Scheduling Weighted Packets With Deadlines Over A Fading Channel*

Fei Li   Zhi Zhang
Department of Computer Science
George Mason University
Fairfax, VA 22030
{lifei, zzhang8}@cs.gmu.edu

Abstract

We consider scheduling weighted packets with time constraints over a fading channel. Packets arrive at the transmitter in an online manner. Each packet has a value and a hard deadline by which it should be sent. The fade state of the channel determines the throughput obtained per unit of time and the channel's quality may change over time. In this paper, we design both offline and online algorithms to maximize weighted throughput, which is defined as the total value of the packets sent by their respective deadlines. We first present polynomial-time exact offline algorithms for this problem. Then, we present online algorithms and their competitive analysis as well as the lower bounds of competitive ratios. Our work is the first one addressing weighted throughput for this important problem in the areas of information theory and real-time communications.

1 Introduction

Time-varying signal strength is a fundamental characteristic of wireless channels. Scheduling packets over fading wireless channels has received much attention (see [20, 11, 10, 19, 22, 5] and the references therein). A scheduling algorithm can significantly improve the communication performance by taking advantages of the changing channel states. Specifically, the packets to be scheduled are associated with deadlines. Time constraints (deadlines) are specified on packets to model the possible network protocol timeouts and the time sensitivity of the information carried by the packets. In the previously studies, the objective is usually to maximize the total number of packets delivered by their deadlines. However, for many practical problems, it is more reasonable to differentiate various packets and take into account the amount and/or the significance level of the information associated with the packets. Thus, in this paper, we address the problem of optimizing *weighted* throughput of packets with time constraints in a fading wireless channel. Our results show that the algorithmic solutions in maximizing weighted throughput as well as their computational complexity are significantly different from those optimizing throughput of uniform-value packets.

Resource allocation for fading channels has been a well-studied topic in the area of information theory. The quantity to maximize is often the Shannon capacity, which is defined as the tightest upper bound of the amount of information (i.e., the total number of packets) that can be reliably transmitted over a communication channel. Tse and Hanly [20] have found capacity limits and optimal resource allocation policies for such fading channels. They also studied the greedy approach for channel allocations in multi-access fading channels, assuming all packets arriving at the transmitter are successfully delivered. Prabhakar et al. [11] have considered proactively adjusting the rate of packet transmission for saving energy where the quality of the fading channel is assumed to be fixed and the consumed energy is a convex function of the transmission speed. The discrete version of this algorithm has been proposed in [21] in a more general problem setting. In [10], the authors applied a dynamic programming approach to get the optimal solution for scheduling uniform-value packets under both time and energy constraints. However, this algorithm [10] runs in exponential-time in overloaded systems. A polynomial-time optimal offline solution of scheduling packets with hard deadlines was given in [19, 22]. In their problem settings, energy is minimized under the assumption that all arriving packets are successfully delivered. An optimal offline algorithm maximizing throughput and a heuristic online approach of scheduling uniform-value packets with possibly different deadlines were given in [5]. No theoretical analysis has been provided for their heuristic online solution. Note that in these previous studies, packets have uniform values and their arrivals at the transmitter are usually modeled by a Poisson distribution.

*Research is partially supported by Seed Award from the Office of the Vice President for Research and Economic Development at George Mason University.

In the existing models discussed above, packets are distinguished by their deadlines and release dates only (they have uniform values and sizes). However, packets from different users and various applications may have different significance levels of embedded information. For the sake of being realistic and practical, we associate packets with *weights* (*values*) that indicate the significance of their embedded information. We also associate packets with deadlines to represent the information's time sensitivity in real-time applications. None of the previous algorithms for delivering packets can be generalized to this problem setting, because a schedule with the maximum throughput does not imply its optimality on maximizing weighted throughput. In this paper, we design efficient scheduling algorithms to maximize weighted throughput for packets with time constraints over a fading channel. Our contributions include:

1. Offline algorithms for this model (Section 3.1).
2. Competitive online algorithms and lower bounds of competitive ratios for this model and its variants (Section 3.2).

2 Model

We consider scheduling weighted packets with deadlines over a wireless fading channel. In this model, time is assumed to be discrete. Each unit of time is called a *time step* and a few continuous time steps are called a *time interval*. Packets are released over time. All packets are with the same length $l \in \mathbb{R}^+$ (l is a constant). Each packet p has an integer *release time* (*arriving time*) $r_p \in \mathbb{Z}^+$, a positive real value $w_p \in \mathbb{R}^+$ to represent its *weight* (*value*), and an integer hard deadline $d_p \in \mathbb{Z}^+$ to denote the time by which it should be delivered. The time required to send a packet depends on the *state quality* q_t ($q_t \in [q_{\min},\ q_{\max}]$) of the fading channel during a time step t. Without loss of generality, we assume $l = 1$, $q_{\min} > 0$, $q_{\max} = 1$, and the fade state in a single time step keeps unchanged. If the fading channel is at its highest quality $q_{\max}$, one packet can be sent in a time step. A packet has to be sent in consecutive time steps. Sending a packet p takes $t(p)$ steps where $t(p) = t_2 - t_1$ subject to $\sum_{t=t_1}^{t_2} q_t \geq l, t_1, t_2 \in \mathbb{Z}^+$. Two or more packets cannot share (i.e., to be sent in) the same time step. If a packet p is sent by its deadline d_p, its weight w_p is contributed to our objective. Our goal is to maximize weighted throughput subject to the deadline constraints of packets and the varying fading channel qualities.

We design two kinds of algorithms: *offline algorithms* and *online algorithms*. All input information (including the fade channel states, the packets' characteristics, and the packet sequence) is known to an offline algorithm in advance. For an online algorithm, the packet input sequence is unknown and each packet's characteristics is known to the algorithm only at the time when it actually arrives at the transmitter. The fade state of the channel is unknown or partially known to the online algorithm; which depends on the assumptions in the variants of the online version of this problem. Note that essentially, delivering packets with deadlines in a wireless channel is an online decision making problem. We address the online version under two settings:

- In the *non-preemption setting*, a packet, once it is being delivered, is committed to send without being preempted until it is finished.
- In the *preemption-restart setting*, an online algorithm is allowed to abort a packet during its transmission, and the aborted packet can be restarted (from scratch) and completed later.

In either setting, the algorithm gets credit only for packets that are executed continuously from the beginning to the end by its deadline. Our model can be an *overloaded system* — it is feasible that due to packets' deadline constraints, no algorithm can deliver all packets in the input instance. Note that in an *underloaded system*, the solution is relatively trivial. The classic algorithm `EDF` (Earliest-Deadline-First) delivers all packets non-preemptively by their deadlines and it is optimal in both offline and online settings.

We have realized the connection between this problem and the *bounded-delay model* in buffer management. The bounded-delay model [16, 14, 17, 9, 18] implicitly applies an assumption of idealized channel quality all the time such that in every time step, one packet can be delivered. The offline version of the bounded-delay model has been solved optimally via maximizing a weighted bipartite matching. The online version still remains a very intriguing open problem.

3 Algorithms and Analysis

We classify our algorithms and present them as offline algorithms and online algorithms in Section 3.1 and Section 3.2 respectively. Note that in designing offline algorithms, there is no difference between these two settings, non-preemption and preemption-restart.

Let the input sequence be $\mathcal{I}$ and $|\mathcal{I}| = n$. All packets have the same length l.

3.1 Offline algorithms

In this section, we present a few exact algorithms running in polynomial time for several variants of the problem, assuming all input information is known.

Theorem 1 *[12] Assume the fading channel has a fixed quality $q \in [0,\ 1]$ during all time steps. If all packets* *are with the same value (but they can have arbitrary deadlines), then there exists an exact polynomial-time optimal algorithm running in time $O(n \log n)$.*

We consider an important variant in which packets are with *agreeable deadlines*, i.e., for any two packets p_i and p_j, $r_{p_i} < r_{p_j}$ implies $d_{p_i} \leq d_{p_j}$. This variant allows an optimal algorithm running in an online manner. Here, we look into the Earliest-Deadline-First (EDF) algorithm: If there is no packet being sent, schedule the earliest-deadline pending packet until it is finished. EDF is one of the most studied policies in the area of real-time scheduling. We have

Theorem 2 *Assume the fading channel has a fixed quality $q \in (0,\ l]$ during all time steps. If all packets are with the same value and if they are with agreeable deadlines, then* EDF *is an exact polynomial-time optimal algorithm running in linear time $O(n)$.*

Proof To prove Theorem 2, it is sufficient to show that at any time t (t does not have to be an integer), EDF finishes no fewer packets than any algorithm ALG. We use $\mathtt{A}(\mathcal{I})$ to denote the number of packets delivered by their deadlines in the algorithm A.

The proof consists of proving the following two parts:

1. Given any algorithm ALG and the set of packets $\mathcal{I}'$ ($\subseteq \mathcal{I}$) that ALG schedules, we can create an earliest-deadline-first scheduler $\mathtt{EDF}'$ finishing all packets in $\mathcal{I}'$ by their deadlines; that is,

$$\mathtt{EDF}'(\mathcal{I}') = |\mathcal{I}'| = \mathtt{ALG}(\mathcal{I}). \tag{1}$$

2. Given the input $\mathcal{I}$ for EDF and the input $\mathcal{I}'$ for $\mathtt{EDF}'$, EDF is no worse than $\mathtt{EDF}'$ in finishing as many as packets by their deadlines at any time t; that is,

$$\mathtt{EDF}(\mathcal{I}) \geq \mathtt{EDF}'(\mathcal{I}'). \tag{2}$$

Equation 1 and Equation 2 imply $\mathtt{EDF}(\mathcal{I}) \geq \mathtt{ALG}(\mathcal{I})$.

Given the set of packets $\mathcal{I}'$ that is finished by an algorithm ALG as the input of $\mathtt{EDF}'$, we can use the *exchange argument* to show that $\mathtt{EDF}'$ can finish the packets in $\mathcal{I}'$. Note that if the fading channel is at a fixed quality, for any packet p with length l, it takes $\lceil l/q \rceil$ time steps to deliver p. Since all packets are with the same value and processing time, we can always replace the packets $\in (\mathcal{I}' \setminus \mathcal{I})$ using packets $\in (\mathcal{I} \setminus \mathcal{I}')$ with no later release dates or deadlines. Thus, the second part of the proof is true as well.

The running time analysis is as follows. If packets are with agreeable deadlines, newly arriving packets can be appended at the end of the packet queue. EDF sends the first pending packet which has not expired yet in the next $\lceil l/q \rceil$ time steps when there is no packet currently being sent. The scheduling algorithm runs in linear time $O(n)$. □

In the following, we can prove that there exists an optimal offline policy for this problem. First, we assume that the channel quality's is a fixed constant number. Then, we apply the algorithm into the general setting in which the fade states of the channel vary.

Theorem 3 *Assume the fading channel has a fixed and constant quality $q \in [0,\ 1]$ during all time steps. There exists an optimal algorithm in maximizing weighted throughput.*

Before we proceed to the proof of Theorem 3, we would like to point out that since it may not be feasible to deliver all packets ever arrive at the transmitter in an overloaded system, the optimal solutions in the previously studied models in [10, 5, 15] cannot be directly applied in our model. Instead, we design an exact algorithm that depends on the following two critical observations on the matroidal structure of the model:

Remark 1 *Given a set S of packets, any feasible schedule on S can be converted to an earliest-deadline-first schedule where* the earliest-deadline packet $\in S$ is scheduled as long as it is available for the transmitter.

Remark 2 *Denote S^* as both the optimal solution maximizing the weighted throughput and the set of packets delivered. If a packet $p_j \in S^*$ is pending at time t and it is not scheduled at time t, there must exist a packet $p_i \in S^*$ such that $r_{p_i} \leq t + \lceil l/q \rceil$ and p_i is scheduled at time r_{p_i}.*

Proof Let the set of packets arriving at the transmitter be $\{p_1, p_2, \ldots, p_n\}$. It takes $\lceil l/q \rceil$ continuous time steps to deliver one packet. The set of time steps that a packet can be sent is a subset of all the time steps T

$$T := \bigcup_i [r_{p_i},\ r_{p_i} + n \cdot \lceil \frac{l}{q} \rceil], \tag{3}$$

where q is the constant channel quality. Let the time steps in T be $t_1, t_2, \ldots, t_m$, where $|T| \leq n \cdot n \cdot \lceil l/q \rceil \leq n^2 \cdot l/q + n^2$.

We have a greedy algorithm as follows. Based on Remark 1 and Remark 2, we know that if there are two pending packets available for delivery, we can always pick the one with the earlier deadline to send in a time step $\in T$. We call this order a *canonical order*. Our following algorithm is based on the matroidal property of the model.

The generated schedule in P' is the optimal solution and its correctness is based on the fact that feasible schedules form a matroid. The running time of this algorithm is $O(n \cdot \log n + n \cdot \log n \cdot |T|) = O(l \cdot n^3 \cdot \log n/q)$, where the factor $O(n \cdot \log n)$ for $|T|$ is the time spent on sorting packets in $\mathcal{P}'$ in decreasing order of weights. For each packet p, it takes time $O(|T|)$ to verify the feasibility of adding p into the existing schedule. For this variant, our result improves the algorithm in [3], whose running time is $O(n^{10})$ and which also holds when q is fixed but not a constant number. □

Algorithm 1 `Offline-Optimal`

Initialize the set of packets to be sent $P' = \emptyset$. Initialize the set of packets to be considered $P = \mathcal{I}$ $(= \{p_1, p_2, \ldots, p_n\})$.
Sort all packets in $\mathcal{P}$ in decreasing order of values.
while $|P'| \leq n$ and there are packets left in P **do**
  Remove the maximum-value packet p from P.
  if the set $P' \cup \{p\}$ can be feasibly scheduled in T under the canonical order (i.e., all packets can be sent by their deadlines) **then**
    Insert the packets in P' and update P' as $P' \cup \{p\}$.
  end if
end while
return $\mathcal{P}'$.

Following the proof of Theorem 3, we immediately have

Corollary 1 *Consider scheduling weighted packets with deadlines in a fading channel. There exists an optimal algorithm in maximizing weighted throughput in time $O(n \cdot \log n \cdot m)$, where m is the number of time steps we consider.*

In our model, as long as each interval with time steps $[t_1,\ t_2]$ has $\sum_{t=t_1}^{t_2} q_t \geq l$, a packet can be sent. For each release time r_p, we seek the following n consecutive time intervals such that for each time interval $[t_s,\ t_e]$, $\sum_{t=t_s}^{t_e} q_t \geq l$. Let the union of all such time steps be T'. Then, the number m in Corollary 1 has $m = |T'|$.

Note that our proofs depend on the following three assumptions that (1.) all packets have the uniform length, (2.) packets are sent continuously, and (3.) packets do not share a time step. If any one of these assumptions does not hold, it is easy to conclude that the offline version of this problem is a NP-complete one, via the reduction from the NP-complete Bin-Packing problem or the NP-complete Set-Partition problem. For example,

Theorem 4 *Consider packet scheduling in fading channels. Assume a packet can be preempted before the transmitter finishes it. Only unfinished part of the packet is resumed later. Then, maximizing (weighted) throughput is a NP-complete problem, even if all packets share a common release date and a common deadline.*

Proof To prove the NP-hardness, it is sufficient to show that we can reduce a well-known NP-complete problem to our problem in polynomial time and a candidate solution can be verified in polynomial time. Verifying a candidate solution can be done in linear time. To prove Theorem 4, the remaining work is to reduce the *Set-Partition problem* to our problem.

The *Set-Partition problem* is defined as follows. Given an instance that has a finite set $\mathcal{I}$ and a size $s_i \in \mathbb{Z}^+$ for $i \in \mathcal{I}$, the objective is to find out if there exists a subset $\mathcal{I}' \subseteq \mathcal{I}$ such that $\sum_{i \in \mathcal{I}'} s_i = \sum_{i \in \mathcal{I} \setminus \mathcal{I}'} s_i$. This problem is NP-complete [13].

Now we introduce our NP reduction. Given any instance $\mathcal{I}$ of the Set-Partition problem, we normalize $\mathcal{I}$ such that $\sum_{i \in \mathcal{I}} s_i = 2$. Then we generate the channel quality $q_i = s_i$ for each $i \in \mathcal{I}$ and we have two packets whose deadlines are $\sum_{i \in \mathcal{I}} s_i = 2$. This conversion takes polynomial time. Consider any algorithm `ALG`. If `ALG` returns a throughput of 2, `ALG` returns two sets of fading states such that each of them is with a total quality $\sum_j q_j = 1$. The time step of delivering one packet (respectively, the other packet) consists of one partition set (respectively, the other partition set) for the Set-Partition problem. Since Set-Partition problem is NP-complete, `ALG` cannot schedule two packets by their deadlines optimally in polynomial-time. Hence, maximizing (weighted) throughput with time varying quality, is NP-complete. □

3.2 Online algorithms

Scheduling packets with deadlines (even in a fading channel whose quality is at its maximum) is essentially an online decision problem. In order to evaluate the worst-case performance of an online algorithm lacking of future input information, we compare it with an optimal offline algorithm. The offline algorithm is a clairvoyant algorithm, empowered to know the whole input sequence (including the fading states of the channel, the packet sequence, and all packets' characteristics) in advance to make its decision. *In contrast to stochastic algorithms that provide statistical guarantees under some mild assumptions on input sequences, competitive online algorithms guarantee the worst-case performance.*

Definition Competitive ratio [4]. A deterministic (randomized) online algorithm $\mathtt{ON}_d$ ($\mathtt{ON}_r$) is called *k-competitive* if its (expected) weighted throughput on *any* instance is at least $1/k$ of the weighted throughput of an optimal offline algorithm on this instance:

$$k := \max_{\mathcal{I}} \frac{\mathtt{OPT}(\mathcal{I}) - \delta}{\mathtt{ON}_d(\mathcal{I})}, \qquad \mathtt{ON}_d \text{ is a deterministic algorithm}$$

$$k := \max_{\mathcal{I}} \frac{\mathtt{OPT}(\mathcal{I}) - \delta}{\mathbf{E}[\mathtt{ON}_r(\mathcal{I})]}, \qquad \mathtt{ON}_r \text{ is a randomized algorithm}$$

where δ is a constant, $\mathtt{OPT}(\mathcal{I})$ is the optimal offline solution of an input $\mathcal{I}$, and r is the set of random variables flipped by a randomized online algorithm $\mathtt{ON}_r$. The parameter k is known as the online algorithm's *competitive ratio* [1].

[1]In real-time scheduling terminologies, $1/k$, the reciprocal of the competitive ratio, is called *competitive factor*.

The *upper bounds* of competitive ratios are achieved by some known online algorithms. A competitive ratio less than the *lower bound* is not reachable by any online algorithm. An online algorithm is said to be *optimal* if its competitive ratio reaches the lower bound. If the additive constant δ is no larger than 0, the online algorithm ON is called *strictly k-competitive*. Note that a randomized algorithm does not depend on any assumptions on the input sequence and the randomness r is internal to the algorithm. Competitiveness has been widely accepted as the metric to measure an online algorithm's worst-case performance in theoretical computer science and operations research [4]. In this section, we design and analyze some competitive online scheduling algorithms for maximizing weighted throughput in a fading channel.

We first investigate the challenge of designing efficient online algorithms for this problem. Without time constraints on packets, (weighted) throughput is maximized by simply delivering all packets that ever arrive at the transmitter. When time constraints are enforced on *uniform-value* packets, the objective of this problem becomes to send as many packets as possible before their respective deadlines — this variant is the same problem of scheduling equal-length jobs [8]. A 2-competitive deterministic algorithm and a 1.5-competitive deterministic algorithm have been given for this variant in the non-preemption setting and the preemption-restart setting respectively [8]. Both online algorithms' competitive ratios are tight.

Though optimal competitive online algorithms have been proposed in [8] for a variant in which throughput (of uniform-value packets) is maximized, scheduling packets with deadlines is open and becomes more interesting and complicated when packet weights are considered. Now we present an instance in which the fade state of the channel is ideal (i.e., $q_t = q_{\max} = l, \forall t$) but packets have weights. Consider two packets p_1 and p_2 with $d_{p_1} = 1 < d_{p_2} = 2$ and $w_{p_1} < w_{p_2}$ at time 1 in an overloaded system. Note that the transmitter has no knowledge of future arriving packets. Sending the packet p_1 in the first time step may cause p_2 not to be sent anymore if we assume that another packet p_3 with $d_{p_3} = 2$ and $w_{p_3} > w_{p_2}$ arrives at time 2 (since p_2 and p_3 cannot be sent simultaneously in step 2 by their deadlines). A better (clairvoyant) way is to send p_2 in the first time step and send p_3 in the second time step. One the other hand, if the online algorithm picks p_2 to send in the first time step, it potentially leads to the expiration of the packet p_1. In case p_3 is not released in step 2 in the actual input sequence, the online algorithm loses the value of p_1 — it is better to send p_1 and p_2 in the first two consecutive time steps clairvoyantly. In summary, even under ideal fade states, the challenge of designing efficient online algorithms who are lacking of information about future input is to balance wisely between sending an earliest-deadline packet and a largest-weight packet. Our proposed online algorithms are based on this intuition. Another challenge of this model is due to the uncertainty of the fade states of the wireless channel. We will address more on these challenges and our solutions in the following.

We consider non-preemption and preemption-restart settings separately. We also call the optimal offline algorithm *adversary*. Let $w_{\max}$ and $w_{\min}$ denote the maximum and the minimum value of a packet in the input sequence $\mathcal{I}$ respectively.

3.2.1 In the non-preemption setting

We first show a negative result and then show an optimal online algorithm for a variant of this model.

Theorem 5 *In the non-preemption setting, no online algorithm has a constant competitive ratio, even if the fade state is a fixed number q ($q < q_{\max} = l$) and if packets are with agreeable deadlines. The lower bound of competitive ratios can be up to $w_{\max}/w_{\min}$.*

Note that if packets are with the uniform value and the if the fading channel has a fixed quality (but packets can have arbitrary deadlines), EDF is 2-competitive [8]. Thus, associating values to packets complicates the model.

Proof We set the channel's quality $q = 0.5$. Any packet can be sent in consecutive 2 time steps. Let an online algorithm be ON. We use $(w,\ d)$ to denote a packet with value w and deadline d.

In the first time step, a packet $(w_{\min},\ 2)$ is released. The adversary keeps releasing a packet $(w_{\min},\ 2 \cdot i)$ in each time step $2 \cdot i$ until one of the events happens: (1.) ON picks up a packet $(w_{\min},\ 2 \cdot k)$ to send, or (2.) the adversary has released a such packets with value $w_{\min}$, and ON has not picked up any one of them to send.

For the second case, the adversary stops releasing new packets and it schedules all packets ever released with a total gain of $a \cdot w_{\min}$. On the other side, ON gains nothing overall. For the first case, when ON picks up a packet $(w_{\min},\ 2 \cdot k)$ to send, the adversary releases a packet $(w_{\max},\ 2 \cdot k + 3)$ at time $2 \cdot k + 1$. Note that in the non-preemption setting, ON cannot stop sending the packet $(w_{\min}, 2 \cdot k)$ till the time $2 \cdot k + 2$ when this packet is finished. Thus, ON cannot execute the packet $(w_{\max},\ 2 \cdot k + 3)$ at time $2 \cdot k + 1$ to get it finished by its deadline. After releasing the packet $(w_{\max},\ 2 \cdot k + 3)$, the adversary releases nothing. Overall, the optimal offline algorithm will send all packets $(w_{\min},\ 2 \cdot 1), (w_{\min},\ 2 \cdot 2), \ldots, (w_{\min},\ 2 \cdot (k-1))$ and $(w_{\max},\ 2 \cdot k + 3)$. On the other side, ON executes only one packet $(w_{\min},\ 2 \cdot k)$. The competitive ratio is

$$\frac{(k-1) \cdot w_{\min} + w_{\max}}{w_{\min}} = k - 1 + \frac{w_{\max}}{w_{\min}} \geq \frac{w_{\max}}{w_{\min}}.$$

Then, ON is no better than $(w_{\max}/w_{\min})$-competitive. □

To complement Theorem 5, we note

Theorem 6 *[1, 2] In the non-preemption setting, no online algorithm has a constant competitive ratio, if the fade state is ideal ($q = q_{\max} = l$). The lower bound of competitive ratios can be up to $\sqrt{w_{\max}/w_{\min}}$.*

Given the assumptions that the channel state is a fixed number and packets are with agreeable deadlines, we have proved that for any time t EDF finishes no fewer packets than any algorithm (see the proof of Theorem 2). Given an input $\mathcal{I}$, we assume EDF finishes s packets with a total value $W \geq s \cdot w_{\min}$. Any algorithm finishes no more than s packets with a total value $\leq s \cdot w_{\max} \leq W \cdot (w_{\max}/w_{\min})$. Thus, we immediately have

Corollary 2 *In the non-preemption setting, if the fade state is a fixed number and if packets are with agreeable deadlines,* EDF *is an optimal online algorithm.*

If the fade state is at its maximum all the time (such that a packet is sent in a single time step), this variant of the online problem is same as the bounded-delay model [16, 14, 17, 9, 18]. An optimal online algorithm has been proposed for the agreeable deadline case [17]. For the general case, the best known lower bound of competitive ratios is $\phi := (1 + \sqrt{5})/2 \approx 1.618$ [14] and the best known upper bound is 1.832 [9, 18]. Closing the gap $[1.618,\ 1.832]$ is still an intriguing open problem [7].

3.2.2 In the preemption-restart setting

In the preemption-restart setting, we first provide a bad example to show that if the fading states are unknown to the online algorithms, no online algorithm can have a competitive ratio better than $w_{\max}/w_{\min}$.

Theorem 7 *If the fading states are unknown to online algorithms, no online algorithm can have a competitive ratio better than $w_{\max}/w_{\min}$.*

Proof Consider time 0 and two packets are released. We use $(w,\ d)$ to represent a packet p with value w and deadline d. Let an online algorithm be ON. The fading state at time 0 is 0.5. A packet $p_1 := (w_{\min},\ 2)$ is released at time 0.

The fade state keeps its quality 0.5 since time 0 to time 2. At time 1, a packet $p_2 := (w_{\max},\ 3)$ is released. If ON schedules p_1, we keep the fading state at 0.5 till time 3 and ON cannot finish p_2 by its deadline. The optimal offline algorithm will schedule p_2 instead and the competitive ratio is $w_{\max}/w_{\min}$. On the other hand, if ON schedules p_2 at its arrival, the fade state shapely changes to 0 at the end of time 2 and keeps 0 eventually. Thus, even ON starts to schedule p_2, it cannot finish it thought. Instead, the optimal offline algorithm schedules p_1 and the competitive ratio is $w_{\min}/0$, which is arbitrarily large. □

Based on Theorem 7, we know that if the fade states are unpredictable, without one step of look-ahead, no online algorithm can have a competitive ratio better than $w_{\max}/w_{\min}$. Again, EDF is optimal in this setting. In the following, we consider a practical scenario and make the following assumption that is well-known:

Assumption 1 *[20, 19, 22] The online algorithms have the ability of looking one-step ahead of knowing the fade states of the wireless channel. At the time when an online algorithm starts to schedule a packet, this "committed" packet can be scheduled based on the future fading states. However, note that the online algorithm is allowed to preempt-restart this packet later.*

Assumption applies to all the variants we consider in the following.

In [8], an optimal 1.5-competitive deterministic algorithm has been proposed for a variant in which the fade state is a fixed number (the lower bound of competitive ratios for that variant is 1.5). We note the lower bound can be improved to ϕ for the weighted version of this problem.

Theorem 8 *[6] Assume the channel's quality is fixed at $q_{\max} = l$. The lower-bound of competitive ratios for this variant is $\phi := (1 + \sqrt{5})/2 \approx 1.618$. This lower bound holds even for agreeable deadline instances.*

Theorem 9 *[8] Assume the channel's quality is fixed $< l$. The lower-bound of competitive ratios for deterministic online algorithms is 2. This lower bound holds even for maximizing the number of packets sent by their deadlines.*

From Theorem 9, we know that the variant (in which the fade state is a constant) has the lower bound of 2. For this invariant (we also called it a bounded-delay model), given a set of pending packets S, an online algorithm can calculate the *optimal provisional schedule* S^* (S^* is the one that achieves the maximum total value of packets among all provisional schedules on pending packets S) and send one packet from S^*. Note that S^* can be calculated only if the channel's quality is known beforehand. Since the fade state of the channel is unpredictable, all prior online algorithms on the bounded-delay model cannot be applied in our model.

Assume the fade states and future input information are unknown. Here, we study an algorithm called SEMI-GREEDY. In each time step, the maximum-value pending packet p aborts the currently running packets i, if $w_p \geq \alpha \cdot w_i$.

Algorithm 2 SEMI-GREEDY($\alpha > 1$)

1: Let the maximum-value pending packet with the earliest deadline be p and let the currently being sent packet be i. If p (or i) does not exist, we set $w_p = 0$ (or $w_i = 0$).
2: **if** $w_p \geq \alpha \cdot w_i$ **then**
3: Abort i and send p.
4: **end if**

Before we prove the competitive ratio for the algorithm SEMI-GREEDY, we define a concept that is useful to the proof.

Definition Packet chains. We define a packet chain C of k packets as

$$C := \{p_1,\ p_2,\ p_3,\ \ldots,\ p_k\},$$

with the following property ($\alpha > 1$),

$$w_{p_i} \ \leq\ w_{p_{i+1}} \,/\, \alpha, \quad \forall\, i \,=\, 2,\, 3,\, \ldots,\, k-1.$$

We use $W(C)$ to represent the total value of the packets of C.

Lemma 1 *Given a chain C of $k \geq 2$ packets $p_1,\ p_2,\ \ldots, p_k$, we have*

$$W(C) \ \leq\ (\frac{1}{\alpha - 1} \cdot (\alpha^{n+1} - 1)/\alpha^n) \cdot w_{p_k}. \qquad (4)$$

Proof

$$\begin{aligned}
\frac{W(C)}{w_{p_k}} &= \frac{\sum_{i=1}^{k} w_{p_i}}{w_{p_k}} = \frac{w_{p_1} + w_{p_2} + \cdots + w_{p_{k-1}} + w_{p_k}}{w_{p_k}} \\
&= \frac{w_{p_1} + w_{p_2} + \cdots + w_{p_{k-1}} + \alpha \cdot w_{p_{k-1}} + \epsilon}{\alpha \cdot w_{p_{k-1}} + \epsilon} \\
&= 1 + \frac{1}{\alpha} \cdot \frac{w_{p_1} + w_{p_2} + \cdots + w_{p_{k-1}}}{w_{p_{k-1}}} \\
&\leq 1 + \frac{1}{\alpha} \cdot (1 + \frac{1}{\alpha} \cdot \frac{w_{p_1} + w_{p_2} + \cdots + w_{p_{k-2}}}{w_{p_{k-2}}}) \\
&= 1 + \frac{1}{\alpha} + \frac{1}{\alpha^2} \cdot \frac{w_{p_1} + w_{p_2} + \cdots + w_{p_{k-2}}}{w_{p_{k-2}}} \\
&\leq \ldots \\
&\leq 1 + \frac{1}{\alpha} + \frac{1}{\alpha^2} + \cdots + \frac{1}{\alpha^{k-2}} + \frac{1}{\alpha^{k-1}} + \frac{1}{\alpha^k} \\
&= \frac{1}{\alpha - 1} \cdot (\alpha^{k+1} - 1)/\alpha^k
\end{aligned}$$

□

Theorem 10 *The* SEMI-GREEDY *algorithm has a competitive ratio* $\max\{1 + \alpha,\ \frac{1}{\alpha-1} \cdot (\alpha^{n+1} - 1)/\alpha^n)\}$. *It is ($\phi^2 \approx 2.618$)-competitive when* $\alpha = \phi \approx 1.618$.

Proof We use a charging scheme to prove Theorem 10. Let the subset of packets chosen by the adversary (= an optimal offline algorithm) (respectively, SEMI-GREEDY) be Π_1 (respectively, Π_2). Without loss of generality, we assume the adversary sends packets in a canonical order, i.e., for any two pending packets p_i and p_j, the adversary sends the packet with an earlier deadline. We are going to prove that

$$\frac{\sum_{p_j \in \Pi_1} w_{p_j}}{\sum_{p_i \in \Pi_2} w_{p_i}} \leq \max\{1 + \alpha,\ \frac{1}{\alpha - 1} \cdot (\alpha^{n+1} - 1)/\alpha^n)\}.$$

The proof depends on the following two observations:

1. Given a set of packets S at time t, we assume an online algorithm schedules a packet p_i. We consider time $t' > t$. Since all packets are with the same length, if the packet p_i cannot be finished by time t', any packet in S cannot be finished completely by time t', no matter what the fade state of the channel is.

2. Given a set of packets S at time t, the SEMI-GREEDY algorithm schedules a packet p_i. We have $w_{p_i} \geq \max_{p_j \in S} w_{p_j}/\alpha$. We assume p_i is aborted at time $t' > t$ by a packet p_k, we have $w_{p_i} < w_{p_k}/\alpha$ and $p_k \notin S$. If the preempting packet p_k is not sent by the algorithm SEMI-GREEDY, p_k must be aborted by another packet which has the potential of being sent. So on and so forth, we regard all aborted packets and the last-sent packet p_l as a chain. From Lemma 1, all ever-aborted packets have value $\leq w_{p_l} \cdot \frac{1}{\alpha-1} \cdot (\alpha^{n+1} - 1)/\alpha^n$. Note that no chains share a same packet.

For any packet $p \in (\Pi_1 \setminus \Pi_2)$ sent by the optimal offline algorithm, either p expires before SEMI-GREEDY sends it or p is sent, aborted before it is finished, and is never completed by its deadline. If p expires, any packet that SEMI-GREEDY sends since time r_p has a value $\geq w_p/\alpha$ (from the algorithm).

We examine the time intervals (a single packet is sent in such an interval) for the optimal offline algorithm and this online SEMI-GREEDY algorithm in a sequential order. Our charging scheme works as follows:

1. For any packet $p \in (\Pi_1 \setminus \Pi_2)$ that SEMI-GREEDY has not ever run, we charge it to the corresponding time interval that SEMI-GREEDY sends a packet. We note that SEMI-GREEDY must have one pending packet to send in this time step since this packet p is a candidate. The packet SEMI-GREEDY sends, let it be p', in this corresponding time interval has a value no less than w_p/α. Also, SEMI-GREEDY finishes p' no later than the adversary finishes p since p and p' have the same processing time and p and p' are being executed in corresponding time steps when both algorithms send packets.

2. For any packet $p \in (\Pi_1 \setminus \Pi_2)$ that SEMI-GREEDY ever sends but aborts it later, we know that (from above observations) that p belongs uniquely to a chain and the last element of this chain, say p', is sent by SEMI-GREEDY. Thus, we charge w_p to the time interval when p' is sent by SEMI-GREEDY.

3. For any packet $p \in (\Pi_1 \cap \Pi_2)$, we charge w_p to the time interval when SEMI-GREEDY sends p. Clearly, for any packet acting as the last-element of a chain, this charging scheme results that the value ratio is bounded by $\frac{1}{\alpha-1} \cdot (\alpha^{n+1}-1)/\alpha^n$ (see Lemma 1).

The remaining part of the proof is to argue that when we charge a packet $p \in (\Pi_1 \setminus \Pi_2)$ that SEMI-GREEDY has not ever run yet, in the corresponding time interval, SEMI-GREEDY sends a packet p', $w_{p'} \geq w_p/\alpha$. This claim is easy to prove since SEMI-GREEDY chooses the earliest-deadline-first qualified packet to send. If $w_{p'} < w_p/\alpha$, p' will be aborted by p immediately at the time when p arrives. Thus, for each packet p that SEMI-GREEDY sends, the charged value to p for the adversary is bounded by $1+\alpha$ and $\frac{1}{\alpha-1} \cdot (\alpha^{n+1}-1)/\alpha^n$ and all packet that the adversary sends have been charged. Theorem 10 is proved. □

Closing or shrinking the gap $[2,\ 2.618]$ is still an open problem.

Assume the fade states are known to the online algorithms, but the packet input sequence are unknown. We note at first that given the channel quality at its maximum, delivering uniform-value packets in a greedy manner (which runs in an online manner) achieves the best throughput for any algorithms. However, if the channel quality is less than $q_{\max}$, the lower bound of competitive ratios for any deterministic online algorithms is 2 [8]. For this, we conclude that a c-competitive algorithm for the variant with channel quality $q_{\max}$ consistently does not imply a c-competitive algorithm for the variant in which the fade states are known to the online algorithms. The latter variant has its own interests and difficulties.

Now we present an instance in which the fade state of the channel is with $q_t = l/2, w_{p_i} = 1, \forall t, i)$ to illustrate the challenge. Consider one packet p_1 with deadline 5 at time 1. If an online algorithm executes it, the adversary releases another packet p_2 with deadline 3 at time 2. So, the online algorithm cannot finish both jobs and the competitive ratio is 2, given the adversary finishes both in order of packets p_2 and p_1. If the online algorithm aborts p_1 but executes p_2, the adversary releases another packet p_3 at time 2 with deadline 4. Here, the online algorithm cannot finish both p_2 and p_3, but the adversary can finish p_1 and p_3 by their deadlines in order. Thus, the lower bound of competitive ratios for this variant ($w_{p_i} = 1, \forall i$ and fade states keep the same) is 2. It is intuitive to abort a running packet if it can be sent later with the given set of pending packets and fade states of the channel. Our proposed online algorithms are based on this intuition.

We provide an almost earliest-deadline-first algorithm called $\texttt{EDF}_\beta$. We use $p_{\max}$ to denote the packet with the maximum value $w_{\max}$ at time t. Since the fade states are known, there exists an efficient algorithm in calculating the provisional schedule, a feasible schedule of sending a subset of the pending packets by their deadlines. We calculate the optimal provisional schedule at time t. Let the earliest-deadline pending packet be p_e. We either schedule p_e or another packet p_f satisfying $w_{p_f} \geq \max\{\beta \cdot w_{p_e},\ w_{p_{\max}}/\beta\}$.

Algorithm 3 $\texttt{EDF}_\beta$

Abort the currently running packet p only if the new arrival with value $\geq \beta \cdot w_p$, ties are broken in favor of the packet with the earliest deadline.
if there is no currently running packet **then**
  Calculate the optimal provisional schedule, based on the set of pending packets and the known fade states.
  if $w_{p_e} \geq w_{p_{\max}}/\beta$ **then**
    Execute p_e.
  else
    Execute a packet p_f satisfying

$$w_{p_f} \geq \max\{\beta \cdot w_{p_e},\ w_{p_{\max}}/\beta\}.$$

    where ties are broken in favor of the earliest-deadline packet. Note $p_{\max}$ itself is a candidate for p_f.
  end if
end if

Theorem 11 *Assume fade states are known to online algorithms. Algorithm* $\texttt{EDF}_\beta$ *is* $\max\{2,\ \beta,\ (\frac{1}{\beta-1} \cdot (\beta^{n+1}-1)/\beta^n)\}$*-competitive in scheduling packets with deadlines by one transmitter with restarts.* $\texttt{EDF}_\beta$ *is* 2*-competitive when* $\beta = 2$.

Proof We use a potential function method to prove Theorem 11. We compare our algorithm $\texttt{EDF}_\beta$ with the adversary ADV. Let $\Phi_t^{\texttt{ADV}}$ and $\Phi_t^{\texttt{BR}}$ denote the potentials of the adversary and $\texttt{EDF}_\beta$ respectively. Specifically, $\Phi_t^{\texttt{ADV}}$ denotes the total value achieved since time t from the pending packets at time t for the adversary. Let this set of packets be S_t^*. Let $\Phi_t^{\texttt{BR}}$ denote the total value of the optimal petitional schedule of the pending packets at time t for $\texttt{EDF}_\beta$. We use p_t and p_t' to denote the t-th packet sent by $\texttt{EDF}_\beta$ and ADV respectively. If such a packet does not exist, p_t (p_t') is a null packet with value 0. To prove Theorem 11, we need to show that for any t (let $c := \max\{2,\ \beta,\ (\frac{1}{\beta-1} \cdot (\beta^{n+1}-1)/\beta^n)\}$)

$c \cdot w_{p_t} + \Delta\Phi_t^{\texttt{BR}} \geq w_{p'_t} + \Delta\Phi_t^{\texttt{ADV}}$. We provide the following invariants and prove their correctness by case study.

- Denote the pending packets at time t for ADV and $\texttt{EDF}_\beta$ as $\mathcal{P}'_t$ and $\mathcal{P}_t$. $\mathcal{P}'_t \subseteq \mathcal{P}_t$. Note that $\texttt{EDF}_\beta$ may not deliver all packets in $\mathcal{P}_t$.
- In each packet sent, the sum of the actual gain and the credit change is called *amortized gain*. We prove that for the i-th packet sent, ADV's amortized gain is no more than c times of $\texttt{EDF}_\beta$'s amortized gain. $c \cdot w_{p_t} + \Delta\Phi_t^{\texttt{BR}} \geq w_{p'_t} + \Delta\Phi_t^{\texttt{ADV}}$.

For arrivals, with the first invariant, the invariants are easy to prove. Note $w_{p_t} = w_{p'_t} = 0$. In the following, we consider packet deliveries only. Let the packet $\texttt{EDF}_\beta$ chooses to send in this duration be p. One fact that we will use is: Given two packet p and a packet p^* with $d_p \leq d_{p^*}$, if p is not in the optimal provisional schedule, but p^* is, then $w_{p^*} \geq w_p$. This fact further implies that if p is the packet $\texttt{EDF}_\beta$ is currently sending, any packet not in the optimal provisional schedule has a value $\leq \beta \cdot w_p$.

1. Assume ADV sends a packet p'. Assume p is sent successfully.

   Based on the invariants, $w_{p'},\ w_p \leq w_{p_{\max}}$. From the algorithm itself, $w_p \geq w_{p_{\max}}/\beta$. Since all packets have the same length, under any fade states, $\texttt{EDF}_\beta$ finishes p no later than ADV finishes p'. If $d_{p'} < d_p$, we have $w_{p'} < w_p$ in the optimal provisional schedule. Then we charge $w_{p'} + w_p$ to the adversary and we have $w_{p'} + w_p \leq 2 \cdot w_p$. If $d_{p'} > d_p$, p will not be sent by the adversary. Then we charge $w_{p'}$ to ADV and we have $\beta \cdot w_p \geq w_{p_{\max}} \geq w_{p'}$.

2. Assume ADV sends a packet p'. Assume p is aborted before it is finished.

   If the adversary will send p, we will charge w_p to the packet that preempts it. Like the chain we have calculated in Lemma 1, the value gained by sending the last packet of the chain is at least $(\beta - 1) \cdot \beta^n / (\beta^{n+1} - 1)$ times of the total value we charge the adversary.

3. Assume ADV has nothing to send from the currently pending packets for $\texttt{EDF}_\beta$.

   We claim that either p has been sent by ADV or ADV must have one new arrival before $\texttt{EDF}_\beta$ finishes the packet p it chooses to send. Otherwise, ADV can get more credit by delivering p. It does not hurt if we have run p till new arrivals come. This analysis is similar to what we have had for above cases.

□

Theorem 11 implies that extra information (fade states) helps improve the competitive ratio from 2.618 to 2.

Assume the fade states are unknown, but the packet input sequence is known. We first provide the lower bound $\phi \approx 1.618$ of competitive ratio for deterministic online algorithms for this variant. Then we provide competitive algorithms for it.

Theorem 12 *Consider a variant in which the fade states are unknown, but the packet input sequence is known to online algorithms. The lower bound of competitive ratio for deterministic online algorithms is* $\phi \approx 1.618$.

Proof An instance is easy to construct. Assume there are two packets in the input sequence only. One packet p_1 is with value 1 and deadline 2. The other packet p_2 is with value ϕ and deadline 3. These two packets are released at time 0. Let an online algorithm be ON.

If ON schedules p_1, the optimal offline algorithm schedules p_2 and the fade states are 0.5 from time 0 to 3. Note here the Assumption 1 still holds. Then the competitive ratio is ϕ. If ON schedules p_2, the optimal offline algorithm schedules both p_1 and p_2 given the fading states are 0.5 from 0 to 4. Thus, the competitive ratio is $(1 + \phi)/\phi = \phi$. □

In the following, we reveal the relationship between this variant and the bounded-delay model, and we prove that given a c-competitive online algorithm for the bounded-delay model, there exists a c-competitive algorithm for this variant in which fade states are unknown but packet sequence is known to online algorithms. In the bounded-delay model, packets are released in an online manner. Each packet is associated with a value and a deadline by which it should be sent. In each time step, a packet can be sent and the goal is to maximize the total value of the packets sent by their respective deadlines.

Theorem 13 *Consider a variant in which the fade states are unknown, but the packet input sequence is known to online algorithms. A c-competitive algorithm for the bounded-delay model implies a c-competitive algorithm for this variant.*

Proof Consider an input sequence $\mathcal{I}$ for the bounded-delay model. Let the packets sent by an optimal offline algorithm be $\mathcal{O}$ and the algorithm itself be $\texttt{OPT}_d$. Let the length of a packet be l.

Given a time t, we create the fade states such that the optimal offline algorithm $\texttt{OPT}_f$ for the variant achieves the same weighted throughput as $\texttt{OPT}_d$, also, for an online algorithm, the extra given information about the whole input sequence cannot avoid the difficulty brought by the unpredictably of the fade states. The construction of the fade states is as follows.

For the bounded-delay model, let the set of packets $\mathcal{O}$ be $p_1,\ p_2,\ \ldots,\ p_m$ and they are sent in time steps $1,\ 2,\ \ldots,\ m$

respectively. (If there is no packet sent in a step i, we create a dummy packet p_i for step i with $w_{p_i} = 0$. Without loss of generality, all packets p_i can be sent in the earliest-deadline-first manner. Then we modify the deadlines of the packets in $\mathcal{O}$ such that $d_{p_i} < \min\{d_{p_{i+1}}, \ldots, d_{p_m}\}$, for all $i = 1, 2, \ldots, m-1$. At last, we force the quality of the fade states from time d_{p_i} to $d_{p_{i+1}}$ be $l/(d_{p_{i+1}} - d_{p_i})$. This guarantees a packet can be sent under such fade states and if p_i is pending to an online algorithm at time $d_{p_{i-1}}$ and the online algorithm sends any other packet than p_i, p_i cannot be sent by the online algorithm any more. We ensure that the optimal offline algorithm for this variant works the same as the optimal offline algorithm for the bounded-delay model. Also, the extra information about the packet input sequence does not help the online algorithm since it has no known about the fade states. With Assumption 1, the online algorithm known that only one packet can be sent once it is committed and this is exactly as what is assumed in the bounded-delay model. □

Closing or shrink the gap of competitive ratios $[1.618,\ 1.832]$ for the bounded-delay model is an intriguing problem and thus, from Theorem 13, the gap still applies to the variant in which the fade states are unknown, but the packet input sequence is known to online algorithms.

References

[1] W. Aiello, Y. Mansour, S. Rajagopolan, and A. Rosen. Competitive queue policies for differentiated services. In *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 431–440, 2000.

[2] W. Aiello, Y. Mansour, S. Rajagopolan, and A. Rosen. Competitive queue policies for differentiated services. *Journal of Algorithms*, 55(2):113–141, 2005.

[3] Baptiste. Polynomial time algorithms for minimizing the weighted number of late jobs on a single machine with equal processing times. *Journal of Scheduling*, 2:245–252, 1999.

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

[5] W. Chen, M. J. Neely, and U. Mitra. Energy-efficient transmission with individual packet delay constraints. *IEEE Transactions on Information Theory*, 54(5):2090–2109, 2008.

[6] F. Y. L. Chin and S. P. Y. Fung. Online scheduling with partial job values: Does timesharing or randomization help? *Algorithmica*, 37(3):149–164, 2003.

[7] M. Chrobak. 2007 — An offine perspective. *SIGACT News Online Algorithms*, 13:96–121, 2008.

[8] M. Chrobak, W. Jawor, J. Sgall, and T. Tichy. Online scheduling of equal-length jobs: Randomization and restart help? *SIAM Journal on Computing (SICOMP)*, 36(6):1709–1728, 2007.

[9] M. Englert and M. Westermann. Considering suppressed packets improves buffer management in QoS switches. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 209–218, 2007.

[10] A. Fu, E. Modiano, and J. Tsitsiklis. Optimal transmission scheduling over a fading channel with energy and deadline constraints. *IEEE Transactions on Wireless Communications*, 6(1):630–641, 2006.

[11] A. El Gamal, E. Uysal, and B. Prabhakar. Energy-efficient transmission over a wireless link via lazy packet scheduling. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 1, pages 384–394, 2001.

[12] M. Garey, D. Johnson, B. Simons, and R. Tarjan. Scheduling unit-time tasks with arbitrary release times and deadlines. *SIAM Journal on Computing (SICOMP)*, 10(2):256–269, 1981.

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

[14] B. Hajek. On the competitiveness of online scheduling of unit-length packets with hard deadlines in slotted time. In *Proceedings of 2001 Conference on Information Sciences and Systems (CISS)*, pages 434–438, 2001.

[15] T. Heikkinen and A. Hottinen. Delay-differentiated scheduling in a fading channel. *IEEE Transactions on Wireless Communications*, 7(3):848–856, 2008.

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

[17] F. Li, J. Sethuraman, and C. Stein. An optimal online algorithm for packet scheduling with agreeable deadlines. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 801–802, 2005.

[18] F. Li, J. Sethuraman, and C. Stein. Better online buffer management. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 199–208, 2007.

[19] A. Tarello, J. Sun, M. Zafer, and E. Modiano. Minimum energy transmission scheduling subject to deadline contraints. *Wireless Neworks*, 14(5):633–645, 2007.

[20] D.N. Tse and S.V. Hanly. Multiaccess fading channels: Polymatroid structure, optimal resource allocation and throughput capacities. *IEEE Transactions on Information Theory*, 44(7):2796–2815, 1998.

[21] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced CPU energy. In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1995.

[22] M. Zafer and E. Modiano. Optimal rate control for delay-constrained data transmission over a wireless channel. *IEEE Transactions on Information Theory*, 54(9):4020–4039, 2008.