STOCHASTIC OPTIMIZATION FOR ENVIRONMENTALLY POWERED WSNS USING MDP MODELS WITH MULTI-EPOCH ACTIONS

Alexandru E. Şuşu
EPFL, Station 14, 1015 Lausanne, Switzerland

Keywords: Wireless Sensor Nodes, Energy Harvesting, Constrained Markov Decision Processes, Multi-epoch Actions.

Abstract: The controller of an environmentally powered wireless sensor node (WSN) seeks to maximize the quality of the data measurements and to communicate frequently with the network, while balancing the uncertain energy intake with the consumption. To devise such system manager we use the Markov Decision Process (MDP) optimization framework. However, our problem has physical characteristics that are not captured in the standard MDP model: namely, the radio interface takes a non-negligible amount of time to synchronize with the network before starting to transmit the acquired data, which translates into MDP actions spanning over multiple epochs. Optimizing without considering this multi-epoch actions requirement results in sub-optimal MDP policies, which, under certain conditions described in the paper, waste on average 50% of the radio activity. Therefore, we incorporate this new constraint in the MDP formulation, and obtain an optimal policy that performs on average 83% better than a standard MDP policy. This solution outperforms also some heuristic policies we use for comparison by 14% and 154%.

1 INTRODUCTION

Advances in microelectronic technology allow us to build low-cost and low-power miniaturized Wireless Sensor Nodes (WSN), which can sense and transmit the information. Such devices seek to attain a good quality of service, while functioning for a long duration. Therefore, a series of energy management techniques are proposed in the literature (Anastasi et al., 2009) to reduce the power consumption of the sensor node, among which the most notable one is duty cycling the activity of the components of the node, such as powering on and off the radio transceiver periodically. Another possibility is to harvest and use the energy from the environment (Paradiso and Starner, 2005). Even if the harvesters can ensure a theoretically unlimited amount of energy over time, the power they provide is unpredictable. We address this issue by using power storage elements, such as rechargeable batteries or supercapacitors, in order for the system to have energy when it is not available from the harvester. However, these buffers are finite, and, therefore, they cannot completely hide the unreliability of the energy source, for example, when the harvester is not generating energy for a long period of time.

Concrete examples of environmentally powered sensor nodes are found in the literature: solar powered devices (Moser et al., 2008; Jiang et al., 2005; Dubois-Ferrière et al., 2006), thermal powered (Gyselinckx et al., 2005) and vibrational ones (Roundy et al., 2005).

To motivate the novelty of the paper, we focus on a particular characteristic of these energy harvesters. While the solar radiation and, together with it, the energy output of a photovoltaic panel, have normally a slow variation over the range of hours, an eolian harvester can experience fast variations, in the order of seconds (Twidell and Weir, 1986). We call the former type a slow-dynamics harvester, and the latter a high-dynamics one.

In this paper, we consider a wireless sensor node powered by such a high-dynamics harvester, which uses eolian energy. The node runs a reactive application, which senses and transmits wirelessly the data to a basestation at a specified rate. This rate is normally in the same order of magnitude with the frequency of variation of the harvested energy. Managing such a device is a novel contribution and, due to the fast temporal properties of the harvester, it raises new constraints that we need to take into consideration.

Our goal is to devise a sensor node controller that maximizes the number of measurements of the physical property in the time unit (the sensing duty cycle)

E. Şuşu A. (2010).
STOCHASTIC OPTIMIZATION FOR ENVIRONMENTALLY POWERED WSNS USING MDP MODELS WITH MULTI-EPOCH ACTIONS.
In *Proceedings of the 7th International Conference on Informatics in Control, Automation and Robotics*, pages 238-243
DOI: 10.5220/0002940802380243

and transmit the data to the basestation at the specified rate. Similar optimization formulations can be found in (Kansal et al., 2004), which adapts the system duty cycle in order to match the energy profile intake, and in (Niyato et al., 2007), which computes policies that optimize certain Quality of Service (QoS) metrics. Differently from them, we perform multi-criteria optimization, having two types of commands to manipulate the sensing and the radio transceiver duty cycle.

Since, in principle, we also want to have a reduced production cost, in this paper we consider a sensor node system with: i) an energy harvester device that generates on average the required energy level, since this impacts the production cost; ii) a small sized energy storage element - the size of the buffer is determined by the longest "blackout" period. We argue that the problem we present is useful when the system has little energy in the storage element, when prediction of the energy intake can bring great benefit.

The contributions of the paper are: i) we build a representative Markov chain model for the eolian harvester powering the node; ii) we reduce the sensor node control problem to an average reward Constrained Markov Decision Process (MDP) formulation, one of the most complex planning problems; iii) then, we introduce in the optimization problem the proposed multi-epoch action constraints, relevant for our setting; iv) we solve the problem rigorously using various tools we developed on top of existing software, and compare the benefits of our method to some simple heuristic policies we introduce.

2 DESCRIPTION OF THE SYSTEM MODEL AND OF THE OPTIMIZATION PROBLEM

As advocated in (Şuşu et al., 2008; Poggi et al., 2000), an accurate model of the energetic sources is essential for evaluating the system's average productivity or the availability, and, in principle, to ensure the system's management. Simulation provides results only for the period over which environmental energy data is available. Since the results are different if we use other time series with the same statistical properties, we are interested to know the range of these results. Therefore, finding a representative model able to capture the uncertainty of the energy source requires attentive thinking. (Poggi et al., 2000) proposes a first-order stationary Discrete Time Markov Chain (DTMC) model for each month of the year, due to the big monthly variations, built from traces taken over a period of 20 years. In a similar direction goes (Nfaoui et al., 2003) for wind speed measurements.

Similarly, in this paper we use an offline built first-order DTMC model for an eolian energy harvester using wind speed traces collected by the SensorScope project (Barrenetxea et al., 2006) from EPFL. Since we do not perform experiments with a real device, the model for the energy harvester is using simplifying assumptions such as the energy produced by an eolian harvester is directly proportional to the wind speed. Therefore, in this paper we do not put accent on an end-to-end treatment of this problem from theory to full implementation, but focus mostly on the modeling and optimization part.

The method described in this paper is general in the sense it can be applied in settings using other forms of environmental energy, which can be represented by Markovian models.

The system has a dedicated controller that observes the parameters of the node and controls it in order to optimize its functionality. A command specifies the sensing duty cycle (e.g., with values 0%, 50%, 100%) and the power state of the radio transceiver. Our modeling problem uses a discrete time setting: the controller is invoked at each $time\_step = 1$ *second* period (or epoch).

We assume, after studying the real and simulated values of our sensor node platform, that the energy consumed by the operation of the radio transceiver in an epoch is 40 mJ. The energy consumed in an epoch by the sensing equipment and the microcontroller is a multiple of 10 mJ, proportional to the duty cycle. We also assume the sensor node has a small energy storage element of 1,000 mJ. For our model, we assume that a unit of energy represents 10 mJ.

2.1 The MDP Model

As already discussed, we formulate this control problem on a discrete time MDP model, $\mathcal{M}$, which is defined by: i) a finite set of reachable states (under any policy), $\mathcal{S}$; ii) a finite set of actions, $\mathcal{A}$; iii) an initial probability distribution over $\mathcal{S}$, p^1; iv) a transition probability matrix for each action, represented by the function $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$ denoting the probability to transition at a destination state from a source state if using a specific action; v) two reward functions $r_R, r_{SDC} : \mathcal{S} \rightarrow \mathbb{R}$, which we define below.

For our setting, $\mathcal{M}$ is obtained through the parallel composition of the system modules: the energy harvester, the energy buffer, the node (with the sensing and radio components) and the controller, as depicted in Figure 1. The MDP actions are the already discussed commands that control the sensing duty-

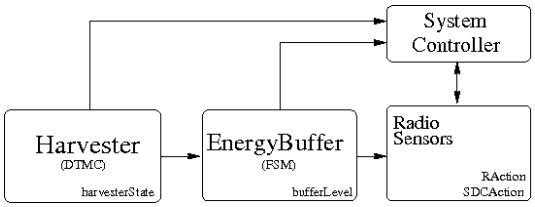

Figure 1: Behavioral black-box model of the system.

cycle and the radio. The duration of an epoch takes $time\_step$ units.

MDP States and Actions: Two of the state variables of the system are $bufferLevel$, which represents the energy stored in the energy buffer at each epoch and $harvesterState$, which describes the energy output of the harvester in a period. The action variables, which are also state variables for the MDP model, are $SDCAction$, which specifies the sensing duty cycle of the node, and $RAction$, which represents the power command to the radio transceiver.

An MDP state $s \in \mathcal{S}$ is defined by the tuple of values of these four variables. Also, an MDP action $a \in \mathcal{A}$ is completely defined by the values of $SDCAction$ and $RAction$ in the following epoch.

MDP Rewards: To express the quality of the sampled data by the node, we add to each MDP state s the reward $r_{SDC}(s)$, which is a function of the sensing duty cycle. We use in this paper a concave function: 0.0 for 0% duty cycle (for $SDCAction = 0$), 8.0 for 50% duty cycle ($SDCAction = 1$) and 10.0 for 100% duty cycle ($SDCAction = 2$).

As already discussed, we can put a soft real time constraint on the functionality of the node: we want to have a certain average period, which we call $T_{lateness}$, of transmitting via radio the acquired data to the basestation. We do this in order that the basestation continuously benefits of sensed data that is about $T_{lateness}$ seconds old.

To account for the frequency of sending via radio the data we put on each state s the reward $r_R(s)$ with value 1.0 if s employs the radio transceiver ($RAction = 1$) and 0.0 if not ($RAction = 0$).

We also do not want to allow the possibility of issuing an action that results in running out of energy without terminating the initiated action. Therefore, we put big negative rewards on such states, such that the solution MDP policy avoids them. This makes sense since the solution either maximizes the objective function value, or keeps the value of the constraint above a specified threshold and using any number of big negative reward states in the solution is not compatible with these goals. For the simulations we perform in Section 3.1, if we run out of power in an epoch (i.e, $bufferLevel' < 0$) we consider the sensing duty cycle reward and the radio reward to be zero.

Semantics: The precise semantics of the MDP model is the following: i) we start with $RAction = 0$ and $SDCAction = 2$ (the radio transceiver is turned off and the node senses with maximum duty cycle), and with $bufferLevel$ set at maximum; ii) the energy intake happens at the beginning of the epoch, instantaneously, and the consumption happens immediately afterwards during the same epoch, by executing the actions for $RAction$ and $SDCAction$, associated to the state; iii) the controller observes the values of the system variables at the beginning of the epoch and computes the values of $RAction'$ and $SDCAction'$ for the next epoch.

2.2 The Optimization Problem

As we have previously stated, we seek to maximize the sensing MDP reward, while meeting an average radio transmission rate of $1/T_{lateness}$.

The solution to our problem is a Markov randomized stationary policy $\pi : \mathcal{S} \rightarrow \triangle(\mathcal{A})$, which prescribes for each state the optimal probability distribution over the actions the controller has to use in order to attain the problem objectives.

The mathematical formulation of our problem is given in (1). $T_{latency}$ is a constant representing the number of epochs the radio needs to transmit, on average, during a $T_{lateness}$ period, for the time being, equal to 1.

$$\max_{\pi} \lim_{N\rightarrow\infty} \frac{1}{N} E^{\pi}\{\sum_{i=0}^{N-1} r_{SDC}(i)\} \qquad (1)$$
$$\text{s.t.} \lim_{N\rightarrow\infty} \frac{1}{N} E^{\pi}\{\sum_{i=0}^{N-1} r_{R}(i)\} \geq \frac{T_{latency}}{T_{lateness}}$$

We can prove our MDP is unichain. In such a case, (Puterman, 1994) arguments we can reduce this optimization problem to the Linear Program (LP) in (2). The variables of the LP, $x_{s,a}$, are called limiting average state-action frequencies and represent the probability the system occupies state s and chooses action a under the solution policy corresponding to x.

$$\max \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} x_{s,a} r_{SDC}(s) \qquad (2)$$
$$\text{s.t.} \sum_{a\in\mathcal{A}} x_{s,a} - \sum_{s'\in\mathcal{S}}\sum_{a\in\mathcal{A}} p(s'|s,a) x_{s',a} = 0, \forall s \in \mathcal{S},$$
$$\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} x_{s,a} = 1,$$
$$\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} x_{s,a} r_R(s,a) \geq \frac{T_{latency}}{T_{lateness}},$$
$$x_{s,a} \geq 0, \forall s \in \mathcal{S}, a \in \mathcal{A}$$

```
MDP

const int PSEMAX = 100; //the capacity of the energy storage element
                        //(1 unit = 10 mJ)
module Harvester
  harvesterState: [0..9] init initHarvesterState;

  [tick] harvesterState = 0 ->  0.927 : (harvesterState' = 0)
      + 0.064 : (harvesterState' = 1) + ...;
  //...
  //the rest of the description of this module is omitted
endmodule

formula update_bufferLevel = bufferLevel + harvesterState - SDCAction
  - 4*RAction;

module EnergyBuffer
  bufferLevel:[0..PSEMAX] init PSEMAX;

  [tick] update_bufferLevel >  0 & update_bufferLevel <= PSEMAX ->
         (bufferLevel' = update_bufferLevel);
  [tick] update_bufferLevel >  PSEMAX -> (bufferLevel' = PSEMAX);
  [tick] update_bufferLevel <= 0 -> (bufferLevel' = 0);
endmodule

//A generic system controller: see both RadioController and
  SDCController
  module RadioController
  RAction:[0..1];

  [tick] true -> (RAction' = 0);
  [tick] true -> (RAction' = 1);
endmodule

module SDCController
  SDCAction:[0..2];

  [tick] true -> (SDCAction' = 0);
  [tick] true -> (SDCAction' = 1);
  [tick] true -> (SDCAction' = 2);
endmodule
```

Figure 2: The PRISM MDP model of the system.

3 PRACTICAL DETAILS OF THE PROBLEM

To concretely solve the problem, we start by specifying the model in PRISM, a probabilistic analysis tool (Kwiatkowska et al., 2004) we use extensively in our project. A simple model example with a freely specified system controller, the unknown of our problem, is given in Figure 2.

We develop some small programs that take the MDP representation exported from PRISM and output the corresponding LP that solves the MDP optimization problem. To make the problem more precise and robust, the coefficients of the LP have 16 decimal digits precision. We solve the LP with CPLEX (IBM ILOG, 2008), a fast commercial (I)LP solver, which is able to handle LPs up to tenths of thousands of variables.

The capacity of the energy storage element, $PSEMAX$, is 100 and $T_{lateness}$ is 10. We also assume there is only one initial state, the one with a full energy storage element, with the maximum level of harvested energy, with $RAction = 0$ and $SDCAction = 2$.

Once the MDP policy is obtained, to assess that it conforms to the specification - mainly that the expected radio reward is met - but also to compute various other properties such as the value of the expected sensing reward, we compose the solution policy with the PRISM specification of the MDP and perform exhaustive simulation (i.e., probabilistic model checking) by using the cumulative reward properties of PRISM to compute the expected total rewards for a specified number of steps, and then compute the average reward per epoch.

Since we want to show the benefits of implementing a stochastic policy controller, we introduce two more runtime policies, which are deterministic and compare the three of them in the following sections. The three different controllers are:

- **MDP:** this is the stochastic policy, which uses the energy harvester model to accurately predict the future energy income. Relying on the current value of $bufferLevel$ and on the prediction, the policy chooses the optimal sensing duty cycle, $SDCAction'$, and the power state for the radio, $RAction'$.
- **Conservative:** the policy varies the sensing duty cycle proportionally with the $bufferLevel$. Also, the policy controls the radio the following way: in the case $bufferLevel$ is too small, the controller assumes the worst-case situation under which it does not receive anymore energy from the environment, and turns off the radio; otherwise, the radio is turned on, periodically.
- **Greedy:** the policy uses the maximum sensing duty cycle, no matter what the value of $bufferLevel$ is. The node tries to turn the radio on and to send data to the basestation periodically, at every $T_{lateness}$ seconds, disregarding the fact it might run out of power.

The Conservative and Greedy policies do not make use of the probabilistic harvester model. The Greedy controller does not care about the state of the energy harvester and observes, at most, only the charge of the energy buffer. In all the experiments we perform in the following sections, we count for the radio and sensing rewards of the policies only the epochs in which the system does not run out of energy.

3.1 Considering The Multi-epoch Radio Transceiver Actions

So far, we did not take into consideration that turning on the radio transceiver of the sensor node followed by transmitting the acquired data are costly operations, which can take longer than one epoch

```
module RadioController
  RAction:[0..1];
  counter:[0..T_LATENESS - 1];

  [tick] counter = 0 -> (RAction' = 0) & (counter' = 1);
  [tick] counter > 0 & counter <= T_LATENCY - 1 & RAction = 0 ->
         (RAction' = 0)
      & (counter' = counter + 1);
  [tick] counter = 0 -> (RAction' = 1) & (counter' = 1);
  [tick] counter > 0 & counter <= T_LATENCY - 1 & RAction = 1 ->
         (RAction' = 1) & (counter' = counter + 1);
  [tick] counter >= T_LATENCY & counter < T_LATENESS - 1 ->
         (RAction' = 0) & (counter' = counter + 1);
  [tick] counter = T_LATENESS - 1 -> (RAction' = 0) & (counter' = 0);
endmodule
```

Figure 3: The PRISM radio controller specification with $T_{latency}$-epochs long actions.

because the node has to synchronize with the network. For the SensorScope platform, which uses the BMAC protocol, the time the radio needs to synchronize to the network and send the data is $T_{latency\_exact} = 2.25$ *seconds* (Şuşu et al., 2008). For the sake of the energy efficiency, since the radio is the most power consuming component of the sensor node, we do not want to turn it off while synchronizing or transmitting. Therefore, under these conditions, the radio power off command should be issued at least $T_{latency} = \lceil \frac{T_{latency\_exact}}{time\_step} \rceil = 3$ epochs later from the moment we turned on the radio.

In order to handle the optimization with multi-epoch actions, we add a new state variable, *counter*, to the PRISM model, which keeps track of how many epochs have passed from the beginning of the current $RAction = 1$ issue. We then allow to change to $RAction = 0$ only if $counter \geq T_{latency}$.

Adding this variable gives us the possibility to turn on the radio only at the beginning of the $T_{lateness}$ period for exactly $T_{latency}$ periods. This implies that the counter variable has to take values between 0 and $T_{lateness} - 1$. The PRISM specification of the radio controller with the variable *counter* is given in Figure 3. We compose this component with the others of the model in Figure 2. The radio constraint in our formulations (1) and (2), specifying to have at least $T_{latency}/T_{lateness}$ average radio reward per epoch, is in accordance with the behavior described above.

4 OPTIMAL SOLUTION AND COMPARISON

We call π_3 the MDP optimal policy that generates only $T_{latency}$-epochs long radio actions ($T_{latency} = 3$) and π_1 the one on which we do not impose this constraint.

We present in Table 1 the simulation results obtained with PRISM for π_3, together with the ones for the Conservative and the Greedy policies, for which we present only the radio rewards that are $T_{latency}$-epochs (or more) long. As we can see, the π_3 policy has average rewards per epoch of 0.30 for the radio and 4.53 for sensing. π_1 attains on average per epoch rewards of 0.30 for the radio (out of which only 49.57% satisfies the $T_{latency}$ constraint) and 4.96 for sensing. To compare π_1 and π_3 we use the product between the expected $T_{latency}$-feasible radio reward and the expected sensing reward. This product is 0.744 for the former and 1.359 for the latter, which means π_3 performs with 83% better than π_1. Using the same metric, π_3 is better with 14% than the Conservative policy and with 154% than the Greedy one. The Greedy policy is the only one that runs out of energy, about 1.73% of the time.

Table 1: The expected total radio (R) and sensing (SDC) rewards of the solution policies for constrained optimization, for various horizon length (*hl*) values.

hl	MDP policy π_3		Conservative policy		Greedy policy	
	R	SDC	R	SDC	R	SDC
10^4	3,099	45,587	1,945	61,768	1,031	52,727
10^5	30,822	453,302	19,395	615,398	10,192	524,071
10^6	308,055	4,530,441	193,893	6,151,698	101,809	5,237,512

The MDP has 39,390 reachable states. The associated LP uses six times more variables and takes more than six days to be solved exactly on a standard computer platform. Therefore, we use competitive approximation methods, the details of which we omit in this paper, to find the solution, which reduce the search time to a couple of hours.

The optimal policy π_3 prescribes for each reachable state of the system the best sensing duty cycle and radio management that maximizes the expected sensing quality, generates only $T_{latency}$-epochs long radio transmissions and does not run out of energy, for the given harvester DTMC model.

5 CONCLUSIONS

In this paper we have modeled and improved the functionality of a wireless sensor node with Markov Decision Processes (MDPs). Because of the long time the radio transceiver takes to synchronize with the network, we have introduced MDP actions that take longer than one epoch to complete. Optimizing without taking into consideration these multi-epoch actions results in suboptimal MDP policies. We proposed a method to find an optimal solution and compared the result to various heuristic policies.

Our problem with multi-epoch actions has some similarities with the Semi-Markov Decision Process

(SMDP) (Puterman, 1994; Hu and Yue, 2007) models with variable sojourn times. Note that our setting is even more different, since we have action types with different completion times.

REFERENCES

Anastasi, G., Conti, M., Di Francesco, M., and Passarella, A. (2009). Energy conservation in wireless sensor networks: A survey. *Ad Hoc Netw.*, 7(3):537–568.

Barrenetxea, G., Dubois-Ferriere, H., Meier, R., and Selker, J. (2006). A weather station for SensorScope. In *Demo Session, In Information Processing in Sensor Networks (IPSN 2006)*.

Dubois-Ferrière, H., Meier, R., Fabre, L., and Metrailler, P. (2006). TinyNode: A Comprehensive Platform for Wireless Sensor Network Applications. In *Information Processing in Sensor Networks (IPSN 2006)*.

Gyselinckx, B., Hoof, C. V., Ryckaert, J., Yazicioglu, R. F., Fiorini, P., and Leonov, V. (18-21 Sept. 2005). Human++: Autonomous wireless sensors for body area networks. In *Custom Integrated Circuits Conference, 2005. Proceedings of the IEEE 2005 , vol., no.pp. 13-19*.

Hu, Q. and Yue, W. (2007). *Markov Decision Processes with Their Applications*. Advances in Mechanics and Mathematics, v. 14. Springer, Dordrecht.

IBM ILOG (2008). ILOG CPLEX User's Manual.

Jiang, X., Polastre, J., and Culler, D. E. (2005). Perpetual environmentally powered sensor networks. In *IPSN*, pages 463–468.

Kansal, A., Potter, D., and Srivastava, M. B. (2004). Performance aware tasking for environmentally powered sensor networks. *SIGMETRICS Perform. Eval. Rev.*, 32(1):223–234.

Kwiatkowska, M., Norman, G., and Parker, D. (2004). Prism 2.0: A tool for probabilistic model checking. *QEST*, 00:322–323.

Moser, C., Thiele, L., Brunelli, D., and Benini, L. (2008). Approximate control design for solar driven sensor nodes. In *HSCC '08: Proceedings of the 11th international workshop on Hybrid Systems*, pages 634–637, Berlin, Heidelberg. Springer-Verlag.

Nfaoui, H., Essiarab, H., and Sayigh, A. (2003). A stochastic Markov chain model for simulating wind speed time series at Tangiers, Morocco. *Renewable Energy 29 1407-1418*.

Niyato, D., Hossain, E., and Fallahi, A. (2007). Sleep and wakeup strategies in solar-powered wireless sensor/mesh networks: Performance analysis and optimization. *IEEE Transactions on Mobile Computing*, 6(2):221–236.

Paradiso, J. A. and Starner, T. (2005). Energy scavenging for mobile and wireless electronics. *Pervasive Computing, IEEE*, 4(1):18–27.

Poggi, P., Notton, G., Muselli, M., and Louche, A. (2000). Stochastic study of hourly total solar radiation in Corsica using a Markov model. *International Journal of Climatology, Volume 20, Issue 14, Pages 1843 - 1860*.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience.

Roundy, S., Leland, E. S., Baker, J., Carleton, E., Reilly, E., Lai, E., Otis, B., Rabaey, J. M., Wright, P. K., and Sundararajan, V. (2005). Improving power output for vibration-based energy scavengers. *Pervasive Computing, IEEE*, 4(1):28–36.

Şuşu, A. E., Acquaviva, A., Atienza, D., and Micheli, G. D. (2008). Stochastic modeling and analysis for environmentally powered wireless sensor nodes. *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops, 2008. WiOPT 2008. 6th International Symposium on*, pages 125–134.

Twidell, J. W. and Weir, A. D. (1986). Renewable energy resources.