Staying Alive: System Design for Self-Sufficient Sensor Networks

NICOLA BUI, IMDEA Networks Institute
MICHELE ROSSI, University of Padova

Self-sustainability is a crucial step for modern sensor networks. Here, we offer an original and comprehensive framework for autonomous sensor networks powered by renewable energy sources. We decompose our design into two nested optimization steps: the inner step characterizes the optimal network operating point subject to an average energy consumption constraint, while the outer step provides online energy management policies that make the system energetically self-sufficient in the presence of unpredictable and intermittent energy sources. Our framework sheds new light into the design of pragmatic schemes for the control of energy-harvesting sensor networks and permits to gauge the impact of key sensor network parameters, such as the battery capacity, the harvester size, the information transmission rate, and the radio duty cycle. We analyze the robustness of the obtained energy management policies in the cases where the nodes have differing energy inflow statistics and where topology changes may occur, devising effective heuristics. Our energy management policies are finally evaluated considering real solar radiation traces, validating them against state-of-the-art solutions, and describing the impact of relevant design choices in terms of achievable network throughput and battery-level dynamics.

Categories and Subject Descriptors: G.1.6 [**Optimization**]: Stochastic Programming

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Energy harvesting, energy self-sufficiency, protocol design, wireless sensor networks

ACM Reference Format:
Nicola Bui and Michele Rossi. 2015. Staying alive: System design for self-sufficient sensor networks. ACM Trans. Sensor Netw. 11, 3, Article 40 (February 2015), 42 pages.
DOI: http://dx.doi.org/10.1145/2700269

1. INTRODUCTION

The operation of wireless sensor networks powered by renewable sources is a very lively area of research, both theoretical and applied. This is due to the increasing inclination toward green systems and to the need for Wireless Sensor Networks (WSNs) that can last unattended indefinitely. In fact, despite the advances in microprocessor fabrication and protocol design, batteries are expected to last for less than 10 years for many applications, and their replacement is in some cases prohibitively expensive. This problem is particularly severe for urban sensing applications, for example, sensors placed below the street level, where the installation of new power cables is impractical. Other examples include body sensor networks or WSNs deployed in remote geographic areas [Wang and Liu 2011]. In contrast, WSNs powered by energy-scavenging devices

The research leading to these results has received funding from the Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 251557 (Project SWAP).
Authors' addresses: N. Bui, IMDEA Networks Institute, Av. Mar del Mediterraneo, 22, 28918, Madrid, Spain; email: nicola.bui@imdea.org. M. Rossi, Department of Information Engineering (DEI), University of Padova, Via Gradenigo 6/B, 35131 Padova, Italy; email: rossi@dei.unipd.it.

DOI: http://dx.doi.org/10.1145/2700269

provide potentially maintenance-free perpetual networks, which are particularly appealing, especially for the highly pervasive Internet of Things [Atzori et al. 2010].

In the past few years, a vast literature has emerged on energy-harvesting WSNs. These networks are made of tiny sensor devices with communication capabilities, which also have an onboard rechargeable battery (also referred to as an energy buffer) and are capable of scavenging energy from the surrounding physical environment. Most of the research papers that have been published so far deal with the *energy-neutral design* of transmission policies, where the concept of energy neutrality accounts for the fact that the energy used, in the long term, should be equal to that harvested. Within this body of work, two well-established approaches have been adopted to find energy-neutral policies, namely, *offline* and *online*. Offline solutions are concerned with finding optimal packet transmission schedules, assuming that the nodes have full knowledge of the harvesting and information generation processes. Although this is unrealistic, it provides useful insights into the design of online strategies. On the other hand, online approaches only assume some prior statistical knowledge about the energy arrival and the input data processes.

Offline approaches: Ozel et al. [2011] consider a single sensor node transmitting data over a wireless fading channel with additive Gaussian noise and causal channel state information at the transmitter. They obtain optimal policies considering two objectives: maximize the throughput by a deadline and minimize the transmission completion time. Yang and Ulukus [2012] generalize the results of Ozel et al. [2011] by relaxing the assumption on packet arrivals, which can now arrive during transmissions. Also, they derive fast search algorithms leveraging structural properties of the solution. In another recent work [Gregori and Payaró 2013] relaxes the assumption that the battery is infinite, obtaining optimal transmission policies for given Quality of Service (QoS) constraints while fulfilling data and energy causality constraints. To the best of our knowledge, no papers in this category studied energy management policies for the network of devices.

Online approaches: These approaches differ in the stochastic model considered for the energy arrival process and in the optimization objective. Notably, only a few contributions addressed aspects related to multiple access and routing in distributed networks. Vigorito et al. [2007] present a decentralized strategy for the control of an energy buffer with stochastic replenishment through the adaptation of the transmission duty cycle. They model the optimal buffer management as an online optimization problem, estimating the system dynamics using a gradient descent update rule and implementing energy-centric policies. Similarly, Hsu et al. [2006] present an adaptive duty cycling algorithm for energy-harvesting sensor nodes.

Kansal et al. [2007] study fundamental properties of energy-harvesting processes and utilize them to devise an algorithm that maximizes the throughput based on energy prediction. Fan et al. [2008] propose a solution for high throughput with fairness guarantees, devising centralized and distributed algorithms that compute the optimal lexicographic rate assignment for all nodes. Lei et al. [2009] develop a Markov decision analysis for a sensor node with i.i.d. stochastic replenishments (i.e., fixed energy arrival rate) and a finite energy buffer. They devise optimal online policies that depend on the importance of packets, which is modeled through a generic probability distribution function (pdf). Sharma et al. [2010] propose throughput as well as delay optimal online policies for a sensor node with infinite data and energy queues. They consider stationary and ergodic arrival processes for data and energy and transmission over fading channels. Michelusi et al. [2013] generalize the results of Lei et al. [2009]: they model energy replenishment through a two-state Markov model and associate a cost with data transmission. Optimal and heuristic policies are characterized considering the long-term data importance of transmitted data through a dynamic

programming formulation. The focus of Luo et al. [2013] is instead on practical circuits for energy-harvesting wireless transmitters and on their impact on the design of optimal transmission policies for TDMA channel access. They optimize the time spent in storing energy and transmitting while accounting for QoS constraints and a TDMA access scheme.

Other approaches dealing with multiple access channels and, in turn, considering the simultaneous interaction of multiple sensor nodes are Gatzianas et al. [2010], Huang and Neely [2013], Michelusi and Zorzi [2013], and Tapparello et al. [2013]. To our knowledge, Gatzianas et al. [2010] is the first contribution that has dealt with the distributed control of energy-harvesting WSNs. There, the authors present an online and adaptive policy for the stabilization and optimal control of these networks using tools from Lyapunov optimization. This line of work has been continued by Huang and Neely [2013], who tackle the distributed routing problem using the Lyapunov optimization theory combined with the idea of weight perturbation (see, e.g., Neely et al. [2008]). Michelusi and Zorzi [2013] consider a single-hop WSN where each node harvests energy from the environment and randomly accesses the channel to transmit packets of random importance to a sink node. Thus, optimal distributed policies, based on a game theoretic formulation of the random access problem, are proposed. Tapparello et al. [2013] present a theoretical framework that extends Gatzianas et al. [2010] and Huang and Neely [2013] by proposing joint transmission, data compression (distributed source coding, DSC), and routing policies that minimize the long-term expected distortion of the signal reconstructed at the sink while ensuring the energetic stability of the network.

Other research directions deal with energy-sharing networks [Zhu et al. 2010] and laser-power beaming [Bhatti et al. 2014]. However, in the present contribution, we look neither at the possibility of exchanging energy among nodes nor at performing wireless energy transfer. Further extensions may involve the adoption of energy-aware programming languages [Sorber et al. 2007].

Our contribution: Our present work belongs to the online category and considers networks of energy-harvesting devices. Specifically, we propose a framework based on the dynamic adaptation of two key protocol parameters, namely, the radio *duty cycle* d_c and the *transmission frequency* for the generated traffic, f_U. This framework permits us to assess the performance of energy-harvesting sensor networks while shedding new light into the pragmatic design of energy management solutions.

Toward this end, we account for (1) the network topology; (2) the transmission of endogenous (own packets) data; (3) the relaying of exogenous (forwarded) data; (4) the amount of energy consumed for transmission, reception, idling, processing, and so forth; (5) the channel access mechanism; and (6) the harvested energy inflow dynamics. For the channel access, we consider the Low-Power Listening (LPL) MAC [Buettner et al. 2006; Bonetto et al. 2012], whereas routing dynamics are modeled through the IETF Routing for low-Power Lossy networks (RPL) [Ko et al. 2011; Bui et al. 2012].

Technically, our first contribution is a model that, for any pair (d_c, f_U), returns the associated average energy consumption of a sensor node, taking (1) through (5) as input. We obtain (in closed form) the pair (d_c^*, f_U^*) that maximizes the node throughput subject to a given energy constraint. We subsequently locate the bottleneck node in the network (the one suffering the highest amount of interference) and we carry out a further optimization step based on (6), keeping this worst case into account. The resulting policies dynamically select the pair (d_c, f_U) considering the state of the bottleneck node along with the stochastic model of the harvested energy. Being dimensioned for the worst case, the obtained policies can be applied at all nodes, leading to the self-sufficient operation of the entire WSN. Hence, we comment on the behavior of the obtained energy management policies and we compare their performance against that

Table I. Notation

Capital letters: N, S, etc.	denote system states and functional blocks.
Capital letters, italic: I_{out}, I_{TX}, etc.	denote average quantities.
Lower letters, italic: t_{off}, t_{dc}, etc.	denote variables.
Calligraphic font: $\mathcal{S}$, $\mathcal{U}$, etc.	denotes sets.
Greek letters: τ, ι, etc.	denote random variables.
Bold letters: p, ρ, etc.	denote vectors.

Table II. Symbol Definitions

S	energy source block in the system model.
B	energy buffer (battery) block in the system model.
N	energy consumer (sensor node) block in the system model.
$\mathcal{N}$	sensor nodes set.
i	harvested current.
u	control policy (drained current).
d_c	duty cycle.
I_{out}	average current consumption for a given network configuration.
f_{U}	packet transmission rate for endogenous traffic (reward).

of competing solutions from the state of the art. Finally, we relax each of the model assumptions, showing that the solutions so obtained are still robust.

In summary, the main contributions of the present article are as follows:

(1) A model for the energy consumption of a network of embedded wireless devices
(2) A closed-form formula for the optimal operating point of the network
(3) A mathematical framework to maximize the throughput performance while allowing the perpetual operation of the entire sensor network
(4) A performance evaluation of the proposed energy management policies
(5) A validation of the proposed solution when the model assumptions are relaxed

In Table I, we introduce the notation used in the rest of the article. Additional definitions will be given at the beginning of each section.

The remainder of this article is organized as follows. In Section 2, we describe the workflow of the article, detailing the objectives of our design and how these are accomplished by the analyses that follow. In Section 3 and Section 4, we characterize the energy consumption of a sensor node according to the network properties and we derive the optimal operating point for the network subject to input energy constraints. In Section 5, we present a stochastic semi-Markov model for the harvested energy, and in Section 6, we obtain energy management policies for self-sufficient networks of embedded devices. In Section 7 and Section 8, we evaluate the proposed policies, and in Section 9, we present our closing remarks.

2. PROBLEM FORMULATION

In this section, we describe the problem formulation as two nested optimization problems. The list of used symbols is given in Table II.

We consider a wireless sensor network $\mathcal{N}$ composed of $N = |\mathcal{N}|$ *homogeneous* embedded devices, where sensor nodes transmit their readings to a data collector node (referred to as *sink*). The nodes are deployed according to a certain multihop topology, and the data packets are routed toward the sink through a predetermined collection tree, as detailed in Section 3. Each sensor node is described through the diagram in Figure 1. Specifically:

—**Energy source (S):** This block accounts for the presence of some energy-scavenging circuitry that feeds a storage unit. The amount of harvested current is described

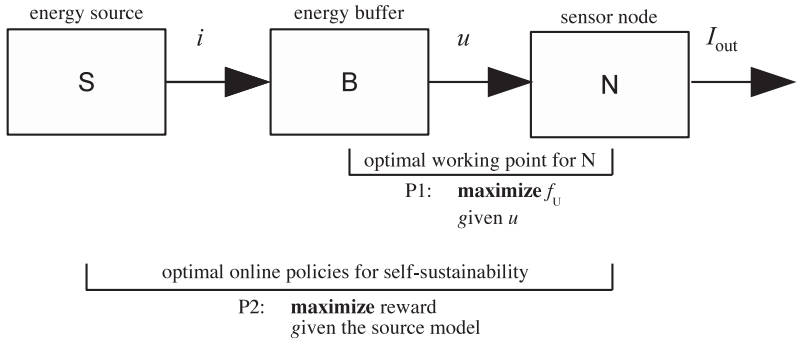


Fig. 1. Sensor node diagram.

by the variable i. A detailed description of a stochastic semi-Markov model of S is provided in Section 5. Note that while the energy scavenged is stochastic across time, we initially assume that it is described by the same Markov source for all nodes. The extension to heterogeneous energy sources is provided in Section 8.1.

—**Battery (B):** The storage unit (e.g., either a rechargeable battery or a supercapacitor) provides an average current u to the following block N; see Section 6.

—**Sensor node (N):** This block models the aggregate energy consumption of a sensor node, which is referred to as I_{out}. This accounts for the energy drained by the sensor node hardware, including the network protocol stack (e.g., routing, channel access, and physical layer), the onboard sensors, and the CPU. The energy consumption of block N is characterized in Section 3.

The overall objective of our analysis is providing dynamic and energy-dependent (i.e., depending on the state of S and B) configurations for the sensor nodes in $\mathcal{N}$ so that the entire network will be energetically self-sufficient.

To accomplish this, for a given network setup, we first identify the so-called *bottleneck* node, which is the node experiencing the highest traffic load. This node is by definition the one subject to the highest energy consumption (more precise details will be given in Section 3 and Appendix D).

Our analysis develops along the following two optimization steps:

1) We first characterize the energy consumption of the bottleneck node for the given routing topology and channel access technology. In detail, we relate its average energy consumption, I_{out} (assumed constant for this first analysis), to two key parameters: the *radio duty cycle*, d_c, and the *transmission frequency* for the endogenous traffic, f_U. Given this, we solve a first optimization problem P1 (the inner problem in Figure 1), where we seek the operational point (i.e., the pair (d_c, f_U)) for which f_U is maximized considering u as the energy consumption constraint. To solve P1, we model the interaction of the bottleneck node with respect to the other sensors in $\mathcal{N}$, accounting for the transmission behavior of all nodes within range (e.g., the amount of relay traffic from the children nodes, the total traffic that these forward on behalf of their children, the number of interferers and their transmission rate, etc.). Subsequently, we derive in closed form the optimal protocol configuration (d_c, f_U) for a given average energy consumption constraint u.
2) In the second optimization step (problem P2), we additionally account for the presence of blocks S and B, where S is modeled through a stochastic time-correlated Markov model, where the harvested current i is assumed to be a time-varying, correlated stochastic process and u is now the control variable. Problem P2 consists of dynamically selecting the control u (or, equivalently, the pair (d_c, f_U), where the

Table III. Symbol Definitions

$x \in \mathcal{X}_{\mathcal{N}}$	node operational state x and state set $\mathcal{X}_{\mathcal{N}}$.
f'_{U}	modified reward function accounting for retransmissions.
$t_{\mathrm{on}}, t_{\mathrm{off}}, t_{\mathrm{data}}, t_{\mathrm{dc}}, t_{\mathrm{U}}, t_{\mathrm{v}}, t_{\mathrm{rpl}}$	sensor node timings.
i_x, I_x	instantaneous (i_x), and average (I_x) currents drained in state x.
$i_{\mathrm{c}}, i_{\mathrm{t}}, i_{\mathrm{r}}, i_{\mathrm{s}}$	currents drained by the cpu (i_{c}), radio ($i_{\mathrm{r}}, i_{\mathrm{t}}$), and sensing unit ($i_{\mathrm{s}}$).
t_x, r_x, f_x	average duration, frequency and fraction of time spent in state x.
k_{U}	constant accounting for energy drained due to sensing and computation.
$n_{\mathrm{c}}, n_{\mathrm{i}}, n_{\mathrm{int}}$	network topology parameters.
$e_{\mathrm{t}}, e_{\mathrm{c}}, e_{\mathrm{p}}$	channel error (e_{t}), collision (e_{c}), and total error (e_{p}) probabilities.

relation $u \rightarrow (d_c, f_{\mathrm{U}})$ follows from the solution of P1) for the given energy source model, so that the bottleneck will maximize its own throughput while being energetically self-sufficient.

At this point, we combine the results of P1 and P2: P1 decides the optimal operating point for the bottleneck as a function of u, whereas P2 dictates how u should vary as a function of the battery state and on some statistical knowledge of the energy-harvesting process. This combined optimization amounts to a dynamic selection of the current level u that has to be drained by the node, depending on the state of S and B, so that the throughput is maximized (P1) and the node is energetically self-sufficient (P2).

After solving this combined problem, the self-sufficiency of all network nodes can be ensured by the following scheme. The time is divided into a number of slots, which depend on the temporal characterization of the energy-scavenging process; see Section 5. A *decision epoch* occurs at the beginning of each slot, that is, when the source model transitions to a new state. Thus, at each epoch, the sink collects the information about the state of the battery of the bottleneck node, computes the optimal actions (using P1 and P2) for the next time slot for this node, and sends back a description of the computed optimal policy to all network nodes. Thus, all nodes will implement, in the next time slot, the policy that is optimal for the bottleneck. Consequently, the energetic stability at all nodes is ensured. This can be conveniently implemented through a practical network management and routing protocol such as RPL [Winter et al. 2010].

In this article, we look at a coarse-grained control of the protocol behavior of the nodes. In fact, one control command has to be sent out to the nodes at the beginning of every time slot, whose duration depends on the number of states that are used to model the energy inflow during a typical day. While our mathematical analysis holds for any number of energy states, practical considerations related to the network overhead incurred in sending control actions to the nodes and to the number of states that is sufficient to accurately model (e.g., typical solar sources) lead to slot durations on the order of hours.

In Section 3, for a given network scenario (i.e., transmission model, topology, and data collection tree), we characterize the energy consumption of the bottleneck node. Thus, in Sections 4 and 6, we respectively solve problems P1 and P2 for this node, assuming that all the remaining nodes in the network behave in the same exact manner as the bottleneck does.

In Section 8.1, we extend our analysis to the case where the sensor nodes harvest different amounts of energy.

3. NODE CONSUMPTION MODEL

The symbols used in this section are listed in the Table III.

In this section, we discuss the sensor node block of our architecture: this entails the definition of a tractable framework to model the interactions among nodes,

including routing and channel access (MAC). We require the model to track network characteristics such as the topology, the adopted MAC protocol, channel errors, and internal processing (assembling data packets, etc.). Although our framework develops along the lines of Fischione et al. [2013], we aim at obtaining simple and meaningful relationships that will make it possible to compute the optimal throughput in closed form.

For tractability, we make the following assumptions:

1) There exists a node that consumes more energy than any other sensor. This node is referred to as the *bottleneck* node.
2) Every sensor operates as the bottleneck node in terms of information generation rate, f_{U} (expressed in packets per second), and duty cycle, $d_c = t_{\mathrm{on}}/t_{\mathrm{dc}} = t_{\mathrm{on}}/(t_{\mathrm{on}}+t_{\mathrm{off}})$, where $t_{\mathrm{dc}} = t_{\mathrm{on}} + t_{\mathrm{off}}$, whereas t_{on} and t_{off} are the durations of the active and sleeping portions of the duty cycle, respectively.
3) The sink at each decision epoch (see Section 6) collects the status of the bottleneck, in terms of energy reserve, and broadcasts a feedback message to adapt the protocol behavior of all nodes. We provide practical considerations on how to deal with dissemination delays in Section 8.
4) The sensor nodes maintain the same behavior for long enough to justify the use of average energy consumption figures. Specifically, the time scale at which the sink takes control actions is much coarser than that related to the radio duty cycling.

To start with, we identify the operational states of a sensor node and, for each of them, the associated energy expenditure (expressed here in terms of the current i_x drained in each state x):

—**TX**: this is the transmission state. Here, both the microprocessor and the radio transceiver are active and the current drained by these components is i_{c} and i_{t}, respectively.
—**RX**: in this state, a node receives and decodes a radio frame. As for the TX state, both the microprocessor and the radio transceiver are on, and in this case, their energy drainage is i_{c} and i_{r}, respectively.
—**INT**: in this state, the node receives a frame that neither is intended for it nor has to be forwarded by it. Here, the node drains exactly the same current as in state RX. In the following analysis, we track this state separately from RX as the rate of interfering and successful transmissions may differ.
—**CPU**: the node is busy with operations that do not require any radio activity (e.g., sensing, data processing, encoding, etc.). In this state, the radio transceiver is off or in a power-saving state, and thus, the consumption is just i_{c}.
—**IDLE**: the node is idle and can switch to some low-power state. However, since preamble-sampling MAC protocols, such as X-MAC [Buettner et al. 2006] or LPL [Moss et al. 2007], need to periodically sample the radio channel while idling, it is convenient to split this state into two substates:
  —**CCA**: in this state, the node samples the channel (Clear Channel Assessment). Hence, it drains the same current as in RX.
  —**OFF**: this is the state with the lowest energy consumption. Here, the microprocessor and the radio transceiver are in power-saving mode and the total current drained by the device is i_{s}, which is much smaller than all the other energy consumption figures ($i_{\mathrm{s}} \ll i_x, x \in \{\mathrm{t}, \mathrm{r}, \mathrm{c}\}$).

We now formally introduce the system state set

$$\mathcal{X}_{\mathcal{N}} = \{\mathrm{TX}, \mathrm{RX}, \mathrm{INT}, \mathrm{CPU}, \mathrm{CCA}, \mathrm{OFF}\}, \tag{1}$$

where for the IDLE state it holds that $\mathrm{IDLE} = \mathrm{CCA} \cup \mathrm{OFF}$. The main idea behind our model consists of computing the average current $I_x = \mathrm{E}[i_x]$ drained by the bottleneck

node for each state $x \in \mathcal{X}_{\mathcal{N}}$ for the given protocol and network parameters. Note that, in our model, computing average currents is equivalent to computing powers, as we assume that the sensors operate according to a fixed supply voltage. For each $x \in \mathcal{X}_{\mathcal{N}}$, we have that $I_x = i_x t_x f_x$, where i_x, t_x, and f_x correspond to the drained current, the average permanence time (duration) in state x, and the average rate (frequency) at which state x is entered, respectively. In addition, we use the quantity $r_x = t_x f_x$ to indicate the average fraction of time the node spends in state x. Hence, the average output current I_{out} is obtained by the sum of the average currents:

$$I_{\text{out}} = \sum_{x \in \mathcal{X}_{\mathcal{N}}} I_x. \tag{2}$$

To find f_x and t_x, we make the following choices:

1) The main function of the nodes is that of sensing environmental data and sending them to the sink (Section 8 describes how to account for event-driven WSNs).
2) At the channel access, we adopt a preamble-based transmitter-initiated MAC protocol, such as X-MAC (that exploits a Low-Power Listening strategy) [Buettner et al. 2006].
3) Network configuration and maintenance is managed via a distributed protocol, such as RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) [Winter et al. 2010].

From the first choice, we assume that the nodes periodically sense the environment and generate their data at a constant rate of f_{U} packets per second, where $t_{\text{U}} = 1/f_{\text{U}}$ is the average interpacket generation time (practical details on how to deal with nonperiodic traffic are provided in Section 8). Also, each data packet is assembled considering the data from $k_{\text{U}} \geq 1$ sensor readings; k_{U} can be used to account for additional processing of data and other operations that do not involve radio activity. Note that f_{U} is the nominal transmission rate, which is only obtained for a collision and error-free channel. In practice, given that multiple nodes share the same transmission medium, packets can be lost due to, for example, collisions or transmission errors. When taking some error recovery into account (retransmissions), the actual transmission rate will be $f'_{\text{U}} \geq f_{\text{U}}$.

For the routing, each node forwards its data packets either to the sink or to its next-hop node (referred to as its parent node). Also, each node sends its own information packets (this is referred to as *endogenous* traffic), as well as the packets generated by other nodes (*exogenous* traffic, in case the node acts as a relay for its children nodes).

To illustrate our network setting, we refer to the topology example of Figure 2, where the bottleneck node is represented as a black dot, while the sink is placed in the center of the network. In this figure, a possible realization of the routing tree is also shown. In particular, the links represented with solid lines belong to the subtree rooted at the bottleneck. White-filled dots indicate the nodes that use the bottleneck to forward their data to the sink (these are referred to as children nodes), while white triangles indicate the nodes whose traffic can interfere with that of the bottleneck (interfering nodes). Crosses indicate the position of all the other nodes.

For our model, we consider the topology, the data-gathering tree, and the coverage range as given. Also, we only track the number of children and interfering nodes, disregarding their actual position. Given this, next we refer to the following quantities as the input parameters for our analysis:

1) n_{c} is the number of children nodes, that is, the total number of nodes in the subtree rooted at the bottleneck. n_{c} governs the total traffic that has to be relayed by the bottleneck node.

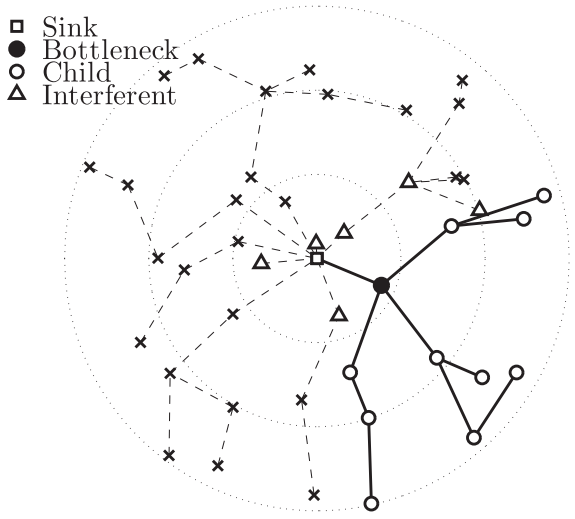


Fig. 2. Topology.

2) n_i is the number of interfering nodes (white triangles of Figure 2). These are within the transmission range of the bottleneck (i.e., within one hop from it), but the latter is not their intended next hop. Any transmission from one of these n_i nodes can be either a spurious reception or a collision for the bottleneck.
3) n_{int} corresponds to the total number of packets the bottleneck may be interfered from, that is, the sum of the traffic load (endogenous and exogenous) from all the interfering nodes. Note that in general, $n_{int} > n_i$.

Note that n_c especially depends on the size of the network in terms of number of communication hops, while n_i and n_{int} increase with the node density. Finally, in the analysis that follows, we assume that no node in the network has larger n_c, n_i, and n_{int} than the bottleneck node, and for each node but the bottleneck, at least one of the three parameters is strictly smaller than that of the bottleneck.

We are now ready to compute the various quantities needed to calculate Equation (2) for the bottleneck node. We start with states TX and RX. Note that packet transmissions and receptions depend on n_c. In fact, given that all the nodes generate a packet every t_U seconds (homogeneous network behavior), on average, the bottleneck will receive n_c packets from its children nodes and will transmit $n_c + 1$ packets (the exogenous traffic plus its own endogenous) every t_U seconds. This leads to

$$f_{TX,DG} = (1 + n_c)/t_U, \tag{3}$$

$$f_{RX,DG} = n_c/t_U, \tag{4}$$

where $f_{RX,DG}$ and $f_{RX,DG}$ are the data-gathering components of the transmission and reception frequencies, disregarding for the moment the traffic due to RPL.

To account for the impact of the MAC protocol, we summarize here its basic functionalities. The X-MAC LPL protocol specifies that each idling node periodically wakes up to perform a clear channel assessment (CCA) operation. The duty-cycle period lasts t_{dc} seconds and is composed of a sleeping phase of t_{off} seconds and a wake-up phase lasting t_{on} seconds, during which CCA is performed. A node wanting to send a unicast packet transmits a burst of short request-to-send (RTS) preambles for long enough so that the intended receiver will detect at least one of these RTSs in its next

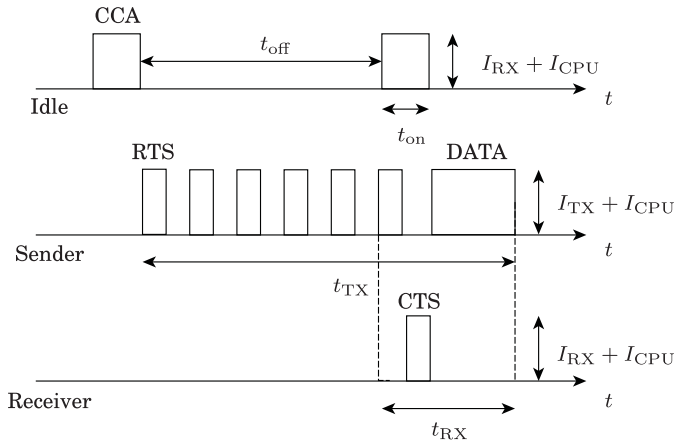


Fig. 3. MAC timings for CCA (top), TX (middle), and RX phases (bottom).

wake-up period. Since the nodes are in general not synchronized, to be sure of hitting the intended receiver, a node will be sending preambles for the duration of an entire duty cycle $t_{\text{dc}} = t_{\text{on}} + t_{\text{off}}$. Due to the lack of synchronization, the receiver can detect an RTS at any time within this period. Whenever a node detects an incoming RTS destined to itself, it sends a clear-to-send (CTS) message back to the sender and waits for the transmission of the actual data packet. After the complete reception of the data, the receiver sends an acknowledgment (ACK) to the sender. This channel access mechanism is illustrated in Figure 3 (where we omit the transmission of the ACK for simplicity). In this figure, the sixth RTS from the sender is detected by the intended receiver, which immediately replies with a CTS. The node at the top of the diagram also detects the RTS, but it does not take any action as it is not the intended destination.

For this channel access scheme, the average time needed to carry out a successful transmission is $t_{\text{TX}} = t_{\text{on}} + t_{\text{off}}/2 + t_{\text{cts}} + t_{\text{data}} + t_{\text{ack}}$, where the term $t_{\text{off}}/2$ follows from the fact that the time needed for the receiver to detect an incoming RTS is assumed to be uniformly distributed in $[0, t_{\text{off}}]$. The terms t_{data}, t_{cts}, and t_{ack} correspond to the durations associated with the transmission of a data packet, a CTS, and an ACK, respectively. The reception time is $t_{\text{RX}} = t_{\text{cts}} + t_{\text{data}} + t_{\text{ack}}$. Note that the RTS time is not considered in t_{TX} or in t_{RX}, because it is accounted for by the CCA state. Also, to simplify the notation, in the following analysis we include t_{cts} and t_{ack} in t_{data}.

Now, if $f_{\text{U}} = 1/t_{\text{U}}$ is the transmission rate (packets/second) for an error-free channel, in the presence of packet collisions and transmission errors, the actual transmission rate becomes $f'_{\text{U}} \geq f_{\text{U}}$. For the sake of clarity, the complete characterization of the channel access problem in this case is provided in Appendices A and B.

Thus, the average transmission time can be expressed as

$$t_{\text{TX}} = t_{\text{on}} + t_{\text{off}}/2 + t_{\text{data}} + (f'_{\text{U}}/f_{\text{U}} - 1)t_{\text{dc}}, \tag{5}$$

where the factor $f'_{\text{U}}/f_{\text{U}} - 1$ represents the average number of retransmissions. Note that Equation (5) implies a stop-and-wait retransmission policy, where an infinite number of retransmissions is allowed for each data packet. Instead, we assume that the impact of channel errors and collisions on spurious receptions and interfering packets is negligible as, in these cases, the intended receiver does not stay awake to receive the data packet and, thus, its energy expenditure is already accounted for by the CCA state.

We now model the energy expenditure associated with the maintenance of the routing topology. The selected routing algorithm, RPL, consists of a proactive technique that periodically disseminates network information through DODAG[1] information objects (DIO) and, subsequently, builds a routing tree by sending destination advertisement objects (DAOs) toward the sink. RPL timing is governed by the trickle timer, which exponentially increases up to a maximum value for a static topology. In this article, we analyze the steady-state phase of RPL, considering static networks. This implies the following operations: for every trickle timer epoch, which lasts t_{rpl} seconds, the bottleneck node must send its own DIO message and its own DAO and has to forward n_{c} DAOs for its children. This leads to a transmission frequency for RPL messages of

$$f_{\text{TX,RPL}} = (2 + n_{\text{c}})/t_{\text{rpl}}. \tag{6}$$

In addition, the bottleneck node will receive n_{c} DAOs from its children and n_{i} DIOs from its interfering nodes (note that DIOs are not treated as interference, as they are broadcast). Thus, the reception frequency for RPL messages is

$$f_{\text{RX,RPL}} = (1 + n_{\text{i}} + n_{\text{c}})/t_{\text{rpl}}, \tag{7}$$

where $f_{\text{TX,RPL}}$ and $f_{\text{TX,RPL}}$ are the contributions of RPL to the transmission and reception frequencies, respectively.

Finally, our model accounts for the energy expenditure due to the reception of messages that are detected during CCA but are not destined to the receiver. In this case, the receiver behaves as during a reception, but, as soon as it decodes the packet header, it recognizes that the message is not intended for itself. At this point, the node drops the message and goes back to sleep. Interfering messages can be due either to data gathering or to networking traffic and occur at a rate proportional to n_{int}. Thus, we have

$$f_{\text{INT}} = n_{\text{int}}(1/t_{\text{U}} + 1/t_{\text{rpl}}). \tag{8}$$

Also, we refer to $t_{\text{int}} < t_{\text{RX}}$ as the time needed to decode the packet header and therefore detect whether a node is the intended destination for that message.

From these reasonings, we are able to express the average current consumption for each state:

$$\begin{aligned} I_{\text{TX}} &= (i_{\text{c}} + i_{\text{t}})[t_{\text{dc}}/2 + t_{\text{on}}/2 + t_{\text{data}} + (f'_{\text{U}}/f_{\text{U}} - 1)t_{\text{dc}}] \\ &\quad \times [(1 + n_{\text{c}})/t_{\text{U}} + (2 + n_{\text{c}})/t_{\text{rpl}}] \end{aligned} \tag{9}$$

$$I_{\text{RX}} = (i_{\text{c}} + i_{\text{r}})t_{\text{data}}[n_{\text{c}}/t_{\text{U}} + (1 + n_{\text{c}} + n_{\text{i}})/t_{\text{rpl}}] \tag{10}$$

$$I_{\text{INT}} = (i_{\text{c}} + i_{\text{r}})t_{\text{int}}n_{\text{int}}(1/t_{\text{U}} + 1/t_{\text{rpl}}) \tag{11}$$

$$I_{\text{CPU}} = i_{\text{c}}t_{\text{cpu}}k_{\text{U}}/t_{\text{U}} \tag{12}$$

$$I_{\text{CCA}} = (i_{\text{c}} + i_{\text{r}})d_{\text{c}}\, r_{\text{IDLE}} \tag{13}$$

$$I_{\text{OFF}} = i_{\text{s}}(1 - d_{\text{c}})r_{\text{IDLE}}, \tag{14}$$

where t_{cpu} is the average time spent in operations that do not involve the radio and r_{IDLE} is the fraction of time that the node spends in the IDLE state, which is computed as one minus the fraction of time spent in the remaining states:

$$r_{\text{IDLE}} = 1 - r_{\text{TX}} - r_{\text{RX}} - r_{\text{INT}} - r_{\text{CPU}}. \tag{15}$$

[1]Destination-oriented directed acyclic graph (DODAG).

Table IV. Symbol Definitions

t_x^*	optimal values for the variable t_x.
x^{lim}	optimal values for the variable x, computed assuming no energy constraint.
$x^{\min}$	optimal values for the variable x, computed assuming zero reward (zero throughput).
$a_i, b_i, c_i, d_i, e_i, f_i$	coefficients. See Appendix C and Table X for their complete definition.
$u_{\max}$	maximum allowed energy consumption for a sensor node.
$u_{\min}$	minimum required energy consumption so that the system remains operational.
ρ	node density.

The total energy consumption is finally given by

$$I_{\text{out}} = I_{\text{TX}} + I_{\text{RX}} + I_{\text{INT}} + I_{\text{CPU}} + I_{\text{CCA}} + I_{\text{OFF}}. \tag{16}$$

4. NODE CONSUMPTION ANALYSIS

In this section, we present the solution of problem P1: identifying the optimal network's operating point given a target consumption $I_{\text{out}} = u$. The symbols used in this section are listed in Table IV.

Problem P1 can be formally written as follows:

Problem P1:

$$\begin{aligned} \underset{t_{\text{U}}, t_{\text{dc}}}{\text{maximize}} \quad & f_{\text{U}} \\ \text{subject to:} \quad & I_{\text{out}} \leq u, \\ & r_x \geq 0, \ \forall x \in \mathcal{X}_{\mathcal{N}}, \\ & t_{\text{U}} \geq 0, \ t_{\text{dc}} \geq t_{\text{on}}. \end{aligned} \tag{17}$$

P1 (Equation (17)) amounts to finding the optimal pair $(t_{\text{U}}^*, t_{\text{dc}}^*)$ that maximizes the node throughput, $f_{\text{U}} = 1/t_{\text{U}}$, subject to the maximum allowed consumption u and to time and frequency constraints. The problem can be numerically solved through two nested dichotomic searches (as shown in Bui and Rossi [2013]): the inner search looks for the optimal t_{off}^* given t_{U},[2] while the outer search looks for the optimal t_{U}^*. Instead, our objective here is to obtain the solution in closed form. This will permit us to solve problem P2 in a reasonable amount of time while also facilitating the implementation of optimal energy management policies on constrained sensor devices.

Despite the simple problem formulation, Equation (5) introduces a polynomial of the n_{i}th degree on the independent variable t_{U}, which makes it difficult to express the solution through tractable and still meaningful equations. Thus, we solve the problem for a collision-free channel and we subsequently adapt the results to keep collisions into account through a heuristic.

In fact, removing collisions allows for a simpler expression for f'_{U}, that is, $f'_{\text{U}} = f_{\text{U}}/(1 - e_t)$, which removes the n_{i}th degree polynomial on t_{U}. In order to illustrate that this approach is reasonable within the solution space, in Figure 4, we show some preliminary results.

Figure 4 shows contour lines in the (d_c, f_{U}) plane for different output current levels ($I_{\text{out}} \in \{5, 10, 30\}$ mA): dotted lines represent the numerical solution for the complete problem, while dash-dotted lines represent the solution for a collision-free channel for the same I_{out} levels. The locations of the optimal operating points in these two cases are also plotted for comparison (white squares and white circles for the complete problem and that without collisions, respectively). For a given I_{out}, the maximum throughput is achieved for a unique value of the duty cycle d_c. Hence, it is not possible to find

[2]Note that in this article, we consider t_{on} as a constant that depends on the considered sensor architecture, whereas the nodes can adapt the duration of their off phase, t_{off}, of the duty cycle. Hence, optimizing over $d_c = t_{\text{on}}/(t_{\text{on}} + t_{\text{off}})$, $t_{\text{dc}} = t_{\text{on}} + t_{\text{off}}$ or t_{off} is equivalent.

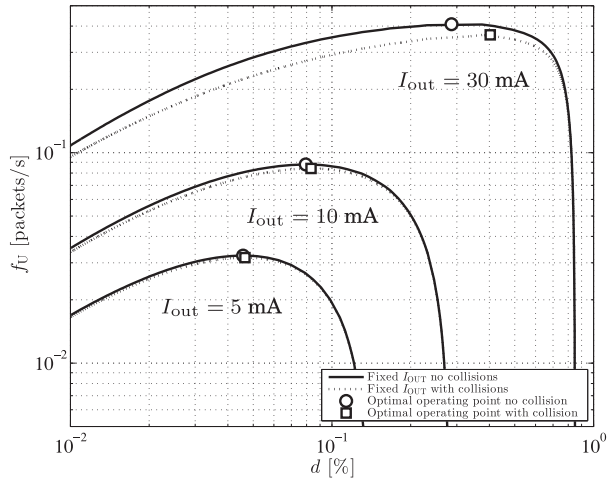


Fig. 4. Contour lines in the (d, f_{U}) plane for different output current levels ($I_{\text{out}} \in \{5, 10, 30\}$ mA): dotted lines represent the numerical solution to the complete problem Equation (17), while dash-dotted lines show the solution for a collision-free channel for the same I_{out} levels. The optimal working points are also plotted for both problems (using white squares for the complete problem and white circles to indicate the solution for a collision-free channel).

a feasible solution with higher throughput or one with the same throughput and a different duty cycle.

From Figure 4, we deduce the following facts:

—The impact of collisions increases with I_{out}, which implies that the difference between the optimal working points with and without collisions is an increasing function of the energy consumption I_{out}.
—The maximum allowed f_{U} increases with I_{out}, which is expected and means that the transmission rate for the endogenous data is an increasing function of the energy consumption I_{out}.
—The duty cycle d_c has a critical point, beyond which the throughput f_{U} suddenly drops, which implies that t_{dc} has a critical point too.
—The search for the optimal operating point involves the joint optimization of the transmission rate f_{U} (t_{U}) and the duty-cycle period (t_{dc}) as these two quantities are intertwined.

For the sake of readability, the full derivation of the closed-form solution in the collision-free case is given in Appendix C. In what follows, we confine ourselves to a discussion of the adopted approach and of the main results. First, I_{out} has been rewritten as a function of t_{dc} and t_{U}, which makes it possible to find the mathematical expression of t^*_{dc} (as a function of t_{U}, which is still a free parameter). This is achieved by taking the partial derivative of I_{out} with respect to t_{dc}, equating it to zero, and solving for t_{dc}. In doing so, we observe that $\partial I_{\text{RX}}/\partial t_{\text{dc}} = 0$, $\partial I_{\text{INT}}/\partial t_{\text{dc}} = 0$, and $\partial I_{\text{CPU}}/\partial t_{\text{dc}} = 0$, as they do not depend on t_{dc}. This leads to

$$\begin{aligned} \frac{\partial I_{\text{out}}(t_{\text{U}}, t_{\text{dc}})}{\partial t_{\text{dc}}} &= \frac{\partial}{\partial t_{\text{dc}}}(I_{\text{TX}}(t_{\text{U}}, t_{\text{dc}}) + I_{\text{CCA}}(t_{\text{U}}, t_{\text{dc}}) + I_{\text{OFF}}(t_{\text{U}}, t_{\text{dc}})) = 0 \\ &\Rightarrow\ t^*_{\text{dc}}(t_{\text{U}}) = \sqrt{\frac{d_6/t_{\text{U}} + d_5}{d_1/t_{\text{U}} + d_3}}, \end{aligned} \tag{18}$$

where coefficients d_1, d_3, d_5, and d_6 are given in Table X.

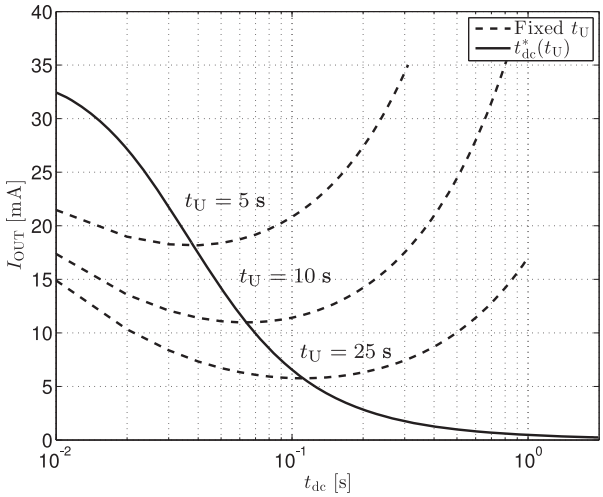


Fig. 5. Dashed lines represent $I_{out}(t_U, t_{dc})$ as a function of t_{dc}, considering a fixed interpacket transmission time $t_U \in \{5, 10, 25\}$ seconds. The locus of the optimal solutions t^*_{dc}, obtained through Equation (18), is plotted as a solid line.

To illustrate the behavior of Equation (18), in Figure 5 we show I_{out} by varying t_{dc} and keeping t_U fixed in the set $t_U \in \{5, 10, 25\}$ seconds (see dashed lines). The locus of the optimal solutions t^*_{dc}, obtained through Equation (18), is plotted as a solid line. The closed form for the optimal t_{dc} crosses I_{out} (without collisions) where the latter is minimized, as requested.

At this point, it is possible to replace t_{dc} with $t^*_{dc}(t_U)$ in $I_{out}(t_U, t_{dc})$ (see Equation (16)) expressing the output current as $I_{out}(t_U, t^*_{dc}(t_U))$, which becomes a function of the single independent variable t_U. Since f_U increases with I_{out}, the maximum achievable f_U for a given target current u is obtained at the equality point $I_{out}(t_U, t^*_{dc}(t_U)) = u$.

Also, u cannot be increased indefinitely, because, beyond a given threshold $t_U \leq t^{lim}_U$, the problem becomes bound by the frequency constraint $r_{IDLE} \geq 0$. In this region, the system drains the maximum current u_{max}, which cannot be further increased as the channel is saturated. t^{lim}_U is the smallest feasible interpacket transmission time for the considered system and can be analytically derived by observing that the optimality condition (see Equation (18)) and the frequency constraint $r_{IDLE}(t_U, t_{dc}) = 0$ must concurrently hold for $t_U = t^{lim}_U$. Thus, from $r_{IDLE}(t^{lim}_U, t_{dc}) = 0$, we obtain the relationship between t^{lim}_U and t_{dc}, that is, $t^{lim}_U(t_{dc}) = (a_1 t_{dc} + a_{11})/(a_{10} - a_3 t_{dc})$, whereas replacing t_U with t^{lim}_U in Equation (18) leads to $t^{lim}_{dc} = t^*_{dc}(t^{lim}_U)$. Using $t^{lim}_U(t^{lim}_{dc})$ in place of t^{lim}_U in the latter equation returns a third-order polynomial in the only variable t^{lim}_{dc}, which allows the calculation of t^{lim}_{dc} and, in turn, of t^{lim}_U. The coefficients $\{a_1, a_3, a_{10}, a_{11}\}$ are given in Table IX, whereas the involved mathematical derivations are detailed in Appendix C. Computing $I_{out}(t_U, t_{dc})$ for $(t^{lim}_U, t^{lim}_{dc})$ returns the maximum current that can be consumed by the bottleneck node using an optimal configuration, that is, $I^{lim}_{out} = I_{out}(t^{lim}_U, t^{lim}_{dc})$. The maximum control is therefore given by $u_{max} = I^{lim}_{out}$.

Conversely, there is a minimum current I^{min}_{out} that has to be drained in order to keep the system running and operational. I^{min}_{out} is found as $I^{min}_{out} = \lim_{t_U \to +\infty} I_{out}(t_U, t^*_{dc})$, which amounts to solely considering the energy consumption due to the periodic transmission of control traffic (taken into account through t_{rpl}). The minimum energy consumption also corresponds to the smallest control action $u_{min} = I^{min}_{out}$.

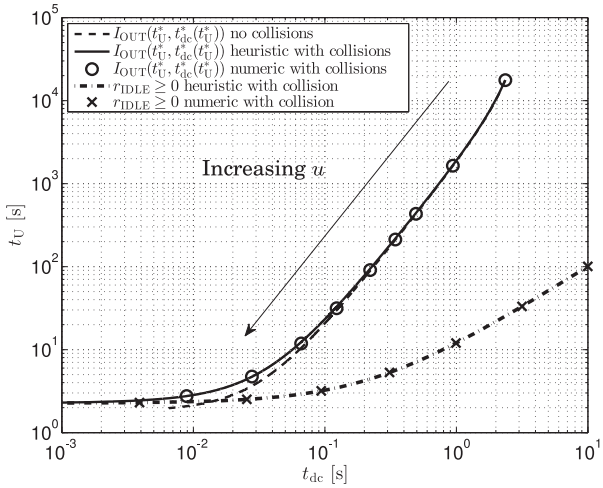


Fig. 6. Comparison between closed-form and exact solution of Equation (17). The dashed line shows the results obtained with the closed-form solution considering a collision-free channel, the dots represent the numerical solution of the problem with collisions, and the solid line corresponds to the closed-form solution, heuristically adapted to keep collisions into account. In addition, the constraint $r_{\mathrm{IDLE}}(t_{\mathrm{U}}, t_{\mathrm{dc}}) = 0$ is also shown (crosses indicate the exact bound obtained numerically, the dash-dotted line is obtained using the heuristically adapted closed-form).

Finally, the optimal working point, t_{U}^*, is found as the solution of $I_{\mathrm{out}}(t_{\mathrm{U}}, t_{\mathrm{dc}}^*(t_{\mathrm{U}})) = u$ with $u \in [u_{\min}, u_{\max}]$, which can be expressed as

$$(t_{\mathrm{U}}^*, t_{\mathrm{dc}}^*) = \begin{cases} (+\infty, \sqrt{d_5/d_3}) & \text{if } u < u_{\min} \\ (t_{\mathrm{U}}^*, t_{\mathrm{dc}}^*(t_{\mathrm{U}}^*)) & \text{if } u_{\min} \leq u \leq u_{\max} \\ (t_{\mathrm{U}}^{\lim}, t_{\mathrm{dc}}^{\lim}) & \text{if } u > u_{\max}, \end{cases} \tag{19}$$

where t_{U}^* is the positive solution of the quadratic equation $e_2 t_{\mathrm{U}}^2 + e_1 t_{\mathrm{U}} + e_0 = 0$ and $t_{\mathrm{dc}}^{\lim}$ is the largest solution of the cubic equation $f_3 t_{\mathrm{dc}}^3 + f_2 t_{\mathrm{dc}}^2 + f_1 t_{\mathrm{dc}} + f_0 = 0$. The reader is referred again to Appendix C for mathematical insights and the definition of the coefficients (see Table X).

Figure 6 shows the optimal operating point $(t_{\mathrm{U}}^*, t_{\mathrm{dc}}^*)$ by varying the control u as the independent parameter. The dashed line corresponds to the result of Equation (17) for a collision-free channel, the white-filled circles represent the numerical results of the complete problem with collisions, and the solid line shows the results achieved from the closed-form solution, which has been adapted through a heuristic to keep collisions into account. In addition, the crosses and the dash-dotted line illustrate the solution of $r_{\mathrm{IDLE}}(t_{\mathrm{U}}, t_{\mathrm{dc}}) = 0$ obtained for the complete problem and using the closed-form heuristically modified, respectively.

The adopted heuristic is a rigid translation of the closed form for a collision-free channel so that the latter equals the numerical solution with collisions for the maximum allowed control $u_{\max}$. The error introduced through this approach is very small for high values of u and increases for decreasing u. However, this error is negligible throughout most of the solution space, as it grows slower than t_{U} does and it always provides a feasible solution for the system.

Finally, in Figure 7, we plot the reward function:

$$r(u) = 1/t_{\mathrm{U}}^*(u). \tag{20}$$

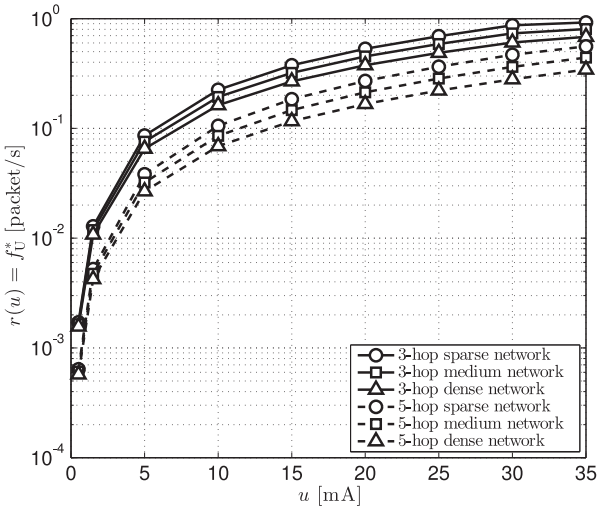


Fig. 7. Reward function $r(u)$ for different network topologies.

Table V. Network Parameters
R is the radio coverage range.

	N	ρ [nodes/R^2]	n_c [nodes]	n_i [nodes]	n_{int} [packets]
3-hop sparse	15	0.53	5	4	16
3-hop medium	25	0.88	5	8	32
3-hop dense	38	1.35	5	13	54
5-hop sparse	42	0.53	15	4	48
5-hop medium	68	0.86	15	8	96
5-hop dense	106	3.23	15	13	160

Table VI. System Parameters

t_{on}	t_{data}	t_{int}	t_{cpu}	t_{rpl}	i_t	i_r	i_c	i_s
6 ms	14 ms	10 ms	40 ms	6 h	14 mA	12.3 mA	42 mA	31 μA

$r(u)$ corresponds to the maximum achievable throughput for the given multihop network. In Figure 7, we show results for dense, medium, and sparse networks (represented with squares, circles, and triangles, respectively) of three and five hops (solid and dashed lines, respectively). The parameters of these networks are given in Table V, where N is the total number of nodes and ρ is the network density. Increasing the number of hops has a much larger impact on the reward function than increasing the node density. All the graphs of this article have been obtained considering a sensor platform characterized by the energy consumption and timing parameters of Table VI. The optimal throughput of Equation (20) will be used in Section 6 as the reward function for problem P2, which considers a stochastic energy source.

5. OPTIMIZATION FRAMEWORK

The objective of the following sections is to solve problem P2, which translates into finding *optimal* and *online* energy consumption strategies for the sensor nodes, given the energy consumption model (see problem P1), their current energy reserve, and a statistical characterization of future energy arrivals (i.e., of the energy source S). This requires one to link the energy consumed to that harvested and to the instantaneous energy buffer state. In the analysis that follows, we assume that the amount of charge in

Table VII. Symbols Used in the Energy Source Model

$x_s \in \mathcal{S}$	energy source state x_s and the set of all energy states, $\mathcal{S}$.
t_k, Δ_k	transition time t_k and epoch duration Δ_k.
τ_{x_s}, $f_\tau(t\|x_s)$	r.v. and pdf describing the permanence time in state x_s.
ι_{x_s}, $f_\iota(i\|x_s)$	r.v. and pdf describing the current harvested in state x_s.
p_{ij}	transition probabilities of the source model's embedded Markov chain.
$\delta = \delta_{\text{in}} - \delta_{\text{out}}$	r.v.s describing the total variation (δ), the harvested (δ_{in}), and the consumed (δ_{out}) charge in a decision epoch.
$f_\delta(d\|u, x_s)$	pdf of the variation of charge in state x_s when the control is u.

the energy buffer is a known quantity or, equivalently, that it can be reliably estimated at the sensor nodes. Based on this, we formulate our optimal control as a Markov Decision Process (MDP). We observe that heuristic approaches, which base their energy consumption policies on energy estimates, are also possible but are not considered here and are left as a future work. Nevertheless, in Section 7.2, the performance of the obtained policies is compared against that of heuristic solutions from the literature.

Here, we present the stochastic model that will be used to describe the source S, as per our sensor diagram of Figure 1. This will be used in Section 6 to solve problem P2. The resulting energy management policies are validated in Section 7.

In Table VII, we define the symbols used in this section.

Energy source: The energy source dynamics are captured by a continuous-time Markov chain with N_S states $x_s \in \mathcal{S} = \{0, 1, \ldots, N_S - 1\}$. We refer to t_k, with $k \geq 0$, as the time instant where the source transitions between states and to $\Delta_k = t_k - t_{k-1}$ as the time elapsed between two subsequent transitions. Also, the system between t_{k-1} and t_k is said to be in *stage* k, and its duration Δ_k is described by an r.v. $\tau_{x_s} \in [t_{\min}(x_s), t_{\max}(x_s)]$, depending on the source state x_s in the stage. τ_{x_s} has an associated probability distribution function (pdf) $f_\tau(t|x_s)$. Moreover, during stage k, the source provides a constant current i_k that is fed into the battery and is assumed to remain constant until the next transition, occurring at time t_k. This input current is described by the r.v. $\iota_{x_s} \in [i_{\min}(x_s), i_{\max}(x_s)]$ with pdf $f_\iota(i|x_s)$. We assume that τ_{x_s} and ι_{x_s} have bounded support. $p_{ij} = \text{Prob}\{x_s(k) = j | x_s(k-1) = i\}$ with $i, j \in \mathcal{S}$ are the transition probabilities of the associated embedded Markov chain, which are invariant with respect to k.

Discrete-time formulation: We describe the energy source model through an equivalent discrete-time Markov process. This will make it possible to conveniently characterize the optimal policies through a Discrete-Time Constrained Markov Decision Process (DT-CMDP) in Section 6. For improved clarity of exposition and conciseness, in the remainder of this article we omit the time index k from the symbols, unless explicitly stated otherwise.

To describe the energy source through a discrete-time model, for any given k, we map the random nature of the stage duration into the corresponding variation of charge during the stage. To do this, we define the two r.v.s δ_{in} and δ_{out} that respectively describe the amount of charge that enters the system during the stage (stored in the energy buffer) and the amount of charge consumed by the sensor node. $\delta = \delta_{\text{in}} - \delta_{\text{out}}$ is the r.v. describing the overall variation of charge during the stage. We recall that u is our control variable, corresponding to the current drained by the sensor node during the stage. u for a given policy is a known quantity and it will be considered as a constant in the following derivations. We have that

$$\delta_{\text{in}} = \tau\iota, \ \delta_{\text{out}} = \tau u, \ \delta = \delta_{\text{in}} - \delta_{\text{out}} = \tau(\iota - u). \tag{21}$$

Hence, the r.v. δ is obtained as the product of the two r.v.s τ and $\iota - u$. From the theory in Papoulis and Pillai [2002], the pdf of δ when the source is in state x_s and the control

Table VIII. Symbols Used in the MDP Analysis

$x_b \in \mathcal{B} = [0, b_{\max}]$	buffer state x_b, buffer state set $\mathcal{B}$, and buffer size $b_{\max}$.
$x = (x_s, x_b) \in \mathcal{X} = \mathcal{S} \times \mathcal{B}$	system state x in the current decision epoch, system state set $\mathcal{X}$, source state set $\mathcal{S}$, and buffer state set $\mathcal{B}$.
$y = (y_s, y_b) \in \mathcal{X}$	system state in the next decision epoch.
$u \in \mathcal{U} = [u_{\min}, u_{\max}]$	action (control) u and action set $\mathcal{U}$.
π, μ	policy π and mapping μ between states x and actions u.
$r(u)$	reward associated with action u.
$R(x,u)$, $C(x,u)$	single-stage expected reward $R(x,u)$ and cost $C(x,u)$.
$J_R(x)$, $J_C(x)$	optimal expected reward $J_R(x)$ and cost $J_C(x)$.
C_{th}	threshold on the cost for the admissibility of the solution.
α	discount factor.
λ, $L_\lambda(x,u)$	Lagrangian multiplier λ and Lagrangian reward $L_\lambda(x,u)$.

is u, $f_\delta(d|u, x_s)$, is obtained as:

$$f_\delta(d|u, x_s) = \int_{t_{\min}(x_s)}^{t_{\max}(x_s)} f_\tau(t|x_s) f_\iota(d/t + u|x_s)|t|^{-1} \mathrm{d}t \ , \ d \in \mathbb{R}. \tag{22}$$

Henceforth, the energy source is equivalently characterized by a discrete-time Markov chain with N_S states and transition probabilities p_{ij}, $i, j \in \mathcal{S}$. Moreover, when the current state is $x_s \in \mathcal{S}$ and the control is u, the corresponding variation of charge during a stage is accounted for by the r.v. δ with pdf given by Equation (22).

6. MARKOV DECISION PROCESS ANALYSIS

This section presents our analysis of the outer optimization problem P2, which is framed as a Markov Decision Process. For improved clarity, this analysis is split into four subsections: in Section 6.1, we define the basic ingredients of the MDP; in Section 6.2, we formulate the optimal policy, discussing its properties and detailing an algorithm for its computation (see Section 6.3). Finally, in Section 6.4, we report our considerations on computational complexity and on the usage model for the computed policies. The list of symbols used in the MDP analysis is given in Table VIII.

6.1. Definitions

We consider the sensor system of Figure 1 and we assume without loss of generality that the system evolves in discrete time. Hereafter, at time $k \geq 0$, the system is said to be in stage k and the terms "time" and "stage" will be used interchangeably in the following analysis. The source S feeds energy into the energy buffer B and is modeled according to the discrete-time Markov chain presented in the previous section. At any time k, the source S is in a certain state x_s, whereas the energy buffer hosts an amount of charge $x_b \in \mathcal{B} = [0, b_{\max}]$, where $b_{\max}$ is the buffer capacity. At the generic time k, we define the system state as $x = (x_s, x_b) \in \mathcal{X}$, where $\mathcal{X} = \mathcal{S} \times \mathcal{B}$. The system state at the following time $k+1$, defined as $y = (y_s, y_b) \in \mathcal{X}$, depends on the dynamics of S, on the control u for the current stage k, and on the total variation of charge δ during stage k. For the battery at the beginning of the next stage $k+1$, y_b, we have

$$y_b = \min\{\max\{x_b + \delta, 0\}, b_{\max}\} = [x_b + \delta]^+, \tag{23}$$

where δ is expressed in Equation (21) and depends on the control u for the current stage k, whereas $[a]^+$ is defined as $[a]^+ = \min\{\max\{a, 0\}, b_{\max}\}$, with $a \in \mathbb{R}$.

We model the sensor system through a discrete-time MDP. At every stage k, a decision u has to be made based on the current state $x \in \mathcal{X}$. In addition to the system state and its dynamics, a Markov decision process is characterized by a control set $\mathcal{U} = [u_{\min}, u_{\max}]$, where $u_{\min} = I_{\text{out}}^{\min}$ and $u_{\max} = I_{\text{out}}^{\lim}$. $\mathcal{U}$ contains all the feasible current consumption

levels for the sensor (see Section 4). In this article, we consider mixed and stationary Markov (i.e., history-independent) policies. The term mixed means that there exists a mapping μ that, for any possible state $x \in \mathcal{X}$, returns a vector of pairs $(u(i), p(i))$, of size $M \geq 1$, with $\sum_{i=1}^{M} p(i) = 1$. This vector represents the decision to be made when the system state is x and indicates that control $u(i)$ must be implemented with the associated probability $p(i)$. A mixed *policy* π is a collection of such mappings $\pi = \{\mu_0, \mu_1, \mu_2, \dots\}$ for all stages. Our problem belongs to the class of MDPs with unichain structure, bounded costs, and rewards. For these, it is sufficient to consider the set of admissible Markov policies as the optimal policy can always be found within this class; see Derman and Strauch [1966] and Altman [1999], or Theorem 13.2 of Feinberg and Shwartz [1995]. The boundedness of rewards and costs follows from the finite support of τ, ι, and from the fact that the instantaneous reward function is also bounded. Thus, for the problem addressed in this article, it is sufficient to restrict our attention to Markov stationary policies, which means that μ_k only depends on the system state at time k (past stages $0, \dots, k-1$ are not considered) and that the mapping functions do not depend on k, that is, $\pi = \{\mu, \mu, \mu, \dots\}$.

Reward: The reward function takes into account the throughput of the system. Specifically, from the derivations in Section 4, we know that for a given control u the optimal instantaneous throughput of a sensor node is given by $r(u)$, as defined in Equation (20). Now, let $x = (x_b, x_s)$, with $x \in \mathcal{X}$, be the system state at the beginning of a generic decision stage k. Moreover, let t and i respectively represent the realization of the r.v. τ_{x_s}, describing the duration of the stage, and the realization of the r.v. ι_{x_s}, quantifying the input current from the source. Taking Equation (21) into account and recalling that the input current i and the control u are both constant during the stage, we have that the amount of charge varies linearly within a stage until it either hits the buffer capacity $b_{\max}$ or drops to 0, depending on the sign of $i - u$. Hence, during the stage, the total variation of charge is $d = t(i - u)$ (see Equation (21)) and the amount of time the level of charge in the energy buffer is greater than zero is given by the following function:

$$g_{>0}(d, t, u, x_b) = \begin{cases} t & d \geq 0 \\ \min\left\{\dfrac{-x_b t}{d}, t\right\} & d < 0. \end{cases} \tag{24}$$

Furthermore, as long as the buffer level is above zero, the throughput remains constant and equal to $r(u)$, whereas it drops to zero in case the energy buffer gets empty. Given this, the single-stage expected reward, when the system state at the beginning of the stage is $x = (x_b, x_s)$ and the control is u, is computed as

$$\begin{aligned} R(x, u) &= \mathrm{E}[r(u) g_{>0}(\xi, t, u, x_b) | x, u] \\ &= \int_{-\infty}^{+\infty} \int_{t_{\min}(x_s)}^{t_{\max}(x_s)} r(u) g_{>0}(\xi, t, u, x_b) f_\tau(t|x_s) f_\iota(\xi/t + u|x_s) |t|^{-1} \mathrm{d}t \mathrm{d}\xi \\ &= r(u)\, \mathrm{E}[g_{>0}(d, t, u, x_b) | x, u], \end{aligned} \tag{25}$$

where $\mathrm{E}[g_{>0}(d, t, u, x_b)]$ represents the average amount of time the energy buffer contains a positive amount of charge during the stage. In the previous equation, $r(u)$ remains constant during a stage when u is given. The actual average throughput is then modulated through the average amount of time the energy buffer state is greater than zero in the stage, that is, $\mathrm{E}[g_{>0}(d, t, u, x_b) | x, u]$.

Cost: For the cost, we account for a penalty whenever the energy buffer drops below a given threshold $b_{\text{th}} \in (0, b_{\max}]$. This threshold is a design parameter that may be related to the minimum energy reserve that is required to keep the system operational and

responsive. Also, b_{th} is in general implementation dependent, and besides depending on application requirements, it depends on hardware constraints. In fact, too low a charge may not be sufficient to guarantee the correct operation of the sensor nodes.

The cost is obtained as the average time spent with the energy buffer level below b_{th}. The amount of time the energy buffer level is below b_{th} is given by the following function:

$$g_{<b_{\text{th}}}(d,t,u,x_b) = \begin{cases} \max\left\{0, \min\left\{\dfrac{(b_{\text{th}}-x_b)t}{d}, t\right\}\right\} & d \geq 0 \\ \min\left\{\max\left\{0, \left(1-\dfrac{(b_{\text{th}}-x_b)}{d}\right)t\right\}, t\right\} & d < 0. \end{cases} \tag{26}$$

Hence, the single-stage expected cost when the system state at the beginning of the stage is $x = (x_b, x_s)$ and the control is u is obtained as

$$\begin{aligned} C(x,u) &= \mathrm{E}[g_{<b_{\text{th}}}(\xi,t,u,x_b)|x,u] \\ &= \int_{-\infty}^{+\infty}\int_{t_{\min}(x_s)}^{t_{\max}(x_s)} g_{<b_{\text{th}}}(\xi,t,u,x_b) f_\tau(t|x_s) f_\iota(\xi/t+u|x_s)|t|^{-1}\mathrm{d}t\mathrm{d}\xi. \end{aligned} \tag{27}$$

6.2. Optimal Policy: Formulation

We now formulate our optimal control problem as a DT-CMDP. The total expected reward that is earned over an infinite horizon by a feasible policy π is expressed as

$$J_R(x_o) = \lim_{N\to+\infty} \mathrm{E}\left[\sum_{k=0}^{N-1} \alpha^k R(x(k),u(k)) \middle| x(0)=x_o, \pi\right], \tag{28}$$

where $\alpha \in [0,1)$ is the discount factor, $x(k)$ and $u(k)$ are respectively the system state and the control at stage k, and x_o is the initial state. If we disregard the cost, having the sole objective of maximizing the throughput (reward), the optimal policy is the one that solves the following Bellman optimality equation:

$$\begin{aligned} J_R(x) &= \max_{u\in\mathcal{U}}\left\{R(x,u) + \alpha\sum_{y_s\in\mathcal{S}} p_{x_s y_s}\int_{-\infty}^{+\infty} f_\delta(\xi|u,x_s) J_R(y)\mathrm{d}\xi\right\}, \\ \text{with: } y &= (y_b, y_s),\ y_b = [x_b+\xi]^+, \end{aligned} \tag{29}$$

where if the current state is x, $J_R(x)$ represents the optimal expected reward from the current stage onward and is obtained, maximizing over the admissible controls, by the sum of the single-stage expected reward (the immediate reward, accrued in the present stage) and the expected optimal reward from the next stage onward (where future rewards $J_R(y)$ are weighted accounting for the system dynamics, i.e., $f_\delta(\cdot)$ and $p_{x_s y_s}$). Equation (29) can be solved through Value Iteration (VI), as detailed in Section 1.3.1 of Bertsekas [2012]. In short, VI amounts to using Equation (29) as an update rule, which is iterated for all states starting from an initial estimate of $J_R(x)$.[3] It can be shown that the optimality equation $J_R(x)$ is a *contraction mapping*. This property ensures that the VI iterations converge, at which point the optimal estimates $J_R(x)$ computed in the previous step equal the new ones, which are obtained using the right-hand side (RHS) of Equation (29). Hence, the optimal policy, for any given $x \in X$, is given by the control u that maximizes the RHS of Equation (29). Note that the optimal control corresponding to Equation (29) is a *pure policy* whereby a single control u is associated with each state

[3]Setting $J_R(x) = 0, \forall x$ in the first iteration of the algorithm also ensures convergence.

$x \in \mathcal{X}$; that is, there exists a mapping function $\mu(x)$ such that $u(x) = \mu(x)$ for each state $x \in \mathcal{X}$ and $u(x)$ is unique for each x.

Analogously, solely taking the cost into account, the total expected and discounted cost of a given policy π for an initial state x is obtained as the solution of the following Bellman equation:

$$J_C(x) = \max_{u \in \mathcal{U}} \left\{ C(x,u) + \alpha \sum_{y_s \in \mathcal{S}} p_{x_s y_s} \int_{-\infty}^{+\infty} f_\delta(\xi | u, x_s) J_C(y) \mathrm{d}\xi \right\},$$
$$\text{with: } y = (y_b, y_s),\ y_b = [x_b + \xi]^+. \tag{30}$$

The DT-CMDP problem for our controlled sensor node is thus written as follows:

Problem P2:

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & \mathrm{E}_x[J_R(x)|\pi] \\ \text{subject to:} \quad & \mathrm{E}_x[J_C(x)|\pi] \leq C_{\text{th}}, \end{aligned} \tag{31}$$

where the maximization is taken over the set of all feasible policies and $\mathrm{E}_x[\cdot]$ represents the expectation taken with respect to the steady-state distribution of $x \in \mathcal{X}$ induced by policy π. C_{th} is a positive constant and a policy is termed *feasible* if its average cost satisfies the constraint of Equation (31). For the selection of C_{th}, note that, as shown in Altman [1999], the average cost per stage corresponding to a total expected cost C_{th} and a discount factor α is obtained as $C'_{\text{th}} = C_{\text{th}}(1-\alpha)$. Moreover, from the definition of the cost (see Equation (26)), this quantity corresponds to the average amount of time in a stage where the amount of charge in the energy buffer is below b_{th}. Thus, dividing C'_{th} by the average stage duration, $T = \mathrm{E}[\tau_{x_s}]$, returns the maximum tolerable fraction of time in a stage during which the amount of charge in the energy buffer can be smaller than b_{th}; that is, a buffer outage occurs. Thus, the average fraction of time in a stage that the buffer is in outage is found as

$$t_{\text{out}} = \frac{C_{\text{th}}(1-\alpha)}{T}. \tag{32}$$

This relation facilitates the tuning of C_{th}, associating it to a tangible concept.

The inequality constraint in Equation (31) limits the maximum energy consumption by imposing a maximum expected cost C_{th}. The optimal policy is thus tunable through α and C_{th}. The former determines how much we look ahead in the optimization; for instance, $\alpha = 0$ represents a *myopic* decision maker where the control is uniquely chosen based on the current stage and the future system evolution is disregarded. Higher values of α generate optimal policies with better look-ahead capabilities. In particular, as $\alpha \to 1$, the associated optimal policies converge to the policy that maximizes the average reward over an infinite time horizon; see White [1993]. Instead, decreasing C_{th} generates less aggressive policies, which will be more parsimonious in the consumption of the energy stored in the buffer.

6.3. Optimal Policy: Computation

From the analysis in Beutlerand and Ross [1985] (Theorem 4.3) and Altman [1999] (Theorem 12.7), we know that Equation (31) can be solved through the definition of a Lagrangian reward $L_\lambda(x,u)$ (referred to as *Lagrangian relaxation*):

$$L_\lambda(x,u) = R(x,u) - \lambda C(x,u), \tag{33}$$

where $\lambda \geq 0$ is the Lagrangian, whereas $R(x,u)$ and $C(x,u)$ are respectively defined in Equations (25) and (27). Thus, we define an unconstrained discounted problem

depending on λ and having the following Bellman optimality equation:

$$\begin{aligned} J_\lambda(x) &= \max_{u\in\mathcal{U}}\{Q(x,u,\lambda)\},\\ \text{with:}\quad & Q(x,u,\lambda) \stackrel{\text{def}}{=} L_\lambda(x,u) + \alpha \sum_{y_s\in\mathcal{S}} p_{x_s y_s} \int_{-\infty}^{+\infty} f_\delta(\xi|u,x_s) J_\lambda(y)\mathrm{d}\xi,\\ \text{and:}\quad & y=(y_b,y_s),\ y_b=[x_b+\xi]^+. \end{aligned} \tag{34}$$

For a fixed λ, Equation (34) represents a standard discrete-time Markov Decision problem and can be solved through VI obtaining the corresponding pure optimal policy π_λ. For a given λ, the function $J_\lambda(x)$ returns the optimal Lagrangian reward associated with the optimal policy π_λ. We denote the expected log-term Lagrangian reward of this optimal policy by $J_\lambda = \mathrm{E}_x[J_\lambda(x)|\pi_\lambda]$, where the expectation is taken over the steady-state distribution of x induced by the optimal policy π_λ.

Intuitively, considering Equation (33), one can easily see that an increasing λ puts more weight on the cost $C(x,u)$, making the policy more conservative, while a smaller λ will instead put more weight on the reward $R(x,u)$, giving a higher priority to the throughput. These facts are used in the algorithm later to exploit λ to search within the solution space. The optimal λ is the one that achieves the maximum reward while leading to an average cost smaller than or equal to C_{th}; see Equation (31).

Next, we propose an efficient algorithm that exploits a dichotomic search over λ. Note that this search strategy is possible because, as proven in Lemmas 3.1 and 3.2 of Beutlerand and Ross [1985], for our discounted MDP, the optimal Lagrangian reward $J_\lambda(x)$ is a uniformly absolutely continuous, monotone, and nonincreasing function of λ. This means that the reward $J_\lambda(x)$ is well behaved as a function of λ; that is, it does not have local minima or maxima.

Moreover, the results in Beutlerand and Ross [1985] (see Theorems 4.3 and 4.4) guide us in the search for the optimal λ. In fact, for the optimal policy, there can only be the following two possibilities: (1) an optimal λ, termed λ^*, exists such that the average cost of $\pi_{\lambda*}$ is equal to C_{th}, and in this case, $\pi_{\lambda*}$ is the optimal policy that we are looking for and belongs to the class of pure policies; or (2) there exist two values of λ, say, λ^- and λ^+ with $\lambda^- < \lambda^+$, for which the cost of π_{λ^-} is larger than C_{th}, whereas that of π_{λ^+} is smaller than C_{th}, and the two policies differ in at most one state and the optimal policy we are looking for is a mixed policy that consists of using, at every decision epoch, π_{λ^-} with a certain probability p and π_{λ^+} with probability $1-p$. Case 2 is always verified, even when a pure policy exists, whereas a pure policy may or may not exist, depending on the structure of the MDP.

Given this, our algorithm seeks a mixed policy that maximizes the total expected Lagrangian reward $J_\lambda = \mathrm{E}_x[J_\lambda(x)|\pi_\lambda]$ while satisfying the constraint $\mathrm{E}_x[J_C(x)|\pi_\lambda] \le C_{\text{th}}$, where we define $C_\lambda = \mathrm{E}_x[J_C(x)|\pi_\lambda]$. The algorithm is described next:

(1) Pick the initial values for λ^- and λ^+, where λ^+ is a small value for which $C_{\lambda^-} > C_{\text{th}}$ and C_{λ^+} is such that $C_{\lambda^+} < C_{\text{th}}$.
(2) Compute $\lambda = (\lambda^+ + \lambda^-)/2$ and apply VI to Equation (34) for this λ. This returns the optimal Lagrangian reward function $J_\lambda(x)$ ($\forall x \in \mathcal{X}$), which is the unique solution of Equation (34). Once $J_\lambda(x)$ is known, the associated optimal policy π_λ is described by the mapping $u(x) = \mu_\lambda(x)$, where:

$$\mu_\lambda(x) = \operatorname*{argmax}_{u\in\mathcal{U}}\{Q(x,u,\lambda)\}, \tag{35}$$

where $Q(x,u,\lambda)$ is defined in the second line of Equation (34).

(3) Obtain the stationary distribution of x induced by π_λ, referred to as $P(x)$, which is computed by numerically solving the recursion

$$P(y) = \int_{x \in \mathcal{X}} P(x) f(y|x, u(x)) \mathrm{d}x \tag{36}$$

under the constraint $\int_{x \in \mathcal{X}} P(x)\mathrm{d}x = 1$, where $P(x)$ represents the steady-state distribution evaluated in state $x = (x_b, x_s) \in \mathcal{X}$, whereas $f(y|x, u(x))$ is the conditional probability distribution function that the system moves to $y = (y_b, y_s) \in \mathcal{X}$ at the end of a given stage, given that the initial state is x and the action taken is $u(x) = \mu_\lambda(x)$. For our problem, Equation (36) specializes to

$$P(y) = \sum_{x_s \in \mathcal{S}} p_{x_s y_s} \int_{x_b \in \mathcal{B}} P(x) \int_{I(x_b, y_b)} f_\delta(\xi \,|\mu_\lambda(x), x_s) \mathrm{d}\xi \, \mathrm{d}x_b, \tag{37}$$

where $x = (x_b, x_s)$, $y = (y_b, y_s)$ and $I(x_b, y_b) = \{y_b - x_b\}$ if $y_b > 0$ and $b < b_{\max}$, whereas $I(x_b, y_b) = [y_b - x_b, +\infty)$ if $y_b = b_{\max}$ and $I(x_b, y_b) = [-\infty, y_b - x_b]$ if $y_b = 0$.

(4) At this point, the average long-term cost performance $J_C(x)$ associated with policy π_λ is obtained by solving Equation (30) through VI, where $\max_{u \in \mathcal{U}}$ is replaced with $\max_{u \in \{\mu_\lambda(x)\}}$, which means that the single optimal action $\mu_\lambda(x)$ is used in place of set $\mathcal{U}$, so the maximization reduces to the evaluation of the RHS of Equation (30) for the optimal action only. Now, using $P(x)$ and $J_C(x)$, we obtain the expected long-term discounted cost C_λ as

$$C_\lambda = \mathrm{E}_x[J_C(x)|\pi_\lambda] = \int_{x \in \mathcal{X}} P(x) J_C(x) \mathrm{d}x. \tag{38}$$

(5) Now, we can have three cases: (C1) $C_\lambda = C_{\mathrm{th}}$, (C2) $C_\lambda < C_{\mathrm{th}}$, or (C3) $C_\lambda > C_{\mathrm{th}}$. In case C1, the algorithm terminates and the optimal policy is the pure policy π_λ. Otherwise, the algorithm continues as follows. In case C2, we update λ^+ as $\lambda^+ = \lambda$, whereas in case C3, we set $\lambda^- = \lambda$ and we initiate a new iteration, going back to step (2) earlier, using the new values of λ^- and λ^+ (which represent our dynamically adapted search interval). If, instead, the difference between C_{λ^-} and C_{λ^+} is smaller than a small constant $\varepsilon > 0$, the algorithm stops returning π_{λ^-}, π_{λ^+}, and the value of the mixing probability p, which is obtained as follows:

$$pC_{\lambda^-} + (1-p)C_{\lambda^+} = C_{\mathrm{th}} \;\Rightarrow\; p = \frac{C_{\mathrm{th}} - C_{\lambda^+}}{C_{\lambda^-} - C_{\lambda^+}}. \tag{39}$$

Hence, the optimal policy that solves Equation (31) is a mixed policy that, at the beginning of each stage, uses policy π_{λ^-} with probability p and policy π_{λ^+} with probability $1 - p$.

6.4. Optimal Policy: Complexity and Usage

Let ε be the desired numerical precision. The number of iterations involved in the dichotomic search for the optimal λ is $O(\log_2(\lambda_{\max}/\varepsilon))$, where $\lambda_{\max}$ is the upper end of the related search interval. A tight upper bound on the complexity of the *value iteration algorithm* (see Equation (35)), which is executed once for each value of λ, is $O(1/((1-\alpha)^2\varepsilon)^{2n+m})$, where in our case $n = 2$ and $m = 1$; see Chow and Tsitsiklis [1989]. The complexities associated with solving Equations (36) and (38) are dominated by that of value iteration.

In general, the proposed algorithm substantially reduces the complexity associated with finding the optimal policy, which would be infeasible through an exhaustive search. Although the computational complexity is rather high, the energy management policies neither have to be computed at runtime nor have to be obtained by the sensor nodes.

Instead, we propose the following. First of all, for the considered location, time of year, and type of solar module, we must derive an energy source model according to the procedure described in Section 5 (see also Miozzo et al. [2014]). This model must then be used with the algorithm of Section 6.3 to obtain *online* optimal energy management policies for the considered settings. This algorithm is executed offline and only once for a given source model. The resulting policies will correspond to simple tables associating the (quantized) amount of charge in the battery with a corresponding optimal action (control u). Note that they can be conveniently stored in memory arrays, preloaded into the nodes' memory, and looked up in $O(1)$ time. Also, note that the policies will be nondecreasing piecewise linear functions of the battery state x_b and, as such, a further compression of the memory required for their storage is possible through numerical fitting.

The sensor nodes will only have to execute at runtime the action dictated by the current policy, which corresponds to retrieving the optimal action from the policy table.

7. NUMERICAL RESULTS

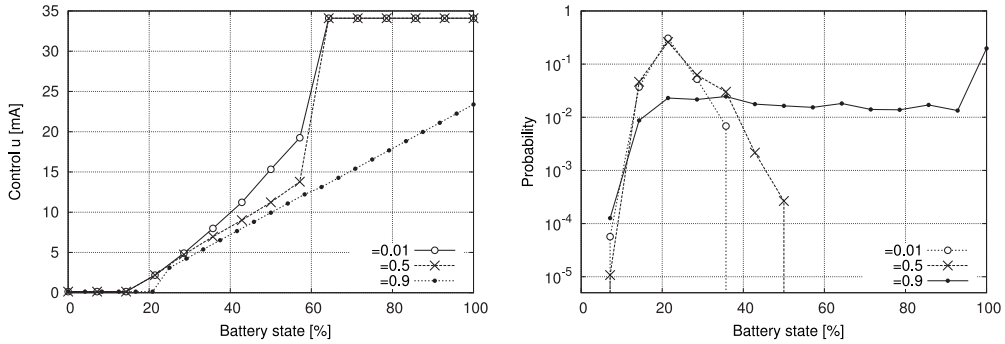
In this section, we comment on the numerical results of the solution of the combined optimization described in Section 2, which includes P1, which finds a suitable reward function $r(u)$ (throughput as a function of the energy consumption u), and P2, which uses $r(u)$ to obtain optimal online energy management policies that maximize the throughput while keeping the bottleneck node (and, as a consequence, all other nodes in the network) energetically self-sufficient. In particular, Section 7.1 discusses the general behavior of the optimal policies; Section 7.2 provides simulation results on their throughput and outage time, comparing our solution with that proposed in Kansal et al. [2007]; Section 8.1 discusses the robustness of our solution when the amount of charge harvested by the sensor nodes differ; and Section 8 considers the relaxation of further assumptions.

Network setup: In the following subsections, we consider a network of $N = 48$ sensor nodes that transmit their data to a sink through a multihop topology of five hops. Problem P1 for this network has been solved in Section 4, where it is referred to as "five-hop medium-network." The corresponding reward function $r(u)$ is plotted in Figure 7 and the corresponding network parameters are given in Table V.

For the energy inflow, we have considered a photovoltaic outdoor power source, adopting the framework of Miozzo et al. [2014] with two states $x_s \in \mathcal{S} = \{0, 1\}$, where $x_s = 0$ is the high-energy state (i.e., modeling daytime), and $x_s = 1$ is a state where the energy harvested is nearly zero (night). For the transition probabilities, we have $p_{ij} = 1$ if $i \neq j$ and $p_{ij} = 0$ if $i = j$ with $i, j \in \mathcal{S}$. The probability distribution functions $f_\iota(i|x_s)$ and $f_\tau(t|x_s)$ are derived using the SolarStat tool as detailed in Miozzo et al. [2014] using their "night-day clustering approach." We have selected Los Angeles as the installation location, considering a tilt of 45° and an azimuthal displacement with respect to the real South of 30° for the solar panels (Solarbotics SCC-3733 Monocrystalline solar technology [Solarbotics Ltd. 2013]). Irradiance data from years 1999–2010 available at National Renewable Energy Laboratory [2013] have been employed for the calculation of $f_\iota(i|x_s)$ and $f_\tau(t|x_s)$.

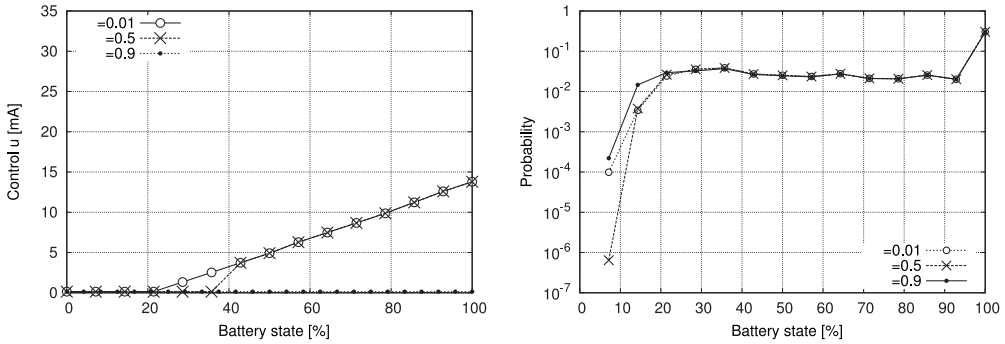
For the characterization of the optimal policies, we have considered square solar panels with sides going from 3 to 12 centimeters (in steps of 1 centimeter), considering the irradiance data collected for the months of August and December as these respectively correspond to the best and worst case in terms of amount of energy harvested.

The energy buffer size has been taken in $b_{\max} \in \{100, 250, 500, 1{,}000\}$mAh, whereas the buffer threshold has been set to $b_{\text{th}} = 50$mAh, imposing an outage of 1%, that is, $t_{\text{out}} = 0.01$ (see Equation (32)).



(a) Optimal policy $u(x) = \mu(x)$ $(x_s = 0)$ (b) Steady-state distribution $P(x)$ $(x_s = 0)$

Fig. 8. Optimal policy $u(x) = \mu(x)$ and associated steady-state distribution $P(x)$ for the energy state $x_s = 0$, $\alpha \in \{0.01, 0.5, 0.9\}$, $b_{\max} = 250$mAh, $b_{\text{th}} = 50$mAh.



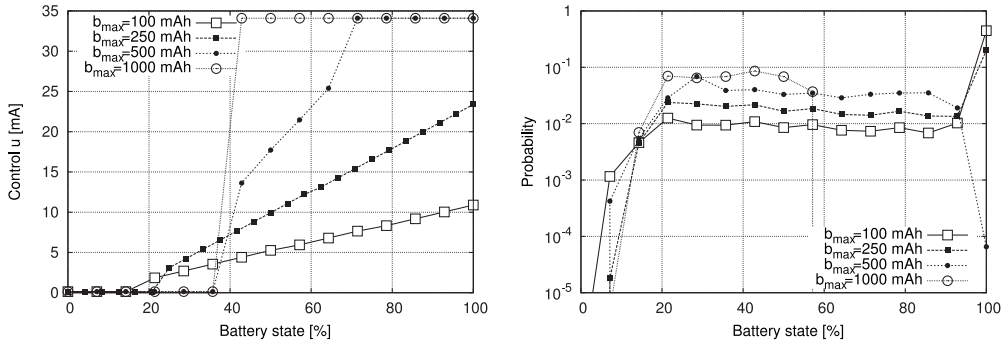
(a) Optimal policy $u(x) = \mu(x)$ $(x_s = 1)$ (b) Steady-state distribution $P(x)$ $(x_s = 1)$

Fig. 9. Optimal policy $u(x) = \mu(x)$ and associated steady-state distribution $P(x)$ for the energy state $x_s = 1$, $\alpha \in \{0.01, 0.5, 0.9\}$, $b_{\max} = 250$mAh, $b_{\text{th}} = 50$mAh.

7.1. Evaluation of the Policies

As an illustrative example, in Figures 8 and 9, we show the optimal action $u(x) = \mu(x)$ and $P(x)$, where $\mu(x)$ is defined as $\mu(x) = p\mu_{\lambda^-}(x) + (1-p)\mu_{\lambda^+}(x)$ and $P(x)$ is the steady-state distribution induced by the optimal policy; see Section 6. Note that the policies shown in these figures are all *feasible* as they satisfy the cost constraint while also providing the maximum possible throughput for the corresponding value of the discount factor α. For these plots, we have considered $\alpha \in \{0.01, 0.5, 0.9\}$, a maximum buffer size $b_{\max} = 250$mAh, $b_{\text{th}} = 0.2b_{\max} = 50$mAh, and $t_{\text{out}} = 0.01$.

With $\alpha = 0.9$ and $x_s = 0$ (see Figure 8(a)), the optimal policy does not transmit when the buffer state x_b is below or close to b_{th}, whereas for higher values of x_b, the optimal action $u(x)$ increases linearly. With our network parameters, the maximum energy consumption is $u_{\max} \simeq 34$mA, which for the considered example is never reached by the optimal policy with $\alpha = 0.9$, even for a full buffer. This is due to the constraint on the minimum buffer level. To see this, we recall that the sensor node has to make its decision $u(x)$ at the beginning of each stage and the amount of energy that will actually be harvested during the stage is only known statistically through $f_\iota(i|x_s)$ and $f_\tau(t|x_s)$. In our case, for $x_s = 0$ and $x_b = 100\%$, picking $u(x) = u_{\max}$ would lead to a violation of the constraint on the buffer. In fact, the optimal policy for each x_b picks the maximum $u(x)$ that, on average, satisfies the constraint with equality. For this example, this value



(a) Optimal policy $u(x) = \mu(x)$ $(x_s = 0)$ (b) Steady-state distribution $P(x)$ $(x_s = 0)$

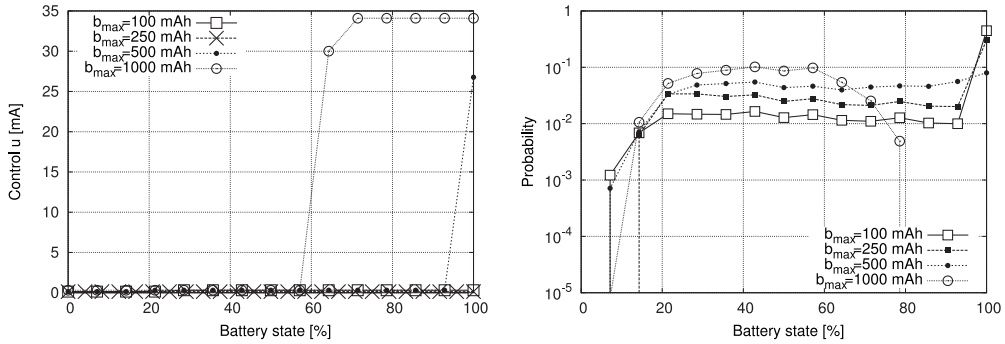
Fig. 10. Optimal policy $u(x) = \mu(x)$ and associated steady-state distribution $P(x)$ for the energy state $x_s = 0$, $\alpha = 0.9$, $b_{\max} \in \{100, 250, 500, 1{,}000\}$mAh, $b_{\text{th}} = 0.2b_{\max}$.

for $x_b = 100\%$ is about 24mA (referred in the following as *maximum admissible energy expenditure*). This means that in the very favorable cases, where the energy inflow is abundant, the optimal policy may leave some of the harvested energy unused (*energy wastage*). As we discuss shortly, this is avoided by increasing the buffer size $b_{\max}$.

By looking at the steady-state distribution for $x_s = 0$ and $\alpha = 0.9$ (see Figure 8(b)), we observe that $P(x)$ remains low for $x_b < b_{\text{th}}$ and is instead maximum for $x_b = b_{\max}$. This means that adopting the optimal policy allows the nodes to maximize the time spent with a full buffer and operating according to the maximum admissible energy expenditure. For state $x_s = 1$, we see that the optimal policy spends the minimum allowed energy consumption, $u_{\min}$, which corresponds to the energy required to keep the network operational, $I_{\text{out}}^{\min}$. In this way, the network saves energy during the low-energy state ($x_s = 1$), resuming the transmission of data packets in the high-energy state ($x_s = 0$). To summarize, each energy management policy induces a steady-state distribution of the buffer state. The optimal policy in this case makes it so that the steady-state probability of operating with a full buffer is maximized when the system is in the high-energy state (see Figure 8(a)); this is a desirable property as the sensor nodes can then maximize the time during which the maximum admissible energy expenditure is allocated. On the other hand, in the low-energy state, the node will only allocate $u_{\min}$. This leads to a modest energy consumption, which, in turn, implies that the steady-state distribution of the buffer state is preserved during the low-energy state (e.g., night) so that the node at the beginning of the next high-energy state (e.g., day) has a full buffer and can transmit right away using the maximum allowed rate.

As discussed in Section 6, a small α corresponds to a greedy transmission behavior. This is evident from the policies in Figures 8(a) ($x_s = 0$) and 9(a) ($x_s = 1$) for $\alpha \in \{0.01, 0.5\}$. The increased greediness reshapes the steady-state distribution $P(x)$. In particular, for $\alpha \in \{0.01, 0.5\}$ and $x_s = 0$, $P(x)$ assumes negligible values when $x_b > 50\%$. Hence, although the optimal policies would dictate to transmit using $u_{\max}$ for these values of x_b, the time spent in these states is negligible. As a result, the throughput achieved by the greedier policies is smaller (the throughput reduction is as large as 23% for the considered example).

In Figures 10 and 11, we look at the same performance for a fixed discount factor $\alpha = 0.9$ and a varying buffer size $b_{\max} \in \{100, 250, 500, 1{,}000\}$mAh. In particular, from Figure 10(a), we observe that the buffer size greatly affects the shape of the optimal policy. In fact, an increasing $b_{\max}$ also implies that the excess energy that is harvested during a stage can always be accumulated. Also, whenever the buffer is full or above 50/60%, with large buffers it is possible to transmit allocating the maximum energy



(a) Optimal policy $u(x) = \mu(x)$ $(x_s = 1)$ (b) Steady-state distribution $P(x)$ $(x_s = 1)$

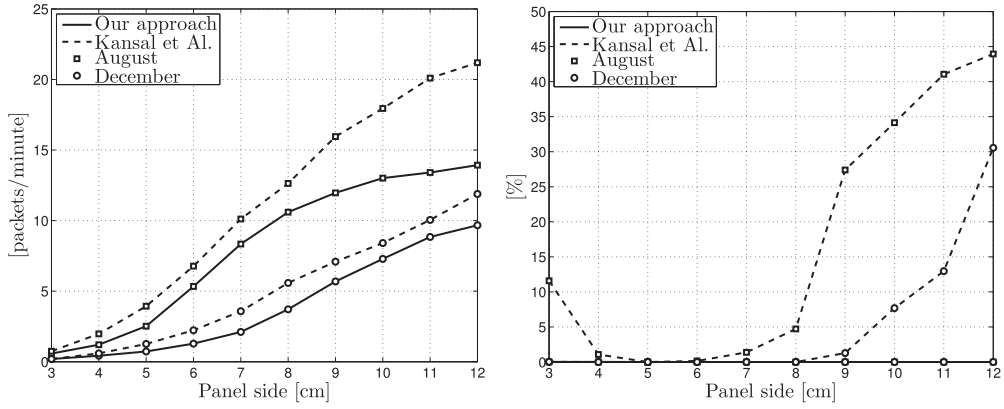
Fig. 11. Optimal policy $u(x) = \mu(x)$ and associated steady-state distribution $P(x)$ for the energy state $x_s = 1$, $\alpha = 0.9$, $b_{\max} \in \{100, 250, 500, 1{,}000\}$mAh, $b_{\text{th}} = 0.2b_{\max}$.

consumption $u_{\max}$, as the buffer is sufficiently large to ensure that the constraint will be satisfied at the end of the stage, irrespective of the amount of energy that will be harvested. While a big leap in performance is observed as we go from $b_{\max} = 100$mAh to $b_{\max} = 500$mAh (the throughput is about three times larger), increasing $b_{\max}$ to 1,000mAh only leads to marginal throughput improvements, smaller than 10%. This is because a buffer size of 500mAh is already sufficient to absorb unexpected energy peaks during the day (therefore minimizing the energy wastage) and to allow for the consumption of the maximum current $u_{\max}$ while satisfying the buffer constraint. The fact that a buffer of 500mAh suffices in our scenario is also shown by the steady-state distribution in Figure 10(b), where we see that a buffer size of $1,000$mAh has a negligible probability of getting filled beyond 58%. Finally, we discuss the impact of b_{th}. The energy buffer is allowed to decrease below this threshold to an extent controlled by t_{out} (see Equation (32)). When $t_{\text{out}} \to 0$, optimal policies effectively maintain the buffer above b_{th}, and this is equivalent to having a reduced battery capacity (of size $b_{\max} - b_{\text{th}}$). This, in turn, leads to less aggressive policies (see Figure 10(a)) that result in a smaller throughput. As an example, for the considered setup and $b_{\max} = 500$, when b_{th} goes from 100 to 300, we observe a throughput reduction of about 35%. This reduction is nonlinear in $b_{\max} - b_{\text{th}}$ (a linear relation would imply a reduction of 50%).

7.2. Performance Analysis

In this section, we evaluate the performance of the proposed solution focusing on a single network instance and considering the setup discussed at the beginning of Section 7. Also, we implemented the technique proposed in Kansal et al. [2007], referred to here as "Kansal," comparing it against our approach for the same network topology and energy arrival trace, obtained from real data for the city of Los Angeles; see Miozzo et al. [2014]. For a fair comparison, we implemented Kansal's energy prediction model with parameter $\alpha = 0.5$ and setting the $\rho_{\min}$ and $\rho_{\max}$ parameters as our optimal working points for the minimum and maximum drained current, that is, $(t_{\text{U}}^{\min}, t_{\text{dc}}^{\min})$ and $(t_{\text{U}}^{\lim}, t_{\text{dc}}^{\lim})$, respectively. In addition, we implemented their dynamic duty-cycle adaptation strategy not only by letting it change the duty cycle but also by letting it set the new optimal working point according to the new desired energy expenditure. Finally, for the energy buffer, we set $b_{\max} = 250$mAh, $b_{\text{th}} = 50$mAh; for the computation of our optimal policies, we set $\alpha = 0.9$,[4] and the same reward function $r(u)$ (defined in

[4]Not to be confounded with Kansal's $\alpha = 0.5$ mentioned earlier.



(a) Throughput (b) Outage

Fig. 12. Performance comparison between our proposed technique (solid line) and that proposed by Kansal et al. (dashed line) for the months of August and December and varying the panel size.

Section 4, see Figure 7) was used to compute the throughput for both techniques. For the comparison, we choose $b_{\max}$ to evaluate which solution is preferable with different system configurations: in particular, we show that our approach is robust regardless of the amplitude of the energy variations.

Figure 12 shows the average throughput (Figure 12(a)) and the outage probability (Figure 12(b)) for the two schemes. Our solution is represented through solid lines, whereas dashed lines are used for Kansal's. Also, we denote the results related to August and December with square and round markers, respectively. In both figures, the x-axis shows the panel side length in centimeters.

Our solution is outperformed in terms of throughput, but, on the other hand, it effectively maintains the outage probability within the prescribed threshold, while Kansal's scheme spends up to 44% of the time in outage (i.e., with a buffer charge x_b smaller than b_{th}) and up to 32% of the time with an empty battery (not shown due to space constraints). This is because our scheme delivers the maximum throughput, subject to the given buffer outage constraint. As further evidence of the different behaviors of the two techniques, in Figure 13 we show their energy consumption and battery variations for the same energy arrival trace during a time span of 3 days.

In this figure, we show the hourly variations of the harvested current, i; the chosen action (or control), u; and the instantaneous battery state, x_b, for both solutions. Here, i is represented for both approaches through shaded areas, while the control u is indicated with a solid line for Kansal and with a dash-dotted line for our approach. Similarly, the two battery states are represented with dashed and dotted lines for Kansal and our approach, respectively.

Two differences can be observed from Figure 13: the first is that the policy adopted in the low-energy state (night) by our solution is always more conservative than Kansal's, while the same policy is adopted during the day by the two schemes. The second observation is that Kansal's more aggressive behavior leads to battery outages. In fact, while during the second day Kansal successfully maintains energy neutrality, on the first and third days its battery got depleted for about one-third of the time.

In conclusion, we can say that our approach gives priority to the network sustainability, while Kansal's privileges its throughput. This is also reflected by the fact that our control is decided based on the amount of available charge in the battery, while Kansal tries to predict the future current availability to exploit it as efficiently as

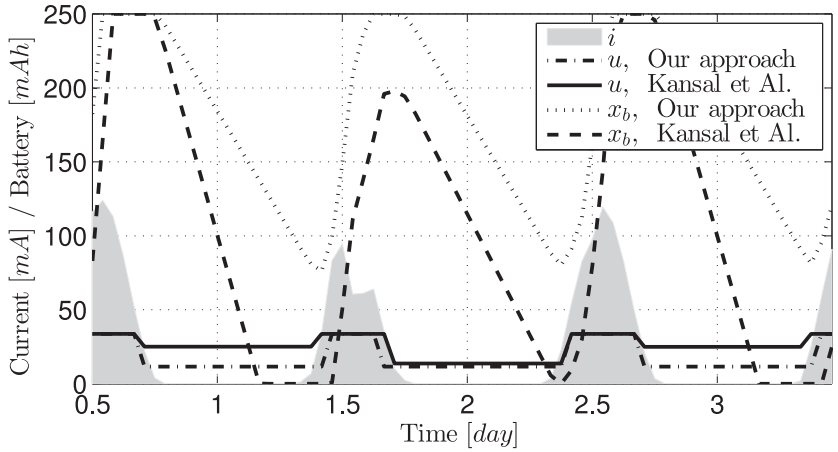


Fig. 13. Performance comparison between our solution and Kansal's during three simulated days for the month of August, considering a panel side of 10cm. The x-axis shows the simulation time, whereas the y-axis is used to visualize the amount of current harvested, drained, and the battery state.

possible. However, large variations in the energy availability are likely to lead to high prediction errors that, in turn, negatively affect the outage probability of Kansal. In conclusion, the adaptability of our scheme to the battery state also makes it robust to the degradation of the battery performance.

8. RELAXATION OF THE ASSUMPTIONS

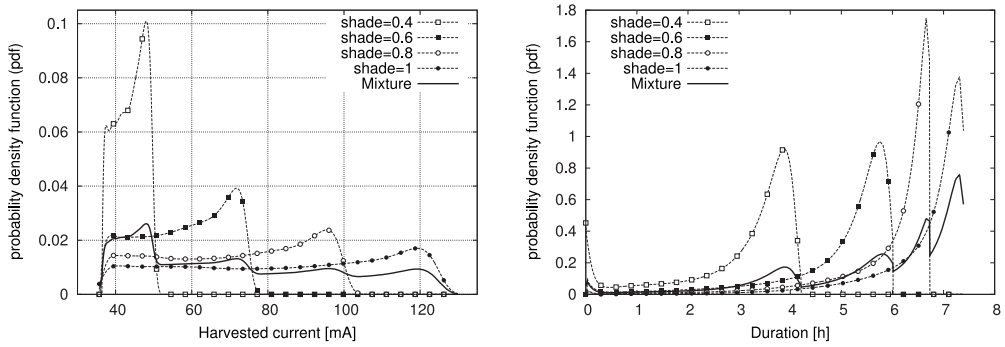
Here, we address the relaxation of the assumptions made during the analysis: namely, the homogeneity of energy sources, the transmission periodicity, the instantaneous network parameter update, and the fixed topology.

8.1. Heterogeneous Energy Sources

The stochastic MDP analysis of Section 6 leads to optimal online policies in the case where the energy arrival process is homogeneous; that is, all nodes have the same energy-harvesting statistics. However, as pointed out in Jeong and Culler [2012], in actual deployments different sensor nodes may be affected by slightly differing conditions such as blockage effects due to the surrounding objects that may partially shade the nodes, obstructing the direct sunlight.

In this section, we adapt our analysis to the case where the energy-harvesting statistics at the nodes differ. We do so following a two-step approach: (1) we extend the energy source model to account for the diversity in the harvested energy and we reuse the analysis of Section 6 with the new source model to obtain a new set of energy consumption policies, and (2) we use these new policies according to a simple and practical heuristic. Simulation results that prove the effectiveness of this approach are provided at the end of this section.

Energy source. For the energy sources, we account for an additional parameter vector $\boldsymbol{p}$, which includes parameters related to the deployment of the solar modules (such as the azimuthal angle, the tilt, the presence of obstructing objects, etc.). Hence, the new statistics for a given node are redefined as $f_i(i|x_s, \boldsymbol{p})$ and $f_\tau(t|x_s, \boldsymbol{p})$ for the input current i and the permanence time t when in state $x_s \in \mathcal{S}$, respectively. Equation (22)



(a) $f_\iota(i|x_s, p)$ (harvested current pdf) (b) $f_\tau(t|x_s, p)$ (permanence time pdf)

Fig. 14. Pdfs $f_\iota(i|x_s, p)$ and $f_\tau(t|x_s, p)$ obtained for $p \in \mathcal{D}$, state $x_s = 0$ (daytime) for Los Angeles in the month of August. A thick solid line is used to indicate the mixture densities.

generalizes to

$$f_\delta(d|u, x_s, \boldsymbol{p}) = \int_{t_{\min}(x_s)}^{t_{\max}(x_s)} f_\tau(t|x_s, \boldsymbol{p}) f_\iota(d/t + u|x_s, \boldsymbol{p})|t|^{-1}\mathrm{d}t\,,\ d \in \mathbb{R}. \tag{40}$$

Now, referring to $\boldsymbol{\rho}$ as the random vector associated with $\boldsymbol{p}$ (its realization), we indicate with $f_{\boldsymbol{\rho}}(\boldsymbol{p})$ the pdf describing the parameter space. Hence, the new pdf of the harvested charge in state x_s is obtained as

$$f_\delta(d|u, x_s) = \int_{\mathcal{D}(\boldsymbol{\rho})} f_\delta(d|u, x_s, \boldsymbol{p}) f_{\boldsymbol{\rho}}(\boldsymbol{p})\mathrm{d}\boldsymbol{p}, \tag{41}$$

where $\mathcal{D}(\boldsymbol{\rho})$ is the parameter space.

As a practical example, for the results that follow, we consider a scalar r.v. ρ describing the amount of shade received during the day by a particular sensor node. In fact, in accordance with Jeong and Culler [2012], we found that this is the parameter that affects the most the amount of harvested energy during the day. Here, we assume that the r.v. ρ can take four distinct values, that is, $\mathcal{D}(\rho) = \{0.4, 0.6, 0.8, 1\}$, which indicate the fraction of sunlight that hits the sensor node. Hence, $p = 1$ means that the solar module receives all the available sunlight for the considered location, whereas with $p = 0.4$, only 40% of the sunlight is absorbed, while the remaining 60% is blocked. Moreover, we considered a mass distribution function $f_\rho(p)$ that assigns a probability 0.55 to $p = 1$ and 0.15 to each of the remaining cases $p \in \{0.4, 0.6, 0.8\}$.

In this case, Equation (41) reduces to the following probability mixture:

$$f_\delta(d|u, x_s) = \sum_{p \in \mathcal{D}(\rho)} f_\delta(d|u, x_s, p) f_\rho(p)\,. \tag{42}$$

We have used the SolarStat tool to obtain $f_\delta(d|u, x_s, p)$ for all $p \in \mathcal{D}(\rho)$. Figure 14 shows the resulting pdfs $f_\iota(i|x_s, p)$ (Figure 14(a)) and $f_\tau(t|x_s, p)$ (Figure 14(b)) for $x_s = 0$ and $p \in \mathcal{D}(\rho)$. Also, a thick solid line is used to indicate the mixture densities.

Note that this approach makes it possible to account for heterogeneity in the solar source statistics, modeling our uncertainty on the actual amount of shade that will be received by each sensor node. This uncertainty is then embedded into the source model and the algorithm of Section 6.3 is reused with this new source model to generate new energy management policies.

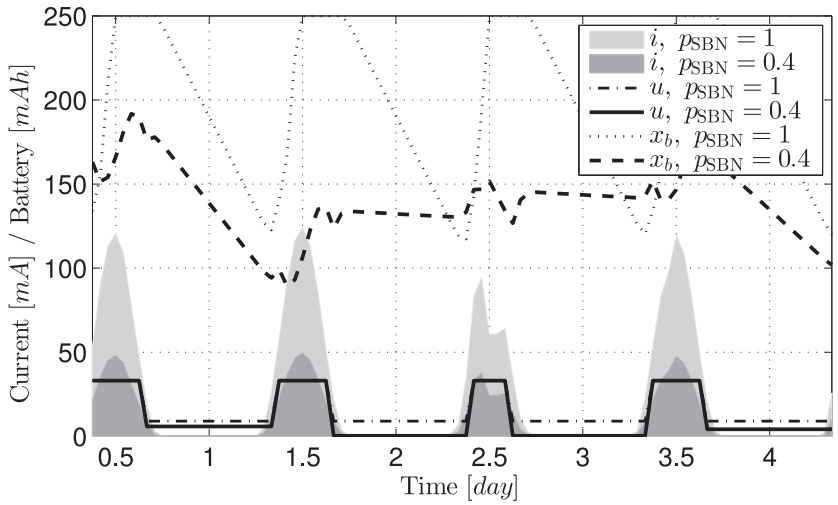


Fig. 15. Microscopic behavior of the heuristic policy in three simulated days. The x-axis shows the simulation time, whereas the y-axis represents, for the SBN node, the amount of current harvested, drained, and the battery state. The comparison is between the best ($p_{\mathrm{SBN}} = 1$) and the worst ($p_{\mathrm{SBN}} = 0.4$) case for the SBN node, considering $p = 1$ (no shading) for the bottleneck node.

Heuristic and results. First, we define the second bottleneck node (SBN) as the node located in the subtree originating from the bottleneck node (BN) that has the second-highest energy consumption, the node with the highest being the BN. The worst case for our control policies is when the BN has a shading coefficient equal to 1 (the available solar radiation is absorbed in full), while the SBN has the smallest shading coefficient 0.4. In this case, the BN is more likely to experience the most abundant energy inflow (see Figure 14). Thus, its energy buffer will be likely fuller than that of the SBN and, in turn, the BN might choose too aggressive a policy than what the SBN can efficiently adopt. Although this problem may be partially mitigated by the smaller energy consumption of the SBN with respect to that of the bottleneck node, it is still possible that the SBN experiences some battery outages.

To make the entire network self-sustainable, an additional expedient is in order. RPL DAO messages are used to periodically report relevant data to the sink, such as the location of the nodes, and so forth. Thus, it is possible to leverage these messages to collect, at the sink, additional information such as the battery state of all nodes and let the sink choose the policy based on the minimum among all buffer states (instead of solely using the energy buffer state of the BN). This worst-case control strategy makes the adopted policy slightly suboptimal due to the delay associated with the delivery of RPL messages but ensures that the entire network is self-sustainable.

Next, we show some simulation results considering that the BN has no shading, that is, $p = 1$, whereas we assume that the SBN has either $p = 0.4$ or $p = 1$. Moreover, we set the topology parameters of the SBN so as to reproduce the worst-case scenario in terms of energy consumption; that is, we assume that the SBN has the same number of interfering nodes (n_{i}) and packets (n_{int}) as the BN and just one node less ($n_{\mathrm{c}} - 1$) for the number of children. In Figure 15, we show the corresponding simulation results considering real solar traces for a time span of 3 days for the best ($p = 1$) and worst ($p = 0.4$) case in terms of energy harvested by the SBN. Note that for $p = 0.4$, the energy collected by the SBN (dark shaded area) is only 40% of that harvested (light shaded area) for $p = 1$. In both cases, our heuristic scheme opts for a rather aggressive policy during the high-energy state (solid and dash-dotted lines for the worst and best case,

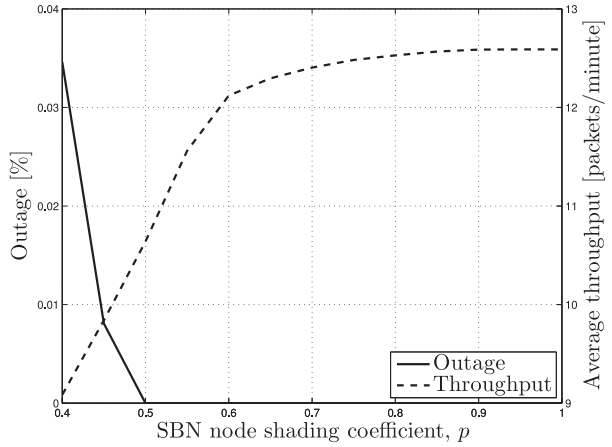


Fig. 16. Performance of the heuristic policy by varying the shade parameter $p \in [0.4, 1]$ (x-axis). On the left y-axis we show the outage probability for the SBN node (solid line), while in the right y-axis we show the throughput.

respectively), whereas in the worst case ($p = 0.4$), it adopts much more conservative policies during the low-energy state. In fact, in this case the battery level used for the selection of the policy is much lower due to the lower amount of current harvested by the SBN. Compare, for instance, the buffer state in the best case (dotted line) with that of the worst case (dashed line) at about time 1.6 days: for $p = 0.1$, the battery is completely filled up during the day, while for $p = 0.4$, the battery is only filled to about half of its capacity and should then be sparingly used to endure a full night.

Finally, in Figure 16, we show the average throughput (dashed line) for the network and the outage probability (solid line) for the SBN varying the shading conditions $p \in [0.4, 1]$. In all the tested cases, we used $p = 1$ for the bottleneck node. Note that the outage probability is always very small and almost always smaller than 0.1%. As expected, using our conservative approach may impact the throughput performance: this impact is negligible (less than 5%) for $p > 0.6$ but becomes substantial (up to 30%) in the most unfavorable case, that is, where the SBN has $p = 0.4$.

8.2. Transmission Periodicity

In Section 3, we assumed that nodes periodically sense the environment and generate their data at a constant rate of f_U packets per second. However, this is not strictly necessary; in fact, what really impacts the energy consumption is the *total number* of packets sent during a decision epoch. We preferred to study a periodic transmission process because it allows for a simpler mathematical analysis, leading to a closed-form solution for problem P1.

In addition, the transmission periodicity can be enforced at the application level by adopting a traffic shaping technique (i.e., by spacing out subsequent packets) through the user of transmission timers, so that the transmission rate will be no higher than f_U packets per second (see, e.g., Castellani et al. [2014]). This implementation trick can be useful to reduce the collision probability. In fact, reducing the traffic burstiness helps maintain the ratio $1/f'_U$ large, which translates into a low number of collisions.

In the article, we considered the network application to periodically sample environmental parameters. However, from the previous discussion, it is easy to see that our solution can also be applied to networks where the objective is that of communicating

alarms or events to the sink. In this case, our scheme supports up to Δ_k/t_U events per epoch per node, where Δ_k is the decision epoch duration.

Finally, note that the latency in the communication from the nodes to the sink is not governed by t_U, but by t_{dc}. In fact, as soon as an event occurs, the node detecting it can send the alarm to its next hop within at most t_{TX} seconds, which is dominated by t_{off} in the low-energy period and by t_{data} in the high-energy period.[5] Thus, delivering an alarm or an event from a node located h-hops away from the sink will take about $h \max(t_{data}, t_{off})$ seconds, independently of t_U.

8.3. Instantaneous Update

Our solution requires that all the nodes change their working point as soon as the energy source transitions to a new state. Although this is infeasible instantaneously, a simple and effective approximation can be employed. In particular, it is possible to exploit the information dissemination service provided by RPL to let the sink broadcast the new working point to all the sensor nodes. This procedure takes a finite amount of time and eventually terminates with all nodes knowing the new working point. During this lapse of time, different nodes in the network may use a different working point.

Soon after the energy source transition, as a consequence of the adoption of new parameters, two different configurations will coexist in the network: a group of nodes will have a rather high duty cycle and another group will instead have a smaller one. Many solutions have been proposed in the literature to allow the interaction of nodes with differing duty cycles. Here, we advocate the use of a very simple technique based on a *grace period*. During the grace period, nodes will wake up according to the highest between the two duty cycles and will send preambles using the t_{off} associated with the smallest of the two.

As a drawback of this procedure, nodes will consume a higher amount of energy during the grace period. However, RPL can disseminate the new configuration to the entire network in about ht_{dc} seconds if the longest path is at most h hops long. Since the length of a grace period is related to RPL dissemination time, the worst-case duration is obtained when the duty cycle is smaller (low-energy state) and for bigger networks; for instance, with our settings and a duty cycle $d_c = 1\%$, the longest grace period is shorter than 1 second, which is negligible compared to the duration of decision epochs. Nevertheless, to overcome this limitation, more advanced techniques can be used, along the lines of Vigorito et al. [2007].

8.4. Fixed Topology

Our reward function, $r(u)$, inherently depends on the topology through n_c, n_i, and n_{int}. Thus, the topology must remain static in order for a policy to maintain its optimality. However, this does not mean that the topology cannot change. In fact, note that topology information is periodically reported to the sink through RPL DAO messages. Hence, the impact of a changed topology can be estimated at the sink through the calculation of new topology parameters. At this point, if the throughput degradation is deemed too high or certain nodes are likely to deplete their batteries due to their increased load, the adoption of a new energy management policy at all nodes can be triggered. In this case, the sink will send a new policy to the nodes as if a transition of the energy source had occurred. When new nodes are added to the network, we let these behave as if they were in a grace period (see our discussion earlier) until they receive the new policy.

[5]We recall that $t_{TX} = t_{on} + t_{off} + t_{data} + (f'_U/f_U - 1)t_{dc}$.

9. CONCLUSIONS

In this article, we have provided a comprehensive mathematical framework for the design of energy-scavenging wireless sensor networks. Specifically, we have investigated the general class of problems related to the long-term and self-sufficient operation of wireless sensor networks powered by renewable energy sources. Our approach consisted of two nested optimization processes: the inner one (P1) characterizes the optimal operating point of the network subject to a given energy consumption figure (assumed constant), while the outer (P2) provides optimal energy management policies to make the system energetically self-sufficient, given the result of the inner problem and the statistical description of the energy source.

As a first step, we have defined an original energy consumption model describing the behavior of the bottleneck node (i.e., the node consuming the highest amount of energy) for a given routing topology and channel access technology. Second, we have proven that it is sufficient to grant the self-sufficiency of the bottleneck to ensure that all network nodes are also self-sufficient. Thus, we have solved P1 analytically, by deriving a closed-form expression for the optimal duty cycle and the optimal information generation rate that are to be used by all nodes to guarantee their perpetual and autonomous operation. This result was derived by neglecting packet collisions at first, and it was subsequently extended through a heuristic to take the effect of packet collisions into account.

Hence, using the solution of P1 and a statistical description of the energy source, we have formulated P2, a DT-CMDP, returning the online policies that maximize the long-term average throughput of the network while ensuring its self-sufficiency in the presence of a stochastic energy source. We have solved P2 using a Lagrangian relaxation technique, which permits a convenient exploration of the solution space. Also, we described how the obtained policies can be implemented to overcome the computational complexity of the approach at the sensor nodes.

We have then used our framework to explore the impact of key system parameters on the design of energy-harvesting sensor networks. In detail, we have assessed the impact of network topologies on the reward function, also studying the impact of battery and photovoltaic panel sizes on the optimal energy consumption strategies. Thus, the framework has been utilized to derive the long-term average network performance, which includes the network throughput and the steady-state probabilities of the battery charge state when the optimal policies are adopted by the nodes. Finally, we thoroughly validated our optimal policies against state-of-the-art approaches, also proving its robustness when our main assumptions are relaxed. Our solution proved to be more conservative than the state of the art, and, although at the price of a slightly lower throughput, it ensures the self-sustainability of all sensor nodes for all battery sizes and environmental conditions.

APPENDIX

A. CHANNEL ACCESS MODELING IN THE PRESENCE OF PACKET COLLISIONS

To take collisions and channel transmission errors into account, we derived the following fixed-point analysis. Note that our collision model is similar to the one considered in previous work (see, e.g., Yang and Heinzelman [2012]). The analysis that we present in what follows differs in the fact that we consider the transmission of periodic endogenous traffic, and this allows for a closed-form expression of the collision probability, which is derived next. We refer to the packet error probability for the transmission of the bottleneck node as e_{t}, which depends on the selected modulation and coding scheme and on the channel impairments (attenuation, noise, etc.); see, for example, Chapter 6 of Goldsmith [2005]. Here, we consider e_{t} fixed. Also, we refer to $n_{\mathrm{i}} \geq 0$ as the number of interfering nodes and to e_{c} as the packet collision probability.

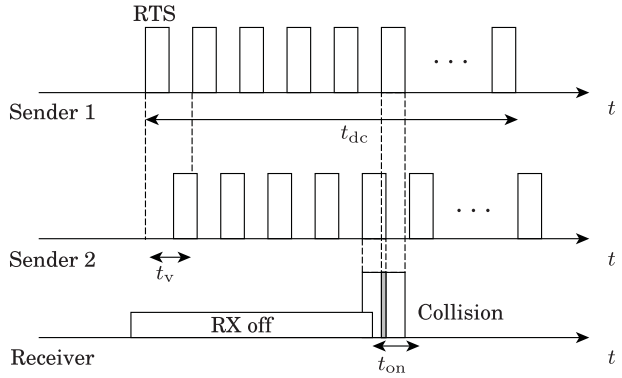


Fig. 17. Graphical example of a collision: the first sender (top) starts sending periodical RTSs; before the vulnerability time, t_v, has elapsed, the second sender starts sending RTSs too. Since none of them is aware of the other, they keep on transmitting RTSs for t_{dc} seconds (the duration of the RTS burst). When one of the intended destinations wakes up, it will receive a corrupted RTS (collision).

Given that a packet is successful when no channel errors occur (w.p. $1 - e_t$) and it is not collided (w.p. $1 - e_c$), the total packet error probability is obtained as $e_p = e_c + e_t - e_c e_t$. Now, note that when $e_p \geq 0$, due to the increased number of packet losses and the associated retransmissions, we have that the packet transmission rate of the bottleneck node increases to $f'_U \geq f_U$, where f_U is the original information rate. Hence, one packet is transmitted on average every $1/f'_U$ seconds, where $1/f'_U$ is the new average intertransmission time. We assume that the transmission within this time period occurs by picking a transmission instant uniformly at random in $[0, 1/f'_U]$. Moreover, given our LPL MAC, whenever a packet is transmitted, there exists a vulnerability period[6] of t_v seconds and a collision event occurs whenever any of the n_i interferers picks its own transmission time within this interval; the probability of this event to occur is $p_c = f'_U t_v$ (see Figure 17 for a graphical example). Note that p_c corresponds to the probability that a given interferer picks its transmission time within period t_v, given that this transmission instant is (assumed) uniformly distributed in $[0, t'_U]$, where $t'_U = 1/f'_U$ is the interpacket transmission interval in the presence of retransmissions.

Given this, the probability that a collision event is due to $k \in \{1, \ldots, n_i\}$ interferers is given by $\binom{n_i}{k} p_c^k (1 - p_c)^{(n_i - k)}$ and the probability that the packet sent by the bottleneck node collides is finally obtained as $e_c = 1 - (1 - p_c)^{n_i} = 1 - (1 - f'_U t_v)^{n_i}$, which corresponds to the probability that at least one of the interferers transmits in the vulnerable interval. Note that the previous equation can be solved for f'_U, expressing the latter as a function, $g_1(\cdot)$, of the other parameters:

$$f'_U = g_1(e_c, t_v, n_i) = [1 - (1 - e_c)^{(1/n_i)}] t_v^{-1}. \tag{43}$$

On the other hand, for a packet error rate e_p, f'_U can be related to f_U through the following function $g_2(\cdot)$:

$$f'_U = g_2(e_c, e_t, f_U) = f_U (1 - e_p)^{-1} = f_U (1 - e_c - e_t + e_c e_t)^{-1}. \tag{44}$$

Observing that f_U is given, t_v is a (hardware-dependent) constant and e_t and n_i are also constant for a given transmission scenario (topology, modulation, and channel model), we have that the only unknown parameter is the collision probability e_c. Since $g_1(\cdot)$

[6]The vulnerability period is a tunable parameter reflecting the time needed for practical architectures to put the radio into the RX state and detect incoming packets.

and $g_2(\cdot)$ both return the packet transmission rate f'_U, the working point for the system is obtained by imposing $g_1(\cdot) = g_2(\cdot)$, solving for e_c and retaining the smallest real solution to the previous equality. The steady-state transmission rate f'_U is attained using this value of the collision probability with either $g_1(\cdot)$ or $g_2(\cdot)$. This is a practical method to obtain f'_U at equilibrium, in the presence of channel errors and collisions. Note, however, that a solution is not always guaranteed to exist, and this occurs when the offered traffic exceeds the maximum capacity of the considered access channel. In Appendix B, we provide an approximated formula to conveniently calculate e_c and a stability analysis to mathematically assess when the channel access admits a solution.

B. COLLISION PROBABILITY AND FEASIBILITY CONDITION FOR THE CHANNEL ACCESS

Collision probability approximation: Here we derive a closed-form approximation for the collision probability e_c at equilibrium. As discussed in Appendix A, this is obtained by looking at the points where $g_1(\cdot)$ and $g_2(\cdot)$ intersect (see later for the necessary condition for this to occur). When these functions do intersect, they have two real solutions in the range $e_c \in [0, 1]$ and the e_c at equilibrium is the smallest real solution. From the equality $g_2(e_c, e_t, f_U) = g_1(e_c, t_v, n_i)$, using $x = e_c$, we get

$$(1 - e_t - x + e_t x)(1 - (1 - x)^{1/n_i}) - f_U t_v = 0. \tag{45}$$

Now, we employ the Taylor expansion of $(1 - x)^{1/n_i}$, around the point $x_0 = 0$:

$$(1 - x)^{1/n_i} = 1 - \frac{x}{n_i} + O(x^2), \tag{46}$$

which, used in Equation (45) leads to

$$x^2(1 - e_t) - x(1 - e_t) + f_U t_v n_i = 0. \tag{47}$$

The discriminant of Equation (47) is $\Delta = 1 - 4(f_U t_v n_i)/(1 - e_t)$; thus, the condition $\Delta \geq 0$ implies $n_i \leq \lfloor (1 - e_t)/(4 f_U t_v) \rfloor$. When the latter is verified, the solution for the collision probability is given by the smallest solution of Equation (47), that is:

$$e_c \simeq \frac{1 - \sqrt{\Delta}}{2}. \tag{48}$$

For illustrative purpose, considering $e_t \leq 0.3$ and $f_U t_v \leq 0.001$, which is largely verified in practice,[7] Equation (48) is accurate up to the third decimal place for $n_i \leq 20$ and up to the second decimal place for $n_i \leq 50$. Note that these settings for e_t and $f_U t_v$ are rather extreme and more accurate results are achieved for the practical network examples of this article. For these, we have that $f_U t_v = 0.0004$, $e_t = 0.1$, and $n_i = 5$, and with these parameters, the gap between the actual average number of retransmissions $n'_{retx} = e_p/(1 - e_p)$ (considering the impact of packet collisions) and the approximation $n_{retx} = e_t/(1 - e_t)$ (considering $e_c = 0$) leads to a relative error of $100(n'_{retx} - n_{retx})/n'_{retx} = 2.18\%$.

Feasibility collision for the channel access: In what follows, we examine the condition under which the channel access problem of Appendices A and B, whereby n_i nodes transmit over the same medium, is feasible. Intuitively, a random access channel has a limited "hosting capacity." When too many users transmit over it at too high a rate, exceeding the capacity limit, the random access system becomes unstable. In this case, the collision probability tends to increase indefinitely, leading to a zero throughput for all users. Next, we mathematically derive the condition under which

[7] As shown in Section 7, feasible values for f_U are typically larger than one packet per minute that, considering $t_v \leq 0.01$ s, leads to $f_U t_v \leq 1.6 \cdot 10^{-4}$.

the channel access system of Section 3 is stable as a function of the parameters f_U, the transmission rate of the node (of their endogenous traffic, without considering collisions); t_v, the vulnerability period; and n_i, the number of nodes that transmit over the same medium (interferers).

Mathematically, a finite solution for e_c exists only when the two curves $g_1(\cdot)$ (see Equation (43)) and $g_2(\cdot)$ (see Equation (44)) intersect. Through a more accurate inspection of the behavior of Equations (43) and (44), it is easy to see that a solution to $g_1(\cdot) = g_2(\cdot)$ does not exist when we have that $g_2(e_c, e_t, f_U) > g_1(e_c, t_v, n_i)$, for all values of $e_c \in [0, 1]$. Through some algebra, it is easy to verify that this condition corresponds to

$$f_U t_v > (1 - e_t)(1 - e_c)(1 - (1 - e_c)^{1/n_i}) \stackrel{def}{=} g_3(e_c, e_t, n_i),\ \forall e_c \in [0, 1]\,. \tag{49}$$

Now, the LHS of Equation (49) is a constant, whereas the RHS is a continuous function of e_c that has a maximum in $e_{c,max}$, where

$$e_{c,max} = 1 - \left(\frac{n_i}{1 + n_i}\right)^{n_i}. \tag{50}$$

Note that Condition (49) is verified if the LHS is strictly greater than the RHS ($g_3(e_c, e_t, n_i)$) for all values of e_c and this *must also hold* for $e_c = e_{c,max}$. In this case, $g_1(\cdot)$ and $g_2(\cdot)$ do not intersect and, in turn, the system does not admit a stable working point. The previous reasonings formally prove that the *feasibility condition* for the channel access is

$$f_U t_v \le g_3(e_{c,max}, e_t, n_i) = (1 - e_t)\left(\frac{1}{1 + n_i}\right)\left(\frac{n_i}{1 + n_i}\right)^{n_i}, \tag{51}$$

as when Equation (51) is verified, $g_1(\cdot)$ and $g_2(\cdot)$ intersect in at least one point.

C. PROBLEM P1: DERIVATION OF THE CLOSED-FORM SOLUTION

In what follows, we derive the closed-form expression of the optimal working point (t_U^*, t_{dc}^*) for a collision-free channel. The first step is to rewrite Equation (9) by neglecting packet collisions, that is, $e_c = 0$, which implies $f'_U/f_U - 1 = e_t/(1 - e_t)$ and, in turn:

$$\begin{aligned} I_{TX} \;=\; & (i_c + i_t)[t_{dc}/2 + t_{on}/2 + t_{data} + (e_t/(1 - e_t))t_{dc}] \\ & \times[(1 + n_c)/t_U + (2 + n_c)/t_{rpl}]. \end{aligned} \tag{52}$$

Subsequently, we rewrite Equations (9) through (14) isolating the terms depending on t_U and t_{dc} and introducing coefficients $\{c_1, \ldots, c_5\}$ and $\{a_1, \ldots, a_{11}\}$ (see Table IX):

$$I_{TX} \;=\; c_1(t_{dc}a_1/t_U + a_2/t_U + t_{dc}a_3 + a_4) \tag{53}$$

$$I_{RX} \;=\; c_2(a_5 t_U + a_6) \tag{54}$$

$$I_{INT} \;=\; c_2(a_7 t_U + a_8) \tag{55}$$

$$I_{CPU} \;=\; c_3 a_9 t_U \tag{56}$$

$$\begin{aligned} r_{IDLE} \;&=\; 1 - r_{TX} - r_{RX} - r_{INT} - r_{CPU} \\ &=\; a_{10} - a_1 t_{dc}/t_U - a_3 t_{dc} - a_{11}/t_U \end{aligned} \tag{57}$$

$$I_{IDLE} \;=\; r_{IDLE}(c_4 + c_5 t_{on}/t_{dc}). \tag{58}$$

Table IX. Coefficients *a*, *b*, and *c*

$a_1 = (1+n_c)(1/2 + e_t/(1-e_t))$	$b_1 = c_1 a_1$
$a_2 = (1+n_c)(t_{data} + t_{on}/2)$	$b_2 = c_1 a_2$
$a_3 = (2+n_c)(1/2 + e_t/(1-e_t))/t_{rpl}$	$b_3 = c_1 a_3$
$a_4 = (2+n_c)(t_{data} + t_{on}/2)/t_{rpl}$	$b_4 = c_1 a_4$
$a_5 = n_c t_{data}$	$b_5 = c_2 a_5$
$a_6 = (1+n_c+n_i)t_{data}/t_{rpl}$	$b_6 = c_2 a_6$
$a_7 = t_{int} n_{int}$	$b_7 = c_2 a_7$
$a_8 = t_{int} n_{int}/t_{rpl}$	$b_8 = c_2 a_8$
$a_9 = t_{cpu} k_U$	$b_9 = c_3 a_9$
$a_{10} = 1 - a_4 - a_6 - a_8$	$b_{10} = -a_1 c_4$
$a_{11} = a_2 + a_5 + a_7 + a_9$	$b_{11} = -a_1 t_{on} c_5 - a_{11} c_4$
$c_1 = i_c + i_t$	$b_{12} = -a_3 c_4$
$c_2 = i_c + i_r$	$b_{13} = -a_3 t_{on} c_5 + a_{10} c_4$
$c_3 = i_c$	$b_{14} = a_{10} c_5 t_{on}$
$c_4 = i_s$	$b_{15} = -a_{11} c_5 t_{on}$
$c_5 = i_c + i_r - i_s$	

Table X. Coefficients *d*, *e*, and *f*

$d_1 = b_1 + b_{10}$	$e_0 = 4d_1 d_6 - d_2^2$
$d_2 = b_2 + b_5 + b_7 + b_9 + b_{11}$	$e_1 = 4d_1 d_5 + 4d_3 d_6 - 2d_7 d_2$
$d_3 = b_3 + b_{12}$	$e_2 = 4d_5 d_3 - d_7^2$
$d_4 = b_4 + b_6 + b_8 + b_{13}$	$f_0 = -a_{10} d_6 - a_{11} d_5$
$d_5 = b_{14}$	$f_1 = a_3 d_6 - a_1 d_5$
$d_6 = b_{15}$	$f_2 = d_1 a_{10} + d_3 a_{11}$
$d_7 = d_4 - u$	$f_3 = -d_1 a_3 + a_1 d_3$

Rewriting Equations (53) through (58) and introducing coefficients $\{b_1, \ldots, b_{15}\}$ (see Table IX) lead to:

$$I_{TX} = b_1 t_{dc}/t_U + b_2/t_U + b_3 t_{dc} + b_4 \tag{59}$$

$$I_{RX} = b_5 t_U + b_6 \tag{60}$$

$$I_{INT} = b_7 t_U + b_8 \tag{61}$$

$$I_{CPU} = b_9 t_U \tag{62}$$

$$I_{IDLE} = b_{10} t_{dc}/t_U + b_{11}/t_U + b_{12} t_{dc} + b_{13} + b_{14}/t_{dc} + b_{15}/(t_{dc} t_U). \tag{63}$$

$I_{out}(t_U, t_{dc})$ is thus obtained using Equation (2). For compactness, $I_{out}(t_U, t_{dc})$ is expressed using a fourth set of coefficients ($\{d_1, \ldots, d_6\}$ of Table X):

$$I_{out}(t_U, t_{dc}) = d_1 t_{dc}/t_U + d_2/t_U + d_3 t_{dc} + d_4 + d_5/t_{dc} + d_6/(t_{dc} t_U). \tag{64}$$

Now, taking the first order derivative of Equation (64) with respect to t_{dc}, we obtain

$$\frac{\partial I_{out}(t_U, t_{dc})}{\partial t_{dc}} = d_3 + d_1/t_U - (d_5 + d_6/t_U)/t_{dc}^2, \tag{65}$$

which leads to the following result:

$$\frac{\partial I_{out}(t_U, t_{dc})}{\partial t_{dc}} = 0 \Rightarrow t_{dc}^*(t_U) = \pm\sqrt{\frac{d_6/t_U + d_5}{d_1/t_U + d_3}}, \tag{66}$$

where the wanted solution is the one with the plus sign. Note that $t^*_{\rm dc}(t_{\rm U})$ is the optimal duty cycle, which minimizes the power consumption for a given interpacket transmission time $t_{\rm U}$ (endogenous traffic). At this point, we compute Equation (64) for $t^*_{\rm dc}(t_{\rm U})$, subtracting u (i.e., the target current budget) and equating to zero:

$$I_{\rm out}(t_{\rm U}, t^*_{\rm dc}(t_{\rm U})) - u = \frac{t^*_{\rm dc}(t_{\rm U})^2(d_1/t_{\rm U} + d_3) + (d_6/t_{\rm U} + d_5)}{t^*_{\rm dc}(t_{\rm U})} + d_2/t_{\rm U} + d_4 - u = 0. \quad (67)$$

Now, raising Equation (67) to the second power and reordering leads to

$$(4d_1d_6 - d_2^2)/t_{\rm U}^2 + (4d_1d_5 + 4d_3d_6 - 2d_2d_7)/t_{\rm U} + 4d_3d_5 - d_7^2 = e_0/t_{\rm U}^2 + e_1/t_{\rm U} + e_2 = 0. \quad (68)$$

Note that Equation (68) is solved for $t_{\rm U}$, with $u \in [u_{\min}, u_{\max}]$, with $I^{\min}_{\rm out} = u_{\min}$ and $u_{\max} = I^{\lim}_{\rm out}$. It is easy to verify that the solution of problem P1, $t^*_{\rm U}$, is the only positive solution of the previous equation.

For the calculation of $I^{\min}_{\rm out}$ and $I^{\lim}_{\rm out}$, we proceed as follows. First, for what concerns the minimum current consumption $I^{\min}_{\rm out}$, we first obtain the optimal duty cycle for the case where no data-gathering operations are performed (i.e., $t_{\rm U}$ goes to infinity), $t^{\min}_{\rm dc} = \lim_{t_{\rm U}\to+\infty} t^*_{\rm dc}(t_{\rm U}) = \sqrt{d_5/d_3}$. Hence, we use this result together with Equation (64) to compute $I^{\min}_{\rm out}$:

$$I^{\min}_{\rm out} = \lim_{t_{\rm U}\to+\infty} I_{\rm out}(t_{\rm U}, t^*_{\rm dc}(t_{\rm U})) = d_3\sqrt{d_5/d_3} + d_4 + d_5\sqrt{d_3/d_5}. \quad (69)$$

To obtain $I^{\lim}_{\rm out}$, we first define $t^{\lim}_{\rm dc} = t^*_{\rm dc}(t^{\lim}_{\rm U})$, where $t^{\lim}_{\rm U}$ is obtained from $r_{\rm IDLE}(t^{\lim}_{\rm U}, t_{\rm dc}) = 0$ (meaning that the node is always busy and maximizes its transmission activity). From the latter equality we get

$$t^{\lim}_{\rm U}(t_{\rm dc}) = \frac{a_1t_{\rm dc} + a_{11}}{a_{10} - a_3t_{\rm dc}}, \quad (70)$$

which, together with Equation (66), leads to

$$\begin{aligned} t^{\lim}_{\rm dc} &= \sqrt{\frac{d_6/t^{\lim}_{\rm U} + d_5}{d_1/t^{\lim}_{\rm U} + d_3}} \\ &= \sqrt{\frac{d_6(a_{10} - a_3t^{\lim}_{\rm dc}) + d_5(a_1t^{\lim}_{\rm dc} + a_{11})}{d_1(a_{10} - a_3t^{\lim}_{\rm dc}) + d_3(a_1t^{\lim}_{\rm dc} + a_{11})}}, \end{aligned} \quad (71)$$

where the equality in the second line follows from replacing $t^{\lim}_{\rm U}$ with Equation (70). Thus, raising Equation (71) to the second power and solving for $t^{\lim}_{\rm dc}$ leads to the third-order equation:

$$f_3\left(t^{\lim}_{\rm dc}\right)^3 + f_2\left(t^{\lim}_{\rm dc}\right)^2 + f_1t^{\lim}_{\rm dc} + f_0 = 0, \quad (72)$$

and $t^{\lim}_{\rm dc}$ is the largest solution of Equation (72). Finally, $t^{\lim}_{\rm U}$ is obtained by plugging $t^{\lim}_{\rm dc}$ into Equation (70) and $I^{\lim}_{\rm out} = I_{\rm out}(t^{\lim}_{\rm U}, t^{\lim}_{\rm dc})$ is finally calculated from Equation (64).

D. ON THE CORRECTNESS OF THE BOTTLENECK ANALYSIS

In this appendix, we analyze the network stability given that the system is tuned on the bottleneck node and all other nodes use the same operating point of the latter. To prove that when the bottleneck node is energetically self-sufficient, the same holds true for all the other nodes in the network, we will show that $I_{\rm out}(n_{\rm c}, n_{\rm i}, n_{\rm int})$ is an increasing function of $n_{\rm c}$, $n_{\rm i}$, and $n_{\rm int}$ (all the other parameters remaining fixed). To this end, let

n_c^b, n_i^b, n_{int}^b, and I_{out}^b respectively be the topology parameters and the output current of the bottleneck node. Given that for all nodes the following inequalities hold:

$$n_c \leq n_c^b, \;\; n_i \leq n_i^b, \;\; n_{int} \leq n_{int}^b, \tag{73}$$

for any sensor node in the network, we have that $I_{out} \leq I_{out}^b$, which proves our claim. In what follows, we only show that $I_{out}(n_c, n_i, n_{int})$ is an increasing function of n_c, as the proof for the other variables develops along the same lines.

First of all, we take the first-order derivative of $I_{out}(n_c, n_i, n_{int})$ with respect to n_c:

$$\begin{aligned}
\frac{\partial I_{out}}{\partial n_c} &= \frac{\partial I_{TX}}{\partial n_c} + \frac{\partial I_{RX}}{\partial n_c} + \frac{\partial I_{IDLE}}{\partial n_c} \\
&= c_1 \frac{\partial r_{TX}}{\partial n_c} + c_2 \frac{\partial r_{RX}}{\partial n_c} + \left(c_4 + c_5 \frac{t_{on}}{t_{dc}} \right) \frac{\partial r_{IDLE}}{\partial n_c} \\
&= c_1 \frac{\partial r_{TX}}{\partial n_c} + c_2 \frac{\partial r_{RX}}{\partial n_c} - \left(c_4 + c_5 \frac{t_{on}}{t_{dc}} \right) \left(\frac{\partial r_{TX}}{\partial n_c} + \frac{\partial r_{RX}}{\partial n_c} \right) \\
&= \frac{\partial r_{TX}}{\partial n_c} \left(c_1 - c_4 - c_5 \frac{t_{on}}{t_{dc}} \right) + \frac{\partial r_{TX}}{\partial n_c} \left(c_2 - c_4 - c_5 \frac{t_{on}}{t_{dc}} \right).
\end{aligned} \tag{74}$$

Hence, we proceed showing that $\partial r_{TX}/\partial n_c$, $\partial r_{RX}/\partial n_c$, $(c_1 - c_4 - c_5 t_{on}/t_{dc})$, and $(c_2 - c_4 - c_5 t_{on}/t_{dc})$ are all positive quantities. For the two derivatives the following holds:

$$\begin{aligned}
\frac{\partial r_{TX}}{\partial n_c} &= \frac{t_{dc}/2 + t_{on}/2 + t_{data} + (f'_U/f_U - 1)}{1/t_U + 1/t_{rpl}} \\
\frac{\partial r_{RX}}{\partial n_c} &= \frac{t_{data}}{1/t_U + 1/t_{rpl}},
\end{aligned} \tag{75}$$

and it is easy to show that all the addends of the two sums are positive, because all of them are either time or frequency quantities that are positive by definition. The term $f'_U/f_U - 1$ is also positive since f'_U is the arrival rate in the presence of channel errors, which implies that $f'_U \geq f_U$.

Finally, for what concerns the other two terms, they can be rewritten as

$$\begin{aligned}
c_1 - c_4 - c_5 t_{on}/t_{dc} &= i_t + i_c - i_s - (i_r + i_c - i_s) t_{on}/(t_{on} + t_{off}) \\
&\geq (i_r + i_c - i_s) t_{off}/(t_{on} + t_{off}) \\
c_2 - c_4 - c_5 t_{on}/t_{dc} &= i_r + i_c - i_s - (i_r + i_c - i_s) t_{on}/(t_{on} + t_{off}) \\
&= (i_r + i_c - i_s) t_{off}/(t_{on} + t_{off}),
\end{aligned} \tag{76}$$

where the inequality in the second line holds since $i_t \geq i_r$ for all radio technologies. Also, note that $i_s \ll i_r$, $i_r + i_c - i_s > 0$, and $t_{off}/(t_{on} + t_{off})$ are by definition positive, which proves that both terms in Equation (76) are greater than or equal to zero. Thus, $\partial I_{out}/\partial n_c$ is the sum of positive terms, which implies that $I_{out}(n_c, n_i, n_{int})$ is an increasing function of n_c and that $I_{out} \leq I_{out}^b$ for every sensor node.

REFERENCES

Eitan Altman. 1999. *Constrained Markov Decision Processes*. Chapman and Hall CRC.

Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The Internet of things: A survey. *Computer Networks* 54, 15 (2010), 2787–2805.

Dimitri P. Bertsekas. 2012. *Dynamic Programming and Optimal Control* (4th ed.). Vol. 2. Athena Scientific.

Frederick J. Beutlerand and Keith W. Ross. 1985. Optimal policies for controlled Markov chains with a constraint. *Journal of Mathematical Analysis and Applications* 112, 1 (Nov. 1985), 236–252.

Naveed Anwar Bhatti, Affan Ahmed Syed, and Muhammad Hamad Alizai. 2014. Sensors with lasers: Building a WSN power grid. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*. IEEE Press, 261–272.

Riccardo Bonetto, Nicola Bui, Michele Rossi, and Michele Zorzi. 2012. McMAC: A power efficient, short preamble multi-channel medium access control protocol for wireless sensor networks. In *ICST/IEEE SIMUTools*.

Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. 2006. X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks. In *ACM SenSys*.

Nicola Bui, Angelo P. Castellani, Paolo Casari, Michele Rossi, Lorenzo Vangelista, and Michele Zorzi. 2012. Implementation and performance evaluation of wireless sensor networks for smart grids. *Smart Grid Communications and Networking*. Cambridge University Press.

Nicola Bui and Michele Rossi. 2013. Dimensioning self-sufficient networks of energy harvesting embedded devices. In *Wireless Access Flexibility*. Springer, 138–150.

Angelo Castellani, Michele Rossi, and Michele Zorzi. 2014. Back pressure congestion control for CoAP/6LoWPAN networks. *Elsevier Ad Hoc Networks* 18, 1 (2014), 71–84.

Chee-Seng Chow and John N. Tsitsiklis. 1989. The complexity of dynamic programming. *Elsevier Journal of Numerical Complexity* 5, 4 (1989), 466–488.

Cyrus Derman and Ralph E. Strauch. 1966. A note on memoryless rules for controlling sequential control processes. *Annals of Mathematical Statistics* 37, 1 (1966), 276–278.

Kai-Wei Fan, Zizhan Zheng, and Prasun Sinha. 2008. Steady and fair rate allocation for rechargeable sensors in perpetual sensor networks. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*. ACM, 239–252.

Eugene A. Feinberg and Adam Shwartz. 1995. Constrained Markov decision models with weighted discounted rewards. *Mathematics of Operations Research* 20, 2 (1995), 302–320.

Carlo Fischione, Pangun Park, and Sinem Coleri Ergen. 2013. Analysis and optimization of duty-cycle in preamble-based random access networks. *Wireless Networks* 19 (2013), 1–17.

Marios Gatzianas, Leonidas Georgiadis, and Leandros Tassiulas. 2010. Control of wireless networks with rechargeable batteries. *IEEE Transactions on Wireless Communications* 9, 2 (2010), 581–593.

Andrea Goldsmith. 2005. *Wireless Communications*. Cambridge University Press.

Maria Gregori and Miquel Payaró. 2013. Energy-efficient transmission for wireless energy harvesting nodes. *IEEE Transactions on Wireless Communications* 12, 3 (2013), 1244–1254.

Jason Hsu, Sadaf Zahedi, Aman Kansal, Mani Srivastava, and Vijay Raghunathan. 2006. Adaptive duty cycling for energy harvesting systems. In *Proceedings of the 2006 International Symposium on Low Power Electronics and Design*. ACM, 180–185.

Longbo Huang and Michael J. Neely. 2013. Utility optimal scheduling in energy harvesting networks. *IEEE Transactions on Networking* 21, 4 (2013), 1117–1130.

Jaein Jeong and David Culler. 2012. A practical theory of micro-solar power sensor networks. *ACM Transactions on Sensor Networks* 9, 1 (April 2012), 1–36.

Aman Kansal, Jason Hsu, Sadaf Zahedi, and Mani B Srivastava. 2007. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)* 6, 4 (2007), 32.

JeongGil Ko, Omprakash Gnawali, David Culler, and Andreas Terzis. 2011. Evaluating the performance of RPL and 6LoWPAN in TinyOS. In *Proceedings of the Workshop on Extending the Internet to Low Power and Lossy Networks (IP+SN'11)*.

Jing Lei, Roy Yates, and Larry Greenstein. 2009. A generic model for optimizing single-hop transmission policy of replenishable sensors. *IEEE Transactions on Wireless Communications* 8, 2 (2009), 547–551.

Shixin Luo, Rui Zhang, and Teng Joon Lim. 2013. Optimal save-then-transmit protocol for energy harvesting wireless transmitters. *IEEE Transactions on Wireless Communications* 12, 3 (2013), 1196–1206.

Nicolò Michelusi, Kostas Stamatiou, and Michele Zorzi. 2013. Transmission policies for energy harvesting sensors with time-correlated energy supply. *IEEE Transactions on Communications* 61, 7 (2013), 2988–3001.

Nicolò Michelusi and Michele Zorzi. 2013. Optimal random multiaccess in energy harvesting wireless sensor networks. In *Proceedings of the IEEE International Conference on Communications (ICC'13)*.

Marco Miozzo, Davide Zordan, Paolo Dini, and Michele Rossi. 2014. SolarStat: Modeling photovoltaic sources through stochastic Markov processes. In *Proceedings of the IEEE Energy Conference (ENERGYCON'14)*.

David Moss, Jonathan Hui, and Kevin Klues. 2007. Low power listening. *TinyOS Core Working Group, TEP* 105 (2007).

National Renewable Energy Laboratory. 2013. Renewable Resource Data Center. Retrieved from http://www.nrel.gov/rredc/.

Michael J. Neely, Eytan Modiano, and Chih-Ping Li. 2008. Fairness and optimal stochastic control for heterogeneous networks. *IEEE Transactions on Networking* 16, 2 (2008), 396–409.

Omur Ozel, Kaya Tutuncuoglu, Jing Yang, Sennur Ulukus, and Aylin Yener. 2011. Transmission with energy harvesting nodes in fading wireless channels: Optimal policies. *IEEE Journal on Selected Areas in Communications* 29, 8 (2011), 1732–1742.

Athanasios Papoulis and S. Unnikrishna Pillai. 2002. *Probability, Random Variables and Stochastic Processes* (4th ed.). McGraw-Hill.

Vinod Sharma, Utpal Mukherji, Vinay Joseph, and Shrey Gupta. 2010. Optimal energy management policies for energy harvesting sensor nodes. *IEEE Transactions on Wireless Communications* 9, 4 (2010), 1326–1336.

Solarbotics Ltd. 2013. SCC-3733 Monocrystalline Solar Cells. Retrieved from http://solarbotics.com/.

Jacob Sorber, Alexander Kostadinov, Matthew Garber, Matthew Brennan, Mark D. Corner, and Emery D. Berger. 2007. Eon: A language and runtime system for perpetual systems. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*. ACM, 161–174.

Cristiano Tapparello, Osvaldo Simeone, and Michele Rossi. 2013. Dynamic compression-transmission for energy-harvesting multihop networks with correlated sources. *IEEE/ACM Transactions on Networking,* 22, 6 (Dec. 2014), 1729–1741.

Christopher M. Vigorito, Deepak Ganesan, and Andrew G. Barto. 2007. Adaptive duty cycling for energy harvesting systems. In *Proceedings of the Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'07)*.

Feng Wang and Jiangchuan Liu. 2011. Networked wireless sensor data collection: Issues, challenges, and approaches. *IEEE Communications Surveys & Tutorials* 13, 4 (2011), 673–687.

Douglas John White. 1993. Markov decision processes: Discounted expected reward or average expected reward? *Mathematics of Operations Research* 172, 2 (1993), 375–384.

Tim Winter, Pascal Thubert, Anders Brandt, Jonathan Hui, Richard Kelsey, Philip Levis, Kristofer Pister, René Struik, Jean Philippe Vasseur, and Roger Alexander. 2010. *RPL: IPv6 Routing Protocol for Low Power and Lossy Networks*. IETF Internet Draft draft-ietf-roll-rpl-12. Retrieved from https://datatracker.ietf.org/doc/draft-ietf-roll-rpl/.

Jing Yang and Sennur Ulukus. 2012. Optimal packet scheduling in an energy harvesting communication system. *IEEE Transactions on Wireless Communications* 60, 1 (2012), 220–230.

Ou Yang and Wendi B. Heinzelman. 2012. Modeling and performance analysis for duty-cycled MAC protocols with applications to S-MAC and X-MAC. *IEEE Transactions on Mobile Computing* 11, 6 (June 2012), 905–921.

Ting Zhu, Yu Gu, Tian He, and Zhi-Li Zhang. 2010. Eshare: A capacitor-driven energy storage and sharing network for long-term operation. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. ACM, 239–252.

Received October 2013; revised August 2014; accepted October 2014