SDF - Solar-Aware Distributed Flow in Wireless Sensor Networks

Immanuel Schweizer, Nils Fleischhacker, Dirk Bradler, Max Mühlhäuser and Thorsten Strufe
Technische Universität Darmstadt
Hochschulstrasse 10
64289 Darmstadt

***Abstract*—Energy is the most limiting factor in wireless sensor networks. Harvesting solar energy is a feasible solution to overcome the energy-constraint in some applications. It enables a theoretically infinite network lifetime, sustaining a mode of operation termed *energy neutral consumption rate* The challenge arises, how can the harvested energy be utilized to maximize the performance of the sensor network.**

Considering a field monitoring application the performance is measured as the sustained *sampling rate* of the sensors. Maximizing the sampling rate needs to take the spatio-temporal distribution of load and energy into account, to prevent the overloading of nodes. In [1] they introduced a optimal, theoretical solution based on perfect global knowledge.

In this paper we propose the *solar-aware distributed flow* (SDF) approach. SDF enables each node to predict the harvested energy, calculate a sustainable flow and control its local neighborhood. To the best of our knowledge it is the first practical solution.

Extensive simulations confirmed that SDF achieves over 80% of the theoretical optimum, while introducing negligible overhead.

I. Introduction

Harvesting environmental energy has become a valid approach to overcome the hard limit imposed by battery capacity, either by supplementing or replacing the battery [2], [3], [4], [5], [6]. Harvested energy is potentially infinite. This reflects a paradigm change to the field, since it enables an entirely new mode of operation. The lifetime of sensor nodes, under absence of hardware failures, suddenly can be considered to be infinite. The nodes in this mode are termed to operate at an *energy neutral consumption rate*.

The main challenge for a traditional sensor network is to maximize the lifetime of the nodes, while guaranteeing a certain network performance, e.g., delay, sampling rate, or tracking accuracy. Considering the possibility to harvest energy, the challenges of ultra-low power operations remain similar, yet, two new challenges emerge: Can the energy neutral operation be guaranteed for each node in the network sustaining the aforementioned network performance. And can the network performance be maximized while still guaranteeing energy neutrality. Since the performance of the network and the energy neutral consumption rate depend on the spatio-temporal distribution of both energy and load throughout the network, these are non-trivial challenges.

Considering a field monitor application, increasing the *sampling rate* of the sensor nodes represents the inertial requirement. In field monitoring sensors observe a phenomena sending periodic data messages to a basestation. By increasing the sampling rate the granularity of the observation is increased. This comes at the cost of additional load to the network and increased energy consumption at each node.

In this paper we tackle the problem of how to maximizing the sampling rate in a solar-powered field monitoring application.

This problem was first introduced by Kansal et al. [1]. They established a theoretical solution for the homogeneous case. In the homogeneous case all nodes are assumed to sample at the same rate. We will assume heterogeneous sampling rates are possible. Their solution is to solve linear equations based on global knowledge. Obtaining this global knowledge about energy production, energy levels and the overall topology is not feasible. But their solution gives a theoretical optimum.

The problem of maximizing the sampling rates consists of two subproblems. Most importantly the sampling rates have to be maximized. Here the energy harvested must be taken into account. But also the possible communication throughput of any node in the network, as this might impose a limit on the data that can be transmitted energy neutral. The second problem is more general. The flow has to be routed from the nodes to the basestation. Throughout this paper we will assume a given routing algorithm.

In this paper we will introduce the solar-aware distributed flow (SDF) approach. To the best of our knowledge it gives the first practical solution to the problem of maximizing sampling rates for field monitoring.

SDF acts in the three steps of (a) establishing an estimation of the long term energy production, (b) calculating the maximum communication flow the node will be able to handle, and (c) distributing the possible flow to the direct neighbors.

Evaluating solar-aware algorithms poses the additional challenge of modeling environmental condition, i.e., solar radiation levels. Clouds or other objects might obstruct the sunlight. This can be mitigated by using real world solar radiation data that must be obtained before [1]. Accurate, detailed cloud models, as found in meteorology [7], [8], are highly complex. Yet, to cover worst case scenarios for a solar-aware flow control, we introduce a simpler model. Covering moving shades, overlapping and fluctuating cloud density, are sufficient.

Our main contribution is SDF, the first practical approach to sampling rate maximization in field monitoring. Extensive simulations of SDF show that it achieves a minimum sampling rate of 80% of the theoretical optimum.

The paper is organized as follows. Section II outlines the related work. Section III will introduce the basic models and assumptions. SDF will be introduced in Section IV. In Section V some basic properties of SDF are discussed. The results of the evaluation will be presented in Section VI. Finally Section VII will conclude the paper and give a short outlook.

II. Related Work

The feasibility of energy harvesting wireless sensor networks has been studied extensively in the recent past. Roundy et al. [9] review a great variety of potential power sources for wireless sensor networks. While they review the classical power storage in batteries and even micro fuel cells or micro heat engines they also research the potential of different energy harvesting techniques. These techniques include photovoltaics (i.e. solar panels), temperature gradients, human power, wind/air flow and vibrations. They showed that solar panels offer the highest power density. If the sensor network is deployed such that solar panels can be used this makes them the best choice for reliably powering sensor nodes.

Figure 1. A solar harvesting sensor node (Heliomote [2])

Another advantage of solar energy is that it is uncontrolled but predictable [1]. Predicting the availability of harvested energy is the basis of any harvesting sensor network. The prediction is used to adapt the duty cycle during times with no or insufficient energy [10] or to assign task to nodes with more energy [11].

Several working prototypes of wireless sensor nodes using solar power have been developed, showing that solar powered wireless sensor networks are indeed feasible. Raghunathan et al. [2] present the *Heliomote* (cmp. Fig. 1) – a Mica2 [12] mote enhanced with a circuit board equipped with a solar panel and NiMH batteries for solar energy harvesting. They show that in principle their device is capable of near perpetual operation.

Jiang et al. [3] design and implement *Prometheus* – a wireless sensor node based on the Telos-mote [13] with a solar panel, super-capacitors and Li-ion batteries. They present and discuss results from a ten day period.

Minmi et al. [4] present *Solar Biscuit*, a wireless sensor node without a battery, solely relying on a solar panel and a super-capacitor, for environmental monitoring.

Sikka et al. [5] present *Fleck1* a solar powered wireless sensor node using a solar panel and NiMH batteries. Their main contribution is the incorporation of a DC-DC converter enabling deeper battery discharge cycles between periods where solar power is available. In [14] they present results of a network operating 24x7 for over two years.

Alippi et al. [6] develop and present a wireless sensor network framework based on solar powered sensor nodes. The wireless sensor network is deployed in Moreton Bay, Brisbane, Australia to deliver temperature and luminosity data of the marine ecosystem. They present and discuss results from a four day period.

To make the operation of environmentally powered wireless sensor networks more efficient, several solar-aware protocols have been developed. Some focus on solar powered wireless sensor networks while others use a more general approach.

Lin et al. [15] developed *E-WME* (Energy-opportunistic Weighted Minimum Energy) an energy-aware routing algorithm for wireless sensor networks. It works with any environmental energy sources and they show that their routing scheme is asymptotically optimal. Other solar-aware routing approaches based on directed diffusion [16] can be found in [17], [18], [19].

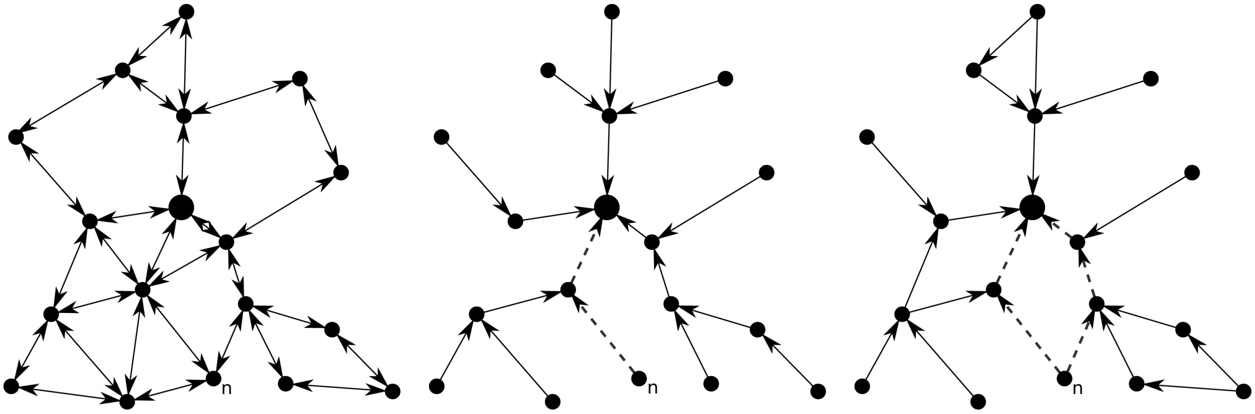
Voigt et al. [20] present a solar-aware clustering protocols that is based on a centralized and distributed variant of LEACH [21]. Their simulation results suggest a significant increases in lifetime from the solar-aware protocols compared with their not solar-aware counterparts. All of the approaches only try to handle a given load better. None is really looking at how the network load can be increased by utilizing the harvested power.

Some related work exists on the topic of assigning sampling rates in wireless sensor networks. Shu et al. [22] try to optimize the network sampling rates in terms of a scheduling problem. They try to maximize network throughput without taking energy consumption into consideration.

Bandyopadhyay et al. [23] analyze what tradeoffs in sensor density, energy usage, throughput and delay have to be made to achieve certain temporal and spatial sampling rates. But those sampling rates are always fixed and predefined.

The problem of maximizing sampling rates in solar harvesting sensor networks is first discussed by Kansal et al. in [1]. They describe a field monitoring application with the goal to maximize homogeneous sampling rates while remaining energy neutral. There approach is to model the flow in the network by using linear equations. Solving the linear equations yields one sampling rate and a corresponding flow that can be sustained by all nodes using no more than the energy neutral consumption rate. To calculate the solution global knowledge is needed over all nodes, flows and energy in the network. This is not feasible for a real-world sensor network.

What is missing is a distributed approach to maximizing sampling rates. If we take harvested energy into account we are no longer dealing with predefined sampling rates. It no longer suffices to optimize the nodes to sample at a given rate and deliver a given flow. We need to predict the available energy to utilize it as much as possible. And while they state the problem in [1] they fail to give a real-world solution. To the best of our knowledge there is no practical solution to the



(a) The network graph G

(b) Valid routing graph for b

(c) Invalid routing graph since n has two active routes to b

Figure 2. Examples of valid and invalid routing graphs derived from the same network graph

problem yet.

III. Preliminaries

The following section defines preliminaries and the models used for SDF and throughout this paper.

SDF will abstract from any given routing algorithm. The corresponding routing graphs needs to adhere to some basic properties we are going to outline here.

We introduce an energy model for nodes supplemented with a solar panel. Battery, solar panel and energy drains are defined here. A model for the solar radiation depending on time of year, time of day and geographical latitude is established. It gives a basic solar radiation level. To account for variations in the spatio-temporal distribution of the solar radiation we define a cloud obstruction model.

A. Routing Graph

SDF is targeted at field monitoring. We will assume a *single base station* b to serve as data sink. The network can be abstractly modeled using a directed graph $G = (N, E)$ with N being a set of vertices representing the network nodes and E being a set of edges with $(v_1, v_2) \in E$ if and only if v_1 can send directly to v_2.

We designate as *routing graph of node x* or $RG(x) = (N, R)$ the subgraph of the network graph $G = (N, E)$ for which

$$R = \{r | r \in E \wedge \exists v \left(v \in N \wedge r \in path\left(v, x\right)\right)\}.$$

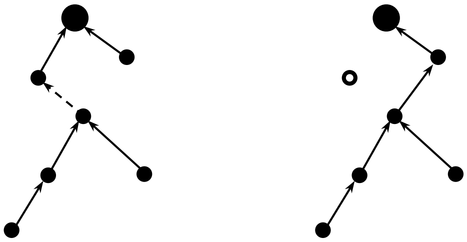
Thus $RG(b)$ is the graph consisting of all paths that are used to route a packet from any node to the base station. The layout and the properties of $RG(b)$ depend heavily on the used routing algorithm. Therefore we need to discuss the following two assumptions made about $RG(s)$.

Simplicity and Connectivity: It is assumed that from any node n in $RG(b)$, there exists exactly one active path to b. The resulting $RG(b)$ is a directed tree. For any node n we can define s as the direct successor and P as the set of predecessors in the routing graph.

Figure 2(a) shows an example network graph G. While 2(b) shows a routing graph that is valid, 2(c) shows a routing graph that would be invalid under this assumption. For node n in figure 2(b) there is only one active path from n to b, while in figure 2(c) there are two. Different routing algorithms have been proposed that fulfill this requirement and we refer to the respective publications [24].

Stability:

The nodes are assumed to be stationary. The topology might still change due to weak mobility, e.g., failure of nodes or changes of connectivity. SDF is able to compensate this by periodic update propagation.



(a) Due to link failure (dashed line) a internal node is now a leaf node

(b) Due to the left node failing a leaf node is now a internal node

Figure 3. Worst cases due to weak mobility

In worst case a internal node becomes a leaf node or vice versa (cmp. Fig. 3). We will see how SDF deals with those cases during the discussion later.

B. Energy Model

The energy model reflects three basic parts. A battery for energy storage, a solar panel to harvest energy and recharge the battery and a set of energy drains, modeling the energy consumption of the node's hardware such as CPU, radio and sensors.

A new energy level for time t_{i+1} can be calculated as

$$E\left(t_{i+1}\right) = \min(C_{max}, E\left(t_i\right) + H\left(t_i, t_{i+1}\right) - D\left(t_i, t_{i+1}\right))$$

where $E(t)$ is the Energy stored at time t, $H(t_i, t_{i+1})$ is the energy harvested by the solar panel, $D(t_i, t_{i+1})$ is the amount of energy consumed by the energy drain D and C_{max} is the maximum battery capacity. The *energy neutral consumption rate* p_n is reached if $H\left(t_i, t_{i+1}\right) = D\left(t_i, t_{i+1}\right))$.

Battery: The battery is assumed to be an ideal storage for electric energy. The battery has a fixed capacity C_{max} up to which it can be charged. For the sake of simplicity we assume that it does not have internal leakage currents and does not age or develop a memory effect due to recharging. We assume it to be completely insensitive to all environmental conditions. C_{max} has to be high enough to sustain the node for at least one day on full load. With sensor platforms allowing nodes a lifetime of several months on a single battery this assumption is easily valid.

Solar panel: The solar panel offers two basic functionalities. It is possible to determine the current power output and the energy accumulated over a specified time interval. The energy harvested between t_i and t_{i+1} can be calculated as $H\left(t_i, t_{i+1}\right) = r\left(t_i, t_{i+1}\right) \cdot A \cdot e$ where A is the panel area, e is a fixed efficiency constant and r is the solar radiation inbound to the node's location. Again for the sake of simplicity we assume a fixed efficiency. While it might change over time due to, e.g., the panel's orientation towards the sun, this does not effect the prediction of the inbound solar energy. The model for solar radiation will be introduced later.

Energy Drains: Energy drains model how much energy different hardware parts have consumed between t_i and t_{i+1}. We use a discrete event model for the different energy drains. Each event, e.g, the change of the CPU's duty cycle, triggers a recalculation of the remaining energy.

The CPU's energy consumption is modeled by active drain a, idle drain i and a duty cycle rate r. It is calculated as

$$D_c(t_i, t_{i+1}) = (t_{i+1} - t_i) \cdot (r \cdot a + (1 - r) \cdot i)\ .$$

The radio's energy consumption is modeled by the radio's state s and fixed drain $d(s)$ for each state. The states are *receive*, *transmit*, *on* and *sleep*. If s_c is the radio's current state, the drain is calculated as

$$D_r(t_i, t_{i+1}) = (t_{i+1} - t_i) \cdot d(s_c).$$

The sensor's energy consumption is modeled similarly. A sensor has an active state a and an idle state i. If s_c is the current state of the sensor, the drain is calculated as

$$D_s(t_i, t_{i+1}) = (t_{i+1} - t_i) \cdot d(s_c).$$

The total energy drain is

$$D(t_i, t_{i+1}) = D_c(t_i, t_{i+1}) + D_r(t_i, t_{i+1}) + D_s(t_i, t_{i+1}).$$

The next step is to model the environmental conditions that influence the solar radiation and therefore the harvested power at each node.

C. Solar Radiation Model

The solar radiation model is based on Brock [25], altered to yield a higher granularity.

The solar radiation R for a given point P and time t is calculated as

$$R(P,t) = max\left(0, \frac{s}{v(t)^2} \cdot cos\left(\alpha(P,t)\right)\right)$$

Here s is the *solar constant* ($s = 1353 Wm^{-2}$ [25]), v is the *radius vector* and α is the *zenith angle*.

The radius vector is used to correct the ellipticity of the earth's orbit. The zenith angle is the angle between the zenith and the position of the sun. It is calculated using the *declination* $d(t)$ and *minute angle* $\beta(t)$. The declination is the angular difference between the equator and the position of the sun at solar noon. The declination can be calculated approximately using the obliquity of the ecliptic as specified by the *international earth rotation & reference systems service* [26].

The minute angle is the angle between the sun and the south point for a certain minute of the day. As the earth moves $\frac{\pi}{720}$ per minute, the minute angle can be calculated as $\beta = (m - 720) \cdot \frac{\pi}{720}$, where m is the minute of the day i.e. the number of minutes that have passed since midnight.

D. Cloud Obstruction Model

Solar-aware algorithms have to adapt to variations in the spatio-temporal distribution of solar energy in outdoor environments. This is mainly due to static objects, i.e., buildings or trees, or moving objects, i.e. clouds, covering sensor nodes.

The prediction of solar energy should be able to adapt to changing energy levels due to obstruction. This is easy for static objects since they obstruct roughly the same amount of solar energy every day. Moving objects like clouds are much harder to predict. In this respect clouds are a worst case for solar energy prediction. They are moving, overlapping, and the cloud density changes over time.

In [1] they model the spatio-temporal distribution of solar radiation by using real world data that is captured a priori. This introduces the overhead of capturing such images and might only cover the solar profile of a specific region. Detailed cloud models as used in meteorology [7], [8] are far too detailed and complex.

We will introduce a simple cloud model to cover the worst cases for SDF. We will assume a given mean cloud density. A variance is introduced that accounts for possible changes in cloudiness. The mean and variance are input parameters of the model. The probability of a given cloudiness decreases when moving away from the mean. This is intuitively covered by assuming a normal distribution for the cloud density with a fixed mean μ and variance σ, where $\mu \geq 0$. Thus the probability density is given by:

$$\phi\left(x\right) = \frac{1}{\sqrt{2 \cdot \pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The model works in discrete time steps and a new density is calculated dependent on the normal distribution. Clouds are either removed or added until the new density is reached. The clouds themselves are modeled as randomly placed ellipses with the properties of transparency, speed, and direction. Since some clouds obstruct more radiation and clouds might overlap we need the notion of transparency t. The minimal and maximal transparency $t_{min}, t_{max} \in [0,1]$ can be set as input parameter.

The total solar radiation after obstruction S at point P and time t can thus be calculated as

$$S(P,t) = R(P,t) \cdot \prod_{\substack{C \in clouds \\ P \in C}} t(C).$$

This model offers a simple way to evaluate a solar-aware algorithm under worst case conditions.

IV. Solar-Aware Distributed Flow

Solar-aware distributed flow (SDF) wants to maximize the sampling rates achieved by each node while allowing energy neutral operation. The basic idea is that energy is mainly consumed due to either generating flow or relaying flow. Since each node has to forward the messages of all predecessors P in the routing graph it will issue control messages to the direct predecessors granting them *allowed flow*. So from a node's perspective SDF consist of three main steps that are repeated.

- Predict the *consumption rate* c
- Calculate the own *sampling rate* and the *allowed flow* of the predecessors P
- Send control messages to all direct predecessors in P and set the own sampling rate

Algorithm 1 Solar-aware distributed flow

```
Input: Allowed flow $f_a$ from successor
Output: Sampling rate $x$ and allowed flow $f_a$ to direct
    predecessors
    $c \leftarrow$ Consumable-Power
    $x \leftarrow$ Calculate-Flow
    $D \leftarrow$ Get-Direct-Predecessor-Set($P$)
    for all $p \in D$ do
        $P_p \leftarrow$ Get-Forwarded($P, p$)
        Send-Flow-Update-Message($pred, x \cdot$Size($P_p$))
    end for
    Set-Sampling-Rate($max(x_{min}, x \cdot s)$)
```

Algorithm 1 gives a sketch of one round of SDF. This will be repeated periodically on each node to yield a maximized sampling rate utilizing as much harvested energy as possible.

A. Consumable power

The first step is to predict the harvested energy. Since solar energy is classified as uncontrolled but predictable [1] this is indeed possible and different methods have already been proposed [1], [11], [10]. They are usually implemented to duty cycle nodes to use less or no energy if less or no energy is harvested. We want to minimize the fluctuation of the sampling rate, e.g., over night. Therefore we need a simple conservative method that approximates the *energy neutral consumption rate* over a longer time span. Generally speaking the energy neutral consumption rate p_n between t_i and t_{i+1} is defined such that the battery level $E(t_i) = E(t_{i+1})$. If the function $H(t)$ models the harvested power over time we can also state that

$$p_n = \frac{\displaystyle\int_{t_i}^{t_{i+1}} H(t)\ dt}{t_{i+1} - t_i}\ .$$

We will define the predicted consumption rate c and it should be bound by $c \leq p_n$.

We do not know $H(t)$ for the future. But we can approximately sample the energy neutral consumption rate for the past. Recall that in our model the solar panel can return the energy accumulated over a specified time interval. This might not be feasible with all implementations. In this case one can reside to measuring battery levels. This is less accurate but might be suffice if the difference between t_i and t_{i+1} is large enough.

We take several such samples at fixed time intervals. We then use their distribution to calculate c. We assume the samples S follow a normal distribution $\mathcal{N}\left(\mu, \sigma^2\right)$, where

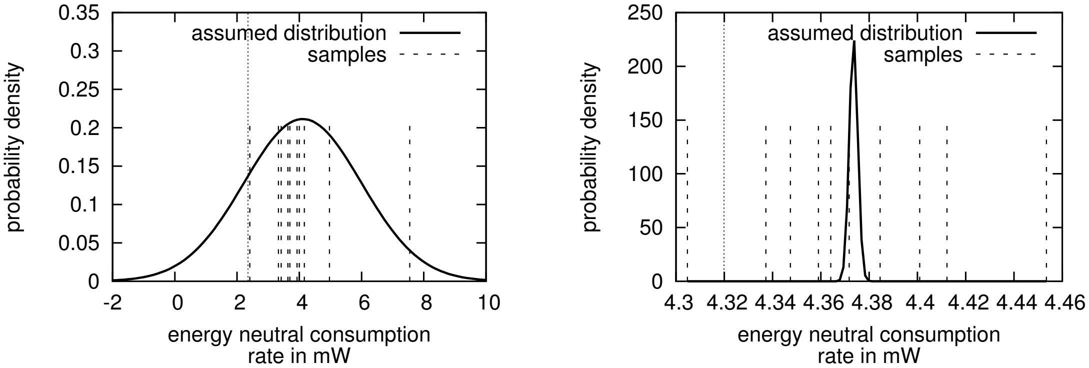
$$\mu = \frac{\sum\limits_{s \in\ S} s}{|S|} \quad and \quad \sigma^2 = \frac{\sum\limits_{s \in\ S} (s-\mu)^2}{|S|}\ .$$

We will then take a quantile of this distribution. The $x\%$ quantile is defined such that at most $x\%$ of the assumed distribution may lie below that value. For our approach we will take the 10% quantile as a very conservative approximation of c. If we assume a fixed energy neutral consumption rate p_n the mean of the distribution will converge against p_n. By using the 10% quantile we introduce a buffer. This will allow the model to handle inaccurate measurements and gives time to adapt to changes in the incoming solar energy.

Figure 4 gives an example as how this is applied to real samples. If the variance is reduced the quantile yields higher values.

At the beginning samples might be highly scattered. Assume that the network starts operation at 9am and the first sample is taken at 4pm. This will yield a high energy neutral consumption rate as the node was able to harvest energy all the time. Similarly if we start at night the first sample might yield 0. Since the approximation is getting better over time we add an initialization period of 24 hours where no samples are taken. This will prevent extreme outliers at the beginning to distort the calculations. After that samples will be taken at an interval of 7 hours.

To calculate a meaningful distribution a sufficient number of samples have to be retained. We will evaluate how much samples are sufficient later. To account for this, we will scale down c even further, until we reach a sufficient number of samples. We introduce an additional factor called *sample confidence* sc. sc is 0 if the number of samples is below two,



(a) Samples with higher variance (b) Samples with lower variance

Figure 4. The samples of the energy neutral consumption rate and the distributions assumed from them. The dotted line shows the distributions' 10% quantile.

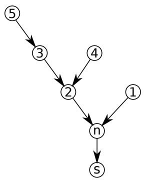
1 if the number of samples is sufficient and scales linear in between.

We return $c = sc * c$ as the predicted consumption rate. It is simple to calculate, while giving a good approximation and leaving buffer for sudden changes in solar radiation.

B. Calculate flow

If c is known, the node can calculate the sampling rate for itself, and the allowed flow for the predecessors. The first step is to determine the routing graph predecessors.

If a message is sent from node n to the basestation b it contains both n and b. Let x be a node on the path from n to b and p be the direct predecessor of x on that path. x can now store that n is a predecessor that is routed through p. Following this procedure each node can save a map containing all predecessors and the corresponding direct predecessor. Figure 5 gives an example of how the resulting map looks like. Since this map might change due to weak mobility each entry is timestamped. Since we have a periodic flow of messages updates propagate quickly.



1	{1}
2	{2,3,4,5}

(a) A example routing graph (b) The predecessor map of node n

Figure 5. An example of the predecessor map of node n in a network of seven nodes

We can now easily determine the direct predecessors and the total number of predecessors $|P|$. There are essentially two different things, that c may be utilized for. The node may either choose to sense its environment, producing samples itself and sending them to base station. Or it allows its predecessors to increase the amount of data sent. Both choices use a certain amount of energy.

Any amount of flow x the node chooses to use for its own samples requires $x \cdot (P_{sense} + P_{tx})$. Any amount it allocates to any predecessor requires $x \cdot (P_{rx} + P_{tx})$. Here P_{sense}, P_{tx} and P_{rx} are the power needed to sense, transmit or receive a given data rate r respectively. They are given by the hardware, e.g. for the $CC2420$ radio used in the MicaZ [27] mote $P_{rx} = 62mW$ and $P_{tx} = 57.42mW(0dBm)$ [28] at a data rate $r = 250kbps$.

So basically each node with $|P| = n$ has to solve

$$\left(\sum_{i=0}^{n-1} x_i \cdot (P_{rx} + P_{tx})\right) + x_n \cdot (P_{sense} + P_{tx}) = c$$

and allocating flow x_i to predecessor p_i for $i \in \{0, \ldots, n-1\}$ and flow x_n to itself. In SDF each predecessors is treated equally leading to

$$n \cdot x \cdot (P_{rx} + P_{tx}) + x \cdot (P_{sense} + P_{tx}) = c.$$

Solving this for the flow x leads to

$$x = \frac{c}{n \cdot (P_{rx} + P_{tx}) + (P_{sense} + P_{tx})}.$$

But the flow is not only bound by the available power but also by the successor. If f_a denotes the allowed flow from the successor ($f_a = \infty$ if $s = b$) and $(n+1) * x > f_a$ we set $x = \frac{f_a}{n+1}$.

As result x gives both the own sampling rate and the allowed flow for each predecessor in P.

C. Control messages

Each node n sends regular control messages to the direct predecessor that contain the allowed flow f_a. For a predecessor p $f_a = x * |P_p|$ where P_p is the set of all predecessors routed through p and p itself. n will set its own sampling rate to $\max(x_{min}, x * s)$. Here x_{min} is the minimum sampling rate that is used during the initialization or if not enough energy can be harvested to guarantee a minimal service level.

V. Discussion

This section will discuss the update propagation and performance implications of the SDF protocol. For the update propagation we focus on worst case analysis. The performance is shown to decrease with the hop distance to the basestation.

A. Update propagation

Each node starts with no allowed flow and is using the minimum sampling rate. If a node is d hops away from the basestation it will take $d-1$ hops until it receives an update issued by a node directly attached to the basestation. In the worst case d might be n if n is the number of nodes. This happens if all nodes are lined up, such that they are only connected to one successor and predecessor. So in worst case it takes $n-1$ rounds until each node has received one update. If the nodes are randomly placed on a quadratic field, what we get on average is a control tree. Here updates propagate in $\log n$ time.

Lets revisit the worst case example illustrated in Figure 3. In the first case a internal node becomes a leaf node (cmp. Fig. 3(a)). Now the possible sampling rate is much higher since no energy is needed to relay messages. The predecessor map is updated if no more messages are received. Therefore it takes one round of SDF for the node to adjust it sampling rate. One round where there was less energy consumed than possible.

The second case is a leaf node that becomes an internal node (cmp. Fig. 3(b)). In the worst case it was a leaf node l directly attached to the base station and is now connecting all remaining $n-1$ nodes to the base station. Let us assume that the allowed flow for the remaining nodes is much higher than l can handle. l also uses all its power for sampling the sensors. So l is using more energy than the energy neutral consumption rate. Again it takes one round until n has knowledge of all predecessor and can adjust the own sampling rate accordingly. We have already established that in the worst case it now takes $n-1$ rounds for this update to propagate to all the attached nodes. It takes a total of n rounds until the node is again working with the energy neutral consumption rate. But at each round another node will receive the message and reduce its sampling rate. So the total power needed to relay the messages will converge to the energy neutral consumption rate. We take an update interval of 30 minutes for SDF. This means an update can propagate 48 hops in 24 hours. We will leave it to the reader to assess if a sensor network should route messages over more than 48 hops. But with SDF we only consider maximizing the sampling rate not optimizing the routing graph, so it might indeed become a problem. Adjusting the update interval accordingly to balance between update propagation and control overhead.

B. Performance

Longer paths lead to a higher delay in update propagation. But longer paths also lead to a lower *minimal sampling rate*. The minimal sampling rate is the lowest sampling rate achieved by any node during the operation of the network. Let us assume a network where each node, that is more than one hop away from the basestation, is not bound by its consumption rate. It is instead bound by the allowed flow granted by the successor. And let us also assume that there is only one node n directly attached to the basestation. Then the whole network will use the same sampling rate r, which is bound by the consumption rate of n. Now consider the same example but we now assume that there is one node $n_i \neq n$ that is bound by the own consumption rate. n_i will now calculate a flow $x < r$. So n_i and all predecessors of n_i will have a sampling rate $r_i \leq x < r$. From here on it is trivial to proof that there exists a leaf node l with sampling rate r_l and $r_l \leq r\forall r \in R$. R is the set of all sampling rates in the network. The main idea of a proof is based on the fact that the own sampling rate is either bound by the allowed flow of the successor or by the own consumption rate in which case it is either equal or lower than the sampling rate of the successor.

On a similar note it can be shown that there exists a node n, directly connected to the basestation, with sampling rate r_n with $r_n \geq r\forall r \in R$. Again the idea is that nodes are bound by their successor or the consumption rate. Since nodes attached to the base station are not bound by their successor, but all their predecessors are bound by them, this result is obvious.

VI. Evaluation

We evaluate our approach using the SIDnet-SWANS simulator [29]. SIDnet-SWANS is a fast event-based simulator for wireless sensor network based on Jist-SWANS [30].

We will use the theoretic optimum introduced by Kansal et al. [1] as the baseline. The baseline can only be calculated at the end of the simulation when all the relevant information is available. It calculates a multi path flow that can be sustained with knowledge of the energy neutral consumption rate. This represents the optimum, which is only possible to reach with global knowledge and a very good performance benchmark for SDF.

First we will describe the simulation setup. Then we will look at the initialization and the performance of SDF. The last step is to evaluate the overhead introduced by SDF.

A. Simulation setup

Unless otherwise noted the simulations are configured as follows. Each simulation will run for 480 hours starting on January 1st at 12:00 am. It is set at a latitude of 50°N with a mean cloud density of 0.6 and a cloud density variance of 0.3. This will roughly correspond to the solar radiation in Darmstadt, Germany. The effects of the cloud density will be evaluated later.

There are 30 nodes randomly positioned on a square field with a side length of $3,500$ feet. The solar panels have an area of $A = 9cm^2$ and an efficiency $e = 0.2$. They operate with a minimum sampling rate of $2.96\frac{bit}{s}$. Samples for the consumption rate are taken every 7 hours and 30 samples are retained. The 10% quantile is chosen and the sample confidence is set to 0.5.

We use the simple geographic greedy routing algorithm implemented in SIDnet-SWANS. It is similar to GPRS [24], without route recovery. Update messages are sent every 30 minutes.

Each configuration is run 20 times with different random seeds.

B. Initialization Phase

During the initialization all nodes start off using the minimum sampling rate. As the number of samples increases the consumption rate can be approximated better. We evaluate first how much samples have to be retained.

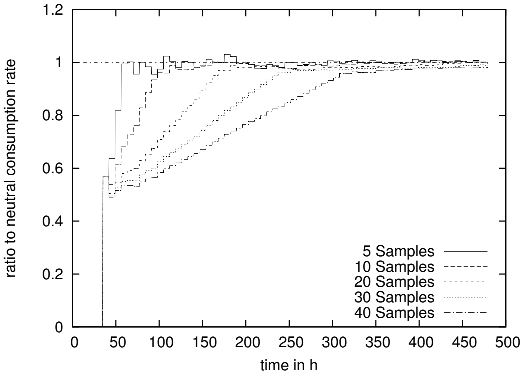


Figure 6. The ratio of the maximal calculated consumption rate to the energy neutral consumption rate for different numbers of retained samples.

Figure 6 illustrates the impact of changing the number of samples retained. We compare the calculated maximal consumption rate to the energy neutral consumption rate. If we retain too many samples the consumption rate will converge slower. This makes it harder for a node to react to changes in solar radiation. If we retain too few samples the consumable power might get higher than the energy neutral consumption rate. If we retain 30 samples we get a good balance between convergence time and quality of approximation. We can also note that after the transient phase is over, the predicition SDF uses is well over 90% of the possible energy neutral consumption rate. This indicates a good performance of the prediction while leaving some buffer for sudden changes in environmental conditions.

C. Performance Evaluation

The most important benchmark for SDF is the sustained sampling rate. As noted before we will compare SDF to the baseline in [1]. We already stated that the baseline assumes a homogeneous sampling rates. This implies that the sampling rate is bound by the weakest node in the network. We measure the sampling rates for SDF and the baseline at the end of the simulation time after 480 hours. For SDF we obtain the minimal, maximal and average sampling rate achieved in the network. For a fair comparison the minimal sampling rate is evaluated against the baseline as it gives the performance of the weakest node. We also show the average and maximal sampling rates as they illustrate the advantage of allowing heterogeneous sampling rate.

We keep the deployment area fixed. Thus by changing the number of nodes we will change the node density. We do abstract from the routing algorithm. But in general increasing the node density will lead to longer average paths. As stated in the discussion this implies a possible decrease in the minimal sampling rate obtainable.

Indeed the aforementioned effect is exhibited in Figure 7(a). Increasing the node density has almost no effect on the baseline while the minimal sampling rate for SDF drops from $34bps$ to $30bps$. The baseline performs better due to the global knowledge on all possible connections between nodes. Figure 7 presents the relative performance of the minimal, average and maximal sampling rate to the baseline. SDF is capable of sustaining over 80% of the baseline performance. It is important to note that while the minimal sampling rate drops the maximal sampling rate increases. Since we increase the node density the number of nodes directly attached to the basestation increases. Both effects, longer average path length and more nodes directly attached to the basestation, lead to the divergence between minimal and maximal sampling rate. On average the sampling rate achieved by SDF stays almost constant.

Increasing the cloud density leads to a drop in the average consumption rate. Thus it must lead to a decrease in all sampling rates. As energy gets scarce the performance becomes crucial. The baseline gives a theoretical optimum. SDF should converge against the baseline as cloud density increases.

Figure 8 illustrates the variation in cloud density. In relation to the baseline the sampling rates achieved with SDF are slightly increasing (cmp. Fig.8(b)). With a mean cloud density of 1.0 we achieve over 85% of the baseline performance for the minimal sampling rate.

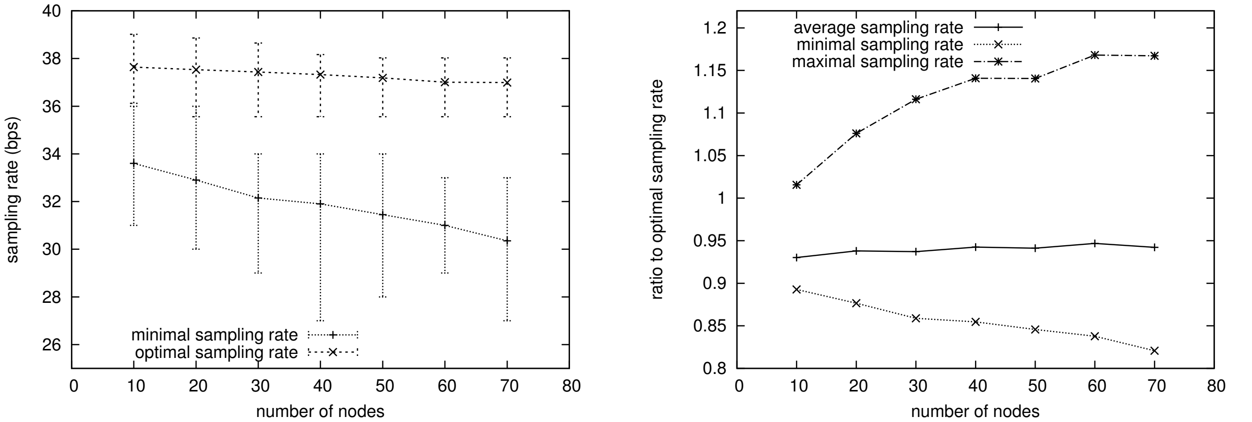
D. Control Message Overhead

SDF introduces control messages. Each node sends one message to each direct predecessor per round. Leaf nodes send no messages. If n is the total number of nodes in the network and m the number of messages sent each round we note that $m < n$.

The goal of SDF is to maximize the load in the network. So introducing additional load is acceptable if it is small compared to the performance gained. We evaluate both the message and bit ratio to the total number of messages and bit send. The bit ratio is important since compared to messages containing sensor data, control messages are small.

The worst case for both ratios is the initialization phase. Each node samples at the minimal sampling rate. The amount of data is small. Since the control overhead is fixed this marks the worst case.

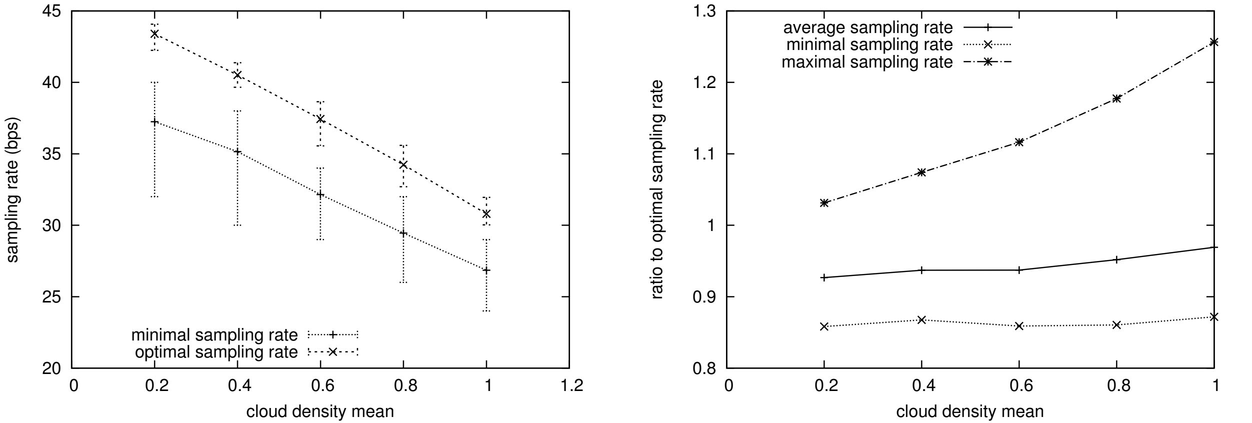
Obviously the overall shape of the message and bit ratios are similar (cmp. Fig. 9). We observe that during the 24 hours of initialization nothing really happens. The message ratio is around 35% with a peak of 43%. So even though all nodes sense at the minimum sampling rate less than 50% are control messages. We also observe the spikes every 30 minutes when control messages are sent. For the bit ratio, even though the



(a) The minimal sampling rates achieved after 480 hours compared with the optimal sampling rates

(b) The ratios of the average, minimal and maximal sampling rates in relation to the optimal sampling rate

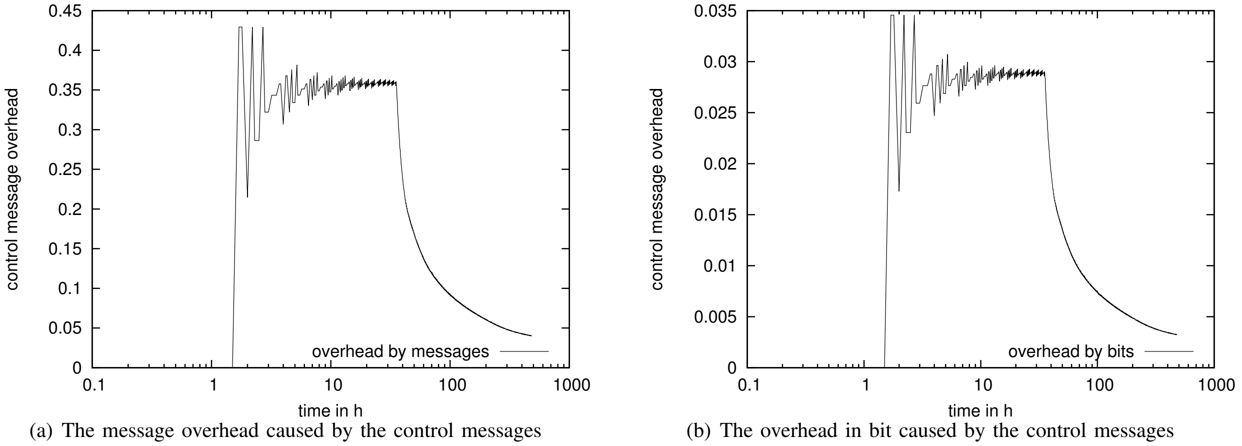
Figure 7. The impact of the node density on the average, minimal and maximal sampling rates



(a) The minimal sampling rate achieved after 480 hours compared with the optimal sampling rate

(b) The ratio of the average, minimal and maximal sampling rate in relation to the optimal sampling rate

Figure 8. The impact of the mean cloud density on the average, minimal and maximal sampling rates



(a) The message overhead caused by the control messages

(b) The overhead in bit caused by the control messages

Figure 9. The overhead caused by control messages relative to the total number of messages

shape is similar, the peaks are around 3.5% of the total bits send in the network. As soon as the initialization phase is over both ratios drop rapidly. The message ratio drops to 4% and the bit ratio becomes negligible at 0.3%. This is all not taking possible enhancements into account. We are now sending control messages even if nothing has changed. While there is still room for improvement the overhead is negligible if a maximization of sampling rates is possible.

VII. Conclusion and Future Work

In this paper we have proposed SDF, an approach to maximizing the sampling rate in solar-harvesting sensor networks. It supports the feasibility of solar-harvesting sensor networks as a means to overcome energy constraints, which traditionally have been posing a hard bound on sensor networks. We introduced the notion of the energy neutral consumption rate, at which the harvested and consumed energy are balanced, to allow theoretically perpetual operation. We further discussed the assumptions and theoretical models that constitute the foundations for SDF. A simple cloud model, which models the spatio-temporal distribution of solar radiation, was introduced to establish a realistic evaluation environment.

SDF utilizes the harvested energy while remaining energy neutral. It works in three simple steps that are repeated periodically on each node. First, a node approximates the energy neutral consumption rate as the own consumption rate. Our evaluation indicated that retaining no more than 30 samples is sufficient for the prediction. Next, a node calculates a flow for itself and the predecessors. This flow is bound by the minimum of either its energy prediction or the possible throughput of its successor nodes on the path to the data sink. During the last step it sends control messages to all direct predecessors bounding their allowed respective flow.

The model shows that updates, in a network with n nodes, are propagated in $\log n$ steps in the expected, or a maximum of $n-1$ steps in the worst cases.

For realistic evaluation, we measured the impact of node and cloud density on the sustained sampling rate, and hence the performance of SDF, comparing it to the theoretical optimum, as given in [1]. The results demonstrated that the minimal sampling rate obtained by SDF is over 80% of the baseline's sampling rate. Comparing the predicted energy to the energy neutral consumption rate illustrated a possible energy utilization of over 90%. Finally, they indicate that the message and bit overhead is negligible compared to the increase in samples retained. The control messages even under adverse situations accounted for only 4% and 0.32% of the message and bit load respectively.

We are currently deploying SDF on a solar harvesting sensor network in Darmstadt to collect real world experience with energy harvesting. In this course we are aiming at analysing the underlying routing. Developing a solar-aware routing protocol that will complement SDF's performance by giving a balanced routing tree with short path lengths, while utilizing available harvested energy, we are expecting to achieve even higher sampling rates.

References

[1] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power Management in Energy Harvesting Sensor Networks," *ACM Transactions on Embedded Computing Systems*, 2007.

[2] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. Srivastava, "Design considerations for solar energy harvesting wireless embedded systems," in *IPSN*, 2005.

[3] X. Jiang, J. Polastre, and D. Culler, "Perpetual Environmentally Powered Sensor Networks," in *IPSN*, 2005.

[4] M. Minami, T. Morito, H. Morikawa, and T. Aoyama, "Solar Biscuit: a battery-less wireless sensor network system for environmental monitoring applications," in *Workshop on Networked Sensing Systems*, 2005.

[5] P. Sikka, P. Corke, and L. Overs, "Wireless sensor devices for animal tracking and control," in *IEEE International Conference on Local Computer Networks*, 2004.

[6] C. Alippi, R. Camplani, C. Galperti, and M. Roveri, "A Robust, Adaptive, Solar-Powered WSN Framework for Aquatic Environmental Monitoring," *IEEE Sensors Journal*, 2011.

[7] T. Clark, "Numerical Simulations with a Three-Dimensional Cloud Model: Lateral Boundary Condition Experiments and Multicellular Severe Storm Simulations." *Journal of Atmospheric Sciences*, 1979.

[8] R. A. Anthes, "A Cumulus Parameterization Scheme Utilizing a One-Dimensional Cloud Model," *Monthly Weather Review*, 1977.

[9] S. Roundy, D. Steingart, L. Frechette, P. Wright, and J. Rabaey, "Power sources for wireless sensor networks," *LNCS*, 2004.

[10] C. Vigorito, D. Ganesan, and A. Barto, "Adaptive control of duty cycling in energy-harvesting wireless sensor networks," in *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 2007.

[11] A. Kansal and M. Srivastava, "An environmental energy harvesting framework for sensor networks," in *Symposium on Low Power Electronics and Design*, 2003.

[12] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *ACM SIGPLAN Notices*, 2000.

[13] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *IPSN*, 2005.

[14] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs, "Long-Duration Solar-Powered Wireless Sensor Networks," in *Workshop on Embedded networked sensors*, 2007.

[15] L. Lin, N. Shroff, and R. Srikant, "Asymptotically Optimal Energy-Aware Routing for Multihop Wireless Networks With Renewable Energy Sources," *IEEE/ACM Transactions on Networking*, 2007.

[16] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, 2003.

[17] T. Voigt, H. Ritter, and J. Schiller, "Utilizing Solar Power in Wireless Sensor Networks," in *IEEE International Conference on Local Computer Networks*, 2003.

[18] T. Voigt, H. Ritter, and J. Schiller, "Solar-aware routing in wireless sensor networks," *Personal Wireless Communications*, 2003.

[19] J. Schiller, A. Liers, H. Ritter, R. Winter, and T. Voigt, "ScatterWeb - Low Power Sensor Nodes and Energy Aware Routing," in *Hawaii International Conference on System Sciences*, 2005.

[20] T. Voigt, A. Dunkels, J. Alonso, H. Ritter, and J. Schiller, "Solar-Aware Clustering in Wireless Sensor Networks," in *International Symposium on Computers and Communications*, 2004.

[21] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, 2002.

[22] W. Shu, X. Liu, Z. Gu, and S. Gopalakrishnan, "Optimal Sampling Rate Assignment with Dynamic Route Selection for Real-Time Wireless Sensor Networks," in *Real-Time Systems Symposium*, 2008.

[23] S. Bandyopadhyay, Q. Tian, and E. Coyle, "Spatio-temporal sampling rates and energy efficiency in wireless sensor networks," *IEEE/ACM Transactions on Networking*, 2005.

[24] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *MobiCom*, 2000.

[25] T. Brock, "Calculating solar radiation for ecological studies," *Ecological Modelling*, 1981.

[26] International earth rotation & reference systems service, "Useful Constants," http://hpiers.obspm.fr/eop-pc/models/constants.html.

[27] Crossbow Technology Inc, "MICAz Datasheet," www.openautomation.net/uploadsproductos/micaz_datasheet.pdf.

[28] A. Chipcon, "CC2420 datasheet," http://focus.ti.com/lit/ds/symlink/cc2420.pdf.

[29] O. Ghica, G. Trajcevski, P. Scheuermann, Z. Bischof, and N. Valtchanov, "SIDnet-SWANS: A simulator and integrated development platform for sensor networks applications," in *ACM Conference on Embedded Network Sensor Systems*, 2008.

[30] R. Barr and Z. Haas, "JiST/SWANS," http://jist.ece.cornell.edu.