GreenCastalia: An Energy-Harvesting-Enabled Framework for the Castalia Simulator

David Benedetti, Chiara Petrioli, Dora Spenza
Computer Science Department
University of Rome "La Sapienza"
Via Salaria 113, 00198, Rome, Italy
{benedetti,petrioli,spenza}@di.uniroma1.it

ABSTRACT

The emergence of energy-scavenging techniques for powering networks of embedded devices is raising the need for dedicated simulation frameworks that can support researchers and developers in the design and performance evaluation of harvesting-aware protocols and algorithms. In this work we present *GreenCastalia*, an open-source energy-harvesting simulation framework we have developed for the popular Castalia simulator. GreenCastalia supports multi-source and multi-storage energy harvesting architectures, it is highly modular and easily customizable. In addition, it allows to simulate networks of embedded devices with heterogeneous harvesting capabilities.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network Architecture and Design—*Wireless communication*

General Terms

Design, Measurement, Performance

Keywords

Energy harvesting, Castalia, Network simulations, Wireless sensor networks

1. INTRODUCTION

Energy harvesting is rapidly affirming as a promising solution towards the goal of energy-autonomous wireless sensor networks (WSNs). The practical feasibility of applying energy-scavenging techniques to networks of embedded systems has been assessed by many works, proposing and validating prototype implementations of devices powered by solar light [28, 3], wind [24, 6], kinetic energy [22], thermal energy [29], RF energy [7], and so on. Such environmentally-powered systems have the potential for unlimited lifetime, but this ambitious goal is challenged by the unpredictable nature of environmental energy sources, which requires dedicated solutions for energy management [1]. In fact, energy-harvesting nodes usually experience an alternation between periods in which energy must be sparely used and others in which there may even be an excess of energy available. Since traditional WSN solutions are not meant to cope with such situations, existing protocols and algorithms must be adapted or re-designed to obtain systems able to adapt their workload scheduling to the stochastic nature of environmental energy sources.

ENSSys'13, November 13 2013, Roma, Italy

http://dx.doi.org/10.1145/2534208.2534215

Due to the relatively hight cost and long timescales needed to deploy and maintain large-scale wireless sensor networks, simulation is traditionally the tool of choice for WSN research. This is especially true for energy-harvesting wireless sensor networks (EH-WSN), as real-life validation of power-scavenging systems is further challenged by time and location dependent harvesting conditions, which largely compromise the repeatability of experiments. Moreover, running real experiments on EH-WSNs is time consuming, as assessing the robustness of a solution requires testing it under a variety of harvesting conditions. Simulations mitigate these issues by providing a cost-effective method to thoroughly test novel algorithms and schemes in a controlled and reproducible environment. Recognizing this need, many recent works have proposed dedicated harvesting-enabled frameworks and modeling tools [17, 31, 13, 9]. However, to the best of our knowledge, none of them is currently available to the wider research community, resulting in a lack of accessible simulation tools to support the early-phase design and testing of harvesting-aware algorithms and protocols.

In an attempt to fill this gap, in this work we describe the design and implementation of *GreenCastalia* [12], an open-source energy-harvesting framework for the popular Castalia simulator [2]. Castalia has recently gained wide acceptance in the WSN research community, as it features one of the most advanced channel and radio model among existing WSN simulators [25]. Castalia also provides a highly flexible model of sensor devices that includes noise and bias, which can be used to define accurate models of physical processes. Despite many desirable features, the current version of Castalia does not support simulation of energy-harvesting systems, and it only provides an ideal battery model. Our proposed solution, GreenCastalia, integrates into the Castalia simulator, extending it with a flexible framework to simulate networks of embedded devices with heterogeneous harvesting and energy storage capabilities. Moreover, it supports multi-source harvesting architectures that are becoming increasingly popular to enhance the overall efficiency and reliability of energy-scavenging system [32].

The rest of the paper is organized as follows. In Section 2 we survey related work on the area, focusing on simulation tools that support energy-harvesting systems. In Section 3 the general structure of GreenCastalia is presented, including a detailed description of each module of the framework. A high-level example of how GreenCastalia can be used to implement harvesting-aware protocols is given in Section 4. Finally, Section 5 concludes the paper.

2. RELATED WORK

Due to the difficulty of deploying real WSN systems, a considerable number of network simulators, both general-purposes and WSN-specific, have been employed and developed in the past years to support research on wireless sensor networks. Popular frameworks include ns-2 [20], ns-3 [21], OMNeT++ [23], and Castalia [2]. These tools, however, offer limited support for energy-aware simulations [17], as they do not natively provide models of energy sources and storages. More recently, the rapid emergence of energy-harvesting techniques for networks of embedded devices has fostered the development of simulation frameworks that allow to model networks of power-scavenging motes. One of the earlier works in the area is WSNsim, an energy-aware simulator proposed by Merrett et al. in [17]. WSNsim offers a number of commendable features, including accurate models of the node physical hardware and of the environment, and the integration of a structured architecture for embedded software. Unfortunately, WSNsim is not currently available to the wider research community. In [34], Wu et al. propose an energy-aware extension of the ns3 network simulator that allows to model the energy consumption of wireless devices, as well as linear and non-linear energy storages. Their implementation is included in ns-3.9 and publicly available, but it does not currently support energy scavenging. An extension for ns-3 for wireless sensor nodes powered by solar energy harvesting and super-capacitors is presented by Sánchez et al. in [31]. Simulations show the model has good accuracy, but different power sources, multi-source energy harvesting and multiple storage devices are not supported. Didioui et al. propose in [9] a tool for co-simulation of solar-powered systems that uses WSNet [33] and Matlab. WSNet simulates the protocol stack, and communicates via TCP sockets with Matlab, which simulates the energy subsystem of the nodes. Using co-simulation allows to accurately model non-linear batteries, but the need for communication and synchronization among simulators typically introduces significant overhead. System-level simulators that focus on low-level details of the design of power supply architectures have also been proposed [13], but, although very accurate, they may be too slow for effective design exploration of harvesting-aware network protocols. A first attempt to integrate an energy-harvesting framework with Castalia is presented by De Mil et al. in [8]. Such an implementation, however, has several shortcomings and limited flexibility. For example, the harvested current is discretized into a limited number of possible values, and there is no support for multiple harvesting and storage devices.

3. GREENCASTALIA

Energy management in Castalia is entirely carried on by the *ResourceManager* module, which holds energy-specific parameters, such as the baseline power consumption of the mote and its initial energy budget. This module keeps track of the remaining energy by performing periodic updates of the energy spent by the node over time. Energy updates are also triggered on-demand by Castalia modules that model hardware components whenever their power consumption changes. The amount of remaining energy is computed by the *ResourceManager* by modeling an ideal primary battery that is linearly discharged.

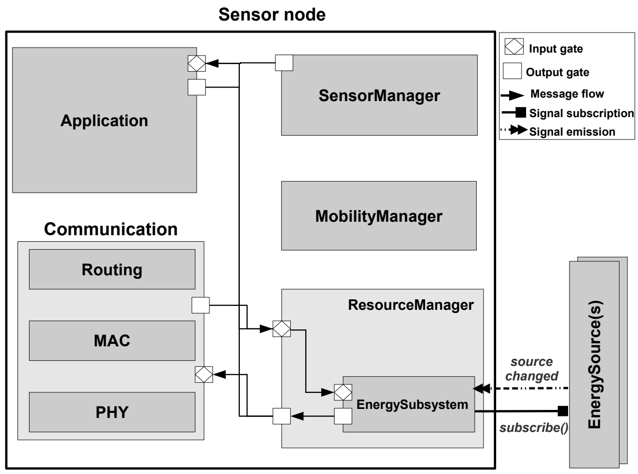


Figure 1: General structure of the SensorNode module in GreenCastalia.

We extended the energy model of Castalia by introducing a new compound module, called *EnergySubsystem*, that completely replaces the *ResourceManager* module for what concerns energy management. To allow easy integration with Castalia, we made the *ResourceManager* a compound module including the new *EnergySubsystem* module (Fig. 1). The *EnergySubsystem* is composed by three main submodules:

- *EnergyHarvester*, which models the energy harvesting process and handles the corresponding devices (Section 3.1);
- *EnergyStorage*, which represent an energy storage device, i.e., a supercapacitor, a rechargeable battery or a disposable battery (Section 3.2);
- *EnergyManager*, which implements the control logic for storage utilization and charging (Section 3.3).

Environmental energy sources are modeled by the external *EnergySource* module (Fig. 2), of which multiple instances can be created to simulate multi-source harvesting systems. The current implementation of GreenCastalia provides a generic *TraceEnergySource* module, which allows to feed the simulator with timestamped power traces collected through real-life deployments and measurement studies [5, 11], or with energy availability traces obtained by data repositories [10] or meteorological stations [19]. The *EnergySource* module can also be extended to support user-defined models of different energy sources. In GreenCastalia, each harvester

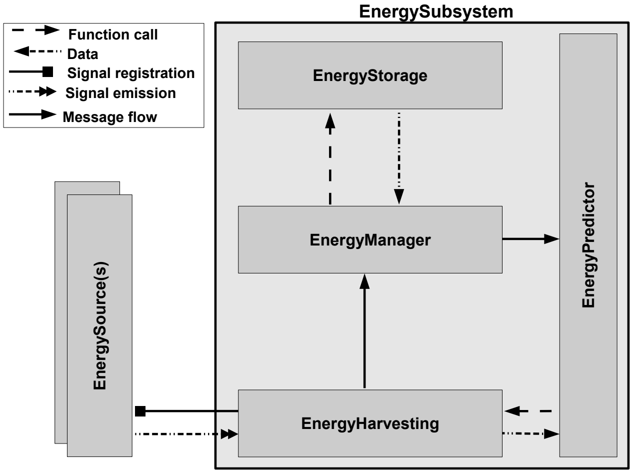


Figure 2: Architecture of the EnergySubsystem module.

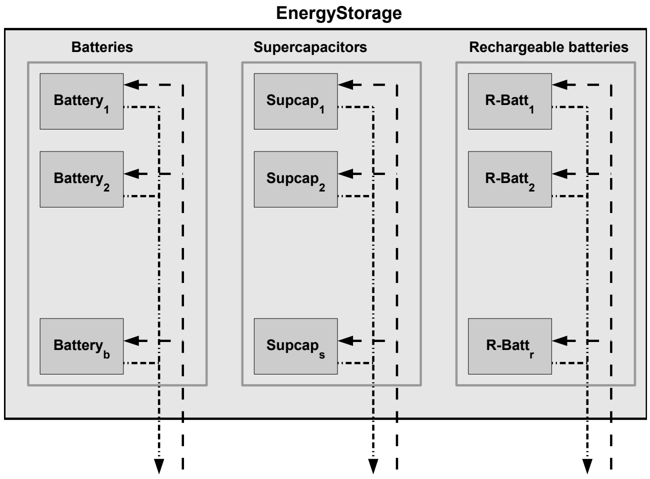


Figure 3: *EnergyStorages* module in GreenCastalia including a combinations of disposable batteries, supercapacitors and rechargeable batteries. Inward and outward arrows represent energy intake and provisioning.

is logically connected to one and only one energy source, while multiple harvesters can scavenge power from the same source. This allows to handle heterogeneous EH-WSNs, in which each node may have different harvesting capabilities.

In addition, GreenCastalia provides support for energy predictions through the *EnergyPredictior* module. Further details are given in Section 3.4.

3.1 Energy harvesters

The *EnergyHarvester* module represents a physical harvesting device connected to a node. The current implementation of GreenCastalia provides a default *EnergyHarvester* module that scavenges power with a given efficiency from the energy source it is connected to. The interface for this module allows to optionally indicate a timestamped file that specifies how the harvesting efficiency of the device varies over time. This allows to model time-varying effects such as moving shadows, temporarily obstructions, changing in harvester orientation, and decreasing efficiency due to dust or aging. Such default module also provides a sample implementation of an *EnergyHarvester*, making the integration of alternative harvesting device models relatively easy.

3.2 Energy storage

The *EnergyStorage* module represents a storage device that supplies energy to the node. The default interface for this module includes parameters such as charging and discharging efficiency, rated capacity, maximum voltage and cutoff voltage. As of today, GreenCastalia provides an implementation of two battery models and of a supercapacitor model:

- *IdealBattery*: a simple model in which the voltage of the battery remains constant over its lifetime and the discharge rate is always proportional to the power drawn from the battery. This is the default battery model provided by Castalia;

- *EmpiricalBattery*: a model similar to *IdealBattery*, but including support to simulate the voltage of the battery decreasing over time based on its depth of discharge. The estimated voltage of the battery is computed by using a piecewise linear approximation of its empirical discharge pattern:

$$V(t) = \begin{cases} a_1 \cdot C(t) + b_1, & C_{R_1} \leq C(t) < C_{R_2} \\ \vdots & \vdots \\ a_n \cdot C(t) + b_n, & C_{R_n} \leq C(t) < C_{R_{n+1}} \end{cases}$$

  where $C_{R_1}, \ldots, C_{R_{n+1}}$ are the residual energy values in which the slope of the discharge curve changes significantly and $a_1, \ldots, a_n$, $b_1, \ldots, b_n$ are constants representing the coefficients of the line segments used for the approximation. An example of such approach is shown in Figure 4.

- *Supercapacitor*: a simple model in which the voltage of the supercapacitor is estimated based on the formula: $E(t) = \frac{1}{2}C(V(t))^2$, where $E(t)$ is the energy stored by the supercapacitor at time t, C is its rated capacity and $V(t)$ is the voltage across it.

As for *RechargeableBattery*, a simple model accounting for remaining cycles is provided, while more accurate models are under investigation.

The *EnergyStorage* modules supply energy to the sensor node through the `discharge()` function called by the *EnergyManager* module. Supercapacitors and rechargeable batteries also implement the `charge()` function that is called by the *EnergyManager* module whenever there is an excess of harvesting energy available. Each storage device can also implement a `selfDischarge()` function, which models the leakage and self-discharge effects suffered by charged supercapacitor and batteries. GreenCastalia currently provides three basic supercapacitor leakage models that can be used in simulations, serving as a starting point for future extension:

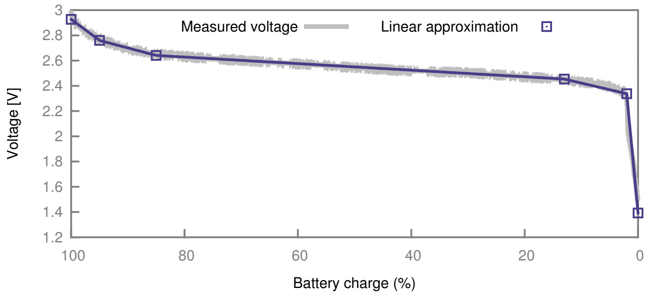


Figure 4: Piecewise linear approximation of battery empirical discharge pattern.

- *Constant current*: The leakage experienced by a charged supercapacitor is modeled as a constant current [15];
- *Exponential*: The leakage experienced by a charged supercapacitor is modeled as an exponential function of the current supercapacitor voltage [30]:

$$P_{leak} \approx P_0 \exp(\alpha V(t)),$$

  where $V_(t)$ is the voltage of the supercapacitor at time t and P_0 and α depends on the particular supercapacitor considered;
- *Piecewise linear approximation*: The leakage, $leak_i(t)$ experienced by the sensor node N_i at time t is modeled by using a piecewise linear approximation of the empirical leakage pattern [35].

More accurate models, such as those recently proposed in [14, 16], can be integrated in GreenCastalia by extending the *Supercapacitor* module and its corresponding class.

3.3 Energy manager

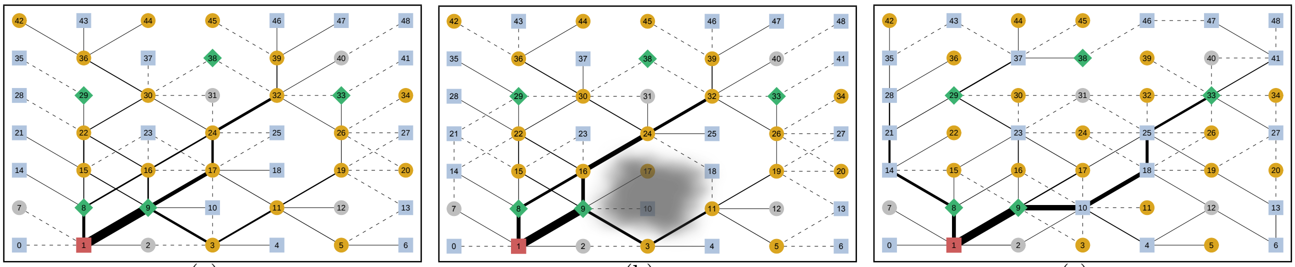
The *EnergyManager* module is the core of the energy subsystem, having a complete view of the power harvested and drawn over time. It implements the control logic for storage devices utilization and charging, simulating the energy flow from harvesters and storage devices to the load, and from harvesters to storage devices. Moreover, it also keeps track of the energy wasted due to storage devices non-idealities, such as charging and discharging efficiency, self-discharge, and limited capacity, which may cause excess energy from harvesting to be lost. The default interface for the *EnergyManager* allows to define parameters such as the baseline power consumption of the node, the cutoff voltage, and the frequency of power updates. Energy updates are performed through the `energyUpdate` function in response to power consumption changes, which are triggered by modules modeling hardware components when their state changes. Updates are also triggered by *EnergyHarvesters*, which asynchronously notify the *EnergyManager* module of variations in their harvesting power, which can occur because of the dynamics of the energy source or due to temporary variations in the surrounding environment (e.g., moving shadows that impact on the amount of power generated by solar cells). In addition, the *EnergyManager* performs periodic updates with the period specified by the `periodicEnergyCalculation Interval` parameter. When the `energyUpdate` function is called, the *EnergyManager* module computes the net power consumption of the node, taking into account its current power consumption and harvesting rate. If the net power consumption is negative, the excess energy is used to recharge storage devices. Otherwise, storage devices are discharged to supply energy to the node. Devices handled by the *EnergyManager* can include a combinations of supercapacitors, rechargeable batteries and disposable batteries (Fig. 3). By extending the *EnergyManager* module, the specific controller for storage devices utilization and charging can be modified so as to simulate a variety of architectures, including hybrid and multi-stage storage systems [18]. During energy updates, self-discharge of storage devices is also taken into account. Whenever the node runs out of energy, the *EnergyManager* notifies other modules by sending an `OUT_OF_ENERGY` message. If energy becomes available again, a `NODE_RESTART` message is sent to simulates a node reset.

3.4 Support for energy predictions

Power-scavenging systems need to deal with the dramatic changes of energy availability over time. In the case of predictable energy sources, such as solar light, energy prediction models are a precious tool to devise smart energy allocation strategies. In fact, by forecasting the expected energy intake in the near future, proactive power management strategies can be implemented to optimize the utilization of the available energy. Acknowledging the importance of energy predictions for the design of harvesting-aware protocols, we included in GreenCastalia an *EnergyPredictor* module. The interface of this module allows to define the energy source for which predictions are delivered, the number of timeslots per day and the frequency with which the energy source is sampled within each timeslot. The current implementation of GreenCastalia includes the widely used EWMA prediction algorithm [15] and two state-of-the-art predictors recently proposed in the literature, WCMA [27] and Pro-Energy [4]. EWMA maintains the history of the energy harvested on past days as an exponential moving average. Predictions are delivered based on the assumption that the energy available at a given time of the day is similar to the energy intake at the same time on the previous days. Similarly to EWMA, WCMA takes into account the average energy availability experienced in previous days, but it improves the prediction accuracy by additionally scaling this average value based on a weighting factor that indicates how much the weather conditions of the current day changed with respect to previous days. Pro-Energy stores and uses a pool of *harvested profiles* observed in the past, which represent the energy intake recorded during different types of "typical" days (e.g., sunny, cloudy, rainy, etc.). When delivering energy predictions, Pro-Energy looks at the stored profile that is the most similar to the current day and computes estimations based on a combination of the energy reported in the stored profile and of the energy observed during the current day.

4. SIMULATION EXAMPLE

In this section, we give a high-level example of how GreenCastalia can be used to implement harvesting-aware protocols. To this purpose we have implemented a novel routing protocol for convergecasting in EH-WSNs that distributes traffic in the network favoring nodes that, thanks to harvesting, have high energy availability. Packet forwarding is realized by opportunistically selecting the next-hop relay through a timer-based contention. In our solution, a node



(a) (b) (c)

Figure 5: Example of routing paths selected by the harvesting-aware routing protocol: (a) data traffic routing at daytime; (b) effect of localized shadow on routes selection and (c) data traffic at nighttime.

that has packets to send starts the relay selection process by broadcasting a Request-to-Send (RTS) packet that contains information about its hop count and the number of packets it intends to forward. Neighbors receiving the RTS packet participate in the contention only if their hop count is not greater than that of the sender. Upon reception of an RTS packets, available neighbors respond with a Clear-to-Send (CTS) message after a random jitter, which is computed based on their harvesting condition, energy reservoir, and hop count. Higher priority is given to nodes that are experiencing an *energy peak*, i.e., having their storage(s) almost at capacity while they are harvesting energy. Such nodes are ideally able to perform forwarding at no cost, as excess energy gathered during an energy peak would be wasted if not immediately used. They thus respond to RTS messages with a lower jitter than the other nodes, which in turn compute their waiting time based on the fraction of energy they have available and on their current harvesting power rate. Among potential relays having similar energy status, hop count is used to select the best node that offers positive advancement toward the sink. The sender selects as next-hop relay the first node responding with a CTS to its RTS message. Once a relay is selected, the sender transmits its buffered packets in a burst, performing a back-to-back transmissions in which data packets are individually acknowledged [26].

Figure 5 shows an example of application of the proposed protocol in a network of heterogeneous wireless sensor nodes modeled in GreenCastalia. In the considered setup, 49 nodes are deployed in 7x7 grid over a field of 120 × 120 meters. Nodes run a data collection application that requires them to periodically collect sensing measurements and to deliver them to the sink (node 1 in Fig. 5), which is assumed to have unlimited energy supply. The network is composed by nodes equipped with solar panels, micro wind turbines, or both. Scavenging devices are represented in Fig. 5 by yellow circles, blue squares and green diamonds, respectively. Two *TraceEnergySource* modules are instantiated in GreenCastalia, which feed the simulator with timestamped power traces of solar and wind availability obtained based on data from the US National Renewable Energy Laboratory [19]. *EnergyHarvester* modules of each node are connected to the appropriate energy source, and, in case of solar energy harvesting, the efficiency of the devices is made variable to represent shadows. Finally, around 10% of the nodes in the network are traditional motes without energy harvesting capabilities (represented by gray circles).

Figure 5 shows a snapshot of the paths selected by the routing protocols at different times of the day. Thicker lines between nodes indicate heavier data traffic. Fig. 5(a) shows routes selected at daytime, during which solar-powered and multi-source nodes experience energy peaks. Such nodes are thus selected with higher priority as next hop relays, while data routing across wind-powered (e.g., nodes 4 and 10) and battery-powered (e.g., nodes 2 and 31) motes is scarce. Fig.5(b) shows the effect of a localized shadow on routes selection: Traffic is routed around node 17, as it is covered by a temporary shadow that greatly reduces its harvesting power. Finally, as shown by Fig.5(c), at nighttime wind-powered and multi-source nodes are selected with higher priority with respect to solar-harvesting and battery-powered nodes.

5. CONCLUSIONS

In this paper we presented *GreenCastalia*, an energy-harvesting framework for the Castalia simulator designed with modularity and ease of extensibility in mind. GreenCastalia allows to model and simulate networks of devices with heterogeneous harvesting capabilities, including multi-source and multi-storage energy harvesting architectures, answering the demand for open-source simulation tools to aid the early-phase design and testing of solutions for energy-harvesting sensor systems.

6. ACKNOWLEDGMENTS

Dora Spenza is a recipient of the Google Europe Fellowship in Wireless Networking, and the research presented in this paper is supported in part by this Google Fellowship. This research was also sponsored in part by the FP7 project GENESI (Green sEnsor NEtworks for Structural monItoring), Contract n. 257916.

7. REFERENCES

[1] S. Basagni, M. Y. Naderi, C. Petrioli, and D. Spenza. Wireless sensor networks with energy harvesting. In *Mobile Ad Hoc Networking: The Cutting Edge Directions*, chapter 20, pages 701–736. John Wiley and Sons, Inc., Hoboken, NJ, 2013.

[2] A. Boulis. Castalia: Revealing Pitfalls in Designing Distributed Algorithms in WSN. In *Proceedings of SenSys 2007*, pages 407–408, New York, NY, USA, 2007.

[3] D. Brunelli, C. Moser, L. Thiele, and L. Benini. Design of a solar-harvesting circuit for batteryless

embedded systems. *IEEE Transactions on Circuits and Systems*, 56(11):2519–2528, 2009.

[4] A. Cammarano, C. Petrioli, and D. Spenza. Pro-Energy: A novel energy prediction model for solar and wind energy-harvesting wireless sensor networks. In *Proceedings of IEEE MASS 2012*, Las Vegas, Nevada, October 2012.

[5] A. Cammarano, D. Spenza, and C. Petrioli. Energy-harvesting WSNs for structural health monitoring of underground train tunnels. In *Proceedings of IEEE INFOCOM 2013, Student Session*, pages 9–10, Torino, Italy, April, 14-19 2013.

[6] D. Carli, D. Brunelli, D. Bertozzi, and L. Benini. A high-efficiency wind-flow energy harvester using micro turbine. In *Proceedings of SPEEDAM 2010*, pages 778–783, 2010.

[7] L. Chen, S. Cool, H. Ba, W. Heinzelman, I. Demirkol, U. Muncuk, K. Chowdhury, and S. Basagni. Range extension of passive wake-up radio systems through energy harvesting. In *Proceedings of ICC 2013*, Budapest, Hungary, June 2013.

[8] P. De Mil, B. Jooris, L. Tytgat, R. Catteeuw, I. Moerman, P. Demeester, and A. Kamerman. Design and Implementation of a Generic Energy-Harvesting Framework Applied to the Evaluation of a Large-Scale Electronic Shelf-Labeling Wireless Sensor Network. *EURASIP J. Wirel. Commun. Netw.*, 2010:7:1–7:12, February 2010.

[9] A. Didioui, C. Bernier, D. Morche, and O. Sentieys. HarvWSNet: A co-simulation Framework for Energy Harvesting Wireless Sensor Networks. In *Proceedings of ICNC 2013*, pages 808–812, San Diego, USA, January 2013.

[10] EH Network Data Repository. http://eh-network.org/data.

[11] M. Gorlatova, M. Zapas, E. Xu, M. Bahlke, I. J. Kymissis, and G. Zussman. CRAWDAD Data Set Columbia/Enhants, April 2011. http://crawdad.cs.dartmouth.edu/columbia/enhants.

[12] GreenCastalia, 2013. http://senseslab.di.uniroma1.it/greencastalia.

[13] J. Jeong and D. Culler. A Practical Theory of Micro-Solar Power Sensor Networks. *ACM Transactions on Sensor Networking*, 9(1):9:1–9:36, November 2012.

[14] A. Kailas, M.-A. Ingram, and D. Brunelli. A Simple Energy Model for the Harvesting and Leakage in a Supercapacitor. In *Proceedings of IEEE ICC 2012*, pages 6278–6282, Ottawa, Canada, June 2012.

[15] A. Kansal, J. Hsu, S. Zahedi, and M. Srivastava. Power Management in Energy Harvesting Sensor Networks. *ACM Transactions in Embedded Computing Systems*, 6(4):Article 32, September 2007.

[16] G. Merrett and A. Weddell. Supercapacitor Leakage in Energy-Harvesting Sensor Nodes: Fact or Fiction? In *Proceedings of EnHaNSS 2012*, Antwerp, Belgium, June 2012.

[17] G. Merrett, N. White, N. Harris, and B. Al-Hashimi. Energy-Aware Simulation for Wireless Sensor Networks. In *Proceedings of SECON 2009*, pages 64–71, Piscataway, NJ, USA, June 2009.

[18] A. Nasiri, S. Zabalawi, and G. Mandic. Indoor Power Harvesting Using Photovoltaic Cells for Low-Power Applications. *IEEE Transactions on Industrial Electronics*, 56(11):4502–4509, 2009.

[19] NREL: Measurement and Instrumentation Data Center, 2011. http://www.nrel.gov/midc.

[20] The network simulator - NS2. http://www.isi.edu/nsnam/ns.

[21] The network simulator - NS3. http://www.nsnam.org.

[22] J. Olivo, D. Brunelli, and L. Benini. A kinetic energy harvester with fast start-up for wearable body-monitoring sensors. In *Proceedings of PervasiveHealth 2010*, pages 1–7, 2010.

[23] OMNeT++ homepage. http://omnetpp.org.

[24] C. Park and P. Chou. Ambimax: Autonomous energy harvesting platform for multi-supply wireless sensor nodes. In *Proceedings of IEEE SECON 2006*, volume 1, pages 168 –177, Reston, Virginia, USA, September 25-28, 2006.

[25] D. Pediaditakis, Y. Tselishchev, and A. Boulis. Performance and scalability evaluation of the castalia wireless sensor network simulator. In *Proceedings of SIMUTools 2010*, pages 53:1–53:6, 2010.

[26] C. Petrioli, M. Nati, P. Casari, M. Zorzi, and S. Basagni. ALBA-R: Load-Balancing Geographic Routing Around Connectivity Holes in Wireless Sensor Networks. *IEEE Transactions on Parallel and Distributed Systems*, 2013.

[27] J. Piorno, C. Bergonzini, D. Atienza, and T. Rosing. Prediction and Management in Energy Harvested Wireless Sensor Nodes. In *Proceedings of Wireless VITAE 2009*, pages 6–10, Aalborg, Denmark, May 2009.

[28] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. Srivastava. Design considerations for solar energy harvesting wireless embedded systems. In *Proceedings of IPSN 2005*, pages 457–462, Apr 15 2005.

[29] Y. Ramadass and A. Chandrakasan. A battery-less thermoelectric energy harvesting interface circuit with 35 mv startup voltage. *IEEE Journal of Solid-State Circuits*, 46(1):333–341, 2011.

[30] C. Renner, J. Jessen, and V. Turau. Lifetime Prediction for Supercapacitor-Powered Wireless Sensor Nodes. In *FGSN 2009*, pages 55–58, August 2009.

[31] A. Sánchez, S. Climent, S. Blanc, J. Capella, and I. Piqueras. WSN with Energy-Harvesting: Modeling and Simulation based on a Practical Architecture using Real Radiation Levels. In *Proceedings of PM2HW2N 2011*, pages 17–24, New York, NY, USA, 2011.

[32] A. S. Weddell, M. Magno, G. V. Merrett, D. Brunelli, B. M. Al-Hashimi, and L. Benini. A survey of multi-source energy harvesting systems. In *Proceedings of DATE 2013*, pages 905–908, 2013.

[33] WSNet Simulator Homepage. http://wsnet.gforge.inria.fr.

[34] H. Wu, S. Nabar, and R. Poovendran. An Energy Framework for the Network Simulator 3 (NS-3). In *Proceedings of SIMUTools 2011*, pages 222–230, 2011.

[35] T. Zhu, Z. Zhong, T. He, and Z.-L. Zhang. Energy-Synchronized Computing for Sustainable Sensor Networks. *Elsevier Ad Hoc Networks Journal*, 2010.