Efficient Scheduling for Energy-Delay Tradeoff on a Time-Slotted Channel

Yanting Wu[1], Rajgopal Kannan[2], Bhaskar Krishnamachari[1]

1. Ming Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles CA 90089, USA. Email: yantingw@usc.edu, bkrishna@usc.edu

2. Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803

Email: rkannan@bit.csc.lsu.edu

Computer Engineering Technical Report Number CENG-2015-03

Ming Hsieh Department of Electrical Engineering – Systems
University of Southern California
Los Angeles, California 90089-2562

April, 2015

Efficient Scheduling for Energy-Delay Tradeoff on a Time-Slotted Channel

Yanting Wu*, Rajgopal Kannan†, Bhaskar Krishnamachari*

*Dept. of Electrical Engineering, University of Southern California
{yantingw,bkrishna}@usc.edu

†Dept. of Computer Science, Louisiana State University
rkannan@bit.csc.lsu.edu

***Abstract*—We study the fundamental problem of scheduling packets on a time-slotted channel to balance the tradeoff between a higher energy cost incurred by packing multiple packets on the same slot, and higher latency incurred by deferring some packets. Our formulation is general and spans diverse scenarios, such as single link with rate adaptation, multi-packet reception from cooperative transmitters, as well as multiple transmit-receive pairs utilizing tunable interference mitigation techniques. The objective of the system is to minimize the weighted sum of delay and energy cost from all nodes. We first analyze the offline scheduling problem, in which the traffic arrivals are given in advance, and prove that a greedy algorithm is optimal. We then consider the scenario where the scheduler can only see the traffic arrivals so far, and develop an efficient online algorithm, which we prove is O(1)-competitive. We validate our online algorithm by running simulations on different traffic arrival patterns and penalty functions. While the exact competitive ratio in principle depends on the energy function, in our simulations we find this ratio to be always less than 2.**

Keywords: greedy algorithm, online scheduling, competitive analysis, energy-delay tradeoff

I. Introduction

During the past two decades, we have witnessed a fast growth in wireless networks and the continuous increasing demand of wireless services. A key tradeoff in these networks is that between energy and delay.

In this paper, we formulate, study, and solve a fundamental problem pertaining to the energy-delay tradeoff in the context of transmissions on a time slotted channel. In a general form, this problem consists of multiple independent nodes with arbitrary packet arrivals over time. At each slot, every node decides at the beginning of a slot whether to send one or more of its queued as well as newly arrived packets, or to defer some or all of them to a future slot. The more the total number of packets sent on a given slot, the higher the energy cost (modeled as an arbitrary strictly convex function of the total number of simultaneously scheduled packets); on the other hand, deferral incurs a delay cost for each deferred packet.

In this system, there is a centralized scheduler, who knows the arrivals and schedules the various arriving packets to different slots in order to minimize a cost function which combines both the deferral penalty and the energy penalty. We first assume that the centralized scheduler has a complete knowledge about the packet arrivals, and develop a backward greedy algorithm, which is proved to be optimal. Then we consider a more realistic scenario, in which the centralized scheduler only knows about the arrivals up to the current time slot, and develop an efficient online algorithm for this scenario, which is proved to be O(1)-competitive of the optimal.

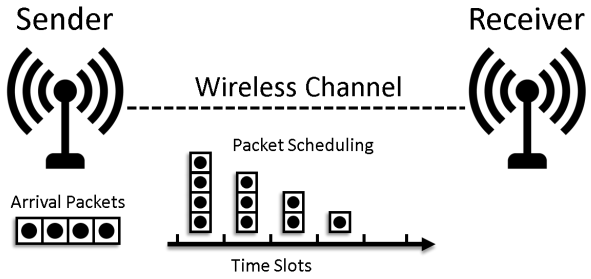

Figure 1: Illustration Example

The paper is organized as follows: section II gives a concrete motivating example; section III reviews some related work; section IV introduces the problem and the model; section V investigates the optimal policy for a centralized scheduler when all the arrivals are known in advance; section VI considers the scenario when the centralized scheduler only knows about the arrivals so far, we develop an efficient online algorithm and prove that this algorithm is $O(1)$-competitive; section VII tests the online algorithm by running a large number of simulations with different traffic arrival patterns and energy functions; finally, we conclude the paper in section VIII.

II. Motivating Example

We present one example which motivates the general problem we investigate in this work. Consider the simplest case where there is one sender and one receiver, and there is a single time-slotted channel between the sender and the receiver, shown in figure 1. There are arrivals over time. The sender can send multiple packets at a higher power cost, or defer some of them, incurring a delay penalty for each packet for each time-slot it spends in the queue.

For a single channel, the rate obtained can be modelled

Table I: Comparison of three scheduling scheme

Scheduling scheme	Deferral	Energy	Total Penalty
Batch: (10,0,0,0,0)	0	32	31
Constant rate: (2,2,2,2,2)	20	5	25
Optimal: (4,3,2,1,0)	10	6.2	16.2

as $R = B \cdot log_2(1 + \frac{P_r}{N})$, where B is the bandwidth, P_r is received power, and N is noise power. The transmission power $P_t \propto P_r$. Assume N is a constant, $P_t \propto 2^{\eta R}$, where $\eta = \frac{1}{B}$. We can consider R as the number of packets sent in a time slot. For illustration, we consider three options for the sender: batch scheduling (sending all packets in one slot), constant rate scheduling (sending packets with a constant rate), and dynamic scheduling (sending different number of packets in different slots). 10 packets arrive at once, and there are 5 slots to use. We use a linear function to represent the deferral penalty (i.e. a packet arrives at the j^{th} slot is transmitted in $k^{th}(> j)$ slot, the deferral penalty for this packet is $k - j$). We use $f(x) = 2^{0.5x} - 1$ to represent the transmission power, where x is the number of packets scheduled in the same slot.

Let $(S_1, S_2\, S_3, S_4\, S_5)$ denote the number of packets scheduled in 5 slots. The scheduling for the three illustrative schemes and the corresponding cost is as shown in Table I.

From Table I, we can see that the batch scheduling has small deferral penalty but high energy cost, while the constant rate scheduling has small energy cost but high deferral penalty. These two scheduling are more costly than the optimal. Intuitively, we should schedule more packets in earlier slots than later slots to balance the tradeoff between the deferral penalty and the energy cost. However, in a real system, in which the arrivals can happen at any slot, the optimal scheduling is not obvious. In this paper, we investigate this problem, and try to find the optimal or close-to-optimal scheduling for any random arrivals. The problem is particularly challenging when considering the online case, where scheduling and deferring decision must be made without knowledge of future arrivals.

Other examples that fit the general formulation we introduce and solve in this paper include multi-packet reception from cooperative transmitters (in which it is assumed that higher numbers of packets spent at the same slot incur a higher energy cost), as well as multiple sender-receiver pairs employing interference mitigation strategies (such as successive interference cancellation) where too there is a higher energy cost for allowing multiple interfering packets in a given slot. Though in wireless network, the transmission power grows exponentially with the rate (the number of packets scheduled in a slot), our model is in fact even more general, as the only requirement we impose on the energy cost is that it be strictly convex and increasing.

III. Related Work

We briefly review some papers in the literature that have treated similar problems. One related work is the energy efficient packet transmission in wireless network. For a wireless fading channel, to achieve optimal power allocation, a commonly used approach is water-filling over time [1]–[3]. Since power affects the rate, the optimal adaptive technique should use variable rate based on channel conditions. With high power, the transmission time is shorter, however, the energy consumed is higher. In these previous works, the objective is mainly on minimizing the total transmission power either without considering the delay, or considering the delay only has some deadline constraint [4]. Our work takes into account both side of the tradeoff: transmission time (reflected in deferral penalty) and energy consumption. Another difference is that the previous works assume that the channel conditions changes over time, but we do not have such an assumption in our problem. In future work, we also plan to consider the variation of the channel condition, which could be modeled by having a time-varying energy cost.

Another highly related work is dynamic speed scaling [10]–[15]. In a speed scaling problem, the objective is to minimize a combination of total (possibly weighted) flow (i.e. the number of unfinished job) and total energy used. Speeding up can reduce the flow processing time. However, it will consumes more energy. This indicates a flow versus energy trade-off. In [15], Bansal *et al.* study a problem which minimizes the integral of a linear combination of total flow plus energy. The authors propose an online algorithm which schedules the unfinished job with the least remaining unfinished work, and runs at speed of the inverse of the energy function. By using the amortized competitive analysis [12], the authors prove that this online algorithm is 3-competitive. Andrew *et al.* improve the results to be 2-competitive in [13] by using a different potential function in the amortized competitive analysis. The online algorithm proposed in this paper is based on [15]. However, due to the discrete feature of our problem (i.e. the number of packets scheduled in each slot has to be an integer, and cost updates happen at the integer time point), the analysis of the performance of our online algorithm is more challenging and needs additional steps compared to [15].

Packet scheduling in an energy harvesting communication system is also related. To efficiently use the harvested energy, it requires the scheduler to dynamically adapt the transmission rate. [5]–[8] study the optimal offline policy when the energy arrivals are assumed to be known in advance. In [9], Vaze removes such an assumption, and develops an 2-competitive online algorithm which dynamically adapts the transmission rate to minimize the total transmission time under the energy constraint. Our work also dynamically changes transmission rate, however, unlike energy harvesting, where the transmissions are constrained by the total energy arrived so far, in our problem, we do not have such an energy constraint. However, we still want to efficiently use the energy, and this is reflected in the energy function in our problem.

Load Balancing in data centers is also related [16]–[18]. Wierman *et al.* model an Internet-scale system as a collection of geographically diverse data centers and consider two types of costs: operation costs (energy costs plus delay costs) and switching costs. The offline problem in which the scheduler has all future workload information is modeled as convex

optimization problem and the optimal is achieved by solving the convex problems backwards in time [17]. Based on the structure of the offline optimal, the authors develop efficient online algorithms for dynamic right-sizing in data centers. Similarly, our problem also has energy costs and delay costs, however, we do not have a switching cost. The way we get the offline optimal is also schedule backward in time, however, our algorithm for this problem is greedy, much simpler in calculation. Another difference is that Wierman *et al.*'s work are in continuous domain, while ours is discrete.

IV. Problem Formulation

Consider a wireless network in which every node interferes with each other. The channel in this network is time slotted. Packets arrive at the beginning of a time slot, and are scheduled by a centralized scheduler. A packet can be transmitted immediately at the same slot as its arrival, in which case the transmission deferral is 0, or in a latter slot, in which case the transmission deferral is a positive number. We consider that the deferral penalty is linear. A packet transmitted in a slot is affected by the "interference" from all packets which are transmitted at the same time slot. This is modeled as an energy cost that is purely determined by the number of packets transmitted in the same slot.

We use $f(X)$ to represent the energy cost in a slot, where X is the number of the packets sent at the same time slot. $f(X)$ is assumed to be any function that is strictly convex and increasing with X.

We define the total penalty $J(\mathcal{A})$ as weighted sum of the deferral penalty and the energy cost, as shown in the Eq. 1.

$$J(\mathcal{A}) = w_1 \sum_{i=1}^{N} d_i + w_2 \sum_{j=1}^{M} f(X_j), \tag{1}$$

where w_1, w_2 are the weights, and M is the number of slots (M can be ∞), N is the total number of packets arrived. The objective of the centralized scheduler is to minimize $J(\mathcal{A})$.

Let $w = \frac{w_2}{w_1}$, maximizing $J(\mathcal{A})$ is equivalent to minimizing the total cost of $C(\mathcal{A})$, as shown in the Eq.2

$$C(\mathcal{A}) \quad = \quad \sum_{i=1}^{N} d_i + w \sum_{j=1}^{M} f(X_j). \tag{2}$$

Please note that in this model, what really matters is the number of packets scheduled in each slot and the aggregate number of deferrals for each packet; the model is not concerned with where the packets are from, and which particular nodes or packets should be scheduled at which slot.

V. Offline

A. Greedy Algorithm for A Single Arrival

In this subsection, we consider the scenario that there is a single arrival. We can start our timing $t = 1$ at the slot of the first arrival. We assume that the number of packets in this single arrival is N, and it can be scheduled to any slot $t \geq 1$. The Greedy algorithm, denoted as $\mathcal{G}(N)$, is as shown in algorithm 1.

Algorithm 1 Greedy Algorithm for A Single Arrival

Initialization:

- N packets arrive at the 1^{th} time slot; we number them as packet $1, 2, \cdots, N$.
- The number of packets scheduled in each slot: $X_j = 0$ for $j = 1, 2, \cdots$.
- Total cost $C = 0$.

Greedy Schedule:

for Packet index $i = 1, 2, \cdots, N$ **do**

Put packet i in the slot which minimizes the marginal cost

Update the number of packets in the selected slot

Update the total cost

end for

If there is a tie in minimizing the marginal cost, the slot with the smallest index is selected, which ensures that Greedy algorithm is unique since the slot selected at each step is uniquely determined.

Let $mc_j^{(i)}$ denote the marginal cost to put the i^{th} packet in the j^{th} slot. We can get the following lemmas from the Greedy algorithm.

Lemma 1. *The least marginal cost for Greedy algorithm keeps increasing.*

Proof: Let us compare the marginal cost to schedule the i^{th} packet and the $(i+1)^{th}$ packet. When scheduling the $(i+1)^{th}$packet, the number of packets in each slot is exactly the same as scheduling the i^{th} packet, except one slot, in which the i^{th} packet is put. We assume that the i^{th} packet is put in the k^{th} slot. $mc_j^{(i)} \geq mc_k^{(i)}$ for $j \neq k$. If the $(i+1)^{th}$ packet is scheduled in a slot $j \neq k$, we know $mc_j^{(i+1)} \geq mc_k^{(i)}$.

If the $(i+1)^{th}$ packet is scheduled in slot k, let us now compare the marginal cost of putting the $(i+1)^{th}$ packet in k^{th} slot with the marginal cost of putting the i^{th} packet in k^{th} slot. Since the energy cost function $f(X)$ is convex and increasing, we have

$$mc_k^{(i+1)} - mc_k^{(i)} = w(f(X_k + 1) - 2f(X_k) + f(X_k - 1)) \geq 0,$$

where X_k is the number of packets in slot k after scheduling the i^{th} packet in the k^{th} slot. $mc_k^{(i+1)} \geq mc_k^{(i)}$.

Thus, $min_{j=1,2...}(mc_j^{(i)}) \leq min_{j=1,2...}(mc_j^{(i+1)})$, the least marginal cost for Greedy algorithm keeps increasing. ■

Lemma 2. *Assume that $(..., g_1, g_2, ...g_k, ...)$ is a schedule got from Greedy algorithm, and $(..., g_1^{'}, g_2^{'}...g_k^{'}, ...)$ is also a schedule got from Greedy algorithm, if $\sum_{t=1}^{k} g_t \leq \sum_{t=1}^{k} g_t^{'}$, we can conclude that $g_i \leq g_i^{'}$ for $i = 1, ..., k$.*

Proof: When a greedy algorithm selects a slot in $[t+1, t+2, ..., t+k]$, it selects the one with the least marginal cost, and

the index is the smallest if there is a tie. Since such a slot is unique, the schedule order of applying Greedy algorithm in $[t+1, ..., t+k]$ is uniquely determined. Since $\sum_{t=1}^{k} g_t \leq \sum_{t=1}^{k} g_t^{'}$, the schedule has to reach $(..., g_1, g_2, ...g_k...)$ first, which indicates $g_i \leq g_i^{'}$ for $i = 1, ..., k$. ■

Define Separate Greedy (denoted as $\mathcal{SG}$) algorithm as follows: to schedule N packets on M slots. Here, we borrow the notation M for simplicity of the following discussions, M is large enough, for example, the maximum energy cost in a slot is bounded by $f(N)$, when $M > wf(N)$, this indicates that there will be no packet scheduled in slot M in the optimal scheduling. We separate the N packets to be two parts: $N-k$, k; we also separate the slots to be two parts: the first m, the last $(M-m)$. The $(N-k)$ packets are scheduled in the first m slots, while the k packets are scheduled in the last $(M-m)$ slots. The problem becomes $(N-k, m)$ and $(k, M-m)$. We apply Greedy algorithm separately on each part. Please note, when we apply Greedy algorithm on $(k, M-m)$, we construct a fictitious arrival which assumes that the k packets arrives at the $(m+1)^{th}$ slot, denoted as Fictitious arrival.

Let $\mathcal{G}(M, N) = (g_1, g_2, ...g_m, g_{m+1}, ..., g_M)$ denote the schedule result of applying Greedy algorithm to schedule N packets in M slots. Let $\mathcal{SG}(m, N-k;\ M-m, k) = (g_1^{'}, g_2^{'}, ...g_m^{'}; g_{m+1}^{'}, g_{m+2}^{'}, ...g_M^{'})$ denote the schedule result of applying Separate Greedy algorithm. The cost of $\mathcal{SG}$ is defined as $C(\mathcal{SG}(m, N-k;\ M-m, k)) = C(\mathcal{G}(m, N-k)) + C(\mathcal{G}(M-m, k)) + km$.

Lemma 3. $C(\mathcal{G}(M, N)) \leq C(\mathcal{SG}(m, N-k;\ M-m, k))$**.**

Proof: Since the delay is linear, compare the original marginal cost when the arrival happens at the 1^{st} slot with the Fictitious arrival, the difference is a constant m. Thus, the order of selections of Greedy algorithm on the Fictitious arrival will be the same as applying Greedy algorithm on the original problem.

Assume $k' = \sum_{t=m+1}^{M} g_t$. If $k' = k$, then those two algorithms give the same solution since the Greedy algorithm result is unique. If $k' < k$, compare $\mathcal{SG}$ algorithm with $\mathcal{G}$, the first m slot lacks $(k-k')$ steps and the cost reduction is no more than $(k-k')mc^*$, where mc^* is the largest least marginal cost, i.e. the N^{th} step marginal cost in $\mathcal{G}$; the last $(N-m)$ slots obtain $(k-k')$ more steps and the cost increase is no less than $(k-k')mc^*$. Since $\mathcal{G}$ stops at the k^{th} step for the second part and the marginal cost always increases.

Therefore, the total cost of $\mathcal{SG}$ is no less than $\mathcal{G}$ for $k' < k$. Similarly, we can prove $k' > k$.

In other words, considering that we sort the two parts schedule steps of $\mathcal{SG}$ based on the increasing of the marginal cost. There will be no swap after the sorting, i.e. in $\mathcal{SG}$, step k is before step m, after sorting, this still hods. We compare the cost of each step using $\mathcal{SG}$ with $\mathcal{G}$ one by one. The step cost of $\mathcal{SG}$ is no less than $\mathcal{G}$. Thus, $C(\mathcal{G}(M, N)) \leq C(\mathcal{SG}(m, N-k;\ M-m, k))$**.**

■

Theorem 4. *Greedy algorithm is optimal.*

Proof: Let $OPT(M, N)$ =$(o_1, o_2, ...o_m) = OPT(m, N)$ be the optimal, where $m \leq M$, $o_m > 0$, and $\sum_{t=t_1}^{m} o_t = N$. Once the o_m is fixed, the total cost is determined by the schedule of the remaining $(m-1)$ slots. To minimize the total cost of the scheduling N packets in M slots is equivalent to minimizing the total cost of scheduling the remaining $(N - o_m)$ packets in the first $(m-1)$ slots. Thus, $OPT(m, N) = (OPT(m-1, N-o_m), o_m)$.

Let $\mathcal{G}(m, N) = (g_1, g_{2,}...g_m)$ be the Greedy schedule for schedule N packets in m slots, $\sum_{t=t_1}^{m} g_t = N$.

We use the induction to prove the theorem.

Basis step: $N = 1$, $OPT(M, 1) = (1)$; $\mathcal{G}(M, 1) = (1)$.

Inductive step: Assume that for k packets, $0 < k < N$, the Greedy algorithm can find an optimal solution, then we consider the optimal schedule for $k = N$.

Since $o_m > 0$, $N_{OPT \setminus m} = \sum_{t=1}^{m-1} o_t < N$, the Greedy algorithm can give an optimal solution. $OPT(M, N)$ =$(g_1^{'}, g_2^{'}, ...g_{m-1}^{'}, o_m)$, where the schedule of the first $m-1$ slots is got from Greedy algorithm. $OPT(M, N) = \mathcal{SG}(m-1, N_{OPT \setminus m}; 1, o_m)$. According to lemma 3, $C(\mathcal{G}(m, N)) \leq C(OPT(M, N))$. $C(\mathcal{G}(M, N)) = C(\mathcal{SG}(m, N; M-m, 0))$. Apply lemma 3 again, $C(\mathcal{G}(M, N)) \leq C(\mathcal{G}(m, N))$. Thus, the Greedy algorithm is optimal.

■

B. Backward Greedy Algorithm

In this subsection, we consider the general scenario in which packets arrive at different slots. We assume that there are K arrivals, and let N_i $(i = 1, ..K)$ be the number of packets arrive at each arrival. The Backward Greedy algorithm (denoted $\mathcal{BG}$) is as shown in Algorithm 2.

Algorithm 2 Backward Greedy Algorithm

Initialization:

- The time slot index for new arrivals: $t_1, t_2, \cdots, t_K$
- The number of packets for each new arrival: $N_1, N_2, \cdots, N_K$
- The number of packets scheduled in each slot: $X_j = 0$ for $j = 1, 2, \cdots$.
- Total cost $C = 0$.

Backward Greedy Schedule:

for Arrival index $a = K, K-1, \cdots, 1$ **do**
  for Each packet from the a^{th} arrival **do**
    Put the packet in a slot from t_a to ∞ which minimizes the marginal cost.
    Update the number of packets in the selected slot
    Update the total cost
  end for
end for

We first schedule the last arrival's (the K^{th} arrival) packets by greedily selecting the slot from t_K to ∞ which minimizes the marginal cost, this is equivalent to the case of single arrival. Then we consider the second last arrival's packets (the $(K-1)^{th}$ arrival), and schedule them one by one by selecting the

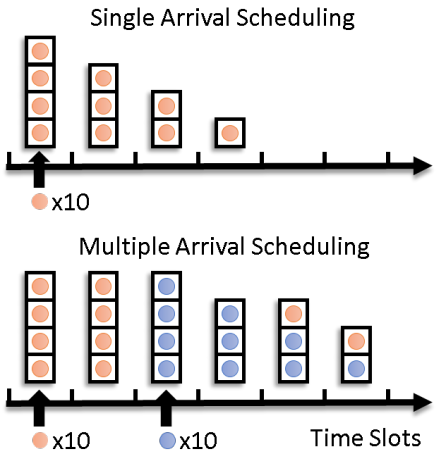

Figure 2: Backward Greedy Algorithm Illustration Example

slot from t_{K-1} to ∞ which minimize the marginal cost. Such a process keeps going until all packets are scheduled.

The figure 2 is a demonstration of how backward greedy works. We consider two arrivals, and compare the scheduling when there is only the first arrival. From the figure 2, we can see due to the second arrival, the packets from the first arrival are pushed to earlier slots. Please note that fairness is not the concern of this paper, so some of the earlier packets could be scheduled later than the later arrived packets. However, as the backward greedy algorithm determines only the number of packets scheduled in each slot, it is still possible to readjust the order of packet transmissions (e.g., to a FIFO order) to enable fairness.

Lemma 5. *In $\mathcal{BG}$, considering the packets at the i^{th} arrival, the marginal cost when scheduling these packets keeps increasing.*

Proof: For the i^{th} arrival, the marginal cost for scheduling a packet in $[t_j, t_{j+1}]$ $(j \geq i)$ keeps increasing based on lemma 1. We can sort the marginal cost without swapping the schedule steps, i.e. step k happens before step m in some $[t_j, t_{j+1}]$ $(j \geq i)$, in the sorted steps, this relationship still holds. The selection order of $\mathcal{BG}$ is exactly the same as based on the sorted marginal cost. ■

Theorem 6. *The Backward Greedy Algorithm is optimal.*

Proof: The total cost is composed of two parts: the total cost of deferral, and the total cost of energy.

We first consider the total cost of deferral, denoted as C_d. Since the deferral cost is a linear function, $C_d = \sum_{t=1}^{M} r_i$, where r_i is the number of packets which are not sent at the i^{th} slots. Whether the packets left are from new arrival or from some old arrival does not affect C_d. Because of this, without considering the fairness, we can freely schedule the new arrival packets first. The energy cost, denoted as C_i, purely depends on the number of packets scheduled in each slot. Thus, backward does not affect the total cost.

We next consider the simplest multiple arrival case, in which there are two arrivals: N_1 packets arrive at t_1, and N_2 packets arrive at t_2, $t_1 < t_2 \leq M$ (M can be ∞).

We assume that the optimal schedule is to put $(N_1 - r)$ packets from slot t_1 to $(t_2 - 1)$, and $(N_2 + r)$ packets from slot t_2 to M. If $r = 0$, the problem is decoupled to be two single arrival problem. According to Theorem 4, the $\mathcal{BG}$ algorithm is optimal.

If $r > 0$, the deferral cost for these r packets is $r(t_2 - t_1)$ at the beginning of slot t_2. Assume that these r packets arrive at slot t_2 (denoted as Fictitious arrival), the total cost is $r(t_2 - t_1)$ less than the original problem. To minimize the total cost of the original problem is equivalent to minimizing the total cost of the Fictitious arrival. This Fictitious arrival can be decoupled as two single arrival problem, and according to Theorem 4, we can apply Greedy algorithm on both single arrival problems to get an optimal solution.

Let $\mathcal{G}_L(N_1 - r, t_2 - t_1) = (g'_{l1}, g'_{l2}, ..., g'_{lk})$ denote the schedule of the $(N_1 - r)$ packets, where $k = t_2 - t_1$; and $\mathcal{G}_R(N_2 + r, M - t_2) = (g'_{r1}, g'_{r2}, ..., g'_{rm})$ denote the schedule of $(N_2 + r)$ packets, where $m = M - t_2$, in $\mathcal{G}_R(N_2 + r, M - t_2)$, we can still consider the N_2 packets are scheduled first. ($\mathcal{G}_\mathcal{L}$,$\mathcal{G}_R$) is an optimal schedule. Assume the $\mathcal{BG}$ algorithm gives a schedule which $(N_1 - r')$ packets is scheduled from slot t_1 to $(t_2 - 1)$, and $(N_2 + r')$ packets from slot t_2 to M.

If $r' = r$, when scheduling the r packets in slots t_2 to M, the marginal cost got from $\mathcal{BG}$ algorithm is the marginal cost got from $\mathcal{G}_\mathcal{R}(N_2 + r, M - t_2)$ in the Fictitious arrival plus a constant $r(t_2 - t_1)$. Since both $\mathcal{BG}$ and $\mathcal{G}_\mathcal{R}$ in the Fictitious arrival select the slots based on the increasing marginal cost, The $\mathcal{BG}$ and ($\mathcal{G}_\mathcal{L}$,$\mathcal{G}_R$) give the same schedule result.

If $r' < r$, considering the schedule of the first arrival packets N_1, compare ($\mathcal{G}_\mathcal{L}$,$\mathcal{G}_R$) with $\mathcal{BG}$ algorithm, the first $(t_2 - t_1)$ slot lacks $(r - r')$ steps and the cost reduction is no more than $(r - r')mc^*$, where mc^* is the largest least marginal cost, i.e., N_1^{th} step marginal cost for schedule the first arrival packets in $\mathcal{BG}$; the last $(M - t_2 + 1)$ slots obtain $(r' - r)$ more steps and the cost increase is no less than $(r' - r)mc^*$ since $\mathcal{BG}$ stops at the r'^{th} step for the second part (schedule packets from t_1 to t_2) and the marginal cost always increases. Thus, the total cost $\mathcal{BG}$ is no less than ($\mathcal{G}_\mathcal{L}$,$\mathcal{G}_R$). $\mathcal{BG}$ is optimal in this case.

Similarly, we can prove for $r' > r$, $\mathcal{BG}$ is optimal.

We finally consider there are N arrivals. $\mathcal{BG}(N_1, \mathcal{BG}(N_2, ...\mathcal{BG}(N_{k-1}, N_k))$.

$\mathcal{BG}(N_{k-1}, N_k)$ is a two arrival case, as we proved, $\mathcal{BG}$ gives an optimal scheduler. Assume that $OPT_{k-1} = \mathcal{BG}(N_{k-1}, N_k)$.

Then we consider schedule of $\mathcal{BG}(N_{k-2}, OPT_{k-1})$, OPT_{k-1} can be considered as a "single" arrival, then the problem becomes a two arrival case.

Similarly, we can recursively prove that $\mathcal{BG}$ is optimal. ■

Please note:

1. The optimal scheduling changes with the set of arrivals. For example, we assume that there are K arrivals, with different K, the optimal scheduling is different, but given a K, we can find the optimal by scheduling backward in time.

2. It is crucial that the energy cost function is convex. It is not difficult to construct a counter-example of a non-convex energy cost function for which the optimality does not hold.

VI. Online

In this section, we propose an efficient online algorithm which is $O(1)$-competitive. Our online algorithm is based on [15]; before we introduce our algorithm, let us briefly review the scheduling mechanism in [15]. In [15], the author develops an online dynamic speed scaling algorithm for the objective of minimizing a linear combination of energy and response time. An instance consists of n jobs, where job i has a release time r_i, and a positive work y_i. An online scheduler is not aware of job i until time r_i, and, at time r_i, it learns y_i. For each time, a scheduler specifies a job to be run and a speed at which the processor is run. They assume that preemption is allowed, that is, a job may be suspended and later restarted from the point of suspension. A job i completes once y_i units of work have been performed on i. The speed is the rate at which work is completed; a job with work y run at a constant speed s completes in $\frac{y}{s}$ seconds. The objective of the online scheduler is to minimize $\int_I G(t)dt$, where $G(t) = P(s^t)+n^t$, s^t is the speed at time t, n^t is the number of unfinished jobs, and I is the time interval. P also needs to satisfy the following conditions: P is defined, continuous and differentiable at all speeds in $[0, \infty)$; $P(0) = 0$; P is strictly increasing; P is strictly convex; P is unbounded.

The authors propose an algorithm as follows: The scheduler schedules the unfinished job with the least remaining unfinished work, and runs at speed s^t where

$$s^t = \begin{cases} P^{-1}(n^t+1) & \text{if } n^t \geq 1 \\ 0 & \text{if } n^t = 0 \end{cases}. \tag{3}$$

Every time when a new job is released or a job is finished, s^t is updated. Please note that in the dynamic speed scaling problem, the job can be released and finished at any time t, where t is a real number. We call this algorithm continuous online algorithm, denoted as $\mathcal{A}_C^{On}$.

In our problem, each packet can be considered as a job with unit work, $P(x) = wf(x)$ in our case, since $f(x)$ satisfies all the conditions for P, so is $P(x) = wf(x)$; n^t is the number of unscheduled packets at time t. We develop our online algorithm, denoted as $\mathcal{A}_D^{On}$ (D means discrete), as Algorithm 3 shown.

In $\mathcal{A}_D^{On}$, the number of packets scheduled at each slot is calculated based on the number of packets scheduled by $\mathcal{A}_C^{On}$. In algorithm A_C^{On}, a transmission of a packet does not necessarily finish at the integer time point. At the end of a slot, if a packet is scheduled across two slots in $\mathcal{A}_C^{On}$, we push the packet in the earlier slot in $\mathcal{A}_D^{On}$, as shown in figure 3.

Though $\mathcal{A}_D^{On}$ is based on $\mathcal{A}_C^{On}$, they are quite different in several aspects:

1. The speed in $\mathcal{A}_C^{On}$ varies in a slot while in $\mathcal{A}_D^{On}$, the speed in a slot can be considered as a constant.

2. The number of packets scheduled in a slot in $\mathcal{A}_C^{On}$ is a real number, while in $\mathcal{A}_D^{On}$, it has to be an integer.

Algorithm 3 Online Algorithm

Initialization:

- The total number of packets scheduled so far based on algorithm A_C^{On}: $n_s = 0$ (n_s can be fractional later)
- The number of packets arrived so far: $n_t = 0$
- The number of packets scheduled in each slot: $X_t = 0$ for $t = 1, 2, \cdots$.
- Total cost: $C = 0$.

Online Schedule:

for $t = 1, 2, \cdots$, **do**

Update n_t if there are new arrivals

Let n be the number of packets waiting to be scheduled at the start of interval t, excluding the fractional packet possibly leftover by A_C^{On} but already scheduled by A_D^{On} in the previous interval. During the interval $[t, t + \Delta t_1]$, A_C^{On} schedules this leftover packet. A_D^{On} tracks A_C^{On}'s schedule from $(t + \Delta t_1)$ onwards using Eq. 3 starting with n remaining packets.

Update n_s based on the number of packets (can be fractional) scheduled in a slot using A_C^{On}.

Update $X_t = \lceil n_s^{t+1} \rceil - \lceil n_s^t \rceil$

Update C based on the schedule of X_t

end for

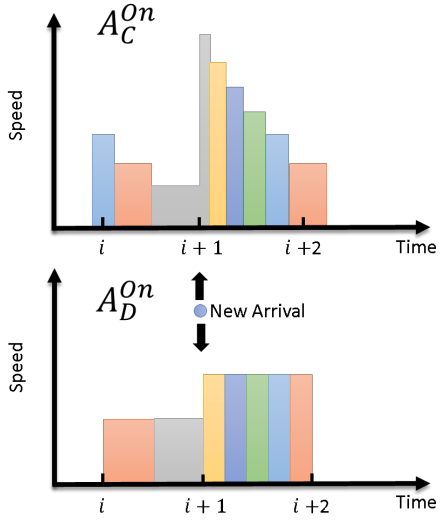

Figure 3: Example of Scheduling in $\mathcal{A}_C^{On}$ and $\mathcal{A}_D^{On}$

3. In $\mathcal{A}_C^{On}$, cost update happens every time when some new packets are coming or a packet is leaving. In other words, the cost is calculated in a continuous way with respect to time. However, in $\mathcal{A}_D^{On}$, the cost is updated at the end of a slot (i.e. the integer time point), which is discrete with respect to time.

Due to these difference, the competitive analysis in this problem is challenging and the idea of amortized competitive analysis used in [12], [15] can not be directly applied to our problem. Thus, we take a different path, which uses $\mathcal{A}_C^{On}$ as a bridge to do the competitive analysis.

Lemma 7. *There exists a constant c such that $\frac{C(\mathcal{A}_D^{On})}{C(\mathcal{A}_C^{On})} \leq c$.*

Proof: First, the packets scheduled by $\mathcal{A}_D^{On}$ and $\mathcal{A}_C^{On}$ is roughly the same, the difference is less than 1 for each slot, and up to any time T, the difference of the total number of packets scheduled by $\mathcal{A}_D^{On}$ and $\mathcal{A}_C^{On}$ is less than 1, so we compare the cost of $\mathcal{A}_D^{On}$ and $\mathcal{A}_C^{On}$ slot by slot.

Let $C(\mathcal{A})^t$ represent the cost at slot t. Assuming that during slot t, $\mathcal{A}_D^{On}$ schedules k packets, the cost of $\mathcal{A}_D^{On}$ is $C(\mathcal{A}_D^{On})^t = P(k) + n - k$.

For algorithm $\mathcal{A}_C^{On}$, besides the possible packet leftover at the beginning of a slot (scheduled in $[t, t + \Delta t_1]$), it is also possible that there is a packet being processed partially at the end of a slot, let Δt_2 be the time interval to process this packet in slot t. Let us assume that there are $k - 1$ packets between these two fractional packets if $\Delta t_2 > 0$; k packets if $\Delta t_2 = 0$.

Case 1: We first consider $\Delta t_2 > 0$

$$\Delta t_1 + \Delta t_2 + \frac{1}{P^{-1}(n+1)} + \frac{1}{P^{-1}(n)} + \cdots + \frac{1}{P^{-1}(n+3-k)} = 1$$

Since P is increasing, $\frac{1}{P^{-1}}$ is decreasing, we have

$$\begin{cases} \frac{k-1}{P^{-1}(n+1)} & \leq 1 \\ \frac{k+1}{P^{-1}(n+2-k)} & \geq 1 \end{cases} \Longrightarrow \begin{cases} P(k-1) & \leq n+1 \\ P(k+1) & +k \geq n+2 \end{cases}$$

As to the cost of the the $\mathcal{A}_C^{On}$, the cost processing the packet when there are n unscheduled packet is $\frac{1}{P^{-1}(n+1)} \times (P(P^{-1}(n+1)) + n) = (2n+1)/P^{-1}(n+1)$

$$\begin{aligned} C(\mathcal{A}_C^{On})^t &\geq \frac{2n+1}{P^{-1}(n+1)} + \cdots + \frac{2(n+2-k)+1}{P^{-1}(n+3-k)} \\ &\geq \frac{1}{P^{-1}(n+1)}(2n+1+\cdots+2(n+2-k)+1) \\ &= \frac{k-1}{P^{-1}(n+1)}(2n-k+3) \end{aligned}$$

The first inequality is by ignoring the cost of the fractional packets at the beginning and the end of a slot. The second equality is by scaling the time interval to process each packet based on the smallest whole packet processing time.

Since $k \leq P^{-1}(n+1)$, the cost of $\mathcal{A}_D^{On}$ is

$$C(\mathcal{A}_D^{On})^t = P(k) + n - k \leq P(P^{-1}(n+1)) + n - k = 2n - k + 1.$$

Since the energy cost function $P(x)$ is strictly convex and increasing, the least expensive function (the function with the slowest growing speed) satisfying such a condition is polynomial function $f(x) = x^\alpha$, where $\alpha > 1$.

$P(x) = wf(x) = wx^\alpha$, let $P(c_1 x) = w(c_1 x)^\alpha \geq w c_1^\alpha x$, as long as $w c_1^\alpha \geq 1$, $P(c_1 x) \geq x$ for $\forall x \geq 1$.

Similarly, $x + P(x+2) \leq P(c_1 x) + P(3x) \leq 2max\{P(c_1 x), P(3x)\} = 2P(c_2 x)|_{c_2 = max\{c_1, 3\}}$, since $P(x)$ is strictly convex and increasing, there exists a constant $c \geq 2c_2$, such that $P(cx) \geq x + P(x+2)$ for $\forall x \geq 1$.

Plug $x = k - 1$ in, we get

$$P(c(k-1)) > P(k+1) + k - 1 \geq n + 1$$

Thus $\frac{k-1}{P^{-1}(n+1)} \geq \frac{1}{c}$.

For other strictly convex and increasing functions which are growing faster than $f(x) = x^\alpha$, similar approach can apply to prove that there exists such a constant c.

$$\frac{C(\mathcal{A}_D^{On})^t}{C(\mathcal{A}_C^{On})^t} \leq c\frac{2n-k+1}{2n-k+3} \leq c.$$

$$\frac{C(\mathcal{A}_D^{On})}{C(\mathcal{A}_C^{On})} = \frac{\sum C(\mathcal{A}_D^{On})^t}{\sum C(\mathcal{A}_C^{On})^t} \leq c.$$

Case 2: $\Delta t_2 = 0$, there are two possibilities to make $\Delta t_2 = 0$: either $\mathcal{A}_C^{On}$ finishes processing all the packets before a slot ends or it just finishes processing a whole packet. In both cases, the number of whole packets scheduled by $\mathcal{A}_C^{On}$ and $\mathcal{A}_D^{On}$ are k. We can use the same approach to prove that there exists a constant c, such that $\frac{C(\mathcal{A}_D^{On})}{C(\mathcal{A}_C^{On})} \leq c$.

Here, we give an example on c using the motivating example in section II, where $P(k) = 2^{0.5k} - 1$. $P(cx) \geq x + P(x+2)$ for $\forall x \geq 1$ when $c \geq 3.88$. ∎

We also use the following lemma which is implied by the results in [15].

Lemma 8. *Assume that the optimal cost for the continuous online scheduling which minimize $\int_I G(t)dt$ is $C(\mathcal{A}_C^{Opt})$, $\frac{C(\mathcal{A}_C^{On})}{C(\mathcal{A}_C^{Opt})} \leq 3$.*

Lemma 9. *Let $\mathcal{A}_D^{Opt}$ denote the optimal scheduling ($\mathcal{BG}$) of our original problem (discrete version), there exists a constant c' such that $\frac{C(\mathcal{A}_C^{Opt})}{C(\mathcal{A}_D^{opt})} \leq c'$.*

Proof: To measure the gap between the cost of our algorithm $C(\mathcal{A}_D^{Opt})$ and the cost of $\mathcal{A}_C^{Opt}$, we introduce an inter-medium scheduling mechanism, which the speed is only updated at the integer point of time, but cost is calculated in an integral fashion. We call such an inter-medium scheduling mechanism fictitious continuous algorithm, denoted as $\mathcal{A}_{FC}^{Opt}$. $\mathcal{A}_{FC}^{Opt}$ works as follows: according to algorithm $\mathcal{BG}$, we know the number of packets sent at each slot: X_t, $t = 1, 2, ...$ $\mathcal{A}_{FC}^{Opt}$ uses the same speed X_t at slot t. Similar to $\mathcal{A}_C^{On}$, the cost is $\int_{t=0}^{T}(P(s^t) + n^t)dt$ up to time T.

Considering slot t, the cost of optimal scheduling is to schedule $k = X_t$ packets in slot t,

$$C(\mathcal{A}_D^{Opt})^t = P(k) + n - k,$$

the cost of $\mathcal{A}_{FC}^{Opt}$ is

$$\begin{aligned} C(\mathcal{A}_{FC}^{Opt})^t &= \frac{1}{k}(P(k)+n) + \cdots + \frac{1}{k}(P(k)+n-k+1) \\ &= P(k) + \frac{2n-k+1}{2} = P(k) + n - k + \frac{k+1}{2}. \end{aligned}$$

Similar to the proof in lemma 7, there exists a constant c', such that

$$(c'-1)(P(k)+n-k) \geq (c'-1)(P(k)) \geq \frac{k+1}{2}$$

Table II: Competitive Ratio in Simulation

Energy cost function	Average	Worst
$f(x) = 0.5x^2$	1.3157	1.6076
$f(x) = x^{1.1}$	1.2971	1.5924
$f(x) = 0.25x^3$	1.2957	1.8169
$f(x) = 0.25(2^x - 1)$	1.1803	1.8375
$f(x) = 2^{0.5x} - 1$	1.1663	1.8547

Thus,

$$\frac{C(\mathcal{A}_{FC}^{Opt})}{C(\mathcal{A}_{D}^{Opt})} = \frac{\sum C(\mathcal{A}_{FC}^{Opt})^t}{\sum C(\mathcal{A}_{D}^{Opt})^t} \leq c'$$

Since the cost of both $\mathcal{A}_{FC}^{Opt}$ and $\mathcal{A}_{C}^{Opt}$ are calculated in the same way,

$$\frac{C(\mathcal{A}_{C}^{Opt})}{C(\mathcal{A}_{FC}^{Opt})} \leq 1.$$

Thus,

$$\frac{C(\mathcal{A}_{C}^{Opt})}{C(\mathcal{A}_{D}^{Opt})} = \frac{C(\mathcal{A}_{C}^{Opt})}{C(\mathcal{A}_{FC}^{Opt})} \frac{C(\mathcal{A}_{FC}^{Opt})}{C(\mathcal{A}_{D}^{Opt})} \leq c'.$$

Here, we give an example on c' using $P(k) = 2^{0.5k} - 1$. Since $n \geq k$, $(c'-1)(P(k)+n-k) \geq \frac{k+1}{2}$ when $c' \geq 3.42$. ■

Theorem 10. *$\mathcal{A}_{D}^{On}$ is O(1)-competitive.*

Proof: From the above lemmas, we get that

$$\frac{C(\mathcal{A}_{D}^{On})}{C(\mathcal{A}_{D}^{Opt})} = \frac{C(\mathcal{A}_{D}^{On})}{C(\mathcal{A}_{C}^{On})} \frac{C(\mathcal{A}_{C}^{On})}{C(\mathcal{A}_{C}^{Opt})} \frac{C(\mathcal{A}_{C}^{Opt})}{C(\mathcal{A}_{D}^{opt})} < 3cc',$$

which is $O(1) - competitive$. ■

VII. Simulations

In this section, we run simulations to test the performance of online algorithm. We mainly use two sets of energy functions: polynomial and exponential and three arrival patterns: burst arrival, constant arrival and random arrival. In these simulations, we change the coefficient of the interference function to represent different weight w.

• $f(x) = 0.5x^2$. • $f(x) = e^{0.5x} - 1$. The simulation results are as shown in Figure 4.

From Figure 4, we can see that with different energy function, the scheduling is different. Although in some slot, the number of packets scheduled by the online algorithm and the optimal can be large, the increased cost is amortized among other slots. Thus, the competitive ratio is small, indicating that the online algorithm's performance is close to the optimal.

We also run the tests more intensively by selecting different energy cost functions and run the simulation over 1000 times for each function. We randomly generate traffic arrival patterns for each run, and record the average and the largest competitive ratio for each function, as shown in Table II. Although the competitive ratio in principle depends on the cost function, in our simulation, we find this ratio is always less than 2.

VIII. Conclusion

In this paper we have studied the optimal centralized scheduling of packets in a time slotted channel to effect desired energy-delay tradeoffs. Under a very general energy cost model that is assumed to be any strictly convex increasing function with respect to the number of packets transmitted in a given slot, and a deferral penalty that is linear in the number of slots each packet is deferred by, we aim to minimize the weighted linear combination of deferral penalties and energy penalties.

We have proved that given the full knowledge of the arrivals, the centralized scheduler can optimally schedule the packets in each slot using a simple greedy algorithm. We also considered the more realistic scenario which the centralized scheduler only knows the arrivals so far, and we have developed an efficient online algorithm which is $O(1)$-competitive.

For future work, we plan to consider distributed scheduling, in which different nodes make independent decisions. This could be potentially modelled in a game-theoretic framework. Another promising direction is to consider a time-slotted channel where the channel conditions varies over time. This could be modeled by a time-varying energy penalty.

IX. Acknowledgment

The first author would like to thank Zheng Li and Li Han for their helpful suggestions and comments.

References

[1] A. J. Goldsmith and P. Varaiya, "Capacity of fading channels with channel side information," *IEEE Trans. Inf. Theory*, vol. 43, no. 6, pp.1986 -1992 1997.

[2] J. Tang and X. Zhang, "Quality-of-service driven power and rate adaptation over wireless links," *IEEE Trans. Wireless Commun.*, vol. 8, no. 8, Aug. 2007.

[3] E. Uysal-Biyikoglu, A. El Gamal, B. Prabhakar, "Adaptive transmission of variable-rate data over a fading channel for energy-efficiency," *Proceedings of the IEEE Global Telecommunications Conference, GLOBECOM*, 21(1):98-102, November 2002.

[4] E. Uysal-Biyikoglu, B. Prabhakar, A. El Gamal, "Energy-efficient packet transmission over a wireless link," *IEEE/ACM Transactions on Networking*, 10(4):487-499, August 2002.

[5] J. Yang and S. Ulukus, "Transmission completion time minimization in an energy harvesting system," in *Proc. 2010 Conf. Information Sciences Systems*, Mar. 2010.

[6] J. Yang and S. Ulukus, "Optimal packet scheduling in an energy harvesting communication system," *IEEE Trans. Commun.*, vol. 60, no. 1, pp. 220–230, Jan. 2012.

[7] O. Ozel, K. Tutuncuoglu, J. Yang, S. Ulukus, and A. Yener, "Transmission with energy harvesting nodes in fading wireless channels: optimal policies," IEEE J. Sel. Areas Commun., vol. 29, pp. 1732–1743, Sep. 2011.

[8] K. Tutuncuoglu and A. Yener, "Optimum transmission policies for battery limited energy harvesting nodes," *IEEE Trans. Wireless Commun.*, vol. 11, no. 3, pp. 1180–1189, Mar. 2012.

[9] R. Vaze, "Competitive Ratio Analysis of Online Algorithms to Minimize Data Transmission Time in Energy Harvesting Communication System," in *Proc. INFOCOM*, April, 2013.

[10] N. Bansal, T. Kimbrel, and K. Pruhs, "Speed scaling to manage energy and temperature," *Journal of the ACM*, 54(1):1–39, Mar. 2007.

[11] Nikhil Bansal, Kirk Pruhs, and Cliff Stein, "Speed scaling for weighted flow time," in *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 805–813, 2007.

[12] Kirk Pruhs, "Competitive online scheduling for server systems," *SIGMETRICS Perform. Eval. Rev.*, 34(4):52–58, 2007.

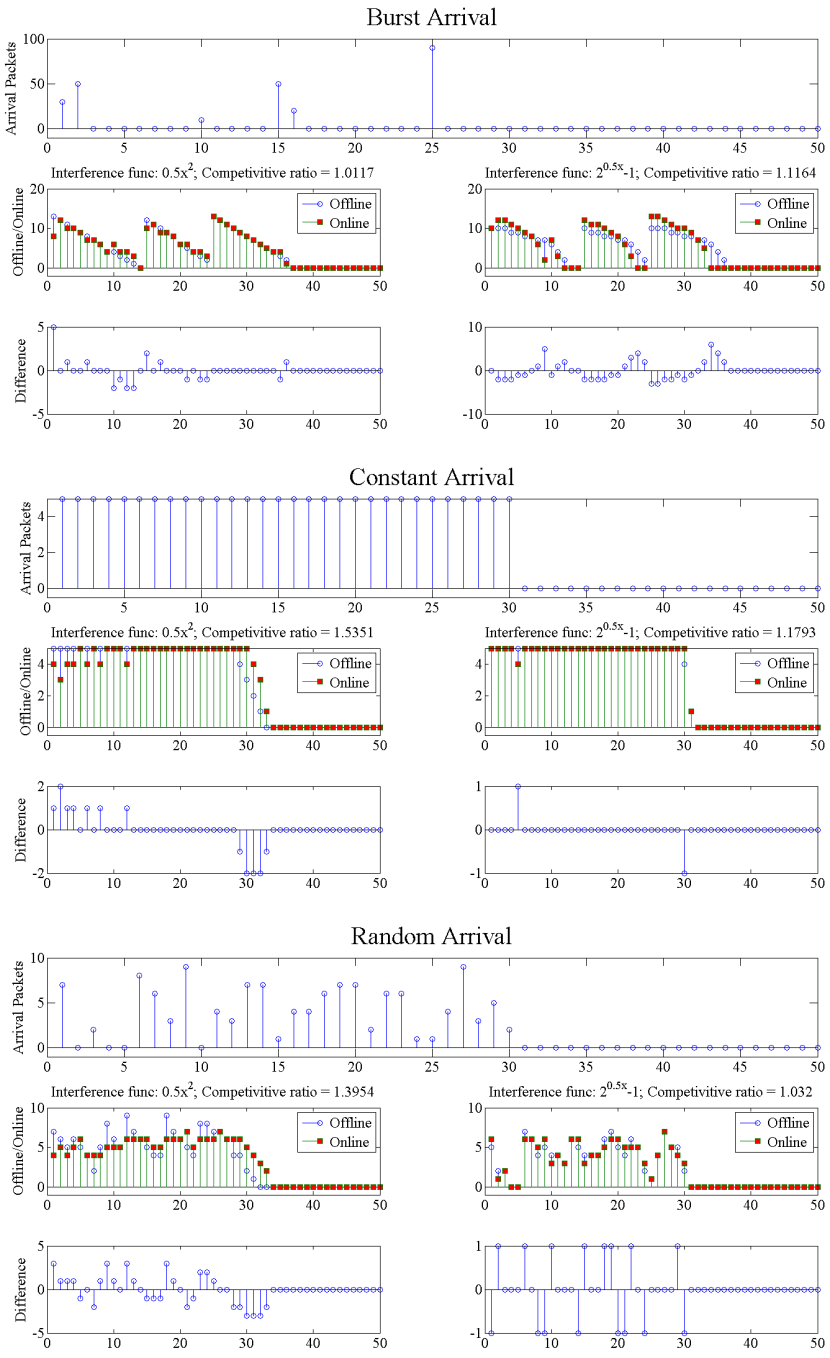

Figure 4: Simulation Results

[13] L.L.H. Andrew, A. Wierman, A. Tang, "Optimal speed scaling under arbitrary power functions", *SIGMETRICS Perform. Eval. Rev.,* 37(2), 39–41, 2009.

[14] A. Wierman, L. L. H. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. INFOCOM*, 2009, pp. 2007–2015

[15] N. Bansal, H.-L. Chan, and K. Pruhs. "Speed scaling with an arbitrary power function", in *SODA '09: Proc. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2009.

[16] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, San Jose, CA, 7-11 Jun 2011, pp. 233–244.

[17] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *Proc. INFOCOM*, April, 2011.

[18] M. Lin, Z.Liu, A.Wierman and L.L.H. Andrew, "Online algorithms for geographical load balancing," in *Proceedings of IEEE IGCC*, 2012.