Statistical Energy Neutrality in IoT Hybrid Energy-Harvesting Networks

Soledad Escolar[1], Antonio Caruso[2], Stefano Chessa[3], Xavier del Toro[4], Félix J. Villanueva[1], Juan C. López[1]
[1]*School of Computing Science, University of Castilla-La Mancha, Ciudad Real, Spain*
[2]*Dept. of Mathematics and Physics "Ennio de Giorgi", University of Salento, Lecce, Italy*
[3]*Computer Science Department, University of Pisa, Pisa, Italy*
[4]*Energy Research and Industrial Applications Institute, University of Castilla-La Mancha, Ciudad Real, Spain*

***Abstract*—Scheduling tasks appropriately in an IoT device powered by multiple energy-harvesting sources is a challenging problem. In this paper, we model this problem, and we present a scheduling algorithm that optimally sets the overall node power consumption based on the utility, and on the energy required by tasks. The algorithm schedules high-level tasks, it uses the weather forecast informations available at the beginning of each scheduling period (typically a day), and the level of the battery, to define an optimal schedule. The main goal is to find a schedule that is energy neutral on average, over a period longer than the single scheduling window, for example a week. We test our scheduler on a simulated platform with the same specs of an Arduino node, equipped with a small (portable) solar panel, and attached to a small wind turbine. We see from the simulations that the scheduler performs as expected and that the utility of the scheduling improves as the error between the expected forecast and the real harvested energy is reduced.**

***Index Terms*—Hybrid Energy Harvesting Sensor Networks; Scheduling; Dynamic Programming.**

I. Introduction

Energy harvesting [1] is an effective method to provide a virtually unlimited lifetime to environmental IoT devices that remain unattended for long periods. The usual approach is to provide a device with a single energy harvesting technology, often related to a source that is *predictable* (even if often uncontrollable), so that it is possible to forecast the amount of energy the harvester can provide in order to appropriately design the system. One source often used is the Sun, whose cycles of irradiance can be predicted even in cloudy days by using weather forecast [2], and whose irradiance can be transformed into electricity by solar panels to charge the battery of the device. Here designing the system means choosing a battery capacity and the type and size of the solar panel so that they can match the consumption of the device and thus make the battery depletion an unlikely event. Over the recent years, the technologies for energy harvesting are improving and making accessible more and more sources like wind, rf-signals, vibrations, etc. [1], and researchers are now addressing devices equipped with multiple energy harvesters [2]. This work follows this research trend. Specifically, we consider devices equipped with several energy harvesters that extract energy from different sources found in the natural environment (for instance from the Sun and from the wind) and that contribute simultaneously to charge the battery of the device. On the other hand, since the different sources are independent and uncontrollable, there is still the chance that their energy production interrupts (for example harvesters of sun and wind will not provide any charge in a night without wind). For this reason, a device with energy harvesters usually adopt strategies aimed at dynamically modulating its power consumption [3], [4], [5], [6], [7], [8] in order to be *energy neutral* (i.e. its energy budget between its energy production and consumption is greater or equal to zero in a reference period, so that it never stops working). In particular, some approaches [9], [10], [11] modulate the energy consumption of the device by choosing between alternative tasks, each with different power consumption and utility, by choosing a scheduling of the tasks over that optimizes the overall utility while keeping the device energy neutral.

In this paper we reconsider the latter approach in the case of devices equipped with multiple energy harvesters and we propose a dynamic programming algorithm that finds the scheduling of the tasks with optimum utility, under a constraint of statistical energy neutrality, which is a relaxation of the energy neutrality concept. In the next Section we review works related to this, in Section III we present how we model tasks, their parameters and the model for the harvesters, in Section IV we present the scheduling algorithm, finally in Section V we present the results of simulations for the case of an Arduino connected to a solar panel and a wind turbine, and conclude with some comments in Section VI.

II. Related Work

The notion of *energy neutral scheduling* introduced in Kansal et al. [6] is of fundamental importance, for all the research work in this area. Kansal shows how to properly design the battery size as a function of the energy harvested by the node. The proposed methodology is at the base of our system model and scheduling problem. However, they only consider the optimization of a single power-management policies, i.e. the duration of the duty cycle of the node, while in this paper we prefer to optimize a high-level *utility* parameter, that is much simple to manage from the user level (or API level), like a *nice* level in POSIX systems, but representing a complex combination of several low-level power-management policies. In [12] the authors present an experimental testbed of a set of sensor nodes equipped with a solar panel and, similarly to this work, use the data from public weather forecast services to optimize the operation of the nodes. However, they use

an approach based on different machine learning strategies while we use the error between the expected seasonal average power output of the panel, the data from the weather prediction and the effective power output of the panel to optimize the scheduling of tasks, without at this stage using any form of machine learning. Other surveys related to *energy harvesting* are in [13] and in [14], [15], [1] and more recently in [16], [17], [18], [19]. Usually the scheduling algorithm does not use the instantaneous output of the harvesting source and the battery level but a *prediction scheme* [20], that gives some information (deterministic or stochastic) about the energy harvested in the future. All authors emphasize that good prediction schemes are a key ingredient for the robustness and quality of the scheduler. The duty cycle is not the only parameter optimized, for example in [21] a control-theoretic model selects the best *sensing rate* according to the estimated future production of the panel. A common tool to describe the scheduling problem is usually a mixed integer programming to model. Some papers use a dynamic programming [20] approach similar to us. Other papers deal with global, network-wide optimization problems like the integration of duty-cycling and routing [22]. In the next sections we present our model of energy production, then the task model that runs in an IoT node, and the scheduling problem.

III. System Model

We consider energy-harvesting devices equipped with several energy-harvesters, able to extract energy from different sources found in the natural environment, for instance from the Sun or from the wind, thus making use of unending energy sources, with the purpose of keeping the operation of the device. In this paper, we are considering multiple energy harvesters that contribute simultaneously to increase the overall energy production. On the other hand, since the energy that is extracted is hardly controllable, one or several of these productions could be interrupted for short periods of time, which makes still necessary to recharge the devices' batteries, with the purpose of sustaining the minimal operation of the device even in periods of scarce or null production.

The operation of the device can be defined in terms of tasks. Let us define a task as an instance of an application that comprises several activities like computing some function, sampling sensors, or sending messages to other devices. We denote as $\mathcal{T}$ to the set of n tasks belonging to one application, i.e. $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$. Each task t_i, $i \in [1, n]$ is associated with a utility u_i, which is defined as the utility gained by the device when t_i is executed, and a cost c_i which is defined as the charge expenditure due to the execution of t_i. In order to indefinitely prolong the execution of the application, we discretize a fixed finite time of window T (for example, a day) into k slots of duration $\tau = \frac{60 \times 24}{k}$ minutes, which we assume to be an integral fraction of an hour. The system scheduler should be periodically executed at the beginning of each time of window to find the best assignment of tasks to slots such that the maximal overall utility is achieved at the end of the reference time T.

Let us also characterize the battery of the device by a minimal $L_{\min}$ and a maximum $L_{\max}$ level of charge, where the first is the minimal level that could keep the node operating while the latter is the maximum capacity of the battery. We denote as L_i to the level of the battery at the beginning of slot i, with $i \in [1, k]$ and we denote with L_{k+1} to the battery level at the end of slot k. Ignoring for the moment the use of energy harvesters that could increase the battery level, the device's battery will be discharged at a rate that depends on the consumption of the task being executed, specifically the time to deplete the battery will be $\frac{L_{\max}}{I}$, where I is the integral of the current consumptions of the components involved in the execution of the scheduled task.

In our system, we consider the simultaneous usage of multiple energy-harvesters able to recharge the battery level of the device. To this purpose, let us define $\mathcal{S}$ as the set of m energy-harvester sources (with $m \geq 1$) attached to the device. In a given slot of time i, each energy-harvester source $s \in \mathcal{S}$ provides an amount of charge $\phi_i(s)$. Thus, we denote as Φ_i to the overall charge in slot i computed as the sum of the charges provided by each individual source:

$$\Phi_i = \sum_{s \in \mathcal{S}} \phi_i(s)$$

where Φ_i can be accumulated into the battery up to achieve the maximum capacity $L_{\max}$, which cannot be exceeded. For a practical use of our system, we trust in some service of forecasting of the energy that can be produced by each energy-harvester device, which could differ from the actual charge. We denote with $\psi_i(s)$ to the expected charge from the source $s \in \mathcal{S}$ in the slot i. Thus, the overall expected charge $\Psi_i(s)$ comes given by the sum of the contributions of all of them:

$$\Psi_i = \sum_{s \in \mathcal{S}} \psi_i(s)$$

Usually the weather forecast service has limited accuracy, but could provide information from which we can derive the expected production of each harvesting source with a bounded expected error between the production that is forecasted and the real one. Let us define with $\xi_i(s)$ to the relative error of the source s in a specific slot i with regard to the expected amount of charge, that is computed as:

$$\xi_i(s) = \frac{\phi_i(s) - \psi_i(s)}{\psi_i(s)}$$

An upper bound of $\xi_i(s)$ is $\varepsilon(s)$, which is the maximum relative error of the source s with high probability; then we denote with ε_i to an upper bound of the absolute error of Ψ_i and we compute it in the following way:

$$\varepsilon_i = \sum_{s \in S} \varepsilon(s) \Psi_i$$

The values of ε_i are upper bounds of the absolute error. As we will see in the next section, we use those values to set the level of the battery that we want to achieve at the end of the

scheduling period, or for a longer period of time, like an entire week.

IV. Scheduling problem and algorithm

The proposed model of energy harvesting, battery and consumption is used by the device to schedule the tasks in order to maximize the overall utility and, at the same time, to keep the battery at a level of charge that can keep the device operative for an indefinite time. The algorithm is executed by the device at the beginning of a time window and to assign the appropriate task to each slot. The size of the time window can be arbitrary, but, considering that solar panels are widely used and that they have a cycle of production of 24 hours, it is convenient to describe the algorithm referring to time windows of one day, and slot duration that are integral fraction of hours. We also assume that the device runs the scheduling algorithm every day at midnight, using weather forecast available for the next day, and starting at the beginning of slot 1.

The scheduling problem consists in finding an assignment of one task for each slot so that the sum of the utilities of all the tasks is maximum, subject to some constraints due to the battery. In particular, the battery level $L_i \geq L_{min}$ for all slots i in the day and the final battery level L_{k+1} should be greater or equal to a given Short-term Target Level STL. Let $A = [a_{i,j}]$ be the $k \times n$ matrix that represents the assignment of a task to a slot, so that $a_{i,j} = 1$ iff task t_j is assigned to slot i, and $a_{i,j} = 0$ otherwise.

The scheduling problem can thus be formulated as:

Problem 1 (*Max Utility Task Scheduling*)**.**

$$\text{maximize} \quad u = \sum_{i=1}^{k}\sum_{j=1}^{n} a_{i,j}u_j \tag{1}$$

$$\sum_{j=1}^{n} a_{i,j} = 1 \tag{2}$$

$$a_{i,j} \in \{0,1\} \forall i \in [1,k], j \in [1,n] \tag{3}$$

$$L_{i+1} = \min\Big\{ L_{\max}, L_i + \eta[(\Psi_i - \varepsilon_i) - \sum_{j=1}^{n} a_{i,j}c_j]^+ - [\sum_{j=1}^{n} a_{i,j}c_j - (\Psi_i - \varepsilon_i)]^+ \Big\} \forall i \in [1,k] \tag{4}$$

$$L_{\min} \leq L_i \forall i \in [1,k] \tag{5}$$

$$L_{k+1} \geq STL \tag{6}$$

where $[x]^+ = \max(x, 0)$

Where the objective function (1) is the sum of the utilities of the scheduled tasks, under the constraints (2-6). In particular, constraints (2) and (3) state that to a slot can be assigned exactly one task; constraint (4) relates the battery level at the beginning of slot $i+1$ to the battery level at the beginning of the previous slot, to the expected charge provided to the battery and to the consumption due to the execution of the task in the slot [6]; the value of $\eta \in [0,1]$ models the efficiency of the battery, while constraint (5) states that the battery level in each slot cannot be smaller than the minimum battery level L_{min} and, finally, constraint (6) states that the battery level at the end of the day should be at least the Short-term Target Level STL.

Note that, in this formulation we consider the pessimistic case in which the battery charge expected in each slot equals the minimum possible, i.e. it is $\eta(\Psi_i - \varepsilon_i)$ for each i, which is a lower bound to the actual battery charge Φ_i with high probability. This guarantees that the system will meet the constraints even with the actual battery charge.

The formulation of problem is similar to the problem in [23], which is known to be NP-complete, for which we propose an algorithm based on dynamic programming. In the following, if n is the number of tasks in the system, and STL is the short term level, we consider the problem of statistical optimal scheduling, energy neutral with respect to STL, i.e. the residual energy level in the battery, at the end of the period, must be greater than STL, and define the system *state* as a pair of integer (i, l), we associate to each pair a subproblem with value $opt(i, l)$ that considers the slots from i to K and with initial battery level l. Note that the solution of the complete problem is given by $opt(1, L_0)$, this function is defined in the following Lemma:

Lemma 1. *The optimal solution of Problem 1 is given by* $Q^* = opt(1, L_1)$ *with opt defines as the following backward recursive Bellman's equations:*

$$opt(K,l) = \max_{t=1\ldots n} \{\ u_t \mid l - c_t + \eta * (\Psi_K - \varepsilon_K) \geq STL\ \} \tag{7}$$

$$opt(i,l) = \max_{t=1\ldots n} \{\ u_t + opt(i+1, g(l,t)) \mid g(l,t) \geq L_{\min}\ \}$$

$$\textit{with: } g(l,t) = \min\{L_{\max}, l - c_t + \eta * (\Psi_i - \varepsilon_i)\}$$

Proof. The base case is the subproblem with only one (last) slot to schedule. In this case, we select the best feasible tasks, if l is (by definition) the energy at the beginning of this slot, the overall energy available is $l + \eta * (\Psi_K - \varepsilon_K)$, with Ψ_K the forecasted total production and ε_K an estimated upper bound to the average error in the forecast. Since we are in the last slot and the entire schedule must be energy neutral, we must save at least STL. Hence we schedule the higher utility task t with $c_t \leq l - STL + \eta * (\Psi_K - \varepsilon_K)$.

The second definition gives the recursive form of the dynamic programming approach: We consider slot i with an initial level of battery l, let $g(l,t) = l + \eta * (\Psi_i - \varepsilon_i) - c_t$ be the level of the battery if we schedule task t; if $g(l,t) \geq L_{min}$ task t is feasible and the utility of the schedule with t assigned to slot i will be u_t plus the optimum schedule that we obtain (recursively) starting from slot $i+1$ with a battery level of $g(l,t)$. In the definition, we consider all possible feasible t and select the best one. So evaluating $opt(1, L_1)$ we obtain the optimal schedule. □

The above equations lead directly to the following pseudocode of a dynamic programming algorithm that finds in pseudo-polynomial time the optimal schedule.

One important aspect in the configuration of the algorithm is the value assigned to the Short-term Target Level STL. In conventional approaches, the objective is to keep the device energy neutral, i.e. the battery level at the end of the day

Algorithm 1: Dynamic Programming Procedure to optimally solve the Statistical Energy Neutral Scheduling Problem.

Data: Battery Levels, Estimated Solar Output, Estimated Error Averages
Result: Short Term Level Neutral Scheduling

```
function OptSchedule(Tasks,K,L_max,L_min,STL,ε)
begin
    opt ← schedule ← 0;
    for i ← K − 1 to 0 by −1 do
        Evaluate Ψ_i from the weather forecast data;
        for l ← 1 to L_max do
            opt[i][l] ← schedule[i][l] ← 0;
            u_max ← −100; id_max ← −1;
            for t ∈ Tasks = {1, ..., n} do
                if (i = K and
                    l − c_t + η ∗ (Ψ_i − ε_i) ≥ STL and
                    M[i][l] < u_t) then
                    opt[i][l] ← u_t;
                    schedule[i][l] ← t;
                else
                    lr ← min(l − c_t + η ∗ (Ψ_i − ε_i), L_max);
                    if lr ≥ L_min then
                        u ← opt[i + 1][lr];
                        if (u ≠ 0 and u + u_t > u_max) then
                            u_max ← u + u_t;
                            id_max ← t;
                        end
                    end
                end
            end
            opt[i][l] ← u_max;
            schedule[i][l] ← id_max;
        end
    end
end
```

should be greater or equal to the level that the battery had at the beginning. If we set STL following the same approach, i.e. $STL \geq L_1$, then taking the case of a week of operation, the scheduler is forced to provide an energy-neutral schedule for each day, i.e. the battery will be always over L_1, even if, the real goal of the system is to be energy neutral on the longer time window of one-week.

Moreover, since our scheduling algorithm assigns the tasks to the slots under the worst-case hypothesis that the actual battery charge will always be equal with the minimum possible, the result will be that, in presence of a charge $\Phi_i > \Psi_i - \varepsilon_i$ (which will happen with high probability), the possibility that the battery reach $L_{\max}$, wasting energy from the harvesters, without properly increasing the utility of the tasks.

The approach that we propose in this paper, is to set the value of STL taking into account the knowledge about the bounded error between the forecasted energy and the real production. Setting a long-term level $LTL = L_1$ and $STL = LTL - \sum_{i\in[1,k]} \varepsilon_i$ we produce a schedule that optimizes the utility of tasks and is on the average above or equal LTL anyway.

V. Evaluation

Given the abstract energy harvesting model defined in Section III, we show now how it can be instantiated in a specific case. We consider the Arduino UNO-class IoT device powered with two energy harvesting sources, namely a solar panel (s_1) and a wind turbine (s_2). The Arduino UNO is also attached to a battery, with minimum battery level $L_{\min} = 100mAh$ and maximum battery level $L_{\max} = 1000mAh$.

The production of the solar panel follows the model introduced in [24]. The model computes the hourly irradiance for every day in the year, in any geographic location on the Earth, assuming a day perfectly clear of clouds. On the other hand, this model can be generalized by considering different weather conditions concerning the density and height of the clouds.

Concerning the wind turbine, the amount of charge it can provide depends on the rotation speed of the rotor that is proportional to the wind speed. For this reason, the charge per hour can be obtained by a simple model leveraging on the forecast of the wind speed. We adopt the same model as in [2], in which the power generated by a wind turbine is proportional to the cube of the wind speed (provided the wind speed is within the operational range), with the following equation:

$$power = a \times (WindSpeed)^3 + b$$

where a and b are two constants depending on specific properties of the actual turbine in use.

We have evaluated our proposal by means of a simulator developed in Python. The objective of the simulation is to demonstrate the continuous operation along the time for an IoT device equipped with multiple energy-harvesters when the energy-neutrality condition is relaxed. For this purpose, our simulator determines the assignment of tasks to slots for a long-term execution of an application under different error levels, and then computes the average remaining battery level and the corresponding average utility provided at the end of the considered period. We are also interested in knowing how far is the final average battery budget from the optimal value given by LTL.

Since the solar energy production has a natural cycle of 24 hours, we consider a time frame of $T = 24$ hours, which we discretize in a number of $k = 24$ slots per day of duration $\tau =$ 1 hour. Then, we generate the expected charge due to s_1 and s_2 and the actual charge such that, the latter will have a maximum relative error (equal for both sources) as $\varepsilon \in \{5\%, 10\%, 15\%\}$

The energy production from a wind turbine s_2 attached to an Arduino device is computed by using the data provided by the forecast website Weather Underground [1], which provides a complete list of weather data for any location around the world, including the wind speed (in Km/h). We obtained the average, maximum, and minimum daily wind speed in Ciudad Real (Spain). Thus, for a specific day of the year, we defined an scenario where we assign to each slot $i \in [1, k]$ an energy production computed by using the average wind speed.

[1] Weather Underground: https://www.wunderground.com/.

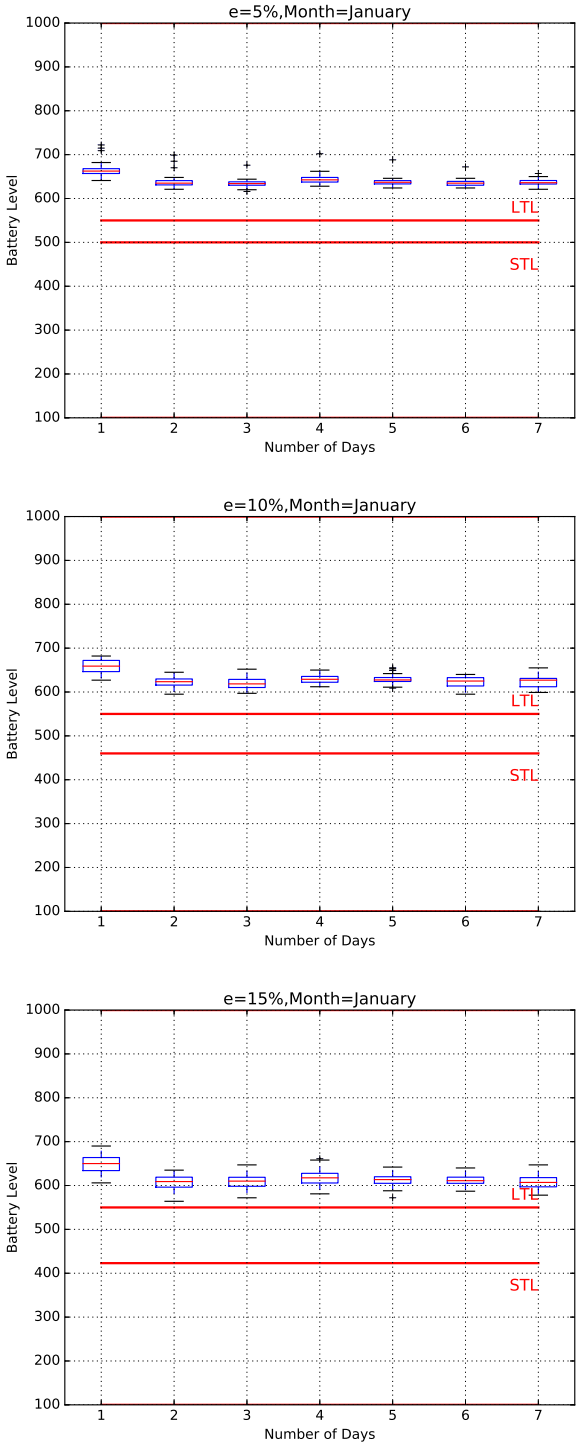

Fig. 1. Battery level at the end of each day with $\varepsilon = 0.05$ (above), with $\varepsilon = 0.1$ (center) and with $\varepsilon = 0.15$ (below)

Our simulator generates a large set of $A = 50$ applications, where each application is composed by a random number of tasks $n \in [6, 10]$. Each task is provided with a random utility $u_i \in [0, 100]$ (in percentage) and a cost c_i (in mAh), which is a function of a duty cycle $dc_i \in [0.01, 1]$ randomly generated; i.e. $c_i = dc_i \times I_{active} + (1 - dc_i) \times I_{idle}$, where I_{active} and I_{sleep} correspond to the average currents (in milliamperes) in active and idle state, respectively, of the platform where the task is executed (specifically for Arduino UNO, $I_{active} = 50$ and $I_{sleep} = 30$). The cost of the task per slot in mAh is then computed as $\frac{c_i}{24}$. Each application is provided with an idle task t_0, with a utility $u_0 = 1$ and a duty cycle $dc_0 = 0.01$, which guarantees the sustainable operation of the application under conditions of scarce production. Without loss of the generality, we assume that the higher utility, the higher cost, i.e. given two tasks $t_1 = \langle u_1, c_1 \rangle$ and $t_2 = \langle u_2, c_2 \rangle$, if $u_1 < u_2$ and $c_1 \geq c_2$, then t_1 is inefficient and it could be excluded a priori.

We are interested in knowing the battery levels and the average utility for A applications at the end of each day along a timeframe fixed to be 1 week, and for the different values of ε considered. Figure 1 shows three box plots corresponding to three different values of $\varepsilon = 0.05$ (above), $\varepsilon = 0.1$ (center) and $\varepsilon = 0.15$ (below). They show the distribution of the ending battery levels for each day of the week in the month of January (one of the months with most scarce solar production). The axis x of the figure shows the days and the axis y shows the battery levels in the range $[L_{\min}, L_{\max}]$; STL and LTL are also highlighted with horizontal red lines. As observed, the larger the ε, the larger the difference between STL and LTL. In each plot, the seven boxes represent the minimum and maximum value of the battery level each day of the week, as well as the quartiles Q1, Q2 (median) and Q3. The atypical values or outliers are represented with the symbol '+'. As observed, the larger the ε, the larger the distance between the minimum and maximum value. For small values of ε, the deviation between the forecasted and the real production is also small, and the values of STL and LTL are closer, since $STL = LTL - \sum_{i \in [1,k]} \varepsilon_i$ (note that if the last term is 0 we will have a perfect forecast). When ε increases, the deviation between the forecasted and the real production also increases, which means that the accuracy of the forecast is lower. Since we are using an upper bound on the error, the utility returned by the scheduler decreases with respect to the same experiment with lower errors. In the three boxplots, the ending battery values are larger than STL and LTL.

Finally, in Figure 2 we compare the average utility (in percentage) achieved after running the set of A applications each day of the week. As observed, ε impacts meaningfully on the overall utility achieved: the larger the ε, the lower the average utility. Specifically, utility ranges between $[80, 85]$ in the best case (with $\varepsilon = 5\%$), and between $[60, 75]$ in the worst case (with $\varepsilon = 15\%$).

VI. Conclusions and Future Works

In this paper we presented a scheduling algorithm that produces an optimal schedule of tasks to run in an IoT device

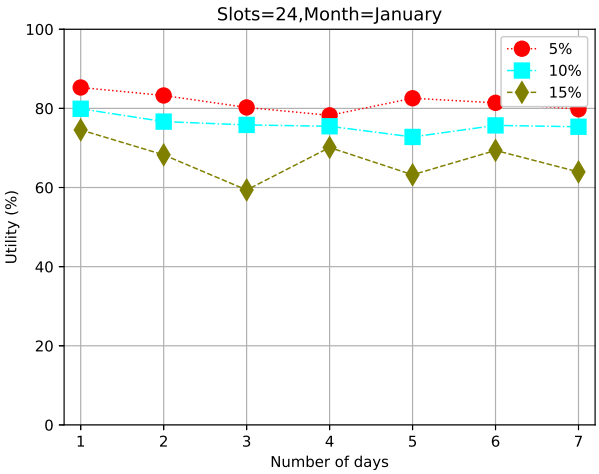

Fig. 2. Average utility obtained in the month of January, after the execution of A applications, for three different ε.

equipped with multiple energy-harvesting devices. The schedule is LTL-energy neutral on the long run on average, and STL-energy neutral after each day. We defined that value of STL as a function of the final LTL level and the total expected error in the forecasted production (based on weather forecast data). As a future work, we will investigate on learning algorithms to better capture the error between the forecasted and the real harvested energy, and the tradeoff concerning costs/lifetime due to different hardware configurations with solar panels with different capacities.

ACKNOWLEDGMENTS

This work has been funded by the Programme for Research and Innovation of University of Castilla-La Mancha, co-financed by the European Social Fund (Resolution of 25 Aug 2014) and by the Spanish Ministry of Economy and Competitiveness under projects REBECCA (TEC2014-58036-C4-1-R), PLATINO (TEC2017-86722-C4-4-R) and CitiSim Itea3 (TSI-102107-2016-4).

REFERENCES

[1] S. Sudevalayam and P. Kulkarni, "Energy Harvesting Sensor Nodes: Survey and Implications," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 443–461, 2011.

[2] N. Sharma, J. Gummeson, D. Irwin, and P. Shenoy, "Cloudy computing: Leveraging weather forecasts in energy harvesting sensor systems," in *In SECON '10: Proceedings of the 7th Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks*. IEEE, 2010, pp. 1–9.

[3] A. Kansal and M. B. Srivastava, "An Environmental Energy Harvesting Framework for Sensor Networks," in *Proceedings of the 2003 International Symposium on Low Power Electronics and Design, ISLPED'03*. IEEE, 2003, pp. 481–486.

[4] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. Srivastava, "Design Considerations for Solar Energy Harvesting Wireless Embedded Systems," in *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, April 2005, pp. 457–462.

[5] J. Hsu, S. Zahedi, A. Kansal, M. Srivastava, and V. Raghunathan, "Adaptive Duty Cycling for Energy Harvesting Systems," in *Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, ser. ISLPED '06. New York, NY, USA: ACM, 2006, pp. 180–185. [Online]. Available: http://doi.acm.org/10.1145/1165573.1165616

[6] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power Management in Energy Harvesting Sensor Networks," *ACM Trans. Embed. Comput. Syst.*, vol. 6, no. 4, Sep. 2007. [Online]. Available: http://doi.acm.org/10.1145/1274858.1274870

[7] A. Kansal, J. Hsu, M. Srivastava, and V. Raghunathan, "Harvesting Aware Power Management for Sensor Networks," in *Proceedings of the 43rd Annual Design Automation Conference*, ser. DAC '06. New York, NY, USA: ACM, 2006, pp. 651–656. [Online]. Available: http://doi.acm.org/10.1145/1146909.1147075

[8] G. Amato, A. Caruso, and S. Chessa, "Application-driven, energy-efficient communication in wireless sensor networks," *Computer Communications*, vol. 32, no. 5, pp. 896–906, 2009.

[9] Escolar, Soledad and Chessa, Stefano and Carretero, Jesus, "Energy-neutral networked wireless sensors," *Simulation Modelling Practice and Theory*, vol. 43, pp. 1–15, 2014.

[10] S. Escolar, S. Chessa, and J. Carretero, "Optimization of quality of service in wireless sensor networks powered by solar cells," in *2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications*, July 2012, pp. 269–276.

[11] Escolar, Soledad and Chessa, Stefano and Carretero, Jesus, "Energy management of networked, solar cells powered, wireless sensors," in *Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems*. ACM, 2013, pp. 263–266.

[12] F. A. Kraemer, D. Ammar, A. E. Braten, N. Tamkittikhun, and D. Palma, "Solar Energy Prediction for Constrained IoT Nodes Based on Public Weather Forecasts," in *The 7th International Conference on the Internet of Things (IoT 2017)*. Linz, Austria: ACM, October 2017.

[13] S. Basagni, M. Y. Naderi, C. Petrioli, and D. Spenza, *Wireless Sensor Networks with Energy Harvesting*. John Wiley & Sons, Inc., 2013, pp. 701–736. [Online]. Available: http://dx.doi.org/10.1002/9781118511305.ch20

[14] J. M. Gilbert and F. Balouchi, "Comparison of Energy Harvesting Systems for Wireless Sensor Networks," *international journal of automation and computing*, vol. 5, no. 4, pp. 334–347, 2008.

[15] M. K. Stojčev, M. R. Kosanović, and L. R. Golubović, "Power Management and Energy Harvesting Techniques for Wireless Sensor Nodes," in *Telecommunication in Modern Satellite, Cable, and Broadcasting Services, 2009. TELSIKS'09. 9th International Conference on*. IEEE, 2009, pp. 65–72.

[16] H. Jayakumar, K. Lee, W. S. Lee, A. Raha, Y. Kim, and V. Raghunathan, "Powering the internet of things," in *Proceedings of the 2014 international symposium on Low power electronics and design*. ACM, 2014, pp. 375–380.

[17] J. A. Khan, H. K. Qureshi, and A. Iqbal, "Energy management in Wireless Sensor Networks: A survey," *Computers & Electrical Engineering*, vol. 41, pp. 159–176, Jan. 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0045790614001773

[18] A. A. Babayo, M. H. Anisi, and I. Ali, "A Review on energy management schemes in energy harvesting wireless sensor networks," *Renewable and Sustainable Energy Reviews*, vol. 76, pp. 1176–1184, sep 2017. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1364032117304598

[19] G. V. Merrett and B. M. Al-Hashimi, "Energy-driven computing: Rethinking the design of energy harvesting systems," in *2017 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2017, pp. 960–965.

[20] D. Zhang, Y. Liu, J. Li, C. J. Xue, X. Li, Y. Wang, and H. Yang, "Solar power prediction assisted intra-task scheduling for nonvolatile sensor nodes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 5, pp. 724–737, 2016.

[21] C. Moser, D. Brunelli, L. Thiele, and L. Benini, "Real-time Scheduling for Energy Harvesting Sensor Nodes," *Real-Time Systems*, vol. 37, no. 3, pp. 233–260, 2007.

[22] G. Han, Y. Dong, H. Guo, L. Shu, and D. Wu, "Cross-layer optimized routing in wireless sensor networks with duty cycle and energy harvesting," *Wireless communications and mobile computing*, vol. 15, no. 16, pp. 1957–1981, 2015.

[23] S. Escolar, S. Chessa, and J. Carretero, "Energy management in solar cells powered wireless sensor networks for quality of service optimization," *Personal and Ubiquitous Computing*, vol. 18, no. 2, pp. 449–464, 2014.

[24] Escolar, Soledad and Chessa, Stefano and Carretero, Jesús, "Quality of service optimization in solar cells-based energy harvesting wireless sensor networks," *Energy Efficiency*, pp. 1–27, 2016. [Online]. Available: http://dx.doi.org/10.1007/s12053-016-9458-3