Round Robin Cycle for Predictions in Wireless Sensor Networks

[#] Le Borgne Yann-Ael, Bontempi Gianluca
Machine Learning Group, Universite Libre de Bruxelles
CP 212, Bd Triomphe
1050 Bruxelles - Belgium
{*yleborgn,gbonte*}*@ulb.ac.be*

Abstract

Use of prediction models in sensor networks proves to be efficient with respect to energy savings, as it allows sensors whose readings are predicted to remain in their idle mode, thereby consuming orders of magnitude less energy that in the active mode. In the context of continuous monitoring, where a set of sensors is typically required to regularly send their readings to a central server, an interesting approach consists in splitting the set of sensors in two subsets, such that readings of one subset are used to predict readings of the second subset. In this paper, we propose to identify several sensor subsets for predictions, that are used in turn in a round robin fashion. Identification of different sensor subsets allows to detect erroneous models or sensor failure, and to better distribute energy consumption. Efficiency of the proposed procedure is demonstrated on a set of experiments using real world sensor data.

INTRODUCTION

Many applications of wireless sensor networks require a large set of sensors (potentially hundreds or thousands) to regularly report their readings to a central server to undertake off-line data analysis. These applications, for instance environmental or structural monitoring, HVAC systems, or object tracking, also require the sensors to be running for as long a period of time as possible. Limited energy ressources available on a sensor module (a.k.a. mote) drive the need for efficient scheduling of sensing and networking activities, so as to maximize the application lifetime.

On a mote, efficient energy consumption is achieved by switching to a sleeping mode components not in use. Different operating modes allow the mote to switch on and off components such as micro-controller unit (MCU), sensors, flash memory or radio. Radio transmission and reception are known to be the dominant factors of energy consumption on a mote [1], as their energy cost is at least an order of magnitude higher than the sole use of the MCU. Many strategies for in-network compression have been suggested in the litterature to decrease use of radio communication, such as distributed source coding, routing compression [7], or cluster-based aggregation [4]. These compression methods entail energy savings by reducing the communication time between motes. However, these techniques still need the sensors to collect measures at regular intervals, and so even though overall radio communication activity is reduced, motes still consume energy by keeping their MCU and sensors ON when data is sensed from the environment.

To further decrease energy consumption, it has recently been proposed to use a *model driven* approach to reduce the number of sensors solicited in a sensing task [2]. In this approach, a subset of sensors is identified (the prediction subset), from which readings of remaining sensors (the predicted subset) can be predicted within a user specified error threshold and confidence level. This offers optimal energy savings for sensors not solicited, as in current technology OFF operation mode, energy consumption is three orders of magnitude less than when the MCU is ON, and four orders of magnitude less than when the MCU and the radio are ON [1].

Two undesirable consequences however follow from the use of a single prediction subset:

1) **Erroneous models**: If dependencies between sensor readings change over time, prediction models will become outdated, entailing a possibly rapid deterioration in predicted sensor values.
2) **Unequal energy distribution**: Continuous solicitation of the same sensor subset will lead to the energy depletion of these sensors.

The first issue is related to the fact that the data distribution of the readings generated by a sensor network can not be assumed to be stationary. The data set used to select a prediction subset may not be consistent with subsequent measurements of the sensor network. Strategies to detect sensor malfunctioning and concept drift must therefore be implemented to avoid erroneous predictions. The second issue stems from the repeated use of the same subset of sensors.

In this article, we propose relying on a round robin system to adress the above mentionned issues. The principle consists in designing a cycle such that a different prediction subset is used at each step, and such that all sensors are at least queried once during a cycle. This scheme allows to both detect faulty sensors or changes in the dependencies between sensors by keeping on collecting measures from all sensors. Moreover, by taking into account estimates of sensor remaining energy, the cycle can be defined so as to minimize use of sensors

with low remaining energy, thus extending the whole network lifetime.

After presenting the network model considered in this paper in section 1, we define in section 2 the notions of prediction and predicted subsets of sensors. In section 3, we elaborate on the cycle construction algorithm, central to the scheme presented in this paper. This is followed in section 4 by a set of experimental results related to prediction accuracy and distribution of energy consumption.

1. NETWORK MODEL

The focus of the paper is on sensor network applications requiring a continuous delivery of sensor measurements. In this class of applications, all sensor measurements need to be transmitted at regular time intervals to a central server [5]. We suppose the central server to be unique, the sensing nodes to be static, and the sensor network to be organized in clusters. A cluster head is associated to each cluster, and all sensors in a cluster are in transmission range with the cluster head (1-hop cluster). Sensor readings are first sent to their corresponding cluster head before being routed to the central server, see figure 1. In order to simplify the sensor energy consumption model, we also suppose that cluster heads are endowed with renewable energy ressources, so that routing costs between cluster heads and central server are negligeable.

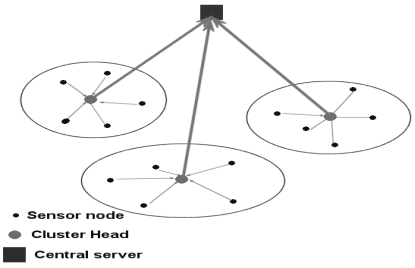

Fig. 1: Model of the network. 1-hop cluster organization, 1 central server

To estimate the lifetime of a sensor, we propose relying on a measure defined as the query budget QB, corresponding to an estimate of the number of queries that can be made before depletion of a sensor. If we let Q_{remain} (in Joules) be the energy remaining in a sensor, Q_{acq} (in Joules) be the acquisition cost of a measure, and P_{link} be the probability for the communication between the sensor and the head cluster to succeed, then we can estimate the query budget by $QB = P_{link}\frac{Q_{remain}}{Q_{acq}}$.

2. PREDICTABILITY

The proposed systems relies on the notion of predictability between sensor readings. We define formally in this section the notion of prediction and predicted subsets.

A. Prediction models

Let $\mathcal{S} = \{s_1, s_2, ..., s_S\}$ be a set of S sensors deployed over an environment, and let $s_i(t)$ be the value read by sensor s_i at epoch number t. We suppose that the monitoring task starts at epoch $t = 1$, and let D_T be an observation set collected over T epochs. Let

$$\hat{s}_i(t) = h(\mathbf{s}(t), \alpha)$$

be a prediction model for sensor s_i, where $\mathbf{s}(t)$ denotes a vector of inputs and α the set of parameters. Parameters α are identified through a supervised learning procedure based on the past observation set D_T [8].

Inputs $\mathbf{s}(t)$ to the model are readings of different sensors at instant t (use of readings obtained before t, that could capture temporal dependencies, are left for future work as they may not be reliable. Indeed, once the system runs in the model driven mode, the past observation set D_T becomes partially filled with predicted values). For example, inputs for $\hat{s}_1(t)$ could be $s_2(t)$ and $s_4(t)$ in a network with $S \geq 4$ sensors. Choice for input variables to the prediction model is a feature selection problem that will be adressed in section 3-D.

B. Prediction error assessment

The prediction error of a model for a sensor s_i is typically assessed by a validation procedure, such as cross validation, that relies on the use of a loss function $C(s_i(t), \hat{s}_i(t))$. The loss function quantifies how much predictions $\hat{s}_i(t)$ deviate from their actual values $s_i(t)$. Different loss functions can be used, such as quadratic loss function $C(s_i(t), \hat{s}_i(t)) = (s_i(t) - \hat{s}_i(t))^2$, absolute loss function $C(s_i, \hat{s}_i(t)) = |s_i(t) - \hat{s}_i(t)|$, or ϵ-approximation based loss function $C(s_i, \hat{s}_i(t)) = P(|s_i(t) - \hat{s}_i(t)| > \epsilon)$. Choice for the loss function is user or application dependent.

C. Predictability

In our frame, we define sensor s_i as *predictable* if a prediction model $\hat{s}_i(t) = h(\mathbf{s}(t), \alpha)$ can be identified, such that its average estimated prediction error is inferior to a user defined error threshold T, i.e. $C(s_i(t), \hat{s}_i(t)) < T$ on average. The notion of predictability is thus user defined. In the remaining of the paper, predictability always suppose an underlying user defined T error threshold.

The subset of sensors whose readings are inputs to $\hat{s}_i(t)$ is defined as the prediction subset $\mathcal{S}^i_{pred}$ for s_i. By extension, we define as a prediction subset for $\mathcal{S}$ a subset $\mathcal{S}_{pred}$ such that $\forall i, \exists \mathcal{S}^i_{pred} s.t. \mathcal{S}^i_{pred} \subset \mathcal{S}_{pred}$. The complementary subset $\mathcal{S} \backslash \mathcal{S}_{pred}$ is refered to as the *predicted subset*. Note that $\mathcal{S}$ is always a prediction subset as it provides all sensor readings. The goal in our frame is however to find prediction subsets of minimal size to limit the number of solicited sensors at a given instant.

3. ROUND ROBIN CYCLE

A. Round robin principle

The proposed approach consists in identifying a set K prediction subsets $\mathcal{S}_k$ that are queried in turn, thus forming a round robin cycle, such that all sensors are at least queried once during a cycle. The main idea in designing such a cycle is twofold.

First, by keeping on collecting data from all sensors, techniques can be implemented to check for potential sensor malfunctioning or changes in the reading dependencies among sensors. Design of such techniques is a problem that will not be further studied in this paper, as we focus here on designing a cycle that allows such techniques to be implemented.
Second, using different prediction subsets may allow to more evenly distribute energy consumption among sensors. This second point depends on the predictability between sensors, and it may not be always possible to evenly distribute energy. However, as will be shown in the experimental section, if the error tolerance and the cycle length defined by the user are not too strict, energy can almost be evenly distributed in the network.
The cycle length is a parameter that is user or application dependent. It is a trade off between the reliability of the monitoring system, and the distribution of energy consumption. In the general case, the higher the cycle length, the less frequent the possibilities to check for erroneous models or sensor failures, but the higher the flexibility in designing a cycle that distribute energy consumption among sensors.

B. Illustration

We propose in this section to illustrate the different cases that can be encountered. Given a simple network with five sensors s_1, s_2, s_3, s_4, and s_5, let us consider the three following examples.
Example 1: Suppose that any sensor is predictable from any other. Then a cycle of length $K = 5$, with $\mathcal{S}_1 = \{s_1\}$, $\mathcal{S}_2 = \{s_2\}$, $\mathcal{S}_3 = \{s_3\}$, $\mathcal{S}_4 = \{s_4\}$, and $\mathcal{S}_5 = \{s_5\}$ can be designed, thus reducing energy consumption in the network by a factor 5. Higher cycle lengths would allow to rely more on sensors with higher query budgets, and smaller cycle lengths would yield suboptimal solution in terms of energy savings.
Example 2: Suppose that sensors s_1, s_2, and s_3 are all predictable with one another, and that sensor s_4 is only predictable from sensor s_5 and vice-versa. A 6-step cycle, for example $\mathcal{S}_1 = \{s_1, s_4\}$, $\mathcal{S}_2 = \{s_2, s_5\}$, $\mathcal{S}_3 = \{s_3, s_4\}$, $\mathcal{S}_4 = \{s_1, s_5\}$, $\mathcal{S}_5 = \{s_2, s_4\}$, and $\mathcal{S}_6 = \{s_3, s_5\}$, would allow to optimize the sensors' lifetimes if they initially had equal query budgets.
Example 3: Finally, suppose the extreme case where no sensor readings are predictable. In this worst case, the only solution is a cycle of one step, with $\mathcal{S}_1 = \{s_1, s_2, s_3, s_4, s_5\}$, and there is no possibility to solicit some sensors more than others.

Designing such a cycle is a complex optimization problem, involving user defined criteria (cycle length and accuracy), and sensor parameters (sensor query budgets). We propose in section 3-D an algorithm that drives the search for prediction subsets by minimizing use of sensors with low query budgets. Before introducing the algorithm, we first describe how the data gathering process can be simply implemented in the network.

C. Data gathering process

The design of the cycle takes place at the central server. A first stage consists in collecting readings from all sensors to obtain an obervation set D_T. Second, the cycle construction algorithm presented in the next section can be applied to obtain a set of K prediction subsets. This set can be represented as an activity table of S rows and K columns, whose element $[i, k]$ is set to 'S' (for send) if sensor s_i is to report its reading at the k-th step of the cycle, and 'I' (for idle) otherwise. Each row of this matrix defines the activity schedule of a sensor, that is sent over to the corresponding mote. The mote then applies this activity schedule repeatedly, switching between active and idle modes according to its schedule. Such an activity schedule is represented on figure 2.

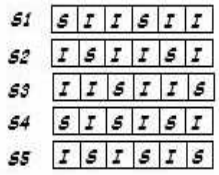

Fig. 2: Activity schedule of the five sensors for the example 2 of section 3-B.

Transmission of the activity schedule requires sending S packets of K bits (by coding 'S' state by 1 and 'I' state by 0). The cost associated to this transmission is therefore negligeable if the average ratio of predicted sensors is not dramatically low.
If sensor failures or erroneous models are detected, it may be necessary to construct a new cycle after some time to keep up with the user defined accuracy level. The updated observation set D_T is however partially filled with predicted values. Two solutions can be implemented to adress this issue: either predicted values in D_T can be given low weights during the stage of prediction model identification, or the system can revert back to the non predictive mode, collecting values from all sensors to get a new and complete observation set.

D. Cycle construction

We propose in this section an algorithm to construct the set of prediction subsets that will form the cycle. The number of possible arrangements for predicted and prediction sensor subsets grows exponentially with S, and feature selection methods must therefore be used to guide the search for prediction sensor subsets. Existing techniques for feature selection can be grouped into two main approaches: the filter and the wrapper approach [6]. The filter approach selects features using a preprocessing step independently of the learning algorithm, whereas the wrapper approach takes into account a learning algorithm to estimate, through validation methods such as cross validation, the quality of a feature subset.
As the feature selection problem faced here is constrained by a user defined accuracy on the prediction errors, the wrapper

approach is preferable. In the following, we consider that the learning algorithm used is a black box which can return, for a given sensor subset, an estimate of prediction errors on other sensor measurements.

Standard algorithms that find feature subsets using the wrapper approach follow an incremental search, either in a forward fashion, i.e. starting from a null subset, and then adding features until the features to predict reach a given level of accuracy, or in a backward fashion, i.e. starting with the whole set of variables and removing features as long as the features to predict remain above the prespecified level of accuracy. Addition or removal of a feature at a given step of these algorithms is decided by a measure taking into account different criteria of interest in selecting or discarding a feature. There is a priori no decisive argument to opt for a forward or backward search. We present in table 1 an algorithm for the cycle construction, based on a backward search. The algorithm takes as inputs the set of sensors S, the remaining query budget vector QB, and the user defined cycle length K and prediction error threshold T. Creation of a cycle is made of two stages:

First stage: A first step $\mathcal{S}_1 = \mathcal{S}$ containing all the sensors is defined. A backward feature selection is applied, which tries to remove sensors [1] by increasing order of their remaining query budgets. Once such a sensor is found, a second step $\mathcal{S}_2 = \{\mathcal{S} \backslash s\_current\}$ is created, and the first step $\mathcal{S}_1$ remains the same. Then the algorithm tries to remove the following sensors, still by increasing order of their remaining query budgets. If a sensor can be removed from both $\mathcal{S}_1$ and $\mathcal{S}_2$, a third step $\mathcal{S}_3 = \{\mathcal{S}_3 \backslash s\_current\}$ is created.

The rationale of this procedure is that each newly created step is obtained by splitting on the last step of the current cycle, whose sensors all have query budgets higher than at least one of the sensor in the preceding steps. This allows to make sure that a sensor that can be removed from all existing steps at a given point of the cycle construction will be assigned to one and only one step. As sensors are taken by increasing order of their remaining query budgets, this gives more chance to sensors with low query budget to be assigned to only one step of the cycle.

For example, let us consider a set of 4 sensors s_1, s_2, s_3 and s_4, with query budgets of 4, 8, 9 and 10 respectively, such that any two sensors can predict the remaining two, but that no sensor is predictable by a single sensor. The first step is initially $\mathcal{S}_1 = \{s_1, s_2, s_3, s_4\}$. s_1 can be removed, as it is predictable from $\{s_2, s_3, s_4\}$. The first step is left unchanged, and a second step $\mathcal{S}_2 = \{s_2, s_3, s_4\}$ is created. s_2 can be removed from $\mathcal{S}_1$ and from $\mathcal{S}_2$, so a new step $\mathcal{S}_3 = \{s_3, s_4\}$ is created, and s_2 is left in $\mathcal{S}_2$. s_3 can then be removed from $\mathcal{S}_1$ and $\mathcal{S}_2$ but not $\mathcal{S}_3$, so no new step is created, and we have $\mathcal{S}_1 = \{s_1, s_4\}$, $\mathcal{S}_2 = \{s_2, s_4\}$, and $\mathcal{S}_3 = \{s_3, s_4\}$. Finally, s_4 cannot be removed from any step, and so the first stage of the algorithm would stop here. Note that s_1, s_2 and s_3 all belong to a different step of the cycle, as they were removed first, and that s_4 belongs to all steps as it was removed last.

Second stage: The second stage simply consists in creating new steps, applying a backward feature selection by removing sensors by increasing order of their updated query budget. Given the three first steps obtained in the first stage of the algorithm, updated query budgets are 3, 7, 8 and 6 for s_1, s_2, s_3 and s_4, respectively. A new step $\mathcal{S}_4 = \{s_1, s_2, s_3, s_4\}$ is created, from which are removed s_1 and s_4 as they have the lowest updated query budgets. Therefore $\mathcal{S}_4 = \{s_2, s_3\}$. Updated query budgets are now 3, 6, 7 and 6, which leads to $\mathcal{S}_5 = \{s_2, s_3\}$ or $\mathcal{S}_5 = \{s_2, s_4\}$ depending on the draw between s_2 and s_4. This procedure is reiterated until the desired cycle length K is reached.

Finally, note that if steps cannot be created in stage 1 because the specified cycle length K has been reached, then a sensor that can be predicted by any of the existing steps is reassigned to the step with the lowest number of sensors, which aims at balancing the number of sensors among the cycle steps.

[1]The function predictable(Y,X,T) returns true if the sensor subset Y is predictable from X with accuracy T

4. EMPIRICAL STUDY

A. Data set

To assess our procedure, we realized a set of experiments, varying the different parameters. The data set used was a set of temperature readings obtained from a 54 Mica2Dot sensor deployment at the Intel research laboratory at Berkeley [3]. We discretized the original dataset to 2-minute interval measurements over a 10 days period. After preprocessing, the dataset contained a trace of 7200 readings from 52 different sensors (sensors 5 and 15 were removed as they collected too few data).

B. Prediction model

Probability distribution of a set of environmental readings such as temperature, light, or humidity have been shown to be well approximated by multivariate gaussian models [2]. Multivariate gaussian models offer a certain number of properties, such as adaptavity to missing values and on-line updating, that make them attractive in the context of sensor network monitoring. They also allow to capture in a single model, by the means of the covariance matrix, all the dependencies between sensor measurements [2].

Given an observation set D_T, the probability distribution of readings of D_T can therefore be approximated by the multi-dimensional gaussian:

$$P(\mathbf{s}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_{D_T}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{s}-\mu_{D_T})\Sigma_{D_T}^{-1}(\mathbf{s}-\mu_{D_T})^T}$$

where μ_{D_T} and Σ_{D_T} are the mean vector and covariance matrix of the observation set D_T, and $\mathbf{s}$ represents a vector of readings for the 52 sensors. An advantage of this model is that, by using simple matrix computations, it is possible to extract any sensor's mean value together with its variance, possibly conditioned on measurements of other sensors [8]. This allows to easily derive estimates for quadratic and ϵ-based loss functions. We use in the following the ϵ-based loss

TABLE 1: CYCLE CONSTRUCTION ALGORITHM

```
Algorithm Cycle_Construction(S, QB, K, T)

Sort S by increasing order of sensors remaining QB
Size_S ← size(S)
S_1 ← S
K_stage1 ← 1

For (s in 1:Size_S)
   s_current ← S[s]
   used ← False
   For (k in 1:K_stage1)
     new_set ← S_k \ s_current
     If (predictable(S \ new_set, new_set, T))
       S_k ← new_set
     Else
       used ← True
     EndIf
   EndFor
   If (used==False)
     If (K_stage1 < K)
       K_stage1 ← K_stage1 + 1
       S_{K_stage1} ← S_{K_stage1-1}
       Add s_current to S_{K_stage1-1}
     Else
       Add s_current to S_k lowest number of sensors
     EndIf
   EndIf
EndFor

K_stage2 ← K_stage1
While (K_stage2 < K)
   update QB with S_k, 1 ≤ k ≤ K_stage2
   Sort S by increasing order of sensors remaining QB
   K_stage2 ← K_stage2 + 1
   S_{K_stage2} ← S
   For (s in 1:Size_S)
     s_current ← S[s]
     new_set ← S_{K_stage2} \ s_current
     If (predictable(S \ new_set, new_set, T))
       S_{K_stage2} ← new_set
     EndIf
   EndFor
EndWhile

Return the set of S_K
```

function $P(|s_i(t) - \hat{s}_i(t)| > \epsilon) < 1 - \delta$, together with its confidence level δ, which is naturally derivable from the mean vector and covariance matrix of D_T. The error threshold to be defined by the user is therefore the pair (ϵ, δ).

In the following experiments, all results reported are average results obtained through ten-fold cross validations on the entire data set D_T, each validation consisting in keeping readings of 9 days as training set, and readings of the remaining day as test set.

C. Prediction error and average sensor queries

In this first experiment, we varied the error threshold ϵ from 0.1 to 2.5 and the confidence level δ from 0.9 to 0.99, and reported the average number of sensor solicited at each epoch and the overall prediction error in table 2. The cycle length was fixed in this first experiment to 52, i.e. the number of sensors. We observe that on this data set, predictions can be accurately obtained for ϵ as low as 0.5 degree with a confidence level of $\delta = 0.95$. Prediction error estimates are however violated for higher confidence levels, which is explained by the fact that two different estimation methods were used to assess prediction errors. During the cycle construction and the search for prediction subsets, prediction error estimates were derived from the covariance matrix Σ_{D_T}, which are slightly over optimistic, whereas prediction errors reported in table 2 are obtained through the cross validation method.

TABLE 2: AVERAGE SENSOR QUERIES PER TIME UNIT AND PREDICTION ERROR (IN BRACKETS) USING GAUSSIAN MODELS

	δ		
	0.90	0.95	0.99
ϵ=0.1	1 (0)	1 (0)	1 (0)
0.5	0.63 (0.05)	0.72 (0.031)	1 (0)
1	0.47 (0.043)	0.52 (0.032)	0.75 (0.014)
1.5	0.32 (0.04)	0.41 (0.028)	0.53 (0.014)
2	0.21 (0.043)	0.31 (0.024)	0.39 (0.017)
2.5	0.12 (0.042)	0.22 (0.025)	0.31 (0.013)

D. Network lifetime vs cycle length

Figure 3 plots the percentage of sensors with non exhausted query budgets for different cycle lengths, with an inital query budget of 100 for all sensors. The reference is a cycle of length 1, which corresponds to the solicitation of all sensors. After 100 epochs, all sensors have exhausted their query budgets. Plots for cycle lengths of $K = 10$ and $K = 25$ correspond to cycles where energy consumption is not evenly distributed. The cycle length is too small to get an equal distribution of energy, and therefore some sensors run out of energy quickly. This entails that some sensor subsets that were deemed as prediction subsets during the cycle construction phase are no more reliable, and a new cycle should be designed, as justified in 3-C, that takes into account an updated view of sensors' remaining query budgets.

Once the cycle is long enough, from $K = 40$, we see that the algorithm proposed can find a set of prediction subsets such that all sensors are evenly solicited. This is optimal with respect to the network lifetime, as it allows to keep all sensors operational in a quasi homogeneous way. Network lifetime can be extended in this case by a factor close to 3.

E. Network lifetime vs accuracy

We also run a set of experiments illustrating the prolongation in the network lifetime for varying ϵ, with a cycle length of $K = 100$. Results are reported on figure 4. This experiment shows that for tight error thresholds, a subset of sensors may quickly exhaust its energy budget. This subset corresponds to sensors whose readings are not or virtually not predictable by other sensor readings.

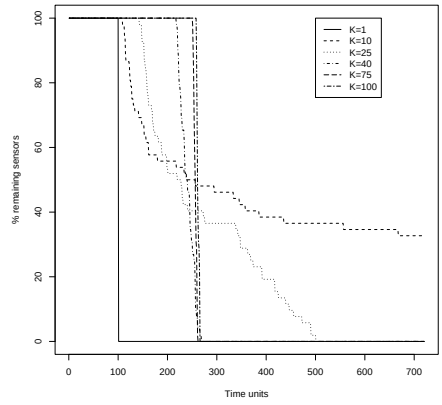

Fig. 3: Sensors lifetime for varying cycle length (Initial QB: 100, $\epsilon = 1.5$, $\delta = 0.95$).

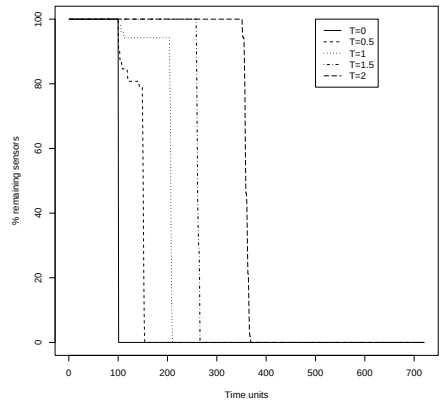

Fig. 4: Sensors lifetime for varying accuracy (Initial QB: 100, $T = \epsilon$, $\delta = 0.95$, $K = 100$).

F. Energy offset

Finally, we illustrate on figure 5 how a round robin cycle on prediction subsets can offset unequal query budgets. Each curve corresponds to a different ratio between sensors whose initial query budget is 100 and sensors whose intial query budget is 50. Sensors with initial query budgets of 50 were drawn at random from the set of sensors. Each plot is an average result for ten different random draws.

For N=52, all sensors have an inital query budget of 50, and network lifetime is extended by a factor close to 3 as was observed in experiment 4-D. For N=0, all sensors have an initial query budget of 100, and network lifetime is extended by the same factor. In between, sensor lifetimes are quasi similar for a given N, which, due to the fact that each curve is an average over ten draws, accounts for an evenly distributed energy consumption among sensors, .

CONCLUSION

This paper lies within the framework of using prediction models for data acquisition in sensor networks. It provided

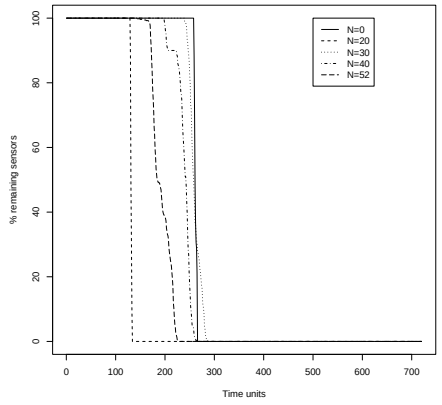

Fig. 5: Sensors lifetime for unequal query budgets (N: number of sensors with initial QB=50, $\epsilon = 1.5$, $\delta = 0.95$, $K = 100$).

a general purpose definition for the notion of predictability of a sensor subset, and proposed relying on a round robin cycle on different prediction subsets to increase the reliability of the model driven monitoring system. The two key contributions of using a cycle are, first, to evenly distribute energy ressources among sensors, and second, to offer a way to check for sensor failure or erroneous models by keeping on collecting readings from all sensors. Experimental results demonstrated the efficiency of the approach with respect to the prolongation of the network lifetime. More importantly, results showed that, if the user defined accuracy is not too strict, it is possible to extend in an homogeneous way lifetimes of all sensors, avoiding anticipated depletion of a subset of sensors. Use of different prediction subsets therefore appears as a promising way for increased reliability of model-driven data acquisition in sensor networks.

ACKNOWLEDGMENTS

This work was supported by the **COMP²SYS** project, sponsored by the Human Resources and Mobility program of the European Community (MEST-CT-2004-505079).

REFERENCES

[1] J. Polastre, R. Szewczyk, D. Culler, "Telos: Enabling ultra-Low power wireless research", In: *Proceedings of the 4th International Conference on Information Processing in Sensor Networks*, Los Angeles, USA, 2005.

[2] A. Desphande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks", In: *Proceedings of the 30th VLDB Conference*, Toronto, Canada, 2004.

[3] *http://db.lcs.mit.edu/labdata/labdata.html*.

[4] G. Bontempi, Y. Le Borgne, "An adaptive modular approach to the mining of sensor network data", In: *Workshop on Data Mining in Sensor Networks*. SIAM SDM April 2005, Newport Beach, USA.

[5] S. Tilak, N. Abu-Ghazaleh, W. Heinzelman, "A taxonomy of wireless micro-sensor network models", *ACM Mobile Computing and Communications Review (MC2R)*, Volume 6, Number 2, 2002.

[6] I. Guyon, A. Elisseeff, "An introduction to variable and feature selection", *Journal of Machine Learning Research, 3:1157-1182*, 2003.

[7] M. Ilyas, I. Mahgoub, *Hanbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*, 1st edition, CRC Press, 2005.

[8] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.