Available online at www.sciencedirect.com
ScienceDirect
Sustainable Computing: Informatics and Systems 00 (2014) 1–17
Sustainable Computing

Worst-Case Performance Guarantees of Scheduling Algorithms Maximizing Weighted Throughput in Energy-Harvesting Networks

Huangxin Wang

Department of Computer Science, George Mason University. Fairfax, VA 22030

Jean X. Zhang

Department of Accounting, School of Business, Virginia Commonwealth University. Richmond, VA 23284

Fei Li

Department of Computer Science, George Mason University. Fairfax, VA 22030

Abstract

Energy harvesting has recently emerged as a technique to enable longer operating time of sensor networks. However, due to harvesting energy's not-completely-predictable stochastic nature, some packets may still fail to be transmitted due to insufficient energy supply. Also, packets in sensor networks are usually associated with sensitive time-critical information. Based on these observations, we theoretically study algorithms scheduling weighted packets with deadlines in energy-harvesting networks. In our model, packets arrive in an online manner, each packet has a value representing its priority and a value representing its deadline. Harvesting energy is gathered over time and transmitting one packet takes a unit of energy. The objective is to maximize the total value of the packets sent, subject to energy and deadline constraints. In this paper, we design both offline and online algorithms maximizing weighted throughput. We analyze these algorithms' performance guarantees against their worst-case scenarios and empirically compare them with the conventional and classic scheduling algorithms. The simulation results show that our online algorithms have far better performance than conventional ones.

Keywords: online scheduling, packet scheduling, competitive analysis, harvesting energy

1. Introduction

In wireless sensor networks, each sensor is equipped with a battery with limited capacity to provide its power supply. Due to the inconvenience of replacing batteries or recharging them using electricity cables, wireless sensors often have very limited operating time and the whole network's lifespan heavily depends on its "weakest" functional node. To make a sensor network sustainable in operation, energy harvesting is a technique introduced to collect and convert energy (such as solar or wind energy) from the environments to replenish the sensors' power supply. This technique enables wireless sensor networks to work for a longer period or even last forever if power consumption can be carefully managed [1]. Several research work including [1, 2] has studied various models of energy-harvesting sensor networks and proved the feasibility and benefits of using solar energy to fuel the sensor networks.

Email addresses: hwang14@gmu.edu (Huangxin Wang), jxzhang@vcu.edu (Jean X. Zhang), lifei@cs.gmu.edu (Fei Li)

Scheduling packets in energy-harvesting sensor networks is a significant research topic. Packets scheduled in sensor networks, particularly for those real-time applications, are usually associated with time-critical information and thus any packet forwarded after a particular time frame may become meaningless to its receiver. Also, different packets may embed different amount of information and hence packets to be transmitted should have various priorities. For example, for a temperature-monitoring energy-harvesting sensor node, packets which embedded higher temperature value are considered more important, as a higher temperature indicates a more possible overheating or fire [3]. Based on these properties, it is natural to consider scheduling packets with priorities and deadlines in energy-harvesting wireless sensor networks rather than scheduling uniform-value packets without time constraints (see [4, 5, 6, 7] and the references therein). Furthermore, people have observed that in the platform where there exists energy to be harvested in the sensor networks, conventional time-critical packet scheduling algorithms, such as EDF (Earliest Deadline First) or ALAP (As Late As Possible), are not energy-efficient and can no longer work well due to the stochastic nature of harvesting energy [4]. Therefore, it is important to develop novel energy-efficient packet scheduling algorithms such that the wireless sensor networks are able to operate in an energy-efficient and sustainable way with harvesting energy.

In this paper, we study the problem of scheduling packets with values and deadlines for a harvesting sensor node. The packet scheduler runs within the wireless network card of each sensor node. Transmitting a packet takes a unit of energy. As the battery power is limited and it is hard to predict completely the future harvesting energy supply, it is possible that some real-time packets cannot be delivered successfully upon packet overflow or energy constraints. Thus, it is natural to deliver packets with more "importance" by efficiently using the limited energy. In this model, each packet has a value representing its priority. At the same time, packets forwarded with critical time requirements are associated with deadlines. The objective is to maximize weighted throughout, defined as the total value of the packets sent successfully by their deadlines. We consider both offline and online settings.

It is easy to see the dilemma of designing online algorithms for this model when the information of the released packets or the harvesting energy is unclear or unable to be predicted precisely beforehand: On one hand, if we use the deposited energy in the battery to deliver low-value packets that are currently pending for delivery, then we may fail to transmit the potential high-value packets which will arrive in the future due to lacking of energy. On the other hand, if we reserve the deposited energy for any "potential but unpredictable" future high-value packets without sending the pending low-value packets right now, then we will lose the values from these low-value packets due to their time constraints and there may not exist such high-value packets to appear in the future for us to reimburse the loss. To overcome this dilemma, a line of prior research [8, 3, 9] makes various assumptions on the distributions of the harvesting energy and/or the distributions of the packets released. With these assumptions, expected performance of the proposed algorithms for these models is evaluated. Unfortunately, expected performance only guarantees an algorithm's weighted throughput in the long run only when the stochastic assumptions made are correct. For a finite short time period (especially in wireless sensor networks), worst-case performance guarantee independent of any stochastic assumptions over the input instance is more of significance as it reveals the information fluctuation from the micro-level perspective. At the same time, for the purpose of analytical study, worst-case performance guards are demanded as well.

In this paper, we generalize the models studied in [8, 3, 9] by considering scheduling time-critical packets in energy-harvesting sensor networks. We remark here that we do not address the wireless signal interference constraints as this model (as well as those studied in [8, 3, 9]) is built upon the coordination among various wireless nodes. We focus on a single wireless sensor node and all the time slots in which it is allowed to send packets. We develop both offline and online algorithms for this generalized model and conduct theoretical and empirical studies. Particularly, we design competitive online algorithms which do not rely on any assumptions made on the input information. Note that studying an algorithm's worst-case performance is necessary for the model whenever those input assumptions are not easy to get or to verify. Since the solutions given in [8, 3, 9] are based on stochastic assumptions made over energy arrival patterns, packet arrival patterns, or packet value distributions, therefore these solutions are not applicable to our model. The main contributions of this paper are summarized as below.

- We generalize the models proposed in [8, 3, 9] by incorporating packet deadlines and removing any assumptions made over packet value distributions, packet arrival patterns, or harvesting energy arrival patterns.
- We present an exact offline algorithm.
- We provide a lower bound of competitive ratio for deterministic online algorithms. We also provide a family of

optimal deterministic online algorithms.

- We provide a randomized algorithm, against a non-idling oblivious adversary.

  Our efforts show that randomization helps in maximizing weighted throughput in the online setting. Note that this randomized algorithm does not depend on the stochastic features of the input. The randomization is internal to the algorithm.

- We experimentally evaluate the performance of the proposed offline and online algorithms against some known work.

The rest of the paper is organized as follows. In Section 2, we present a system model and a problem statement motivated by scheduling weighted packets with deadlines in energy-harvesting networks. In Section 3, we show an optimal offline algorithm in polynomial-time. In Section 4, we study online algorithms and compare their performance against worst-case scenarios using competitive ratios. We propose two heuristic deterministic online algorithms and one randomized algorithm. We prove that randomization helps in improving competitive ratios in non-idling settings. In Section 5, we empirically test our algorithms against the conventional scheduling algorithms. Related work is addressed in Section 6 and Section 7 concludes this paper.

2. System Model

We consider a packet scheduling problem raised in an energy-harvesting wireless sensor node which has battery capacity constraints. In this model, time is assumed discrete. In each time slot, some (may be 0) packets arrive at this node. For each packet p, it has a release time $r_p \in \mathbb{N}$ indicting the time from which p is available for the sensor node to forward, a deadline $d_p \in \mathbb{N}$ by which p should be sent before its information expires, and a value $v_p \in \mathbb{R}^+$ contributed to our objective if p is transmitted successfully by d_p. In each time slot t, a sensor node receives harvesting energy $h(t) \in \mathbb{N}$ and this node has an embedded battery whose capacity is C — at any time, the battery's power cannot exceed $C \in \mathbb{N}$. It is assumed that there is no battery wear or fatigue such that the energy capacity C of a wireless node remains constant throughout the lifetime of the network. Due to the battery capacity constraint, at time t, if the harvesting energy $h(t)$ is more than what the battery can accommodate, then the battery will be charged to its maximum capacity C and any remaining energy will be lost immediately. Note that if no energy arrives at time slot t, then the harvesting energy $h(t)$ is defined as 0. Without loss of generality, in each time slot, the node forwards at most one packet upon sufficient energy left in its battery. Delivering one packet takes a constant energy cost 1. We gain a value v_p only if the packet p is forwarded by the node by its deadline d_p. The objective is to maximize the total value gained by delivering packets, subject to the packet deadline constraints and the harvesting energy constraints.

The model that we study generalizes the ones proposed in [8, 3, 9] by incorporating packet deadlines and removing artificially-enforced stochastic assumptions over the packets and the harvesting energy. We develop both offline and online algorithms. In designing online algorithms, there are two constraints: (1) the packet deadline constraints (all the released packets may not be delivered successfully even if energy is sufficient), and (2) the energy constraints (all the released packets may not be delivered successfully unless energy is sufficient). In this paper, we emphasize on studying the impact of insufficient harvesting energy and thus, we add the following restriction in the problem setting. This assumption has been used in real-time scheduling literature under hard energy constraints (see [10] and its references therein).

Assumption 1. *Assume that the input instance is schedulable, that is, all the packets in the input can be scheduled by their deadlines subject to sufficient energy supply.*

We remark here that given no energy restrictions, we can simply use EDF to check whether an input instance is schedulable or not.

Actually, the model that we study can be regarded as an assignment problem. A solution to this problem is to specify in which time slot to send or not to send which packet subject to the given time and energy constraints. In this paper, we design combinatorial algorithms for this problem, considering in both the offline and the online settings in Section 3 and Section 4 respectively. We remark that this model in the offline setting can also be solved optimally via solving a 0-1 integer program. For reference, the notations used in this paper are listed in Table 1.

notation	meaning
I	an input instance
C	battery capacity
$e(t)$	battery energy balance at time t
$h(t)$	harvesting energy at time t
r_p	a packet p's release time
d_p	a packet p's deadline
v_p	a packet p's value

Table 1: Notations used in this paper and their meanings

3. Optimal Offline Algorithm

In this section, we present an optimal offline algorithm named OPT for the model that we study, assuming that all the input information is known beforehand. We point out here that packet scheduling is essentially an online problem and thus, an optimal offline algorithm is not a practical solution to be employed by a wireless sensor node. However, OPT can provide us a benchmark to compare the performance of various online algorithms.

Let I denote an input instance, OPT denote an optimal offline algorithm, and O denote the set of packets scheduled by OPT. Given a set of packets S, let EDF(S) denote an EDF schedule of the packets in S. We define a *non-idling* algorithm which sends a pending packet whenever it has sufficient energy. At first, we use the following toy example to illustrate the difficulty of designing an optimal algorithm for the model.

Example 1. *Assume there are* 2 *packets* p_1 *and* p_2 *with* $r_{p_1} = 1$*,* $d_{p_1} = 3$*,* $r_{p_2} = 2$ *and* $d_{p_2} = 2$*. The packet* p_2 *has to be sent in time slot* 2 *or otherwise discarded. Assume* $C = 1$ *and the battery is fully charged at time* 1*. One unit of harvesting energy arrives at time* 3 *(*$h(3) = 1$*). For this instance, EDF will schedule* p_1 *at time* 1 *and fails to schedule* p_2 *due to energy constraints. An optimal schedule becomes idle at time* 1*, schedules* p_2 *at time* 2 *and* p_1 *at time* 3*, achieving a total throughout of* 2*.*

Note that in designing OPT, we not only need to find the set O but also to specify the time slots in which the packets in O are scheduled. Example 1 shows that EDF(O) may not be OPT and OPT may be *idle* (not transmitting any pending packet even though the energy supply is sufficient) at some time slots. We consider all time steps of OPT and partition them into three categories:

- *idle period*: a maximal time interval in which there are pending packets and sufficient energy, but no pending packets are transmitted
- *busy period*: a maximal time interval of sending packets
- *sleep period*: a maximal time interval in which there are no pending O-packets or there exist pending O-packets but no energy.

One observation on OPT addresses the following feature.

Lemma 1. *If OPT has one idle period at some time, then this idle period's following period must be a busy one. If OPT has a busy period and its following period is an idle but not a sleep one, then all the packets scheduled in this schedule are scheduled at their latest time steps (that is to say, they cannot be delayed further in scheduling without hurting the gained weighted throughput).*

Proof. The first part of Lemma 1 is easy to prove as there is no energy consumed in an idle period and thus, its following period must be a busy one. We use a contradiction method (specifically, an exchange argument) to prove the second part of Lemma 1. Assume the second claim in the lemma is not true, then OPT can shift its busy period as a whole block to a later time slot without losing the weighted throughput gained. □

Recall that we target on finding the matching between the O-packets and the time slots in which they are sent, as well as the matching between the time slots and the energy assigned over them. Based on Lemma 1, we understand that if we have a candidate set of m packets to be scheduled (for example, we have the set O), then we can create a schedule in which all the packets are scheduled at their latest feasible time steps. This procedure can be implemented through any sorting algorithm over the packet deadlines upon the energy constraints, with running time $O(m \log m)$. The remaining question for us to answer is to find out O. We construct O in an iteratively way — Initially, we have an empty set and we add O-packets into this set one by one. If we apply this approach, we have to show that any packet we ever put into the feasible set must be an O-packet and thus, we need to prove that the model we are studying is a matriod.

Lemma 2. *Given C is unlimited, the model that we study satisfies the properties of a matriod, having the hereditary property and the exchange property.*

The proof of Lemma 2 is a standard one using the exchange argument, along with the feature described in Lemma 1.

Now, we are ready to describe an optimal offline algorithm. Let P' define an optimal packet set. Initially, P' is an empty set, and we construct P' by iteratively adding packets one by one in the non-decreasing order of packet weights. A packet which can be feasibly scheduled with the existing packets in P' is added into P', otherwise, it will be discarded. Denote an input instance as $I = \{p_1, p_2, \cdots p_n\}$. We have an algorithm in Algorithm 1.

Algorithm 1 OPT (I)

```
Initialize the set of packets to be sent P' = ∅.
Initialize the set of packets to be considered P = I = {p_1, p_2, ··· , p_n}.
Sort all packets in P in decreasing order of the values.
while |P'| ≤ n and P ≠ ∅ do
   remove the maximum-value packet p from P;
   if the set P' ∪ {p} can be feasibly scheduled, upon their deadline constraints and the energy constraints then
      insert packet p into P' and update P' as P' ∪ {p}.
   end if
end while
return P'
```

Using the matriod property of our model (Lemma 2), we immediately have the following result.

Theorem 1. *Algorithm OPT is optimal and it runs in polynomial-time.*

We remark here that there exist dynamic programming based algorithms for this problem and the existing ones are not polynomial-time solutions (see [11] and the references therein). As a complimentary result, we state here that our model in the offline setting can also be solved optimally via solving a 0-1 integer program even if C is a bounded number.

Consider an input instance. Let x_{tp} be an indicator variable associated with the event that a packet p is scheduled to be sent in time slot t. We have $x_{tp} = \begin{cases} 1, & \text{if } p \text{ is scheduled at time } t \\ 0, & \text{otherwise} \end{cases}$ and $x_{tp} + x_{tq} \leq 1, \forall p \neq q$.

The energy consumption at time t is $\sum_p x_{tp}$. Define $e(t)$ as the energy budget of a node at time t. Given the limit of the battery capacity, the change of battery energy balance from time t to time $t+1$ can be represented as below. (Recall C is the battery capacity.)

$$e(t+1) = \min\left\{ e(t) + h(t) - \sum_p x_{tp}, C \right\} \tag{1}$$

Note that Equation (1) can be expressed as $e(t+1) \leq e(t) + h(t) - \sum_p x_{tp}$ and $e(t+1) \leq C$ with incorporating $\sum_t e(t)$ in the maximization objective. In our following formulation, we stick with Equation (1). The model can be solved using a 0-1 integer program $\mathcal{IP}$ shown as below.

$$
\begin{aligned}
&\mathcal{IP}: \\
&\max \sum_t \sum_p x_{tp} \cdot v_p \\
\text{subject to } &\sum_t x_{tp} \le 1, && \forall p \\
&\sum_{t<r_p} x_{tp} = 0, && \forall p \\
&\sum_{t>d_p} x_{tp} = 0, && \forall p \\
&\sum_p x_{tp} \le 1, && \forall t \\
&\sum_p x_{tp} \le e(t) + h(t), && \forall t \\
&e(t+1) = \min\left\{ e(t) + h(t) - \sum_p x_{tp}, C \right\} && \forall t \\
&x_{tp} = \{0, 1\}, && \forall t, p
\end{aligned}
$$

In the above set of constraints, the first inequality shows that one packet can be scheduled at most in one time slot. The second and the third inequalities show that a packet can not be scheduled any time before it arrives or after it expires. The fourth inequality indicates that in one time slot at most one packet can be transmitted. The fifth inequality shows the harvesting energy constraints, which enforce that a node forwards a packet only if there is enough remaining energy. The sixth inequality is derived from the battery capacity constraints such that the battery can store at most C units of energy and any extra energy will overflow and be wasted. Also, each step corresponds a unit of energy consumption if a packet is sent. The last inequality is on the 0-1 indicator variables that we have defined. Note that there is no known polynomial-time algorithm solving a 0-1 integer program unless P = NP [12].

4. Online Algorithms

Packet scheduling is essentially an online problem. In this section, we design competitive algorithms and analyze their competitive ratios in the online setting. For such an online problem, we discard any stochastic assumptions made over packet value distributions, harvesting energy arriving sequences or packet arriving sequences. *Competitive ratio* [13] is a classical and widely used metric to evaluate online algorithms. This measure is used to compare the output of an online algorithm with that of an (impractical) optimal offline algorithm which is assumed to know all the input information (including the harvesting energy arriving sequences and packets arriving sequences) beforehand to make its decisions.

Definition 1 (Competitive ratio [13]). *A deterministic (respectively, randomized) online algorithm ON (ON_d) is called k-competitive if its (respectively, expected) weighted throughput of any instance is at least $1/k$ of the weighted throughput of an optimal offline algorithm. The optimal offline algorithm is also called (respectively, oblivious) adversary. Let OPT(I) denote the optimal offline solution of an input I. For a deterministic online algorithm ON,*

$$k := \max_I \frac{OPT(I) - \delta}{ON(I)},$$

where δ is a constant and $ON(I)$ is ON's output of an input I. For a randomized online algorithm ON_d,

$$k := \max_I \frac{OPT(I) - \delta}{E[ON_d(I)]},$$

where δ is a constant and $E[ON_d(I)]$ is ON_d's expected output of an input I, with its decision selected from a distribution d created by the online algorithm.

Note that unlike the stochastic algorithms which rely on the statistical assumptions on the input sequences, competitive online algorithms guarantee the worst-case performance in any given finite time frame.

It is a folklore that EDF is optimal in scheduling packets with the same value. If packets have distinct values, this online problem of optimizing weighted throughput has been open and become more interesting and complicated [14]. Here, we take packet values into consideration and address weighted throughput. Let $v_{\max}$ and $v_{\min}$ denote the maximum value and the minimum value of a packet in the input sequence I respectively.

Theorem 2. *The lower bound of competitive ratio for deterministic online algorithms can be up to* $v_{\max}/v_{\min}$*, even if the harvesting energy is sufficient to send up to* $1/2$ *of the packets released.*

Proof. To prove Theorem 2, we provide an instance. We set the harvesting energy arriving at $2k-1$ and $2k$ slots as 1 and 0 respectively, $\forall k = 1, 2, \ldots$. Let the initial battery energy be 0. The battery capacity $C = 1$. Let an optimal offline algorithm (also called the *adversary*) be ADV. We use (v, d) to denote a packet with value v and deadline d. Let a deterministic online algorithm be ON.

In the first time step, a packet $(v_{\min}, 1)$ is released. The adversary keeps releasing a packet $(v_{\min}, i)$ at time slot i until one of the two events happens: (1) ON picks up a packet $(v_{\min}, k)$ at time k to send, or (2) the adversary has released a packet with value $v_{\min}$, and ON has not picked it up to send.

Note that packet $(v_{\min}, i)$ can only be sent at time i or otherwise lost.

Assume the first case occurs. Assume ON picks a packet $(v_{\min}, 2k-1)$ to schedule at time k. Then ADV releases a packet $(v_{\max}, 2k)$ at time $2k$. Since the harvesting energy arriving at time $2k-1$ has been used by ON to send the packet $(v_{\min}, 2k-1)$, ON has no energy left to send the packet $(v_{\max}, 2k)$. After releasing the packet $(v_{\max}, 2k)$, ADV releases nothing. Overall, ADV gains a value $v_{\max}$ and ON gains a value $v_{\min}$. The competitive ratio is $v_{\max}/v_{\min}$.

Assume the second case occurs. ADV stops releasing new packets and it schedules the packet $(v_{\min}, 1)$ at time 1. Thus ON sends nothing and gains throughput 0. The competitive ratio is $v_{\min}/0 = \infty$. □

From Theorem 2, we know that no deterministic online algorithm ON has a constant competitive ratio. As the example shown in Theorem 2, there are two reasons accounted for this pessimistic analytical result. On one hand, if ON always transmits packets whenever energy is sufficient regardless of the packet values, then energy may be used up by the low-value packets resulting in the high-value packets unable to be transmitted. On the other hand, if ON is too conservative in using energy to transmit low-value packets, then an optimal offline algorithm will collect low-value packets anyhow. Therefore, ON has to take into account both the packets' deadlines and the packets' values when making its decisions about which packet to send.

We now design deterministic online algorithms with the upper bounds of competitive ratio. We consider *non-idling* online algorithms which send a pending packet whenever they have sufficient energy. The only flexibility for such a non-idling online algorithm is to specify which packet to send, subject to the deadline constraints and the values of the packets pending in the energy-harvesting sensor node. We have the following result on a *rational online algorithm*, which will not send a less-value packet with a later-deadline in each time step.

Theorem 3. *Any rational deterministic non-idling online algorithm achieves the optimal competitive ratio* $v_{\max}/v_{\min}$.

Proof. We use a charging scheme to prove Theorem 3. Let ADV denote an adversary. ADV is allowed to charge its energy cost and gain of weighted throughput in a time step earlier. Let ON denote a deterministic non-idling online algorithm. In order to prove Theorem 3, in each time step, we only need to show the following three invariants:

1. ON has no less energy left than ADV does.
2. The set of packets pending for ON's buffer is a superset of that for ADV.
   This invariant implies that if ON has no pending packets to send, then ADV has no pending packets as well.
3. The amortized value gained by ON is no less than $v_{\min}/v_{\max}$ times of what ADV's gains.

It is easy to see that the above three invariants hold at the beginning of schedules (time 0). Now, we inductively prove that they hold for any time step. Theorem 3 holds when these invariants hold at the end of schedules. At time t, assume ON and ADV satisfy these invariants. Then if ON sends any packet, ON consumes one unit of energy. We have two cases:

Assume ADV sends one packet at time t. As ADV has one packet to send, then ON has at least one packet pending in the buffer for it to send, due to the second invariant. Both ADV and ON spend one unit of energy in this step.

- Assume ON and ADV send the same packet.

  Then the amortized gain ratio is 1 and the invariants hold immediately.

- Assume ON and ADV send different packets and ON has more amortized value than ADV achieves.

  Then we charge two packets' values to ADV in this step — the one sent by ON and the one sent by ADV. The competitive ratio of amortized gains is bounded by 2. The invariants hold.

- Assume ON and ADV send different packets and ON has less amortized value than ADV achieves.

  Then the packet that ON sends should not be pending for ADV (otherwise, ADV should select this packet to send due to packet deadline constraints as ON is rational). ON gains a value $\geq v_{\min}$ and ADV gains a value $\leq v_{\max}$. The ratio of amortized gain between ADV and ON is bounded by $v_{\max}/v_{\min}$. The invariants hold.

Assume ADV sends nothing and it reserves one unit of energy, if any, to send some packet in a later time step. Then we charge ADV that the value of the packet, which is sent in a later time step, in this step. We also charge ADV that unit of energy in this step. The ratio of the amortized gains between ADV and ON is bounded by $v_{\max}/v_{\min}$ as well since ON gains a value $\geq v_{\min}$ and ADV gains a value $\leq v_{\max}$.

In all cases, the above three invariants hold and thus, Theorem 3 is proved. □

Among the family of deterministic non-idling online algorithms, we can carefully select one packet to send in one time step, subject to the packet deadlines and values. We note that transmitting a packet whenever energy is sufficient will not result in future packet overflow (see Assumption 1). Naturally, we have a greedy approach or a semi-greedy one to select a packet to schedule. The simplest online algorithm is to consider packet values alone — sending the maximum value packet available at each time slot whenever energy is available. We name this algorithm GREED.

Algorithm 2 GREED

```
for time slot i = 1, 2, ··· do
    if energy is available (i.e., e(t) + h(t) ≥ 1) then
        schedule the maximum value pending packet.
    end if
end for
```

Note that only packet values play in the algorithm GREED while only packets' deadline play in the algorithm EDF. Actually, we can integrate GREED and EDF by taking packet values and packet deadlines into consideration at the same time. A semi-greedy approach is to send one packet with a sufficiently large value and with a sufficiently earlier deadline. With this concern, we propose a deterministic online algorithm EDF_α. We use p_l to denote the packet with the highest (largest) value at time t. Let the earliest deadline (first) pending packet be p_f. We either schedule p_f or another packet p which satisfies $v_p \geq \{\alpha \cdot v_{p_f}, v_{p_l}/\alpha\}$. Note that GREED is EDF_1.

The following simple example illustrates how EDF_α could make balance between packet values and packet deadlines, and thus outperforms both EDF and GREED. We use (r_p, d_p, v_p) to denote a packet p with arriving time r_p, deadline d_p and value v_p. Assume there are 3 units of energy available at time slot 1 and there are no harvesting energy arrivals over time. Assume there are 4 packets arriving over time: $p_1 = (1, 1, 10)$, $p_2 = (1, 2, 20)$, $p_3 = (1, 3, 21)$, and $p_4 = (1, 4, 22)$. Then EDF will schedule p_1, p_2, and p_3 at time slots 1, 2, and 3 respectively and fails to schedule p_4 due to energy constraints, achieving a total revenue of 51; GREED will schedule p_4 and p_2 at time slots 1 and 2 respectively, and fail to schedule p_1 and p_3 due to deadline constraints, gaining a total revenue of 43. While EDF_α ($\alpha = 2$) will schedule p_2, p_4, and p_3 at time slots 1, 2, and 3 respectively, achieving a revenue of 63. Compared with EDF, EDF_α tends to schedule higher-value packets; Compared with GREED, EDF_α tends to schedule more packets. Therefore, by taking both packet values and packet deadlines into account, EDF_α would have better performance in the scenarios when energy is limited and packets are with deadline constraints.

Algorithm 3 EDF_α

if energy is available **then**
  if there is only one pending packet **then**
    transmit p_f;
    {Let p_f denote the earliest-deadline pending packet while p_l denote the largest-value pending packet.}
  else if $v_{p_f} \geq v_{p_l}/\alpha$ **then**
    transmit p_f;
  else
    transmit a packet p satisfying $v_p \geq \max\{\alpha \cdot v_{p_f}, v_{p_l}/\alpha\}$, with ties broken in favor of earliest-deadline packet.
  end if
end if

Corollary 1. *Both online algorithms EDF_α and EDF are optimal; they reach the optimal competitive ratio $v_{\max}/v_{\min}$. (The greedy algorithm GREED is EDF_1.)*

As shown in [15], we have the following result.

Corollary 2. *In the non-idling setting (such that an algorithm has to send a pending packet, if any, when there exists sufficient energy), GREED is 2-competitive.*

As what we can see from Theorem 3, theoretically speaking, deterministic online algorithms are pessimistic when they are used to schedule packets under scarce harvesting energy scenarios. The main difficulty in designing a deterministic online algorithm is: When we schedule one packet at a time, we do not know whether future harvesting energy can be sufficient to schedule other pending packets or not. We raise the idea that randomness may help to solve this dilemma when we consider oblivious adversaries: We select one packet to send with one unit of energy consumed, under some probability distribution which is internal to the randomized algorithm. This probability distribution is not a stochastic feature owned by the complete input instance, instead, it is from the currently pending packets at time t. In the following, we investigate the power of randomness by designing randomized online algorithms against oblivious adversaries.

Let us use a simple case to illustrate our idea. Consider a wireless sensor's buffer state at a time t and denote the set of pending packets as $S(t)$. Based on Assumption 1 and the algorithm to be introduced (that is, during each time slot, the earliest-deadline packet is either sent or discarded by the randomized algorithm), all the packets in $S(t)$ can be transmitted by their deadlines given sufficient harvesting and/or battery energy. Let the current energy available at time t be $e(t)$ ($e(t) \leq C$). Then we understand any online algorithm can send at most $e(t)$ packets if no future harvesting energy is replenished. Assume $S(t)$ has only two packets for now and $e(t) = 1$. Consider the case $S(t) = \{p, q\}$ and $d_p \leq d_q$ and $v_p < v_q$. If $|S(t)| \leq e(t)$, it is natural to send the earliest-deadline packet p in $S(t)$, as we bear an intuition that later harvesting energy can fuel the battery and be used to transmit later arriving packets. If $|S(t)| > e(t)$, we have to decide whether to send q with a more gain in this step or to leave q in the buffer and face the potential lost of value v_q given the unpredictable later harvesting energy. Randomness works here to solve this dilemma — We deliver p or q based on a probability derived from their values (see below). Now, we extend this idea to multiple packets in $S(t)$. If $S(t)$ contains more than 2 packets, we let the earlier-deadline packet be p and the largest-value packet be q, if any. Among these two packets, if $v_p \geq v_q$, we immediately send p. Otherwise, we randomly select p to send with probability x and select q to send with probability $1 - x$. This randomness provides us an expected competitive ratio r of weighted throughput, which is based on x,

$$r = \min\left\{\frac{v_p \cdot x + v_q \cdot (1 - x)}{v_q}, \frac{(v_p + v_q) \cdot x + v_q \cdot (1 - x)}{v_p + v_q}\right\}. \tag{2}$$

The first term of the right hand side of Equation (2) is the ratio of expected gain when there is insufficient harvesting energy for sending q while the second term is the ratio when the future harvesting energy can cover sending both packets. r is maximized when

$$x = \frac{v_p \cdot v_q}{v_q^2 + v_p \cdot v_q - v_p^2}$$

and thus, given $v_p < v_q$, we have

$$\begin{aligned} r &= \frac{v_p \cdot x + v_q \cdot (1-x)}{v_q} \\ &= 1 - \frac{v_q - v_p}{v_q} \cdot x \\ &= 1 - \frac{v_q - v_p}{v_q} \frac{v_p \cdot v_q}{v_q^2 + v_p \cdot v_q - v_p^2} \\ &= \frac{v_q^2}{v_q^2 + v_p \cdot v_q - v_p^2} \\ &\geq \frac{1}{1.25} \end{aligned} \quad (3)$$

Based on the above discussion, we extend the idea illustrated in the above simple case and derive a randomized algorithm called RAND. The algorithm is a non-idling one and is described in Algorithm 4.

Algorithm 4 RAND in the non-idling setting

1: Let f and l denote the earliest-deadline pending packet and largest-value pending packet respectively, if any.
2: **if** $v_f \geq v_l$ **then**
3: send f;
4: **else**
5: send f with probability $\frac{v_f \cdot v_l}{v_l^2 + v_f \cdot v_l - v_f^2}$;
6: send l and discard f with the probability $\frac{v_l^2 - v_f^2}{v_l^2 + v_f \cdot v_l - v_f^2}$.
7: **end if**

Theorem 4. *In the non-idling setting, the randomized online algorithm RAND is* 1.25*-competitive against an oblivious adversary. RAND is better than any deterministic online algorithms in the same setting.*

Proof. Consider the second part of Theorem 4. To show that RAND is better than any deterministic online algorithms, we only need to point out that the lower bound of competitive ratio for deterministic online algorithms in the same setting is at least $(1 + \sqrt{5})/2 \approx 1.618$ [16, 17].

We use a potential function method to prove the first part of Theorem 4. Let ADV denote an oblivious adversary. We will show

$$\beta \cdot E[w(t)] + \Delta\Phi_t \geq u(t) \quad (4)$$

where $E[w(t)]$ is the expected gain for RAND in time slot t, $u(t)$ is the gain for ADV in time slot t, and Φ is the difference between RAND and ADV's potentials. (We will define Φ in the following.) In order to prove Theorem 4, we need to show that there exists a potential Φ and $\beta = 1.25$ such that $\Phi_0 = 0$ and Inequality (4) holds for any time t.

Consider ADV and RAND's buffers at a time. Let $e(t)$ and $e^*(t)$ denote RAND and ADV's energy balances respectively. We only keep the packets that are scheduled by ADV in ADV's buffer at all the times. Based on Assumption 1 and the algorithm in Algorithm 4 (particularly, during each time slot, the earliest-deadline packet f is either sent or discarded), all the pending packets in both buffers can be transmitted by their deadlines given sufficient harvesting and/or battery energy for RAND. Recall that we consider non-idling setting. Thus, in each time slot, ADV sends one packet when its buffer contains pending packets. We conclude that at any time, the battery balance in RAND is no less than ADV's. So, we have $e^*(t) \leq e(t)$, $\forall t$.

Now, we define

$$\Phi := OPT\left(B^{RAND} \setminus B^{ADV}, e(t) - e^*(t)\right) - \sum_{j \in B^{ADV} \setminus B^{RAND}} v_j$$

where B^{RAND} and B^{ADV} denote RAND and ADV's current buffers at time t, $OPT\left(B^{RAND} \setminus B^{ADV}, e^*(t)\right)$ denotes the maximum total value that an algorithm can get out the packets in RAND's current buffer but not in ADV's current buffer subject to the difference between RAND and ADV's current energy constraints $e(t) - e^*(t)$.

We consider two events: *packet arrivals* and *packet deliveries*, in proving Inequality (4).

Packet arrivals. For any packet that is arriving, if ADV accepts it, then RAND accepts it as well since due to the schedulability of the original input instance, RAND is able to send successfully all pending packets and the future arrivals as long as energy permits. So, for packet arrivals, we have

$$\begin{aligned} w(t) &= 0. \\ u(t) &= 0. \\ \Delta\Phi &\geq 0. \end{aligned}$$

So, Inequality (4) holds.

Packet deliveries. We case study packet deliveries.

- Assume ADV and RAND send the same packet, say f.

  Then we charge v_f to both algorithms and the gain ratio of $E[w(t)]/u(t) = w(t)/u(t) = v_f/v_f = 1$. $\Delta\Phi = 0$.

  $$\beta \cdot E[w(t)] + \Delta\Phi = \beta \cdot v_f + 0 \geq v_f.$$

  Inequality (4) holds.

- Assume ADV sends a packet $p \notin B^{RAND}$.

  $$\begin{aligned} \Delta\Phi &= v_p \\ \beta \cdot E[w(t)] + \Delta\Phi &\geq \beta \cdot E[w(t)] + v_p \\ &\geq v_p. \end{aligned}$$

  Inequality (4) holds.

- Assume ADV sends a packet $p \in B^{RAND}$ and $p \neq f$.

  $\Delta\Phi = 0$.

  In the following, we need to show $\beta \cdot E[w(t)] = 1.25E[w(t)] \geq u(t) = v_p$.

  Recall that the original input instance is schedulable. Then, if RAND does not send $p \neq f$ in time slot t, then p is still scheduable along with any future released packets. Thus, for ADV, if it sends $p = l$, then the competitive ratio is ≤ 1.25, as shown below.

  $$E[w(t)] = \frac{v_f \cdot \frac{v_f \cdot v_l}{v_l^2 + v_f \cdot v_l - v_f^2} + v_l \cdot \left(\frac{v_l^2 - v_f^2}{v_l^2 + v_f \cdot v_l - v_f^2}\right)}{v_l} \geq 1.25 v_l.$$

  Now, we examine the case when ADV sends $p \neq f, l$. In this case, as ADV is able to schedule p again along with other released jobs if we drop f from ADV's buffer, then we force ADV to send l instead of p in this single time slot. This modification does not hurt ADV's overall gain, but its schedule sequence is changed. After this modification, we have

  $$E[w(t)] = \frac{v_f \cdot \frac{v_f \cdot v_l}{v_l^2 + v_f \cdot v_l - v_f^2} + v_l \cdot \left(\frac{v_l^2 - v_f^2}{v_l^2 + v_f \cdot v_l - v_f^2}\right)}{v_l} \geq 1.25 v_l.$$

- Assume ADV sends a packet $p \in B^{RAND}$ and $p = f$.

  The expected RAND's gain is

$$E[w(t)] = v_f \cdot \frac{v_f \cdot v_l}{v_l^2 + v_f \cdot v_l - v_f^2} + v_l \cdot \frac{v_l^2 - \cdot v_f^2}{v_l^2 + v_f \cdot v_l - v_f^2}$$

  Note that ADV must send l since otherwise, to maximize its total gain, ADV should have selected l instead of f to send in this time slot. We consider the total energy consumption and the total packet value as continuously cumulated values instead of discrete ones for ADV. We have

$$\begin{aligned} \beta \cdot E[w(t)] + \Delta\Phi &= \beta \cdot E[w(t)] + \Delta E[\Phi] \\ &= E[\beta \cdot w(t) + \Delta\Phi] \\ &= 1.25 \cdot E\left[v_f \cdot \frac{v_f \cdot v_l}{v_l^2 + v_f \cdot v_l - v_f^2} + v_l \cdot \frac{v_l^2 - \cdot v_f^2}{v_l^2 + v_f \cdot v_l - v_f^2} + \Delta\Phi\right] \\ &= 1.25 \cdot E\left[\frac{v_f \cdot v_l}{v_l^2 + v_f \cdot v_l - v_f^2} \cdot \left(v_f + 0\right) + \frac{v_l^2 - \cdot v_f^2}{v_l^2 + v_f \cdot v_l - v_f^2} \cdot \left(v_f + v_l\right)\right] \\ &\geq u(t) \end{aligned}$$

  The second equation is based on the linearity of expectation.

According to the calculation that we have given in Inequality 3, we prove Theorem 4. □

Recall that if in an input instance, the packets are allowed to overflow but there is no energy constraints, then our model becomes the well-studied *bounded-delay model* [15]. In [17], a $\left(\frac{e}{e-1} \approx 1.582\right)$-competitive randomized algorithm has been given and the lower bound was proved 1.25. In our setting, energy constraints (but not packets' deadlines) are the bottle-neck of transmitting all the weighted packets released in the input instance. Therefore, we are able to get a randomized algorithm RAND which matches the lower bound of competitive ratio given for the similar but different bounded-delay model.

5. Simulations

In this section, we empirically study the proposed offline algorithm OPT and the online algorithms GREED, EDF$_\alpha$ and RAND. For purpose of comparison, we also implement classic algorithms EDF and ALAP, as well as the stochastic algorithm in [8]. To fully evaluate their performance, we simulate various settings with different job arrival patterns and job value distributions. A real solar trace is used as the harvesting energy input in the simulation. The simulation results show that even the pessimistic-in-theory deterministic online algorithms EDF$_\alpha$ and GREED outperform EDF and ALAP in practice. In addition, we find the RAND algorithm has its performance close to the better one of GREED and EDF$_\alpha$ in almost all cases and thus can better guarantee the worst-case performance among all. The simulation results also demonstrate that the proposed online algorithms have neglected time overhead and thus, they are suitable for real time applications.

5.1. Evaluation of the optimal offline algorithm

First, we evaluate the optimal offline algorithm OPT and compare it with the threshold-based online algorithm proposed in [8]. As the threshold-based algorithm approach only works under the setting where packet value distributions are known exactly (particularly, packets and energy arrivals satisfying Poisson processes), we use the same experiment setting used in [8] and see how OPT outperforms the threshold-based online algorithm under this setting.

In the setting, packets satisfy a Poisson arriving process with arrival rate $\lambda = 100$, harvesting energy also arrives at a Poisson process with arrival rate $\gamma = 1$. Note that in the setting [8], the harvesting energy can only be either 0 or 1 unit. We simulate three packet value distributions (uniform, Poisson and binary) and use the maximum *reward* *rate* (normalized weighted throughput) to measure the performance, defined as the total value of packets sent over the overall value of all ever released packets. The experiment is repeated 100 times and the result is plotted in Figure 1. The dotted line represents the average performance of our optimal offline algorithm OPT. The performance of OPT matches with the theoretical average reward rate in [8] which is represented with solid line in Figure 1. This result confirms OPT's optimality.

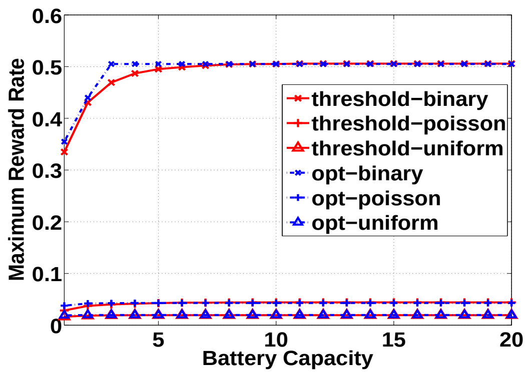


Figure 1: Maximum reward rates for the threshold-based algorithm and an optimal offline algorithm OPT

5.2. *Evaluation of the competitive online algorithms*

Next, we evaluate the proposed online algorithms GREED, EDF$_\alpha$ and RAND using the performance of OPT as the benchmark. Note that in our problem model, packets are associated with deadlines and there are no any stochastic assumptions made over packets or energy. Hence the solutions proposed in [8, 3] do not work for our model. The experiment settings are described as below.

Harvesting energy. We use a real solar energy trace as the harvesting energy input in the simulation. The real solar energy trace is collected by Computer Science Whether Station at University of Massachusetts, Amherst [18]. We scale down the solar power trace to make it compatible with our simulated time scale.

To fully evaluate the algorithms under different harvesting energy productions, we choose three types of days with "high", "medium", and "low" solar energy productions. Note for each type of days, we select a 5-day-time period. Simulations are conducted under these three harvesting energy settings. We only present the results when energy is "high" since we got similar results under these 3 harvesting energy conditions. A solar trace of "high" days is shown in Figure 2.

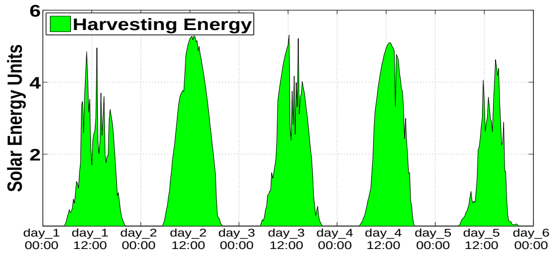


Figure 2: Solar energy trace for arbitrary 5 days.

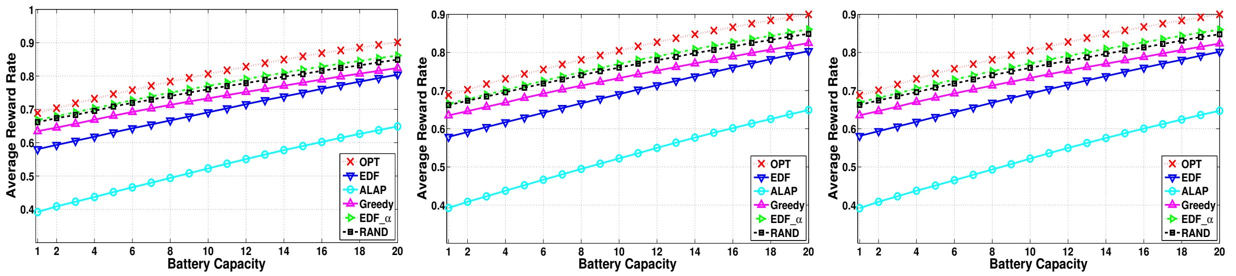
Packet setting. As the performance of online algorithms can be sensitive to the type of packet sequences (as what we shall see), we simulate multiple types of packet settings in order to thoroughly evaluate the performance of online algorithms. In particular, we simulate three packet arrival patterns (1) Poisson arrivals [8, 3], (2) uniform arrivals [19] and (3) power-law distribution arrivals [20].

We simulate a 5-day-time period, which has 480 time slots. For all these three types of packets' arrivals, the range of the packets' release times and deadlines are $[0, 480]$. For uniform arrival packets, the deadline of each packet is a random number chosen between its release time and 480. For Poisson arrivals and power-law arrivals, we set the *slack time* (the time span between a packet's arriving time and its deadline) of each packet a fix number 48. The generated packet number for uniform arrival packet is set as 400. While for Poisson packet arrivals, the generated packet number is determined by the arrival rate, which is set as 1. For power-law distributed packet arrivals, the packet arrival times satisfy $y = (x/80)^{-0.2}$ power-law distributions.

We also simulate three types of distributions for packet values: uniform distributions [8], Poisson distributions [8], and exponential distributions [8]. For uniform distributions, the weight of each packet is a float number chosen uniformly randomly from $[0, 100]$. For Poisson distributions, the mean value is 50. For exponential distributions, the rate parameter is set as 1.

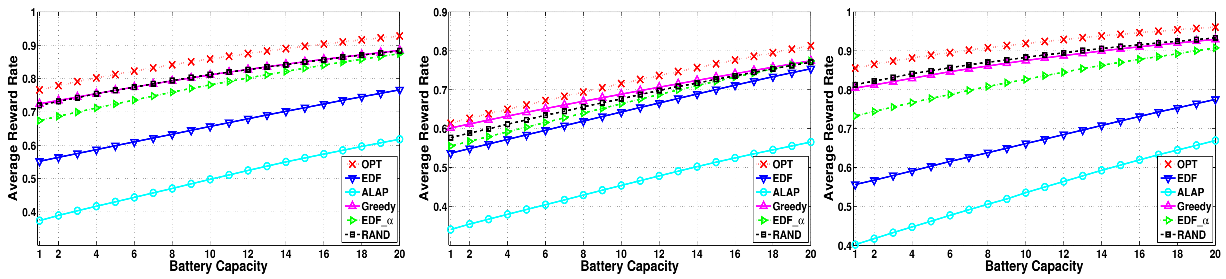
Experiment methodology. As the model we study is a more general one that packets are associated with deadlines and no assumptions made over the input, the threshold approach proposed in [8, 3] cannot be applied to our model. Therefore, to evaluate our proposed deterministic online algorithms, we compare them with the throughput's upper bound generated by the optimal offline algorithm OPT. We also compare them with two conventional scheduling algorithms EDF and ALAP [4]. EDF always transmits the packet which has the earliest deadline when energy is available. ALAP is conservative in using energy so that it tries to transmit each packet at its deadline unless the battery energy is going to overflow. OPT is the optimal offline algorithm proposed in Section 3, providing the upper bound of weighted throughput.

To evaluate the heuristic online algorithms, we conduct simulations under various energy and packet settings. We repeat each experiment 100 times and calculate the average reward rate for each online algorithm. The average reward rate is defined as the total value of scheduled packets over the overall value of all packets ever released. Note that in the simulation, for EDF_α, we set $\alpha = 2$.



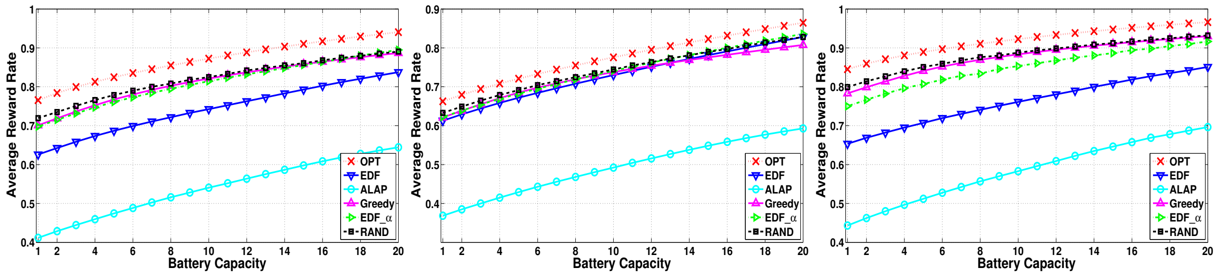
(a) (b) (c)

Figure 3: Average reward rate under various battery capacities. Packets have uniform arrival patterns. (a) packets' values are uniformly distributed. (b) packets' values are Poisson distributed. (c) packets' values are exponential distributed.



(a) (b) (c)

Figure 4: Average reward rate under various battery capacities. Packets have Poisson arrival patterns. (a) packets' values are uniform distributed. (b) packets' values are Poisson distributed. (c) packets' values are exponential distributed.



(a) (b) (c)

Figure 5: Average reward rate under various battery capacities. Packets have power-law arrival patterns. (a) packets' values are uniform distributed. (b) packets' values are Poisson distributed. (c) packets' values are exponential distributed.

Simulation results. The simulation results under various packet settings are shown in Figure 3, Figure 4 and Figure 5. Figure 3(a) shows the average reward rate in the setting when packets have uniform arrival rates and packets' values are uniformly distributed. In the figure, y-axis represents the average reward rate of scheduling algorithms under various battery capacities. We can see that GREED, EDF$_\alpha$ and RAND gain 5% more reward rate than EDF, 30% more reward rate than ALAP, and 3% less than OPT. When packets have uniform arrival patterns and packets' values satisfy either Poisson distributions or exponential distributions, we get similar result, as that shown in Figure 3(b) and Figure 3(c). In all sub-figures in Figure 3, we can see that EDF$_\alpha$ and RAND perform better than GREED, and RAND almost perform as good as EDF$_\alpha$.

However, for the case when packets have Poisson arrival patterns as that in Figure 4, GREED tends to achieve about 5% higher reward rate than EDF$_\alpha$ especially when battery capacity is low. In this case, RAND behaves almost as good as or better than GREED. The underlying reason that GREED has better performance than EDF$_\alpha$ when packets have Poisson arrivals is because by always scheduling the largest-value packets, GREED can avoid the situation that high-value packets fail to be scheduled due to deadline constraints or energy constraints. And for Poisson arrivals and power-law arrivals, the situation of deadline conflicts or energy conflicts is more likely to happen. The results in which packets have power-law arrivals are shown in Figure 5. Under this setting, we observe that GREED, EDF and EDF$_\alpha$ almost have the same performance when packets' values have uniform or Poison distributions. When packets' value satisfy exponential distributions, RAND and GREED have better performance than EDF$_\alpha$.

In summary, for all the energy and packet settings in the simulation, we get similar conclusions that GREED, EDF$_\alpha$ and RAND have better performance than EDF and ALAP, and their performance are close to optimal. We find that EDF$_\alpha$ may have worse performance than GREED in some cases as it tries to balance packets' values and packets' deadlines. Although GREED and EDF$_\alpha$ may outperform each other under different settings, RAND can always perform as good as the better one of them. This indicates RAND can better guarantee the worst-case performance.

Running-time overhead. As the sensor node is often of limited computing resource, an algorithm's running time overhead is very important. Therefore we further conduct simulations to measure the running time of each algorithms. The simulation is implemented using C# and it runs at a Microsoft Windows platform. We run 100 repetitions under various packets settings with battery capacities varying from 1 to 20, packet arrival rates varying from 0.1 to 1. The results got are similar to each other under various setting, therefore we only show one group of results under one setting in Table 2 as a representative. Under this setting, packets have Poisson arrival rates and packets' values have uniform distributions, battery capacity is 10 and packet arrival rates changed from 0.1 to 1. Note that the running-time overhead of RAND and EDF$_\alpha$ can be further reduced by using priority queue.

From Table 2, we can see that the classic algorithm EDF and ALAP have lower time overhead compared with our proposed online algorithms. EDF$_\alpha$, RAND and OPT have similar time overhead. Precisely speaking, GREED has double running time of ALAP or EDF, and RAND has double running time of GREED. EDF$_\alpha$ and OPT have double running time than RAND. The overhead of GREED is relatively lower compared with EDF$_\alpha$ and RAND, which indicates GREED is an efficient and effective algorithm for our problem model. Overall, the time overhead of all these online algorithms is negligible.

arrival rates	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
EDF	0.68	1.82	2.44	3.54	4.64	5.51	6.90	8.28	9.44	11.36
ALAP	0.92	2.03	3.32	4.66	6.14	7.43	8.52	9.95	11.07	12.86
GREED	1.84	4.18	5.92	8.99	11.87	14.25	17.52	20.52	23.41	27.01
RAND	3.74	9.38	13.91	20.88	28.50	35.57	43.58	52.75	59.87	68.27
EDF_α	4.04	9.91	15.63	23.82	33.52	41.92	51.73	62.81	73.262	83.94
OPT	3.30	11.36	19.89	33.36	48.05	61.92	76.45	90.56	105.80	120.35

Table 2: Running-time overhead (in *ms*) under the settings when packets have Poisson arrival rates and packets' values satisfy uniform distributions. Battery capacity is set as 10. Packet arrival rate is changed from 0.1 to 1.

6. Related Literature

There has been much research work in developing algorithms for energy harvesting sensor networks and various models have been studied. In [4], a lazy scheduling algorithm was proposed to schedule packets with deadlines, assuming the whole released set of packets is schedulable in real-time systems. [21] provided an offline algorithm to maximize the sum of throughput with a given common deadline, and provided an online policy which has empirical near-optimal performance. The work [22] maximized the minimum energy reserved in the sensor nodes in a given transmission path. In [5, 6, 7], the authors assumed different service levels had different power consumptions and they proposed an algorithm to allocate the service level subject to energy constraints in order to maximize the overall reward. The above research makes various assumptions on the distributions of the harvesting energy and/or the distributions of the packets released. Along this line of research, [8] derived a threshold-based optimal transmission policy to maximize the average reward rate with the assumption that packet value distributions are known, the patterns of packet arrivals and harvesting energy arrivals are known as well. In [3, 9], the authors developed and evaluated a heuristic online algorithm which adapts the average transmit probability to the harvesting energy availability with the assumption that the packets value distribution is known. In their work, harvesting energy has only two states (0 or 1) and energy arrivals satisfy independent and identically distributed Bernoulli process. All the scheduling policies in [8, 3, 9] are binary-based transmission policies such that the packets are transmitted or discarded immediately at the time when they arrive at the sensor node. Also, all these proposed scheduling policies are based on the assumptions of knowing the distributions of packet values, packet arrival patterns, and energy arrival patterns. The worst-case performance guarantees of these algorithms have not been discussed yet and the claimed expected performance heavily depends on the assumptions made. Different from the above research, we study algorithms' performance against the worst-case scenarios in this paper.

7. Conclusions

In this paper, we study the problem of scheduling weighted packets in an energy-harvesting sensor node. We design an optimal offline algorithm to maximize the weighted throughput. We also provide a lower bound of competitive ratio for deterministic online algorithms. We propose two optimal deterministic online algorithms EDF_α and GREED. Randomized algorithm RAND is developed as well with a better competitive ratio against oblivious adversary in the non-idling setting. We experimentally compare the online algorithms with conventional scheduling algorithms EDF and ALAP, using real solar traces and various simulated packet sequences. The empirical results show that EDF_α, GREED and RAND have far better performance than conventional algorithms, and RAND is the best among them.

In our future work, we will study semi-online algorithms, assuming that limited information (such as a loading factor) on the input instance is known while a particular exact probabilistic distribution is still unavailable.

Acknowledgement

This material is based upon work supported by NSF under Grants No. CCF-0915681 and CCF-1216993. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

References

[1] X. Jiang, J. Polastre, D. Culler, Perpetual environmentally powered sensor networks, in: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN), 2005, p. Article No. 65.

[2] A. Kansal, J. Hsu, S. Zahedi, M. B. Srivastava, Power management in energy harvesting sensor networks, ACM Transactions on Embedded Computing Systems (TECS) 6 (4) (2007) Article No. 32.

[3] N. Michelusi, K. Stamatiou, M. Zorzi, On optimal transmission policies for energy harvesting devices, in: Proceedings of Information Theory and Applications Workshop (ITA), 2012, pp. 249–254.

[4] C. Moser, D. Brunelli, L. Thiele, L. Benini, Real-time scheduling for energy harvesting sensor nodes, Real-Time Systems 37 (3) (2007) 233–260.

[5] C. Moser, J.-J. Chen, L. Thiele, Reward maximization for embedded systems with renewable energies, in: Proceedings of the 14th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), 2008, pp. 247–256.

[6] C. Moser, J.-J. Chen, L. Thiele, Optimal service level allocation in environmentally powered embedded systems, in: Proceedings of the 24th Annual ACM Symposium on Applied Computing (SAC), 2009, pp. 1650–1657.

[7] C. Moser, J.-J. Chen, L. Thiele, Power management in energy harvesting embedded systems with discrete service levels, in: Proceedings of the 14th ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2009, pp. 413–418.

[8] J. Lei, R. Yates, L. Greenstein, A generic model for optimizing single-hop transmission policy of replenishable sensors, IEEE Transactions on Wireless Communications 8 (2) (2009) 547–551.

[9] N. Michelusi, K. Stamatiou, M. Zorzi, Performance analysis of energy harvesting sensors with time-correlated energy supply, in: Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2012, pp. 839–846.

[10] V. Devadas, F. Li, H. Aydin, Competitive analysis of online real-time scheduling algorithms under hard energy constraint, Real-Time Systems 46 (1) (2010) 88–120.

[11] R. Srivastava, C. E. Koksal, Energy optimal transmission scheduling in wireless sensor networks, IEEE Transactions on Wireless Communications 9 (5) (2010) 1550–1560.

[12] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, 1979.

[13] A. Borodin, R. El-Yaniv, Online Computation and Competitive Analysis, Cambridge University Press, 1998.

[14] M. H. Goldwasser, A survey of buffer management policies for packet switches, ACM SIGACT News 41 (1) (2010) 100–128.

[15] A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, M. Sviridenko, Buffer overflow management in QoS switches, SIAM Journal on Computing (SICOMP) 33 (3) (2004) 563–583.

[16] B. Hajek, On the competitiveness of online scheduling of unit-length packets with hard deadlines in slotted time, in: Proceedings of the 35th Annual Conference on Information Sciences and Systems (CISS), 2001, pp. 434–438.

[17] F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, J. Sgall, T. Tichý, Online competitive algorithms for maximizing weighted throughput of unit jobs, Journal of Discrete Algorithms (JDA) 4 (2) (2006) 255–276.

[18] Umass amherst computer science weather station, `http://traces.cs.umass.edu`.

[19] D. Rajan, A. Sabharwal, B. Aazhang, Delay-bounded packet scheduling of bursty traffic over wireless channels, IEEE Transactions on Information Theory 50 (1) (2006) 125–144.

[20] D. A. Benson, R. Schumer, M. M. Meerschaert, Recurrence of extreme events with power-law interarrival times, Geophysical Research Letters (GRL) 34 (L16404).

[21] K. Tutuncuoglu, A. Yener, Sum-rate optimal power policies for energy harvesting transmitters in an interference channel, coRR, abs/1110.6161 (2011).

[22] B. Zhang, R. Simon, H. Aydin, Harvesting-aware energy management for time-critical wireless sensor networks with joint voltage and modulation scaling, IEEE Transactions on Industrial Informatics 9 (1) (2013) 514–526.