A Distributed Smart Application for Solar Powered WSNs

T.V. Prabhakar[1,2], S.N. Akshay Uttama Nambi[1], R. Venkatesha Prasad[2], S. Shilpa[1], K. Prakruthi[1], and Ignas Niemegeers[2]

[1] Centre for Electronics Design and Technology, IISc, Bangalore, India
[2] Delft University of Technology, The Netherlands
{tvprabs,akshay,sshilpa,kprakruthi}@cedt.iisc.ernet.in,
{r.r.venkateshaprasad,I.G.M.M.Niemegeers}@tudelft.nl

Abstract. Energy harvesting (EH) is a major step in solving the critical issue of availability of energy for sensor nodes. However, it throws many challenges. The applications built on the sensor networks powered by EH need to adapt their operations yet serve the purpose. We propose a distributed smart application for a multihop sensor network and in general in the future Internet of Things (IoT) where a network node executes an optimal number of policies to minimize the difference between available energy and consumed energy (called residual energy) for the execution of an application *policy*. We formulate this as a multi-criteria optimization problem and solve it using linear programming Parametric Analysis. We demonstrate our approach on a testbed with solar panels. We also use a realistic solar energy trace with a three year database including seasonality. The smart application is capable of adapting itself to its current energy level as well as that of the network. Our analytical results show a close match with the measurements conducted over testbed.

Keywords: Energy Harvested Wireless Sensor Network (EHWSN), multi-criteria optimization, distributed smart application.

1 Introduction

Recently Energy Harvesting Wireless Sensor Network (EHWSN) is a reality making WSNs independent of batteries. This is paving way to rapid growth of interesting applications for sensing and control. The "ZigBee Green" [1] is specifically designed to run on energy harvesting (EH) sources. Recent advancements in material science and MEMS research have made harvesting using Thermo Energy Generators (TEGs) and vibration (unusable till recently) as potential energy sources. Moreover, energy storage in thin film batteries and low leakage super capacitors with several thousand charge-discharge cycles offer efficient energy storage which is fueling the enormous growth. Thus our aim in this work is to make the best possible use of all these positive developments. EHWSN are used wherever remote monitoring and control are required over a wide spectrum of scenarios. One end of this spectrum is an intrusion detection system deployed

R. Bestak et al. (Eds.): NETWORKING 2012, Part II, LNCS 7290, pp. 291–303, 2012.

in a wireless tripwire paradigm for monitoring an international border, and the other end is a simple wireless switch. While the former application requires an energy storage buffer, the latter application requires energy generation and its usage on the fly. Moreover, we could see that they cover multihop and single-hop settings respectively. Other applications are intelligent transportation, smart buildings, pollution monitoring, agriculture and climate change, health care including body area networks and other similar applications covering the complete domain of Internet of Things (IoT). For outdoor applications, photovoltaic panels offer significantly higher power compared to other harvesting sources, which use reflected light or even a partial shade. This is specifically used in intrusion detection settings where an intruder cannot camouflage as a sensor node himself. Additionally, seasonal variation in sunlight plays an important role in ensuring continuous and untethered operation of solar EHWSN nodes. Thus, applications have to continuously adjust to varying instantaneous power and yet accomplish their primary assigned task. Usually EHWSN nodes are wide spread and the communication range is limited. Thus multihop operations are sought within acceptable performance deterioration. There are significant differences between

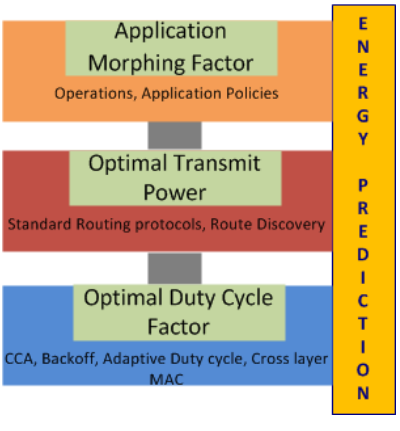

Fig. 1. Optimal Parameters for Application, Routing and MAC Layer

battery driven Wireless Sensor Networks (WSNs) and EHWSNs. In WSNs it is sought to: (a) reduce the energy consumption to increase the battery life, and (b) have a network wide policy to support a network lifetime of about "X" (say) number of hours or to support the largest partitioned network for a longer duration. A large body of work in WSN is limited to maximizing the policy subject to a given energy constraint or minimizing energy consumption to satisfy a network policy requirement. In direct contrast, for an energy harvested network, the objective is to maximize the application policies and also maximize the energy consumption. In other words, minimize the residual energy between the available and consumed energy in each discrete time slot and maximize the application policy (in turn usage of EHWSNs). In nodes with EH, if energy is not utilized

properly, it would be lost due to leakage or lack of storage capacity, thus the goal is to minimize residual energy. Our work in this paper uses a multi-criteria optimization approach to find an optimal application policy versus residual energy curve. The significance of optimal application policy versus residual energy curve is to show the trade-off between global perspectives of application policy and residual energy while using EH.

In this paper, the goal is to implement an example of a distributed smart application for environment monitoring application where the nodes are powered with EH, *esp.* Solar power. Fig.1 shows the parameters to be controlled at different layers to achieve maximization of application policy and minimization of residual energy. Results obtained using analysis followed by our testbed implementation are encouraging and even "near real time" applications can perform optimally.

2 Related Work

Literature on EHWSN networks is mostly limited to a single hop network or extensive simulation studies. Several energy harvesting sources like thermal, mechanical, solar, acoustic, wind and wave for embedded systems are discussed in [2]. Improvement in life time is shown in [3] where supercapacitor and battery are together charged from a solar panel. The applications can survive for several years by adjusting their duty cycle. Solar energy predictions using simple time series such as Exponential Weighted Moving Average (EWMA) is commonly used [4]. In [5] a three state markov chain model is used to predict the solar energy source and their transition probabilities are restricted for day time only. Energy samples from a simulation model are also generated from the model. Several optimization techniques have been discussed in literature for battery driven multihop WSN's in the past where the objective is to maximize the network lifetime by minimizing energy consumption [6], [7]. With a view on utilizing the harvested energy in an optimized manner, adapting the performance of an application while respecting the limited and time-varying amount of available power in EHWSN is discussed in [8]. A formal model that actively adapts application parameters such as "rate of sensing" is used to optimize the performance. The simulation study does not consider network related parameters or network wide energy levels. A "Lazy Scheduling Algorithm" is proposed and tested for its effectiveness by assigning several power values to the system in [9]. Task admittance and future energy prediction is carried out with energy variability characterization curves generated by modeling the power source. Dynamic adaptation of the duty cycle of a node to ensure energy neutral operation by observing deviations in EH from an estimated model is discussed in [10]. Adaptive control theory to formulate a linear-quadratic optimal tracking problem for maximizing task performance and minimal duty cycle variations is proposed in [11]. A comparison of several energy harvesting based routing protocols in a simulation setup is in [12]. We demonstrate a distributed smart application which optimizes the necessary parameters to ensure maximization of application policy and minimization of residual energy. We believe performing

energy neutral operations based on node's own energy is not optimal. Instead we propose to derive the optimal number of operations and the residual energy based on currently available energy at the node, predicted energy and network energy (i.e., neighbouring nodes). Particularly, we tune the duty cycle factor, transmission power factor and the application morphing factor to achieve optimal performance. Amongst the rich literature on EHWSN, *inter alia*, we position our work based on these attributes: (a) Harvested energy is the only available source; (b) EHWSN nodes are deployed in a testbed with MAC layer contention and adaptive duty cycling; (c) Inter-node distance of about 25-30ft with a realistic channel where Wi-Fi and other 2.4GHz radios are present; (d) Energy prediction from physical solar trace data including seasonality over a three year database is used; and (e) A smart application which morphs itself into several forms based on the energy level of the node as well as its network in a manner where performance deterioration is within acceptable limits.

3 Smart Application – The Model

We argue that node based task scheduling and pre-emption is perhaps not best suited for multihop EHWSN. For instance, it is possible that low priority tasks located at the head of the queue gets executed during energy stress disregarding network's requirement and energy budgets. We propose that applications have to be "smart". A typical WSN application comprises of operations such as sense, compute, store and communicate. These operations are required to be executed based on a policy over several time slots. Communication comprises of packet transmission and reception. Packet forwarding refers to relaying i.e., reception and transmission. Transmission power control is mandated due to energy considerations. Smart application design and its successful implementation require several parameters such as: (a) Available energy in the storage buffer called "*real energy*" (E_A). (b) Predicted energy that is possibly harvested in the next two or three time slots -"*virtual energy*" (E_P). (c) The "*network energy*" (E_N) which is essentially the energy available in the neighbouring nodes (or energy available along a route). Thus our smart application has to exploit the energy in the slot to the maximum such that the "*residual energy*" between the available energy and consumed energy in a slot is minimum. Since energy replenishment occurs in EHWSN, energy utilization should be such that maximum number of tasks or operations should be completed. As one can observe, Packet Reception Ratio (PRR) and application performance depends on accurate measurement of the above parameters. Since available energy is stored in a super-capacitor, we calculate the real energy by sensing the voltage across the capacitor. We use Holt-Winters model to predict the virtual energy in a node. We leverage on the RSSI values contained in routing messages for periodic update on the energy level in the network. Since available energy on each node varies, the application morphs itself into several forms in a manner that performance deterioration is within acceptable limits. For the purposes of ease of implementation, we discretize the available energy levels into 4 levels represented by $E_i, i \in \{0, 1, 2, 3\}$.

E_0, E_1, E_2 and E_3 correspond to `Lowest survivable`, `Minimum`, `Intermediate` and `High` energy levels respectively. Similarly, the residual energy levels are represented by $E'_i, i \in \{0, 1, 2, 3\}$, where E'_0, E'_1, E'_2, and E'_3 correspond to `Lowest survivable, Minimum, Intermediate and Higher` residual energy levels respectively. Let the stored energy, harvested energy, predicted (virtual) energy at time slot k on node 'n' be represented by $E_{S(k)(n)}$, $E_{H(k)(n)}$ and $E_{P(k)(n)}$ respectively. Let $E_{N(k)(s)}$ be the energy available on neighbouring node 's' at time slot k and $E_{DC(k)(n)}$ denotes the minimum energy required for the node for one duty cycle. We denote the maximum energy capacity of a supercapacitor as E_{max}. The harvested energy for node n is $E_{H(k)(n)} \geq 0$. The available energy at each time slot k on node n is represented by $E_{A(k)(n)} = E_{S(k)(n)} + E_{H(k)(n)}$ and $E_{A(k)(n)} \leq E_{max}$.

3.1 The Base Application and Morphing

We call a 'base application' to essentially mean a set of policies which would be executed if available energy is high. At the beginning of each time slot, the application checks the available energy, virtual energy and the policy that requires to be executed. If there is sufficient harvested energy, the node initiates network support for its policy and prepares itself to execute the corresponding policy, else, the application on the node decides to morph into a slower versions (lesser functionalities than the base application that is specified *apriori*) in as many steps as energy is quantified. For example, when ensuing two time slots reports the virtual energy for the node is low and current energy level is low, the application morphs to a lower version by partial execution of the policy with a few operations backlogged for later execution. A partial fulfillment for a specific policy is necessary to ensure that priority for certain operations in the policy has soft guarantees. For instance, when the node is high on own energy and virtual energy (i.e. both the energy values are at E_3), the node can complete its policy execution in the current time slot including any backlog operations. This energy expenditure is possible only when the network provides the necessary support. With its completion, the node may move to either E_1 or E_0 depending on the level of energy depletion. At the same time, if the future time slots predict a low energy, the base application chooses to morph to a slower form and thus deplete lesser energy and move to say either to E_2 or E_1. In essence, $E_{A(k+1)(n)} = E_i, \forall i \in \{0, 1, 2, 3\}$.

3.2 Policy Models

Policy Model - Operation Set: The operation set includes both node and network related operations. Basic operations may include sensing, computing, communication and storage. Each element is associated with fixed energy consumptions, where, transmission of 128 bytes at 0dBm consumes 0.314mJ, reception of 128 bytes packet consumes 0.4 mJ, reading and writing 1 byte data to flash consumes 0.3 μJ and 1.23 μJ respectively.

Policy Model - Policy Set (P): A policy set P defines the set of operations and their order of execution obtained from an operation vector that a sensor node n has to follow. Typically, a sensor node might be associated with two or more policies, and the specific set of policy to be executed in a particular time slot is defined as part of the base application. Thus the policy set for a node can be defined as,

$$P = \{P_1, P_2, P_3, ..., P_x\}. \tag{1}$$

A typical policy on a sensor node D can be $P_d = \{P_1, P_2, P_3\}$ where, $P_1 =$ {Sense, Transmit}, $P_2 =$ {Sense, Compute, Write} and $P_3 =$ {Read, Compute, Transmit}. For example, a relay node R may have the policy set $P_r = \{P_4, P_5, P_6\}$ where, $P_4 =$ {Receiving, Forwarding}, $P_5 =$ {Read, Compute, Forward} and $P_6 =$ {Receive, Write}. At each time slot, the node decides to execute subset of policy set denoted by P'. The energy required for policy execution in time slot k is given by,

$$E_{R(k)} = \sum_{i=1}^{m} E_i {P_i}', \quad \begin{cases} m = x, P'_i = P \\ m < x, P'_i \subset P \end{cases} \tag{2}$$

where, P' is the subset of the policy set, x the maximum number of policies and E_i is the energy consumed for i^{th} policy set.

3.3 Solar Energy Prediction Model

One of the key requirements of our approach is accurate prediction of virtual energy at each node. We used solar energy trace database with data collected over three years by Lowry range solar station [13] from year May 2008 to August 2011 with three data sets corresponding to direct, reflected, and diffused sunlight. We applied these spatio-temporal data sets across the complete network as shown in Fig.3. We evaluated two energy prediction models *viz* Exponentially Weighted Moving Average (EWMA) prediction model [4], and the Holt-Winters (HW) time series prediction model for these data sets. Since solar energy output can have seasonal variations, intelligent prediction algorithms should exploit the trend and seasonality available within the data sets. We applied Holt-winters (HW) time series prediction model commonly used for forecasting in stock markets, and is given by,

$$E_P(k) = \epsilon \frac{E_S(k)}{I(k-1)} + (1-\epsilon)(E_P(k-1) + b(k-1)), \tag{3}$$

where, $E_P(k)$ is the predicted value at k^{th} time slot, $E_S(k)$ is the stored energy value. Further, $b(k) = \gamma(E_P(k) - E_P(k-1)) + (1-\gamma)b(k-1)$ and $I(k) = \beta \frac{E_S(k)}{E_P(k)} + (1-\beta)I(k-1)$. The terms ϵ, β, γ are the weighting, trend and seasonality factors respectively. These are constants whose values are estimated such that their LMSE is minimized and their estimates are found to be 0.906, 0.1 and 0.650. We compared the real solar data with virtual solar data obtained from EWMA and HW prediction models for five days of a month. We found that maximum error percentage of about 45% for EWMA prediction model and around 7% for HW energy prediction model.

4 Multi-criteria Optimization Problem

The difference between available energy and consumed energy should be minimized for a given real, virtual and network energy to use the energy as much as possible. The problem is to find an optimal policy execution. We cast this problem as multi-criteria optimization with two objectives: (a) Maximizing application policy execution, and (b) Minimizing residual energy.

4.1 Maximizing Application Policy

In each time slot k, the objective is to maximize P' policies executed by a node. Thus the application policy utility U is given by,

$$U = \sum_{i=1}^{m} \alpha_i {P_i}' \qquad s.t. \quad 0 \leq \alpha_i \leq 1, \tag{4}$$

where, ${P_i}'$ is the set of policies to be executed and α_i indicates the morphing factor for the i^{th} policy set. When $\alpha_i = 1$, the entire policy set P is executed (i.e., ${P_i}' = P$).

4.2 Minimizing Residual Energy

Let X_k be the total available energy in time slot k. This is a function of $E_{A(k)}$, $E_{P(k)}$ and $E_{N(k)}$. Let Y_k be the energy required to execute the policies in time slot k as given by Eq. (2) and Z_k is the energy required for operation of the node with a duty cycle factor of δ. In the event of $X_k < [Y_k - Z_k]$, we have

$$\hat{Y_k} = f(Y_k, \alpha_k) \qquad s.t. \;\; 0 \leq \alpha_k \leq 1; \qquad Z_k = f(E_{DC}, \delta_k) \qquad s.t. \;\; 0 \leq \delta_k \leq 1; \tag{5}$$

$\hat{Y_k}$ is the consumed energy to execute the set of policies based on the morphing factor 'α_k' and Z_k is the energy required for adaptive duty cycle based on harvested energy in each time slot. We define the residual energy utility, V, as

$$V = [X_k - \hat{Y_k} - Z_k] s.t. \qquad X_k \leq E_{max}, \;\; [X_k - \hat{Y_k} - Z_k] \geq E_0'; \tag{6}$$

where E_0' is the lowest survivable residual energy level required for the system to be operational. Now, our multi-criteria optimization problem is formulated as,

$$\text{MOPT: } max \; U = \sum_{i=1}^{m} \alpha_i * {P_i}' \tag{7}$$

$$\text{and} \qquad min \; V = [X_k - \hat{Y_k} - Z_k]$$

First we consider a single objective optimization problem for a given V (i.e., fixing one of the objectives).

$$\text{OPT}(V) \; max \;\; U = \sum_{i=1}^{m} \alpha_i * {P_i}'; \qquad s.t. \; [X_k - \hat{Y_k} - Z_k] = V \tag{8}$$

For all possible values of $V \in [E_0', E_3']$ we obtain their corresponding OPT(V), optimal points in application policy versus residual energy curve. This gives a mapping from V to U, which is denoted as $f : V \rightarrow U$, where for each point $(U, V), U = f(V)$ is the maximum application policy utility that can be obtained.

5 Solution

The solution to OPT(V) is obtained by using the special structure of linear programming such as the Parametric Analysis (PA). The overall approach is to obtain $f(V)$ by solving a finite number of linear programs to provide the morphing factor "α", transmission power factor and duty cycle factor "δ". These values ensure minimization of residual energy and maximization of application policies. Using PA, we study the perturbation of V and its effect on the optimality of OPT(V) to obtain the perturbation factor λ. We use boldface to denote matrices and vectors. For a given V, the current optimal basis of OPT(V) could still be optimal when there is a perturbation on V. Thus, the range $[E_0', E_3']$ can be partitioned into consecutive small intervals, each corresponding to a different optimal basis. For a given V, let the optimal basis matrix be $\mathbf{B}$ and the non-basic matrix be $\mathbf{N}$. Let the optimal solution to OPT(V) is $(\mathbf{x_B}, \mathbf{x_N})$, where x_B and x_N denote the values of basic and non-basic variables respectively. Further, let $\mathbf{c_B}$ and $\mathbf{c_N}$ denote the coefficient vectors of the objective function of application policy utility U for the basic and non-basic variables respectively. $\mathbf{b}$ is the vector with coefficients of V . The corresponding canonical equations are:

$$U + (\mathbf{c_B}\mathbf{B^{-1}}\mathbf{N} - \mathbf{c_N})\mathbf{x_N} = \mathbf{c_B}\mathbf{B^{-1}}\mathbf{b}$$
$$\mathbf{x_B} + \mathbf{B^{-1}}\mathbf{N}\mathbf{x_N} = \mathbf{B^{-1}}\mathbf{b}$$

Let the perturbation on parameter V be $V + \lambda$. Then the vector $\mathbf{b}$ is replaced by $\mathbf{b} + \lambda\mathbf{b}'$ with vector $\mathbf{c_B}\mathbf{B^{-1}}\mathbf{N} - \mathbf{c_N}$ unchanged. Now $\mathbf{B^{-1}}\mathbf{b}$ will be replaced by $\mathbf{B^{-1}}(\mathbf{b}+\lambda\mathbf{b}')$ and accordingly the objective becomes $\mathbf{c_B}\mathbf{B^{-1}}(\mathbf{b}+\lambda\mathbf{b}')$. As long as $\mathbf{B^{-1}}(\mathbf{b}+\lambda\mathbf{b}')$ is non-negative, the current basis remains the optimal basis. The value of λ for another basis to become optimal can be determined as follows: Let $S = \{i : \overline{\mathbf{b_i}'} < 0\}$ where $\overline{\mathbf{b_i}'} = \mathbf{B^{-1}}\mathbf{b}'$. If $S = \emptyset$, then the current basis is optimal for all values of $\lambda \geq 0$. Otherwise, let

$$\hat{\lambda} = \min_{i \in S} \frac{\overline{\mathbf{b_i}}}{-\overline{\mathbf{b_i}'}}. \tag{9}$$

Let $\lambda_1 = \hat{\lambda}$, then the current optimal basis is optimal for $\lambda \in [0, \lambda_1]$, where $\mathbf{x_B} = \mathbf{B^{-1}}(\mathbf{b}+\lambda\mathbf{b}')$ and the optimal objective is $\mathbf{c_B}\mathbf{B^{-1}}(\mathbf{b}+\lambda\mathbf{b}')$. When $\lambda > \lambda_1$, the basis $\mathbf{B}$ is no longer optimal. Thus, we need to choose a variable $\mathbf{x_r}$ to leave the basis, where the minimum in Eq.(9) is attained for $i = r$. $\mathbf{x_s}$ is chosen by the dual simplex method rule [14] and we update the canonical equations based on the new optimal basis obtained and get (U, V) pair as defined earlier for OPT(V). The process is repeated to find the range $[\lambda_1, \lambda_2]$ over which the new basis is optimal. Thus, starting from $V = E_0'$, we repeat the steps iteratively to find

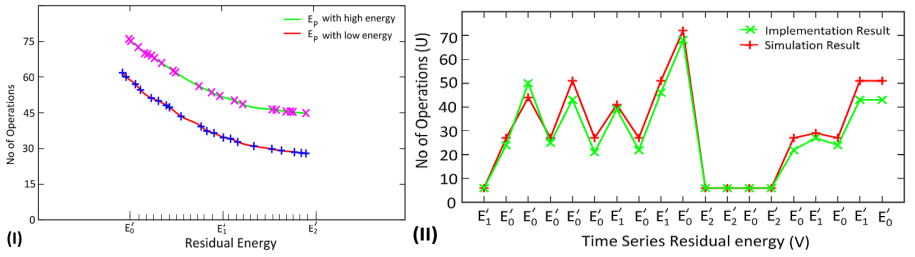

Fig. 2. (I) Optimal application policy versus residual energy when available energy is E_3; (II) Residual energy versus number of operations

different bases until we reach E'_3. The series of $\hat{\lambda}$ for these bases will partition $[E'_0, E'_3]$ into small intervals. Thus, by executing the above steps repeatedly, we obtain a series of (U, V) pairs, each corresponding to an optimal basis. We obtain the application policy versus residual energy optimal curve by connecting these endpoints consecutively. Fig.2(I) shows the simulation results where x-axis represents the residual energy (V) between the available and the consumed energy and y-axis represents the total number of operations executed by the node i.e., U. Fig.2(I) shows the optimal curve for a node when available energy is E_3 and its neighbour node energy is either E_1 or E_2 or E_3. Curves 1 and 2 indicate the optimal curve when the energy prediction for future slots is high and low respectively. When the available energy on the node was E_3, the morphing factor α obtained were 0.9890, 0.6269, 0.2797 for residual energy of E'_0, E'_1 and E'_2 respectively. Similar curves are obtained for other available energy levels i.e., E_2, E_1 on the source node. Fig.2(II) shows the time series of residual energy against the number of operations performed by the source node obtained via simulation and implementation. The source node runs the diffused energy profile from the LRSS. We can see that the simulation result closely matches the implementation. The difference in number of operations between simulation and implementation is around 6% due to error in prediction. For instance, at time slot 3 of Fig.2(II), the available energy at the source node was wrongly predicted and hence the node performed more operations compared to the simulation result.

6 Experimental Setup and Results

6.1 Implementation and Deployment Scenario

Each node in the network is equipped with α, δ factors for various values of available, predicted and network energy (E_A, E_P and E_N) as part of the application initialization. For our experiments we used a time slot of 2s where the application is designed to "read" a sensor value multiple times and "compute" an average value. This value is then stored in memory using a "write" operation and also schedule a packet for "transmission". The packet size containing the

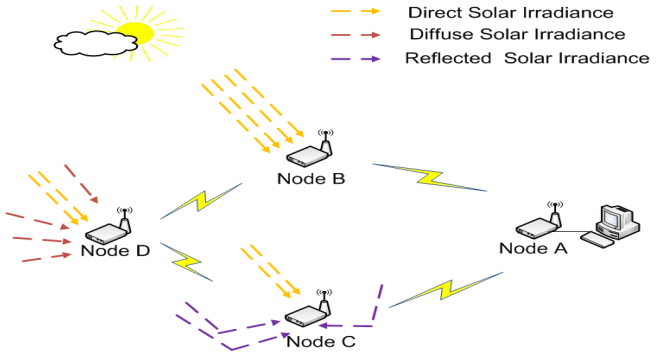

Fig. 3. Experimental Setup

sensor value is 128 bytes with base station as the specified destination address. Our setup was tested with 4 nodes in a tree topology with one node requiring at least two hops to reach the base station as shown in Fig.3. We used solar energy harvesters with varying energy profile across the network. Since we require realistic spatio-temporal varying power across all the network nodes, the solar trace database available from [13] is replicated over the experimental setup. The three year database with trend and seasonality has power output for direct, diffused and reflected sunlight. We built our own customized hardware motes using TI's MSP430 microcontroller and Chipcon's IEEE 802.15.4 standard CC2520 radio. We broadly divided the network into data collecting *sensor nodes* and data relaying *communication/relay nodes*. From Fig.3, Nodes B, C are communicating nodes and Node D is the sensor node. Communication nodes have the task of forwarding packets to the base station. Node A is utility powered and it is configured as the base station or data sink node for the network. In Fig.3, Node B runs on direct sunlight profile, Node C runs on reflected sunlight profile and Node D runs on diffused sunlight profile. The power output from the solar panels was varied by switching "on" and "off" electrical lamps placed above solar panels. We scaled down the power output obtained from Lowry Range Solar Station (LRSS) to match the laboratory solar harvesters. Note that various other combination of the solar power pattern is possible, however we found that using the model from the source is tractable for repetition of the scenarios. We also omit here the details of minor modifications to MAC and other protocols (such as exchange of energy information) due to paucity of space. Each node in the network has two modes of operation *viz*, (a) active mode and (b) sleep mode. While, in *active mode*, a node can transmit and receive packets, in *sleep mode*, node turns off the radio transceiver to reduce the energy consumption and its duration is dictated by duty cycle factor δ.

6.2 Application

We show the performance of the distributed smart application running on sensor Node D (in Fig.3) associated with a diffused energy profile. The solar emulator

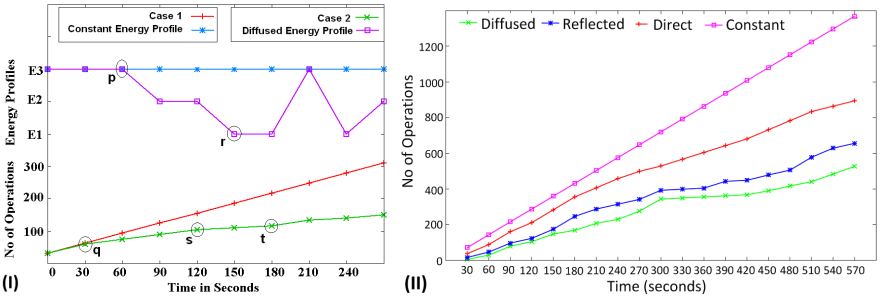

Fig. 4. (I) Morphed application on the source node for diffused and constant energy profile. (II) Morphed application on the source node for various energy profiles.

placed above Node D generates the diffused energy profile where most of the time either (E_1) or (E_2) amount of harvested energy is available. The sensor node uses the Holt-Winters energy prediction algorithm in every time slot to obtain the virtual energy. For generating a reference, we exposed Node D to a constant high energy profile to fix E_3. Fig.4(I) shows the behaviour of the morphed application on sensor Node D. The plot shows the comparison for two cases: *Case-1* is the reference (E_3) energy level and *Case-2* when the node is subjected to the diffused energy profile. In *Case-1*, the source and the neighbour's energy was always high at E_3 and thus Node D could perform all the policies associated with the energy level E_3. This can be considered as the best case and serve as a reference for the distributed smart application. In *Case-2*, the source is associated with diffused energy profile and the neighbour node's energy profile is reflected. Fig.4(I) shows the results of application morphing under energy fluctuation. We captured these fluctuations as events and enumerated them as: (a) when the energy of node is low and predicted energy for future time slots is high and vice-versa; (b) when energy level of the node is high but the network's energy level is low and vice-versa. In Fig.4(I), several regions namely p, q, r, s, t show energy level transitions as well as application morphing. The region 'p' indicates the change in energy level from E_3 to E_2 on the source node. The region 'q' indicates the morphing of the application in one time slot before the change in energy level on the source node. The application morphed to a slower version of the base application as shown in region 'q' where the number of operations performed by the source node was significantly less compared to the previous time slot. This morphing improves the available energy on the energy buffer. The region 'r' shows the low energy profile E_1 on the source node and finally region 's' shows the application morphing to a slower version of the base application. This application morphing event is due to source node energy level being low and future time slot energy level is high. However, in the next time slot it can be clearly seen that Node D morphed to a faster version towards the base application as the future energy level is E_3 and region 't' indicates the increase in number of operations performed by the sensor Node D. Fig.4(II) shows the implementation result of application morphing on the source node for various energy profiles obtained

from LRSS. The neighbouring node was running reflected energy profile. As expected, it can be clearly seen that application morphing was enabled even when the source node was running on direct energy profile. In summary, the upper bound on application performance is dependant on source node's own energy as well as network's energy. Application performance when source node runs other energy profiles is also shown. Further, when the source node has diffused energy profile and neighbour has reflected profile, the source performed 354 node operations and 172 network operations. The source node performed 475, 656 node operations and 190, 212 network operations for reflected and directed energy profiles respectively.

7 Conclusions and Future Work

We have implemented a multihop EHWSN using solar energy. The efficacy of the proposed optimization algorithm was studied by evaluating node and network operations for various energy profiles. Clearly, the predicted, network and available energy contribute to the working of the distributed application. The smart application was able to maximize its operations by adjusting its application policy to satisfy the least residual energy criteria. In this work the number of hops is limited and we have only showed possible way of building application morphing also considering available energy in the nodes of the network rather than source node alone. We are encouraged by the results and propose to extend the network to include more number of hops and entire routes to study the scalability of our scheme. To the best of our knowledge this is the first implementation of a multihop energy harvested WSN, *albeit* two hops. We believe that our results are a step forward towards larger EHWSN deployments. We plan to generalize this work by proposing a cognitive networking stack for EHWSNs. The idea here is to provide hooks that consider EH including predictions at every layer of the networking stack. Moreover, we propose to study the distributed algorithms also with handles such as dynamically varying available voltage and possible frequency scaling.

Acknowledgements. We thank iCore project. This article describes work partially undertaken in the context of the iCore project, Internet Connected Objects for Reconfigurable Ecosystems (http://www.iot-icore.eu/). iCore is an EU Integrated Project funded within the European 7th Framework Programme, contract number: 287708. The contents of this publication are the sole responsibility of iCore project and can in no way be taken to reflect the views of the European Union.

References

1. ZigBee Green, `http://www.zigbee.org/Standards/Overview.aspx`
2. Chalasani, S., Conrad, J.M.: A survey of energy harvesting sources for embedded systems. In: IEEE Southeastcon 2008 (April 2008)

3. Jiang, X., Polastre, J., Culler, D.: Perpetual environmentally powered sensor networks. In: Fourth International Symposium on Information Processing in Sensor Networks (2005)
4. Kansal, A., et al.: Power Management in Energy Harvesting Sensor Networks. ACM Transactions on Embedded Computing Systems (2007)
5. Audet, D., de Oliveira, L.C., MacMillan, N., Marinakis, D., Wu, K.: Scheduling recurring tasks in energy harvesting sensors. In: IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS (2011)
6. Khodaian, A.M., Khalaj, B.H.: Delay-constrained utility maximisation in multihop random access networks. IET Communications 4(16), 1908–1918 (2010)
7. Palomar, D.P., Chiang, M.: A tutorial on decomposition methods for network utility maximization. IEEE Journal on Selected Areas in Communications 24(8), 1439–1451 (2006)
8. Moser, C., Thiele, L., Brunelli, D., Benini, L.: Adaptive Power Management for Environmentally Powered Systems. IEEE Transactions on Computers 59(4), 478–491 (2010)
9. Moser, C., et al.: Lazy scheduling for energy harvested sensor nodes. In: Conference on Distributed and Parallel Embedded Systems, DIPES 2006 (2006)
10. Hsu, J., Zahedi, S., Kansal, A., Srivastava, M., Raghunathan, V.: Adaptive Duty Cycling for Energy Harvesting Systems. In: Proceedings of the 2006 International Symposium on Low Power Electronics and Design (2006)
11. Vigorito, C.M., Ganesan, D., Barto, A.G.: Adaptive Control of Duty Cycling in Energy-Harvesting Wireless Sensor Networks. In: IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (2007)
12. Hasenfratz, et al.: Analysis, Comparison, and Optimization of Routing Protocols for Energy Harvesting Wireless Sensor Networks. In: Sensor Networks, Ubiquitous, and Trustworthy Computing, SUTC (2010)
13. Lowry Range Solar Station, Colarado State Land Board, `http://www.nrel.gov/midc/lrss`
14. Bazaraa, M.S., Jarvis, J.J., Sherali, H.D.: Linear Programming and Network Flows, 2nd edn. John Wiley & Sons Inc. (2008)