Adaptive sleep scheduling protocol in wireless sensor networks

GERGELY ÖLLÖS, ROLLAND VIDA

Budapest University of Technology and Economics,
Department of Telecommunications and Media Informatics
{ollos, vida}@tmit.bme.hu

Keywords: wireless sensor networks, distributed sampling, adaptive regression

Energy efficiency in wireless sensor networks is a major issue, since the sensors usually have limited and irreplaceable power sources. Sleep scheduling solutions proved to be exceptionally effective strategies to achieve this goal. Numerous such algorithms have been proposed and examined, but virtually without any considerable support for dynamic systems. In this paper we propose and analyze an adaptive, soft-state, fully distributed and robust sleep scheduling method that can easily cope with frequent node failures. The proposed scheme can dynamically eliminate the redundancy and estimate the deficient data based on learned relations in a way to ensure low and balanced energy consumption. This is done without the need for offline pre-computations, dedicated phases, time synchronization, localization, or base station assistance. We compare our technique with deterministic clustering methods, provide parameter sensitivity analysis and discuss the simulation results.

1. Introduction

Distributed sensor networks were in the focus of researchers since the early 1990s. There was a trend to move from centralized, highly reliable, powerful but expensive platforms to a large number of cheap, decentralized and potentially unreliable components that as a group are capable of far more complex tasks than any individual super-node. Wireless sensor networks (WSNs) are formed by one or more base stations (sinks), where the collected data is sent, and a large number of sensors distributed over the monitored area and connected through radio links. Sensors are low-cost and low-power tiny nodes equipped with limited sensing, computing, and radio communication capabilities. They typically have irreplaceable power sources, designed for single usage, and are deployed in an unplanned manner.

There is an essential difference in our terminology, as compared to the usual one, related to cluster definition. By cluster we indicate a subset of entities that could be *potentially* monitored (e.g., a set of coordinates where sensors could be placed), and not a subset of sensor nodes; thus, in our terms, cluster formation mainly depends on the environment and the physical phenomena in which we would like to find the redundancy. The nodes can move over those clusters, which are slowly changing in time. In order to better understand our model, we introduce some basic definitions. Let F be a set of entities that could be potentially monitored. Then, $f_i \in F$ is the *i-th cluster*, i.e., a subset of F in which each of the entities can be mutually described based on another arbitrary entity in the same set, within a user specified error bound. Thus, we need to sample only one of the entities in the cluster, and then can estimate any other entity in the same set. When cluster f_i is monitored using k nodes, we call it *k-coverage*, where the redundancy is 1:k; thus, k–1 nodes can be sent to sleep mode. The number and the topology of the clusters f depend on several factors such as the monitored physical phenomena, the environment, or the error bound. The clusters might also dynamically change in time. If we have two clusters f_i and f_j, and we manage in a way to estimate any of the entities in f_i based on the readings of any entity in f_j, the two abstract clusters will merge.

When $\forall f_i \in F$ is monitored by one and only one node n_i, we call the coverage perfect. This can be achieved only if $N \geq |f|$ where $|f|$ is the number of clusters and N is the number of nodes. Since the structure of the clusters is unknown, we overdeploy the field F in order to increase the probability $P(\forall j \exists i: n_i \rightarrow f_j)$, where $n\_i \rightarrow f_j$ means that node n_i measures one of the entities in cluster f_j. Therefore, the global lifetime of the network GL (i.e., the time until $\forall j \exists i: n_i \rightarrow f_j$ holds) can be easily computed:

$$GL = \min_{\forall j} \left\{ \sum_{i=1}^{|M_j|} \frac{B_{M_j^i}}{C} \right\} \quad (1.1)$$

where M_j is a set of nodes that measure entities in cluster f_j, M_j^i is the i-th node's index in this cluster, B_i is the battery capacity of node n_i (in [mA/h]), and C is the power consumption of a particular node in [mA] (i.e., we suppose that all of the nodes are similar and have the same power requirements).

In this paper we propose thus a dynamic sleep scheduling protocol that aims at maximizing global network lifetime while ensuring that all clusters are monitored by at least one awake node all the time.

This paper is organized as follows. Section 2 discusses the related work, while in Section 3 we show a short real-world case study to emphasize the linear association between temperature measurements; based on the described properties, we propose a simple model for the analysis that is done in the next sections. In Section 4 we describe and analyze the adaptive regression method, which is the main component of the Adaptive Sampling Protocol (ASP) we propose. Then, in Section 5 we present the complete, fully distributed network level solution. In Section 6 we show the simulation results, define the deterministic clustering to which we compare our solution, and investigate the power balancing capabilities of the protocol. Finally, in Section 7 we conclude the paper.

2. Related work

The early papers on energy efficiency were discussing fault tolerance [2] or energy-efficient routing [3], but sleep scheduling, i.e., sparing the energy of the network by placing a subset of nodes into sleeping mode, is a relatively new approach [4]. It is true that sleep based protocols are common in wireless networks generally, as battery energy can be significantly preserved if the mobile device is in sleep mode. However, sleep scheduling in wireless sensor networks is a much more sophisticated problem. Sensors are not standalone devices, they are responsible together for the monitoring task. Therefore, a sleep scheduling protocol should enable sensors to take turns in sleeping and preserve their energy while ensuring however, that the monitoring quality is not affected. In the last few years, many papers discussed a wide range of sleep-scheduling solutions. For instance [5] discussed localized sleeping algorithms based on distributed detection for differential surveillance, [4] discussed system issues and focused on prototyping, while [6] focused on the detection of rare events. However, all of these solutions are based on static, and not adaptive methods. In this paper we discuss an application layer approach, as opposed to many other methods and protocols that achieve higher energy efficiency operating on lower layers, like the MAC – medium access control layer [7-9]. Similar ideas to our method are also explored in [10].

In [11] authors proposed a similar coverage-preserving node-scheduling scheme which can reduce energy consumption and therefore increase system lifetime by turning off some redundant nodes. They presented a basic model for coverage-based off-duty eligibility rule and then extended it to several different scenarios. Each node in the network autonomously and periodically makes decisions on whether to turn itself on or off, using only local neighbor information. To preserve sensing coverage, a node decides to turn itself off when it discovers that its neighbors (sponsors) can help in monitoring its whole working area. To avoid blind points, which may appear when two neighboring nodes expect each other's sponsoring, a backoff-based scheme is introduced to let each node delay its decision with a random period of time. This method assumes however that nodes know their position and sensing range, which in addition is circular and has the same radius for all nodes. Further, this method can not fully exploit the linear correlations between the measurements and it is not able either to balance the available power levels in the network.

In [12] authors proposed a scheme in which the lifetime of a sensor node is divided into epochs. For each epoch, the base station computes a minimum set of active nodes, based on the current level of coverage requirement, i.e., each sensor samples the field only if it is chosen by the base station to do so. In [13] the authors' approach has two phases. The first one is the development of models (offline) for predicting the measurements of one sensor using data from other sensors. The second is the creation of the maximal number of subgroups of disjoint nodes so that for each such subgroup the measured data is sufficient to recover the measurements of the entire sensor network. For prediction of the sensor measurements, the authors introduced a new optimal non-parametric polynomial time isotonic regression. To capture the evolving dynamics of the instrumented environment, they monitor the prediction errors occasionally to trigger adaptation of the models.

These schemes usually assume a static level of coverage, but even if adaptivity is ensured, either the parameters of the adaptive model are computed offline, or the adaptive algorithm is controlled by a central base station, or the used model is too restricted. There are three main disadvantages to an adaptive, but centralized approach. First, there is a significant communication overhead. Second, the response time to dynamic events might be unacceptably high. Third, if the base station is temporarily unavailable, the sleep scheduling on the whole network is disrupted; as a result, the sensor network cannot continue to function efficiently.

Our proposed method differs from existing works, since the adaptation to dynamic events is online and continuous; there is no need for dedicated phases, nor for base station assistance, since the measurements of sleeping nodes are approximated locally. Also, our proposed ASP protocol can support mobility, it is fully distributed, can be gradually enabled on the network, does not need position information, the model has no restriction on sensing range and finally it is a robust solution in terms of node failures.

3. Short case study

In this section we will shortly describe the correlation and statistical properties of temperature and luminosity samples, in order to support our model.

In a dormitory room we placed five sensors that measured temperature and luminosity for three days. The room residents were living their normal daily life without any interruption or alteration. We used identical sensors, deployed as shown by the numbered points in *Fig. 1 (left)*. We can see that node n_5 was close to the heater and to the windows, it had therefore the biggest temperature interferences. On the same figure we can see the ceiling fluor lamps and the reading lamps (marked with *L*)

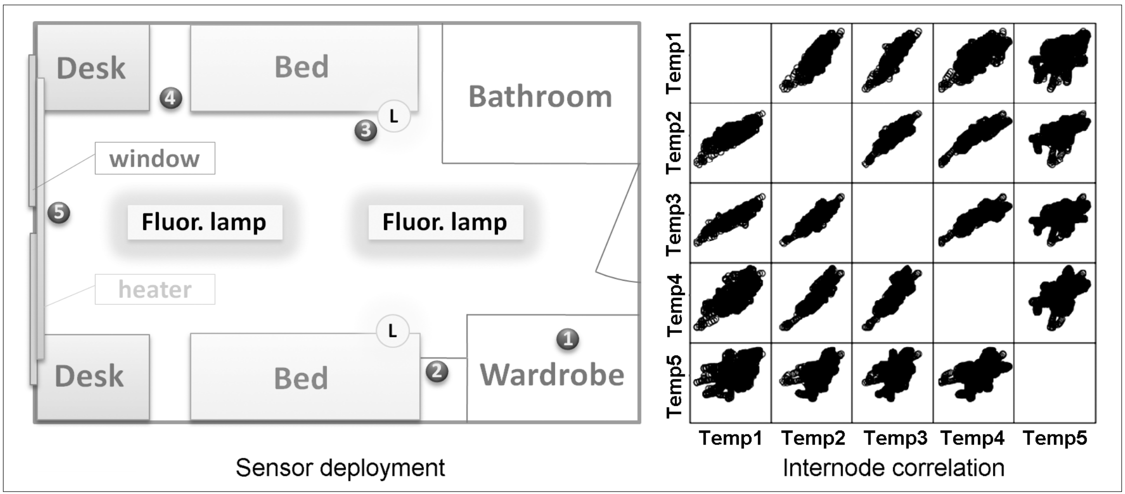


Figure 1. Deployment (left) and internode correlation (right)

as well. We can thus see that node n_3 had the largest light interference caused by a reading lamp. Naturally, the room residents caused additional interferences. This arrangement ensured real-world measurements. The used nodes were Crossbow MICA-Z motes [14], running the Tiny OS [15] operating system with Zigbee stack. Motes were equipped with an ISM radio transceiver (2.4 GHz IEEE 802.15.4) with a maximum data rate of 250 kbps, and had 4 Kbytes of internal memory. The sensor and data acquisition card plugged into the processor radio board collected light and temperature measurements with a one minute sampling rate. In short, we collected temperature and light samples from five different sensors over three days, sampled every minute without interruption.

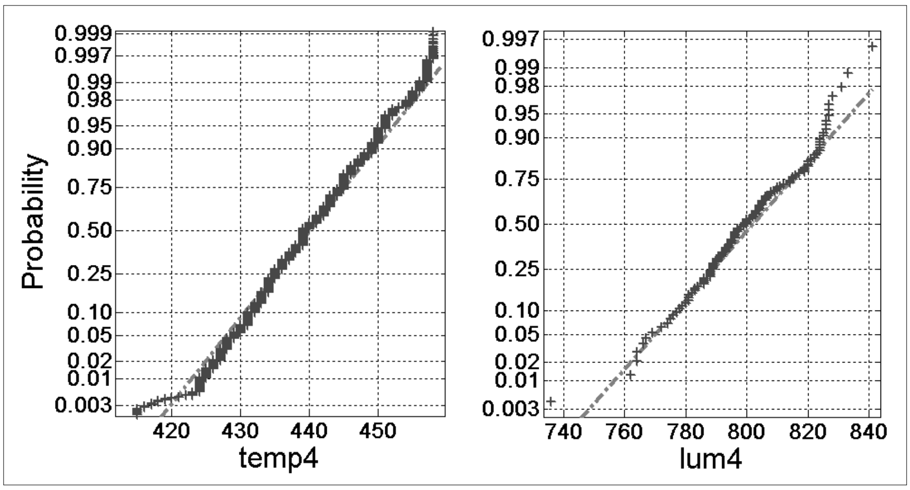
In *Fig. 1 (right)* we see the correlations between measurements, represented on specific graphs for each pair of nodes. For instance, let's take the graph that describes the relations between temperature readings of nodes n_3 (Temp3) and n_2 (Temp2). For any given time t there is a corresponding point on the graph, with the readings on node n_3 represented on the x axis, and the readings of node n_2 on the y axis. As it can be seen, every graph has its inverse, which does not hold extra information, because the sub-graphs are symmetric, i.e. [Temp3, Temp2] is the exact inverse of [Temp2, Temp3]. We can see that node n_2 is well correlated with the measurements of node n_3, because they are close to each other and are far from the windows. By contrast n_3 and n_4 are closer to each other, but n_4 is close to the window as well, and it is thus exposed to significant temperature disturbances (the room residents often ventilated the room). In our measurements a significant spatial correlation can be observed, which we exploit (among others) in the Adaptive Regression procedure our protocol builds on (details will be given later).

Fig. 2 presents the sampled temperature (the Mote's ADC output can be converted to degrees using the Steinhart-Hart equation) and luminosity data; superimposed on the plot is a line joining the first and third quartiles of the samples. This line is extrapolated out to the ends of the sample to help evaluate the linearity of the data. The purpose of a normal probability plot is to graphically assess whether the data could come from a normal distribution or not. If the data are normal, the plot will be linear, while other distribution types will introduce curvatures in the plot. We can see that our samples approximately follow a normal distribution; thus, we assume normality of samples originating from short sampling intervals (window size of several minutes).

Since the strong linear correlation in an over-deployed sensor network is typical, especially when monitoring temperature, humidity, light, etc., we have chosen a simple linear regression model. Another reason is the well known fact that if the samples are normally distributed then the relation between the measurements could only be linear.

We will generate a number of artificial samples for two nodes of different (linear) correlation structure, in order to observe and compare the adaptation response and behavior. The first node's (n_i) measurements $X \in N(\mu_2, \sigma_2)$ are modeled by normal distribution. The sec-

Figure 2. Two typical normal probability plots of temperature and luminosity samples



ond node's (n_j) measurements Y are modeled through a linear relation, as follows. Let $Z \in N(\mu_1, \sigma_1)$ then:

$$Y = (1-k)(aX+b) + kZ \tag{3.1}$$

where a, b are the parameters that affect the linear relation, while k affects the strength of the relation between X and Y. If (3.1) is true then:

$$Y \in N\left(a(1-k)\mu_2 + k\mu_1 + b(1-k);\ \sqrt{a^2(1-k)^2\sigma_2^2 + k^2\sigma_1^2}\right) \tag{3.2}$$

The α and β parameters of the linear regression ($Y \approx \alpha X + \beta$) are as follows:

$$\begin{aligned}\alpha &= \frac{cov(X,(1-k)(aX+b)+kZ)}{\sigma^2(X)} \\ &= \frac{cov(X,(1-k)(aX+b)) + cov(X,kZ)}{\sigma^2(X)} \\ &= \frac{cov\big(X, aX(1-k)\big) + cov(X,kZ)}{\sigma^2(X)} \\ &= \frac{a(1-k)cov(X,X) + kcov(X,Z)}{\sigma^2(X)} \\ &= \frac{a(1-k)\sigma^2(X)}{\sigma^2(X)} = a(1-k)\end{aligned} \tag{3.3}$$

$$\begin{aligned}\beta &= E(Y) - \alpha E(X) \\ &= E\big((1-k)(aX+b)+kZ\big) - a(1-k)E(X) \\ &= E(X(a-Ka)+b-kb) - a(1-k)E(X) \\ &= a(1-k)E(X) + b(1-k) - a(1-k)E(X) \\ &= b(1-k)\end{aligned} \tag{3.4}$$

Further the Pearson product-moment correlation coefficient for this model is:

$$\begin{aligned}r &= \frac{cov(X,Y)}{\sigma(X)\sigma(Y)} = \frac{cov(X,(1-k)(aX+b)+kZ)}{\sigma(X)\sigma(Y)} \\ &= \frac{a(1-k)\sigma^2(X)}{\sigma(X)\sigma(Y)} = a(1-k)\frac{\sigma(X)}{\sigma(Y)} \\ &= \frac{a(1-k)\sigma_2}{\sqrt{a^2(1-k)^2\sigma_2^2 + k^2\sigma_1^2}}\end{aligned} \tag{3.5}$$

This coefficient is zero (no relationship) if k=1 and there is an exact linear relationship ($Y=aX+b$) if k=0:

$$r_{k=1} = 0, \qquad \sigma_1, \sigma_2 \neq 0 \tag{3.6}$$

$$r_{k=0} = \begin{cases} 1, & a > 0 \\ 0, & a = 0 \\ -1, & a < 0 \end{cases} \qquad \sigma_1, \sigma_2 \neq 0 \tag{3.7}$$

This model will be assumed throughout the analytical evaluations of the proposed protocol (or for sample generation during simulations if not stated differently).

4. The adaptive regression method

In this section we describe the main component of our Adaptive Sampling Protocol (ASP), the adaptive regression (AR) method. This method is used to track other nodes in the network and interpolate the measurements of those nodes if needed. Thus, the AR method deals only with two nodes: the local node that executes the AR method and a distant node being tracked by the local node.

The adaptation, i.e., the adaptive regression method is simple. The main idea is as follows. In each iteration (when a sample arrives from the distant node) the local node samples the environment as well and pushes the sample pair (local and distant measurement) into a sample FIFO buffer. We use this buffer for estimating the linear regression parameters and the expected error. For each monitored neighboring node, the local node has separate FIFO buffers, and each newly received distant sample is pushed into the proper buffer, along with the latest local measurement. Then, the node recalculates the parameters of the linear regression when needed.

The length of the FIFO in an ideal situation is 2 since this is enough to determine the linear relation. However, in real world measurements there is a significant noise; therefore, we need to have more than two samples (typically 10-30). The optimal number depends on the amount of noise present (lower bound) and on how fast the correlation structure changes (upper bound). Basically, the length of the FIFO should be determined empirically; however, a 20 unit long FIFO is a good trade-off between correlation detection time and noise immunity in most cases regarding temperature or humidity measurements.

The method can dynamically determine if two different clusters f_i and f_j can be merged together (if there is a strong linear relation), switch off the redundant node and therefore prolong the global lifetime GL of the network. In the beginning, we assume that $\forall j \exists! i: n_i \rightarrow f_j$; the coverage is perfect, $N \geq |f|$ is satisfied. Let $x[t]$ be a sample from one of the entities in cluster f_i (a realization of X), sampled by node n_i at moment t. Similarly, let $y[t]$ be a sample from one of the entities in cluster f_j (a realization of Y), sampled by node n_j at the same moment. Two clusters f_i and f_j can be merged at moment t_k, for a period t_p, if $\exists a[t_k], b[t_k]$ so that:

$$\frac{1}{t_p}\sum_{t=t_k}^{t_k+t_p}\{y[t] - a[t_k]x[t] - b[t_k]\}^2 \leq U_{err} \tag{4.1}$$

where U_{err} is the user specified mean square error (MSE). Naturally, we have to know the MSE of our model before we send nodes n_i or n_j to sleep mode for a time interval t_p. We continuously estimate the mean square error of our model, and if (4.1) is satisfied, we presume that the process is stationary for another time interval t_p. Then, we send one of the nodes to sleep mode for t_p, while the awaken node will regress the sleeping node's measurements (based on the estimated regression line) and send them to the sink, on behalf of the sleeping node. The parameter estimation in case of linear regression [16,17] is well known, so we only summarize the equations.

Let $\{x[t_k], y[t_k]\}, \{x[t_k+1], y[t_k+1]\}, \ldots, \{x[t_k+t_p], y[t_k+t_p]\}$ be the discrete samples from clusters f_i and f_j, sampled by nodes n_i and n_j.

If
$$a[t_k] := \frac{S_{xy}}{S_x{}^2} \approx a = \frac{cov(X,Y)}{\sigma^2(X)} \quad (4.2)$$
and
$$b[t_k] := M\left(\underline{y}\right) - a[t_k]M\left(\underline{x}\right) \approx b$$
$$= E(Y) - \frac{cov(X,Y)}{\sigma^2(X)}E(X) \quad (4.3)$$

the sum specified in (4.1) will be minimal; thus, the linear model is optimally set. In our algorithm we are continuously pushing the $[x,y]$ pairs to a FIFO queue, and with each new learning pair we update the latest a, b, a_{inv}, b_{inv} parameters. For all of the monitored clusters f, node n_i can have separate and independent FIFO queues (as in the ASP protocol described later).

The sum specified in (4.1) is the estimation of the expected error. For simplicity let $\{x[t_k], y[t_k]\}, \{x[t_k+1], y[t_k+1]\}, \ldots, \{x[t_k+t_p], y[t_k+t_p]\}$ be the discrete samples from clusters f_i and f_j, sampled by nodes n_i and n_j.
$$\frac{1}{t_p}\sum_{t=t_k}^{t_k+t_p}\{y[t] - a[t_k]x[t] - b[t_k]\}^2$$
$$\approx E[(Y - aX - b)^2] \quad (4.4)$$

Lemma 1: If (4.2) and (4.3) is true then
$$E[Y - aX - b] = 0 \quad (4.5)$$

Proof:
$$E[Y - aX - b] = \quad (4.6)$$
$$= E\left[Y - \frac{cov(X,Y)}{\sigma^2(X)}X - E[Y] + \frac{cov(X,Y)}{\sigma^2(X)}E[X]\right]$$
$$= E[Y] - \frac{cov(X,Y)}{\sigma^2(X)}E[X] - E[Y] + \frac{cov(X,Y)}{\sigma^2(X)}E[X] = 0$$

The expected error calculated by the adaptive regression method is:
$$E[(Y - aX - b)^2]$$
$$= E^2[Y - aX - b] + \sigma^2(Y - aX - b)$$
$$= \sigma^2(Y - aX - b) = \sigma^2(Y - aX)$$
$$= \sigma^2(Y) + a^2\sigma^2(X) - 2a\,cov(X,Y)$$
$$= \sigma^2(Y) + \frac{cov^2(X,Y)}{\sigma^2(X)} - 2\frac{cov^2(X,Y)}{\sigma^2(X)}$$
$$= \sigma^2(Y) - \frac{cov^2(X,Y)}{\sigma^2(X)} \quad (4.7)$$

As we pointed out earlier, if the expected error is lower than the error specified by the user, one of the nodes goes to sleep mode, depending on which node has less energy remaining; this will ensure proper power balancing, which extends the GL (1.1) global lifetime of the network.

5. The Adaptive Sampling Protocol

Each sensor node that executes the Adaptive Sampling Protocol (ASP) operates in three phases: adaptation, bargaining, and interpolation. Moreover, the bargaining phase has three steps: interpolation request, interpolation response, and election. The protocol is composed of (and can be well described by) three distinct extended finite state machine (EFSM) models. However, because of space limitations, we focus only on the overall behavior of the ASP protocol. Our solution exploits a potential that we pointed out in Section 3, namely that the nodes close to each other are usually well correlated. We also assume that these close by nodes can hear each other as well.

In *Fig. 3.1* we can see the adaptation phase. We will focus on node X and its neighboring nodes Y_n. Each node in the adaptation phase grabs packets from its surroundings. In this example nodes Y_1, Y_2, Y_3 can receive the packets transmitted by node X node and execute a copy of the adaptive regression method for node X. On the other hand, let's say that node Z receives the packets as well, but it does not monitor node X, since it does not have enough resources (the maximum number of monitored nodes has been already reached). The unlabeled nodes can't receive the packets of node X, since they are too far away. With each new sample that X sends to the base station, the Y nodes adapt their model as described in the previous sections.

In *Fig. 3.2* we can see the interpolation request step of the bargaining phase. In order to decrease the communication overhead, the interpolation request is implicit; it is thus embedded into the packet which carries the latest measurement. The nodes are sending interpolation requests by uniform distribution. The probability of sending the request could be increased based on the available power levels, gradually enabling there-

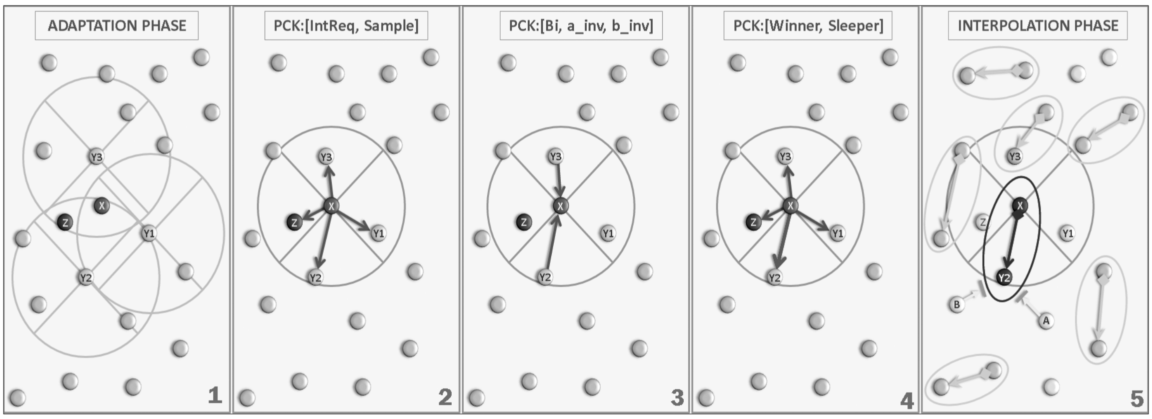


Figure 3. Adaptation (1), bargaining – interpolation request (2), interpolation response (3), election (4) –, interpolation (5) phases

fore the sleep scheduling in the network. Since the interpolation request is included in a packet that carries the samples, only those nodes can hear the request which have heard the samples and potentially adapted their model to node X. An interpolation request does not generate any unnecessary overhead (since it is embedded), because if some node replies to it, an interpolation phase will certainly follow (i.e., one of the nodes will certainly sleep).

Since the energy spared during sleeping is much more than the energy spent for replying the request (maximum each neighbor can send one single reply, where the number of neighbors is typically below five to ten), increasing the probability to send a request results in more energy savings. When this probability is 1 (constantly sending the request), then the ASP protocol spares as much energy as the user specified quality threshold and the correlation structure of the measurements permits.

In Fig. 3.3 we can see the interpolation response step of the bargaining phase. Only those nodes (from the vicinity of node X) answer the interpolation request whose model's expected quality of regression is below the user specified threshold of expected quality T_{eq}. Here we extended the adaptive regression method with a quality definition. The expected quality is a weighted sum of the expected error (as described in Section 4) and the average age of the sample pairs in the sample FIFO queue, based on which we calculate the parameters of the linear regression. In distributed environments this extension is necessary since we have more than two nodes in the network and they can constantly move as well. It can easily happen that the monitored node X moves out of the receiving range of nodes Y_n, and then it moves back after a while; this would cause the samples to be fragmented in time in the sample buffer of nodes Y_n. In other words, the samples in the FIFO queue will either be too old or current ones, but nothing between them; the samples will not be uniformly distributed, which distorts the regression. We can detect and control the effects of this distortion since the average age of the samples is increasing with the dispersion of the age of sample pairs.

In this example, let say that the dispersion of the age of sample pairs in the buffer of node Y_1 is too high; thus, the expected quality of regression is high as well, which results in node Y_1 not answering the request of node X. Please note that the expected quality metric is inverted (the lower the expected quality, the better the extrapolation). Since node Z did not monitor the measurements of node X, it does not answer the request either. Nodes Y_2 and Y_3 answer the implicit interpolation request of node X with an interpolation response that carries the particular node's actual power level and its inverse regression parameters.

In *Fig. 3.4* we can see the election step of the bargaining phase. After node X received the actual power level and inverse regression parameters, of the candidate nodes (Y), it selects the node with the minimal power level (in our example Y_2). At this point, we determined which pair to involve in the regression phase. However, node X has also to decide which of the two nodes (X or Y_2) will go to sleep. The decision is simple: the node with less energy remaining. After the decisions, node X informs the winner candidate, and the selected node goes to sleep mode for a predetermined time interval T_p.

In *Fig. 3.5* we can see the interpolation phase. In our example node Y_2 goes to sleep for a predetermined interval T_p, and during that period node X interpolates its samples and sends them to the base station on behalf of node Y_2. In order to interpolate the measurements of node Y_2, node X uses the inverse regression parameters that Y_2 has sent in the interpolation response step of the bargaining phase. Nodes X and Y_2 are in interpolation and sleeping mode, respectively. During this state, sleep request from other nodes (in this example nodes A and B) are ignored.

Since the protocol is soft state and fully distributed, it can handle frequent node failures as well. If a node fails during the adaptation or interpolation phase, this is equivalent with the situation when the failed node doesn't have a good model for interpolating the requesting node's measurements (i.e., it does not answer the request in any way). If a node fails in any other phase, the worst case scenario (when the node doing the interpolation fails) is that for a single T_p time interval (which is measured in seconds) we lose the measurements of that cluster (two nodes). After that, the protocol naturally recovers from the failure through the next interpolation request.

The ASP protocol takes into account node power levels, in order to ensure proper power balancing, as well as the expected quality of the interpolation. Since the expected quality, as a statistical measure, is much less reliable, and the power balancing is an important task (to extend the global lifetime of the network), the protocol ensures the node that goes to sleep is always the one which has the less energy remaining, in the vicinity of node X (including node X itself). In the same time, only those nodes will answer the interpolation request of node X for which the expected quality (composed of expected error, as described earlier, and the average age of the samples in the buffer) is less than a user specified expected quality bound. This allows us to influence the interpolation error. Furthermore, the sleep scheduling protocol can be gradually enabled on the network, as described earlier.

In the ASP the expected quality of the regression is calculated as a weighted sum of the expected error and the average age of the samples. Since the expected error is not symmetric, we have to calculate the expected quality on both sides of the regression. The average age of samples is symmetric, since we are calculating the average age of learning points from the same buffer, for each side. The symmetry of the average age of samples is straightforward; however, the asymmetry of the expected error needs a minor explanation.

Lemma 2: If the standard deviations of the measurements of two nodes X, Y are finite, nonzero, different, and the Pearson product-moment correlation coefficient is not ±1, then the expected error (as we defined it in Section 3) is not symmetric (which is usually the case):

$$E[(Y-aX-b)^2] \neq E[(X-mY-n)^2] \quad (5.1)$$

Proof:

$$E[(Y-aX-b)^2] = E[(X-mY-n)^2] \quad (5.2)$$

$$\sigma^2(Y) - \frac{cov^2(X,Y)}{\sigma^2(X)} = \sigma^2(X) - \frac{cov^2(X,Y)}{\sigma^2(Y)} \quad (5.3)$$

$$cov^2(X,Y)\left(\frac{1}{\sigma^2(Y)} - \frac{1}{\sigma^2(X)}\right) = \sigma^2(X) - \sigma^2(Y) \quad (5.4)$$

$$\frac{cov^2(X,Y)}{\sigma^2(X)\sigma^2(Y)} = 1 \quad (5.5)$$

$$R^2(X,Y) = 1 \quad (5.6)$$

$$R(X,Y) = \pm 1 \quad (5.7)$$

This is the reason why we need to send the a_{inv} and b_{inv} parameters in the interpolation response phase along with the battery status.

6. Simulation results

In this section we analyze and compare the Adaptive Sampling Protocol with the deterministic clustering approach. First we discuss the sample generation process and its statistical properties. Then, we introduce the reference model to which we compare the proposed method throughout this paper. We provide then performance and overhead analysis, and finally we discuss the power balancing property of our protocol, which significantly extends network lifetime.

A) Sample generation and analysis

We measured the properties of the samples in the following manner. First we generated 25x10000 samples for 25 nodes (for each node 10000 samples). The sampling frequency of the nodes was 1Hz and they were able to execute 100 logical operations in 1 second (100 state transitions per second in the EFSMs). Thus, for 100 seconds of simulation time, we needed 100x100=10000 samples, where each sample represents 10 ms holdup in time. The typical sample's buffer length was 20, there was thus a 20-entries long FIFO in which we shifted the samples (pushed one to the top, and discarded one from the bottom of the FIFO, in each second), while we continuously computed the statistical properties of the samples in the FIFO. We have randomly chosen two nodes, and plotted the result in *Fig. 4* as illustration.

As we can see, the covariance, the dispersion and the expected error on both nodes are continuously changing (as we discussed in Section 3). The range and profile of the curves are similar between each pair of measurements, but the maximum and the inflection points are differently situated. What is typical for each pair is the decreasing determination index (or coefficient). The reason why we generated such samples needs a short explanation. The variation of the samples is made up of two parts: the part that can be explained by the regression equation (this is the determination index) and the part that can't be explained by the regression. The determination index can have many different definitions, depending on the class of problem. In our case (linear regression) the determination index (or coefficient) is exactly the square of the Pearson product-moment correlation coefficient.

According to **Lemma 1** and the definition of the expected error (in Section 4):

$$E[(Y-aX-b)^2] = \sigma^2(Y) - \frac{cov^2(X,Y)}{\sigma^2(X)} \quad (6.1)$$

Since the regression function $f(X)$ in our case is linear $f(X)=aX+b$:

$$R_f^2 = 1 - \frac{E[(Y-f(X))^2]}{E[(Y-E(Y))^2]} = 1 - \frac{E[(Y-aX-b)^2]}{\sigma^2(Y)}$$

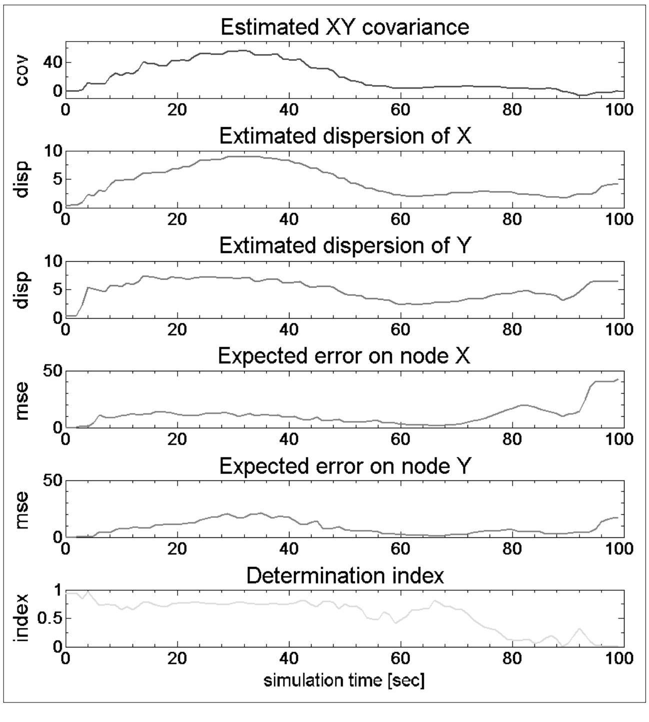
$$= 1 - \frac{\sigma^2(Y) - \frac{cov^2(X,Y)}{\sigma^2(X)}}{\sigma^2(Y)}$$

$$= \frac{\sigma^2(Y) - \sigma^2(Y) + \frac{cov^2(X,Y)}{\sigma^2(X)}}{\sigma^2(Y)}$$

$$= \frac{cov^2(X,Y)}{\sigma^2(X)\sigma^2(Y)} = R^2(X,Y) \quad (6.2)$$

Therefore, if the determination index is 1, there is no introduced error by the adaptive regression (there is an exact linear relation between the measurements of the local and the distant node). As the determination index is decreasing the performance of the adaptive regression (AR) is declining as well. If the determination index is zero then the linear regression can't explain the variance of the extrapolated node by definition. If the samples are

Figure 4. Typical statistical properties of samples



normally distributed (as we presumed and demonstrated in our scenario) they are independent as well.

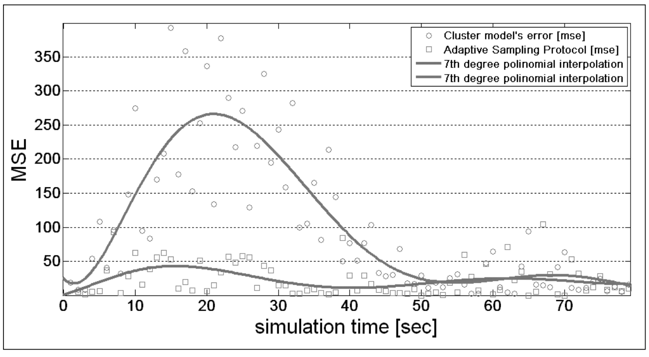
B) Deterministic clustering

The deterministic clustering approach is (implicitly) widely used in existing sleep scheduling protocols. The main idea is that if we form the clusters in an overdeployed network based on topological distances (the nodes that are close to each other form a cluster), then with the measurements of one node we can approximate the other nodes in the same cluster (they are measuring approximately the same values, as discussed in Section 3). We select a cluster head which measures the environment, and the rest of the nodes are going to sleep mode until the cycle ends. In each cycle a new node (from the same cluster) assumes the role of the cluster head, this node being chosen in a deterministic or probabilistic manner. These protocols are however unaware of the current measurements, and the clustering is static; they do not support dynamic environments, where the correlations between nodes are changing.

Our reference model (referred as deterministic clustering) is as follows. We divide the network into two-node clusters (if there is an odd number of nodes, then the last cluster consists of three nodes) in order to be comparable with the ASP protocol which dynamically creates two-node clusters as well. In each cluster one of the nodes is always sleeping, while the awaken node samples the environment and sends the measurement to the sink on behalf of both nodes. In each cycle (sampling period) the nodes assume reverse roles, in order to sustain the network's power dispersion. The error that this model makes is the squared measurement difference between the measurements of the awake and the sleeping node. If we compute the average error in the network for each cluster, we get the mean squared error (MSE) of the deterministic clustering in a given cycle.

In *Fig. 5* we can see the comparison between the MSE obtained for the deterministic clustering protocol (not aware of measurements) and for the Adaptive Sampling Protocol, which is aware of the measurements and the correlation structure. The reason for the high MSE values between seconds 10 and 40 is that the dispersion of the measurements between the samples is higher (due for example to node or event mobility). After 60 seconds of simulation, the dispersion is significantly smaller, which means that the nodes are measuring similar values (that's why the blind deterministic clustering performs so well). As you can see, the ASP protocol can adapt to dynamic environments. In the rest of the analysis we will usually compute the average MSE of the protocol per simulation run. In this case, for 80 seconds of simulation (one run) the average error made by ASP is approximately 27, while the average error made by the deterministic clustering scheme is approximately 103.

Figure 5. Deterministic clustering versus ASP



C) Performance of the ASP protocol and comparisons

In this section we compare and analyze the Adaptive Sampling Protocol. First we provide a parameter sensitivity analysis for the length of the sample buffer and the user specified threshold of expected quality; then, we also discuss the protocol's overhead.

In *Fig. 6* we can see the protocol's behavior if we change the length of the sample FIFO (queue) buffer, as well as its effect on the protocol overhead. As it can be expected, if we increase the size of the sample buffer, the response time of the ASP protocol (for correlation changes) increases. In other words, if the correlation structure between two monitored nodes changes (relatively) quickly, the expected quality will not decrease below the user specified threshold (TEq) so rapidly; therefore, the interpolation request will be rejected and there will be thus less sleep cycles (in general), as the figure shows. The protocol's overhead is measured in the number of extra packets sent, which is strongly correlated with the number of sleep cycles. In order to avoid redundancy, we will discuss this issue in detail later.

If the number of sleep cycles is decreasing then naturally the power consumption is increasing. *Fig. 7* shows

Figure 6. Sample buffer size sensitivity (sleep cycles)

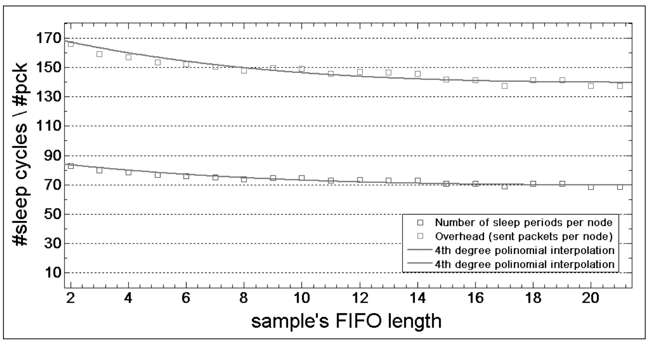
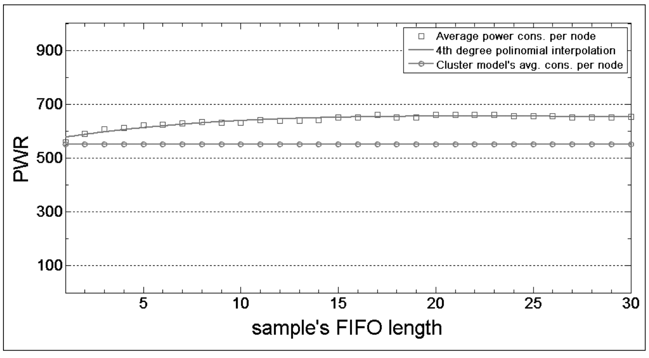


Figure 7. Sample buffer size sensitivity (power consumption)



the average power consumption per node, for both the ASP protocol and the deterministic clustering. We simulated both protocols with the following parameters: there were 25 nodes in the network (5x5 grid), the simulation time was 1000 seconds, and the speed of the EFSMs was 100 ticks/sec. For a sleeping node we chose the power consumption to be 0.001 units/tick and for the awake node 0.01 units/tick. Given the parameters, it is easy to compute the average power consumption of a node for the deterministic clustering: it is 550 units in this scenario (550 = 500*100*0.01 + 500*100*0.001) since each node is sleeping in half of the time. Given that the deterministic clustering has no sample FIFO, this consumption is independent of the FIFO length; that's why in the figure it appears as a straight line. If we increase the length of the samples FIFO, the adaptation to correlation changes is slowing down; many of the old samples are still in the FIFO, they overweight the new samples, and thus the protocol can't exploit short term correlations. This means that the number of sleep cycles is decreasing (as we can see in Fig. 7) and, therefore the power consumption is slightly increasing.

Figure 8. TEq sensitivity analysis (estimation error)

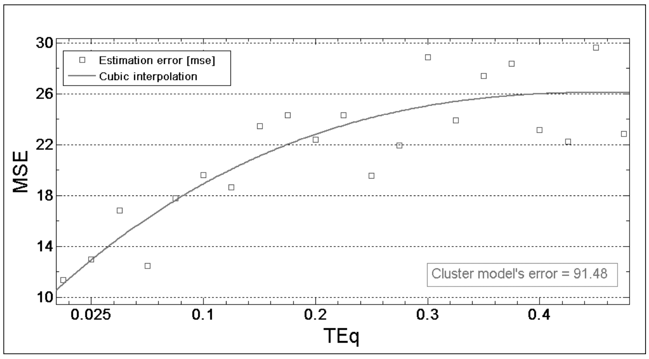


Figure 9. TEq sensitivity analysis (power consumption)

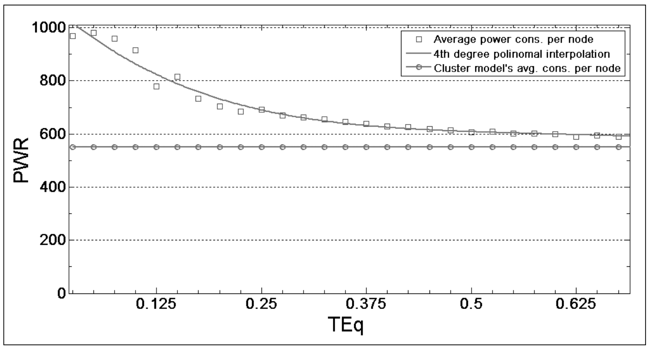


Figure 10. TEq sensitivity analysis (sleep cycles)

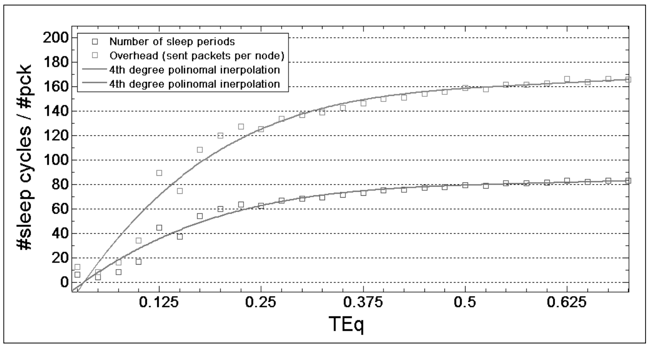
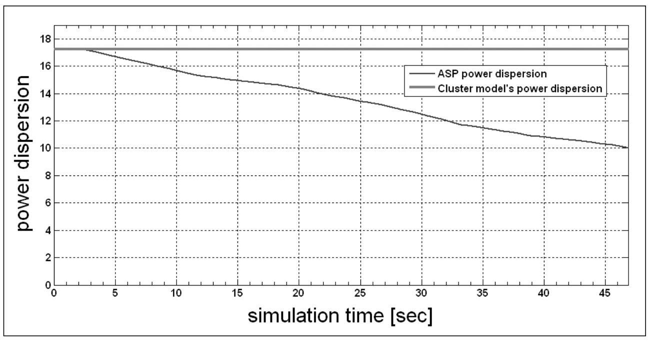


Figure 11. Power dispersion comparison



In *Fig. 8* we can see that if we increment the user specified threshold of expected quality, the extrapolation error is increasing as well. The simulation configuration is as we described earlier. There are 25 nodes (in a 5x5 grid arrangement) in the network, and they are not moving. The samples that are fed to the network are as we described in Section 6/A. Each node can track 8 nodes and in the 5x5 grid each node has maximum 8 neighbors. The simulation time is 1000 seconds and the sample buffer length is 15. Like in the previous comparison, in this static environment the ASP protocol outperforms the deterministic clustering roughly 3-4 times regarding the estimation error. Please note that the samples between nodes are virtually not correlated in 30% of the simulation time, given that the typical determination index is decreasing (as we discussed in Section 6/A.).

As we mentioned it earlier, the blind deterministic clustering scheme results in the theoretically minimal energy consumption. *Fig. 9* indicates how much does the power consumption of the ASP converge to this minimum. However, as the power consumption is decreasing, the extrapolation error is increasing (Fig. 8).

In *Fig. 10* we can see the average number of sleep cycles per node and the average number of sent messages per node, as we change the *TEq* (threshold of expected quality) parameter. As we pointed it out earlier, if the *TEq* parameter increases, the number of sleep cycles increases as well, and thus the power consumption decreases. Before each sleep cycle, there is a three step negotiation, with the first step (interpolation request) being implicit (carried in the packet along with the sample). The remaining two steps result in overhead packets, the average overhead per node (in sent packets) is therefore strongly correlated with the number of sleep cycles, and is increasing as the *TEq* parameter is increasing. The number of overhead packets is approximately equal to the number of sleep cycles times two; however, this relation is strongly varying from node to node, although in average (per node) this is a close estimation, as shown in Fig. 10.

This overhead could slightly increase if the neighborhood of nodes is dense, since in this case more nodes can apply for the competition (send an interpolation response). The increase in node density means that a node will probably have more neighbors. Since the number of nodes that a particular node can track has a fixed upper bound, the number of answers to interpolation requests

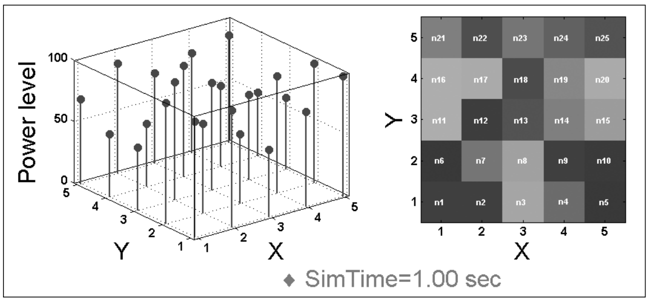


Figure 12. Network power level status in the first second

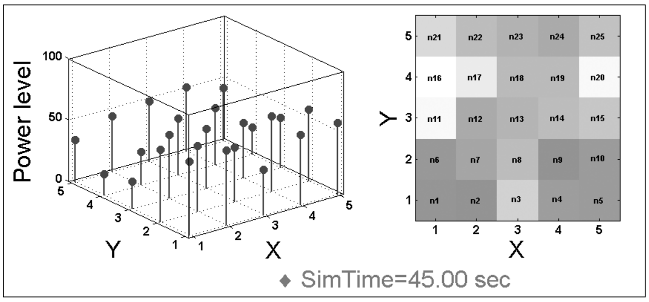


Figure 13. Network power level status after 45 seconds

in the network has an upper bound as well. When we simulated the network's behavior, we have set this parameter to be equal to the number of neighbors, maximizing thus the overhead in the described scenario. This means that if the density of nodes increases with the number of interpolation requests remaining constant, the overhead will not be significantly higher. Of course, this could increase the radio interference on the MAC layer, but because of the distributed nature of the protocol this would not affect significantly the overall behavior of the ASP protocol.

D) Power balancing

In this section we discuss the power balancing capabilities of the Adaptive Sampling Protocol.

In *Fig. 11* we can see the comparison of power balancing capabilities of the ASP and the deterministic clustering. The deterministic functioning of the cluster based approaches assures the detection time of events. This capability usually infers the constant power dispersion in the network through time, which could significantly decrease the global lifetime of the network. As we can see in Fig. 11 the ASP balances the energy reserves of the network, and increases thus the global lifetime of the network (1.1).

Fig. 12 is a snapshot of energy reserves in the network in the first second of the simulation time. As we mentioned it earlier, the nodes were arranged into a 5x5 grid. During this simulation, each node could monitor maximum 4 nodes and each node had maximum 4 neighbors. The user specified threshold of expected quality was 0.15, the samples buffer size was 20, and the initial power dispersion was random (uniform distribution). As we compare the snapshot from the first second with the energy snapshot after 45 seconds *(Fig. 13)* of simulation time, we can see that the nodes with higher energy reserves consume more energy than the others.

In other words, the ASP has the ability to logically transfer energy reserves between nodes so as to extend the global lifetime GL (1.1) of the network.

7. Conclusion

This paper proposes the Adaptive Sampling Protocol, a fully distributed WSN protocol. Some of the applications of the proposed method are target tracking, environmental monitoring, surveillance, early warning systems, etc.

In ASP the nodes in the network are monitoring each other's measurements, dynamically learn the linear relations among them (if any), eliminate (send to sleep) the redundant nodes, and estimate the deficient data without the need for offline pre-computations, dedicated phases, or base station assistance. There is no need for time synchronization or localization. The algorithm is based on continuous correlation monitoring and estimation, where the extrapolation error can be influenced by a user specified threshold of expected quality. The ASP protocol can be gradually enabled on the network, i.e., from a deterministic functioning, when the detection time is guaranteed, to the fully adaptive mode, when ASP spares as much energy as the correlation patterns and the user specified threshold permit. Another advantage of the ASP protocol is the strong energy balancing capability which could significantly extend the lifetime of the network.

The ASP protocol is designed to support adaptive environments and as the survey [1] indicates, it's a first of its kind. ASP is a robust protocol and can function even if the network has broken up to isolated segments; it can easily cope with frequent node failures as well. Further, the protocol overhead is well correlated with the number of sleep cycles, which can be influenced by the *TEq* parameter (as simulations showed). Since the communication is local, the power requirements for the overhead frames are minimal. Also, there are no network level interferences introduced, as opposed to the base station centralized approaches. If the measurements are not correlated, then the ASP protocol switches back to deterministic mode, but only on that part of the network where the linear association is under the threshold. The disadvantage of our protocol is that the power consumption only converges to the theoretical minimum, but never reaches it. Further, one node can monitor multiple distant nodes but can interpolate only one at a time. This means that the protocol can dynamically create only two sized clusters and thus the energy spared is limited to 50%.

The main component of the ASP protocol is the adaptive regression core, which we first discussed separately. We supported our assumptions with a short case study, discussed the properties of the measurements and based on this knowledge, we generated the samples for simulations. The results show that the ASP protocol generally outperforms the clustering approaches and it converges to the theoretically minimal energy consumption. We showed that the power balancing capabilities of the ASP protocol are strong. Furthermore, we showed that with the user specified threshold of expected quality, the real estimation error can be well influenced.

In the future we will work on a distributed model which can predict various occurrences of discrete events in dynamic environments, based on a fully distributed (neuro) Fuzzy architecture.

Acknowledgement

The work described in this paper has been (partially) supported by EC FP7 EFIPSANS project (INFSO-ICT-215549).

Authors

GERGELY ÖLLÖS is a Ph.D. student at the Budapest University of Technology and Economics. He obtained his MSc degree in Technical Informatics from the same university in 2008. He was project leader at WetCom s.r.o., Bratislava, Slovakia, in 2008 where he designed and built a laser stabilization controller for accurate positioning along with its communication firmware. He worked at Nokia, Komárom, Hungary in 2008 as High Frequency Radio Engineer, P2P and C lecturer. In the last few years he worked on a neural network based navigation system and built a robot for demonstration for which he received diplomas in Prague and Budapest in 2008 and 2007, respectively. Now his interests lie predominately in the area of low-power wireless networking and machine learning..

ROLLAND VIDA is Associate Professor at the Budapest University of Technology and Economics. He obtained his BSc and MSc degrees in Computer Science from the Babes Bolyai University, Cluj-Napoca, Romania, in 1996 and 1997 respectively, and his PhD degree from the Université Pierre et Marie Curie, Paris, in 2002. Between 2003 and 2005 he obtained the György Békésy Postdoctoral Fellowship, and in 2007 the János Bolyai Research Fellowship. In the last five years Dr. Vida has acted as organizer, TPC member or reviewer for more than 30 international conferences, participated in several national and European research project, and taught different networking courses at universities in Romania, Slovakia and Hungary. In 2008 he was elected as Chair of International Affairs of the Scientific Association for Infocommunications, Hungary.

References

[1] L. Wang and Y. Xiao, "A survey of energy-efficient scheduling mechanisms in sensor networks" In Mobile Networks and Applications, 2006.

[2] F. Ye, G. Zhong, S.L.J. Cheng, and L. Zhang, "Peas: A robust energy conserving protocol for long-lived sensor networks," In Proc. of ICDCS 2003, 23rd IEEE International Conf. on Distributed Computing Systems, Providence, USA, May 2003, pp.28–37.

[3] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly resilient energy-efficient multipath routing in wireless sensor networks," ACM SIGMOBILE Mobile Computing and Com. Review, Vol. 5, April 2001, pp.11–25.

[4] C. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: random and coordinated sleep algorithms," In Proc. of IPSN 2004, The 3rd International Symp. on Information Processing in Sensor Networks, Berkeley, USA, April 2004, pp.433–442.

[5] T. Yan, T. He, and J. Stankovic, "Differential surveillance for sensor networks," In Proc. of ACM Sensys 2003, Los Angeles, USA, November 2003, pp.51–62.

[6] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic, "Towards optimal sleep scheduling in sensor networks for rare event detection," In Proc. of IPSN 2005, The 4th International Symp. on Information Processing in Sensor Networks, Los Angeles, USA, August 2005, pp.20–27.

[7] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," In Proc. of the IEEE INFOCOM, June 2002.

[8] S. Du, A. K. Saha, and D. B. Johnson, "Rmac: A routing-enhanced duty-cycle mac protocol for wireless sensor networks," In Proc. of the IEEE INFOCOM, May 2007.

[9] T. van Dam and K. Langendoen, "Adaptive energy-efficient mac protocol for wireless sensor networks," In Proc. of ACM SenSys, Los Angeles, USA, November 2003.

[10] F. Ye, G. Zhong, S. Lu, and L. Zhang, "Conserving protocol for longlived sensor networks," In Proc. of the 23th IEEE International Conf. on Distributed Computing Systems (ICDCS), Providence, Rhode Island, USA, May 2003.

[11] D. Tian and N. D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," In Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA), New York, USA, 2002.

[12] H. Ling and T. Znati, "Energy efficient adaptive sensing for dynamic coverage in wireless sensor networks," In Proc. of WCNC 2009, Budapest, April 2009.

[13] F. Koushanfar, N. Taft, and M. Potkonjak, "Sleeping coordination for comprehensive sensing using isotonic regression and domatic partitions," In Proc. of IEEE Infocom 2006, Barcelona, Spain, April 2006.

[14] Crossbow technology, micaz 2.4ghz mote (Online). Available: *http://www.xbow.com/Products/productdetails.aspx?sid=164* [retr. Okt. 2009]

[15] Open technology alliance, tinyos (Online). Available: *http://www.tinyos.net* [retr. Okt. 2009]

[16] J. Cohen, P. Cohen, S. West, and L. Aiken, Applied multiple regression and correlation analysis for the behavioral sciences. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 2003.

[17] N. Draper and H. Smith, Applied Regression Analysis. Wiley Series in Probability and Statistics, 1998.