Power-Aware Data Dissemination Protocols in Wireless Sensor Networks

Sotiris Nikoletseas
Department of Computer Engineering and Informatics
University of Patras, and Computer Technology Institute (CTI), Greece
nikole@cti.gr

Abstract

Recent rapid technological developments have led to the development of tiny, low-power, low-cost sensors. Such devices integrate sensing, limited data processing and communication capabilities.The effective distributed collaboration of large numbers of such devices can lead to the efficient accomplishment of large sensing tasks.

This talk focuses on several aspects of energy efficiency. Two protocols for data propagation are studied: the first creates probabilistically optimized redundant data transmissions to combine energy efficiency with fault tolerance, while the second guarantees (in a probabilistic way) the same per sensor energy dissipation, towards balancing the energy load and prolong the lifetime of the network.

A third protocol (in fact a power saving scheme) is also presented, that directly and adaptively affects power dissipation at each sensor. This "lower level" scheme can be combined with data propagation protocols to further improve energy efficiency.

1 Introduction

Recent dramatic developments in micro-electro-mechanical (MEMS) systems, wireless communications and digital electronics have already led to the development of small in size, low-power, low-cost sensor devices. Current devices have a size at the cubic centimeter scale, a CPU running at 4 MHz, some memory and a wireless communication capability at a 4Kbps rate. Also, they are equipped with a small but effective operating system and are able to switch between "sleeping" and "awake" modes to save energy. Pioneering groups (like the "Smart Dust" Project at Berkeley, the "Wireless Integrated Network Sensors" Project at UCLA and the "Ultra low Wireless Sensor" Project at MIT) pursue further important goals, like a total volume of a few cubic millimeters and extremely low energy consumption, by using alternative technologies, based on radio frequency (RF) or optical (laser) transmission.

There is a wide range of applications based on the possible use of various sensor types (i.e. thermal, visual, seismic, acoustic, radar, magnetic, etc.) to monitor a wide variety of conditions (e.g. temperature, object presence and movement, humidity, pressure, noise levels etc.). Thus, sensor networks can be used for continuous sensing, event detection, location sensing as well as micro-sensing. For an excellent survey see [2].

Features including the huge number of devices involved, the severe power, computational and memory limitations, the dense deployment and frequent failures, pose *new design, analysis and implementation challenges* which are different not only with respect to distributed computing but also to ad-hoc networking. Because of these rather unique characteristics, efficient and robust distributed protocols and algorithms should exhibit the following properties: *a) Scalability:* Distributed protocols should be highly scalable, i.e. should operate efficiently in extremely large networks, *b) Efficiency:* Protocols for sensor networks should be efficient, with respect to energy and time, *c) Fault-tolerance:* The network should continue its proper operation for as long as possible despite failures of certain nodes.

One of the most crucial goals in designing efficient protocols is minimizing energy consumption, in various ways: (a) minimizing the total energy spent in the network (b) minimizing the number (or the range) of data transmissions (c) combining energy efficiency and fault-tolerance, (d) maximizing the number of "alive" sensors over time, prolonging the system's lifetime, and (e) balancing the energy dissipation among sensors to avoid the early depletion of certain sensors and thus the breakdown of the network.

We note that it is very difficult to achieve all the above goals at the same time and trade-offs exist between some of these goals. Furthermore, the importance and priority of each of these goals may depend on the particular application. Thus, it is important to have a variety of protocols (and hybrid combinations of protocols), each of which may

1-4244-0054-6/06/$20.00 ©2006 IEEE

focus at some of the energy efficiency goals (while still performing well with respect to the other goals).

We present three energy efficient protocols from our research: *a) the Probabilistic Forwarding Protocol (PFR)*, that creates redundant data transmissions that are probabilistically optimized, to trade-off energy efficiency with fault-tolerance. *b) the Energy Balanced Protocol (EBP)*, that focuses on guaranteeing the same per sensor energy dissipation, in order to prolong the lifetime of the network. *c) the Adaptive Power Conservation Protocol (APCP)*, that aggresively affects power consumption of each sensor, in an adaptive manner based on implicitly and locally monitoring network conditions. APCP can be combined with higher level protocols, like PFR, EBP and Directed Diffusion.

We believe that a complementary use of rigorous analysis and large scale simulations is needed. Asymptotic analysis may lead to provable efficiency and robustness guarantees towards the scalability of protocols for networks of extremely large size. On the other hand, simulation allows to investigate the detailed effect of a great number of technical specifications of real devices.

In addition to algorithmic design, several other aspects of distributed computing are also very important, such as high level abstractions and systematic design methodologies and tools. Topics including programming models, abstractions for modular design task allocation and reconfiguration, languages and lightweight operating systems, scalable architectures and middleware should be investigated. For an excellent discussion of programming abstractions for real-time communication, see [1].

2 Probalistic Data Forwarding

To combine energy efficiency and fault tolerance, particularly in faulty and sparse networks, the Probabilistic Forwarding Protocol (PFR) has been introduced by Chatzigiannakis, Dimitriou, Nikoletseas and Spirakis in [4], where it is assumed that sensors are *randomly deployed* in a given area. Such a placement may occur e.g. when throwing sensors from an airplane. *As a special case*, they consider the network being a lattice (or grid) deployment of sensors. This grid placement is motivated by certain applications, where it is possible to have a pre-deployed sensor network, where sensors are put (possibly by a human or a robot) in a way that they form a *2-dimensional lattice*.

We assume that each sensor has the following abilities: (i) it can estimate the direction of a received transmission (e.g. via using a direction-sensing antenna), (ii) it can estimate the distance to a nearby particle (e.g. via signal attenuation), (iii) it knows the direction towards the sink S. (iv) all particles have a common coordinates system. Notice that GPS information is not needed. Also, there is no need to know the global structure of the network.

The basic idea of the protocol lies in probabilistically favoring transmissions towards the sink within a *thin zone* of particles around the line connecting the particle sensing the event $\mathcal{E}$ and the sink. Note that transmission along this line is energy optimal. The protocol evolves in two phases:

Phase 1: The "Front" Creation Phase. Initially the protocol builds (by using a limited, in terms of rounds, flooding) a sufficiently large "front" of particles, to guarantee the survivability of the data propagation process. During this phase, each particle deterministically forwards received data towards the sink.

Phase 2: The Probabilistic Forwarding Phase. During this phase, each particle P possessing the information under propagation, calculates an angle ϕ by calling the subprotocol "ϕ-calculation" (see description below) and broadcasts $info(\mathcal{E})$ to all its neighbors with probability $\mathbb{P}_{fwd}$ (or it does not propagate any data with probability $1 - \mathbb{P}_{fwd}$) defined as follows:

$$\mathbb{P}_{fwd} = \begin{cases} 1 & \text{if } \phi \geq \phi_{threshold} \\ \frac{\phi}{\pi} & \text{otherwise} \end{cases}$$

where ϕ is the angle defined by the line EP and the line PS (see Fig. 1) and $\phi_{threshold} = 134^o$.

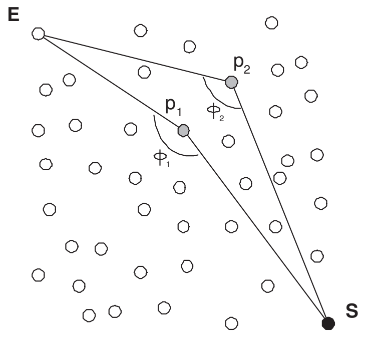

Figure 1. Angle used in PFR

Properties of PFR: Consider a partition of the network area into small squares of a fictitious grid G. By randomly deploying a sufficiently large number of sensors, we can show, by occupancy arguments, that with very high probability (tending to 1) all squares get particles. [4] conditions all the analysis on this event, call it F.

The Correctness of PFR: wlog. we assume each square of the fictitious lattice G to have side length 1. In [4] the correctness of PFR is proved, by using a geometric analysis. Consider any square Σ intersecting the ES line. By the occupancy argument above, there is w.h.p. a particle in this square. Clearly, the worst case is when it is located in one of the corners of Σ. By geometric calculations, [4] finally proves that the angle ϕ of this particle is $\phi > 134^o$. But the initial square (i.e. that containing E) always broadcasts and any intermediate intersecting square will be notified (by induction) and thus broadcast. Thus the sink will be reached.

Lemma 2.1 ([4]) PFR *succeeds with probability 1* in sending the information from E to S given the event F.

The Energy Efficiency of PFR: The analysis of the energy efficiency of PFR considers particles that are active but are as far as possible from ES. [4] estimates an upper bound on the number of particles in an $n \times n$ (i.e. $N = n \times n$) lattice. If k is this number then $r = \frac{k}{n^2}$ ($0 < r \leq 1$) is the "energy efficiency ratio" of PFR.

In [4] the authors consider the area around the ES line, whose particles participate in the propagation process. The number of active particles is thus, roughly speaking, captured by the size of this area, which in turn is equal to $|ES|$ times the maximum distance from $|ES|$ (where maximum is over all active particles).

This maximum distance is clearly a random variable. To calculate the expectation and variance of this variable, the authors in [4] basically "upper bound" the stochastic process of the distance from ES by a random walk on the line, and subsequently "upper bound" this random walk by a well-known stochastic process (i.e. the "discouraged arrivals" birth and death Markovian process). Thus:

Theorem 2.2 ([4]) The energy efficiency of the PFR protocol is $\Theta\left(\left(\frac{n_0}{n}\right)^2\right)$ where $n_0 = |ES|$ and $n = \sqrt{N}$, where N is the number of particles in the network. For $n_0 = |ES| = o(n)$, this is $o(1)$.

The Robustness of PFR: To prove the following robustness result, the authors in [4] consider particles "very near" to the ES line. Clearly, such particles have large ϕ-angles (i.e. $\phi > 134^o$). Thus, even in the case that some of these particles are not operating, the probability that none of those operating transmits (during the probabilistic phase 2) is very small. Thus, [4] proves the following.

Lemma 2.3 ([4]) PFR manages to propagate the crucial data across lines parallel to ES, and of constant distance, with *fixed* nonzero probability (not depending on n, $|ES|$).

3 Energy Balance

Most data propagation techniques (including PFR) do not *explicitly* take care of the possible overuse of certain sensors in the network. Eg, in multihop transmissions towards the sink, the sensors lying closer to the sink tend to be utilized exhaustively (since all data passes through them). Thus, these sensors may die out very early, resulting to network collapse although there may be significant amounts of energy in the other sensors. Similarly, in clustering techniques the cluster-heads that are located far away with respect to the sink, tend to spend a lot of energy.

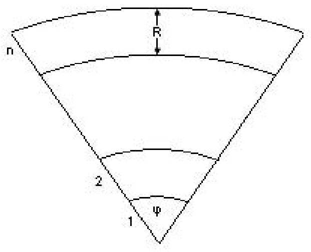

Figure 2. Sensor Network with n ring sectors, angle ϕ and ring "width" R

In this section, we present a protocol trying to balance energy dissipation among the sensors in the network: the EBP (Energy Balance) protocol, introduced in [6], that probabilistically chooses between either propagating data one hop towards the sink or sending directly to the sink. The first choice is more energy efficient, while the latter bypasses the critical (close to the sink) sectors. The appropriate probability for each choice in order to achieve energy balance is calculated in [6].

The sensors are spread in the network area randomly uniformly so their number in a certain area is proportional to the area's size. Sensors can be aware of the direction (and position) of the sink, as well as of their distance to the sink. We assume the transmission range of sensors can vary with time. Events occur at random uniform positions.

We virtually "cover" the network area by a cycle sector of angle ϕ (see Fig. 2). The cycle sector is divided into n ring sectors or "*slices*". The first slice has radius R (i.e. the sensors' transmission range). Slice i ($2 \leq i \leq n$) is defined by two cycles sectors, one of radius $i \cdot R$ and the other of radius $(i-1) \cdot R$. Taking a sufficiently large angle ϕ and/or by taking multiple sectors, we can cover the whole area.

Definition 1 The area between two consecutive cycle sectors is called *a ring sector* (or "slice"). Let T_i ($1 \leq i \leq n$) be the i-th ring sector. Let S_i be the area size of T_i.

We wish to solve the "*energy balanced data propagation problem*", i.e. to propagate data to the sink in such a way that the "average" energy dissipation in each sensor is at each time the same. The average energy dissipation per sensor is the fraction of the total energy spent by sensors in a ring sector over the number of sensors in that sector.

Randomization is used to achieve some “load balancing” by evenly spreading the “load” (energy dissipation). In particular, on ring sector T_i each event is propagated to T_{i-1} (i.e. the “*next*” sector towards the sink) with probability p_i, while with probability $1 - p_i$ it is propagated directly to the sink S. Each message in T_i is handled stochastically independently of the other events’ messages.

The choice of probability p_i for T_i is made so as the average energy consumption per area unit (and thus per sensor) is the same for the whole network. There is a trade-off in choosing p_i: if p_i increases then transmissions tend to happen locally, thus energy consumption is low, however sensors closer to the sink tend to be overused since all data passes through them. On the other hand, if p_i decreases, there are distant transmissions (thus a lot of energy is consumed) however closer to sink particles are bypassed.

We aim at calculating p_i for each i in order to ensure the energy balance property. Let us consider sector T_i.

Definition 2 An area T_i *“handles”* an event generated in ring sector j if either the message was generated in the area T_i (i.e. $j = i$) or the message was propagated to T_i from the ring sector T_{i+1}. Let h_i be the number of the messages that are “handled” by the area T_i.

Definition 3 Let ϵ_{ij} a random variable which measures the energy that dissipates the sector T_i so as to handle the message j. For ϵ_{ij} we have that:

$$\epsilon_{ij} = \begin{cases} cR^2 & \text{with probability } p_i \\ c(iR)^2 & \text{with probability } 1 - p_i \end{cases}$$

where cR^2 is the energy dissipation for sending a message j from T_i to its adjacent ring sector T_{i-1} and c is a constant.

Thus, the expected energy dissipation in sector i for handling a message is

$$E[\epsilon_{i,j}] = cR^2 \cdot [i^2 - p_i(i^2 - 1)] \tag{1}$$

Definition 4 Let $\mathcal{E}_i$ the *total energy* spent by sensors in T_i.

$$\mathcal{E}_i = \sum_{j=1}^{h_i} \epsilon_{ij} \tag{2}$$

Energy balance is defined as follows:

Definition 5 The network is energy balanced when the average per sensor energy dissipation is the same for all sectors:

$$\frac{E[\mathcal{E}_i]}{S_i} = \frac{E[\mathcal{E}_j]}{S_j} \quad i, j = 1, \ldots, n \tag{3}$$

Definition 6 Let g_i be the number of the messages that are *generated* in the area T_i. Let f_i be the number of the messages that are *forwarded to* the area T_i.

We notice the following important relation:

$$h_i = g_i + f_i \tag{4}$$

By linearity of expectation, we get:

Lemma 3.1 $E[h_i] = E[g_i] + E[f_i]$

We establish a relationship between $E[f_i]$ and $E[h_{i+1}]$.

Lemma 3.2 $E[f_i] = p_{i+1} \cdot E[h_{i+1}]$

In what follows, we guarantee the above balance property, requiring a certain recurrence relation to hold. This recurrence basically relates 3 successive terms of the $E[f_i]$ sequence (the $E[g_i]$ terms depend only on i and on input parameters).

Theorem 3.3 To achieve energy balance in the network, the following recurrency equation should hold:

$$a_{i+1}E[f_{i+1}] - (d_i + a_i)E[f_i] + d_{i-1}E[f_{i-1}] = \\ = a_iE[g_i] - a_{i+1}E[g_{i+1}]$$

where

$$a_i = \tfrac{i^2}{2i-1} \qquad d_i = \tfrac{(i+1)^2-1}{2i+1}$$

Solving the above recurrency we get:

$$E[f_i] = -\textstyle\sum_{k=1}^{n-i} \frac{\prod_{j=k}^{n-i+1} a_{n-j}}{\prod_{j=k}^{n-i} d_{n-j}} \cdot \\ \cdot \left(\textstyle\sum_{j=1}^{n-k}(a_jE[g_j] - a_{j+1}E[g_{j+1}]) + a_1 \cdot E[f_1]\right)$$

where $\prod_i^{i-1} a_i = 1$. Now, the calculation of p_i is easy.

Theorem 3.4 The energy balance property is achieved if any ring sector (say T_i) propagates each message it handles with probability p_i to the next ring sector, T_{i-1}, and with probability $1 - p_i$ it propagates the message directly to the sink. The value of each p_i is given by the following relation

$$p_i = \frac{E[f_{i-1}]}{E[g_i] + E[f_i]}$$

The exact derivation of p_i’s can be easily performed by the sensors carrying out very simple calculations. Under specific assumptions (that we discuss and motivate in [6]) we can make the calculation of probabilities p_i simpler.

Theorem 3.5 If $E[f_i] \simeq E[f_{i-1}]$, $3 \leq i \leq n$, then the one-hop forwarding probability, guaranteeing energy balance, is

$$p_i = 1 - \frac{3x}{(i+1)(i-1)}$$

where $p_2 = x \in (0, 1)$ a free parameter and $p_1 = 0$.

4 Adaptive Energy Saving

The Protocol: Data propagation protocols (like Directed Diffusion, PFR and EBP) try to minimize energy dissipation at a high level by affecting the way transmissions happen. Explicit power saving schemes operate at a "lower" level and can be combined with higher layer distributed protocols like the ones for data propagation. The Adaptive Power Conservation Protocol (APCP, proposed in [5]) uses a power-switch mechanism: each device goes through alternating periods of "sleeping" and "awake". During a sleeping period, the devices cease any communication with the environment, thus the power consumption is assumed to be minimal and practically insignificant, whereas when a device is awake, it consumes the regular (non-trivial) amount of energy. The sleeping/awake time periods alternate in each device and have durations T_s, T_w respectively. This is achieved by using a simple timer in each device, which is initially set at a random time point chosen from the sleep-awake time frame, i.e. $T = T_s + T_w$. The timer's expiration marks a switch over to the alternate mode. Let now $en = \frac{T_s}{T_s+T_w}$ be the energy saving specification, measuring how drastic the power saving is, a global ratio of the proportion between the durations of the sleep and awake periods. A different power saving strategy towards optimal sleep-awake schedules is proposed in [3].

Clearly, if the parameters that affect the performance of propagation protocols (e.g. density of the network, the broadcast range) are known in advance, the energy saving specification en can be optimally adjusted by the network operator to maximize the energy-efficiency and keep the network functional for as long as possible. However, in real environments, measuring the density of the network may be a non-trivial task (if possible at all), especially in cases where the devices are dropped randomly in the area of interest. Moreover, the network density continuously changes, as the network evolves over time, since the power of the sensors will be exhausted. It is also possible for devices to stop functioning due to physical damage (i.e. destruction by external factors) or failure of the (low-cost) equipment. Because of this dynamic nature of sensor networks, we expect that density $\mu(R)$ will decrease over time, as sensors disconnect from the network. In this sense, as the network evolves over time, the initial value for en will make the network operate at suboptimal levels. Moreover, it is possible to *redeploy additional devices* while the network is in operation, in order to "replace" the malfunctioning devices or due to change in the task dynamics [2]. In this way the network operator can reinstate density $\mu(R)$ at the desired levels. Still, the nature of the *redeployment* process is such that precise positioning of sensor devices (and thus the "local" densities) can not be achieved.

Furthermore, some nodes will eventually exhaust their power and disconnect, affecting in this way the number of active neighbors. To this end, we are interested in evenly distributing energy consumption among devices by adjusting the sleep interval of strained devices. In order to detect that a device consumes energy faster than others, an estimation of the average energy of the nearby devices is required. This is achieved by providing an estimate of 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.

Finally, we want our protocol to be capable of sensing and handling changes to the local conditions and to adjust its local energy saving specification en. In order to sense the local devices' density, and the neighboring nodes's available energy, we build upon the following subprotocols.

The Density-Sensing subprotocol ($P_{density}$): We call d_{local} the number of neighbors a device senses over a certain area (i.e. the local density). Initially $d_{local} = d_{init}$, where d_{init} is a value set by the operator. $P_{density}$ maintains a table where it stores all the senders' 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. $P_{density}$ measures the number of neighbors that the device perceives as active (i.e. $\mu_a(R)$).

The Energy-Sensing subprotocol (P_{energy}): Whenever a message is transmitted, P_{energy} includes in each message an estimation of the device's remaining energy $E_{(i)}$. $P_{density}$ 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. P_{energy} calculates E_{avg}, i.e. the average energy of the neighboring devices that a device senses over a certain area.

Initially, all devices select a random sleep-awake schedule with the duration of $T_s(0)$ and $T_w(0)$ fixed by the network operator. Then, each device (before switching to the sleep state) computes a new value for T_s and T_w based on the information gathered by $P_{density}$ and P_{energy} and the particular value of d_{init} (set by the protocol implementor, reflecting the desired conditions of the network), as follows:

$$dSW_d = T_s(0) \cdot \frac{d_{local}(t)}{d_{init}} - T_s(0) \tag{5}$$

$$dSW_e = T_s(0) \cdot \frac{E_{avg}(t)}{E_{(i)}} - T_s(0) \tag{6}$$

$$T_s(t) = T_s(t-1) + \alpha \cdot dSW_d - \beta \cdot dSW_e \tag{7}$$

$$T_w(t) = T_w(t-1) - \alpha \cdot dSW_d + \beta \cdot dSW_e \tag{8}$$

In Eq. 5 and Eq. 6 the contribution of each sub-protocol to the adaptation process is calculated. When the measured density drops, the sleep period should decrease (to maintain network connectivity), while when it increases, the sleep period should increase as well. On the other hand, when the measured energy drops, the sleep period should increase (to save energy and achieve energy balance), while when the energy increases, the sleep period should decrease.

In Eq. 7 and Eq. 8 these values are combined in order to adjust the sleep and awake interval of the node. Due to the information gathered by the two sub-protocols, each responds best to a particular type of network condition and function complementary to each other. By combining both sub-protocols we expect to achieve significant gains in any network setting. The values α and β are set by the network operator and provide fine-grained control over the adaptation process. By setting $\alpha = 0$, the energy-sensing component dominates the adaptation process, while by setting $\beta = 0$, the density-sensing component becomes dominant. Our setting allows devices to sleep more (less) when the detected local density is higher (lower) than d_{init} specified by the operator. Also, our protocol puts the devices to sleep more when "local" energy supplies decrease, towards saving energy and achieving balanced energy dissipation among the sensors. Thus, the protocol tries to converge to an optimal (with respect to current network conditions) power save scheme.

Due to Eq. 7,8, the sum $T_s(t) + T_w(t)$ 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.

In order to evaluate our protocols we conduct an extensive experimental study using our extended version of **ns-2** as the simulation platform. We implemented and evaluated APCP with both stand-alone and integrated with APCP versions of Directed Diffusion and PFR in several settings. The sensor network is deployed in a rectangular area of $400 \times 400m$ where $n \in [300, 700]$ nodes are dropped randomly, the transmission range of the sensor devices is set to $50m$ and the sink is positioned at $(0, 0)$. Based on our model, we use two deployment methods: (i) nodes are deployed randomly and uniformly and (ii) nodes are deployed randomly in a way such that their distance from $\mathcal{S}$ follows an appropriate distribution, i.e. more nodes are dropped closer to the sink.

In contrast to [7] where node density in all instances is fixed at $\mu(R) \approx 9.8$, we evaluate Directed Diffusion in a variety of denser networks. Furthermore, PFR is a multipath protocol which is shown to be very efficient when $\mu(R) \approx 31.5$, thus in this work both protocols are evaluated together for the first time, in a fair way. In particular, for the case where nodes are deployed randomly and uniformly, the density $\mu(R)$ in our simulation instances is set to $\{14.726, 19.634, 24.543, 29.452, 34.361\}$ for $n = \{300, 400, 500, 600, 700\}$.

A sensing task is assumed to operate on-top of Directed Diffusion and PFR that generates $\lambda = 2$ events per second. Each event is being sensed by a randomly chosen node and in our simulation 2000 events are generated. The energy available to the nodes was set to low levels so as to not be enough for all messages to be propagated. We made this choice in order to observe if and for how long the sleep-awake family of protocols manages to prolong the system lifetime. 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.

We start by evaluating the effect of density on the protocols' performance. In Fig. 3 we see that the use of APCP *more than doubles* the performance of Directed Diffusion in terms of the achieved success rate. Also, PFR *outperforms* Directed Diffusion both when used stand-alone and when combined with APCP, especially on high densities. In terms of consumed energy, Directed Diffusion with APCP outperforms again all protocols with greater energy savings achieved on the higher densities, which shows that APCP properly detects the surplus of nodes. All other protocols consume almost 100% of their energy. Finally, in terms of propagation delay, the use of APCP increases the delay of Directed Diffusion by at most 4.5 sec, while stand-alone Directed Diffusion is the fastest protocol. The increased latency is caused by contention in the MAC layer in stand-alone Directed Diffusion and PFR, while when using APCP it is mostly caused by nodes that fall asleep in the propagation path. The behavior of PFR is not drastically improved by the use of APCP, possibly due to the probabilistic nature of PFR and the use of directional transmissions, $P_{density}$ and P_{energy} fail to gather enough information about the network. This, combined with the fact that PFR operates best on very high densities, lead APCP to adjust the sleep interval of the nodes close to 0, thus the behavior of PFR is almost unaffected. This depicts an interesting property of APCP, that the adaptation process will fallback on the original protocol and the performance will not be hindered by it. We continue our experimentation by evaluating the performance of APCP for different values of α and β in respect to the network density.

Heterogeneity. In our second set of experiments we measure the effect of heterogeneity on the performance of the protocols. At $t = 0$ we deploy nodes in two groups: in the first group nodes have the usual amount of energy while in the second group the nodes have twice as much energy (we call these nodes "super-nodes"). The total number of nodes is calculated in such a way that the overall amount of en-

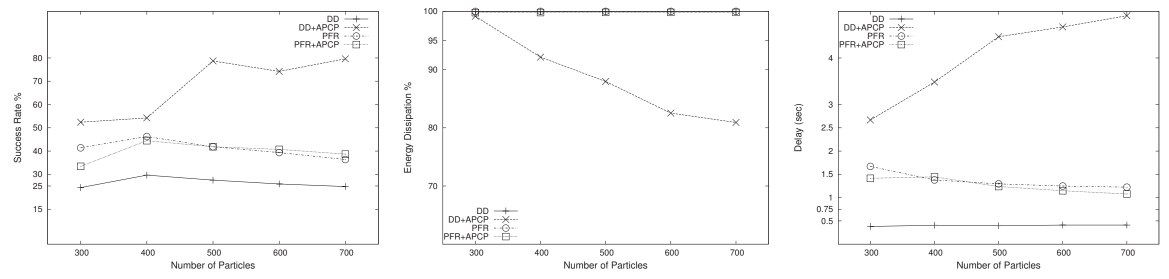

Figure 3. Success Rate, Energy Dissipation and Propagation Delay for different number of nodes ($n \in [300, 700]$)

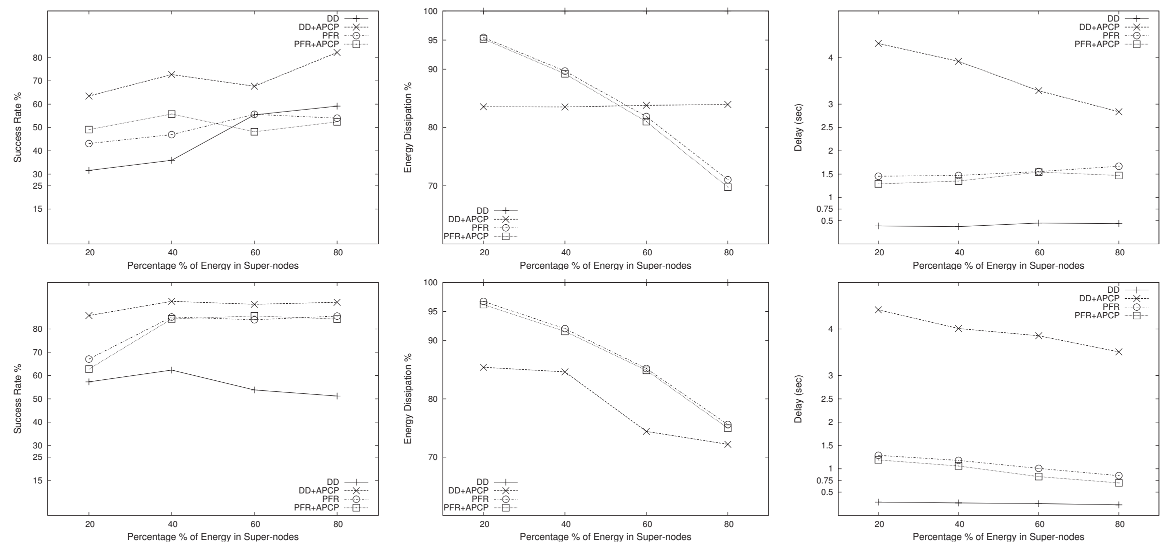

Figure 4. Success Rate, Energy Dissipation and Propagation Delay when deploying $20, 40, 60, 80$ per cent of the total energy in super-nodes using a Uniform Deployment (top row) or Non-Uniform Deployment (bottom row)

ergy in the network is the same as in the case of 500 nodes. We vary the percentage of energy contained in super-nodes to be $20, 40, 60, 80$ per cent. This means that when the percentage increases we drop more super-nodes and less regular nodes. In this experiment we also examine for the first time a new form of heterogeneity where the super-nodes are deployed in a non-uniform way.

In Fig. 4 we observe that again the combination of Directed Diffusion and APCP is outperforming all other protocols. The effect of having more super-nodes is different for the protocols, stand-alone PFR and stand-alone Directed Diffusion benefit from many super-nodes but when 80% of the energy is in super-nodes PFR's success rate drops, since for this instance the smallest number of nodes is deployed. The use of APCP has the effect that from the range of 40% to 60%, the performance of the protocols drops and then rises again. This behavior is observed at that point where $P_{density}$ overpowers P_{energy} and thus a new kind of adaptation is performed. For the case of non-uniform deployment, we can clearly observe that the success rate of all protocols is *greatly improved* and in particular for Directed Diffusion with APCP the success rate goes as high as 92%, while PFR surpasses by about 30% the original Directed Diffusion. This can be explained by the fact that the nodes closer to the sink propagate more messages and the non-uniform redeployment replenishes these nodes.

In terms of consumed energy, Directed Diffusion with APCP achieves to maintain the energy consumption almost stable and in the case of non-uniform redeployment achieves significant gains. Moreover, PFR does not con-

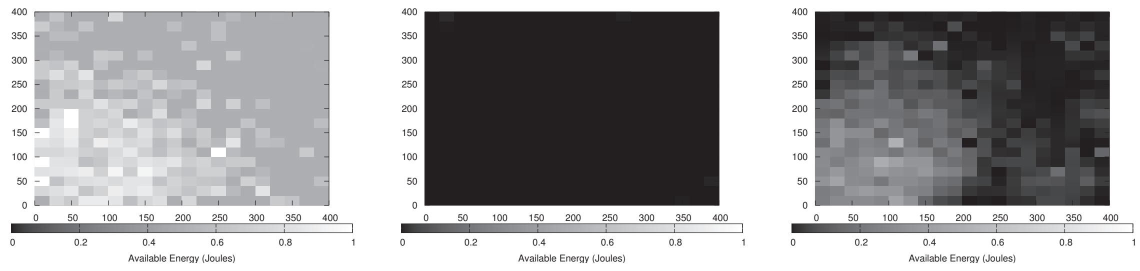

Figure 5. Map of Available Energy when $t = 0$ (left), at $t = 1000$ for Directed Diffusion (middle) and at $t = 1000$ for Directed Diffusion with APCP, when deploying 40 per cent of the total energy in super-nodes using a Non-Uniform Deployment. Light areas mean high energy availability while dark areas mean low energy availability, where each square corresponds to the average energy contained in the devices of a 20m $\times$ 20m area.

sume all the energy of the super-nodes as shown by the energy dissipation curve which drops while the percentage of super-nodes increases. In order to further illustrate this, we include in Fig. 5 the map of available energy for the case when deploying 40% of the total energy in super-nodes using a Non-Uniform Deployment. In the middle map we see that Directed Diffusion exhausts the energy resources of the sensors, while when applying APCP (right map), the energy consumption drops significantly.

The propagation delay findings are similar to the previous experiment with the exception that the propagation delay of Directed Diffusion with APCP drops as the percentage of super-nodes increases, which means that paths are formed where "super-nodes" sleep less.

Node Redeployment. In our third experiment we measure the effect of node redeployment. We use $n = 500$ nodes and separate them in g groups $g_0, g_1, \ldots$. The group g_0 is deployed at $t = 0sec$ and each g_i (for $0 < i < g$) is deployed at $t = i \times 200sec$. We also examine the effect of non-uniform redeployment for the nodes of groups $g_1, g_2, \ldots$, while the nodes of group g_0 are always deployed uniformly.

In [5], we show that node redeployment benefits protocols and especially the original Directed Diffusion which, in terms of success rate, manages to surpass Directed Diffusion with APCP as the number of groups g increases. PFR benefits from node redeployment in the non-uniform case: as g increases PFR's performance almost matches that of Directed Diffusion. The degradation of the protocols' performance when using APCP is probably due to its failure to adjust fast enough to changes in the network topology. Similar observations hold in terms of energy consumption and propagation delay. Directed Diffusion consumes the energy of the nodes fast and by choosing to redeploy nodes we can improve its performance.

References

[1] T. Abdelzaher and et al J. Stankovic. *Real-Time Communication and Programming Abstractions for Sensor Networks*. CRC Press, July 2004.

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Journal of Computer Networks*, Volume 38:393–422, 2002.

[3] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic. Towards optimal sleep scheduling in sensor networks for rare-event detection. In *Intl. Conference on Information Processing in Sensor Networks (IPSN '05)*, 2005.

[4] I. Chatzigiannakis, T. Dimitriou, S. Nikoletseas, and P. Spirakis. A probabilistic forwarding protocol for efficient data propagation in sensor networks. *Ad-Hoc Networks Journal*, 2005. Also in the proc. of the 5th European Wireless Conference (EW 2004), pp. 344-350.

[5] I. Chatzigiannakis, A. Kinalis, and S. Nikoletseas. An adaptive power conservation scheme for heterogeneous wireless sensors. In *Proceedings of the 17th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA 2005)*, pages 96–105. ACM Press, 2005.

[6] I. Chatzigiannakis and S. Nikoletseas. A sleep-awake protocol for information propagation in smart dust networks. In *3rd Workshop on Mobile and Ad-Hoc Newtworks (WMAN), IPDPS Workshops*, 2003.

[7] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *6th ACM/IEEE International Conference on Mobile Computing – MOBICOM*, 2000.