Link Scheduling in Rechargeable Wireless Sensor Networks with Harvesting Time and Battery Capacity Constraints

Tony, Sieteng Soh, Mihai Lazarescu
School of Electrical Engineering, Computing and Mathematical Sciences
Curtin University
Perth, Australia
tony@postgrad.curtin.edu.au, {s.soh, m.lazarescu}@curtin.edu.au

Kwan-Wu Chin
School of Electrical, Computer and Telecommunications Engineering
University of Wollongong
Wollongong, Australia
kwanwu@uow.edu.au

***Abstract*—A link scheduler ensures the transmissions in rechargeable Wireless Sensor Networks (rWSNs) are collision-free. Hence, it plays a critical role in ensuring high network capacity and the energy used for transmission/reception is not wasted due to collisions. This paper proposes a scheduler that generates a Time Division Multiple Access link schedule for use in a rWSN. Different from most prior works, our scheduler considers the time required by each node to harvest sufficient energy to transmit/receive a packet. Further, it utilizes the more efficient Harvest-Use-Store (HUS) model and considers sensor nodes with finite battery capacity. We present a greedy heuristic that activates links according to the earliest time in which their end nodes have sufficient energy to transmit/receive a packet. Our simulation results show that the time to recharge a sensor node significantly increases the link schedule or superframe lengths; i.e., by up to 563.9% as compared to the case where sensor nodes have no energy constraint. Further, in comparison to the Harvest-Store-Use (HSU) model, using HUS can reduce superframe lengths by up to 45.3%. Our experiments also show that increasing battery capacity does not effect the superframe length significantly; i.e., it reduces the length only by up to 2.5%. Finally, our proposed heuristic can generate superframe lengths that are on average 23% longer as compared to the lower bound on the superframe length when nodes have energy constraint.**

I. Introduction

Wireless sensor networks (WSNs) will play a critical role in the future development of Internet of Things (IoTs) [2]. In particular, WSNs help collect information/data such as activities in a smart home [4] or the blood pressure of a person [8]. Hence, it is important for nodes to transmit their sensed data to a sink or fusion center efficiently. This is particularly important for military and health applications. A key concern, however, is the available energy on sensor nodes as they are likely to have a rechargeable battery or a capacitor with a finite capacity. Nodes replenish their battery by harvesting energy from the environment [15]. It is worth noting that once a node's battery is full, any subsequent energy arrival is loss. Also, the energy harvesting rate is random and location specific. In theory, assuming a sensor node has a lower energy consumption rate than its energy harvesting rate, it can operate perpetually, but with *delays*. As an example, consider the energy that is generated by air-flow, which has a power density of 360 μW/cm^2 [7]. Assuming a surface area of 50 cm^2, its energy harvesting rate is 18 mJ/s. Now consider a Mica2 mote [1] that needs 72 mJ/s of energy to transmit/receive a packet. The mote needs to wait up to four seconds, aka *harvesting/recharging times* or *cycles*, before it can transmit or receive one packet.

To date, researchers have proposed numerous link schedulers for rechargeable WSNs (rWSNs) that aim to minimize co-channel interference, energy usage and/or optimize Quality of Service (QoS). A link scheduler aims to ensure the transmissions of nodes experience minimal or no interference [19]. This paper considers Time Division Multiple Access (TDMA) as it ensures collision-free transmissions, meaning energy is not wasted due to collisions. A link scheduler is responsible for allocating a transmission slot to each link, and ensuring concurrently active links experience no interference. Ideally, we want a link schedule to be as short as possible because it is repeated once derived. This means a short schedule allows links to transmit frequently, and thereby, affording links a high capacity. Moreover, we want to have as many non-interfering links scheduled into each slot. This improves network capacity.

A link scheduler must consider the varying energy harvesting rates of sensor nodes. Otherwise, in their allocated transmission/reception slot, they might not have any energy to transmit/receive. Specifically, a link can be scheduled only if its two end nodes have accumulated sufficient energy to transmit and receive one data packet. Thus, each node must wait for its battery to be recharged when it has insufficient energy; we call this the *harvesting time* constraint. Another issue to consider is that each battery has limited capacity; we call this the *battery capacity* constraint. The constraints can result in longer link schedule. Thus, in addition to interference, a link scheduler must also consider the harvesting time and the battery capacity constraints. In the sequel, we discuss these issues with the aid of an example.

Figures 1a and 1b show two examples of rWSNs with four nodes and three directed links; the number next to each link shows its activation time slot. Note that links (v_1, v_2), (v_3, v_2), and (v_4, v_2) interfere with each other and thus cannot be scheduled to transmit concurrently. Hence, we see that link (v_1, v_2), (v_2, v_3) and (v_4, v_2) have been allocated time slots $t = 1$, $t = 2$, and $t = 3$, respectively. This means we have a TDMA schedule or *superframe* length of three slots. A key assumption in the example in Figure 1a is that nodes have sufficient energy to transmit and receive in its allocated time slot. However, as mentioned earlier, as sensor nodes have a different energy harvesting cycle, this assumption may not hold.

In Figure 1b, each node has a different *recharging cycle* (in terms of slots); this governs when a node gains sufficient energy to transmit/receive again. For example, node v_1 is able to transmit/receive every five slots; denoted as $v_1|5$. Consequently, unlike the previous example, the schedule length now exceeds three slots as each node must wait its battery to recharge when it has insufficient energy. Although node v_2 has sufficient energy at time slot $t = 2$, none of its incoming links can be activated at time 2 because its neighbors have insufficient energy to transmit a packet. More specifically, link (v_1, v_2) can be scheduled no earlier than slot $t = 5$ because node v_1 can only transmit after time $t = 5$.

Another critical issue is the battery capacity of sensor nodes. First consider sensor nodes with a sufficient battery capacity. For this case, at time $t = 3$, node v_2 continues to accumulate energy, and thus at time $t = 5$, it has sufficient energy to receive three consecutive packets. Thus, links (v_1, v_2), (v_3, v_2), and (v_4, v_2) can be scheduled at time $t = 5$, $t = 6$, and $t = 7$, respectively, giving an optimal schedule of length 7. Now assume the battery of node v_2 has capacity to store the energy required to receive only for one packet. This means the battery of node v_2 can be recharged only after it is used at time $t = 5$. Thus, node v_2 can receive the first packet at time $t = 5$ but the second packet no earlier than at time $t = 5 + 2 = 7$, i.e., after waiting for another round of energy harvesting time. Similarly, the third packet can be received no earlier than at time $t = 7 + 2 = 9$. Hence, when the battery has a capacity of one unit, the optimal schedule length is 9, i.e., links (v_1, v_2), (v_3, v_2), and (v_4, v_2) can be scheduled at time $t = 5$, $t = 7$, and $t = 9$, respectively; see Figure 1b.

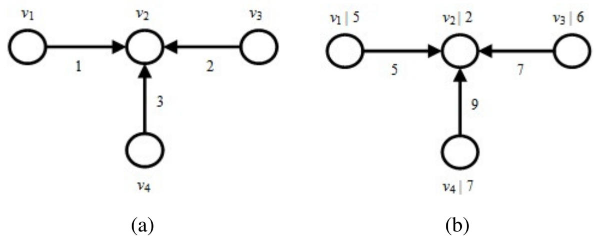


Figure 1. An example (a) with only interference constraint, and (b) interference and energy constraints. The number next to each link denotes activation time, and $v_x|z$ denotes node x requires z time slots to recharge its battery to a level before the next transmission or reception is possible.

This paper contains the following *contributions*. First, we propose a TDMA link scheduler to maximize throughput in rWSNs in which (i) sensor nodes have a different energy harvesting cycle, (ii) sensor nodes have finite battery capacity, and (iii) each link i has a weight $w_i \geq 1$ and must be scheduled at least w_i times. To the best of our knowledge, except for reference [17], no link scheduler has considered all of the above factors, i.e., (i)-(iii). While the authors of [17] consider unlimited battery capacity, we investigate the impact of battery size on superframe lengths. Further, reference [17] uses the Harvest-Store-Use (HSU) battery charging model whereas we use Harvest-Use-Store (HUS) to reduce schedule length as well as potential energy loss due to energy storage inefficiency [9]. Second, we develop an efficient greedy technique to generate TDMA link schedules. Our technique does not require an *extended* conflict graph as in [17], and thus it is more efficient; see Section II for details. The results in Section V show that using HUS on average reduces superframe lengths by 28.55% as compared to using HSU. Further, we found that increasing harvesting time produces, on average, 289.5% longer superframes. We also found that while using larger battery size reduces superframe length, it does not significantly affect the superframe length. Specifically, a larger battery size only reduces the superframe length by up to 2.5%. Finally, the results show that our heuristic can produce superframes that are on average only 1.7 times longer than the lower bound length.

The rest of this paper is organized as follows. Section II reviews related work, while Section III presents network model and problem at hand. The details of our solution is described in Section IV, and its performance evaluation is reported in Section V. Finally, Section VI concludes the paper and provides future research directions.

II. Related Works

To the best of our knowledge, except for reference [17], there is no work that solves a similar problem to ours. Liu et al. [11] study TDMA transmission schemes based on the energy harvesting profiles and the battery capacity of users. They use two cases: (i) infinite, and (ii) finite battery capacity in many-to-one network. The aims are to maximize throughput and to minimize transmission time. The authors of [6] aim to find efficient schedulers and to increase the average throughput using an iterative technique for energy harvesting in multiple access channel. Lenka et al. [10] aim to derive a schedule that minimizes latency and collisions using a distributed scheduler. Sun et al. [17] consider the HSU model [9] in which the harvested energy must be stored in the battery first before it can be used. Each battery has a *recharging time* that determines when a node has sufficient energy to transmit/receive one packet. The battery has unlimited capacity and is leakage free. Each link can be scheduled only if the battery of its end nodes have accumulated sufficient energy to transmit/receive one data packet. Thus, each node with insufficient energy must wait for at least one recharging time before it can activate one link. The authors propose two link schedulers to maximize network throughput: (i) without link weight (or $w_{i,j} = 1$), and (ii) with link weight ($w_{i,j} > 1$). For (i), they generate a *conflict graph* $C_G(V', E')$ from $G(V, E)$ and link interference models. For (ii), their scheduler requires an *extended* conflict graph $C'_G(V'', E'')$, which is generated from C_G and $w_{i,j}$ of each link such that each link (i, j) appears $w_{i,j}$ times in C'_G.

The HUS protocol was first introduced in [12]. As compared to HSU, the HUS scheme has a higher achievable harvesting rate and lower energy loss [9]. Recently, there are works that consider HUS, but these works do not consider our problem. More specifically, in [22], Yuan et al. consider HUS over two channels: (i) static, and (ii) block fading channel. They propose optimal energy policies based on a discrete-time energy model to maximize throughput. They then extend their work in [21] and aim to minimize the energy used for transmission subject to finite delay constraint.

In *summary*, prior link scheduling works for rWSN, except for [17], do not consider battery recharging time and capacity. While Sun et al. [17] consider recharging time, their solution can potentially recharge battery with infinite amount of energy. Further, the authors consider HSU [9]. In contrast, we use the HUS model as it offers better performance as compared to the HSU model [9]. Since HUS allows nodes to use harvested energy immediately, our approach produces shorter link schedule, meaning larger throughput, than link schedule produced using the HSU model. Note that if nodes use HSU, then any harvested energy in slot t can only be used in subsequent slots. Further, considering Ni-MH rechargeable battery, only 70% of the harvested energy can be stored [9], and thus in HSU, significant amount of valuable energy is loss due to energy storage inefficiency. In contrast, the HUS model stores only any unused energy harvested at slot t in the battery for future use, and hence our approach reduces potential energy loss. Finally, unlike the solution in [17] that requires an extended conflict graph, we only require a conflict graph. This is advantages as the extended conflict graph becomes computationally expensive to use with large link weight $w_{i,j}$.

III. Preliminaries

We first describe our rWSN model and introduce key notations. Then, we formalize the problem.

A. Network Model

We model a rWSN as directed graph $G(V, E)$, where each node $v_i \in V$ is a sensor node i and each link $l_{i,j} \in E$ denote a directional link from v_i to v_j. Note that i and j refer to node label. Each node v_i has a transmission range of $\mathcal{R}_i$. The Euclidean distance between v_i and v_j is denoted as $||v_i - v_j||$. If the condition $||v_i - v_j|| \leq \mathcal{R}_i$ is true, node v_i can transmit or receive packets to/from v_j. Each link $l_{i,j}$ has weight $w_{i,j} \geq 1$ to denote the number of required time slots in the resulting schedule. For example, in Figure 2a, $w_{1,2} = 3$.

We assume the protocol interference model [13]. Primary interference occurs when a node transmits and receives a packet simultaneously, or receives more than one transmission at the same time; that is, each node is half-duplex. Secondary interference occurs when say a node A, while receiving a

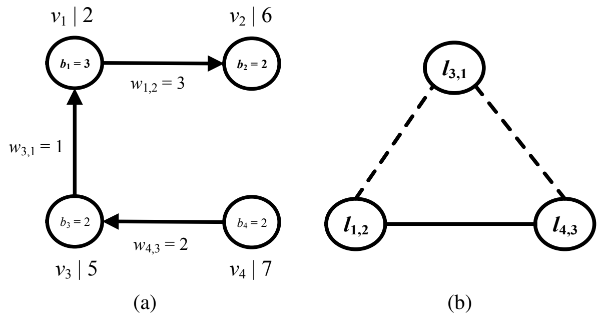


Figure 2. A rWSN model. (a) Graph G, and its (b) conflict graph C_G. Also shown are primary (dashed lines) and secondary (solid line) interference.

packet from its neighbor B, also receives a transmission from node C that is intended for another node D. In Figure 2a, there are two primary interferences, i.e., link $l_{4,3}$ with $l_{3,1}$, and $l_{3,1}$ with $l_{1,2}$. Also shown is secondary interference at node v_3 that is caused by v_1.

The interference between links is aptly modeled by a conflict graph $C_G(V', E')$ [5]. For a given $G(V, E)$, one can construct its corresponding conflict graph as follows: (i) each vertex in V' represents a link in E, i.e., $|V'| = |E|$, and (ii) each edge in E' represents two links of G that experience primary or secondary interference if they are active together. Figure 2b shows the conflict graph C_G for the rWSN in Figure 2a. Primary and secondary interference are represented by a dashed and a solid line, respectively. Given a conflict graph, one can then apply any graph coloring algorithms, such as edge coloring [3], to determine the links that can co-exist together. That is, all links with the same color do not interfere and thus can transmit together. Note that our problem and its solution, described in Section III-B and IV respectively, can be used for other interference models, e.g., the RTS/CTS-based model [17].

We define a TDMA superframe or a link schedule as a collection of consecutive, equal-sized time slots. All links in each slot do not experience primary and secondary interference. Indeed, after coloring a conflict graph, we can place all links with the same color in a slot. Let $\mathcal{S}$ represent the superframe and $|\mathcal{S}|$ denote the schedule length (in slots). Each slot has size τ (in seconds), and is sufficient to transmit one packet. Without loss of generality, we set τ to one millisecond. Each slot is either *empty* or contains one or more non-interfering, concurrently active links. A slot is empty when no sensor nodes have sufficient energy to transmit/receive. This fact obviates the use of prior link schedulers that assume nodes have no energy constraint as the derived schedule is likely to contain idle slots.

A sensor node consumes energy when sensing, computing, and communicating, which include transmitting, receiving, listening for messages on the radio channel, sleeping, and switching state [14]. In this work, we assume communication is the only source of energy expenditure. This is reasonable because as shown in [14], the energy consumption of nodes attributed to communications is significantly larger than other operations, e.g., 180.10 mJ, 17.242 mJ, and 5.2 mJ for communication, sensing, and computing, respectively. Similar to [18], we assume the energy usage for transmission and reception is equal. We write ϵ (in Joule) as the energy consumed when transmitting or receiving one packet. For example, assuming a TI CC2420 transceiver that uses 226 nJ/bit for transmission [16], and a packet size of 125 bytes or 1000 bits, one can compute the total amount of energy used to transmit one packet as $\epsilon = 226$ μJ.

We consider HUS [9], where harvested energy is first stored in a capacitor for immediate use and any unused energy is stored in a rechargeable battery for use in future slots. Specifically, a node i contains a harvester that generates energy from its environment, e.g., the sun, and two energy storage devices: (i) a super capacitor with a capacity of c_i, and (ii) a rechargeable battery with a capacity of b_i; both capacities are in unit of ϵ. Let $r_i > 0$ (in slots) be the total number of slots or *harvesting time* required by a node i to accumulate 1ϵ amount of energy. Thus, a node has a harvesting rate of $\frac{\epsilon}{r_i}$ per time slot. We assume each capacitor for node i has sufficient capacity to store all harvested energy in each slot, i.e., $c_i \geq 1/r_i$. Further, it has energy storage efficiency of 100% and leakage factor of zero [12]. When the harvester delivers energy that is larger than the needs of a node, i.e., when $r_i < 1$, any excess energy is stored in a rechargeable battery that has very small leakage factor for future use [9]. In this paper, we assume the rechargeable battery of all nodes has a leakage factor of zero. Further, each battery is initially empty, and the energy level of each capacitor is zero at the start of each time slot. Each rechargeable battery supports *shallow recharge* [15], meaning it can be recharged even though it is partially discharged. Let $b_i(t)$ be the energy level of node i's rechargeable battery at time t. For each node i, we use $A_{i,t}$ (in unit of ϵ) to denote the amount of energy that node i can use within slot t; note, $\frac{1}{r_i} \leq A_{i,t} \leq b_i(t) + \frac{1}{r_i}$. The value of $A_{i,t}$ is the sum of energy level of node i's rechargeable battery and capacitor at time t. Formally, we have $A_{i,t} = b_i(t) + \frac{1}{r_i}$. As the capacitor has a high leakage factor [9], the energy level in each capacitor is always equal to the energy harvested in each slot, i.e., $\frac{1}{r_i}$. When $A_{i,t} < 1\epsilon$, node i cannot transmit or receive packets at time t. In contrast, if $r_i \leq 1$ or $A_{i,t} \geq 1\epsilon$, a node can transmit/receive at any time, assuming there is an available packet. Note that the available energy $A_{i,t}$ is a function of node i's rechargeable battery capacity (b_i), energy harvesting rate (r_i) and energy usage. Further, although a battery cannot be charged and discharged at the same time [20], it is possible that its energy level increases even when the node uses energy at the same time slot t. This case occurs when $r_i < 1$. On the other hand, when $r_i > 1$ and $b_i(t) \geq 1 - \frac{1}{r_i}$, then node i will draw the fraction $1 - \frac{1}{r_i}$, aka energy shortfall, from its rechargeable battery.

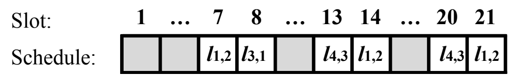
Let T_i be the earliest time slot when node i has at least 1ϵ of energy to transmit/receive one packet; i.e., T_i is the earliest slot such that $A_{i,T_i} \geq 1\epsilon$. Thus the earliest time link $l_{i,j}$ can be scheduled is at time $t_{i,j} = \textbf{max}(T_i, T_j)$, i.e., when the end nodes of the link have at least 1ϵ of energy. For each node i, we initialize $T_i = r_i$; it is updated whenever node i transmits/receives a packet. We use $t(i)$ to denote the last (most recent) time node i transmits/receives a packet.

Figure 2a shows an example rWSN with four nodes. The energy level of their battery is $b_1 = 3$ and $b_2 = b_3 = b_4 = 2$, and their harvesting time is $r_1 = 2$, $r_2 = 6$, $r_3 = 5$ and $r_4 = 7$ time slots. At time $t = 1$, the available energy of node v_1 is $A_{1,1} = 0.5\epsilon$, while at time $t = 2$, it increases to $A_{1,2} = 1\epsilon$. Thus, the earliest time node v_1 can transmit/receive is $T_1 = 2$. For the other nodes, we have $T_2 = 6$, $T_3 = 5$, and $T_4 = 7$. The earliest time in which v_1 and v_2 can transmit/receive is therefore $t_{1,2} = \textbf{max}(2, 6) = 6$; for other nodes, we have $t_{3,1} = 5$, and $t_{4,3} = 7$. Notice that the smallest $t_{i,j}$ is five. Thus, the first four slots in schedule $\mathcal{S}$ are empty. Assume a scheduler selects link $l_{1,2}$ first at time 6; thus $t(1) = t(2) = 6$, and the next earliest time node 1 and 2 can transmit or receive is at slot $T_1 = 6 + 2 = 8$ and $T_2 = 12$, respectively.

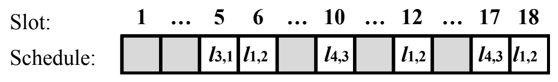
B. Problem Statement

We are now ready to define the **Link Scheduling in Harvest-Use-Store (LSHUS)** problem: Generate the TDMA link schedule $\mathcal{S}$ with the shortest length $|\mathcal{S}|$ for a given rWSN such that (i) each link $l_{i,j}$ that is allocated a time slot t satisfies $A_{i,t} \geq 1\epsilon$ and $A_{j,t} \geq 1\epsilon$, and (ii) each link $l_{i,j} \in E$ is scheduled at least $w_{i,j}$ times in $\mathcal{S}$. For example, in Figure 2a, link $l_{3,1}$ can be scheduled no earlier than $t = 5$; and link $l_{1,2}$ needs to be scheduled three times because $w_{1,2} = 3$.

To illustrate the effect of link scheduling on $|\mathcal{S}|$, consider the example in Figure 2. Figure 3a shows one *feasible* schedule. A schedule is called *feasible* if it satisfies constraints (i) and (ii). The optimal solution can be found in Figure 3b. Note that the figure shows only non empty slots, i.e., each empty slot is represented as "...". Our problem aims to generate a schedule $\mathcal{S}$ with the shortest length, e.g., the schedule in Figure 3b that has the shortest length of $|\mathcal{S}| = 18$.



(a) One feasible TDMA schedule.



(b) The optimal TDMA schedule.

Figure 3. TDMA schedules for rWSN in Fig. 2

Let $\mathcal{S}_{\mathcal{E}}$ denote the superframe generated when there is no interference; i.e., the activation of links is delayed by insufficient energy as opposed to interference. One can use $|\mathcal{S}_{\mathcal{E}}|$ as the lower bound of superframe length for LSHUS, computed as

$$|\mathcal{S}_{\mathcal{E}}| = max\left(r_i\left(\sum_{j \epsilon N(i)} w_{i,j} + \sum_{j \epsilon N(i)} w_{j,i}\right)\right) \quad (1)$$

where $N(i)$ is the set of node i's neighbours.

IV. Solution

This section first describes two propositions to be used in our greedy algorithm to solve LSHUS. Then it describes the algorithm.

A. Propositions

Our algorithm aims to schedule links that can be activated at the earliest time. It uses the following two propositions.

Proposition 1: The amount of energy (in unit of ϵ) that node i can use at time $t \geq t(i)$ is,

$$A_{i,t} = \mathbf{min}(b_i + 1/r_i, A_{i,t(i)} + (t - t(i))/r_i) \tag{2}$$

Proof: Let $t(i)$ be the time in which node i last draws energy from its battery. The maximum amount of energy that can be harvested since $t(i)$ is therefore $\frac{t-t(i)}{r_i}$, where t is the current time. However, the available energy at node i is bounded by its battery's capacity b_i and harvesting rate within one slot. This implies $A_{i,t} \leq b_i + 1/r_i$. ■

Proposition 2 computes the next value of T_i after node i transmits/receives one packet. Let $\alpha_{i,t}$ be a Boolean variable such that $\alpha_{i,t} = 1$ ($\alpha_{i,t} = 0$) if at time t node i has $A_{i,t} < 1\epsilon$ ($A_{i,t} \geq 1\epsilon$). Recall that when $A_{i,t} = 1\epsilon$, node i can transmit/receive one packet at time t.

Proposition 2: The next earliest time slot when node i has sufficient energy to transmit/receive one packet is,

$$T_i = \sigma + \alpha_{i,t} \times (r_i \times (1 - A_{i,t})) \tag{3}$$

Proof: After node i consumes 1ϵ of energy at time t, the next value of T_i depends on the remaining available energy in node i, i.e., $A_{i,t}$, and harvesting time r_i. Eq. (3) considers two cases. First, when $A_{i,t} < 1\epsilon$, the harvester needs to scavenge an extra $(1 - A_{i,t})\epsilon$ such that the node has 1ϵ of energy. It takes $r_i \times (1 - A_{i,t})$ slots to harvest the additional energy. Thus $T_i = t + r_i \times (1 - A_{i,t})$, and for this case, in Eq. (3), $\alpha_{i,t}$ = 1, and $\sigma = t$. Second, when $A_{i,t} \geq 1\epsilon$, after node i uses 1ϵ of energy at time t, the node still has sufficient energy to transmit/receive another packet at time t. However, the primary interference does not allow a node to transmit/receive more than one packet at the same time slot. Thus, the earliest time the node can transmit/receive a packet is in the next slot, i.e., $T_i = t + 1$. For this case, in Eq. (3), $\alpha_{i,t} = 0$ and $\sigma = t + 1$. ■

Eq. (3) is used to compute T_i only when $t > 0$. For $t = 0$, we set $T_i = r_i$ because each battery is initially empty, and it takes r_i time slots for the harvester to achieve $A_{i,t} = 1\epsilon$.

B. Algo-1

We propose a greedy algorithm, called Algo-1, to schedule all non-interfering links at the earliest possible time slot. Specifically, it selects each non-interfering link with end nodes that have sufficient energy to transmit/receive one packet at the earliest time. The algorithm uses the conflict graph C_G to check for interfering links.

Lines 1-5 of Algo-1 initialize $t(i)$ to the last time slot in which node i draws energy from its battery, and $A_{i,t(i)}$ to the amount of energy that node i can use at time t to zero. It also sets the earliest time, T_i, node i has 1ϵ energy to r_i slots. Algo-1 starts at $t = 0$ and the battery is initially empty. *Lines 6-8* compute the earliest time each link (i, j) can be be scheduled, while *Lines 9-10* generate a set K to contain each link (i, j) that has the earliest activation time, and sort them in order of decreasing weight $w_{i,j}$. Links with equal $w_{i,j}$ are sorted in decreasing node degree of its end nodes and if it is a tie, links are sorted in increasing order of their node labels. *Line 11* sets t to the earliest slot. The loop in *Lines 12-26* aim to schedule each link $l_{i,j} \in K$ in order. Each slot in $\mathcal{S}$ is initially empty. *Line 13* ensures each selected link does not cause interference or is interfered by links already scheduled in slot t. The weight of each selected link $l_{i,j}$ is decremented by one. Once the weight reaches zero, it is removed from contention; see *Lines 15-18*. Further, Algo-1 uses Eq. (2) to recompute the available energy $A_{i,t}$ and $A_{j,t}$ at each end node of the selected link in *Lines 19-20*. The available energy is subtracted by one because a node needs 1ϵ energy to transmit/receive one packet. It then sets the last time the end nodes use energy to the current time in *Line 21*. Then, *COMPUTE_T$_i$()* in *Lines 22-23* uses Eq. (3) to recompute the T_i and T_j of the two end nodes. Finally, the function in *Line 24* recomputes the earliest time schedule of each link that has v_i or v_j as one of its end nodes. The steps from *Line 9* is repeated until all links have $w_{i,j} = 0$.

We now present an example. Consider the rWSN and conflict graph C_G in Figure 2. *Lines 1-5* of Algo-1 set $t(1) = t(2) = t(3) = t(4) = 0$, $A_{1,t(1)}$ = $A_{2,t(2)}$ = $A_{3,t(3)}$ = $A_{4,t(4)}$ = 0, and T_1 = r_1 = 2, T_2 = 6, T_3 = 5, and T_4 = 7. *Lines 6-8* compute $t_{1,2}$ = **max** (2, 6) = 6, $t_{3,1}$ = 5, and $t_{4,3}$ = 7. *Lines 9-10* obtain $K = \{l_{3,1}\}$, and *Line 11* sets $t = 5$. *Line 14* inserts $l_{3,1}$ into $\mathcal{S}[5]$, and *Line 15* reduces $w_{3,1}$ by one and hence it becomes zero. *Lines 19-20* compute $A_{3,5}$ = 0.2 and $A_{1,5}$ = 2, while *Line 21* sets $t(3) = t(1) = t = 5$. *Lines 22-23* then compute the earliest time nodes 3 and 1 have sufficient energy to transmit/receive one packet, i.e., $T_3 = 11$ and $T_1 = 6$. *Line 24* updates the earliest time that links can be scheduled, i.e., $t_{1,2}$ = **max** (6, 6) = 6, $t_{3,1}$ = 11, and $t_{4,3}$ = 7. *Line 27* repeats the steps from *Line 9* until all links have $w_{i,j} = 0$. Finally, Algo-1 produces the link schedule $\mathcal{S}$ in Figure 3b.Notice that there are 12 empty slots as nodes are unable to transmit/receive due to insufficient energy.

Proposition 3: The time complexity of Algo-1 is $O(W|E|^2)$, where $W = \sum_{(i,j)\in|E|} (w_{i,j})$.

Proof: Lines 1-5 take $O(|V|)$, while *Lines 6-8* require $O(|E|)$. *Lines 9-11* require sorting at most E links and thus take $O(E \ log \ E)$. The conflict graph C_G is represented as an adjacency matrix, and thus *Line 13* needs to access the matrix $|\mathcal{S}[t]|$ times or $O(|E|)$ to check for interference between a selected link (i, j) and the already scheduled links in $\mathcal{S}[t]$. *Lines 14 to 23* take $O(1)$ each. *Line 24* requires $O(|V|)$. Thus, the for loop in *Lines 12-26* take at most $O(|E|^2)$ since $|V| \leq |E|$. Finally, *Line 27* repeats *Lines 9-26* W times, and hence the time complexity of Algo-1 is $O(W|E|^2)$. ■

Algorithm 1 Algo-1: a greedy algorithm that schedules links according to earliest activation time first

Input: $G(V, E)$, r_i and b_i of each node $i \in V$, weight $w_{i,j}$ of each link $l_{i,j} \in E$, and conflict graph C_G
Output: Superframe $\mathcal{S}$ with the shortest length $|\mathcal{S}|$

```
1:  for each node $i \in V$ do
2:      $t(i) \leftarrow 0$
3:      $A_{i,t(i)} \leftarrow 0$
4:      $T_i \leftarrow r_i$
5:  end for
6:  for each link $l_{i,j} \in E$ do
7:      $t_{i,j} \leftarrow \mathbf{max}(T_i, T_j)$
8:  end for
9:  Put each node $l_{i,j}$ in $C_G$ with $\mathbf{min}\{t_{i,j}\}$ into a set $K$
10: Sort $K$ in order of (i) decreasing $w_{i,j}$, (ii) decreasing degree($i$ or $j$), and (iii) increasing NodeLabel($i$ or $j$)
11: $t \leftarrow \mathbf{min}\{t_{i,j}\}$
12: for each $l_{i,j} \in K$ do
13:     if NOT CONFLICT($l_{i,j}, \mathcal{S}[t]$) then
14:         $\mathcal{S}[t] \leftarrow \mathcal{S}[t] \cup l_{i,j}$
15:         $w_{i,j} \leftarrow w_{i,j} - 1$
16:         if $w_{i,j} = 0$ then
17:             remove node $l_{i,j}$ from $C_G$
18:         end if
19:         $A_{i,t} \leftarrow \mathbf{min}(b_i + 1/r_i, A_{i,t(i)} + (t - t(i)) / r_i) - 1$
20:         $A_{j,t} \leftarrow \mathbf{min}(b_j + 1/r_j, A_{j,t(j)} + (t - t(j)) / r_j) - 1$
21:         $t(i) \leftarrow t(j) \leftarrow t$
22:         $T_i \leftarrow$ COMPUTE_$T_i(t, i)$
23:         $T_j \leftarrow$ COMPUTE_$T_j(t, j)$
24:         UPDATE_$t_{cd}(T_i, T_j)$
25:     end if
26: end for
27: repeat Line 9-26 until all $w_{i,j} = 0$
```

V. Evaluation

We implemented Algo-1 in C++ and conducted our experiments on a computer with an Intel Core i7 CPU @ 3.4GHz and 16GB of memory. We first analyze the effect of energy harvesting time. Then, we study the effect of battery capacity. Finally, we evaluate the performance of Algo-1 as compared to the lower bound of $|\mathcal{S}|$. We consider arbitrary networks with 20 to 50 nodes randomly deployed on a 40 × 40 m^2 area. Each node has a transmit and interference range of 15 and 30 meters, respectively. Our results are averaged over 100 random node deployments.

A. Effect of Harvesting Time on $|\mathcal{S}|$

We first study the effect on $|\mathcal{S}|$ when using HUS or HSU, followed by the impact of network size.

1) HSU versus HUS: We consider various r_i values; namely, $1, 5, 10, 15, 20$, in a rWSN with 50 nodes. We randomly fixed the battery capacity b_i and link weight $w_{i,j}$, each to a value between 1 and 5. As shown in Figure 4, harvesting time significantly affects the link schedules in both models. Specifically, when r_i increases from 1 to 20, $|\mathcal{S}|$ jumps from 769 to 2995 slots in HUS and from 990 to 3744 slots in HSU, an increase of 289.5% and 278.2%, respectively. Note that when $r_i = 1$, each node in HUS has sufficient energy to transmit/receive one packet in any slot. Thus, the required number of slots is due to link interference only. The figure also shows that increasing harvesting time consistently creates longer $|\mathcal{S}|$, i.e., when $r_i = 5$, 10, and 15, HUS produces 870, 1506, and 2250 slots while HSU generates 1508, 2253, and 2998 slots respectively.

Figure 4 shows that the superframe length $|\mathcal{S}|$ when using HSU is longer than HUS. For example, when $r_i = 1$, HSU results in 221 more slots as compared to when using HUS; i.e., 28.74% longer. The results are consistent for other harvesting times; i.e., $r_i = 5, 10, 15, 20$. The difference between $|\mathcal{S}|$ ranges between 600 to 800 slots. HSU produces superframes that are 73.33%, 49.6%, 33.24% and 25.01% longer than those by HUS, respectively. HUS generates shorter superframes as compared to HSU because the former can directly use harvested energy without storing it first. In the following experiments, we only consider HUS because HSU generates similar trends but longer superframes.

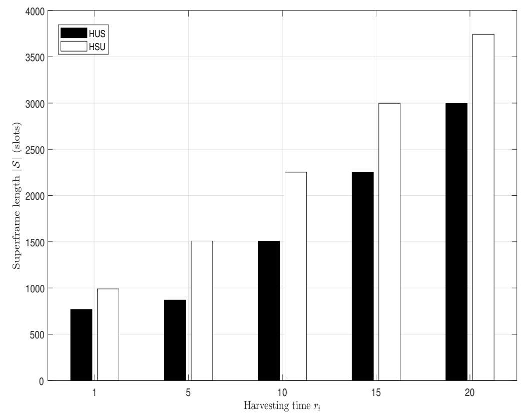


Figure 4. HUS vs HSU in rWSNs of 50 nodes

To further analyze the effect on harvesting time on $|\mathcal{S}|$, we classify slots in $\mathcal{S}$ into two categories: (i) all slots α in which the end nodes of links have sufficient energy, and (ii) all slots α in which the end nodes of at least one link has insufficient energy, and thus the link can be activated only at slot $\beta > \alpha$ after its end nodes have harvested energy. Thus, due to insufficient energy $|\mathcal{S}|$ is increased by $\beta - \alpha$ slots.

In this experiment we investigate the effect of insufficient energy for case (ii). As shown in Figure 5, when $r_i = 1$, the superframe length $|\mathcal{S}|$ of case (ii) is zero because each node has sufficient energy at any slot. When we increase r_i to 5, some links must be scheduled in later slots because their nodes have no energy. Thus, those links increase $|\mathcal{S}|$ by 47 slots from 815 slots. Further, Figure 5 shows that increasing the value of r_i will increase the number of slots under case (ii) because more nodes need longer time to harvest energy. We see that

$|\mathcal{S}|$ for this case is increased by 217, 711, and 1418 slots when $r_i = 10, 15, 20$, respectively.

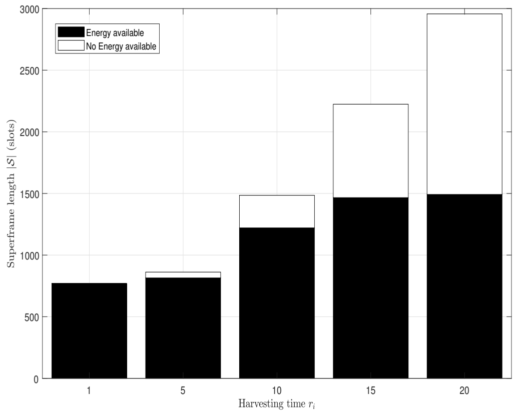


Figure 5. The effect of energy unavailability on $|\mathcal{S}|$

2) Effect of network sizes on $|\mathcal{S}|$: We consider a rWSN with 20 to 50 nodes with the harvesting time from 1 to 20. We set $b_i = 3$ and each link has weight $w_{i,j}$ = 3. As shown in Figure 6, energy harvesting time has a significant negative effect on $|\mathcal{S}|$ for all network sizes, i.e., increasing harvesting time results in longer superframes. More specifically, for rWSN with 20 nodes, when r_i increases from one to 20, the superframe length $|\mathcal{S}|$ jumps from 194 to 1288 slots, i.e., an increase of 563.9%. Similarly, for rWSN with 30, 40, and 50, there are 467.3%, 300%, and 283.5% increases in $|\mathcal{S}|$, respectively. The increase in superframe length $|\mathcal{S}|$ is because nodes need more time to harvest energy. We observe that for $r_i \geq 5$, the superframe length $|\mathcal{S}|$ increases almost linearly for each network size. More specifically, when r_i increases from five to 20, in an interval of five, for networks with 20, 30, 40, 50, the number of slots is increased on average by 320, 458, 545, and 698 slots, respectively. The standard deviation values range between 46.1 and 388. The increase in smaller networks is less than in larger networks because more nodes mean more links need to be scheduled. Also, more links will have to wait for sufficient energy before they can be activated. As shown in Figure 6, the result for dense networks with 100 nodes support our explanation that harvesting time constraint affects the superframe length significantly.

B. Effect of Battery Capacity on $|\mathcal{S}|$

We now study battery capacity and its impact on the superframe length $|\mathcal{S}|$. We set $w_{i,j}$ = 10. There are 50 nodes with r_i set to 1, 5, 10, 15 or 20. For each r_i value, we increase the battery capacity b_i from one to 20, in a multiple of five. From Figure 7, we see that increasing battery capacity has an insignificant effect on the superframe length $|\mathcal{S}|$. As an example, when $r_i = 5$, increasing b_i from one to 20 reduces $|\mathcal{S}|$ only from 3124 to 3048 slots, a decrease of 2.5%. Note that the standard deviation values range between 592.2 and

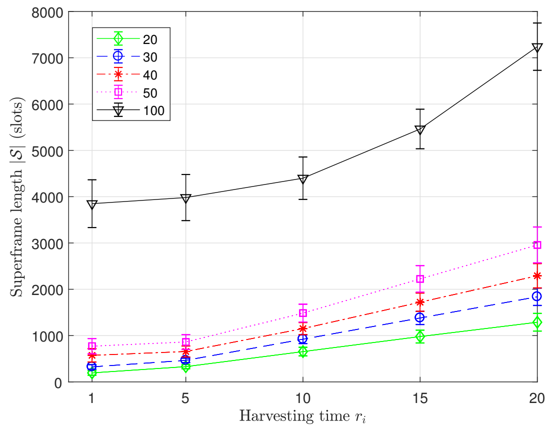


Figure 6. Effect of network sizes with varying r_i on $|\mathcal{S}|$

1192. The main reason for the insignificant effect of battery capacity on $|\mathcal{S}|$ is explained as follows.

Consider a node i with $r_i > 1$ that needs to activate multiple links to/from its m neighbors that have a harvesting time of at least $r_j > r_i$. Assume one neighbor has harvesting time r_j while each of the others has a harvesting time of $r_j + k$, $r_j + 2k, \cdots, r_j + mk$, for $k \geq 0$. In this case, node i needs to activate m consecutive links, one every k slots. Assume the current time is $t = 0$. One can observe that $|\mathcal{S}|$ is minimized if node i can accumulate energy in its battery at time $t = r_j$ such that it is sufficient to activate one link every k slots. Specifically, to minimize $|\mathcal{S}|$, it is necessary that (i) $r_i \leq k$ or, (ii) when $r_i > k$, b_i must contain more than 1ϵ energy at time r_j, i.e., $b_i > 1$. Thus, one can see that $b_i > 1$ is needed for case (ii). As an example, node 2 in Figure 1b receives three consecutive packets from its neighbors in three consecutive slots, i.e., $k = 1$ and $r_2 = 2 > k$ and increasing b_i from one to three reduces $|\mathcal{S}|$ from nine to seven. It is important to note that node i can accumulate more than 1ϵ energy only if $r_i < r_j$ for each neighbor j; in the example r_2 is smallest. Otherwise, if $r_i > r_j$, battery i will never accumulate more than 1ϵ of energy because node i will consume it immediately to activate one of its links. Thus, for this case $b_i = 1$ is sufficient, i.e., larger b_i does not reduce $|\mathcal{S}|$.

C. Effectiveness of Algo-1

To analyze the effectiveness of Algo-1, we compute the ratio between its generated $|\mathcal{S}|$ and the lower bound $|\mathcal{S}_{\mathcal{E}}|$ in Eq. (1); a smaller ratio means the algorithm has better performance. We set $w_{i,j} = [1, 5]$ and $b_i = [1, 5]$.

As shown in Figure 8, for rWSN with 20 nodes and $r_i = 1$, i.e., when nodes always have energy, Algo-1 achieves a performance ratio of 2.88. However, for $r_i \geq 2$, i.e., when nodes have energy harvesting constraint, Algo-1 performs better, with an average performance ratio of 1.23. Specifically, for $r_i = 2$ and $r_i \geq 5$, with $|V| = 20$, Algo-1 has an average performance ratio of 1.49 and 1.03, respectively. Further, it shows that increasing network size from 20 to 50 worsen performance due to more interference. Note that the standard

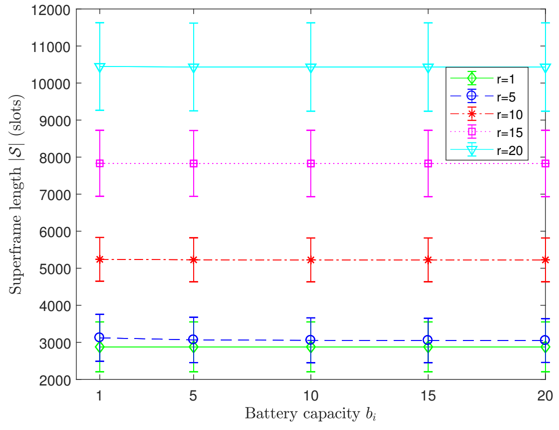


Figure 7. Effect of b_i on $|\mathcal{S}|$ in network with 50 nodes

deviation values in Figure 8 range between 0 and 1.01.

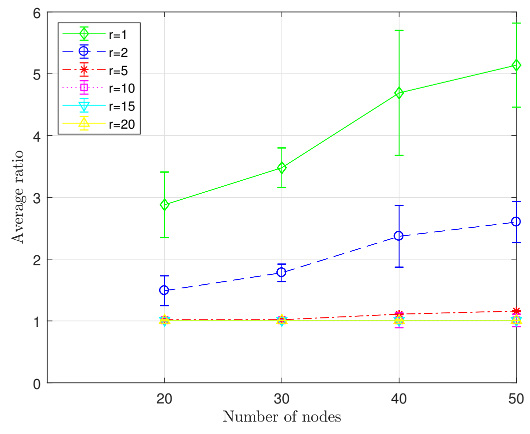


Figure 8. The performance of Algo-1

VI. Conclusion

This paper considers link scheduling in rWSNs where sensor nodes use the HUS battery recharging model. It presents a novel problem called LSHUS that aims to produce a superframe $\mathcal{S}$ with the minimum length. It also presents a greedy algorithm, called Algo-1, to solve LSHUS. Extensive simulations show that: (i) increasing harvesting time increases superframe length $|\mathcal{S}|$ by up to 563.9%, (ii) using HUS model produces up to 42.31% shorter superframes as compared to using HSU, (iii) increasing battery capacity slightly reduces $|\mathcal{S}|$, i.e., only by up to 2.5%, (iv) When nodes have energy harvesting time constraint, Algo-1 produces superframes that are on average 1.23 times longer than the lower bound of superframe length. As future works, we aim to consider battery charging efficiency, leakage factor, and develop a distributed solution.

Acknowledgment

The first author acknowledges support from Indonesia Lecturer Scholarship (BUDI) by Indonesia Endowment Fund for Education (LPDP). This scholarship is a collaboration between the Ministry of Research, Technology, and Higher Education with the Ministry of Finance, Republic of Indonesia.

References

[1] [Online]. Available: https://www.eol.ucar.edu/isf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf

[2] L. Atzori, A. Iera, and G. Mirabito, “The internet of things: A survey,” *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[3] J. A. Bondy and U. S. R. Murty, *Graph Theory With Applications*. New York: American Elsevier, 1976.

[4] O. L. Dipak Surie and T. Pederson, “Wireless sensor networking of everyday objects in a smart home environment,” in *Proc. Int. Conf. Intelligent Sensors Sensor Networks Inform. Process. (ISSNIP)*, Sydney, NSW, Australia, Dec. 2008, pp. 189–194.

[5] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, “Impact on interference on multi-hop wireless network performance,” in *ACM MobiCom*, San Diego, CA, USA, Sep. 2003, pp. 66–80.

[6] S. Kapoor and S. R. B. Pillai, “Distributed scheduling schemes in energy harvesting multiple access,” *IEEE Wireless Commun. Lett.*, vol. 6, no. 1, pp. 54–57, Feb. 2017.

[7] H. Kim, Y. Tadesse, and S. Priya, *Piezoelectric Energy Harvesting, in Energy Harvesting Technologies*, S. Priya and D. J. Inman, Eds. New York: Springer, 2009.

[8] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, and M. Welsh, “Wireless sensor networks for healthcare,” *Proc. IEEE*, vol. 98, no. 11, pp. 1947–1960, Nov. 2010.

[9] M.-L. Ku, W. Li, Y. Chen, and K. J. R. Liu, “Advances in energy harvesting communications: Past, present, and future challenges,” *IEEE Commun. Surveys Tutorials*, vol. 18, no. 2, pp. 1384–1412, 2016.

[10] M. R. Lenka, A. R. Swain, and M. N. Sahoo, “Distributed slot scheduling algorithm for hybrid csma/tdma mac in wireless sensor networks,” in *Proc. IEEE Int. Conf. Netw. Architecture Storage (NAS)*, Long Beach, CA, USA, Aug. 2016.

[11] J. Liu, H. Dai, and W. Chen, “On throughput maximization of time division multiple access with energy harvesting users,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2457–2470, Apr. 2016.

[12] V. S. R Rajesh and P. Viswanath, “Capacity of fading gaussian channel with an energy harvesting sensor node,” in *Proc. IEEE GLOBECOM*, Kathmandu, Nepal, Dec. 2011, pp. 1–6.

[13] S. Ramanathan, “A unified framework and algorithm for channel assignment in wireless networks,” *Wireless Netw.*, vol. 5, no. 2, pp. 81–94, Mar. 1999.

[14] M. A. Razzaque and S. Dobson, “Energy-efficient sensing in wireless sensor networks using compressed sensing,” *Sensors*, vol. 14, no. 2, pp. 2822–2859, 2014.

[15] S. Sudevalayam and P. Kulkarni, “Energy harvesting sensor nodes: survey and implications,” *IEEE Commun. Surveys Tuts.*, vol. 13, no. 3, pp. 443–461, 2011.

[16] J. Suhonen, M. Kohvakka, V. Kaseva, T. D. Hämäläinen, and M. Hännikäinen, *Low-Power Wireless Sensor Networks: Protocols, Services and Applications*. New York: Springer, 2010.

[17] G. Sun, G. Qiao, and L. Zhao, “Efficient link scheduling for rechargeable wireless ad hoc and sensor networks,” *EURASIP J. Wireless Commun. Netw.*, vol. 1, no. 1, pp. 1–14, Dec. 2013.

[18] J.-P. Vasseur and A. Dunkels, *Smart object hardware and software, in Interconnecting Smart Objects with IP*. San Francisco: Morgan Kaufmann, 2010.

[19] L. Wang, K.-W. Chin, S. Soh, and T. He, “A novel flow-aware fair scheduler for multi transmit/receive wireless networks,” *IEEE Access*, vol. 5, pp. 10 456–10 468, 2017.

[20] T. Wang and C. G. Cassandras, “Optimal control of multibattery energy-aware systems,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 5, pp. 1874–1888, Sep. 2013.

[21] F. Yuan, S. Jin, K.-K. Wong, Q. T. Zhang, and H. Zhu, “Optimal harvest-use-store design for delay-constrained energy harvesting wireless communications,” *J. Commun. Netw.*, vol. 18, no. 6, pp. 902–912, Dec. 2016.

[22] F. Yuan, Q. T. Zhang, S. Jin, and H. Zhu, “Optimal harvest-use-store strategy for energy harvesting wireless systems,” *IEEE Trans. Wireless Commun.*, vol. 14, no. 2, pp. 698–710, Feb. 2015.