect that $\mu(R)$ (and in extend $\mu_a(R)$) will decrease over time, as sensor nodes are forced to disconnect from the network. In this sense, as the network evolves over time, the initial value for $en$ (that will lead to a network with a particular $\mu_a(R)$) will make the network to operate at suboptimal levels.

As discussed in Sec. 1, it is possible to re-deploy additional particles while the network is in operation, to “replace” the malfunctioning particles or due to change in the task dynamics [1]. In this way the network operator can reinstate $\mu(R)$ at the desired levels. Still, the nature of the re-deployment process is such that precise positioning of sensor devices (and thus the “local” densities) can not be performed.

In the light of the above, we want the protocol to be capable of sensing and appropriately handling changes to the local conditions and suitably adjust its local energy saving specification $en$, so that the target $\mu_a(R)$ is approximated. If the values of $T_s, T_w$ remain fixed for the duration of the protocol, any change in the local conditions of the network will not be reflected in the way the energy saving specification is selected. In order to sense the local particles’ density, DA-SWP, builds upon the following subprotocol.

**The Density-Sensing subprotocol ($P_{density}$).** We call $d_{local}$ the number of neighbors a particle senses over a certain area (i.e., the local density). Initially $d_{local} = d_{init}$, where $d_{init}$ is a value set by the protocol implementor.

$P_{density}$ maintains a table where it stores all the sender’s $ids$ encountered along with a time counter indicating the time the message was received. In fact, the subprotocol is continuously inspecting all packets received and updates the local table. For every entry in the table a counter is defined that is initialized to a predefined period of time called $t_{inactive}$ (e.g. $t_{inactive} = 1hr$). Periodically, $P_{density}$ will go through the list of $ids$ and remove those neighbors whose counter has reached zero.

In this way, $P_{density}$ measures the number of neighbors that the particle perceives as active (i.e. $\mu_a(R)$).
Remark that the calculation of $d_{local}$ is performed dynamically and is subject to change over time; in this sense, we will define it as a function of time, i.e. $d_{local}(t)$ and $d_{local}(0) = d_{init}$.

Based on the $P_{density}$ sub-protocol and a particular value of $d_{init}$ set by the protocol implementor to reflect the desired conditions of the network, the local energy saving specification $en$ is adjusted as follows:

$$d_{SW} = T_s \cdot \frac{d_{local}(t)}{d_{init}} - T_s$$

$$T_s = T_s + d_{SW}$$

$$T_w = T_w - d_{SW}$$

This setting allows particles to sleep more (or less) when the detected local density is higher (or lower) than $d_{init}$ specified by the network implementor. Thus, the protocol tries in each adaptation phase to converge to an optimal (with respect to current network conditions) power save scheme. Based on Eq. 3,4, the sum $T_s + T_w$ is constant and remains unchanged by the adaptation. This design decision was made to prevent certain sensors from continuously increase $T_s$ (or $T_w$) due to the locally sensed conditions, leading to situations where the sensors sleep (or are awake) for extremely long period of time.

### 3.3 The Energy Adaptive Sleep-Awake Protocol (EA-SWP)

We now propose an adaptive way to adjust the sleep interval of each node by explicitly taking into account the energy available at the sensor devices. The density adaptive protocol only implicitly considers the energy by monitoring the active number of neighbors; as the network evolves, some nodes will exhaust their power and disconnect, affecting in this way the number of active neighbors.

In contrast to this approach, the Energy Adaptive Sleep-Awake Protocol (EA-SWP) attempts to evenly distribute energy consumption among particles by adjusting the sleep interval of strained particles. Balancing the energy dissipation among the sensors in the network avoids the early energy depletion of certain sensors and thus increases the lifetime of the system by preventing from early network disconnection. In order to detect that a particle consumes energy faster than others, an estimation of the average energy of the nearby particles is required. Essentially, this is achieved by providing an estimate on the energy levels of the node to the neighboring nodes every time a message is transmitted. In this way nodes can keep track of the available energy in their neighborhood. Such knowledge can be gathered by slightly modifying the Density Sensing subprotocol $P_{density}$ defined in Sec. 3.2. More specifically, in order to detect the energy of the neighboring particles, EA-SWP uses the energy-sensing subprotocol ($P_{energy}$) which operates as follows.
The Energy-Sensing subprotocol ($P_{\text{energy}}$). Whenever a message is transmitted, $P_{\text{energy}}$ includes in the header of the message an estimation of the particle’s remaining energy $E_{(i)}$.

$P_{\text{density}}$ maintains a table where it stores all the sender’s id’s encountered along with the energy counter that indicates the remaining energy when the message was transmitted. In fact, the subprotocol is continuously inspecting all packets received and updates the local table. Based on this table of energy counters stored, $P_{\text{energy}}$ calculates $E_{\text{avg}}$, i.e. the average energy of the neighboring particles that a particle senses over a certain area.

Remark that the calculation of $E_{\text{avg}}$ is performed dynamically and is subject to change over time; in this sense, we will define it as a function of time, i.e. $E_{\text{avg}}(t)$ and $E_{\text{avg}}(0) = E_{(i)}$.

Initially, all particles select a random sleep-awake schedule with the duration of $T_{s}$ and $T_{w}$ fixed by the network operator. Then, as the network evolves in time, each particle before switching to the sleep state computes a new value for $T_{s}$ and $T_{w}$ by using the average energy $E_{\text{avg}}$ estimated by $P_{\text{energy}}$.

\[
dSW = T_{s} \cdot \frac{E_{\text{avg}}(t)}{E_{(i)}} - T_{s}
\]

\[
T_{s} = T_{s} - dSW
\]

\[
T_{w} = T_{w} + dSW
\]

This setting allows particles to sleep more when their energy supplies are less than the average in their neighborhood, or forces particles to awake in the opposite case. Note that the sum $T_{s} + T_{w}$ is constant and remains unchanged by the adaptation process.

3.4 Integrated Communication Architecture

A major challenge when designing protocols that will be incorporated in an existing protocol stack, such as the family of sleep-awake protocols presented in this paper, is to integrate these protocols as smooth as possible with the existing ones. Below we examine two different approaches that can be used to integrate our protocols in a generic sensor device’s network stack.

The first approach, presented in figure 1a, encapsulates the sleep-awake protocols in the low levels of the protocol stack, namely the MAC layer. This approach has the advantage that modifications are made to a single place; only one layer of the network stack is modified. Still, the sleep-awake protocols can effectively control the behavior of the above network layers. The fact that the sleep-awake protocols are isolated from the rest of the network stack forbids the interaction with energy aware applications. We refer to this approach as the “strict” sleep-awake implementation.
In the second approach, presented in figure 1b, the sleep-awake protocols are exposed in all levels of the network stack and interaction is permitted between application, sleep-awake protocols and the MAC layer. As suggested in [1], when the different network layers can interact, the performance of the network stack can increase since the lower layers can be tuned in order to best suit the requirements of the application. The downside of this approach is that carefully designed interfaces are required in order to facilitate the interaction without disturbing the functionality of the protocols. We call this approach the “integrated” sleep-awake implementation.

In the next section, we describe the necessary extensions made to the simulation platform so that we can implement the two different approaches. We carry out an extensive simulation study to identify the advantages and disadvantages of the “strict” and “integrated” approaches.

4 Experimental Framework

4.1 Implementation Details

In contrast to previous work of our team (see [8–11, 24]) where we were using the lightweight graph-based LEDA environment to be able to study instances with a very large number of network nodes, in this work we chose to use the Network Simulator (ns-2 version 2.26, [25]), a detailed network simulator that more accurately takes into account the network parameters. The ns-2 simulator provides a quite detailed implementation of the physical and MAC layers and allows detailed measurements of many variables (such as the energy dissipation) in simulations of wireless networks. Implementing a new protocol in the lower levels of the ns-2 network stack, might require a great deal of code modifications. We feel it is interesting to provide a brief description of the implementation details of our protocols.

Starting with the high level sensing application, we implemented it, based on the description in Sec.2,
as a Directed Diffusion application. This application properly initializes Directed Diffusion by issuing the necessary interest publications and data subscriptions (see [18]), and is also responsible for the generation of events during the simulation.

The sleep-awake protocols were implemented as modules of the Energy Model of ns-2. The Energy Model is responsible for book-keeping the nodes’ state (sleep, awake, etc.) as well as the nodes’ energy supplies. The \( P_{\text{density}} \) and \( P_{\text{energy}} \) subprotocols were also incorporated in the Energy Model and used the information already exposed to the Energy Model by the MAC layer. By doing so we kept our protocols isolated from the rest of the simulator and confined modifications in a small portion of the simulator code base.

An additional implementation issue we dealt with was the sleep-awake behavior of the default ns-2 MAC layer. In the default MAC layer, if the particle is in the sleep state, when a packet arrives from a higher network layer, the MAC layer sets the node in the awake state and transmits the packet. This behavior partly matches the “intergrated” approach described in Sec.3.4. We implemented the “strict” approach by creating a modified MAC layer that queues packets when the node is asleep and transmits those packets when the node is put in the awake state again. Since the default implementation of Directed Diffusion is not aware of the sleep-awake schedule of the node, it’s interesting to see how these different MAC layers affect the protocols’ performance.

Finally, we provided some further extensions to ns-2 to implement the phase of re-deploying a number of nodes (i.e. create new nodes) while the simulation is being executed. In the original version of ns-2, although such a scenario can be properly described, the simulation can’t be executed (i.e. the program crashes). The reason is that some functions responsible for initializing the components of the simulator assume that the initialization takes place at simulation time 0.0 while some other components assume that the number of nodes is known at the beginning of the simulation. We had to modify the simulator to work around both problems.

At this point we would like to mention that we plan to make our modifications available to the ns-2 community in order to possibly contribute to the ns-2 simulator so that it can simulate an even larger variety of networks.

4.2 Experimental Setup

In order to evaluate the performance of the proposed protocols an extensive analysis via simulation was conducted.

The energy model presented in Sec. 2 was implemented by the ns-2, the exact values of \( \epsilon_{\text{trans}} \), \( \epsilon_{\text{recv}} \) and \( E_{\text{idle}} \) were set to match as close as possible the specifications of the mica mote platform [23], while the transmission range of each particle were set to \( R = 50m \).

The sensor network is considered as a rectangular area of size \( A = 500m \times 500m \), where a number of
Table 1: Calculated Particle density using Eq. 1, where $A = 500m \times 500m$ and $R = 50m$

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\mu(R)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>9.424</td>
</tr>
<tr>
<td>400</td>
<td>12.566</td>
</tr>
<tr>
<td>500</td>
<td>15.707</td>
</tr>
<tr>
<td>600</td>
<td>18.849</td>
</tr>
</tbody>
</table>

particles are randomly distributed in the area. The number of particles $n$ ranges in $[300, 600]$. Table 1 lists the corresponding density (calculated using Eq. 1) for the values of $n$ used in the experiments. We here note that in [18] the experimental study conducted also considered networks of different sizes and number of particles but with an almost fixed particle density $\mu(R) \approx 9.817$.

The sink is always positioned at point $(0, 0)$. A sensing task is assumed to operate on-top of Directed Diffusion that generates $\lambda = 2$ events per second. Each event is being sensed by a randomly chosen particle and in our simulation $[1000, 2000]$ events are generated. The simulation duration is calculated according to the event rate and is long enough to allow all messages to be generated. Another 15 seconds of simulation time are added to allow the arrival of delayed messages.

The energy available to the particles was set to low levels, so as to approximately be enough, under optimal conditions, for propagating 1000 events but not 2000 events. We made this choice in order to observe if and for how long the sleep-awake family of protocols manages to prolong the lifetime of the propagation process.

4.3 Experimental Results and Discussion

Our experimental evaluation investigates the performance of Directed Diffusion when operated in combination with each of the energy-conservation protocols of Sec. 3. We focus on (i) the success rate achieved by the network, in terms of the fraction of events reported to the control center over all events generated in the network, (ii) the propagation delay for the reported events to reach the control center and (iii) the total energy dissipated by the sensor devices, as a percentage to the initially available energy.

We start by evaluating the impact of the important network parameter of particle density on the performance of the network when propagating 1000 events; essentially we variate the number of deployed particles, based on Tab. 1. In Fig. 2, the right column depicts the measured performance for the “strict” implementation of the energy-conservation protocols (as shown in Fig. 1a) while the left column depicts the corresponding measurements for the “integrated” implementation (as shown in Fig. 1b). We first observe that Directed Diffusion achieves a success rate of 70% for low densities and then slowly degrades to 60% as the network density increases. This behavior suggests that Directed Diffusion cannot take advantage of the redundant
Figure 2: Success Rate, Energy Dissipation and Delay (in sec) of Directed Diffusion in combination with SWP, DA-SWP and EA-SWP for various particle densities ($n \in [300, 600]$), where $s = 5$ sec, $w = 10$ sec $\rightarrow$ $en = 0.33$
nodes to improve the success rate of the network. When we employ the energy-conservation protocols in the network, this performance aspect is improved; the performance of Directed Diffusion increases as more sensor devices are deployed in the network and in some cases (e.g. SWP, EA-SWP in Fig. 2ai) even achieves higher success rates than 80%.

Clearly, success rate is an important performance metric, still it must be considered in combination with the propagation delay. Time efficiency is also an important issue in sensor networks. In fact, Fig. 2aii and Fig. 2bii demonstrate how these two aspects are correlated; forcing particles to “sleep” causes events to travel longer distances until the control center is found. This trade off between success rate and propagation delay is noticeable for all energy-conservation protocols in the “strict” implementation (see Fig. 2aii) and even for the case of SWP, EA-SWP in the “integrated” implementation setting. Similar conclusions can be drawn from Fig. 2aiii and Fig. 2biii. The energy-conservation protocols reduce the energy dissipation and improve the lifetime of the network, i.e. the network can disseminate a higher number of events. Note that without using a sleep-awake protocol, Directed Diffusion consumes all energy in the network and thus the curve is drawn over the upper frame of figures 2aiii and 2biii.

We now consider the effect of the energy saving specification as a parameter to the performance of the network. In this set of experiments we modify the initial en set to the devices by adjusting \( T_s \). This setting is maintained for the duration of the experiments by SWP while DA-SWP and EA-SWP can modify it as the network evolves in time. The results of the second set is depicted in Fig. 3 for both implementation approaches and for the three performance metrics considered.

We first observe that for all protocols, when increasing the sleep duration the achieved success rate drops, while the energy consumption is reduced. Interestingly, the DA-SWP protocol seems to follow a different pattern of behavior, the density sensing protocol detects the low number of active neighbors and attempts to adapt by reducing the sleep duration of the particles. Thus, it manages to keep the success rate in acceptable levels as well as the energy consumption in slightly increasing rate. Furthermore, as in the previous set of experiments, it looks like the “integrated” approach achieves lower propagation delays as the sleeping period of the nodes is disrupted by Directed Diffusion. This is also reflected in Fig. 3iii, where the disruption of the sleeping periods clearly increases the total energy dissipated.

In the third set of experiments we consider the possibility of re-deploying a number of particles, for propagating 2000 events, as the network is in operation. We consider two different scenarios. In the first scenario we initially deploy 300 particles at time \( t = 0sec \) and then re-deploy an extra 300 particles at time \( t' = 400sec \). In the second scenario we initially deploy 300 particles at time \( t = 0sec \) and then carry our 2 re-deployment phases of 150 particles each at times \( t' = 300sec \) and \( t'' = 600sec \). We compare the results of these two scenarios with the cases of having only 300 and 600 particles for the whole duration of the protocol execution.
Figure 3: Success Rate, Energy Dissipation and Delay (in sec) of Directed Diffusion in combination with SWP, DA-SWP and EA-SWP for different sleep durations \(s \in [1, 10]\) with \(w = 10 \text{sec} \rightarrow \text{en} = (0.09, 0.5)\)
We here note that the possibility of a re-deployment phase could in fact be implemented with a single deployment, where a given number of particles is set to a deep-sleeping mode for a long period of time. This may have the same effect as dropping the new particles at the end of that period.

To evaluate the performance of the various protocol combinations we study the success rate and energy dissipation as the network evolves over time. In Fig. 4 we include the graphics for the “integrated” implementation of the protocols. We here note that similar results hold for the corresponding cases of “strict” implementation, however these graphics were omitted due to page size limitations. The average propagation delay achieved by each protocol combination is depicted in Tab. 2.

The main result of this set of experiments is that all protocols benefit from the use of a re-deployment phase compared to the static deployment of the total number of devices from the start. This can be explained as follows. Based on Fig. 4a, deploying 300 particles the network achieves high success rates, while throwing additional (redundant) nodes from the beginning does not improve the performance of the network, as shown in Fig. 4d. In fact, all particles spend their energy from the very start of the network operation and more or less their power is exhausted simultaneously. For both cases (i.e. where $n = 300$ and $n = 600$), we observe that the performance of the network starts to deteriorate, i.e. stops disseminating data to the control center, about at time $t = 400sec$. However, by delaying the deployment of the 300 nodes for a period of 400sec, we manage to somehow replace the strained nodes with the newly deployed ones. By doing so, the network lifetime is extended. As a result, the final achieved success rate in the case of redeployment (see Fig. 4b) is higher than in the case of having just 300 particles (Fig. 4a) or even 600 particles (Fig. 4d) from the beginning.

We extend this idea by evaluating the possibility of having two, smaller, re-deployment phases. We redeploy a smaller number of particles $n' = 150$ at time $t = 300sec$, when the network is still operating at high success rates (see Fig. 4a). Then we deploy the remaining $n'' = 150$ particles at time $t = 600sec$. In this case, although the overall success rate is almost the same, we observe that the deterioration is milder. This observation is clearly seen in the energy dissipation, where the two re-deployment phases take place before the network has dissipate all the available energy.

Regarding the propagation delay experienced by the network, by closely examining Tab. 2, we can safely conclude that Directed Diffusion does not suffer any additional delay. This conclusion is in agreement with the results of the first set of experiments, see Fig. 2ii, where the propagation delay of Directed Diffusion remains constant regardless of the network particles density.

In this set of experiment we observe that even though the energy-conservation protocols do not manage to considerably improve the success rate of Directed Diffusion it seems that they do manage to reduce the overall energy dissipation. In this sense, we believe that the network lifetime is extended and for the case of more than 2000 events, the energy-conservation protocols could surpass the simple Directed Diffusion.
Figure 4: Success Rate and Energy Dissipation of Directed Diffusion in combination with SWP, DA-SWP and EA-SWP over simulated time (sec) for various particle densities (n ∈ [300, 600]) and up to 2 redeployment phases where s = 5sec, w = 10sec → en = 0.33
Table 2: Propagation Delay (in sec) of Directed Diffusion in combination with SWP, DA-SWP and EA-SWP for different particle densities ($n \in 300, 600$) and up to 2 re-deployment phases ($s = 5sec, w = 10sec -> en = 0.33$)

<table>
<thead>
<tr>
<th>Initial deployment</th>
<th>1$^{st}$ Re-deployment</th>
<th>2$^{nd}$ Re-deployment</th>
<th>No SWP</th>
<th>SWP</th>
<th>DA-SWP</th>
<th>EA-SWP</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>0</td>
<td>0</td>
<td>0.555</td>
<td>1.415</td>
<td>1.966</td>
<td>1.427</td>
</tr>
<tr>
<td>600</td>
<td>0</td>
<td>0</td>
<td>0.532</td>
<td>1.315</td>
<td>4.380</td>
<td>1.586</td>
</tr>
<tr>
<td>300</td>
<td>300</td>
<td>0</td>
<td>0.567</td>
<td>1.411</td>
<td>2.925</td>
<td>2.191</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>150</td>
<td>0.551</td>
<td>1.384</td>
<td>3.298</td>
<td>2.709</td>
</tr>
</tbody>
</table>

We here note that to the best of our knowledge this is the first time that the possibility of one or more re-deployment phases is investigated in an experimental evaluation. We plan to further experiment with the possibility of re-deployment phases, in terms of number of extra deployment phases and the time of occurrence.

In our final set of experiments we investigate the performance of Directed Diffusion in combination with our family of protocols in heterogeneous sensor networks. In particular we divide the sensor devices into two groups (A and B) based on the initial energy resources: sensor nodes that belong in Group A start with 0.5J energy while those of Group B start with 1.0J energy. Fig. 5b depicts the success rate and energy dissipation of the network that is made up from $n_A = 400$ particles (belonging to group A) and $n_B = 100$ particles (belonging to group B). The total initial energy available is 300J. By keeping the total initial energy at the same level, we repeat the experiment by using a different composition for group A and B. Fig. 5b depicts the success rate and energy dissipation of the network that is made up from $n_A = 200$ particles (belonging to group A) and $n_B = 200$ particles (belonging to group B).

A possible way to compare in a fair way the performance of the heterogeneous network with the homogeneous networks considered in the previous experiments is to use the same total initial energy, i.e. $\sum^n_i E_{(i)}$ in each case. Fig. 5d shows the same performance measures for a homogeneous network with total initial energy 300J. Another way to compare the performance of the heterogeneous network with the homogeneous networks is to use the particle density. Fig. 5a shows the same performance measures for a homogeneous network made up from $n = 500$ nodes.

We can observe that all protocols benefit from the use of high power particles that can withstand more load. Using the particles density as the comparison criteria between heterogeneous and homogeneous networks (with $n = 500$), only a slight improvement on the success rate is observed. However based on the total initial energy comparison criteria (i.e. with $n = 600$) it is evident that the heterogeneous network outperforms the homogeneous network. Interestingly, for the case where $n_A = n_B = 200$ the results are significantly better. More importantly, the energy-conservation protocols seem to achieve an almost stable behavior in terms of success rate for the whole duration of network operation.
Figure 5: Success Rate and Energy Dissipation of Directed Diffusion in combination with SWP, DA-SWP and EA-SWP over simulated time (sec) for various particle densities ($n \in [300, 600]$) and for up to 2 groups of initial energy resources where $s = 5\text{sec}$, $w = 10\text{sec} \rightarrow en = 0.33$
We here discuss the peculiar behavior of EA-SWP that achieves very low success rates when \( n_A = 400 \), \( n_B = 100 \) but performs better than all other protocols when \( n_A = 200 \), \( n_B = 200 \), can be explained by the adaptation function. In the first case, the nodes of group A detect the high energy nodes of group B and sleep excessively long but the nodes in group B are too few to handle the load and are eventually depleted. The nodes in group A fail to respond quickly to that change in the neighbor’s energy because group B has very few members. However, in the second case, nodes in group B can handle the load and when they are depleted the nodes in group A properly adapt their sleep-awake schedule and take over the propagation task.

Regarding the propagation delay experienced by the network, by closely examining Tab. 3, again we observe that Directed Diffusion does not suffer any additional delay as in the case of having one or more re-deployment phases. The propagation delay of Directed Diffusion remains constant regardless of the network particles density and their heterogeneous composition.

Finally, based on the energy dissipation graphics, we believe that by employing the energy-conservation protocols the network lifetime can be extended. This is based on the observation that in both Fig. 5b and Fig. 5c the total energy dissipation is less than 100%. In this sense, if we extend the total number of events to more than 2000 we could benefit further from the remaining energy at the sensor devices.

5 Closing Remarks

We have proposed, implemented and evaluated a family of energy conservation protocols for efficient data propagation in wireless sensor networks. Our protocols are adaptive, in the sense that they locally monitor the network conditions and accordingly adjust towards optimal operation choices. This dynamic feature is particularly beneficial in heterogeneous settings and in cases of re-deployment of sensor devices in the network area.

We implement our protocols and evaluate their performance through a detailed simulation study using our extended version of ns-2. In particular we combine our schemes with Directed Diffusion, a known

<table>
<thead>
<tr>
<th>Group 1</th>
<th>Group 2</th>
<th>No SWP</th>
<th>SWP</th>
<th>DA-SWP</th>
<th>EA-SWP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(( E_{(i)} = 0.5J ))</td>
<td>(( E_{(i)} = 1J ))</td>
<td>0</td>
<td>0.543</td>
<td>1.407</td>
<td>2.869</td>
</tr>
<tr>
<td>400</td>
<td>0</td>
<td>0.504</td>
<td>1.373</td>
<td>4.028</td>
<td>1.670</td>
</tr>
<tr>
<td>500</td>
<td>0</td>
<td>0.532</td>
<td>1.315</td>
<td>4.380</td>
<td>1.586</td>
</tr>
<tr>
<td>600</td>
<td>0</td>
<td>0.478</td>
<td>1.325</td>
<td>3.495</td>
<td>3.634</td>
</tr>
<tr>
<td>400</td>
<td>200</td>
<td>0.536</td>
<td>1.410</td>
<td>3.326</td>
<td>2.450</td>
</tr>
</tbody>
</table>

Table 3: Propagation Delay (in sec) of Directed Diffusion in combination with SWP, DA-SWP and EA-SWP for different particle densities \( (n \in [400, 600]) \) with different initial energy \( (s = 5sec, w = 10sec \rightarrow en = 0.33) \)
communication paradigm, and evaluate their performance under two different levels of integration with the MAC layer. The simulation findings demonstrate significant gains and good trade-offs in terms of delivery success, delay and energy dissipation. Specifically, the proposed protocols effectively prolong network lifetime and increase overall achieved delivery rate by reducing the energy dissipation. This is achieved at the cost of a relatively small increase in propagation delay. The findings are similar in both levels of MAC integration, a fact that can be attributed to Directed Diffusion being sleep-aware unaware.

We here note that to the best of our knowledge this is the first time that the possibility of one or more re-deployment phases is investigated in an experimental evaluation. We plan to further experiment with the possibility of re-deployment phases, in terms of number of extra deployment phases and the time of occurrence. Furthermore, we plan to investigate “deeply” integrated architectures of the network stack, that will enable a close cooperation between the various network layers and facilitate the adaptation to the application requirements.

References


