Maximizing Throughput in Energy-Harvesting Sensor Nodes

Stanley P. Y. Fung

Department of Computer Science, University of Leicester, Leicester, United Kingdom.
pyf1@le.ac.uk

Abstract. We consider an online throughput maximization problem in sensor nodes that can harvest energy. The sensor nodes generate and forward packets, which cost energy; they can also harvest energy from the environment, but the amount of energy that can be harvested is not known in advance. We give a number of algorithms and lower bounds for the case of a single node. We consider both the general case and some types of 'non-idling' adversaries where we can get better bounds. We also consider the case of networks with multiple nodes and demonstrate that some very simple scenarios already admit no competitive algorithms.

1 Introduction

Background. Sensor networks are often deployed in areas where it is infeasible to maintain a constant energy supply to the sensor nodes. Often the nodes are equipped with batteries, and a node can only operate until its battery is exhausted. There are many research work on how to extend the useful lifetime of the sensor node or the sensor network by careful scheduling. If the sensor node is equipped with some energy-harvesting device, e.g., solar cells so it can replenish used energy, it can help make the system work longer or even indefinitely. This creates a challenge of designing algorithms that can make use of this harvested energy effectively.

The model. We consider the scenario where each sensor node senses the environment, generates packets and sends them to a target destination. First consider a single node. The model was defined in [12]. Time consists of discrete time steps 1, 2, ... A packet j is specified by a 3-tuple $(r(j), d(j), v(j))$, which represents its release time, deadline and value. A packet with release time $r(j)$ and deadline $d(j)$ can only be sent in one of the time steps between $r(j)$ and $d(j)$, inclusive. Sending a packet costs one unit of energy. The sensor is equipped with a battery with a capacity of C, and an energy-harvesting device that may harvest some amount of energy $h(t)$ at each time step t. Let $e(t)$ denote the energy level of the battery at the beginning of time t (excluding energy harvested at this time step). A packet can only be sent if the node has sufficient energy, i.e. $e(t)+h(t) \geq 1$. The energy remaining at the next step is given by $e(t+1) = \min(C, e(t)+h(t)-x(t))$ where $x(t) = 1$ if a packet is sent at time t and $x(t) = 0$ otherwise. We assume there is no 'leak' of the battery so the energy level stays the same when no packets are sent. The objective is to maximize the *profit* or *weighted throughput* of the schedule, i.e. the sum of values of all packets sent.

Note that $h(t)$ is not known in advance and only become known at time t. Packet arrivals are also unknown in advance: packets with release time $r(j)$ are not known until time $r(j)$. Therefore, this is an *online* problem. We measure the performance of online algorithms using competitive analysis [1]: an online algorithm A is r-competitive if the value produced by A is always at least $1/r$ that of the optimal offline algorithm OPT over all input instances. For randomized online algorithms we use the expected value of A instead for comparison.

Generalizing from a single node, we also consider the model where nodes are connected into a network. Packets may have different sources and destinations, and each sensor node needs to forward traffic generated by other nodes as well. In our model, in each time step each node can send one packet to another node. Each packet takes one time unit to pass the link between two nodes. Thus if a packet is sent at time t in an upstream node, it appears as a packet with release time $t + 1$ in the next downstream node. Sending a packet takes one unit of energy, and we ignore the energy required to listen to or receive packets. The objective is to maximize the total value of packets reaching their destinations.

Before going any further we introduce some definitions and notations. Let $V = \max_j v(j)/\min_j v(j)$. An instance is *underloaded* if all packets can be sent by OPT, respecting deadlines and energy availability. An algorithm or an adversary (the optimal offline algorithm) is *non-idling* if, at every time step, it must send a packet as long as there is energy available and there are packets pending.

Previous work and our contributions. For the case of a single node, the problem without energy considerations is known as the *unit job scheduling* problem (UJS) and was studied extensively; see [5] for a survey. The current best deterministic upper and lower bounds are 1.828 [4] and 1.618 [3, 6, 14] respectively while for randomized algorithms they are 1.58 [2] and 1.25 [3].

There has been a lot of work in the sensor network community on the problem of energy harvesting although most of them study the problem with somewhat different objective functions, or assume that there are knowledge of probability distributions or even complete knowledge of packet arrivals and/or energy harvesting. For example, [11] assumed that future energy harvesting is known; [8] assumed both the packet arrivals and energy replenishment follow a Poisson process. The only algorithmic, worst-case analysis without prior probability assumptions that we are aware of is [12]. It considers the case of a single node, and the authors gave deterministic upper and lower bounds of V against general adversaries. Then they turned their attention to non-idling adversaries and claimed to give a randomized algorithm that is 1.25-competitive against such adversaries. We show that this is not true even when energy is not a limiting factor. (Note: in subsequent communications [9] one of the authors stated that their 'non-idling' adversary is more restricted than just not being allowed to idle; it is not allowed to have any kind of 'reserving' of energy by scheduling fewer packets. It was not made precise what it means, but it seems to share similar spirit of the strongly non-idling adversary that we define later. In any case, we show that their upper bound is not correct even when there is unlimited energy, and in such scenarios any definition of non-idling is irrelevant since there is always no harm in moving packets earlier to those idle time steps. The authors have also since published a corrigendum [13] which gave a 2.5-competitiveness proof.) In fact we prove a general lower bound of 2 for all randomized algorithms, and a lower bound of $\Omega(\sqrt{V})$ for deterministic algorithms, against oblivious, non-idling adversaries.

As can be seen, a non-idling adversary is still very powerful. Thus we define a more restricted *strongly non-idling* adversary, and against such adversaries we prove a deterministic upper bound of 2 [1] and a matching randomized lower bound.

Back to the general adversary case, we show that the correct deterministic competitive ratio should in fact be $V+1$. We also consider the unweighted packet case and show that if packets have agreeable deadlines, i.e., packets released earlier have earlier deadlines, then the Earliest Deadline First algorithm (EDF) is 1-competitive.

Finally we consider the case of a network of nodes. When energy is not a restriction, the problem becomes the one considered by [10]. They considered the case of an *uplink tree*, where the nodes are connected into a tree and the root node is the sink, and packets can originate in any node but the destination is always the sink. This is a common scenario in sensor network applications. They showed that it is possible to achieve 1-competitiveness for unweighted, underloaded instances. For general network topologies and general source/destination pairs they gave a tight $O(P \log P)$ competitive ratio bound, where P is the maximum route length. In the case with energy we demonstrate that the problem has poor cometitive ratios even for some very simple scenarios.

Due to space constraints some proofs will only appear in the full paper.

2 Non-idling adversary

Proposition 1. *The competitive ratio of RAND [12] is at least* 1.265 *against an oblivious, non-idling adversary, even when there are no energy limitations.*

In fact we show the following lower bounds for all non-idling randomized algorithms:

Theorem 1. *No non-idling randomized algorithm is better than* $(2 - \frac{2}{\sqrt{V}+1})$*-competitive against an oblivious, non-idling adversary. For deterministic algorithms the lower bound is* $\Omega(\sqrt{V})$*.*

[1] In [12] it was stated that the greedy algorithm is 2-competitive against non-idling adversaries, apparently as a corollary from [7] which is about UJS. However our problem is not a special case of UJS, even for strongly non-idling adversaries. We give a separate 2-competitive proof, both because of this and because of the difference in the (strongly) non-idling definitions.

Proof. Consider a setting with two packets $j_1(1, 1, \sqrt{V})$ and $j_2(1, 2, 1)$, a battery with $C = 2$ and an initial $e(1) = 2$, and no energy harvested throughout. Suppose an online randomized algorithm A sends j_1 at time 1 with probability p and j_2 with probability $1 - p$. (These are the only two possibilities as it is non-idling.) If $p \leq \frac{1+V}{2V}$, no further packets are released. A can send j_2 at time 2 if it has not already done so at time 1, so the expected profit of A, $E[A] = p(\sqrt{V} + 1) + (1 - p)(1) = 1 + p\sqrt{V}$. The optimal profit is clearly $1 + \sqrt{V}$, so the competitive ratio is at least

$$\frac{1+\sqrt{V}}{1+p\sqrt{V}} \geq \frac{1+\sqrt{V}}{1+\frac{1+V}{2V}\sqrt{V}} = \frac{2V+2V\sqrt{V}}{2V+V\sqrt{V}+\sqrt{V}} = \frac{2V+2\sqrt{V}}{V+2\sqrt{V}+1} = 2 - \frac{2}{\sqrt{V}+1}$$

Otherwise if $p > \frac{1+V}{2V}$ then $j_3(3, 3, V)$ arrives. If A sent j_1 at time 1 then it must send j_2 at time 2 since it is non-idling, leaving no energy for j_3, whereas if it sent j_2 at time 1 then there is no pending packet to send at time 2 and so has the remaining energy to send j_3. Hence $E[A] = p(\sqrt{V} + 1) + (1 - p)(1 + V) = 1 + V - p(V - \sqrt{V})$. OPT will send j_2 at time 1 and j_3 at time 3. Note that this OPT is non-idling. The competitive ratio is therefore

$$\frac{1+V}{1+V-p(V-\sqrt{V})} > \frac{1+V}{(1+V)+\frac{1+V}{2V}(\sqrt{V}-V)} = \frac{2V}{2V+(\sqrt{V}-V)} = 2 - \frac{2}{\sqrt{V}+1}$$

For deterministic algorithms, the proof is basically the same but p can only take on two discrete values $\{0, 1\}$. If an online algorithm A sends j_2 at $t = 1$ (i.e. $p = 0$) then no more packets arrive and the competitive ratio is $1 + \sqrt{V}$. Otherwise if j_1 is sent ($p = 1$) then j_3 arrives and the competitive ratio is $\frac{1+V}{1+V-(V-\sqrt{V})} = \Theta(\sqrt{V})$. □

3 Strongly non-idling adversary

The instances in Theorem 1 illustrate a curious aspect of the problem. When faced with two packets p and q with $v(p) > v(q)$ and $d(p) < d(q)$, it seems natural to give preference to p over q. Such algorithms are called *rational*. Here however, the algorithm has to send q and discard p, even when $v(p)$ is much higher than $v(q)$, in order to get good performance by saving the energy for a later packet.

To get around this, we put further restrictions on what the adversary can do. We say a packet p *dominates* another packet q if (i) $v(p) > v(q)$ and $d(p) \leq d(q)$, or (ii) $v(p) \geq v(q)$ and $d(p) < d(q)$. We call a schedule S *irrational* if there are two packets $p \notin S$ and $q \in S$, q is sent in a time step t such that $r(p) \leq t \leq d(p)$, and yet p dominates q. We call an adversary *strongly non-idling* if it is non-idling and it never returns an irrational schedule. Note that when there is no energy limitation or when non-idling is not required, this additional assumption is redundant: clearly substituting q with p gives a schedule at least as good. However, what may happen is that sending p first may mean the adversary is forced to send q later due to its non-idling property, consuming the energy that could be used for sending future high-value packets, whereas sending q first may 'kill off' p and thus save the energy. The situation in the proof of Theorem 1 would not happen in strongly non-idling adversaries: OPT would not be allowed to discard j_1.

For strongly non-idling adversaries we first show a simple deterministic lower bound of 2, then show that the greedy algorithm is 2-competitive and thus optimal.

Theorem 2. *Any deterministic algorithm is at least* 2*-competitive against a strongly non-idling adversary.*

Proof. Consider a setting with two packets $j_1(1, 1, 1)$ and $j_2(1, 2, 1 + \epsilon)$, where $\epsilon > 0$ is very small, a battery with $C = 2$ and an initial $e(1) = 2$, and no energy harvested throughout. Clearly OPT can send both packets, hence if an online algorithm A does not send both packets then no more packets arrive, giving a competitive ratio of at least $(2 + \epsilon)/(1 + \epsilon) \approx 2$. Otherwise A sends j_1 at $t = 1$ and j_2 at $t = 2$, consuming all energy. Then $j_3(3, 3, V)$ arrives, which A has no energy to send. OPT sends j_2 at $t = 1$, dropping j_1 which a strongly non-idling adversary can do (it would not be allowed to do so if $v(j_1) \geq v(j_2)$), and then send j_3. The competitive ratio is $\frac{1+\epsilon+V}{2+\epsilon} > 2$ for large V. $\square$

We first define a total ordering of packets as follows. For two packets x and y, we say $x \succ y$ if (i) $v(x) > v(y)$, or (ii) $v(x) = v(y)$ and $d(x) < d(y)$, or (iii) $v(x) = v(y)$ and $d(x) = d(y)$ and $ID(x) < ID(y)$, where $ID()$ is a unique ID given to each packet for tie-breaking purposes. The algorithm $GREEDY$ works as follows: at each time step, as long as there is energy to send a packet and there is at least one pending packet, send the one that is 'largest' according to the $\succ$ ordering, i.e. the packet x such that there is no other packet with $y \succ x$.

We assume OPT and $GREEDY$ tie-break using the IDs consistently: if two packets x and y have the same values and deadlines, and $ID(x) < ID(y)$ (so $GREEDY$ favours x), then OPT would not leave x out of its schedule but include y. In addition, we can assume that if OPT sent two packets x and y, where $x \succ y$, one at time step t_1 and another at t_2, where $t_1 < t_2$, and that both packets are available during $[t_1..t_2]$, then x is sent at t_1 and y at t_2 and not the other way round. This follows from a simple exchange argument; note that this does not affect the energy levels or the non-idling requirement at any other time steps.

Theorem 3. *GREEDY is* 2*-competitive against a strongly non-idling adversary.*

Proof. Let G denote the schedule produced by $GREEDY$. Let $e(t)$ and $e^*(t)$ denote the energy in the battery at time t of G and OPT respectively. We prove by induction the invariant that

(Inv-E): at any time t, $e(t) \geq e^*(t)$

and at the same time describe how the packet values in OPT can be charged to those in G.

Clearly (Inv-E) is true initially. When energy is harvested the battery of G increases at least as much as that of OPT, unless the battery of G is fully charged before that of OPT in which case (Inv-E) holds anyway.

Consider a time t, and assume (Inv-E) is true up to time t. If $GREEDY$ does not send a packet at t, then clearly (Inv-E) is maintained at $t+1$. Moreover, if OPT sends a packet x then by (Inv-E) $GREEDY$ also has the energy to send packets, so the only reason that it is idle is because x has already been sent earlier. Charge x to itself in G.

Now suppose $GREEDY$ sends a packet y and OPT sends a packet x. Clearly (Inv-E) remains true at $t+1$. If $v(x) \leq v(y)$, simply charge the value of x to y. If $v(x) > v(y)$, then x must already be sent by G earlier since otherwise G would have sent it instead at this time step; charge x to itself in that earlier time step.

Finally suppose $GREEDY$ sends a packet y but OPT idles. We will show below that this can only happen if OPT has zero energy ($e^*(t) = 0$ and $h(t) = 0$). This means (Inv-E) is still maintained after this time step. No packet values from OPT need to be charged.

Consider each packet in G, it receives at most two charges, one from a future copy of itself in OPT and another from a packet sent by OPT at the same time step which has at most the same value as the packet in G. Summing over all packets in G, this shows that $GREEDY$ is 2-competitive.

We now return to prove that if $GREEDY$ sends a packet but OPT idles at time t, then OPT must have zero energy. Suppose this is not true. Then OPT must have no pending packets at t since it is non-idling. Let x_1 be the packet sent by G at t. This packet x_1 must have been sent by OPT at an earlier time $t_1 < t$, since otherwise it would be pending for OPT at t. Consider the packet x_2 sent by G at time t_1. This packet must exist, i.e., G cannot idle at t_1, because x_1 is pending, and G must have energy to send it because of (Inv-E) and the fact that OPT has energy to send x_1. Moreover, $x_2 \succ x_1$ because otherwise x_1 would be sent here instead by $GREEDY$. x_2 must be sent by OPT: otherwise if $d(x_2) \geq t$ then it would still be pending at t so OPT could not idle at t, whereas if $d(x_2) < t$ then $d(x_2) < d(x_1)$ and so x_2 dominates x_1, and thus a strongly non-idling adversary could not have discarded x_2 and schedule x_1 at t_1. Let t_2 be the time where OPT sent x_2. It must be that $t_2 < t$, since otherwise OPT would not idle at t. In fact it must be that $t_2 < t_1$: otherwise, if $t_1 < t_2 < t$ then both x_1 and x_2 have been released and have not reached their deadlines during $[t_1..t_2]$, so by our assumption OPT would have sent x_2 first (because $x_2 \succ x_1$).

We then repeat the argument: again G cannot be idle at t_2 and must send a packet x_3, because it has the energy to do so by (Inv-E) and because x_2 is pending at t_2. Moreover this means $x_3 \succ x_2 \succ x_1$. Then, x_3 must appear in OPT: if $d(x_3) \geq t$ then OPT would not idle at t; if $t_1 \leq d(x_3) < t$ then $d(x_3) < d(x_1)$ and x_3 dominates x_1 and thus OPT could not have discarded x_3 when it could schedule it at t_1; if $d(x_3) < t_1$ then $d(x_3) < d(x_2)$ and similarly x_3 dominates x_2. Furthermore it must appear in a time step t_3 where $t_3 < t_2$: it cannot appear after t since OPT could send it at t; if it appeared between t_2 and t_1 then OPT would swap x_2 and x_3; and if it appeared between t_1 and t then OPT would swap x_1 and x_3.

Continuing like this, we can build a 'chain' of x_i's. In general, let t_i be the time where OPT sent x_i, where $t_i < t_{i-1} < ... < t_1 < t$ and $x_i \succ x_{i-1} \succ ... \succ x_1$. G cannot be idle at t_i because x_i, which G sent at t_{i-1}, is pending and it has at least one unit of energy by (Inv-E). Let x_{i+1} be the packet sent by G at t_i. Moreover $x_{i+1} \succ x_i$, since otherwise G would have sent x_i instead of x_{i+1} at t_i. Then x_{i+1} must appear in OPT or else a strongly non-idling adversary must include x_{i+1} and discard one of $x_1, ..., x_i$ instead, depending on its deadline. Moreover it must appear in a time step t_{i+1} where $t_{i+1} < t_i$: it cannot appear after t since OPT could send it at t; if it appeared between t_j and t_{j-1} for some $j > 1$ then OPT would swap x_{i+1} and x_j; and if it appeared between t_1 and t then OPT would swap x_{i+1} and x_1.

This process can go on indefinitely, but there are only a finite number of time steps before t and all these time steps $t_1, t_2, ...$ and packets $x_1, x_2, ...$ are distinct. Hence we will eventually run into a contradiction. □

In fact we give a randomized lower bound of 2, showing that randomization does not help.

Theorem 4. *No randomized algorithm is better than $(2-\epsilon)$-competitive against a strongly non-idling (and oblivious) adversary.*

Proof. In the following we give a construction involving k rounds, and which shows a lower bound of $2-\frac{1}{k+1}$, for any integer k. Since k can be made arbitrarily large this proves the theorem.

Fix the capacity $C = 2$. At time 1 the battery is full. Fix a large x. At each round $i \geq 1$, an *early packet* $j_i(2i-1, 2i-1, x^{i-1})$ and a *late packet* $k_i(2i-1, 2i, x^{i-1}+\delta)$ arrive, where $\delta > 0$ is very small (in the following calculations we ignore δ). Also at round $i \geq 2$ a unit of energy is harvested at the beginning, i.e., $h(2i-1) = 1$.

First consider round 1 and suppose at time 1 an online algorithm A sends j_1 with probability $1-p_1$ and k_1 with probability p_1. If it sent j_1 first then it must send k_1 at time 2, consuming all energy, while if it sent k_1 first then j_1 expires.

If $p_1 < \frac{k+1}{2k+1}$, then a *big packet* $(3, 3, x)$ arrives and no further rounds are released. The expected profit of A is $E[A] = p_1(1+x) + (1-p_1)(2) = p_1 x + 2 - p_1$, while OPT sends k_1 and the big packet. Hence the competitive ratio $R = \frac{1+x}{p_1 x + 2 - p_1} \approx \frac{1}{p_1} > \frac{2k+1}{k+1}$ for large x. On the other hand, if $p_1 \geq \frac{2k}{2k+1}$, then no more packets or rounds arrive. OPT gets 2 while $E[A] = p_1(1) + (1-p_1)(2)$, hence $R = \frac{2}{2-p_1} \geq \frac{2k+1}{k+1}$. Finally, if $\frac{k+1}{2k+1} \leq p_1 < \frac{2k}{k+1}$, we proceed to round 2.

In general, we only proceed to round i if $\frac{k+1}{2k+1} \leq p_1..p_j < \frac{2k+1-j}{2k+1}$ for all previous rounds $1 \leq j \leq i-1$. Suppose we are at the beginning of round i, and two packets and one unit of energy is released. Consider the event (*):

In all previous rounds the late packets were sent immediately on arrival.

Thus none of the early packets were sent and this leaves one unit of energy (plus the one just arrived). This happens with probability $p_1p_2...p_{i-1}$. Let p_i be the conditional probability that A sends k_i at time $2i-1$, conditional on (*) happens. In this case j_i cannot be sent, and there is one unit of energy left afterwards. And with conditional probability $1-p_i$, again conditional on (*), j_i is sent instead, forcing k_i to be sent at the next time step and with no energy left afterwards. Finally, with the rest of probability $1-p_1p_2...p_{i-1}$, in at least one of the previous rounds both the early and the late packets were sent, meaning there is no energy left at the beginning of round i (other than the one just harvested), so only one of j_i or k_i can be sent (and one of them must be sent). We now consider three cases.

Case 1: $p_1p_2...p_i < \frac{k+1}{2k+1}$. A big packet $(2i+1, 2i+1, x^i)$ arrives and no more rounds arrive. Since x^i is much larger than any other packet values, we only consider the value of this big packet in the profits. The only way A can send this big packet is to have (*) and also send the late packet at this round immediately on arrival; thus $E[A] = (p_1..p_i)(x^i)$. Clearly OPT can get x^i. Hence $R = \frac{1}{p_1..p_i} > \frac{2k+1}{k+1}$.

Case 2: $p_1p_2..p_i \geq \frac{2k+1-i}{2k+1}$. No further packet arrives. The two packets in round i, of value x^{i-1}, dominate the profits, hence we only consider them. The only way that A can sent both of these packets is to have (*), then send the early packet j_i first; this happens with probability $p_1..p_{i-1}(1-p_i)$. In all other scenarios, A can send one of the two packets this round. Thus $E[A] = x^{i-1}(1 + p_1..p_{i-1}(1-p_i))$. OPT gets $2x^{i-1}$. Hence

$$R = \frac{2}{1+p_1..p_{i-1}(1-p_i)} = \frac{2}{1+p_1..p_{i-1} - p_1..p_i} \geq \frac{2}{1+\frac{2k+1-i+1}{2k+1} - \frac{2k+1-i}{2k+1}} = \frac{2k+1}{k+1}.$$

Case 3: $\frac{k+1}{2k+1} \leq p_1p_2..p_i < \frac{2k+1-i}{2k+1}$. We proceed to round $i+1$.

Since Case 3 cannot happen when $i = k$, the construction stops latest at round k and in all cases the lower bound is at least $(2k+1)/(k+1)$. □

4 Unrestricted adversary

4.1 Weighted instances

In [12] it was shown that, against general adversaries (i.e., they can idle), any deterministic non-idling rational algorithm is V-competitive. We first show that the correct competitive ratio for any deterministic non-idling algorithms is in fact $V+1$, rational or not.

Consider the following counterexample with two packets $j_1(1,3,1), j_2(2,2,V)$, battery capacity $C=1$, initial energy $e(1)=1$, and harvesting energy $h(3)=1$ and $h(t)=0$ for any other t. A non-idling algorithm must send j_1 at time 1 and then cannot send j_2. OPT would send j_2 and j_1 at time 2, 3 respectively, obtaining a profit of $V+1$. Thus the competitive ratio is at least $V+1$.

The following lemma is useful for a number of results later on. Given the schedules of OPT and that of an online algorithm A, we say a time step t is an

OPT-only step if OPT sends a packet at t, but A does not despite having at least one pending packet, because it has no energy. We call a time step *A-only* if A sends a packet, but OPT does not despite having at least one unit of energy.

Lemma 1. *For the k-th OPT-only step in the schedule, there must be at least k A-only steps before it in the schedule.*

Theorem 5. *Any non-idling algorithm is $(V+1)$-competitive.*

Proof. We consider how to charge the values of packets sent by OPT to those by the online algorithm A. Any packet sent by OPT is charged to itself in A if it is also sent by A. If at time t OPT sends a packet x that A does not send, and A sends another packet y instead at this time step, then charge x to y. Clearly $v(x)/v(y) \leq V$. If at time t OPT sends x but A idles because it has no pending packets, then x must have been sent by A already and therefore its value is already charged. Thus the only remaining case is when OPT sends x but A idles because it has no energy to send any packet, i.e., it is an OPT-only step.

Suppose there are a total of k OPT-only steps. By Lemma 1, there are at least k A-only steps. We charge each of the k packets in these OPT-only steps to each of these k packets in A in A-only steps (in some arbitrary way). Again, if x is the packet in OPT making the charge and y is the one in A receiving it then $v(x)/v(y) \leq V$.

Each packet in A is charged by at most two packets: one which is itself, and the other either from OPT in the same time step, or from some OPT-only time steps, but not both. Thus the ratio of total charges received by a packet to the value of the packet sent by A is at most $V+1$. This shows that A is $(V+1)$-competitive. $\square$

We can also easily prove matching randomized upper and lower bounds of $\Theta(\log V)$:

Theorem 6. *Against unrestricted adversaries, any randomized algorithm is $\Omega(\log V)$-competitive. There exists an $O(\log V)$-competitive randomized algorithm.*

4.2 Unweighted instances

It might appear that if packets are unweighted, *EDF* is optimal. However it is not the case: following the same example in the beginning of the previous subsection, *EDF*, or any non-idling algorithm, is not better than 2-competitive. It also follows from Theorem 5 that any non-idling algorithm is 2-competitive.

It can be observed from those examples that such 'deadline inversion' is the problem to getting optimal schedules. We formalise this by showing that for instances with *agreeable deadlines*, i.e. $d(i) < d(j)$ implies $r(i) \leq r(j)$, *EDF* is 1-competitive against unrestricted adversaries. Note that *EDF* is 1-competitive for unweighted instances against non-idling adversaries (without the agreeable deadline assumption) since neither OPT nor the online algorithm can idle and clearly it is best to send the packet with the earliest deadline when it is the only thing that distinguishes packets. Therefore, in a sense we can replace the requirement of a non-idling adversary with agreeable deadlines to get to 1-competitiveness. Note that agreeable deadline instances include the case where all packets have the same 'lax time' $(d(j) - r(j))$ as a special case.

Similar to Theorem 3, we use IDs as a consistent way of tie-breaking deadlines. We assume *EDF* prefers packets with earlier release times among those that have the same deadline, and if release times are also equal, then the one with a smaller ID. We say $x \prec y$ if $d(x) < d(y)$, or $d(x) = d(y)$ and $r(x) < r(y)$, or $d(x) = d(y)$ and $r(x) = r(y)$ and $ID(x) < ID(y)$.

We also assume OPT follows a canonical structure, in that: (i) if it sent a packet x at time t_1 before sending a packet y at time t_2, and $r(y) \leq t_1$, then it must be that $x \prec y$; (ii) OPT does not idle unnecessarily, i.e., if OPT was idle at t_1 and sends a packet x at a later time step t_2, then it must be that x cannot be moved earlier to t_1 without affecting other parts of the schedule (e.g. due to energy availability), or that simply x was not released at t_1, or that there is no energy available at t_1. Both assumptions are without loss of generality by applying standard exchange arguments.

Lemma 2. *Let $e^*(t)$ and $e(t)$ be the energy in the battery of OPT and EDF at time t respectively. Then at any time t,*
Claim 1: $e^*(t) \geq e(t)$.
Claim 2: if OPT sent a packet x at t then EDF could not send x before t.

Proof. We prove both claims together by induction on t. Both claims are obviously true for the first time step $t = 1$. It is also easy to see that Claim 1 is true for $t = 2$: it can only be falsified if OPT sent a packet at time 0 but *EDF* idles, but they have the same starting energy and the same set of pending packets, so *EDF* must also send a packet if OPT can.

Suppose Claim 1 is true for all time steps up to and including t, and Claim 2 is true for all time steps up to but excluding t. Claim 1 is true for time $t + 1$ unless OPT sends a packet x at t but *EDF* idles. It is also true if any idling of *EDF* is due to that it has no energy $(e(t) + h(t) = 0)$. But if $e(t) + h(t) > 0$, *EDF* will send x instead of staying idle unless x has already been sent. Hence it remains to prove that x cannot have been sent earlier in *EDF*, i.e., to prove Claim 2 is true at time t.

So suppose x was sent by *EDF* at time $t' < t$. Consider the two cases.

Case 1: OPT is idle at t'. *EDF* has the energy to send a packet, so $e(t') + h(t') > 0$, and applying the induction hypothesis of Claim 1 to time t', $e^*(t') \geq e(t')$. Hence $e^*(t') + h(t') > 0$ and OPT has the energy to send a packet at t'. So the only reason why x is not sent by OPT at t' must be that during $(t'..t]$, there is an *energy-critical time step*, i.e. a step s where $e^*(s) + h(s) = 1$ and a packet z is sent by OPT there, so that if x was sent at t' instead it would use up one unit of energy and z then could not be sent at s. Furthermore assume s is the earliest such energy-critical step in $(t'..t]$. We have $z \prec x$ since otherwise OPT would swap x and z. Hence either $d(z) < d(x)$, which implies $r(z) \leq r(x)$ by the definition of agreeable deadlines, or $d(z) = d(x)$ and the definition of $\prec$ also implies $r(z) \leq r(x)$. But then z could have been sent by OPT at t' because it has energy available and because there are no other energy-critical step between t' and s. Hence there is a contradiction.

Case 2: OPT sent a packet y at t'. y must still be pending in *EDF* at t' by induction hypothesis on Claim 2, yet *EDF* chooses to send x, hence $x \prec y$. But then OPT would have swapped x and y (note that $d(x) \leq d(y)$). □

Theorem 7. *EDF is* 1*-competitive for unweighted instances with agreeable deadlines (against unrestricted adversaries).*

Proof. Consider each packet x sent by OPT at a time step t. If *EDF* sends some packet at t, charge x to that packet. Otherwise, *EDF* idles despite the fact x is still pending (by Claim 2 of Lemma 2), so it can only be because it has no energy, i.e., it is an OPT-only step. By Lemma 1, there must be at least as many A-only steps as OPT-only steps, so pair them up arbitrarily and charge the packet values as in the proof of Theorem 5.

Any packet sent by *EDF* can only receive charge from one other packet: if it is an A-only step that it only receives from a packet in an OPT-only step, and if it is a step where both OPT and A send packets then it gets charged from the corresponding packet in OPT. □

As a note, this automatically means that *EDF* is V-competitive for weighted, agreeable-deadline instances.

5 Network Topologies

Here we consider a network with more than one node. We will restrict ourselves to unweighted packets. We use the notation $(r(j), d(j), s(j), t(j))$ for a packet j where $s(j)$ and $t(j)$ are the source and destination nodes. We use $h_N(t)$ to denote the energy harvesting function for node N.

The situation is already very bad even for unweighted instances:

Proposition 2. *The competitive ratio is unbounded even for line networks and even for unweighted instances if packets have different destinations.*

Proof. Consider a line network with four nodes a, b, c, d and two packets $p_1(1, 3, a, c)$, $p_2(1, 5, a, d)$. All batteries are initially empty. We have $h_a(1) = 1$ and $h_a(t) = 0$ for $t \geq 2$, $h_b(1) = h_c(1) = 0$. Hence an online algorithm A can only send one of the two packets. If A sends p_1, then $h_b(2) = 0$, so p_1 will expire. OPT sends p_2 instead, with $h_b(3) = 1, h_c(4) = 1$. If A sends p_2 instead, then $h_b(2) = 1$ but $h_c(t) = 0$ for all t, so p_2 expires while OPT sends p_1. □

Proposition 3. *EDF has an infinite competitive ratio even when all packets have the same source and destination in a line network with only three nodes.*

Proof. Consider a line network a, b, c and two packets $p_1(1, 3, a, c), p_2(1, 4, a, c)$. Again nodes have empty batteries initially, $h_a(1) = 1$ and $h_a(t) = 0$ afterwards, and $h_b(t) = 0$ for all $t \neq 3$ and $h_b(3) = 1$. *EDF* sends p_1 first, but node b has no energy at time 2 and hence p_1 expires, and node a has no energy at time 2 onwards so p_2 also expires. *OPT* sends p_2 at time 1, waits at node b at time 2 until it has energy at time 3. Thus *EDF* gets 0 while *OPT* gets 1. □

To try to get around this, we make an additional assumption that the instance is underloaded. We note that it is quite common in the real-time systems community to consider underloaded instances. However we still have the following:

Proposition 4. *For a line network where all packets have a common destination (the sink), any non-idling algorithm is at least $(n + 1)$-competitive for unweighted and underloaded instances against unrestricted adversaries, where n is the number of nodes (excluding the sink).*

Proof. Consider a line network with n nodes (in this order) $N_0, N_1, ..., N_n$ where N_0 is the sink. Each node $N_1..N_n$ have $C = 1$, initial battery energy 0 and the following energy harvesting function: $h(1) = 1, h(t) = 0$ for $2 \leq t \leq n + 1$, and $h(t) = 1$ for $t \geq n + 2$. Packet p_0 is released to node N_n with $r(p_0) = 1$ and $d(p_0)$ very large. For each $1 \leq i \leq n$, packet p_i is released to node N_i with $r(p_i) = n + 1$ and $d(p_i) = n + i + 1$. These packets are tight, i.e., they must be forwarded immediately at every node to reach N_0 in time. A non-idling algorithm will send p_0 along the line from time 1 to n, consuming the only unit of energy at each node along the way. Then when the tight packets arrive at time $n + 1$, they cannot be forwarded immediately and hence all are lost. *OPT* withholds p_0 and stays idle up to and including time n. At time $n + 1$ it forwards each of $p_1..p_n$ by one node. Starting at time $n + 2$ all nodes have plenty of energy, so they continue to forward packets $p_1..p_n$ to the sink. Finally p_0 is sent. □

We believe the bound is indeed tight, i.e., for underloaded instances any non-idling algorithm is $O(n)$-competitive in line networks with a common sink, or even for uplink trees where n is the total number of vertices. Note that without energy limitations EDF is 1-competitive for uplink trees, but for arbitrary non-idling algorithms it can also be as bad as $(n + 1)$-competitive. Also, it is not true that the competitive ratio may be upper bounded by the depth of the tree rather than the number of nodes: we have an example to show that any non-idling algorithm is $\Omega(n)$-competitive for an uplink tree even with a depth of 2.

6 Conclusion

Most importantly we want to get an upper bound in the case of uplink trees or at least line networks. In the single node case, it is interesting to see whether there are other ways to get non-trivial competitiveness with reasonable assumptions. The power of randomized algorithms, or algorithms that choose to idle, remain to be investigated. For example in the unrestricted adversary case, it is not clear whether it is possible to get (idling) algorithms with competitive ratio better than $V + 1$; or for non-idling algorithms, what are the upper bounds.

References

1. Borodin, A. and El-Yaniv, R.: Online Computation and Competitive Analysis. Cambridge University Press, New York (1998)
2. Chin, F. Y. L., Chrobak, M., Fung, S. P. Y., Jawor, W., Sgall, J. and Tichý, T.: Online Competitive Algorithms for Maximizing Weighted Throughput of Unit Jobs. Journal of Discrete Algorithms 4(2), 255–276 (2006)
3. Chin, F. Y. L., Fung, S. P. Y.: Online Scheduling with Partial Job Values: Does Timesharing or Randomization Help? Algorithmica 37(3), 149–164 (2003)
4. Englert, M., Westermann, M.: Considering Suppressed Packets Improves Buffer Management in Quality of Service Switches. SIAM Journal on Computing 41(5), 1166–1192 (2012)
5. Goldwasser, M. H.: A Survey of Buffer Management Policies for Packet Switches. SIGACT News 45(1), 100–128 (2010)
6. Hajek, B.: On the Competitiveness of Online Scheduling of Unit-Length Packets with Hard Deadlines in Slotted Time. In: Proceedings of 35th Annual Conference on Information Sciences and Systems, pp. 434–438 (2001)
7. Kesselman, A., Lotker, Z., Mansour, Y., Patt-Shamir, B., Schieber, B., Sviridenko, M.: Buffer Overflow Management in QoS Switches. SIAM Journal on Computing 33(3), 563–583 (2004)
8. Lei, J., Yates, R., Greenstein, L.: A Generic Model for Optimizing Single-Hop Transmission Policy of Replenishable Sensors. IEEE Transactions on Wireless Communications 8(2), 547–551 (2009)
9. Li, F.: Personal communication (2014)
10. Mao, Z., Koksal, C. E., Shroff, N. S.: Optimal Online Scheduling with Arbitrary Hard Deadlines in Multihop Communication Networks. In: Proceedings of IEEE INFOCOM, pp. 2463–2471 (2013)
11. Moser, C., Brunelli, D., Thiele, L., Benini, L.: Real-time Scheduling for Energy Harvesting Sensor Nodes. Real-Time Systems 37, 233–260 (2007)
12. Wang, H., Zhang, J. X., Li, F.: Worst-Case Performance Guarantees of Scheduling Algorithms Maximizing Weighted Throughput in Energy-Harvesting Networks. Sustainable Computing: Informatics and Systems 4, 172–182 (2014)
13. Wang, H., Zhang, J. X., Li, F.: Corrigendum to "Worst-case Performance Guarantees of Scheduling Algorithms Maximizing Weighted Throughput in Energy-Harvesting Networks". Sustainable Computing: Informatics and Systems 5, 64 (2015)
14. Zhu, A.: Analysis of Queueing Policies in QoS Switches. Journal of Algorithms 53, 137–168 (2004)