Prediction-based Approaches to Construct the Energy Map for Wireless Sensor Networks*

Raquel A. F. Mini[1†], Badri Nath[2] , Antonio A. F. Loureiro[1]

[1]Department of Computer Science – Federal University of Minas Gerais
Belo Horizonte, MG, 31270-010, Brazil

[2]Department of Computer Science – Rutgers University
Piscataway, NJ, 08854, USA

raquel@dcc.ufmg.br,badri@cs.rutgers.edu,loureiro@dcc.ufmg.br

Abstract. *The key challenge in the design of wireless sensor networks is maximizing their lifetime. The information about the amount of available energy in each part of the network is called the energy map and can be useful to increase the lifetime of the network. In this paper, we address the problem of constructing the energy map of a wireless sensor network using prediction-based approaches. We also present an energy dissipation model that is used to simulate the behavior of a sensor node in terms of energy consumption. Simulation results compare the performance of the prediction-based approaches with a naive one in which no prediction is used. The results show that the prediction-based approaches outperform the naive in a variety of parameters.*

Resumo. *O maior desafio enfrentado no projeto de redes de sensores sem fio é maximizar o seu tempo de vida. A informação sobre a quantidade de energia disponível em cada parte da rede é chamada de mapa de energia e este mapa pode ser útil para maximizar o tempo de vida da rede. Neste artigo, é avaliado o problema da construção do mapa de energia para redes de sensores sem fio utilizando abordagens baseadas em predição. Um modelo de dissipação de energia também é apresentado e utilizado para simular o comportamento de um nodo sensor em termos do consumo de energia. Resultados de simulação comparam o desempenho das abordagens baseadas em predição com uma abordagem simples na qual nenhuma predição é utilizada. Os resultados mostram que para uma variedade de parâmetros, as abordagens baseadas em predição são mais eficazes que as abordagens simples.*

1. Introduction

Since their emergence in the 1970s, wireless networks have become increasingly popular in the computing industry. Currently, there are two variations of wireless networks

*This work has been partially supported by DARPA under contract number N-666001-00-1-8953 and a grant from CISCO systems.
†Supported by CNPq - Brazil.

[Royer and Toh, 1999]. The first is known as the infrastructured network. The bridges for these networks are known as base stations. A mobile unit within these networks connects to and communicates with the nearest base station that is within its communication radius. As the node travels out of range of one base station and into the range of another, a "hand-off" occurs from the old base station to the new, and the node is able to continue communication seamlessly throughout the network. Typical applications of this type of network include cellular network and office wireless local area network.

The second type of wireless network is the infrastructureless network, commonly known as ad hoc network. An ad hoc network has no fixed routers, all nodes are capable of movement and can be connected dynamically in an arbitrary manner, acting as routers, discovering and maintaining routes to other nodes. Typically, each node is a personal information appliance such as a personal digital assistant (PDA) outfitted with a fairly sophisticated radio transceiver. The main goal of an ad hoc network is to form and maintain a connected multi-hop network capable of transporting multimedia traffic between nodes [Sohrabi et al., 2000].

Up to now, ad hoc networks have been studied in the context of high mobility and high power nodes and, furthermore, its size is moderate, typically tens to hundreds of communicating nodes [Sohrabi et al., 2000]. However, recent advances in integrated circuit technology have contributed much to the introduction of wireless micro sensors [Pottie and Kaiser, 2000] which are low-cost devices that can communicate with each other in a wireless manner, have limited computing capability and memory and operate with limited battery power. These sensors can produce a measurable response to changes in physical conditions, such as temperature, pressure, magnetic field, humidity, noise levels, lighting conditions and soil makeup. A network of such sensors linked by a wireless medium or "wireless sensor network" forms a kind of ad hoc network with a new set of characteristics and challenges. The main goal of such network is to perform distributed sensing tasks particularly for applications like environmental monitoring. Their main advantage is their ability to be deployed in almost all kinds of terrain with a hostile environment where it might not be possible to use traditional wired networks.

Unlike conventional wireless ad hoc networks, a wireless sensor network potentially comprises of hundreds to thousands of nodes [Sohrabi et al., 2000]. The sensors have to operate in noisy environments and, in order to achieve good sensing resolution, higher densities are required. Therefore, in a sensor network, scalability is a crucial factor. Different from nodes of a customary ad hoc network, sensor nodes are generally stationary after deployment. Although the nodes are static, these networks still have dynamic network topology. During periods of low activity, the network may enter a dormant state in which many nodes go to sleep to conserve energy. Also, nodes go out of service when the energy of the battery runs out or when a destructive event takes place [Park et al., 2000]. Another characteristic of these networks is that the sensors have limited resources, such as limited computing capability, memory and energy supplies, and they must balance these restricted resources in order to increase the lifetime of the network. In addition, the sensors will be battery powered and it is often very difficult to change or recharge batteries for these nodes. Therefore, in sensor networks, we are interested in prolonging the lifetime of the network, and then the energy conservation is one of the most important aspects to be considered in the design of these networks.

Figure 1: Example of an energy map of a wireless sensor network.

The information about the remaining available energy in each part of the network is called the *energy map* and could aid in prolonging the lifetime of the network. We could represent the energy map of a sensor network as a gray level image as the one illustrated in Figure 1, in which light shaded areas represent regions with more remaining energy, and regions short of energy are represented by dark shaded areas. Using the energy map, a user may be able to determine if any part of the network is about to suffer system failures in near future due to depleted energy [Zhao et al., 2002]. A monitoring node is a special node responsible to collect information from the sensor nodes. Typically, this node is named observer or end user and it is interested in obtaining information from the sensor nodes about the observed phenomenon. The choice of the best location for the monitoring node can be made based on the energy map. We know that nodes near the monitoring node probably will spend more energy because they are used more frequently to relay packets to the monitoring node. Consequently, if we move the monitoring node to areas with more remaining energy we could prolong the lifetime of the network. The knowledge of low-energy areas can also aid in incremental deployment of sensors because additional sensors can be placed selectively on those regions short of resources. Routing protocols can take advantage of the available energy information in each part of the network as well. A routing algorithm can make a better use of the energy reserves if it selectively chooses routes that use nodes with more remaining energy, so that parts of the network with small reserves can be preserved. It can also form a virtual backbone based on connecting high energy islands. Therefore, the energy map is an important information for sensor networks. However, the naive approach, in which each node sends periodically only its available energy to the monitoring node, would spend so much energy due to communications that probably the utility of the energy information will not compensate the amount of energy spent in this process. For that reason, better energy-efficient techniques have to be devised to gather the information about the available energy in each part of a sensor network.

In this paper, we focus on proposing mechanisms to predict the energy consumption of a sensor node in order to construct the energy map of a wireless sensor network. We believe that in many instances the node can predict its energy consumption based on its own past history. If a sensor can predict efficiently the amount of energy it will dissipate in the future, it will not be necessary to transmit its available energy often. This node can just send one message with its available energy and the parameters of the model that describe its energy dissipation. With this information, the monitoring node can update often its local information about the available energy of this node. Clearly the effective-

ness of this paradigm is dependent on the accuracy with which prediction models can be generated. We analyze the performance of probabilistic and statistical models, and compare them with a naive approach in which no prediction is used. In order to evaluate the approaches to construct the energy map, we have to have a clear idea of how is the energy drop in a sensor node. Thus, we also propose an energy dissipation model that is used to simulate the behavior of a sensor node in terms of energy consumption. Simulation results show that the use of prediction-based models decreases the amount of energy necessary to construct the energy map of wireless sensor networks.

The remainder of this article is organized in the following way. In section 2., we briefly survey the related works. Section 3. presents the model that we propose to describe the behavior of a sensor node and consequently to simulate its energy drop. In section 4., we describe two approaches to construct a prediction-based energy map for wireless sensor networks. We evaluate the performance of our approaches in section 5. and conclude by giving directions for our future work in section 6..

2. Related Work

Some researches are exploring issues related to the design of sensors to be as energy-efficient as possible [Asada et al., 1998], [Kahn et al., 1999], [Pottie and Kaiser, 2000], [Rabaey et al., July]. In particular, the WINS [Asada et al., 1998], [Pottie and Kaiser, 2000] and PicoRadio [Rabaey et al., July] projects are seeking ways to integrate sensing, signal processing, and radio elements onto a single integrated circuit. Researches involved in SmartDust [Kahn et al., 1999] aim to design millimeter-scale sensing and communicating nodes.

The energy efficiency is the primary concern in designing good media access control (MAC) protocols for the wireless sensor networks. Another important attribute is scalability with respect to network size, node density and topology. A good MAC protocol should easily accommodate such network changes [Woo and Culler, 2001].

A lot of energy-aware routing schemes have been proposed for wireless sensor networks. *Directed diffusion*, proposed in [Intanagonwiwat et al., 2000], is a new paradigm for communication between sensor nodes. The basic operation of the directed diffusion can be described as following: a sensing task is disseminated throughout the sensor network as an interest for named data. This dissemination sets up gradients within the network designed to draw events. Events start flowing towards the originators of interests along multiple paths. The originator of the interest reinforces one, or a small number of these paths by which the data will be delivery. This work is extended in [Intanagonwiwat et al., 2001] in which a more energy-efficient approach is proposed to construct the data aggregation.

The work proposed by [Zhao et al., 2002] is the most similar to ours. The technique described in that paper tries to obtain the energy map of sensor networks by using an aggregation based approach. A sensor node only needs to report its local energy information when there is a significant energy level drop compared to the last time the node reported it. Energy information of neighbor nodes with similar available energy are aggregated in order to decrease the number of packets in the network. The main difference between the approach proposed in that article and ours is that, in that approach, each node

sends to the monitoring node only its available energy, while in our work each node sends also the parameters of a model that tries to predict the energy consumption in the near future. Then, in our approach, each node sends to the monitoring node its available energy and also the parameters of the model chosen to represent its energy drop. With these parameters, the monitoring node can update locally the current available energy in each node of the network, decreasing the number of energy information packets in the network.

3. Energy Dissipation Model

In this section, we address the problem of modeling the energy dissipation in a sensor node. To our knowledge, there is only one work that has addressed this problem [Zhao et al., 2002]. In this work, two energy dissipation models are proposed. The first one is the *uniform dissipation* model. During a sensing event, each node n in the network has a probability p of initiating a local sensing activity, and every node within a circle of r centered at n consumes fixed amount of energy e. The other one is the *hotspot dissipation* model. In this model, there are h fixed hotspots uniformly distributed randomly on the sensor field. Each node n has a probability of $p = f(d)$ to initiate a local sensing activity, and every node within a circle of r centered at n consumes fixed amount of energy e, where f is a density function and $d = min_{\forall i}\{|n - h_i|\}$ is the distance from n to nearest hotspot. The main drawback of these models is that they do not take into account that the lack of energy of these networks will influence their behaviors. For example, to conserve energy, some sensors have to sleep during some part of the time. Other problems include the assumption that all nodes working in a sensing event will consume the same amount of energy and that all events have the same radius of influence. In this section, we propose a model that tries to represent more realistically the behavior of a sensor network.

The conservation of energy is the paramount issue to be considered in the design of sensor networks. The best way to save energy is to make unused components inactive whenever possible. This can be achieved in a framework in which the nodes have various modes of operation with different levels of activation and consequently different levels of energy consumption and, as soon as possible, they have to go to a mode that consumes less energy. This approach has received much attention and probably, in all kind of sensor networks, the nodes will have to change between different states of activation. Using this idea, we propose a model to describe the behavior of a sensor node and consequently to simulate its energy dissipation. In this model, each node has 4 modes of operation: *state 1*: sensing off and radio off; *state 2*: sensing on and radio off; *state 3*: sensing on and radio receiving; *state 4*: sensing on and radio transmitting.

In this model, the following parameters are used: λ: arrival rate of the events; *sleep_time*: time in which the node will sleep; *sleep_prob*: when a node is not acting in a sensing event, it will be in state 1 with probability *sleep_prob*, and in state 2 with probability (1− *sleep_prob*); *event_radius_min* and *event_radius_max*: the radius of each event will be a random variable uniformly distributed between *event_radius_min* and *event_radius_max*; *event_duration_min* and *event_duration_max*: the duration of each event will be a random variable uniformly distributed between *event_duration_min* and *event_duration_max*; *statei_prob*: probability of being in state *i* during an event; *dist_line*: distance of influence when an information is relayed to the monitoring node.

The behavior of the sensor node can be described by the diagram of Figure 2. At the beginning of the simulation, each node goes to state 1 with probability *sleep_prob* or to state 2 with (1−*sleep_prob*).

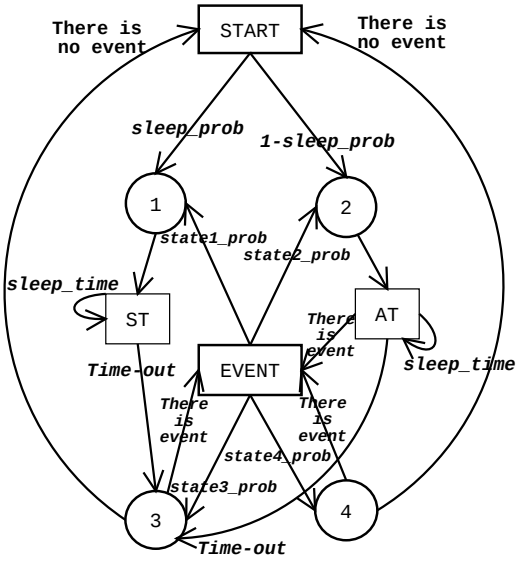


Figure 2: Diagram of the state transition model: 1,2,3 and 4 represent the node modes of operation, ST is a synchronous timer and AT is an asynchronous one.

When a node goes to state 1, it will be sleeping for *sleep_time* seconds. During this period, this node will be saving energy but it will not be able to communicate or to sense any event. After *sleep_time* seconds, the node wakes up and goes to state 3 to see if there is any event for it or if there is any node trying to communicate with it. If there is an event, the node will go to states 1, 2, 3 or 4 with probabilities *state1_prob*, *state2_prob*, *state3_prob* and *state4_prob*, respectively. If there is no event, the node will go to state 1 with probability *sleep_prob* and to state 2 with (1−*sleep_prob*).

If a node goes to state 2, it will be in this state for *sleep_time* seconds, but unlike in state 1, a node that is in state 2 can see the occurrence of an event because in this state the sensing is on. If an event occurs during the *sleep_time* seconds, the node will go to states 1, 2, 3 or 4 with probabilities *state1_prob*, *state2_prob*, *state3_prob* and *state4_prob*, respectively. If the time *sleep_time* ends and no event has happened, the node goes to state 3 to see if there is any node trying to communicate with it and again it will go to state 1 with probability *sleep_prob* and to state 2 with (1−*sleep_prob*).

In this model, the events are simulated by a Poisson process with parameter λ. Therefore, the number of events in each second of simulation is described by the random variable:

$$P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!} \tag{1}$$

When an event occurs, a position (X, Y) is randomly chosen for it. The radius of influence of each event is a random variable uniformly distributed between *event_radius_min* and *event_radius_max* and all nodes within the circle of influence of an event will be affected by it. This means that when these nodes realize that there is an

event for them (the nodes have to be in states 2, 3 or 4), they will go to states 1, 2, 3 or 4 with the probabilities *state1_prob*, *state2_prob*, *state3_prob* and *state4_prob*, respectively. The duration of each event is uniformly chosen between *event_duration_min* and *event_duration_max* seconds. After that time, the data have to be propagated to the monitoring node. We simulate this behavior making all nodes distant *dist_line* for the straight line between the point (X, Y) and the monitoring node go for a short time to state 3 and after to state 4.

The states transition described above tries to capture the behavior of a sensor node specially in terms of energy consumption. As there are no real large sensor networks implemented already, we have no information about the real energy dissipation of a sensor node. But, we believe that, for our purposes, this model can represent the energy drop in an acceptable way.

4. Prediction-based Energy Map

As described earlier, the knowledge of the available energy reserves in each part of the network is important information for sensor networks. In this section, we discuss the possibilities of constructing the energy map using prediction-based approaches. Basically, each node sends to the monitoring node the parameters of the model that describes its energy drop and the monitoring node uses this information to update locally the information about the available energy in each node. The motivation that guided us to this work is that if a node is able to predict the amount of energy it will spend, it can send this information to the monitoring node and no more energy information will be sent during the period that the model can describe satisfactorily the energy dissipation. Then, if a node can efficiently predict the amount of energy it will dissipate in the future time, we can save energy in the process of constructing the energy map of a sensor network.

In order to predict the dissipated energy, we studied two models. In section 4.1., we describe a probabilistic model based on the Markov chains, and, in section 4.2., we present a statistical model in which the energy level is represented by a time series and the ARIMA (Autoregressive Integrated Moving Average) model [Box and Jenkins, 1976] is used to make the predictions.

4.1. Probabilistic Model

In this section, we claim that each sensor node can be modeled by a Markov chain. In this case, the node modes of operation are represented by the states of the a Markov chain and the random variables represent the probability of staying in each state in a certain time. Then, if each sensor node has M modes of operations, each node will be modeled by a Markov chain with M states.

Using this model, in each node, we have a sequence of random variables $X_0, X_1, X_2, ...$ that represents its states during the time. Then, if $X_n = i$, we say that the sensor node is in mode of operation i at time-step[1] n. In addition, at each time the node is in state i, there is some fixed probability, P_{ij}, that it will next be in state j. This probability can be represented by: $P_{ij} = P\{X_{m+1} = j | X_m = i\}$. We can also define the

[1]A time-step is a small amount of time. We suppose that all state transitions occur at the beginning of any time-step.

n-step transition probability, $P_{ij}^{(n)}$, that a node presently in state i will be in state j after n additional transitions [Ross, 1998]: $P_{ij}^{(n)} = \sum_{k=1}^{M} P_{ik}^{(r)} P_{kj}^{(n-r)}$, for any value of $0 < r < n$.

With the knowledge of the probabilities $P_{ij}^{(n)}$ for all nodes and the value of X_0 (initial state of each node), it is possible to estimate some information about the network that can be useful in many tasks. In this work, we will use these probabilities to predict the energy drop of a sensor node. The first step to make this prediction is to calculate for how many time-steps a node will be in a state s in the next T time-steps. If the node is current in state i ($X_0 = i$), the number of time-steps a node will stay in the state s can be calculated by: $\sum_{t=1}^{T} P_{is}^{(t)}$. Also, if E_s is the amount of energy dissipated by a node that remains one time-step in state s, and the node is currently in state i, then the expected amount of energy spent in the next T times, E^T, is:

$$E^T = \sum_{s=1}^{M} (\sum_{t=1}^{T} P_{is}^{(t)}) * E_s \tag{2}$$

Using the value E^T, each node can calculate its energy dissipation rate (ΔE) for the next T time-steps. Each node then sends its available energy and its ΔE to the monitoring node. The monitoring node can maintain an estimation for the dissipated energy in each node by decreasing the value ΔE periodically for the amount of remaining energy of each node. The better the estimation the node can do, the fewer the number of messages necessary to obtain the energy information and consequently the fewer the amount of energy spent in the process of getting the energy map.

4.2. Statistical Model

In this section, we present the statistical model used to forecast the energy level in the sensor nodes. In this model, we represent the energy drop of a sensor node as a time series. A time series is a set of observations x_t, each one being recorded at a specific time t [Brockwell and Davis, 2002]. A discrete-time time series is one in which the set T_0 of times at which observations are made is a discrete set. Continuous-time time series are obtained when observations are recorded continuously over some time interval. There are two main goals of time series analysis [StatSoft, 2002]: identifying the nature of the phenomenon represented by the sequence of observations, and forecasting (predicting future values of the time series variable). In this work, we are interested in using the time series analysis to forecast future values of the available energy in a sensor node. We will use the discrete-time time series in such a way that each node will verify its energy level in a discrete time interval.

We can observe that the time series which represents the energy drop of a sensor node has a clear decreasing trend[2] because we suppose that there is no replacement in the battery and no seasonality[3] because no periodic fluctuation with a well defined frequency will be present in this time series. The decreasing trend will also imply in a decreasing mean and then the energy level will also be a nonstationary time series[4].

[2]Trend refers to a gradual, long-term movement in the data.

[3]Seasonality refers to periodic fluctuations that are generally related to weather factors or to human-made factors such as holidays and vacations.

[4]A stationary time series is one whose statistical properties such as mean, variance, autocorrelation, etc. are all constant over time.

In this work, we will use the ARIMA (Autoregressive Integrated Moving Average) model to predict future values of the time series. The ARIMA models were proposed by Box and Jenkins [Box and Jenkins, 1976] and they consist of a systematic methodology for identifying and estimating models that could incorporate both autoregressive and moving average approaches. This makes ARIMA models a powerful and general class of models [Nist, 2002]. The "Integrated" part of the model is due to the differencing step necessary to make the series stationary.

The first step in developing an ARIMA model is to determine if the series is stationary. When the original series is not stationary, we need to difference it to achieve stationarity. Given the series Z_t, the differenced series is a new series $X_t = Z_t - Z_{t-1}$. The differenced data contain one less point than the original one. Although one can difference the data more than once, a small number of differences is usually sufficient to obtain a stationary time series [Nist, 2002]. The number of differencing applied in the original series is represented by the parameter d.

The next step in the construction of the ARIMA model is to identify the AR terms. An autoregressive model is simply a linear regression of the current value against one or more prior values of the series. The value of p is called the order of the AR model. Then, an autoregressive model of order p can be summarized by the equation:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + Z_t \quad (3)$$

where X_t is the time series, $\phi_1, \phi_2, ..., \phi_p$ are the autoregressive model parameters, and Z_t represents normally distributed random errors.

After defining the differencing and the autoregressive parameters, we have to identify the MA terms. A moving average model is essentially a linear regression of the current value of the series against the random shocks of one or more prior values of the series [Nist, 2002]. The random shocks at each point are assumed to come from the same distribution, typically a normal distribution, with constant location and scale. The distinction in this model is that these random shocks are propagated to future values of the time series. A moving average model of order q is represented by the equation:

$$X_t = Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + ... + \theta_q Z_{t-q} \quad (4)$$

where X_t is the time series, $\theta_1, \theta_2, ..., \theta_q$ are the moving average model parameters and the Z_t are random shocks to the series.

Then, in order to use the ARIMA model we have to identify the values of p (order of the autoregressive model), d (number of differencing required to achieve stationarity), q (order of the moving average model) and the coefficients of the autoregressive and moving average models. So, a time series T_t can be represented by an ARIMA(p,d,q) model if, after differencing this series d times, we find a stationary time series X_t, such that for every t:

$$X_t = \phi_1 X_{t-1} + ... + \phi_p X_{t-p} + Z_t + \theta_1 Z_{t-1} + ... + \theta_q Z_{t-q} \quad (5)$$

When using equation (5), we can predict the value of the time series in time t using the previous values and some random variables that represent the errors in the series. In general, estimation of these parameters is not a trivial task. In [Nist, 2002], [StatSoft, 2002], the authors describe some techniques to help in the process of parameters identification.

5. Simulation Results

In order to analyze the performance of the proposed schemes, we implemented the prediction-based energy maps in the ns-2 simulator. The approaches implemented were: the Markov, in which each node sends periodically to the monitoring node its available energy and its predicted energy consumption rate; and the ARIMA, in which each node sends to the monitoring node its available energy and the parameters of this model. These approaches are compared with the naive one in which each node sends periodically to the monitoring node only its available energy.

In our simulations, we use the energy dissipation model, presented in section 3., to describe the behavior of sensor nodes and consequently to simulate their energy dissipation. Therefore, each node has four modes of operation: state 1 (sensing off, radio off), state 2 (sensing on, radio off), state 3 (sensing on, radio receiving) and state 4 (sensing on, radio transmitting). The values of power consumption for each state were calculated based on information presented in [Hill et al., 2000]: state 1: 25.5 μW, state 2: 38.72 mW, state 3: 52.2 mW and state 4: 74.7 mW. These values will be used throughout all simulations.

In the Markov model, each node sends its available energy and its energy dissipation rate to the monitoring node. To obtain its energy dissipation rate, each node locally calculates its own probabilities, $P_{ij}^{(n)}$. In this case, P_{ij} will be the number of times the node was in state i and went to state j divided by the total number of time-steps the node was in state i. With these probabilities, each node uses equation (2) to find its energy dissipation rate. If each node can predict efficiently its energy dissipation rate, this approach can save energy compared with the naive, because no more energy information packet has to be sent while the energy dissipation rate describes satisfactorily the energy drop in this node.

In the implementation of the ARIMA model, we have to identify the parameters p, d, q and to estimate the coefficients of the AR and MA models. The first step in fitting an ARIMA model is the determination of the order of differencing needed to stationarize the series (parameter d). Normally, the correct number of differencing is the lowest order of differencing that yields a time series which fluctuates around a well-defined mean value and whose autocorrelation function plot decays fairly rapidly to zero, either from above or below [Fuqua, 2002]. If the series still exhibits a long-term trend, lack of tendency to return to its mean value, or if its autocorrelations are positive out to a high number of lags, it needs a higher order of differencing. In general, the optimal order of differencing is often the one at which the standard deviation is lowest [Fuqua, 2002]. In addition, if the lag 1 autocorrelation is -0.5 or more negative, the series may be over-differenced. In our simulation, we choose the smallest value of d that produces the lowest standard deviation in such a way that the lag 1 autocorrelation is not more negative than -0.5. The number of AR and MA terms was found using the autocorrelation and partial autocorrelation functions. The lag at which the partial autocorrelation function cuts off indicates the number of AR terms, and the number of MA terms is determined by the lag at which the autocorrelation function cuts off. The values of the coefficients of the AR and MA models were calculated based on a CSS-ML (minimize conditional sum-of-squares and maximum likelihood) method implemented in [R-Project, 2002].

In all simulations we use the parameter *threshold* that determines the accuracy required or the maximum error acceptable in the energy map. If we define a threshold of

Parameter	Value	Parameter	Value
λ	0.5	*state1_prob*	0.01
sleep_time	10 sec	*state2_prob*	0.2
sleep_prob	0.7	*state3_prob*	0.45
event_radius_min	10 m	*state4_prob*	0.34
event_radius_max	30 m	Threshold	3%
event_duration_min	10 sec	Initial Energy	100 J
event_duration_max	50 sec	Communication Range	20 m
dist_line	20 m	Time-steps	1 sec

Table 1: Default values used in the simulations

3%, a node will send another energy information to the monitoring node only when the error between the energy value predicted by the monitoring node and the correct value is greater than 3%. Each node can locally determine this error by just keeping the parameters of the last prediction sent to the monitoring node. Then, adjusting the value of the threshold, we can control the precision at which the energy maps are constructed.

The numerical values chosen for the base case of our simulations can be seen in Table 1. Unless specified otherwise, these values are used as the parameters throughout the remainder of this work. Moreover, in all simulations, the monitoring node is positioned at the middle of the field at the position $(50, 50)$, all nodes are immobile and can communicate with other nodes within their communication range.

In Figure 3, we plot the correct value of the available energy in a sensor node and the values found using the naive, Markov and ARIMA models during a 1000 second simulation, when the value of λ is 0.1. The other values are the default parameters of Table 1. The scenario used consists of 200 nodes in a 100x100m^2 field in which the average degree of each node is 22.7. This figure shows that making the prediction using the Markov model, during the 1000 seconds of simulation, this specific node had to send four energy information packets (at times 21, 128, 397 and 793 seconds) in order to keep its energy information in the monitoring node with an error no greater than 3% (value of the parameter threshold). Using the ARIMA model, it was necessary to send five energy packets (at times 21, 129, 296, 776 and 889 seconds) and using the naive approach the node sent 11 packets (at times 21, 118, 174, 260, 401, 536, 642, 718, 790, 880 and 946 seconds). Then, in this sensor node, the performance of the ARIMA model was similar to the Markov. We will see in other simulations that the capacities of making prediction of these two models are similar but, in term of energy efficiency, the Markov model is better due to its simplicity and also because, in this model, we only need to send the available energy and the current dissipation rate, while, in the ARIMA, we have to send a lot more information in order to permit the monitoring node to make the prediction.

In order to analyze the performance of the approaches in situations where it is necessary an energy map with very low error (small threshold) and also when we can tolerate a greater error (big threshold), we changed the value of the parameter threshold. We ran the naive, Markov and ARIMA algorithms for 200 nodes in the same scenario described above. Figure 4-a shows the average number of energy information packets that each node had to send to the monitoring node, during 1000 second simulation, in order to construct

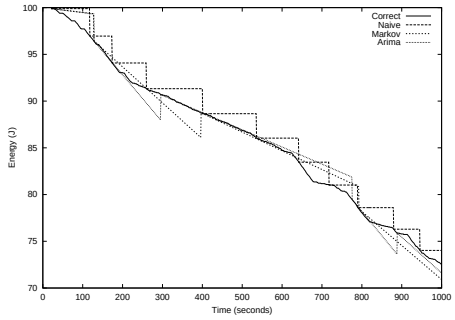
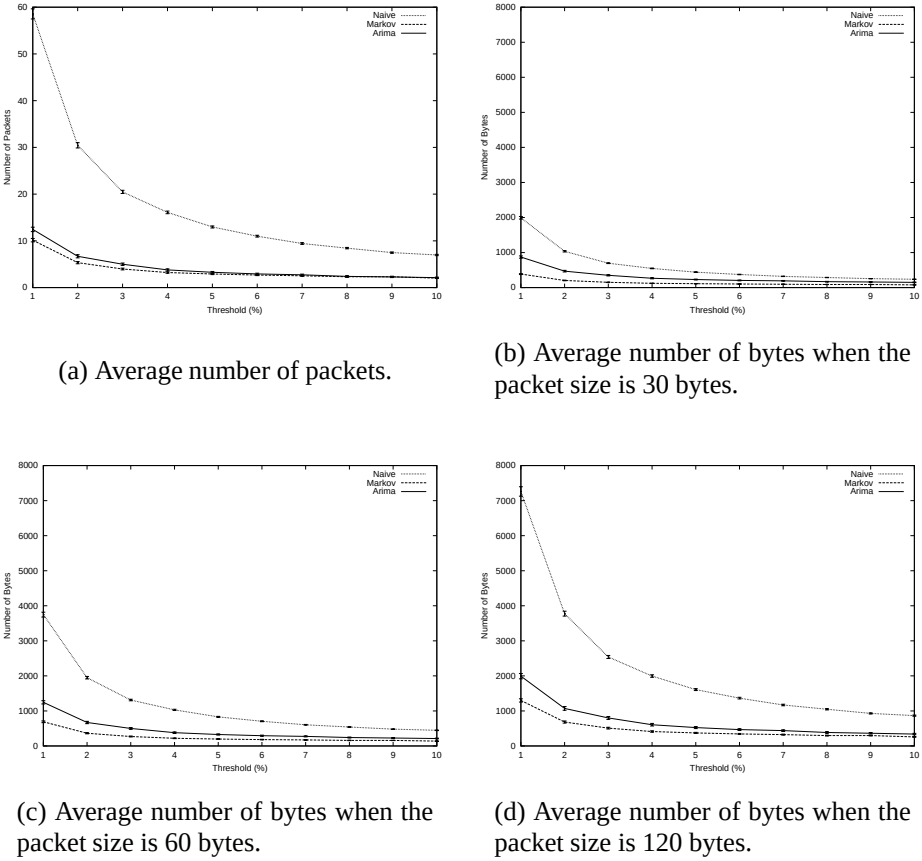


Figure 3: The correct available energy in a sensor node and the values found using the naive, Markov and ARIMA models when the value of λ is 0.1.

an energy map with an error no greater than the corresponding threshold. These results correspond to an average of these values for five different runs and the confident interval showed correspond to a 95% confidence level. We can see that the Markov approach is better than the other two for all values of threshold. But its performance is very close to the ARIMA model, meaning that both approaches have similar power of prediction for all values of threshold. However, the graph of Figure 4-a is not a fair way of comparing the three approaches because when a node, running the naive algorithm, has to send an energy information packet, the size of the extra information required is only 4 bytes (its available energy). In the Markov algorithm, the overhead is of 8 bytes (its available energy and its current power consumption) and in the ARIMA model the overhead is about 40 bytes (with the parameters p, d, q and the coefficients of the AR and MA models). In order to perform a fair comparison between the three approaches, we have to analyze the average number of bytes that each node has to send when running the naive, Markov and ARIMA algorithms. So, the metric used to define energy efficiency will be the number of bytes transmitted. Figure 4-b compares the average number of bytes that each node had to send to the monitoring node if the normal packet size of a sensor network is 30 bytes. In this situation, each time a node has to send its energy information, it will have to send 34 bytes (30 bytes of the normal packet plus 4 bytes of the naive overhead) in the naive algorithm, 38 in the Markov and 70 bytes in the ARIMA. We can see that when we compare the number of bytes instead of the number of packets, the performance of the ARIMA is closer to the naive, and the Markov is still the best of the three. Figures 4-c and 4-d show what happens when the normal size of a packets is 60 and 120 bytes, respectively. As the normal packet size increases, the naive becomes even worse because, in these situations, the overhead of the large amount of information required by the ARIMA has a smaller impact in the total number of bytes sent. Then, for all values of threshold analyzed, the Markov model was more energy-efficient than the other two models, and for sensor networks whose size of the packet is small, the performance of the ARIMA is very close to the naive approach.

Next we altered the value of the parameter λ in order to study the behavior of each approach when the number of events increases. We executed the three approaches using the same scenario described above, during 1000 seconds of simulation. Figure 5-a shows the average number of packets when we increase the number of events in the network. In these simulations, the threshold was fixed in 3%. These results correspond to an average of these values for five different runs and the confident interval showed corre-



(a) Average number of packets.

(b) Average number of bytes when the packet size is 30 bytes.

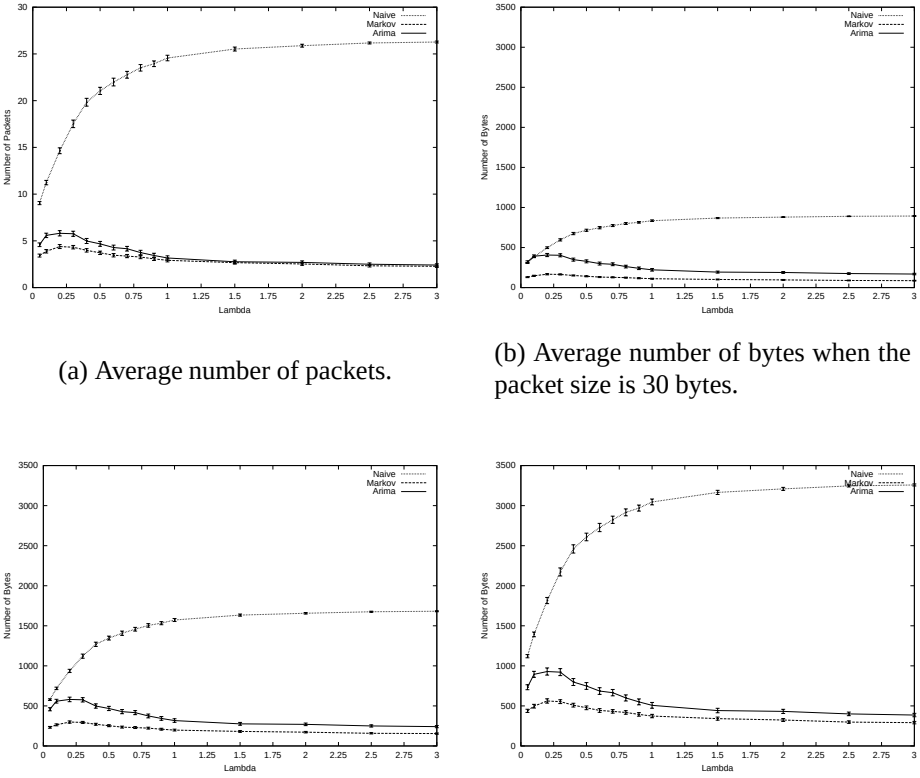
(c) Average number of bytes when the packet size is 60 bytes.

(d) Average number of bytes when the packet size is 120 bytes.

Figure 4: Comparison between the three approaches when we change the value of the threshold.

spond to a 95% confidence level. We can see that the power of making prediction of the Markov is very similar to the ARIMA, but still better for all values of λ. Also, as the network becomes more active, the difference between the number of packets required by the naive and by the prediction-based approaches is getting bigger. Nevertheless, as described above, to do a fair comparison, we have to analyze the number of bytes transmitted by each approach. These results are shown in Figures 5-b, 5-c and 5-d. We can see that the Markov approach is still better than the other two for all values of packet size, and also that when the size of the packet increases, the difference between the number of bytes transmitted by the prediction-based approaches and the naive increases. One interesting fact is that the prediction approaches have a better behavior when the number of events is very small or big. The worst case of these approaches happens for medium values of λ. This means that the fact of having more events does not make the problem of prediction more difficult. The more difficult situations for the prediction approaches are when there is a medium number of events. On the other hand, in the naive approach, as more events happen, more energy will be spent by a node and more often it will have to send energy information packets to the monitoring node. Then, the prediction approaches scale well when the number of events increases or, the power of making prediction does not decrease when the activity of the network increases.

Due to the nondeterministic characteristic of the sensor networks, it is better to



(a) Average number of packets.

(b) Average number of bytes when the packet size is 30 bytes.

(c) Average number of bytes when the packet size is 60 bytes.

(d) Average number of bytes when the packet size is 120 bytes.

Figure 5: Comparison between the three approaches when we change the value of the parameter λ.

perform predictions that are simple both in terms of the computation required to find the parameters of the prediction model, and mainly in terms of the number of parameters that have to be sent to the monitoring node. This feature becomes clear when we compare the two prediction techniques. Even though both present similar capacity of making prediction, the Markov approach is better because, in this model, only one parameter describes the energy dissipation in a sensor node, and consequently only the available energy and the current dissipation rate have to be sent to the monitoring node. So, in the construction of prediction-based energy maps, it is better to use simple models instead of sophisticated predictions that demand a lot of communication between the sensors and the monitoring node.

6. Conclusions and Future Directions

In this article, we have studied the problem of constructing the energy map of wireless sensor networks. We analyzed two prediction-based energy maps based on probabilistic and statistical models. In the prediction-based energy maps, each node tries to estimate the amount of energy it will spend in the near future and it sends this information, along with its available energy, to the monitoring node. Using the energy dissipation model proposed in this work, simulations were conducted in order to compare the performance of two

prediction-based approaches with a naive one, in which only the available energy is sent to the monitoring node. Simulation results indicate that the prediction-based approaches are more energy-efficient than the naive, and also that these approaches are more scalable with respect to the number of sensing events.

As discussed in this work, we believe that prediction-based technique is a good approach to construct the energy map of wireless sensor networks. We intend to extend this work in two main directions. The first one is to decrease the amount of energy necessary to construct the energy map. As it is expected that sensors in close proximity are likely to have correlated energy dissipation, it is not necessary that all nodes in the network send their energy information to the monitoring node. We can sample the energy information in some nodes and estimate that the neighboring nodes have correlated behavior. Sampling this manner, we could decrease the amount of energy spent in the energy map construction. In this case, there is a trade-off between the accuracy of the energy map and the amount of energy used to construct it. If we need only a low resolution map, we can sample a small number of nodes, and then little communication will be necessary to construct the map. On the other hand, if we sample lots of nodes, we will have a very accurate energy map but, more energy will be spent in this process.

Our other objective is to determine the length of time for which a sensor node will be able to sense, rather than merely being concerned with the amount of energy a sensor node has. In other words, we would like to verify what is the expected lifetime of each sensor node. This information can be easily obtained from the prediction-based energy map proposed in this work. However, more valuable is the expected lifetime of each geographical area of the network. For example, we could make the question: "For how many hours will the sensor network be able to sense a target at the position (x, y)?" To answer this question, we will work in the direction of constructing the *lifetime map* of a sensor network.

Acknowledgment

The authors would like to thank the comments and suggestions of Prof. Narayan Mandayam and the members of the DATAMAN group of Rutgers University.

References

Asada, G., Dong, T., Lin, F., Pottie, G., Kaiser, W., and Marcy, H. (1998). Wireless integrated network sensors: Low power systems on a chip. In *European Solid State Circuits Conference*, The Hague, Netherlands.

Box, G. E. P. and Jenkins, G. M. (1976). *Time series analysis: forecasting and control*. San Francisco: Holden-Day.

Brockwell, P. J. and Davis, R. A. (2002). *Introduction to time series and forecasting*. New York : Springer, 2nd edition.

Fuqua (2002). School of business. Forecasting. http://www.duke.edu/ rnau/411home.htm.

Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D., and Pister, K. (2000). System architecture directions for networked sensors. In *Proceedings of the 9th International*

Conference on Architectural Support for Programming Languages and Operating Systems.

Intanagonwiwat, C., Estrin, D., Govindan, R., and Heidemann, J. (2001). Impact of network density on data aggregation in wireless sensor networks. Technical Report 01-750, University of Southern California Computer Science Department.

Intanagonwiwat, C., Govindan, R., and Estrin, D. (2000). Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the sixth annual international conference on Mobile computing and networking*, pages 56–67, Boston, MA USA.

Kahn, J. M., Katz, R. H., and Pister, K. S. J. (1999). Next century challenges: Mobile networking for smart dust. In *In Proceedings of MOBICOM*, pages 271–278, Seattle.

Nist (2002). Nist/sematech - e-handbook of statistical methods. http://www.itl.nist.gov/div898/handbook.

Park, S., Savvides, A., and Srivastava, M. B. (2000). SensorSim: a simulation framework for sensor networks. In *Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 104–111, Boston, MA USA.

Pottie, G. and Kaiser, W. (2000). Wireless integrated network sensors. In *Communications of the ACM*, volume 43, pages 551–8.

R-Project (2002). The R project for statistical computing. http://www.r-project.org/.

Rabaey, J. M., Ammer, M. J., da Silva Jr., J. L., Patel., D., and Roundy, S. (July). Picoradio supports ad hoc ultra-low power wireless networking. *IEEE Computer*, 33(7).

Ross, S. (1998). *A First Course in Probability*. Prentice Hall, fifth edition.

Royer, E. M. and Toh, C.-K. (1999). A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pages 46–55.

Sohrabi, K., Gao, J., Ailawadhi, V., and Pottie, G. (2000). Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, 7:16–27.

StatSoft (2002). Inc. (2002). Electronic Statistics Textbook. Tulsa, OK: StatSoft. http://www.statsoft.com/textbook/stathome.html.

Woo, A. and Culler, D. E. (2001). A transmission control scheme for media access in sensor networks. In *The seventh annual international conference on Mobile computing and networking 2001*, pages 221–235, Rome, Italy.

Zhao, Y. J., Govindan, R., and Estrin, D. (2002). Residual energy scans for monitoring wireless sensor networks. In *IEEE Wilress Communications and Networking Conference (WCNC'02)*, Orlando, FL, USA.