Available online at www.sciencedirect.com

SCIENCE DIRECT®

Ad Hoc Networks 3 (2005) 235–253

www.elsevier.com/locate/adhoc

# Prediction-based energy map for wireless sensor networks

Raquel A.F. Mini *, Max do Val Machado, Antonio A.F. Loureiro, Badri Nath

*Departamento de Ciencia da Computacao, Universidade Federal de Minas Gerais, Caixa Postal 702, 30123-970 Belo Horizonte, MG, Brazil*

Available online 17 August 2004

**Abstract**

A fundamental issue in the design of a wireless sensor network is to devise mechanisms to make efficient use of its energy, and thus, extend its 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 approach. Simulation results compare the performance of a prediction-based approach with a naive one in which no prediction is used. Results show that the prediction-based approach outperforms the naive in a variety of parameters. We also investigate the possibility of sampling the energy information in some nodes in the network in order to diminish the number of energy information packets. Results show that the use of sampling techniques produce more constant error curves.
© 2004 Elsevier B.V. All rights reserved.

*Keywords:* Energy map; Sensor networks; Prediction-based techniques

## 1. Introduction

Wireless sensor networks are those in which nodes are low-cost sensors 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 or magnetic field. The main goal of such networks is to perform distributed sensing tasks, particularly for applications like environmental monitoring, smart spaces and medical systems. These networks form a new kind of ad hoc networks with a new set of characteristics and challenges.

Unlike conventional wireless ad hoc networks, a wireless sensor network potentially has hundreds to thousands of nodes [11]. Sensors have to operate in noisy environments and higher densities are required to achieve a good sensing resolution. Therefore, in a sensor network, scalability is a crucial factor. Different from nodes of a customary

* Corresponding author. Tel.: +55 31 3499 5865; fax: +55 31 3499 5858.
*E-mail address:* raquel@dcc.ufmg.br (R.A.F. Mini).

ad hoc network, sensors are generally stationary after deployment. Although 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 [7]. Another characteristic of these networks is that sensors have limited resources, such as limited computing capability, memory and energy supplies, and they must balance these restricted resources to increase the lifetime of the network. In addition, 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 thus the energy conservation is one of the most important aspects to be considered in the design of these networks.

The information about the remaining available energy in each part of the network is called the *energy map* and can aid in prolonging the lifetime of the network. We can represent the energy map of a sensor network as a gray level image as depicted in Fig. 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 [13]. The knowledge of low-energy areas can aid in incremental deployment of sensors because additional sensors can be placed selectively on those regions short of resources. The choice of the best location for the monitoring node can be made also based on the energy map. A monitoring node is a special node responsible for collecting information from sensor nodes. 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. Therefore, if we move the monitoring node to areas with more remaining energy, we could prolong the lifetime of the network.

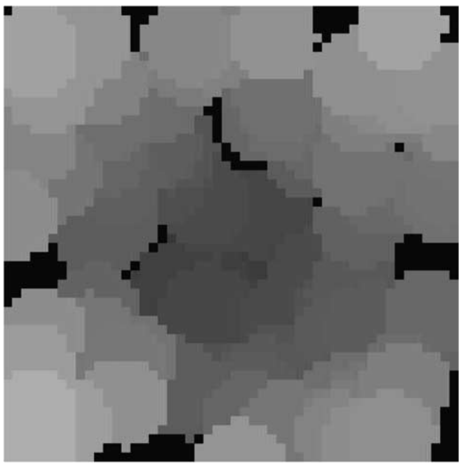


Fig. 1. Example of an energy map of a wireless sensor network.

A routing algorithm can make a better use of the energy reserves if it chooses routes that use nodes with more residual energy. The protocol proposed in [5] is an example of a routing protocol that could take advantage of the energy map. In that work, it is described the trajectory based forwarding protocol that is a new forwarding algorithm suitable for routing packets along a predefined curve. The idea is to embed the trajectory in each packet, and let the intermediate nodes make the forwarding decisions based on their distances from the desired trajectory. If this protocol had the information about the energy map, the trajectory could be planned in order to pass through regions with more energy, thus preserving or avoiding regions of the network with small reserves. Again, the goal here is to make better use of the energy reserves to increase the lifetime of the network.

Other possible applications that could take advantage of the energy map are reconfiguration algorithms, query processing and data fusion. In fact, it is difficult to think of an application and/or an algorithm that does not need to use an energy map. However, the naive approach to construct the energy map, in which each node sends periodically 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 construct the energy map.

In this paper, we focus on proposing mechanisms to predict the energy consumption of a sensor node to construct the energy map of a wireless sensor network. There are situations in which a 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 frequently its available energy. This node can just send one message with its available energy and the parameters of the model that describes its energy dissipation. With this information, the monitoring node can update its local information about the available energy of this node. Clearly the effectiveness of this paradigm depends on the accuracy with which prediction models can be generated. We analyze the performance a probabilistic model, and compare it with a naive approach in which no prediction is used. Simulation results show that the use of the prediction-based model decreases the amount of energy necessary to construct the energy map of wireless sensor networks. We also investigate the energy map construction using sampling techniques in a way that it is not necessary that all nodes send their energy information to the monitoring node. The energy dissipation rate of a node that did not send its energy information packet is estimated using the information received from its neighboring nodes. In situations in which neighboring nodes spend their energy similarly, we can save energy sampling the energy information. Results show that the use of sampling techniques produce more constant error curves, and can reduce the number of energy information packets needed to construct the energy map.

The rest of this paper is organized as follows. In Section 2, we briefly survey the related work. In Section 3, we describe an approach to construct a prediction-based energy map for wireless sensor networks. In Section 4, we present the energy dissipation used to describe the energy consumption in a sensor node. In Section 5, the prediction-based energy map construction is evaluated and compared with the naive approach. In Section 6, we analyze the possibilities of using sampling techniques to construct the energy map. Finally, in Section 7, we conclude giving directions for future work.

## 2. Related work

In [1,4,8,9], the authors explore issues related to the design of sensors to be as energy-efficient as possible. In particular, the WINS [1,8] and Pico-Radio [9] projects are seeking ways to integrate sensing, signal processing, and radio elements onto a single integrated circuit. The SmartDust project [4] aims 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 [12]. In addition, a lot of energy-aware routing schemes have been proposed for wireless sensor networks. *Directed diffusion*, proposed in [3], is a new paradigm for communication between sensor nodes. In this paradigm, the data are named using attribute-value pairs and data aggregation techniques are used to dynamically select the best path for the packets. This enables diffusion to achieve energy savings.

The work proposed in [13] obtains 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 to decrease the number of packets in the network. In [13], each node sends to the monitoring node only its available energy, whereas in our work each node sends also the parameters of a model that tries to predict the energy consumption in the near future. With these parameters, the monitoring node can update locally its information about the current available energy at each node, decreasing the number of energy information packets in the network.

## 3. Prediction-based energy map

As described earlier, the knowledge about the amount of available energy in each part of the network is an important information for sensor networks. A naive solution to construct the energy map is to program each node to send periodically its energy level to the monitoring node. As a sensor network may have lots of nodes with limited resources, the amount of energy spent by this approach is prohibitive. For that reason, better energy-efficient techniques have to be designed to gather the information about the available energy in each part of a sensor network.

In this work, we discuss the possibilities of constructing the energy map using a prediction-based approach. 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 describes satisfactorily the energy dissipation. Thus, 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 a probabilistic model based on Markov chains. In this model, each sensor node can be modeled by a Markov chain. In this case, the node operation modes are represented by the states of a Markov chain and, if a sensor node has $M$ operation modes, it is modeled by a Markov chain with $M$ states. Using this model, at each time the node is in state $i$, there is some fixed probability, $P_{ij}$, that, in the next time-step, [1] it will be at state $j$. This probability can be represented by $P_{ij} = P\{X_{m+1} = j|X_m = i\}$. We can also define the $n$-step transition probability, $P_{ij}^{(n)}$, that a node currently in state $i$ will be in state $j$ after $n$ additional transitions [10]: $P_{ij}^{(n)} = \sum_{k=1}^{M} P_{ik}^{(r)} P_{kj}^{(n-r)}$, for any value of $0 < r < n$.

With the knowledge of probabilities $P_{ij}^{(n)}$ for all nodes and the 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 state $s$ in the next $T$ time-steps. If the node is in state $i$, 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, $E^T(i)$, is

$$E^T(i) = \sum_{s=1}^{M} \left( \sum_{t=1}^{T} P_{is}^{(t)} \right) \times E_s. \quad (1)$$

Using the value $E^T(i)$, each node can calculate its energy dissipation rate ($\Delta E$) for the next $T$ time-steps. Each node then sends its available energy and its $\Delta E$ to the monitoring node. The monitoring node maintains an estimation for the dissipated energy at each node by decreasing the value $\Delta E$ 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, thus, the fewer the amount of energy spent in the process of getting the energy map.

In this work, each node locally constructs its own transition probability matrix based only on its past history. In this case, $P_{ij}$ will be the number of times a 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 this matrix, each node uses Eq. (1) to find its energy dissipation rate. If the prediction is good, this approach can save energy compared with the naive solution, because an energy information packet is not transmitted while the energy dissipation rate describes satisfactorily the energy drop in this node. In Section 5.4, we discuss the computational cost of this approach.

## 4. Energy dissipation model

When simulation is used to analyze the performance of the energy map construction or any

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

other energy related problem, we have to know how the energy dissipation happens in sensor nodes. To this end, in this work, we use the *state-based energy dissipation model* (SEDM) to model the energy drop in sensor nodes.

In the SEDM, nodes have various operation modes with different levels of activation and, thus, different levels of energy consumption. In this model, each node has four operation modes: *mode* 1: sensing off and radio off; *mode* 2: sensing on and radio off; *mode* 3: sensing on and radio receiving; *mode* 4: sensing on and radio transmitting. The transitions between these modes are described by the diagram of Fig. 2. In that diagram, the operation modes are represented by states 1, 2, 3 and 4. In addition, it was necessary to represent more two states $2'$ and $3'$. The state $i'$ also represents the operation mode $i$. The only difference is that when a node goes to state $i$, it always starts a timer, whereas in state $i'$, it verifies if is there any event for it. In terms of energy consumption, state $i$ is exactly the same as state $i'$. However, the behaviors of states $i$ and $i'$ are different.

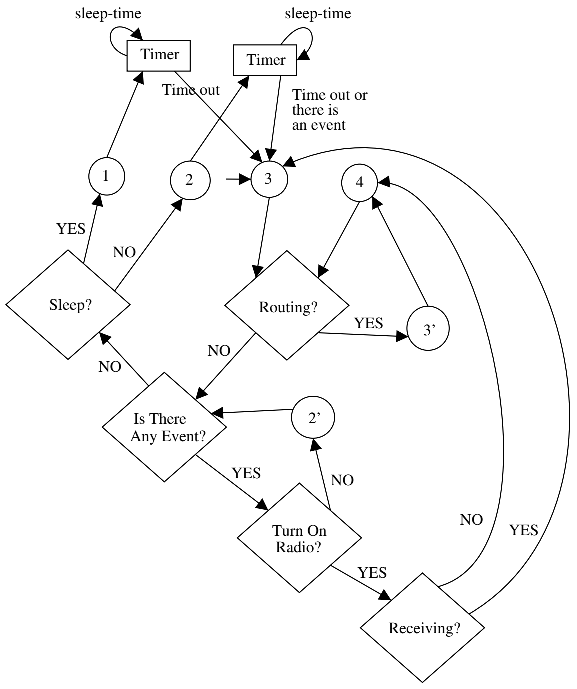


Fig. 2. Diagram of the state-based energy dissipation model.

The diagram of Fig. 2 shows the "commands" performed along the path (transition) between states. It means that whenever a node changes its current state it performs tests and actions until the new state is reached. The tests are: "routing"—checks whether a message has to be routed; "sleep"—determines whether the node will sleep or not; "is there any event"—determines whether a new sensing event is present; "turn on radio"—determines whether the radio must be turned on or not; and "receiving"—determines whether the radio must receive or transmit. "Timer" is an action that starts a timer. The outcome of each test depends on a probabilistic parameter associated with the test. These transitions try to capture the behavior of a sensor node, specially in terms of energy consumption.

It is important to point out that the tests are tied to the events. Clearly, the outcome of the test "is there any event" is always yes when an event is detected, and no otherwise. The "routing" test is yes when the node has to route some sensed information that happened in other part of the sensor field. Thus, this test is also influenced by the events. The "receiving" test depends also on the characteristics of the event. Its value is influenced by the degree of cooperation needed by the application. The "sensing" test is called only if there is no event in the area of the node. If no event happens, this test will depend on the degree of coverage needed by the application. The greater the value of *sleep-prob*, the smaller the coverage.

In the SEDM, two types of arrival models are simulated. In the first one, the event arrival is modeled by a Poisson process with parameter $\lambda$. This process is appropriate to model events that happen randomly and independently from each other. In the second model, the event arrival is modeled by a Pareto distribution. This distribution has a heavy-tailed property that implies that small occurrences are extremely common, whereas large instances represent very few occurrences. When a Pareto distribution is used to simulate the inter-arrival time of the events, they will happen in bursts. This is because most of the inter-arrival time will be small, meaning that we have lots of events. However, the occurrence of large inter-arrival time cannot be neglected, and thus it is possible to have long periods of time without any event. The use of Poisson process and Pareto distribution to model the event arrival comes from the fact that these are the most common models used in traffic generation problems.

When an event arrives, a position $(X, Y)$ is randomly chosen for it, and its behavior is described by an event that is static and has a fixed size. The radius of influence of an event is a random variable uniformly distributed in [*event-radius-min*, *event-radius-max*] meters, and all nodes within the circle of influence of an event will be affected by it. Its duration is uniformly distributed in [*event-duration-min*, *event-duration-max*] seconds.

## 5. Simulation results

In this section, we present the simulation results of the prediction-based approach to construct the energy map and the naive solution. Section 5.1 describes the operation of the analyzed approaches. Section 5.2 analyzes the performance of the approaches when the number of events is changed. Finally, Section 5.3 shows the results when we change the accuracy in which the energy maps are constructed.

### *5.1. Basic operation*

In order to analyze the performance of the proposed schemes, we implemented the prediction-based energy maps in the ns-2 simulator [6]. The MAC protocol used was the default MAC protocol of ns-2. It is a simplified version of the 802.11 protocol. We use no particular routing algorithm, but analyze the effect of the routing process. The energy information packet was routed to the monitoring node using an aggregation tree in which the monitoring node is the root. In fact, the operating modes of a node were defined based on the Berkeley's weC mote information. The protocol stack used in the simulation does not influence these values.

We implemented the Markov chain, in which each node sends periodically to the monitoring node its available energy, and its predicted energy consumption rate, and compare it with the naive one in which each node sends periodically to the monitoring node only its available energy.

In this work, we consider a sensor network with static and homogeneous nodes, replacement of battery is unfeasible or impossible, and there is only one static monitoring node with plenty of energy. Nodes are deployed randomly forming a high-density network in a flat topology. Events are static and their duration and radius of influence are randomly chosen. We simulate an event-driven network in which sensors report information only if an event of interest occurs. In this case, the monitoring node is interested only in the occurrence of a specific event or set of events. The communication model among sensors is cooperative in the sense that is beyond the relay function needed for routing, and sensors communicate with each other to disseminate information related to the event. Besides, we used the energy dissipation model presented in Section 4.

The accuracy required or the maximum error acceptable in the energy map is controlled by the parameter *threshold*. For instance, if its value is 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. The parameter *threshold* is used in both approaches to construct the energy map. Thus, even in the naive solution, another energy information packet is sent when the error is greater than the parameter *threshold*. Thus, in the naive approach, the *threshold* means the drop in the last energy value sent to the monitoring node.

In our simulations, the values of power consumption for each state were calculated based on information presented in [2]: Mode 1: 28.50 μW, Mode 2: 38.72 mW, Mode 3: 52.20 mW and Mode 4: 74.70 mW. These values will be used throughout all simulations.

The numerical values chosen for the base case of our simulations can be seen in Table 1. Unless specified otherwise, these values are used in all simulations in this work. In this scenario, each node has an average of 23.6 neighbors. The

Table 1
Default values used in simulations

| Parameters | Value |
|---|---|
| Number of nodes | 100 |
| Initial energy | 100 J |
| Communication range | 15 m |
| Sensor field size | $50 \times 50\,\text{m}^2$ |
| *threshold* | 3% |
| *event-duration-min* | 5 s |
| *event-duration-max* | 50 s |
| *event-radius-min* | 5 m |
| *event-radius-max* | 15 m |

monitoring node is positioned at the center of the field at position (25, 25), all nodes are immobile, and can communicate with other nodes within their communication range. We assume that the monitoring node knows the initial energy at each sensor. Before a node sends its first energy information packet, the monitoring node assumes that its power consumption is the average of the power consumption of all states. We also assume that nodes spend energy at the rate of 41.41 mW that is the average of power consumption of the four operation modes. In addition, the results of all simulations were obtained as an average of 33 runs and they have a 95% confidence level.

In Fig. 3a, we plot the correct value of the available energy in a sensor node and the values found using the naive and Markov models during a simulation of 1000 s, when the event arrival is modeled by a Poisson process, and $\lambda = 0.001$. This figure shows that making the prediction using the Markov model, during 1000 s of simulation, this specific node had to send three energy information packets (at times 153, 514 and 929 s) to keep its energy information in the monitoring node with an error no greater than 3% (*threshold*). Using the

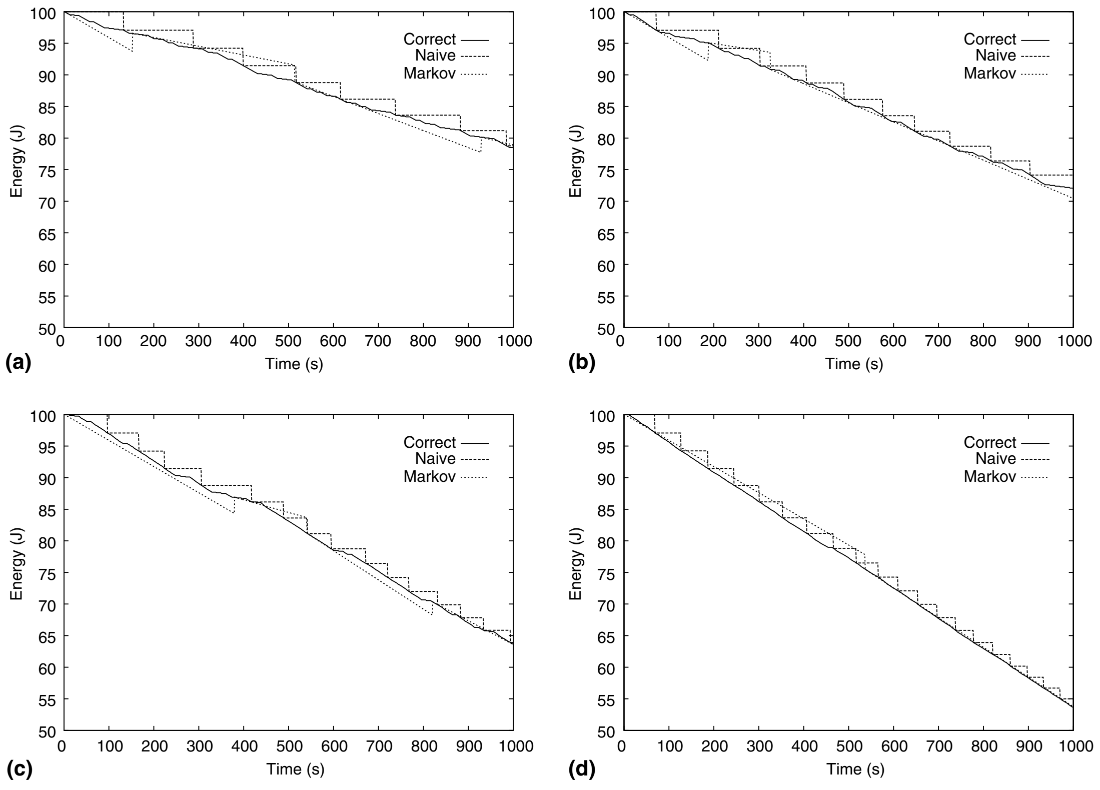


Fig. 3. The correct available energy in a sensor node and the values found using the naive and Markov models for different values of $\lambda$. (a) $\lambda = 0.001$, (b) $\lambda = 0.1$, (c) $\lambda = 0.5$, (d) $\lambda = 0.9$.

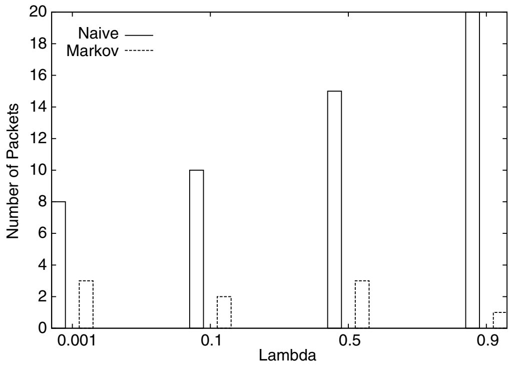


Fig. 4. Number of energy information packets the node of simulations of Fig. 3 had to send.

naive approach, the node sent eight packets (at times 133, 288, 399, 517, 616, 738, 883 and 985 s) to keep its error smaller than 3%. It is important to point out that both approaches use the parameter *threshold* to decide when a new energy information packet has to be sent. Fig. 3b–d, shows what happens in the same sensor node when we change the number of events in the network. In Fig. 4, we plot the number of energy information packets this node had to send in simulations of Fig. 3. We can see that the number of packets sent when using the prediction-based model is less than when using the naive approach.

## 5.2. Changing the number of events

In this section, we analyze the performance of the energy map construction when we change the number of events in the network. Firstly, we use a Poisson process to model the event arrival, and the value of parameter $\lambda$ is changed. Secondly, a Pareto distribution is used, and the parameter $a$ is modified. In Fig. 5, we show the average number of events generated when parameters $\lambda$ and $a$ are changed.

Using the Poisson process to describe the event arrival, we executed the two approaches in the same scenario described above, during 1000 s of simulation. Fig. 6 shows the average number of energy information packets that each node had to send to the monitoring node to construct an energy map with an error no greater than 3%. We can see that, for all values of $\lambda$, the naive spends more energy information packets than the prediction-based approach. In addition, when the network becomes more active, the difference between the number of packets required by the naive and by the prediction-based approach is larger, meaning that the Markov is more scalable in relation to the number of events in the network than the naive solution.

Nevertheless, the graph of Fig. 6 is not a fair way of comparing the two 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 2 bytes (its available energy) and, in the Markov algorithm, the overhead is of 4 bytes (its available energy and its current power consumption). In order to perform a fair comparison between the two approaches, we have to analyze the average number of bytes that each node has to send when running

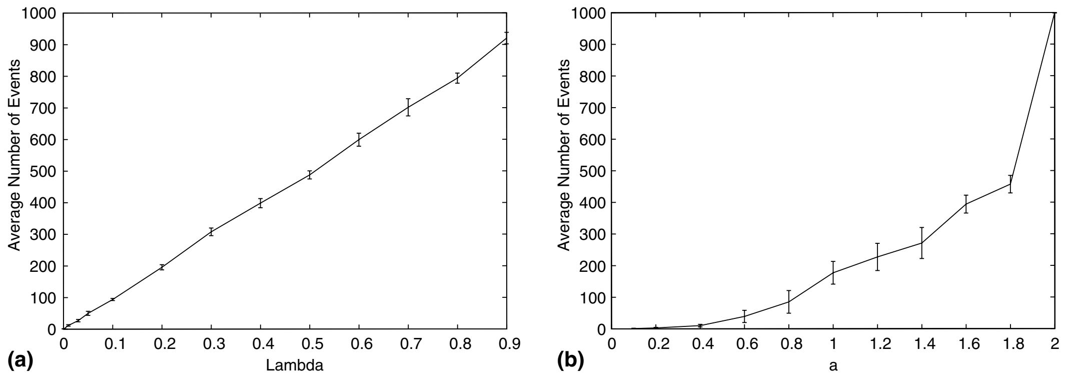


Fig. 5. Average number of events when parameters $\lambda$ and $a$ are changed. (a) Poisson process, (b) Pareto distribution.

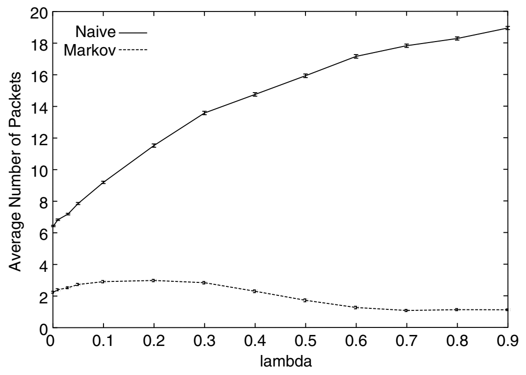


Fig. 6. Average number of packets for different values of $\lambda$, threshold = 3%.

the naive and Markov algorithms. Thus, the metric used to define energy efficiency will be the number of bytes transmitted. Fig. 7a compares the average number of bytes that each node had to send to the monitoring node without taking into account the overhead of the packet header. In this situation, we use piggybacking to send the energy information. We can see that the number of bytes that the naive has to send is even larger than the number sent by the naive approach.

In Fig. 7b, we plot the total number of bytes each node had to send considering that the packet header is of size 30 bytes. In this situation, each time a node has to send its energy information, it will send 32 bytes (30 of header and 2 of payload) in the naive algorithm, and 34 bytes (30 of header and 4 of payload) in the Markov. We can see that, in this case, the Markov is still the best of the two. Fig. 7c and d shows what happens when the packet header is of size 60 and 90 bytes, respectively. In all situations, the Markov approach is still better than the naive for all values of packet size.

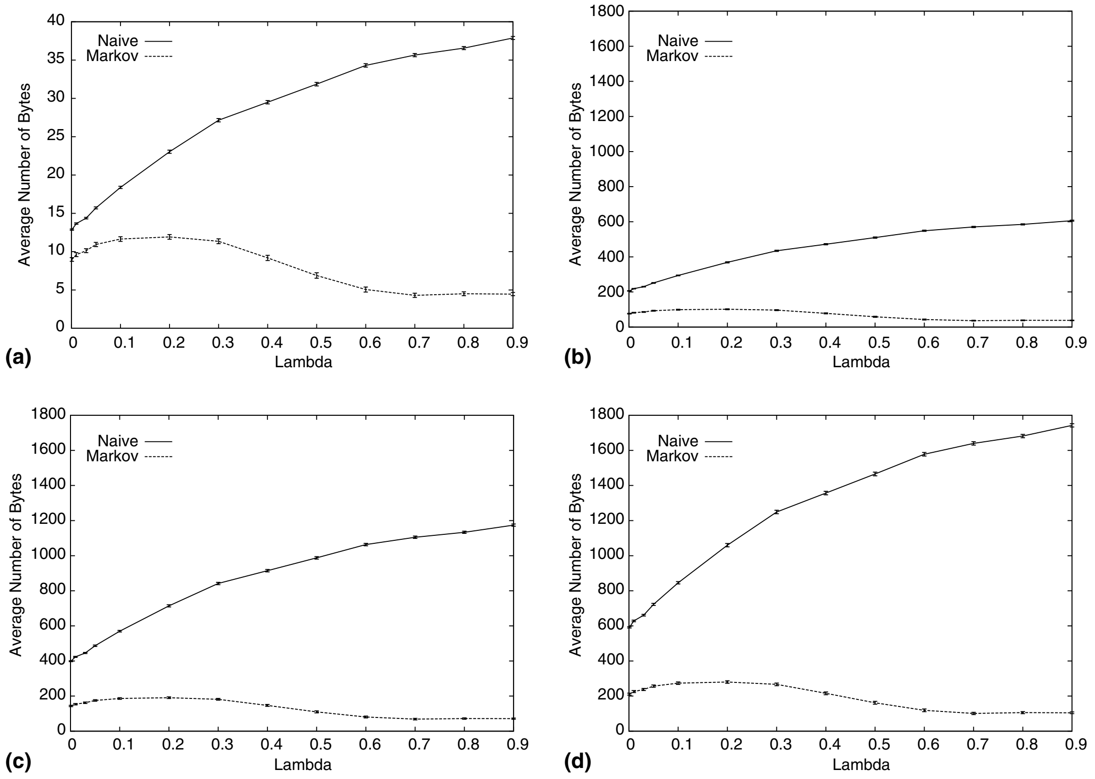


Fig. 7. Average number of bytes for different values of $\lambda$, threshold = 3%. (a) Using piggybacking to send the data, (b) packet header of 30 bytes, (c) packet header of 60 bytes, (d) packet header of 90 bytes.

In the next simulations, we use the Pareto distribution to describe the behavior of the event arrival in the network. Fig. 8 shows the average number of energy information packets each node had to send to the monitoring node. In Fig. 9 we analyze the number of bytes transmitted by each approach.

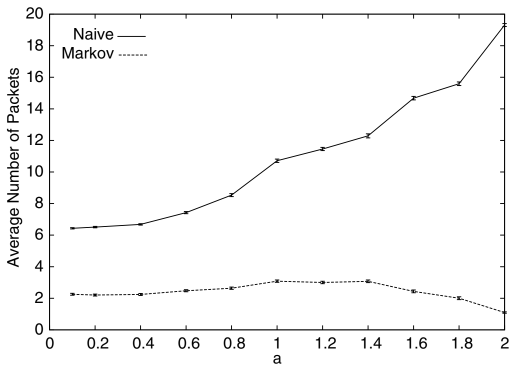


Fig. 8. Average number of packets for different values of *a*, threshold = 3%.

We can observe that the results of the Pareto distribution are similar to the Poisson process. Observing these graphs and the average number of events generated in each model (Fig. 5), we can say that the event arrival model does not influence the performance of the prediction-based energy map construction. In fact, a uniformly distributed event arrival model tends to be slightly easier to be predicted because the future is more likely the present. However, this trend is faintly, and probably will be more observed in long time simulations.

Notice that the prediction approach has a better behavior when the number of events is big or small. The worst case of this approach happens for medium values of number of events. Using

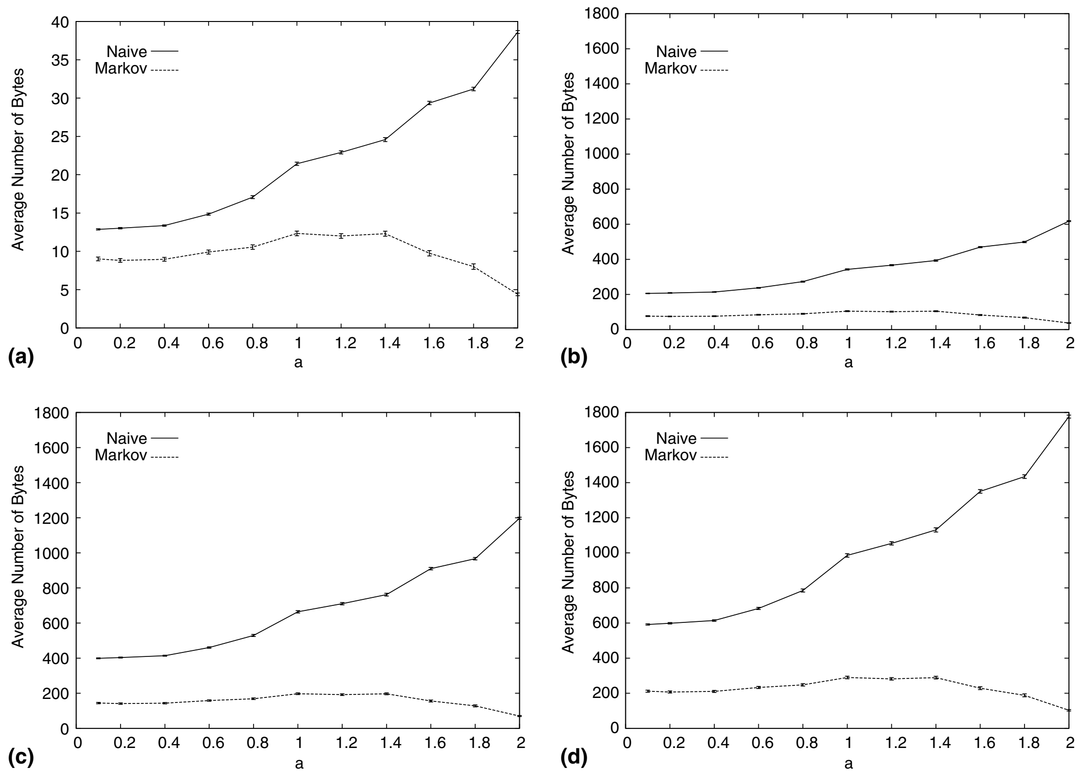


Fig. 9. Average number of bytes for different values of *a*, threshold = 3%. (a) Using piggybacking to send the data, (b) packet header of 30 bytes, (c) packet header of 60 bytes, (d) packet header of 90 bytes.

the Poisson model, the worst case for the Markov is $\lambda = 0.2$, and using Pareto, the worst is when $a = 1.4$. 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 approach happens when there is a medium number of events. In the naive approach, the spent energy is proportional to the number of events since a node will have to send energy information packets more often to the monitoring node. Thus, the prediction-based approach scales well when the number of events increases or, the power of making prediction is improved when the activity of the network increases.

## 5.3. Changing the energy map precision

In order to analyze the performance of the approaches in situations where it is necessary an energy map with a 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 and Markov algorithms for 100 nodes in the same scenario described above, using a Poisson process to model the event arrival. In these simulations, we analyze the worst case for the Markov model that is when the value of $\lambda$ is 0.2.

Fig. 10 shows the average number of energy information packets that each node had to send to the monitoring node, during a simulation of 1000 s, to construct an energy map with an error no greater than the corresponding *threshold*. We

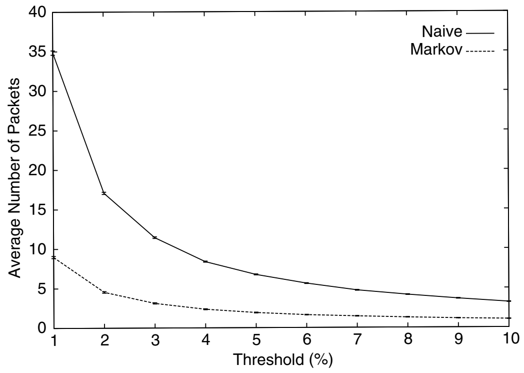


Fig. 10. Average number of packets for different values of *threshold*, $\lambda = 0.2$.

can see that, the Markov approach is better than the naive for all values of *threshold*. Even when we compare the number of bytes instead of the number of packets, the Markov is better than the naive solution. This comparison is shown in Fig. 11.

Fig. 11a compares the average number of bytes that each node had to send to the monitoring node when piggybacking is used to send the energy information. Fig. 11b–d, show the average number of bytes when the packet header has 30, 60 and 90 bytes, respectively. We can see that, for all values of *threshold* analyzed, the Markov model was more energy-efficient than the naive. Recall that results shown in this section represent the worst case for the Markov model. For all other values of $\lambda$, the difference, in terms of energy consumption, between this model and the naive is even higher.

## 5.4. Computational cost of the Markov model

In this section, we analyze the number of operations executed by each sensor node in order to construct the energy map using the Markov model. To construct the prediction-based energy map, each node has to maintain its own probability matrix. This matrix is updated at each time-step of the simulation to keep track of its operation modes. Besides, at each time-step the node verifies if the error in the energy information is greater than the parameter threshold. The number of sums/subtractions, multiplications/divisions, comparisons and assignments performed to execute these tasks is 3, 2, 1 and 3, respectively.

When the error in the energy information reaches the parameter threshold, another energy information packet is sent. In this case, a new value of $E^T(i)$ has to be calculated. The total number of operations executed in this calculation depends on the value of $T$ and on the number of energy information packets sent. In our simulations we used $T = 5$. Table 2 shows the average number of operations executed at each time-step of simulation for some values of $T$. In the analysis of the best and worst cases, we use the simulation results presented in Section 5.2. In the best case, we consider that only one energy information packet is

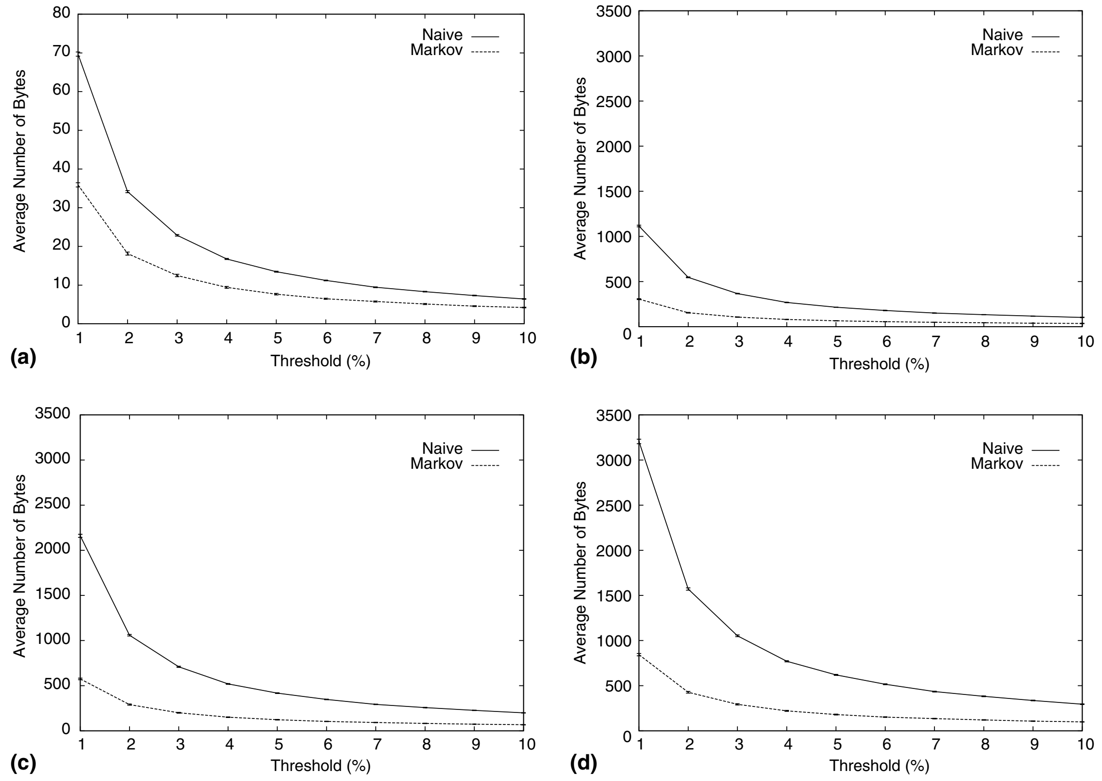


Fig. 11. Average number of bytes for different values of *threshold*, $\lambda = 0.2$. (a) Using piggybacking to send data, (b) packet header of 30 bytes, (c) packet header of 60 bytes, (d) packet header of 90 bytes.

Table 2
Average number of operations performed at each time-step of simulation

| $T$ | Scenario | Operations | | Comparisons | Assignments |
|---|---|---|---|---|---|
| | | + − | × / | | |
| 1 | Best case ($N = 1$) | 3.0 | 2.0 | 1.0 | 3.0 |
| | Worst case ($N = 4$) | 3.0 | 2.0 | 1.0 | 3.1 |
| 5 | Best case ($N = 1$) | 3.2 | 2.1 | 1.1 | 3.1 |
| | Worst case ($N = 4$) | 3.8 | 2.3 | 1.5 | 3.6 |
| 10 | Best case ($N = 1$) | 3.4 | 2.2 | 1.3 | 3.3 |
| | Worst case ($N = 4$) | 4.7 | 2.6 | 2.1 | 4.3 |
| 50 | Best case ($N = 1$) | 5.2 | 2.8 | 2.5 | 4.7 |
| | Worst case ($N = 4$) | 11.9 | 5.2 | 7.1 | 9.7 |

sent during 1000 s of simulation. In the calculation of the worst case scenario, we use the results of Figs. 6 and 8. These results show that, in the worst case, the Markov prediction sends less than four energy information packets during 1000 s of simulation. Thus, the best case was obtained considering that an energy information packet is sent during 1000 s of simulation and, in the worst case, four energy information packets are sent during the same amount of time.

## 6. Energy map construction using sampling technique

In this section, we analyze the use of sampling techniques to construct the energy map. In some sensing applications, neighboring nodes tend to spend their energy similarly. In such situations, we can use sampling techniques in a way that it is not necessary that all nodes send their energy information to the monitoring node. The energy dissipation rate of a node that did not send its energy information packet is estimated using the information received from its neighboring nodes. Simulation results compare the performance of a sampling approach with the Markov model presented in Section 3. Results show that the use of sampling techniques produce more constant error curves, and that these approaches can reduce the number of energy information packets needed to construct the energy map.

This section is organized as follows. Section 6.1 presents a sampling model used to determine when each node will send its energy information packet, and how the energy consumption rate of a node that did not send its energy information is estimated by the monitoring node. Section 6.2 presents simulation results that compare the sampling technique proposed in this section with the original Markov model.

### 6.1. Sampling model

In the original Markov model, when the error between the energy in a sensor node and the corresponding value in the monitoring node is greater than a *threshold*, an energy information packet is always sent to the monitoring node. We define the sampling model in such a way that, when the error reaches the value of *threshold*, an energy information packet is sent with probability $p$. Using this idea, we can consider the original Markov a special case of the sampling model in which the value of $p$ is always 1.

The choice of the parameter *threshold* has to be done locally without any communication between sensor nodes. The value of $p$ can be defined statically or dynamically. In both cases, a constant $d$, that represents the sampling degree, is defined. This constant determines the initial value of probability $p$. In the static sampling, $p$ is always equals to $d$ during all simulation. In the dynamic sampling, its value increases whenever the error reaches the *threshold* and no energy packet is sent. Thus, the larger the error, the larger the value of $p$ and, consequently, the larger the probability of a node to send its energy information packet. To this end, we define probability $p$ according to the following equation:

$$p = d + (1 - d) \times \left(1 - \frac{k}{k + n}\right) \tag{2}$$

where $k$ determines the speed that probability $p$ reaches 1. For small values of $k$, $p$ reaches asymptotically 1 faster. Furthermore, $n$ is the number of times the error reached the *threshold*. Notice that the updating process of $p$ is memory-less. When a new energy information packet is sent, $n$ goes to zero and $p$ is restored to its initial value (the value of $d$ as mentioned above). Fig. 12 illustrates the value of $p$ for different values of $k$ when $d = 0.4$.

The sampling technique described in Eq. (2) diminishes the number of packets used in the map construction, and increases its error. To minimize the error, the monitoring node has to estimate the energy consumption rate of nodes that did not send their energy information packet. We suppose that a node and its neighbors spend energy in a similar way. When the monitoring node receives an energy packet, it uses interpolation to update the energy consumption rate of its

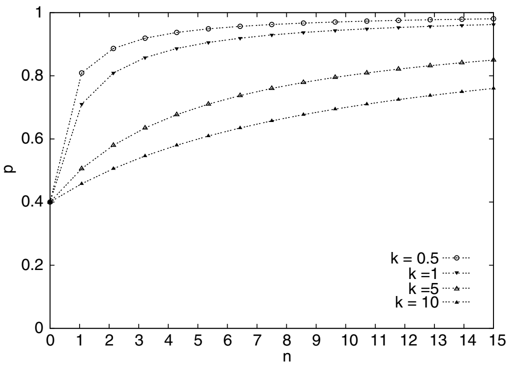


Fig. 12. Probability $p$ for $d = 0.4$.

neighboring nodes of the received packet. This update considers the last consumption rate sent by the node, named $c_{\text{node}}$, and the average energy consumption rate of its neighboring nodes that sent their energy information packet after this node sent the packet, named $c_{\text{neighbors}}$. The trade-off between this two pieces of information is defined by Eq. (3) that determines the weight of the energy consumption rate of the neighboring nodes. In this equation, $n_{\text{interpolations}}$ represents the number of interpolations executed for the node:

$$p_{\text{neighbors}} = (1-d) + d \times \left(1 - \frac{k}{k + n_{\text{interpolations}}}\right). \tag{3}$$

Therefore, when the monitoring node receives an energy information packet, it updates the consumption rate of all neighboring nodes of this packet. This new consumption rate, named $c_{\text{estimated}}$, is defined by Eq. (4):

$$c_{\text{estimated}} = c_{\text{neighbors}} \times p_{\text{neighbors}} + c_{\text{node}} \times (1 - p_{\text{neighbors}}). \tag{4}$$

The goal of Eq. (4) is to update the node consumption rate with a more recent information received from its neighbors. The use of the value $n_{\text{interpolations}}$ in Eq. (3) is justified because, as the node information is estimated several times in the monitoring node, the last energy packet information values loose significance. Consequently, the more recently the energy packet is, the more expressive its value in the map. The value of $p_{\text{neighbors}}$ depends also on the sampling degree. The smaller this value is, the greater the value of $p_{\text{neighbors}}$. It is important to point out that Eq. (3) is only used in the dynamic approach. In the static model, the same equation is $p_{\text{neighbors}} = (1-d)$.

In some situations, when sampling is used, the error from the point of view of the node is smaller than from the point of view of the monitoring node. This happens when an energy information packet has to be sent and, due to the sampling probability $p$, it is not. In this case, from the point of view of the node, the error is zero and the value of $p$ is increased, whereas from the point of view of the monitoring node the error continues increasing. However, the total error is evaluated by using the point of view of the monitoring node.

## 6.2. Simulation results

We implemented the energy map construction using sampling in the ns-2 simulator [6] and compared it with the original Markov model. Unless specified otherwise, the default values used in simulations of this section are the same defined in Section 5.1. Besides, in all simulations of this section, the Poisson process with $\lambda = 0.2$ is used to model the event arrival.

Our goal, in the first simulation, is to analyze the total number of energy information packets sent using the original Markov and the sampling technique for the following values of $d$: 0.2, 0.4, 0.6 and 0.8. Fig. 13 shows these results for the static and dynamic sampling models. As it was

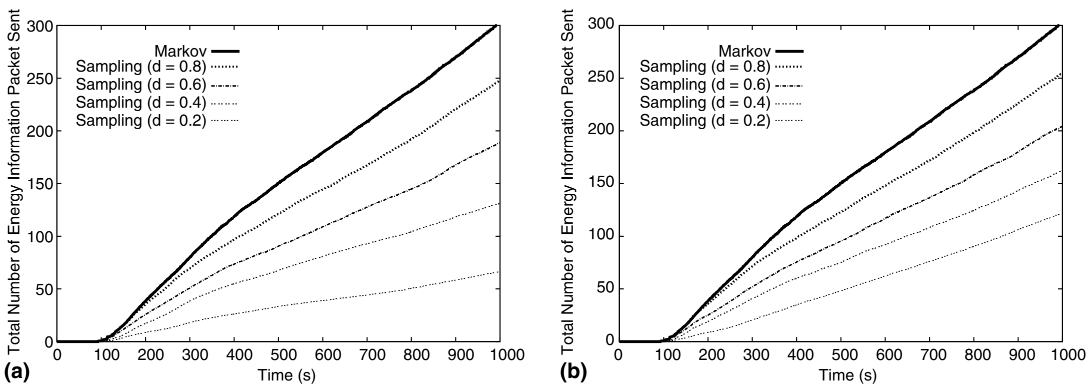


Fig. 13. Number of energy packets sent using the static and dynamic sampling approaches. (a) Static sampling, (b) dynamic sampling.

expected, in all simulations, the total number of energy information packets sent by sampling techniques was less than the amount sent by the Markov. We can observe that, the smaller the sampling degree is, the lower the total number of energy packets sent. In the static approach, this value is probabilistically equals to the sampling degree multiplied by the total number of energy packets sent by Markov. In the dynamic approach, the number of packets sent is greater than this multiplication, because the probability $p$ of sending a packet increases whenever the node reaches the value of *threshold*, and its energy information is not sent. The speed of this increase is determined by the value of $k$. In all simulations, we use $k = 1$. Table 3 shows the number of energy information packets sent at the end of simulation for both models.

Table 3
Total number of energy packets sent using the static and dynamic sampling approaches

| Model | Static approach | Dynamic approach |
|---|---|---|
| Sampling ($d = 0.2$) | 66 | 122 |
| Sampling ($d = 0.4$) | 132 | 162 |
| Sampling ($d = 0.6$) | 189 | 204 |
| Sampling ($d = 0.8$) | 248 | 255 |
| Markov | 303 | |

Fig. 14 shows the average error in percentage for a simulation of 1000 s. We verify that the Markov has the smallest error, followed by the dynamic and static approaches, respectively. This is due to the total number of energy information packets sent in each approach. An interesting point when comparing Figs. 13 and 14 is that the dynamic approach has a better performance than the static one. For instance, the dynamic model using $d = 0.2$ sends 122 packets, while the static

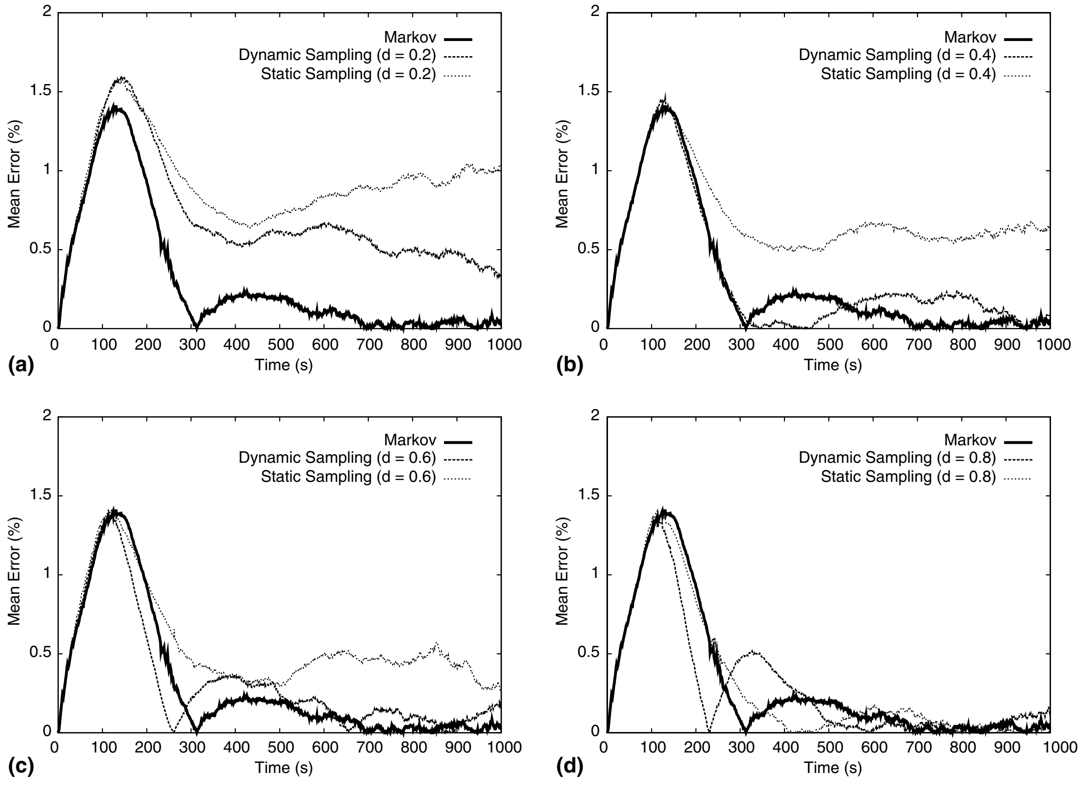


Fig. 14. Error using the static and dynamic sampling approaches. (a) $d = 0.2$, (b) $d = 0.4$, (c) $d = 0.6$, (d) $d = 0.8$.

using $d = 0.4$ sends 132. However, the errors of both approaches are very similar. When we compare the dynamic model using $d = 0.4$ and $d = 0.6$ with the static one using $d = 0.6$ and $d = 0.8$, respectively, we verify that the former sends less energy packets, and has smaller errors. The advantage of the dynamic model is due to the increase in the probability $p$ when a node do not send an energy packet.

Our next goal is to compare sampling techniques with and without the interpolation defined in Eq. (4). Fig. 15 shows that the greater the sampling degree, the smaller the advantage of using the interpolation phase. This is expected because when the sampling degree increases, the number of energy information packets received by the monitoring node also increases. As the interpolation does not have any influence in the sampling phase, the number of energy information packets is exactly the same in both curves of the same figure. It is important to point out that, in our simulation model, failures are not considered. Therefore, if failures are considered, the interpolation phase can improve the map quality because lost information can be estimated from information of neighboring nodes.

As observed in Section 5.3, one way to diminish the number of energy information packets needed to construct the map is to increase the value of the parameter *threshold*. Our next goal is to compare the Markov using a large value of *threshold* with sampling techniques. When we increase the value of the parameter *threshold* in the Markov model, all nodes send their energy information less frequently. In sampling models, few nodes send their energy information more frequently. Fig. 16 compares these two approaches. In Fig. 16a and b, we compare the Markov using threshold = 5% with

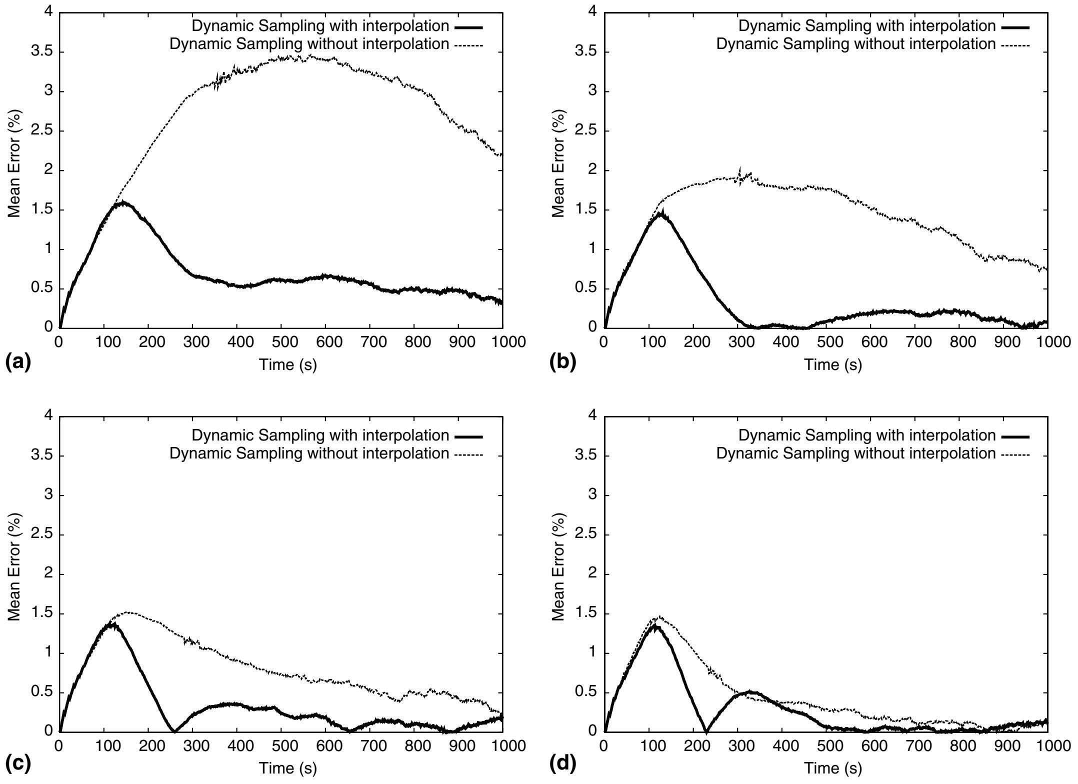


Fig. 15. The influence of the interpolation phase in sampling techniques. (a) $d = 0.2$, (b) $d = 0.4$, (c) $d = 0.6$, (d) $d = 0.8$.

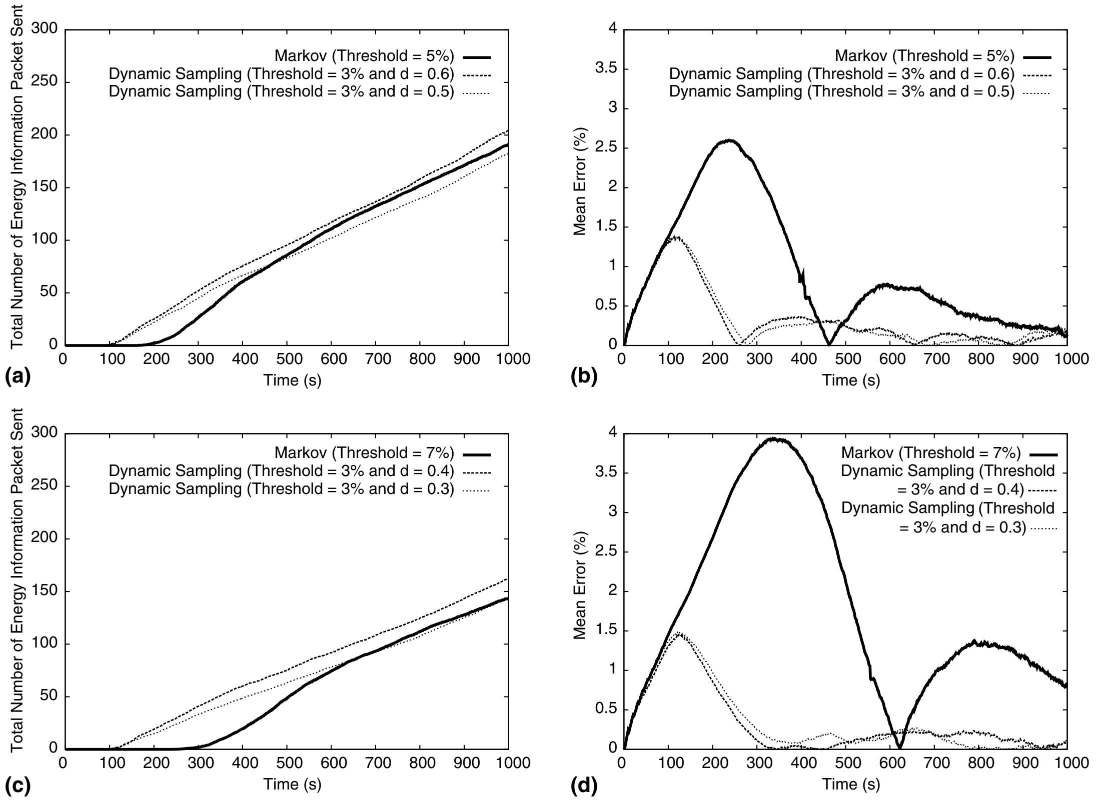


Fig. 16. Sampling models vs. original Markov using different values of *threshold*. (a) Energy packet number, (b) average percentage error, (c) energy packet number, (d) average percentage error.

the sampling model using threshold = 3% and $d = 0.5$ and 0.6. Fig. 16c and d shows the Markov with threshold = 7% and the sampling with threshold = 3% and $d = 0.3$ and 0.4. Fig. 16a and c shows the total number of energy packets sent during all simulation, and Fig. 16b and d shows the mean error in percentage. We can see that, for similar number of energy packets, the error of the sampling model is smaller than the one of the Markov. Fig. 16a and c shows that, at the beginning of simulation, the number of packets sent by Markov is smaller, and, thus, its error is bigger than the error of sampling models. This happens because the value of *threshold* delays the sending of the energy packets in the Markov model. Therefore, the sampling model produces more constant error curves than the original Markov. This is the greatest advantage of sampling models over the original Markov.

## 7. Conclusions and future directions

In this work, we have studied the problem of constructing the energy map of wireless sensor networks using prediction-based approach. In this model, 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. Simulations were conducted in order to compare the performance of a prediction-based approach with a naive one, in which only the available energy is sent to the monitoring node. Simulation results indicate that the prediction-based approach analyzed is more energy-efficient than the naive solution, and also that this approach is more scalable with respect to the number of sensing events. We also analyzed the use of a sampling technique to reduce the number of packets needed to construct the map. Results showed that its most important advantage is to produce more constant error curves.

The next step is to apply the energy map in problems such as the trajectory based forwarding protocol proposed in [5]. The information presented in the energy map could be used to plan the trajectory according to energy reserves, preserving or avoiding regions with small energy reserves. We also plan to study the construction of localized energy maps. In all energy map constructions presented in this work, the map of the entire network was constructed in the monitoring node. However, in some situations, it is enough to know the energy information of a neighboring region. An energy map that gives information about a region surrounding the node is named localized energy map. This localized energy information can be useful to improve the energy efficiency of other algorithms such as routing protocols. The construction of localized energy map is a promising extension of this work.

## Acknowledgement

This work has been partially supported by CNPq, Brazil, under process number 55.2111/2002-3, DARPA under contract number N-666001-00-1-8953, and a grant from CISCO systems. We would like to thank the comments and suggestions of Prof. Narayan Mandayam, Associate Professor of ECE Department at Rutgers University and Associate Director at WINLAB, the members of both the DATAMAN group of Rutgers University and SensorNet at Federal University of Minas Gerais, and also the reviewers who provided useful comments and helped us to improve the final version of the paper.

## References

[1] G. Asada et al., Wireless integrated network sensors: low power systems on a chip, in: European Solid State Circuits Conference, The Hague, 1998.

[2] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister, System architecture directions for networked sensors, in: Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems, November 2000.

[3] C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Proceedings of MOBICOM, Boston, 2000, pp. 56–67.

[4] J.M. Kahn, R.H. Katz, K.S.J. Pister, Next century challenges: mobile networking for smart dust, in: Proceedings of MOBICOM, Seattle, 1999, pp. 271–278.

[5] D. Niculescu, B. Nath, Trajectory-based forwarding and its applications, in: Proceedings of MOBICOM, San Diego, 2003.

[6] ns2, The network simulator, Available from <http://www.isi.edu/nsnam/ns/index.html>, 2002.

[7] S. Park, A. Savvides, M.B. Srivastava, SensorSim: a simulation framework for sensor networks, in: Proceedings of the 3rd ACM Intl Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, Boston, 2000, pp. 104–111.

[8] G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors, Communications of the ACM 43 (2000) 551–558.

[9] J.M. Rabaey, M.J. Ammer, J.L. da Silva Jr., D. Patel, S. Roundy, Picoradio supports ad hoc ultra-low power wireless networking, IEEE Computer 33 (7) (2000) 42–48.

[10] S. Ross, A First Course in Probability, fifth ed., Prentice-Hall, Englewood Cliffs, NJ, 1998.

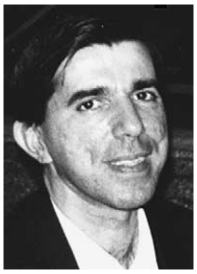
[11] K. Sohrabi, J. Gao, V. Ailawadhi, G.J. Pottie, Protocols for self-organization of a wireless sensor network, IEEE Personal Communications 7 (2000) 16–27.

[12] Alec Woo, David E. Culler, A transmission control scheme for media access in sensor networks, in: Proceedings of MOBICOM, Rome, July 2001, pp. 221–235.

[13] Y. Jerry Zhao, R. Govindan, D. Estrin, Residual energy scans for monitoring wireless sensor networks, in: Proceedings of WCNC, Orlando, 2002.

**Raquel A.F. Mini** holds a B.Sc., M.Sc. and Ph.D. in Computer Science from Federal University of Minas Gerais (UFMG), Brazil. Currently she is an Associate Professor of Computer Science at PUC Minas, Brazil. Her main research areas are sensor networks, distributed algorithms, and mobile computing.

**Max do Val Machado** received the B.S. degree in Computer Science From Pontifical Catholic University of Minas Gerais, Brazil in 2002. Currently, he is a Master's student in Computer Science at the Federal University of Minas Gerais, Brazil. His research interests are algorithms for wireless sensor networks and mobile ad hoc networks.



**Antonio A.F. Loureiro** holds a B.Sc. and a M.Sc. in Computer Science, both from the Federal University of Minas Gerais (UFMG), and a Ph.D. in Computer Science from the University of British Columbia, Canada. Currently he is an Associate Professor of Computer Science at UFMG. His main research areas are wireless sensor networks, mobile computing, distributed algorithms, and network management.

**Badri Nath** is a professor of computer science at Rutgers University. His research interests are in the area of mobile computing and sensor networks. As part of his research, he is developing protocols and services suited for large scale dense networks. His current focus is on developing protocols and services for an information architecture on top of sensor networks. In particular, he is investigating new paradigms for routing, localization, and data management in sensor networks. He is also the recipient of the 10 year best paper award at VLDB-2002. He has served on several program committees of conferences in the area of mobile computing, data management and sensor networks. Currently, he serves on the editorial board of ACM Transactions on Sensor Networks, WINET, Wireless Communications and Mobile Computing, and IEEE Pervasive Computing. He has a Ph.D. from the University of Massachusetts, Amherst.