The Best Energy Map of a Wireless Sensor Network

Raquel A.F. Mini[1 2], **Antonio A.F. Loureiro**[1], **Badri Nath**[3]

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

[2]Department of Computer Science – Pontifical Catholic University of Minas Gerais
Belo Horizonte, MG, 30535-610, Brazil

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

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

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. Due to the paramount importance of energy conservation, it is highly desirable to define the amount of energy each protocol can spend to perform its goal. Using this idea, we can associate a finite energy budget for each network activity, and ask this activity to achieve its best performance using only its budget. This is a new way of dealing with network related problems, and should be considered a new paradigm to design algorithms for networks that are battery powered, specially for wireless sensor networks. In this paper, we present this new paradigm, and show how it can be used to construct the energy map of a wireless sensor network. Our goal is to construct the best energy map using only a defined amount of energy. Simulation results show that we can approach these performance limits using the proposed finite energy budget model.*

1. Introduction

During the last few years there was a great development of small, low-power hardware platforms that integrate devices for sensing (and eventually actuating on the environment), processing data received from or to be sent to the environment, and communicating using a wireless medium. These computational devices are called sensor nodes and are grouped to form a wireless sensor network, producing measurable responses to changes in physical conditions. A wireless sensor network represents a new monitoring and control capability for applications such as environmental monitoring, infrastructure management, public safety, medical, home and office security, transportation, and military [Badrinath et al., 2000, Estrin et al., 2000, Lindsey et al., 2001, Meguerdichian et al., 2001]. Some of these applications foreseen to sensor networks will require a large number of devices in the order of tens of thousands nodes. Traditional methods of sensor networking represent an impractical demand on cable installation, thus, leading to wireless communication. Performing the processing at the source can drastically reduce the computational burden on application, network, and management.

The large use of wireless sensor networks depends on the design of a scalable and low-cost sensor network architecture. Furthermore, the design must consider the energy conservation a fundamental issue and devise mechanisms for extending the network lifetime. This scenario leads us naturally to the following problem: what is the best performance a protocol can achieve given that it can spend only a finite amount of energy? Using this idea, we can associate a finite energy budget with each network activity, and ask this activity to achieve its best performance using only its budget. This is a new way of dealing with network related problems, and should be considered a new paradigm to design algorithms for networks that are battery powered, specially for wireless sensor networks.

The finite energy budget paradigm is highly applicable in the construction of the energy map of a wireless sensor network. The energy map give us the information about the remaining available energy in each part of the network, and it can aid in prolonging the lifetime of the network. In [Mini et al., 2003, Mini et al., 2002], we present prediction-based approaches to construct the energy map for wireless sensor networks. In these approaches, each sensor 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. Simulation results indicate that prediction-based approaches are more energy-efficient than the naive solution in which each node sends only its available energy to the monitoring node.

The finite energy budget paradigm is exceedingly appropriate for the energy map construction because it is worthless if we construct the best energy map spending all available energy. It is highly desirable to define the amount of energy we can spend in the energy map construction, thus leaving the remaining energy to be used by the other network activities. This scenario leads us naturally to the following question: what is the best energy map we can construct, given a finite amount of energy for its construction? In this work, we extend the prediction-based approaches presented in [Mini et al., 2003, Mini et al., 2002] in order to define a way of constructing the energy map in situations where a finite energy budget is defined. Our goal is to achieve the performance limits in the construction of the energy map under the constrain that each node can spend only a certain amount of energy in this construction.

The remainder of this paper is organized in the following way. In Section 2, we briefly survey the related work. In Section 3, we show how the finite energy budget model is applied in the energy map construction. In Section 4, we present the simulation results for this model. In Section 5, we expand the basic finite budget model, presenting an adaptive process to build the energy map. Section 6 shows the simulation results for the adaptive energy map construction. Finally, in Section 7, we present the concluding remarks and future directions of this work.

2. Related Work

Sensor networks are a new kind of ad hoc network with some new characteristics and challenges. These networks perform distributed sensing, wireless communication and distributed processing using very limited resources. The design of wireless sensor networks is a very challenge task and some new issues have to be addressed. These issues potentially affect many aspects of the network design such as device-level communication primitives, routing, addressing mechanisms, application architecture, and security mechanisms [Estrin et al., 1999]. The design of wireless sensor networks is a very fertile area of research and more work needs to be done in order to set up this new kind of 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, 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]. 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. Routing protocols can also take advantage of the available energy information in each part of the network. 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. This protocol can also form a virtual backbone connecting high energy islands. Other possible applications 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. Therefore, the energy map is an important information for sensor networks.

The work proposed in [Zhao et al., 2002] tries to obtain the energy map of sensor networks by using an aggregation based approach. The technique described in that paper tries to obtain the energy map of sensor networks by using an aggregation based approach. 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 [Mini et al., 2003, Mini et al., 2002] is that in the former solution 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. 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. Non-Adaptive Energy Map Construction under a Finite Energy Budget

Due to the paramount importance of energy conservation in wireless sensor networks, it is highly desirable to define the amount of energy we can spend in the energy map construction, thus leaving the remaining energy to be used by the other network activities. In this section, we show how the finite energy budget paradigm can be applied in the construction of the energy map of a wireless sensor network.

In the energy map construction, the term finite energy budget means that each node can spend a certain amount of energy in the process of constructing the network energy map. This amount of energy can be represented by the number of bytes each node can send with energy information to the monitoring node. Knowing the size of this information, we can transform the number of bytes into the number of packets each node can send with energy information. In this work, the number of packets is used as the metric for energy budget, and we deal with it as the maximum number of times each node can send its energy information to the monitoring node. Our goal is to construct the best energy map under the constrain that each node can send no more than a certain number of packets with energy information.

Ideally, a solution that approaches the performance limits in the construction of the energy map should keep the error almost constant during all time, and the budget should finish exactly at the end of the network lifetime. To achieve this goal, we have to decide when is the best time for each node to send its energy information packet under a finite energy budget. In this section, we propose a way of making this decision.

The moment to send the energy information packet is decided locally by each node without exchanging information with its neighbors. Then, periodically, each node should decide if it will send another energy information packet or not. We propose that this decision should be taken according to a certain probability p. In that case, the value of p determines the frequency in which the nodes will send their energy information and, thus, the amount of energy spent in the process of constructing the energy map. The value of p depends on the following parameters: the number of packets a node still can spend, the error between the predicted and the correct energy value, and its estimated lifetime. This last supposition is reasonable because given the restriction of the available energy in the sensor network, we can expect to design a wireless sensor network to work for a period of time that can be estimated during its design phase.

In order to find the best way to determine the value of the probability p for each node, we start defining the sending of an energy information as a binomial distribution in which the event is the sending of an energy packet, and each time-step is an experiment. Thus, in each time-step, we have a probability p that the event occurs (the node sends the energy packet) and a probability $(1-p)$ that the event does not occur (the node does not send the energy packet). Then, we have to find the correct value of p that maximizes the probability that each node sends the number of packets determined by its budget. We call T_{total} the estimated lifetime, and T_{now} the current time. In addition, P_{total} is the number of energy information packets each node can send, and P_{used} the number of packets the node has already used. Thus, a node still can send $(P_{total} - P_{used})$ packets in the remaining $(T_{total} - T_{now})$ seconds. To find the probability of sending a packet in each time-step, we have to maximize the probability of happening $(P_{total} - P_{used})$ events in $(T_{total} - T_{now})$ experiments. If p is the probability of the occurrence of an event, we have that:

$$P(X = P_{total} - P_{used}) = \binom{T_{total} - T_{now}}{P_{total} - P_{used}} p^{(P_{total} - P_{used})} (1-p)^{(T_{total} - T_{now}) - (P_{total} - P_{used})} \quad (1)$$

To find the best value for p in a way that a node will send $(P_{total} - P_{used})$ packets in $(T_{total} - T_{now})$ seconds, we have to maximize the function of equation (1). The value of p that maximizes this function is:

$$p = \frac{P_{total} - P_{used}}{T_{total} - T_{now}} \quad (2)$$

Using equation (2), each node can determine the probability in which it will send an energy packet in each second of simulation. This equation defines the value of p only in terms of the energy budget and the estimated lifetime.

Nevertheless, each node also knows the error between the energy value predicted by the monitoring node and the correct one. It can locally determine this value by just keeping the parameters of the last prediction sent to the monitoring node. Thus, each node keeps track of the error of its energy value. We use the percentage error because the impact of the difference between the correct and the predicted value depends on the available energy at the node. For example, if a node has 100 J of energy and we predict that it has 99 J is better than when we make the same error of 1 J when the total energy available in a node is only 2 J. Because of that, we consider the percentage error as being more suitable than its absolute value. Therefore, in our approach, the error is always considered as a percentage value in relation to the correct available energy.

We intend to use the information of the error to change the value of the probability p in such a way that when the error is small we should decrease the value of p in order to postpone the sending of an energy packet. With this behavior we can save energy to be spent when the error is larger. On the other hand, when the error is large, we should increase the value of the probability in order to force the node to send a new energy information. This behavior can be obtained with the aid of the function: $f(x) = \left(1 - \frac{1}{c^x}\right)$.

In Figure 1, we plot this function. We can see that this function tends to 1 when the value of x increases. The speed of this trend depends on the value of constant c. This constant has to be greater than 1, and the larger its value, the faster $f(x)$ tends to 1. This function will be useful to make the value of the probability p to adapt according to the error in the energy information.

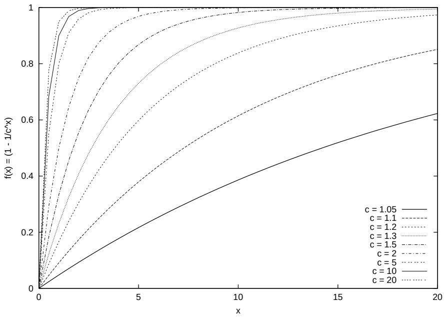


Figure 1: Function $f(x) = (1 - \frac{1}{c^x})$ for different values of c.

The desired adaptation is obtained by equation (3). In that equation, we redefine the value of p and call it $p\prime$. Then, in each time-step, each node will send another energy information packet with probability $p\prime$. In the first part of equation (3), the value of p is multiplied by the function $(1 - \frac{1}{c^{error}})$, and p is decreased when the error is small and it is almost unchanged when the error increases. However, when the error gets larger, the value of the probability of sending a packet should increase and becomes larger than p. This behavior is obtained using the second part of equation (3), in which the value of $(1 - p)$ is multiplied by the same function but with different parameters. The expression $max(0, error - k)$ is different from zero only when the error is larger than k. Using the value of c and k, we can control the shape of the curve that represents the probability of sending a packet.

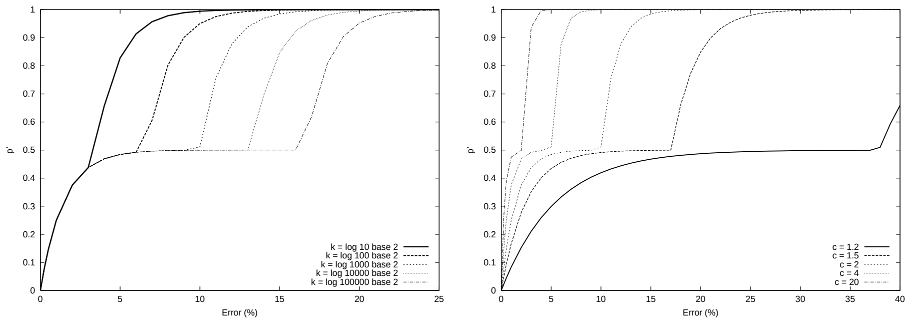
$$p\prime = p \times \left(1 - \frac{1}{c^{error}}\right) + (1 - p) \times \left(1 - \frac{1}{c^{max(0, error - k)}}\right) \tag{3}$$

Before using equation (3), we have to decide on the value of k. This value determines when the second curve will start. For example, if we want that the second curve starts when the curve $(1 - \frac{1}{c^{error}})$ is 0.99 (this means that the value of $p\prime$ is 99% of p), we make $(1 - \frac{1}{c^k}) = 0.99$ and thus we find $k = \log_c 100$. If we want that the second curve starts when the first is 0.999, we make $k = \log_c 1000$. In Figure 2-a, we plot the curve of $p\prime$ when $p = 0.5$ and $c = 2$ for different values of k. We can see that, when we increase the value of k, we are postponing the appearance of the second slope and delaying the increase of the value of $p\prime$.

In Figure 2-b, we plot the curve $p\prime$ when the value of p is 0.5 and $k = \log_c 1000$ for different values of c. We can see that the larger the value of c, the faster the value of $p\prime$ will be 1. Then, if a node has only a small number of packets to spend in the construction of the energy map, it should use a small value of c. On the other hand, if a node can spend a lot of packets with energy information, it should use a larger c.

4. Simulation Results for the Non-Adaptive Energy Map Construction

In order to analyze the performance of the finite energy budget scheme, we implemented the construction of the energy map in the ns-2 simulator. The kind of sensor network we will work is one in which the nodes are static and homogeneous, and also that there is only one static monitoring node with plenty of energy. We suppose that nodes are deployed randomly forming a high-density network in a flat topology. Also, the events are static and their duration and radius of influence are randomly chosen. In relation to the data delivery model, we simulate an event-driven network in such a way that 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



(a) Function $p\prime$ for $p = 0.5$, $c = 2$, and different values of k.

(b) Function $p\prime$ for $k = \log_c 1000$, and different values of c.

Figure 2: Function $p\prime$.

communication model used is a cooperative sensor model in which the communication between nodes is beyond the relay function needed for routing, and sensors communicate with each other to disseminate information related to the event.

In our simulations, we use the State-based Energy Dissipation Model (SEDM) to describe the behavior of sensor nodes and to simulate their energy dissipation. The event arrival is modeled by a Poisson process, and its behavior is described by a static event model. This model represents a basic event that is static and has a fixed size. 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. The duration of each event is uniformly chosen between the values *event-duration-min* and *event-duration-max* seconds.

The SEDM is based on a framework in which the nodes have various operation modes with different levels of activation and, consequently, 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 Figure 3. 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. As example, both states 2 and $2'$ represent the operation mode 2, the only difference is that when a node goes to state 2, it always starts a timer, while in state $2'$, it verifies if is there any event for it. Thus, in terms of energy consumption, state i is exactly the same as state i'. The only difference is that the behavior of a node that goes to state i is different from the one that goes to state i'.

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 in all simulations throughout the remainder of this paper. Moreover, the 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. Besides, the results showed in all simulations correspond to an average of these values for 30 different runs.

In the finite energy budget, presented in the last section, the choice of the best value for the constant c is of fundamental importance to determine the behavior of probability $p\prime$, and consequently, the way each node will spend its budget. In order to analyze the influence of this constant for different values of budget, we ran the prediction-based energy map for 100 nodes in the same scenario described above. Figure 4-a shows the average percentage error when each node

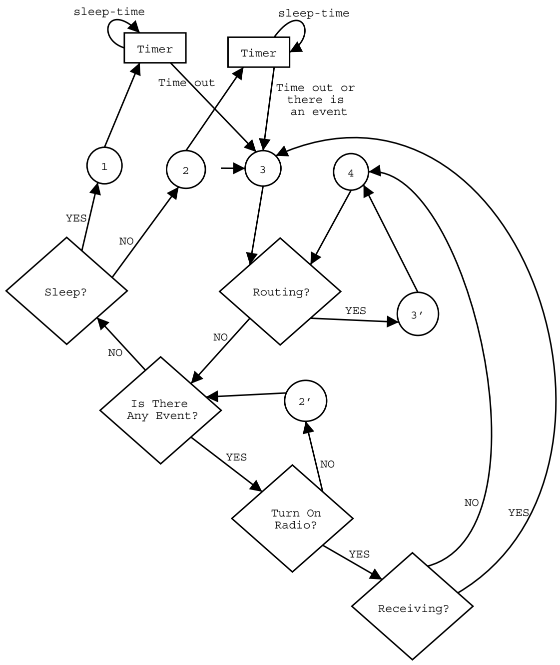


Figure 3: Diagram of the State-based Energy Dissipation Model.

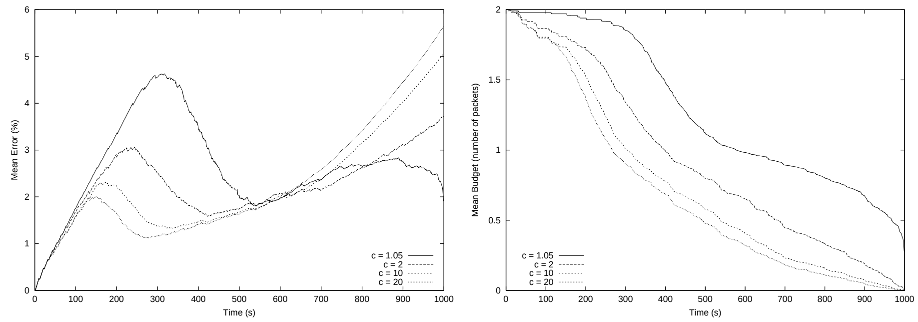
Parameter	**Value**	**Parameter**	**Value**
Number of Nodes	1000	Sensor Field Size	50×50 m^2
Initial Energy	100 J	λ	0.1
Communication Range	15 m	k	$\log_c 1000$
Power consumption of *mode 1*	25.50 μW	*event-duration-min*	5 s
Power consumption of *mode 2*	38.72 mW	*event-duration-max*	50 s
Power consumption of *mode 3*	52.20 mW	*event-radius-min*	5 m
Power consumption of *mode 4*	74.70 mW	*event-radius-max*	15 m

Table 1: Default values used in the simulations.

can send only 2 energy information packets to the monitoring node, and, in Figure 4-b, we have the mean budget for each value of c. We can see that the larger the value of c, the faster the nodes use their budget. For example, for $c = 20$, the budget is used fast and, at the end of simulation, almost all nodes have already spent their budget, increasing the mean error. On the other hand, for $c = 1.05$, all nodes spend their budget slowly.

Using the constant $c = 1.05$ and a budget size of 2 packets/node, the error starts larger and goes down after the start up period, keeping almost constant until the end of simulation. This behavior can be explained by equation (2). As the denominator represents the simulation time, and the numerator the budget size per node, the denominator range value is larger than the numerator one. Therefore, at the beginning of simulation, the value of p tends to be decreased due to the large value in the denominator. This collateral damage is good, because it is better to make an error at the beginning of simulation, when the nodes have more energy, than at the end, when the energy becomes even scarcer. Then, the shape of the error is good for sensor network applications.

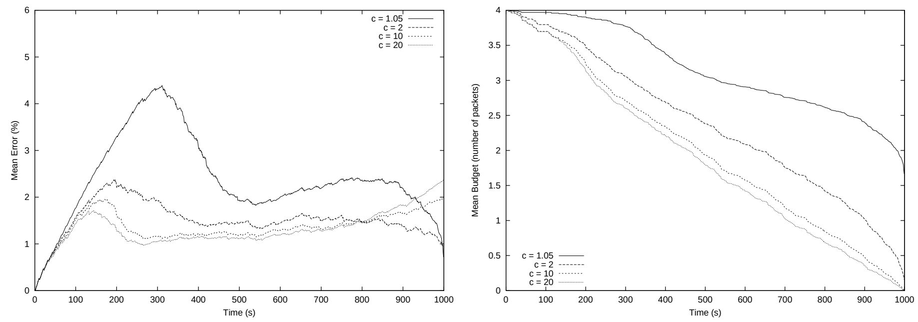
We can see in Figure 4-b that, for $c = 1.05$, the nodes do not spend all their budget. In a situation like this, we can increase the value of c in order to use all the available budget to obtain a smaller error.



(a) Mean Error (%). (b) Mean Budget (number of packets).

Figure 4: Changing the value of c when each node can spend 2 packets with energy information, $k = \log_c 1000$ and $\lambda = 0.1$.

Next, we repeated the simulation above with a budget size of 4 packets/node as presented in Figure 5. Using $c = 1.05$, the nodes do not spend all their budget and the error is large. Using $c = 20$, the nodes spend fast their budget and, at the end of simulation, the error gets larger because all budget is used. But, if we use $c = 2$, we achieve a good performance because the budget is over almost at the end of simulation, and the error keeps almost constant during all the time. This means that the value of $c = 2$ is a good choice when we have a budget size of 4 packets/node.

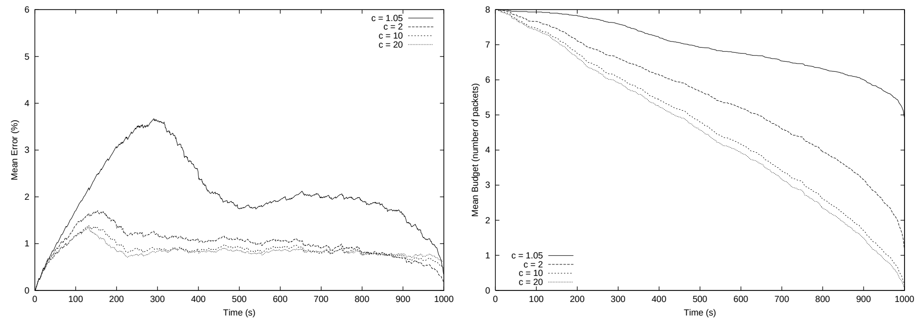


(a) Mean Error (%). (b) Mean Budget (number of packets).

Figure 5: Changing the value of c when each node can spend 4 packets with energy information, $k = \log_c 1000$ and $\lambda = 0.1$.

When the budget size is increased, we should use a larger constant. This can be seen in Figure 6, where a budget size of 8 packets/node is used. In this case, the best performance was achieved using $c = 10$ or 20. We can also observe that for $c = 1.05$, the budget is not appropriately used and, at the end of simulation, the nodes spend more of their budget and the error curve goes down. Even though, for this constant, only a small amount of the available budget is used.

Observing the slope of the budget curve, we can see that equation (3) provides a small adaptation during the simulation. For all values of c, when time goes by, the slope of the budget curve is changed in order to adjust to the remainder budget and simulation time. Nevertheless, this adaptation is not enough to achieve our goal in the energy map construction: use all the available budget keeping the percentage error almost constant. The performance of using equation (3) to decide when sending the energy packet is highly dependable on the right choice of the constant c.



(a) Mean Error (%).

(b) Mean Budget (number of packets).

Figure 6: Changing the value of c when each node can spend 8 packets with energy information, $k = \log_c 1000$ and $\lambda = 0.1$.

If a wrong value of c is chosen, the budget cannot be completely used or it can be used too fast increasing the percentage error. Therefore, in order to achieve the desired behavior, it is necessary to find a way of choosing the value of constant c automatically. This is discussed in the next section.

5. Adaptive Energy Map Construction under a Finite Energy Budget

In the previous section, we saw that the performance of the finite energy budget approach is highly dependable on the value of constant c. Now, we present a way of changing the value of this constant automatically in such a way that when we have a large budget, a big value of c is used and, when a small budget is available, its value should be small.

The key information that guide us to choose the right value of c is the budget curve. We start the adaptive process analyzing the budget curve periodically. As example, for each 3% of the simulation time, we apply a linear regression in the budget curve in order to predict when it would be the end of budget. Using this information, we can make the following observations:

1. If the predicted end of budget is beyond the end of simulation, the value of c must be increased, otherwise, it must be decreased;
2. In other to have a conservative behavior, at the beginning of simulation, we should increase less and decrease more the value of c, depending on the case. On the other hand, at the end of simulation, we should increase more and decrease less its value. This behavior is justified because it is more difficult to keep the percentage error constant at the end of simulation than at the beginning since, at the end, the available energy is lesser.
3. When the value of c is small, we should increase less its value. This is true because for small c, small changes in its value produce more distant curves.

The three remarks described above can be seen as the requirements that an adaptive process must follow. Next, we describe how we deal with each one of these requirements.

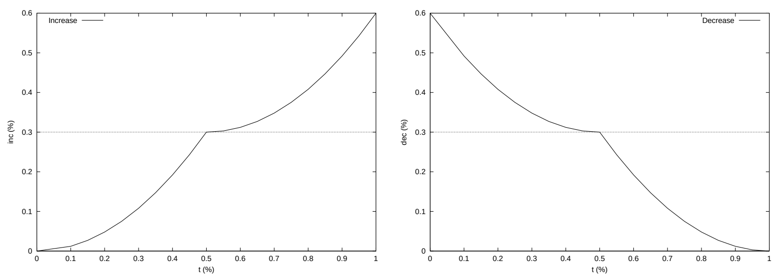
In order to achieve the first requirement, we calculate a value *dif* as being the amount in percentage that we should increase or decrease in the predicted end of budget in order to make this value equal to the end of simulation time. If *dif* > 0, we increase the value of c, otherwise we decrease the value of c. The amount of increasing and decreasing is defined taking into account the second requirement. Knowing that t is the percentage of simulation time already done. The percentage of increasing is given by the function:

$$inc = \begin{cases} \frac{dif}{0.25}\, t^2 & \text{if t} \leq 0.5, \\ dif + \frac{dif}{0.25}(t-0.5)^2 & \text{if t} > 0.5. \end{cases} \quad (4)$$

The amount of decreasing is set according to the following equation:

$$dec = \begin{cases} \frac{dif}{0.25}\, t^2 - \frac{dif}{0.25}\, t + 2\, dif & \text{if t} \leq 0.5, \\ \frac{dif}{0.25}\, t^2 - 8\, dif\, t + \frac{dif}{0.25} & \text{if t} > 0.5. \end{cases} \quad (5)$$

Figure 7 shows these functions when $|dif| = 0.3$. This means that the predicted end of budget should be increased or decreased in 30% in order to achieve the end of simulation. We can see that using this function, at the beginning of simulation, we increase less the value of c and decrease more. On the other hand, at the end of simulation, we increase more and decrease less its value. This approach provides a conservative behavior since it will try to save budget at the beginning of simulation to be used at the end. This is a good approach because, at the end of simulation, the available energy is smaller, and thus it is more difficult to keep the percentage error constant.



(a) Increase function when $dif = 0.3$.

(b) Decrease function when $dif = -0.3$.

Figure 7: Increase and decrease function.

In other to understand the third requirement, we have to notice that the smaller the value of c, more distant will be the curves when we make a change in its value. In Figure 8, we can see that the distance between the curves $c = 1.05$ and $c = 1.05 + 10\% = 1.155$ is bigger than the distance between the curves $c = 2$ and $c = 2 + 10\% = 2.2$. This means that when we are working with a small c, the error is bigger, and a small increase in its value will change $p\prime$ to a curve where the probability of sending a packet for the current error is almost 1. In situations like this, the new value of c will make all nodes send an energy information packet at the same time, ending the budget. In addition, small values of c happens in situations in which a small budget is assigned to the energy map construction, and, in these cases, an error in the decision of when to send an energy information packet can not be undo.

To illustrate the problem with small values of c, we run a 2000 second simulation with a budget size of 1 packet per node. Figure 9-a shows the value of c in each second of simulation and, in Figure 9-b, we plot the error in the energy map. The simulation starts with $c = 1.01$ and, at time 200 s, the adaptive process decides to increase its value to $c = 1.028358$. This small change makes all nodes send their energy packet at the same time, ending the budget. Besides, in this case, the error can not be undo because the budget associated with each node is only 1 packet. Because of that, no more energy information packet can be sent and the error curve goes up.

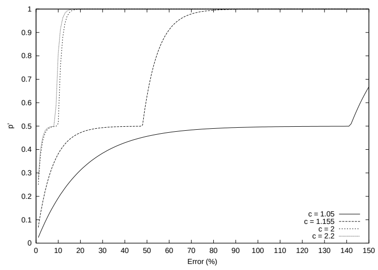
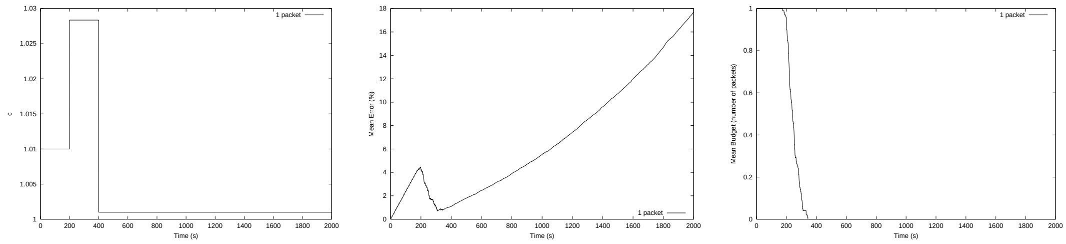


Figure 8: Increasing the value of c in 10%.



(a) Value of constant c.

(b) Mean Error (%).

(c) Mean Budget (number of packets).

Figure 9: Using a small budget: budget size= 1 packet/node, $k = \log_c 1000$ and $\lambda = 0.1$.

In order to solve the problem of increasing and decreasing the value c when it is small, we made another modification in the value of inc for small values of c. If the value of c is smaller than 1.001, we make $inc = \frac{inc}{100}$. If the value of c is between 1.001 and 1.01, we make $inc = \frac{inc}{10}$. The algorithm presented in Figure 10 summarizes all the adaptive process described above.

6. Simulation Results for the Adaptive Energy Map Construction

The goal of this section is to analyze the performance of the adaptive energy map construction described in the last section. To this end, we change the simulation time and the size of the budget, and plot the error and the available budget in each second of simulation.

Figure 11 shows the results for a simulation of 500 s using 1, 3, 5 and 7 packets. For all sizes, the error is almost constant during all the simulation time, and the budget finishes at the end of simulation, meaning that the adaptive process achieved a good performance in these scenarios.

If we take a look at Figure 11-b, it is possible to see that the budget curves are above the straight line defined by the function $y = -\frac{\mathit{budget}}{\mathit{simulationTime}}\, x + \mathit{budget}$. The fact that the budget curve is above this fuction means that the adaptive process is achieving a conservative behavior of saving budget at the beginning of simulation to be spent at the end.

Figure 12 shows the results for 1000 second simulation. In this case, the error curve is not so constant like in the previous case, but the budget curve is still well used during all simulation. Figure 13 illustrates the results for a simulation period of 1500 s. We can see that even when each node can use only 1 packet during all simulation, the use of the budget is distributed during all the simulation time. The same can be seen, in Figure 14, in which a simulation period of 2000 s is analyzed. In all cases, the budget is used uniformly to keep the error almost constant.

It is important to point out that, in this work, the budget is modeled as the number of packet transmissions. However, other budgets could also be considered. As an example, we can

ADAPTIVE ALGORITHM:

```
Input:
    simulationTime                          {total amount of simulation time}
    t                              {percentage of simulation time already gone}
    c                                                   {current value of c}
begin
    endOfBudget ← LinearRegression(budgetCurve);
    dif ← (endOfBudget − simulationTime) / endOfBudget;
    if (dif > 0) then begin
            if (t ≤ 0.5) then
                inc ← (dif × t²) / 0.25
            else inc ← dif + (dif × (t − 0.5)²) / 0.25;
            if (c ≤ 1.001) then
                inc ← inc / 100
            else if (c ≤ 1.01) then
                    inc ← inc / 10;
            c ← c + c × inc;
        end
    else begin
            if (t ≤ 0.5) then
                dec ← (dif × t²) / 0.25 − (dif × t) / 0.25 + 2 × dif
            else dec ← (dif × t²) / 0.25 − 8 × dif × t + dif / 0.25;
            c ← c + c × dec;
        end;
end {Adaptive Algorithm}.
```

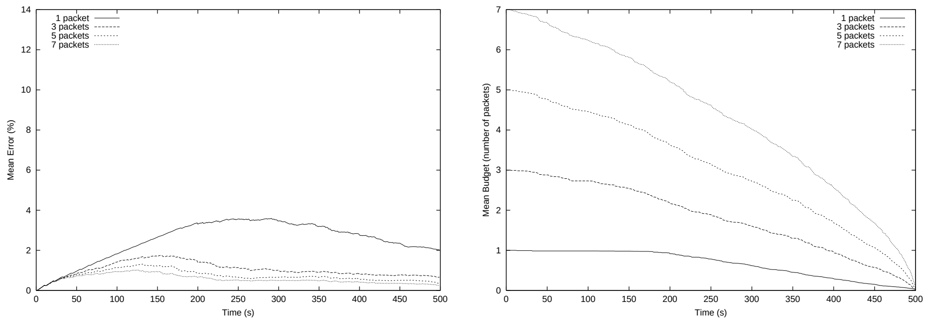
Figure 10: Adaptive Algorithm

define a budget associated with the computations in a way that we define the maximum number of times the linear regression can be executed. We can incorporate this computation budget in our model only changing the frequency in which the budget is verified.

The adaptive process proposed in this paper works fine for the sensor network analyzed in this work. Other kinds of sensor networks must be carefully studied. As example, the adaptive process is highly dependable on the amount of initial energy. The bigger this value, the more difficult to keep the percentage error constant. In other words, more budget will be necessary at the end of simulation to keep the percentage error constant, meaning that the budget curve should be more above the straight line $y = -\frac{budget}{simulationTime}\ x + budget$ than it was in the simulations presented in this work. However, we support the idea that the three requirements described in Section 5 must be achieved by the adaptive processes for all kinds of wireless sensor networks. Consequently, the adaptive process should be guided by the three requirements, but it should also take into account the characteristics of the the sensor networks.

7. Conclusions

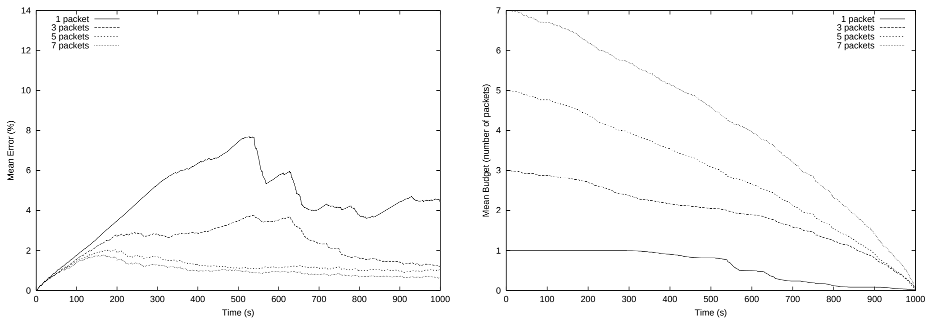
In this work, we presented a new model for constructing the energy map of wireless sensor networks under a finite energy budget. The energy budget was used in the context of defining the maximum number of packets each node can send with energy information to the monitoring node. We propose a model to represent the probability in which a node sends an energy information packet, and an approach to adjust this probability in order to construct the best energy map under a given energy constraint. Simulation results indicate that we can approach the performance



(a) Mean Error (%).

(b) Mean Budget (number of packets).

Figure 11: Changing the budget size in a 500 second simulation, $k = \log_c 1000$, $\lambda = 0.1$.



(a) Mean Error (%).

(b) Mean Budget (number of packets).

Figure 12: Changing the budget size in a 1000 second simulation, $k = \log_c 1000$, $\lambda = 0.1$.

limits using the proposed finite energy budget model. To the best of our knowledge, there is no other work related with the design of protocols for wireless sensor networks under a finite energy budget. This is a novel technique.

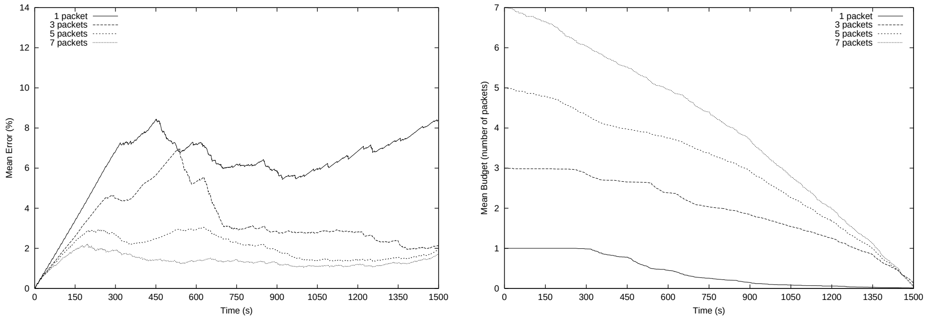
It is important to notice that when we design a wireless sensor network that is battery powered, it would be interesting to determine the energy budget associated with each network activity. This is a new paradigm in the design of wireless sensor networks. We plan to investigate the use of this new paradigm in other network activities. As example, we can design mechanisms that disseminate information to the maximum number of nodes under the constraint that they can use only a determined amount of energy. Another possibility is the design of network management functions to achieve their best performance using only a finite amount of energy.

Acknowledgment

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.

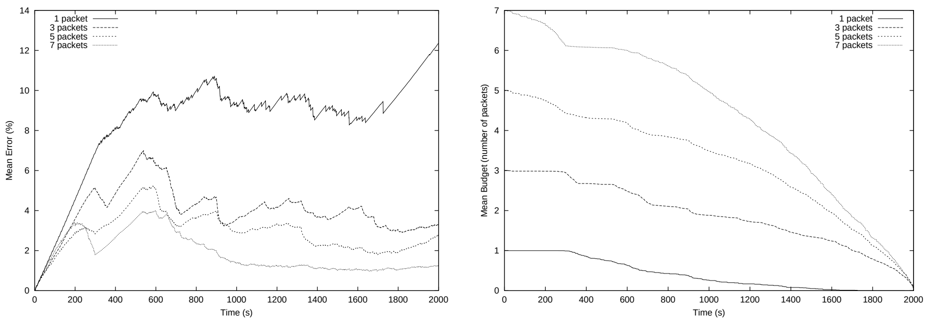
References

Badrinath, B. R., Srivastava, M., Mills, K., Scholtz, J., and Sollins, K. (2000). Special issue on smart spaces and environments. *IEEE Personal Communications*.



(a) Mean Error (%). (b) Mean Budget (number of packets).

Figure 13: Changing the budget size in a 1500 second simulation, $k = \log_c 1000$, $\lambda = 0.1$.



(a) Mean Error (%). (b) Mean Budget (number of packets).

Figure 14: Changing the budget size in a 2000 second simulation, $k = \log_c 1000$, $\lambda = 0.1$.

Estrin, D., Govindan, R., and Heidemann, J. (2000). Embedding the Internet. *Communications of the ACM*, 43(5):39–41. (Special issue guest editors).

Estrin, D., Govindan, R., Heidemann, J., and Kumar, S. (1999). Next century challenges: scalable coordination in sensor networks. In *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, Seattle, WA USA.

Lindsey, S., Raghavendra, C., and Sivalingam, K. (2001). Data gathering in sensor networks using the energy delay metric. In *International Workshop on Parallel and Distributed Computing: Issues in Wireless Networks and Mobile Computing*, San Francisco, USA.

Meguerdichian, S., Koushanfar, F., Potkonjak, M., and Srivastava, M. (2001). Coverage problems in wireless ad-hoc sensor networks. In *INFOCOM*, pages 1380–1387.

Mini, R. A. F., Loureiro, A. A. F., and Nath, B. (2002). A probabilist approach to predict the energy consumption in wireless sensor networks. In *IV Workshop de Comunicação sem Fio e Computação Móvel*, São Paulo, Brazil.

Mini, R. A. F., Loureiro, A. A. F., and Nath, B. (2003). Prediction-based energy map for wireless sensor networks. In *Personal Wireless Communications - PWC 2003*, Venice - Italy.

Zhao, Y., 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.