# Optimal Energy-Efficient Downlink Transmission Scheduling for Real-Time Wireless Networks

Lei Miao, Jianfeng Mao, and Christos G. Cassandras, *Fellow, IEEE*

***Abstract*—It has been shown that by using appropriate channel coding schemes in wireless environments, transmission energy can be significantly reduced by controlling the packet transmission rate. This paper seeks optimal solutions for downlink transmission control problems, motivated by this observation and by the need to minimize energy consumption in real-time wireless networks. Our problem formulation deals with a more general setting than the paper authored by Gamal *et al.*, in which the MoveRight algorithm is proposed. The MoveRight algorithm is an iterative algorithm that converges to the optimal solution. We show that even under the more general setting, the optimal solution can be efficiently obtained through an approach decomposing the optimal sample path through certain "critical tasks" which, in turn, can be efficiently identified. We include simulation results showing that our algorithm is significantly faster than the MoveRight algorithm. We also discuss how to utilize our results and receding horizon control to perform online transmission scheduling where future task information is unknown.**

***Index Terms*—Energy efficiency, optimization, real-time systems, receding horizon control, wireless networks.**

## I. Introduction

BECAUSE wireless nodes are normally powered by batteries and are expected to remain in operation for extended periods of time, how to conserve energy in order to extend node lifetime and network lifetime is a major research issue in most wireless networks. One way to save energy is to operate these nodes at low power as long as possible. However, this will also significantly downgrade their functionality. Therefore, there is a tradeoff between energy and the "quality" delivered by wireless nodes. When "quality" is measured in terms of latency, the tradeoff is between energy and time. Examples arise in real-time computing, where a processor trades off the processing rate for energy [1]; and in wireless transmission, where a transmitter trades off transmission speed for energy [2].

Manuscript received October 28, 2015; revised February 8, 2016; accepted March 15, 2016. Date of publication March 22, 2016; date of current version December 15, 2017. The work was supported in part by the National Science Foundation under Grant DMI-0330171, in part by AFOSR under grants FA9550-04-1-0133 and FA9550-04-1-0208, in part by ARO under Grant DAAD19-01-0610, and in part by Honeywell Laboratories. Recommended by Associate Editor Y. Zhang.

L. Miao is with the Department of Engineering Technology, Middle Tennessee State University, Murfreesboro, TN 37132 USA (e-mail: lei.miao@mtsu.edu).

J. Mao is with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, 518100 China (e-mail: maoeco@live.com).

C. G. Cassandras is with the Division of Systems Engineering and the Department of Electrical and Computer Engineering, Boston University, Brookline, MA 02446 USA (e-mail: cgc@bu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCNS.2016.2545099

When the energy of a wireless node is consumed mostly by communication tasks, scheduling a radio-frequency (RF) transmission efficiently becomes extremely important in conserving the energy of the node. It is well known that there exists an explicit relationship between transmission power and channel capacity [3]; transmission power can be adjusted by changing the transmission rate, provided that appropriate coding schemes are used. This provides an option to conserve the transmission energy of a wireless node by slowing down the transmission rate. Increased latency is a direct side effect caused by the low transmission rate and it can affect other quality-of-service (QoS) metrics as well. For example, excessive delay may cause buffer overflow, which increases the packet dropping rate. The existence of this tradeoff between energy and latency motivates *dynamic transmission scheduling* (DTS) techniques for designing energy-efficient wireless systems.

To the best of our knowledge, the earliest work that captures the tradeoff between energy and latency in transmission scheduling is [4], in which Collins and Cruz formulated a Markov decision problem for minimizing transmission cost subject to some power constraints. By assuming a linear dependency between transmission cost and time, their model did not consider the potential of more energy savings by varying the transmission rate. Berry [5] considered a Markov decision process in the context of wireless fading channels to minimize the weighted sum of average transmission power and a buffer cost, which corresponds to either average delay or the probability of buffer overflow. Using dynamic programming and assuming the transmission cost to be a convex function of time, Berry discovered some structural properties of the optimal adaptive control policy, which relies on information on the arrival state, the queue state, and the channel state. In [6] and [7], Ata developed optimal dynamic power control policies subject to a QoS constraint for Markovian queues in wireless static channels and fading channels, respectively. In his work, the optimization problem was formulated to minimize the long-term average transmission power, given a constraint of buffer overflow probability in equilibrium; dynamic programming, and Lagrangian relaxation approaches were used in deriving the optimal policies, which can be expressed as functions of the packet queue length and the channel state. Neely utilized a Lyapunov drift technique in [8] to develop a dynamic power allocation and routing algorithm that minimizes the average power of a cell-partitioned wireless network. It was shown that the online algorithm operates without knowledge of traffic rates or channel statistics, and yields average power that is arbitrarily close to the offline optimal solution. A related problem of maximizing throughput subject to peak and average power constraints was also discussed in [8].

Today's real-time data communications require a quality-of-service (QoS) guarantee for each individual packet. Another line of research aims at minimizing the transmission energy over a single wireless link while providing QoS guarantee. In particular, it is assumed that each packet is associated with an arrival time (generally random), a number of bits, a hard deadline that must be met, and an energy function. This line of work was initially studied in [9] with follow-up work in [2] where a "homogeneous" case is considered assuming all packets have the same deadline and number of bits. By identifying some properties of this convex optimization problem, Gamal *et al.* proposed the "MoveRight" algorithm in [2] to solve it iteratively. However, the rate of convergence of the MoveRight algorithm is only obtainable for a special case of the problem when all packets have identical energy functions; in general, the MoveRight algorithm may converge slowly. Zafer *et al.* [10] studied an optimal rate control problem over a time-varying wireless channel, in which the channel state was considered to be a Markov process. In particular, they considered the scenario that $B$ units of data must be transmitted by a common deadline $T$, and they obtained an optimal rate-control policy that minimizes the total energy expenditure subject to short-term average power constraints. In [11] and [12], the case of identical arrival time and individual deadline is studied by Zafer *et al.* In [13], the case of identical packet size and identical delay constraint is studied by Neely *et al.* They extended the result for the case of individual packet size and identical delay constraint in [14]. In [15], Zafer *et al.* used a graphical approach to analyze the case that each packet has its own arrival time and deadline. However, there were certain restrictions in their setting, for example, the packet that arrives later must have later deadlines. Wang and Li [16] analyzed scheduling problems for bursty packets with strict deadlines over a single time-varying wireless channel. Assuming slotted transmission and changeable packet transmission order, they are able to exploit structural properties of the problem to come up with an algorithm that solves the offline problem. In [17], Poulakis *et al.* also studied energy-efficient scheduling problems for a single time-varying wireless channel. They considered a finite-horizon problem where each packet must be transmitted before $D_{\max}$. Optimal stopping theory was used to find the optimal start transmission time between $[0, D_{\max}]$ in order to minimize the expected energy consumption and the average energy consumption per unit of time. In [18], an energy-efficient and deadline-constrained problem was formulated in lossy networks to maximize the probability that a packet is delivered within the deadline minus a transmission energy cost. Dynamic programming-based solutions were developed under a finite-state Markov channel model. Shan *et al.* [19] studied discrete rate scheduling problems for packets with individual deadlines in energy harvesting systems. Under the assumption that later packet arrivals have later deadlines, they established connections between continuous rate and discrete rate algorithms. A truncation algorithm was also developed to handle the case that harvested energy is insufficient to guarantee all packets' deadlines are met. Tomasi *et al.* [20] developed transmission strategies to deliver a prescribed number of packets by a common deadline $T$ while minimizing transmission attempts. Modeling the time-varying correlated wireless channel as a Markov chain, they used dynamic programming and a heuristic strategy to address three systems, in which the receiver provides the channel state information to the transmitter differently. Zhong and Xu [21] formulated optimization problems that minimize the energy consumption of a set of tasks with task-dependent energy functions and packet lengths. In their problem formulation, the energy functions include both transmission energy and circuit power consumption. To obtain the optimal solution for the offline case with backlogged tasks only, they developed an iterative algorithm RADB whose complexity is $O(n^2)$ ($n$ is the number of tasks). The authors show via simulation that the RADB algorithm achieves good performance when used in online scheduling. In [22], Vaze derived the competitive ratios of online transmission scheduling algorithms for single-source and two-source Gaussian channels in energy harvesting systems. In Vaze's problem formulation, the goal is to minimize the transmission time of fixed $B$ bits using harvested energy, which arrive in chunks randomly.

In the aforementioned papers, the closest ones to this paper are [2], [14], and [15]. In this paper, we consider the transmission control problem in the scenario that each task has arbitrary arrival time, deadline, and number of bits. Therefore, the problem we study in this paper is more generic and challenging.

Our model also allows each packet to have its own energy function. This makes our results especially applicable to DTS scenarios, where a transmitter transmits to multiple receivers over slow-fading channels. Our contributions are as follows: by analyzing the structure of the optimal sample path, we solve the DTS problem efficiently using a two-fold decomposition approach. First, we establish that the problem can be reduced to a set of subproblems over segments of the optimal sample path defined by "critical tasks." Secondly, we establish that solving each subproblem boils down to solving nonlinear algebraic equations for the corresponding segments. Based on the above decomposition approach, an efficient algorithm that solves the DTS problem is proposed and compared to the MoveRight algorithm. Simulation results show that our algorithm is typically an order of magnitude faster than the MoveRight algorithm.

The main results of the paper were previously published in [23]. In this journal version, we have added Sections III-C and IV, and moved the proofs to an Appendix. The structure of this paper is as follows: in Section II, we formulate our DTS problem and discuss some related work; the main results of DTS are presented in Section III, where an efficient algorithm is proposed and is shown to be optimal; in Section IV, we discuss how our main results can be used to perform online transmission control; finally, we conclude in Section V.

## II. Downlink Transmission Scheduling Problem and Related Work

We assume that the channel between the transmitter and the receiver is an additive white Gaussian noise (AWGN) channel and the interference to the receiver is negligible. The received signal at time $t$ can be written as

$$Y(t) = \sqrt{g(t)}X(t) + n(t) \tag{1}$$

where $g(t)$ is the channel gain, $X(t)$ is the transmitted signal, and $n(t)$ is additive white Gaussian noise [24]. We consider the case that the transmitter is in isolation from other transmitters so that the interference is negligible. Due to channel fading, $g(t)$ is time varying in general. We will consider $g(t)$ to be time-invariant during the transmission of a single packet. Although in practice the channel state may change during the transmission of a packet, our results are still helpful, since it is valid to estimate the unknown future channel state to be static for each packet in an online setting. Note that our results can be possibly extended to fast fading channels as well.

The DTS problem arises when a wireless node has a set of $N$ packets that need to be sent to different neighboring nodes. The goal is to minimize the total transmission energy consumption while guaranteeing hard deadline satisfaction for each individual packet. Since each packet can be considered as a communication task, we use the terms "task" and "packet" interchangeably in what follows. We model the transmitter as a single-server queueing system operating on a non-preemptive and first-come-first-served (FCFS) basis, whose dynamics are given by the well-known max-plus equation

$$x_i = \max(x_{i-1}, a_i) + s_i \tag{2}$$

where $a_i$ is the arrival time of task $i = 1, 2, \ldots, x_i$ is the time when task $i$ completes service, and $s_i$ is its (generally random) service time.

Note that although preemption is often easy and straightforward in computing systems, it is very costly and also technically hard in wireless transmissions. Therefore, we assume a non-pre-emptive model in this paper. Transmission rate control typically occurs in the physical layer, and changing packet order may cause problems in the upper layers of the network stack. Thus, we use a simple FCFS model to avoid packet out-of-sequence problems. It is also worth noting that even if the packet order is changeable, determining the optimal packet order is a separate problem. Once the order of transmission is decided by a specific scheduling policy, our work can be used to minimize the energy expenditure for that specific order.

The service time $s_i$ is controlled by the transmission rate, which is determined by transmission power and coding scheme. However, it turns out that it is more convenient to use the reciprocal of the transmission rate as our control variable in the DTS problem. Thus, we define $\tau$ to be the transmission time per bit and $\omega_i(\tau)$ to be the energy cost per bit for task $i$. Clearly, $\omega_i(\tau)$ is a function of $\tau$. Since the channel gain $g(t)$ in (1) is constant, $\omega_i(\tau)$ is kept fixed during the transmission of task $i$.

We formulate the offline DTS problem as follows:

$$\begin{aligned}
\mathbf{P1}: \quad & \min_{\tau_1,\ldots,\tau_N} \sum_{i=1}^{N} v_i \omega_i(\tau_i) \\
\text{s.t.} \quad & x_i = \max(x_{i-1}, a_i) + v_i \tau_i \le d_i, \quad i = 1, \ldots, N \\
& \tau_i > 0,\ x_0 = 0
\end{aligned}$$

where $d_i$ and $v_i$ are the deadline and the number of bits of task $i$, respectively.

In realistic scenarios, the maximum transmission power of a wireless system puts a constraint on each $\tau_i$, that is, $\tau_i \ge \tau_{i\_\min}$, where $\tau_{i\_\min}$ is the minimum amount of time used for transmitting one bit in task $i$. For ease of analysis, we omitted this constraint in **P1**. However, it is important to note that special handling is needed in real-world systems for the case that the optimal solution $\tau_i^*$ is below the minimum value $\tau_{i\_\min}$. For example, the system may simply choose to drop the packet or transmit the packet using control $\tau_{i\_\min}$. We will discuss the problem that includes this constraint in Sections III-C and IV.

Note that in the offline setting, we consider $a_i$, $d_i$, and $v_i$ are known. The downlink scheduling problem formulated in [2] is a special case of **P1** above: in [2], each task has the same deadline and number of bits, that is, $d_i = T$, $v_i = v$, for all $i$. Note that transmission-rate constraints are omitted in **P1** and we assume the transmission rate can vary continuously. In practical systems, the control can always be rounded to the nearest achievable value [25].

Problem **P1** above is similar to the general class of problems studied in [26] and [27] without the constraints $x_i \le d_i$, where a decomposition algorithm called the Forward Algorithm (FA) was derived. As shown in [26] and [27], instead of solving this complex nonlinear optimization problem, we can decompose the optimal sample path into a number of busy periods. A *busy period* (BP) is a contiguous set of tasks $\{k, \ldots, n\}$ such that the following three conditions are satisfied: $x_{k-1} < a_k$, $x_n < a_{n+1}$, and $x_i \ge a_{i+1}$, for every $i = k, \ldots, n-1$. Notice that **P1** above exploits static control ($\tau_i$ was kept fixed during the service time of task $i$). This is straightforward in wireless transmission control since the transmission rate of a single packet/task is often fixed. In addition, it has been shown in [28] that when the energy functions $\omega_i(\tau)$, $i = 1, \ldots, N$, are strictly convex and monotonically decreasing in $\tau$, there is no benefit in applying dynamic control ($\tau_i$ varies over time during the service time of task $i$). It has also been shown in [29] that when the energy functions are identical in **P1**, its solution is obtained by an efficient algorithm (Critical Task Decomposition Algorithm) that decomposes the optimal sample path even further and does not require solving any convex optimization problem at all. In this paper, we will consider the much harder case that the energy functions are *task dependent*. When the energy functions are homogeneous, it is shown in [29] that the exact form of the energy function does not matter in finding the optimal solutions. The main challenge of having heterogeneous energy functions is that these energy functions will be used to identify the optimal solutions, and this adds an extra layer of complexity. We shall still use the decomposition idea in [29], and we will use $\{\tau_i^*\}$ and $\{x_i^*\}$, $i = 1, \ldots, N$, to denote the optimal solution of **P1** and the corresponding task departure times, respectively.

Typically, $\omega_i(\tau)$ is determined by factors including the channel gain $g(t)$, transmission distance, signal-to-noise ratio, and so on. Therefore, when a wireless node transmits to different neighbors at different times, different $\omega_i(\tau)$ are involved. We begin with an assumption that will be made throughout our analysis.

*Assumption 1:* In AWGN channels, $\omega_i(\tau)$ is non-negative, strictly convex, monotonically decreasing, differentiable, and $\lim_{\tau \to 0} \dot{\omega}_i(\tau) = -\infty$.

Assumption 1 is justified in [2] and channel coding schemes supporting this assumption can be found in [9]. Note that the result obtained in [28] can be readily applied here: the unique optimal control to **P1** is static. This means that we do not need to vary the transmission rate of task $i$ during its transmission time.

## III. Main Results of DTS

### A. Optimal Sample Path Decomposition

The following two lemmas help us decompose the optimal sample path of **P1**. Their proofs are very similar to the proofs for Lemmas 1 in [29], and only monotonicity of $\omega_i(\tau)$ is required. We omit the proofs here.

*Lemma 3.1:* If $d_i < a_{i+1}$, then $x_i^* = d_i$.

*Lemma 3.2:* If $d_i \geq a_{i+1}$, then $a_{i+1} \leq x_i^*$.

Recalling the definition of a BP, Lemmas 3.1, 3.2 show that the BP optimal structure can be explicitly determined by the deadline-arrival relationship, that is, a sequence of contiguous packets $\{k, \ldots, n\}$ is a BP if and only if the following is satisfied: $d_{k-1} < a_k, d_n < a_{n+1}, d_i \geq a_{i+1}$, for all $i \in \{k, \ldots, n-1\}$. After identifying each BP on the optimal sample path, problem **P1** is reduced to solving a separate problem over each BP. We formulate the following optimization problem for BP $\{k, \ldots, n\}$:

$$
\begin{aligned}
Q(k,n): \quad & \min_{\tau_k,\ldots,\tau_n} \sum_{i=k}^{n} v_i \omega_i(\tau_i) \\
\text{s.t.} \quad & x_i = a_k + \sum_{j=k}^{i} v_i \tau_i \leq d_i, \quad i = k, \ldots, n \\
& \tau_i > 0, \quad i = k, \ldots, n \\
& x_i \geq a_{i+1}, \quad i = k, \ldots, n-1.
\end{aligned}
$$

Although $Q(k,n)$ is easier than **P1** (since it does not contain max-plus equations, which are nondifferentiable), it is still a hard convex optimization problem. Naturally, we would like to solve $Q(k,n)$ efficiently. As we will show, this is indeed possible by further decomposing a BP $\{k, \ldots, n\}$ through special tasks called "critical tasks," which are defined as follows:

*Definition 1:* Suppose both task $i$ and $i+1$ are within a BP $\{k, \ldots, n\}$ on the optimal sample path of **P1**. If $\dot{\omega}_i(\tau_i^*) \neq \dot{\omega}_{i+1}(\tau_{i+1}^*)$, task $i$ is **critical**. If $\dot{\omega}_i(\tau_i^*) > \dot{\omega}_{i+1}(\tau_{i+1}^*)$, then task $i$ is **left-critical**. If $\dot{\omega}_i(\tau_i^*) < \dot{\omega}_{i+1}(\tau_{i+1}^*)$, then task $i$ is **right-critical**.

These critical tasks are special because the derivatives of the energy function change after these tasks are transmitted on the optimal sample path. Therefore, identifying critical tasks is crucial in solving $Q(k,n)$. In fact, Gamal *et al.* [2] observed the existence of left-critical tasks. However, they did not make use of them in characterizing the optimal sample path. In order to accomplish this, we need to study the relationship between critical tasks and the structure of the optimal sample path. An auxiliary lemma will be introduced first.

*Lemma 3.3:* If $v_1\tau_1 + v_2\tau_2 = v_1\tau_1' + v_2\tau_2'$, $\tau_1' < \tau_1, \tau_2' > \tau_2$, and $\dot{\omega}_1(\tau_1') > \dot{\omega}_2(\tau_2')$, then $v_1\omega_1(\tau_1) + v_2\omega_2(\tau_2) > v_1\omega_1(\tau_1') + v_2\omega_2(\tau_2')$.

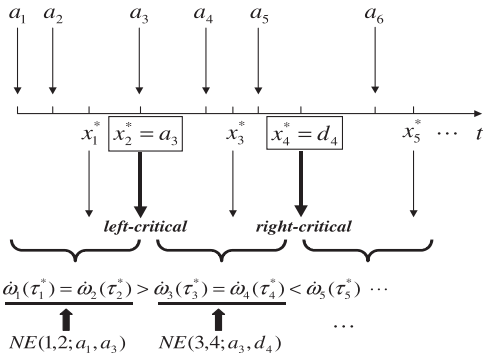


Fig. 1. Illustration of the optimal structure of BP $\{k, \ldots, n\}$.

Lemma 3.3 implies that under Assumption 1 (especially, the convexity assumption), it takes the least amount of energy to transmit two tasks in a given amount of time when the derivatives of the two energy functions have the least amount of difference. As we will see later, this auxiliary lemma will be used to establish other important results. Next, we will discuss what exactly makes the critical tasks (defined in Definition 1) special.

*Lemma 3.4:* Suppose both task $i$ and $i+1$ are within a BP $\{k, \ldots, n\}$ on the optimal sample path of **P1**. i) If task $i$ is left-critical, then $x_i^* = a_{i+1}$. ii) if task $i$ is right-critical, then $x_i^* = d_i$.

This result shows that if a task is *left-critical* or *right-critical* on the optimal sample path, its optimal departure time is given by the next arrival time or its deadline, respectively. The lemma implies that when $a_{i+1} < x_i^* < d_i$, task $i$ is neither *left-critical* nor *right-critical*. In our next step, we will study the commonality among a block of consecutive noncritical tasks, which are in the middle of two adjacent critical tasks. The result will help us better understand the structure of the optimal sample path, using which we will develop an efficient algorithm to solve $Q(k,n)$.

*Remark 3.1:* For any two neighboring tasks $i$ and $i+1$ in a BP $\{k, \ldots, n\}$ on the optimal sample path of **P1**, if task $i$ is not a critical task, then $\dot{\omega}_i(\tau_i^*) = \dot{\omega}_{i+1}(\tau_{i+1}^*)$.

This remark is the direct result of Definition 1. Using this remark and Lemma 3.4, we can obtain the structure of BP $\{k, \ldots, n\}$ on the optimal sample path of **P1** as follows: $\{k, \ldots, n\}$ is characterized by a sequence of tasks $S = \{c_0, \ldots, c_{m+1}\}$, in which $c_0 = k$, $\{c_1, \ldots, c_m\}$ contains all critical tasks in $\{k, \ldots, n\}$ (the optimal departure times of these critical tasks are given by Lemma 3.4), and task $c_{m+1} = n$. Moreover, let $c_i$, $c_{i+1}$ be adjacent tasks in $S$. Then, the segment of tasks

$$
\begin{cases}
\{c_i, \ldots, c_{i+1}\}, & \text{if } i = 0 \\
\{c_i + 1, \ldots, c_{i+1}\}, & \text{if } 0 < i \leq m
\end{cases}
\tag{3}
$$

operates at some $\tau$ such that the derivatives of their energy functions are all the same. To have a better understanding of this optimal structure, see Fig. 1. In this example, task 2 is left-critical and task 4 is right-critical. Their optimal departure times are $a_3$ and $d_4$, respectively. In the set $S = \{1, 2, 4, \ldots\}$, tasks $\{1,2\}$ and $\{3,4\}$ are examples of the segments defined above. Invoking Remark 3.1, $\tau_1^*, \ldots, \tau_4^*$ are characterized by $\dot{\omega}_1(\tau_1^*) = \dot{\omega}_2(\tau_2^*), \dot{\omega}_3(\tau_3^*) = \dot{\omega}_4(\tau_4^*)$, and $\dot{\omega}_2(\tau_2^*) > \dot{\omega}_3(\tau_3^*), \dot{\omega}_4(\tau_4^*) < \dot{\omega}_5(\tau_5^*)$.

In order to obtain our main result of this section and the explicit algorithm that solves $Q(k,n)$, we next define a system of nonlinear algebraic equations as follows with $i<j$, $0\le t_1\le t_2$, and unknown variables $\tau_i,\dots,\tau_j$

$$NE(i,j;t_1,t_2):\quad \begin{aligned}&\sum_{m=i}^{j}\tau_m v_m=t_2-t_1\\ &\dot\omega_m(\tau_m)=\dot\omega_{m+1}(\tau_{m+1})\\ &m=i,\dots,j-1.\end{aligned}$$

Its solution minimizes the total energy of transmitting tasks $\{i,\dots,j\}$ that do not contain critical tasks within time interval $t_2-t_1$. Note that when $i=j$, the above nonlinear algebraic equations reduce to a single linear equation $\tau_i v_i=t_2-t_1$.

In Fig. 1, we illustrated the structure of a BP on the optimal sample path of **P1**. In fact, given all critical tasks in the BP, the optimal solution can be obtained by solving a set of $NE$ systems, one for each segment defined in (3). For example, in Fig. 1, the optimal controls of tasks $\{1,2\}$ and $\{3,4\}$ can be obtained by solving $NE(1,2;a_1,a_3)$ and $NE(3,4;a_3,d_4)$, respectively.

At this point, we have established that solving problem $Q(k,n)$ boils down to identifying critical tasks on its optimal sample path. This relies on some additional properties of the optimal sample path. To obtain them, we need to first study the properties of $NE(i,j;t_1,t_2)$.

We denote the solution to $NE(i,j;t_1,t_2)$ by $\tau_i(t_1,t_2),\dots,\tau_j(t_1,t_2)$. We define the common derivative in $NE(i,j;t_1,t_2)$

$$\sigma_{i,j}(t_1,t_2)=\dot\omega_m\left(\tau_m(t_1,t_2)\right),\ \text{for any } m, i\le m\le j$$

and note that $\sigma_{i,j}(t_1,t_2)$ is the derivative of the energy function of any task in $\{i,\dots,j\}$. When $t_1=t_2$, we set $\sigma_{i,j}(t_1,t_2)$ to $-\infty$. Later, when invoking the definition of critical tasks, we will use $\sigma_{i,j}(t_1,t_2)$ instead of the derivative of the energy function of a single task.

Now, we are ready to introduce the properties of $NE(i,j;t_1,t_2)$ in the next lemma.

*Lemma 3.5:* When $t_1<t_2$, $NE(i,j;t_1,t_2)$ has the following properties:

i) It has a unique solution.
ii) The common derivative $\sigma_{i,j}(t_1,t_2)$ is a monotonically increasing function of $\Delta=t_2-t_1$, that is, $\sigma_{i,j}(t_1,t_2)<\sigma_{i,j}(t_3,t_4)$, if $t_4-t_3>t_2-t_1$.
iii) For any $p$, $i\le p<j$, define the partial sum $S_{ip}\equiv\sum_{m=i}^{p}\tau_m(t_1,t_2)v_m$. Then, $\sigma_{i,p}(t_1,t_1+S_{ip})=\sigma_{p+1,j}(t_1+S_{ip},t_2)=\sigma_{i,j}(t_1,t_2)$.
iv) For any $p$, $i\le p<j$, $t_1<t_3<t_2$, let $c_1=\sigma_{i,p}(t_1,t_3)$, $c_2=\sigma_{p+1,j}(t_3,t_2)$, $c_3=\sigma_{i,j}(t_1,t_2)$. If $c_q\ne c_r\ \forall\, q,r\in\{1,2,3\}$, $q\ne r$, then $\min(c_1,c_2)<c_3<\max(c_1,c_2)$.

### *B. Left and Right-Critical Task Identification*

Based on the aforementioned results, we have characterized the special structure of the optimal sample path of **P1**. To summarize, Lemmas 3.1 and 3.2 show that the BP structure of the optimal sample path can be explicitly determined by the deadline-arrival relationship. This transforms **P1** into a set of simpler convex optimization problems with linear constraints. Although the problem becomes easier to solve, it is still computationally hard for wireless devices without powerful processors and sufficient energy. Note that in the homogeneous case, when all tasks have the same arrival time and deadline, they should be transmitted with the same derivatives of their cost functions. In this case, the optimal solution can be obtained by solving the nonlinear system $NE(i,j;t_1,t_2)$. With the presence of inhomogeneous real-time constraints, we showed in Lemma 3.4 and Remark 3.1 that a set of "critical tasks" plays a key role to determine the optimal sample path, that is, the derivatives of the cost functions only change at these critical tasks. Once they are determined, the original problem $Q(k,n)$ boils down to a set of nonlinear algebraic equations.

Having obtained the properties of $NE(i,j;t_1,t_2)$, we will next develop an efficient algorithm to identify critical tasks. Without loss of generality, we only prove the correctness of identifying the first critical task. Other critical tasks can be identified iteratively. In addition, our proof will focus on right-critical tasks only, and we omit the proof for left-critical tasks, which is very similar.

We will first give some definitions. For tasks $(p,i)$ within a BP $\{k,\dots,n\}$, that is, $k\le p<i\le n$, define

$$T_1(k,p)=\begin{cases}a_k, & p=k\\ x^*_{p-1}, & p>k\end{cases},\ T_2(n,i)=\begin{cases}a_{i+1}, & i<n\\ d_n, & i=n.\end{cases}$$

Recalling the definition of a BP, $T_1(k,p)$ is defined as *the optimal starting transmission time for task* $p$, which is within a BP starting with task $k$. Recalling Lemmas 3.1 and 3.2, $T_2(n,i)$ is defined as *the earliest possible transmission ending time for task* $i$, which is within a BP ending with task $n$. Note that in order to guarantee the real-time constraints, task $i$ must be done by its deadline $d_i$. We will use $T_1(k,p)$, $T_2(n,i)$, and $d_i$ later to identify critical tasks.

We further define

$$R_i=\underset{s\in\{p,\dots,i-1\}}{\arg\max}\begin{cases}\sigma_{p,s}\left(T_1(k,p),d_s\right)\\ \le\sigma_{p,j}\left(T_1(k,p),d_j\right),\\ \text{for } i,p<i\le n,\ \text{and all } j\in\{p,\dots,i-1\}\end{cases}\quad(4)$$

$$L_i=\underset{s\in\{p,\dots,i-1\}}{\arg\max}\begin{cases}\sigma_{p,s}\left(T_1(k,p),T_2(n,s)\right)\\ \ge\sigma_{p,j}\left(T_1(k,p),T_2(n,j)\right),\\ \text{for } i,p<i\le n,\ \text{and all } j\in\{p,\dots,i-1\}\}.\end{cases}\quad(5)$$

Note that $R_i$ and $L_i$ are the tasks with the largest index in $\{p,\dots,i-1\}$ that satisfy the inequalities in (4) and (5), respectively. It is clear that $p\le R_i<i$, $p\le L_i<i$.

A special case of (4) and (5) arises when $p$ is the first task of a BP $\{k,\dots,n\}$, that is, $p=k$. Then, according to the definitions above, we obtain the following inequalities, which will be used in our later results:

$$\begin{aligned}&\sigma_{k,R_i}\left(a_k,d_{R_i}\right)\le\sigma_{k,m}(a_k,d_m),\\ &\quad\text{for } i,k<i\le n,\ \text{and all } m\in\{k,\dots,i-1\}\end{aligned}\quad(6)$$

$$\begin{aligned}&\sigma_{k,L_i}\left(a_k,T_2(n,L_i)\right)\ge\sigma_{k,m}\left(a_k,T_2(n,m)\right),\\ &\quad\text{for } i,k<i\le n,\ \text{and all } m\in\{k,\dots,i-1\}.\end{aligned}\quad(7)$$

After introducing the above definitions and notations, we are now ready to introduce three important lemmas, which will be used to prove our main theorem.

*Lemma 3.6:* Let tasks $\{k,\dots,n\}$ form a BP on the optimal sample path of **P1** and task $r>k$ be the first right-critical task in $\{k,\dots,n\}$. If $\sigma_{k,r}(a_k,d_r)\geq\sigma_{k,L_r}(a_k,a_{L_r+1})$, then there is no left-critical task in $\{k,\dots,r-1\}$.

*Lemma 3.7:* Let tasks $\{k,\dots,n\}$ form a BP on the optimal sample path of **P1**. Consider task $R_i$, for $i$, $k<i\leq n$. If $\sigma_{k,j}(a_k,d_j)\geq\sigma_{k,L_j}(a_k,a_{L_j+1})$ and $\sigma_{k,j}(a_k,a_{j+1})\leq\sigma_{k,R_j}(a_k,d_{R_j})$, for all $j$, $k<j<i$, then there is no right-critical task before task $R_i$.

*Lemma 3.8:* Let tasks $\{k,\dots,n\}$ form a BP on the optimal sample path of **P1**. If $\sigma_{k,i}(a_k,a_{i+1})>\sigma_{k,R_i}(a_k,d_{R_i})$, $\sigma_{k,j}(a_k,d_j)\geq\sigma_{k,L_j}(a_k,a_{L_j+1})$, and $\sigma_{k,j}(a_k,a_{j+1})\leq\sigma_{k,R_j}(a_k,d_{R_j})$, for $i$, $k<i\leq n$, and for all $j$, $k<j<i$, then $R_i$ is right-critical.

Before we introduce the main theorem, we would like to first summarize the above three lemmas.

Lemma 3.6 provides the conditions under which there are no left-critical tasks before the first right-critical task $r$ in a BP.

Lemma 3.7 provides the conditions under which there are no right-critical tasks before a given task $R_i$ in a BP.

Lemma 3.8 provides the conditions under which task $R_i$ in a BP is right-critical.

With the help of the aforementioned auxiliary results, we are able to establish the following theorem, which can identify the first critical task in a BP on the optimal sample path of **P1**:

*Theorem 3.1:* Let tasks $\{k,\dots,n\}$ form a BP on the optimal sample path of **P1**.

i) If

$$\sigma_{k,j}(a_k,d_j)\geq\sigma_{k,L_j}(a_k,a_{L_j+1}) \tag{8}$$

$$\sigma_{k,j}(a_k,a_{j+1})\leq\sigma_{k,R_j}(a_k,d_{R_j}) \tag{9}$$

$$\sigma_{k,i}(a_k,a_{i+1})>\sigma_{k,R_i}(a_k,d_{R_i}) \tag{10}$$

for $i$, $k<i\leq n$, and all $j$, $k<j<i$, then $R_i$ is the first critical task in $\{k,\dots,n\}$, and it is right-critical.

ii) If

$$\sigma_{k,j}(a_k,d_j)\geq\sigma_{k,L_j}(a_k,a_{L_j+1})$$

$$\sigma_{k,j}(a_k,a_{j+1})\leq\sigma_{k,R_j}(a_k,d_{R_j})$$

$$\sigma_{k,i}(a_k,d_i)<\sigma_{k,L_i}(a_k,a_{L_i+1})$$

for $i$, $k<i\leq n$, and all $j$, $k<j<i$, then $L_i$ is the first critical task in $\{k,\dots,n\}$, and it is left-critical.

Let us look at the first part of Theorem 3.1 again. The first right-critical task of a BP on the optimal sample path of **P1** can be correctly identified if we can find $i$ and $R_i$ which satisfy (8)–(10). In essence, (8) guarantees that there is no left-critical task before $R_i$, (9) guarantees that there is no right-critical task before $R_i$, and (10) guarantees that $R_i$ is a right-critical task. A similar argument applies to the second part of the theorem. Therefore, the conditions in Theorem 3.1 are not only sufficient but also necessary for identifying the first critical task.

After obtaining the first critical task, either left-critical or right-critical, the rest of the BP, can be considered as a new BP. Invoking Lemma 3.4, the new BP starts at either the first critical task's deadline (if it is right-critical) or the arrival time of the next task after the first critical task (if it is left-critical). Applying Theorem 3.1 on the next BP, we are able to identify its first critical task, which is the second critical task of the original BP. Iteratively applying Theorem 3.1 helps us find *all* critical tasks on the original optimal sample path. This leads directly to an efficient algorithm which can identify all critical tasks in BP $\{k,\dots,n\}$ on the optimal sample path of **P1**. Meanwhile, as we have illustrated in Fig. 1, after identifying all critical tasks in BP $\{k,\dots,n\}$ on the optimal sample path of **P1**, we can find all segments in $\{k,\dots,n\}$ with the same energy function derivatives. Solving a $NE$ problem for each segment and combining the solutions gives us the optimal solution to $Q(k,n)$.

The *generalized critical task decomposition algorithm* (GCTDA), which identifies critical tasks and solves $Q(k,n)$ is as follows:

---

**step 1** $p=k$;
**step 2** $i=p+1$, Solve $NE(p,p;T_1(k,p),T_2(n,p))$
and $NE(p,p;T_1(k,p),d_p)$;
  Identify the first critical task in $(p,n)$
  while $(i\leq n)$
  {Solve $NE(p,i;T_1(k,p),T_2(n,i))$
  and $NE(p,i;T_1(k,p),d_i)$;
  Compute $R_i$;
  if $(\sigma_{p,i}(T_1(k,p),T_2(n,i))>\sigma_{p,R_i}(T_1(k,p),d_{R_i})$
   {$R_i$ is the first right-critical task in $(p,n)$;
   $\tau_j^*=\tau_j(T_1(k,p),d_{R_i})$, $j=p,\dots,R_i$;
   $x_{R_i^*}=d_{R_i}$;
   $a_j=d_{R_i}$, for all $j$, s.t., $j>R_i, a_j<d_{R_i}$;
   $p=R_i+1$; go to **step 2**;}
  Compute $L_i$;
  if $(\sigma_{p,i}(T_1(k,p),d_i)<\sigma_{p,L_i}(T_1(k,p),a_{L_i+1})$
    {$L_i$ is the first left-critical task in $(p,n)$;
    $\tau_j^*=\tau_j(T_1(k,p),a_{L_i+1})$, $j=p,\dots,L_i$;
    $x_{L_i}^*=a_{L_i+1}$;
    $p=L_i+1$; go to **step 2**;}
    $i=i+1$;
  }
   $\tau_j^*=\tau_j(T_1(k,p),d_n)$, $j=p,\dots,n$;
**END**

---

Note that GCTDA finds the critical tasks in a BP on the optimal sample path of **P1** iteratively. The optimal departure times of these critical tasks can be easily obtained using the results in Lemma 3.4. Finally, the optimal solution to the offline problem $\tau_j^*$, $j=1,\dots,N$ is also calculated in GCTDA.

Regarding the complexity of our algorithm, the most time-consuming part is solving $NE(i,j;t_1,t_2)$. In the worst case, the optimal sample path is a single BP containing $N-1$ critical tasks and the GCTDA algorithm may need to solve $NE(i,j;t_1,t_2)$ $2N_r$ times to identify each critical task, where $N_r$ is the number of tasks remaining. Therefore, the worst case complexity of the GCTDA algorithm is $O(N^2)$.

### C. Maximum Power Constraint

In **P1**, we omitted the constraint: $\tau_i \geq \tau_{i\_\min}$, which is essentially the maximum transmission rate or transmission power constraint for task $i$. This constraint is very important in real-world scenarios because a transmitter simply cannot transmit above the maximum transmission rate/power. We now formulate **P2**

$$\begin{aligned} \mathbf{P2}: \quad & \min_{\tau'_1,\ldots,\tau'_N} \sum_{i=1}^{N} v_i \omega_i(\tau'_i) \\ \text{s.t.} \quad & x'_i = \max\left(x'_{i-1}, a_i\right) + v_i \tau'_i \leq d_i, \quad i = 1,\ldots,N \\ & \tau'_i > \tau_{i\_\min},\ x_0 = 0. \end{aligned}$$

Notice that the only difference between **P1** and **P2** is the constraint on the control. We use $\tau'^{*}_i$ and $x'^{*}_i$ to denote the optimal control and optimal departure time of task $i$ in **P2**, respectively.

It is easy to show that Lemmas 3.1 and 3.2 also apply to **P2**. Similar to how we handled **P1**, we only need to consider a single BP $\{k,\ldots,n\}$ in the optimal sample path of **P2**. We formulate the following problem for BP $\{k,\ldots,n\}$:

$$\begin{aligned} Q'(k,n): \quad & \min_{\tau'_k,\ldots,\tau'_n} \sum_{i=k}^{n} v_i \omega_i(\tau'_i) \\ \text{s.t.} \quad & x'_i = a_k + \sum_{j=k}^{i} v_i \tau'_i \leq d_i, \quad i = k,\ldots,n \\ & \tau'_i \geq \tau_{i\_\min}, \quad i = k,\ldots,n \\ & x'_i \geq a_{i+1}, \quad i = k,\ldots,n-1. \end{aligned}$$

In order to establish the connection between Problems **P1** and **P2**, we now introduce the following assumption, which will be used to derive the results in this subsection.

*Assumption 2:* *a*) If $\tau_{i\_\min} < \tau_{j\_\min}$, then $\dot{\omega}_i(\tau_{i\_\min}) < \dot{\omega}_j(\tau_{j\_\min})$ and $\dot{\omega}_i(\tau) > \dot{\omega}_j(\tau)$; *b*) If $\tau_{i\_\min} \geq \tau_{j\_\min}$, then $\dot{\omega}_i(\tau_{i\_\min}) \geq \dot{\omega}_j(\tau_{j\_\min})$ and $\dot{\omega}_i(\tau) \leq \dot{\omega}_j(\tau)$.

Justification for this assumption can be found in the Appendix. Let

$$\tau_{\min} = \inf_{i=k,\ldots,n} \tau_{i\_\min}.$$

Under Assumption 2, we introduce the following auxiliary lemma:

*Lemma 3.9:* If $\exists \tau_i^* < \tau_{\min}$, then $Q'(k,n)$ is infeasible.

Lemma 3.9 establishes certain connections between the power-unconstrained problem $Q(k,n)$ and the power-constrained problem $Q'(k,n)$. When the problem is homogeneous, that is, the cost functions are identical among the tasks, we can easily derive that if $Q'(k,n)$ is feasible, then the optimal solution to $Q(k,n)$ must also yield the maximum power constraint. In the inhomogeneous case, however, it is possible that $Q(k,n)$ may return an optimal solution above the maximum power constraint while $Q'(k,n)$ is indeed feasible. When this occurs, the optimal solution to $Q'(k,n)$ would be close to $\tau_{i\_\min}$, and the controller could simply apply $\tau_{i\_\min}$ as the control since there is not much benefit to conduct optimization in this case.

TABLE I
DIFFERENT TASK DEADLINES AND IDENTICAL ENERGY FUNCTIONS

| | CPU time (sec) | Cost |
|---|---|---|
| GCTDA | 0.031 | 3.41919 |
| GCTDA_TL | 1.579 | 3.56494 |
| MoveRight | 54.704 | 3.43264 |

### D. Offline Performance Comparisons

Next, we will test the offline performance of the GCTDA algorithm. In this case, all task information, including arrival times, deadlines, and number of bits is known. For comparison purposes, we obtain numerical results for the following algorithms:

- **GCTDA**: Offline algorithm knowing all task information exactly and having full computational capability to solve $NE(i,j;t_1,t_2)$.
- **GCTDA_TL**: Offline algorithm knowing all exact task information and using pre-established tables to find an approximate solution to the nonlinear algebraic system $NE(i,j;t_1,t_2)$. The purpose of this algorithm is to reduce the computational overhead associated with solving $NE(i,j;t_1,t_2)$ at the cost of more energy consumption. Specifically, we pre-calculate the derivatives of 1000 $\tau$ values for each energy function $\omega_i(\tau)$ and save these data into tables. Using these tables and binary search, we find approximate solutions to $NE(i,j;t_1,t_2)$ in GCTDA.
- **MoveRight**: The algorithm proposed in [2]. It is an iterative algorithm that converges to the optimal solution. We choose it for performance comparison purposes because to the best of our knowledge, it is the only other algorithm available for solving problems with task-dependent cost functions.

In each experiment, in order to make the comparison fair, we use the same setting (i.e., same arrival times, deadlines, task sizes, and energy functions) for each algorithm. Note that what the "best" function solves in the MoveRight algorithm is actually a nonlinear system $NE(i,i+1;t_1,t_2)$. All experiments are done using a 1.8 GHz Athlon XP processor.

The setting of the first experiment in Table I is as follows: 500 tasks of Poisson arrivals with mean interarrival time 5 s, each task has its own deadlines (uniformly distributed between $[a_i+5, a_i+20]$ for task $i$), task sizes are different, and the energy functions are the same. The GCTDA algorithm outperforms the MoveRight algorithm in terms of CPU time by two orders of magnitude. Because the optimal sample path is likely to contain multiple BPs and the energy functions are identical, GCTDA is very fast. We terminated the MoveRight algorithm after 10 000 passes. It can be seen that the MoveRight algorithm did not converge at this point yet. (The cost is still higher than the optimal cost returned by GCTDA.) Another observation is that the solution of GCTDA_TL is a good approximation to the one of GCTDA. This makes GCTDA_TL a good candidate for on-line control. However, it can be seen that

TABLE II
DIFFERENT DEADLINES AND DIFFERENT ENERGY FUNCTIONS

| | CPU time (sec) | Cost |
|---|---|---|
| GCTDA | 12.516 | 8.87826 |
| GCTDA_TL | 2.469 | 8.98774 |
| MoveRight | 200.703 | 9.08324 |

TABLE III
IDENTICAL DEADLINES AND IDENTICAL ENERGY FUNCTIONS

| | CPU time (sec) | Cost |
|---|---|---|
| GCTDA | 0.593 | 0.0687325 |
| GCTDA_TL | 98.469 | 0.0688239 |
| MoveRight | 61.969 | 0.0837155 |

TABLE IV
IDENTICAL DEADLINES AND DIFFERENT ENERGY FUNCTIONS

| | CPU time (sec) | Cost |
|---|---|---|
| GCTDA | 48 | 0.1997 |
| GCTDA_TL | 11.594 | 0.200008 |
| MoveRight | 434.687 | 0.72739 |

GCTDA_TL takes longer than GCTDA when the energy functions are identical. The reason is that in this case, the nonlinear system $NE(i,j;t_1,t_2)$ becomes a linear system, which can be easily solved. So there is no benefit in using the table lookup approximation approach. However, when the energy functions are different, as we will see later, the approximation method does help.

In the next experiment for 500 tasks in Table II, we keep the same setting as above, except that we make the energy functions different for each task. We terminate the MoveRight algorithm after 100 passes. It can be seen that in this experiment, GCTDA_TL takes much less CPU time than GCTDA. Both of them are much faster (by an order of magnitude) and MoveRight has not yet converged.

In Table III, we make all 500 tasks have the same deadline and the same energy function. In this case, the optimal sample path contains a single BP. We terminate the MoveRight algorithm after 10 000 passes. It can be seen that at the time of termination, it was still far from converging to the optimal solution. Again, the CPU time of GCTDA_TL is higher than GCTDA, since the energy functions are identical.

In Table IV, the setting is the same as above, except that we now consider 100 tasks with different energy functions. We terminate the MoveRight algorithm after 1000 passes.

## IV. ONLINE CONTROLLER DESIGN

We proved that our offline algorithm GCTDA can return each critical task on the optimal sample path correctly. Therefore, by using it, we can obtain the offline optimal solution. We are also interested in designing good *online* controllers, in which case there are two difficulties: 1) lack of future task information and 2) high computational complexity in solving the nonlinear equations.

To overcome the first difficulty, we design a receding horizon (RH) controller assuming that at each decision point, the controller always has some task information within a given RH window, and nothing beyond this window. The size of the RH window $H$ can be measured either by time units or the number of tasks. In this paper, we use the latter to measure the RH window $H$. This RH window, together with the task information within it, is often referred to as the *planning horizon*. In contrast to the planning horizon, the RH controller will apply controls over an *action horizon*, which contains a subset of tasks over the planning horizon. Such controllers have been proposed and analyzed in [30] and [31] for the homogeneous case that the cost functions are identical. In this paper, we consider the RH control for the inhomogeneous case that the cost functions are *task dependent*.

As we will see later, the offline results we obtained in previous sections provide insight to RH online controller design and performance evaluation. We now introduce some notations similar to the ones in [31]. Let $\tilde{x}_t$ be the departure time of task $t$ on the RH state trajectory, which is also a decision point when the RH controller is invoked with look-ahead window $H$. Let $\tilde{\tau}_t$ be the control associated with task $t$ as determined by the RH controller. When task $t+1$ starts a new BP (i.e., $a_{t+1} > \tilde{x}_t$), then the RH controller does not need to act until $a_{t+1}$ rather than $\tilde{x}_t$; for notational simplicity, we will still use $\tilde{x}_t$ to represent the decision point for task $t+1$ (i.e., the time when the control $\tilde{\tau}_{t+1}$ is determined). Let $h$ denote the last task included in the window that starts at the current decision point $\tilde{x}_t$, i.e.,

$$h = \arg\max_{r \geq t} \{a_r : a_r \leq \tilde{x}_t + H\}.$$

Note that although the value of $h$ depends on $t$, for notational simplicity, we will omit this dependence and only write $h_t$ when it is necessary to indicate dependence on $t$. When the RH controller is invoked at $\tilde{x}_t$, it is called upon to determine $\tilde{\tau}_i$, the control associated with task $i$ for all $i = t+1, \ldots, h$, and let $\tilde{x}_i$ denote the corresponding departure time of task $i$ which is given by $\tilde{x}_i = \max(\tilde{x}_{i-1}, a_i) + \tilde{\tau}_i v_i$. The values of $\tilde{x}_i$ and $\tilde{\tau}_i$ are initially undefined, and are updated at each decision point $\tilde{x}_t$ for all $i = t+1, \ldots, h$. Control is applied to task $t+1$ only. That control and the corresponding departure time are the ones shown in the final RH sample path. In other words, for any given task $i$, $\tilde{x}_i$, and $\tilde{\tau}_i$ may vary over different planning horizons, since optimization is performed based on different available information. It is only when task $i$ is the next one at some decision point that its control and departure time become final.

Given these definitions, we are now ready to discuss the worst case estimation process to be used. If $h = N$, then the optimization process is finalized, so we will only consider the more interesting case when $h < N$. Then, our worst case estimation pertains to the characteristics of task $h+1$, the first one beyond the current planning horizon determined by $h$, that is, its arrival time, deadline, and number of bits which are unknown. We define task arrival times and task deadlines for $i = t+1, \ldots, h+1$ as follows:

$$\tilde{a}_i = \begin{cases} a_i, & \text{if } t+1 \leq i \leq h \\ \tilde{x}_t + H, & \text{if } i = h+1 \end{cases} \tag{11}$$

$$\tilde{d}_i = \begin{cases} d_i, & \text{if } t+1 \leq i \leq h \\ \tilde{a}_{h+1} + \tau_{i\_\min} v_{h+1}, & \text{if } i = h+1. \end{cases} \tag{12}$$

In (11), the arrival times of tasks $i = t+1, \ldots, h$ are known and we introduce a "worst case" estimate for the first unknown task beyond $\tilde{x}_t + H$, that is, we set it to be the earliest it could possibly occur. In (12), the deadlines of tasks $i = t+1, \ldots, h$ are known and we introduce a "worst case" estimate for the first unknown task's deadline to be the tightest possible, since $\tau_{i\_\min}$ is the minimum feasible time per bit. Note that $v_{h+1}$ is, in fact, unknown at time $\tilde{x}_t$, but we will see that this does not affect our optimization process as the value of $\tilde{d}_{h+1}$ is not actually required for analysis purposes. We point it out that we do not have to worry about estimates for the unknown tasks beyond $h+1$ (this is because of the FCFS nature of our system).

Therefore, the optimization problem that the RH controller faces at time $\tilde{x}_t$ is over tasks $t+1, \ldots, h$ with the additional constraint that they must all be completed by time $\tilde{a}_{h+1} = \tilde{x}_t + H$. This is equivalent to redefining $\tilde{d}_i$ as

$$\tilde{d}_i = \begin{cases} d_i, & \text{if } t+1 \le i \le h \\ \min(d_h, \tilde{a}_{h+1}), & \text{if } i = h. \end{cases} \tag{13}$$

Our online RH control problem at decision point $\tilde{x}_t$ will be denoted by $\tilde{Q}(t+1, h)$ and is formulated as follows:

$$\begin{aligned} \tilde{Q}(t+1,h): \quad & \min_{\tilde{\tau}_{t+1},\ldots,\tilde{\tau}_h} \sum_{i=t+1}^{h} v_i \omega_i(\tilde{\tau}_i) \\ \text{s.t.} \quad & \tilde{\tau}_i \ge 0, \quad i = t+1, \ldots, h \\ & \tilde{x}_i = \max(\tilde{x}_{i-1}, a_i) + \tilde{\tau}_i v_i \le \tilde{d}_i, \ \tilde{x}_t \text{ known} \end{aligned}$$

where $\tilde{d}_i$ is defined in (13). We also formulate the online RH control problem with the maximum power constraint

$$\begin{aligned} \tilde{Q}'(t+1,h): \quad & \min_{\tilde{\tau}_{t+1},\ldots,\tilde{\tau}_h} \sum_{i=t+1}^{h} v_i \omega_i(\tilde{\tau}_i) \\ \text{s.t.} \quad & \tilde{\tau}_i \ge \tau_{i\_\min}, \quad i = t+1, \ldots, h \\ & \tilde{x}_i = \max(\tilde{x}_{i-1}, a_i) + \tilde{\tau}_i v_i \le \tilde{d}_i, \ \tilde{x}_t \text{ known.} \end{aligned}$$

Similar to the RH problem in [31], $\tilde{Q}(t+1, h)$ may not be feasible even if the offline problem is feasible. This is due to the worst-case estimation. One way of relaxing the worst-case estimation is to use $\hat{h}$ (defined below), instead of $h$ in $\tilde{Q}$ above. Let

$$\begin{aligned} & \hat{x}_j = \max(\hat{x}_{j-1}, a_j) + \tau_{j\_\min} v_j \\ & \hat{x}_t = \tilde{x}_t, \ j = t+1, \ldots, h \\ & S = \{j : t+1 \le j < h \\ & \hat{x}_i \le \min(d_i, a_{j+1}) \text{ for all } i, \ t+1 \le i \le j\} \\ & \hat{h} = \begin{cases} \sup\ S, & \text{if } S \ne \varnothing \\ \infty, & \text{otherwise.} \end{cases} \end{aligned}$$

We then define $\hat{d}_i$

$$\hat{d}_j = \begin{cases} d_j, & j = t+1, \ldots, \hat{h}-1 \\ \min(d_j, \tilde{a}_{j+1}), & j = \hat{h} \end{cases} \tag{14}$$

TABLE V
RH CONTROL

| | |
|---|---|
| Step 1: | Solve $\tilde{Q}(t+1, h)$ and get $\tilde{\tau}_i^*, \tilde{x}_i^*$, |
| | $i = t+1, \ldots, h$ |
| | If $\tilde{\tau}_i^* \ge \tau_{i\_\min}$ for all $i$, |
| | apply $\tilde{\tau}_{t+1}^*$ to task $t+1$ and go to END. |
| Step 2: | If $\hat{h}$ exists, solve $\hat{Q}(t+1, \hat{h})$ and get $\tilde{\tau}_i^*, \tilde{x}_i^*$, |
| | $i = t+1, \ldots, \hat{h}$. |
| | If $\tilde{\tau}_i^* \ge \tau_{i\_\min}$ for all $i$, |
| | apply $\tilde{\tau}_{t+1}^*$ to task $t+1$ and go to END. |
| Step 3: | Apply $\tau_{t+1\_\min}$ to task $t+1$ |
| END | |

and formulate problem $\hat{Q}(t+1, \hat{h})$

$$\begin{aligned} \hat{Q}(t+1,\hat{h}): \quad & \min_{\tilde{\tau}_{t+1},\ldots,\tilde{\tau}_{\hat{h}}} \sum_{i=t+1}^{\hat{h}} v_i \omega_i(\tilde{\tau}_i) \\ \text{s.t.} \quad & \tilde{\tau}_i \ge 0, \quad i = t+1, \ldots, \hat{h} \\ & \tilde{x}_i = \max(\tilde{x}_{i-1}, a_i) + \tilde{\tau}_i v_i \le \hat{d}_i, \ \tilde{x}_t \text{ known.} \end{aligned}$$

The RH control algorithm at each decision point $\tilde{x}_t$ is shown in Table V. Note that $\tilde{Q}(t+1, h)$ and $\hat{Q}(t+1, \hat{h})$ essentially are smaller scale offline optimization problems. This implies that at each online decision point, we shall use an offline control algorithm, that is, GCTDA or GCTDA_TL, to solve $\tilde{Q}(t+1, h)$ and $\hat{Q}(t+1, \hat{h})$. In the next result, we discuss the feasibility of the proposed online RH control mechanism.

*Theorem 4.1:* If the offline problem **P2** is feasible, then the RH control in Table V is also feasible.

Theorem 4.1 reveals that the RH control in Table V guarantees feasibility when the offline problem **P2** is feasible. Next, we will analyze the performance of the proposed RH controller using simulation. To overcome high computational complexity, we use the GCTDA_TL algorithm, rather than GCTDA algorithm for online RH control. As mentioned previously, the GCTDA_TL algorithm uses some piecewise constant functions to approximate the derivatives of the energy functions at different $\tau$. Optimization can be approximated by searching efficiently in a pre-established table containing these functions.

In our simulation, all tasks have 512 B and a fixed deadline, that is, $d_i = a_i + d$. The value of $d$ is set to 10 s. In Figs. 2 and 3, we run the experiment 1000 times, and 500 tasks are executed in each run. The simulation is performed on a PC with a third-generation Intel Core i5-3570K Ivy Bridge 3.4GHz Quad-Core Desktop Processor. To quantify the deviation of the RH cost from the optimal offline cost, we define the cost difference as: (RH cost—optimal offline cost)/optimal offline cost. Figs. 2 and 3 plot the average cost difference, worst-case cost difference, best-case cost difference, and average calculation time versus the RH window size $H$ in seconds.

In Fig. 2, we consider Poisson arrivals with $\lambda = 0.2$. With RH window size $H$ varying from 1 to 11 s, the cost differences are well below 2%. When $H$ becomes larger, the cost differences are reduced; the calculation time per task increases since at each decision point, and the optimization problem involves more tasks.

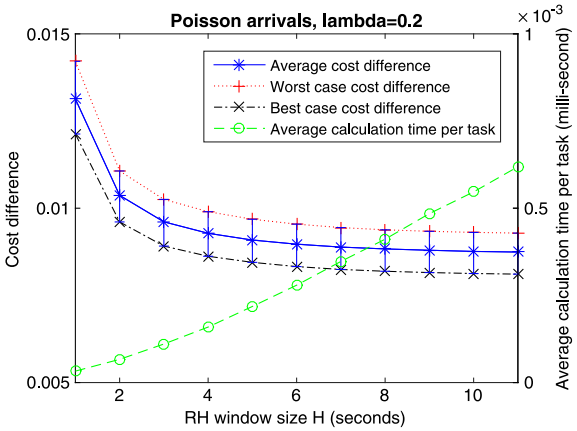


Fig. 2. Simulation results of Poisson and bursty arrivals.

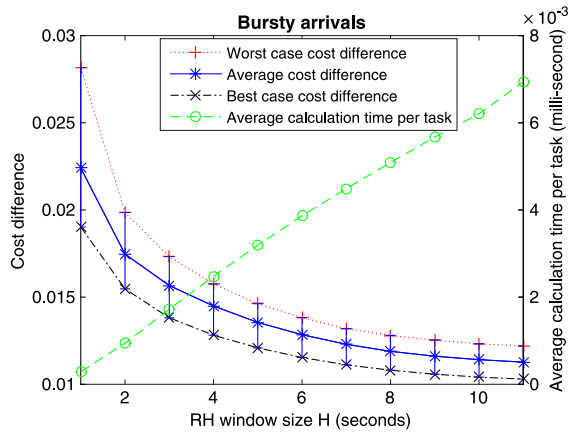


Fig. 3. Simulation results of bursty arrivals.

In the next experiment shown in Fig. 3, we consider bursty arrivals with the burst interval uniformly distributed over [8, 12] s, the number of tasks in each burst chosen from $\{10, \ldots, 20\}$ with equal probability, and the task intervals within the same burst uniformly distributed within [0, 1]. Although the cost differences are just slightly higher than those in the Poisson case, the average calculation time per task is now much larger. This is because in the bursty arrival case, a large number of backlogged tasks are involved at each decision point.

Our simulation results show that the proposed RH control mechanism not only guarantees feasibility when the offline problem is feasible, but also achieves near optimal solutions.

## V. Conclusion

In this paper, we first study the *downlink transmission scheduling* (DTS) problem. A simpler version of this problem has been studied in [2] and [9], where the MoveRight algorithm is proposed. The MoveRight algorithm is an iterative algorithm, and its rate of convergence is obtainable only when the cost function is not *task dependent*. Compared with the work in [2] and [9], we deal with a much harder problem: i) our cost function is task dependent and ii) each task has its own arrival time and deadline. This is essentially a hard convex optimization problem with nondifferentiable constraints. By analyzing the special structure of the optimal sample path, an efficient algorithm, known as the GCTDA, is proposed to solve the problem. Simulation results show that our algorithm is more appropriate for real-time applications than the MoveRight algorithm. Finally, we show that our results can be used in online control to achieve near optimal solutions for Poisson and bursty arrivals.

## Appendix

*Proof of Lemma 3.3:* Since $\omega_i(\tau)$ is strictly convex and differentiable

$$v_1\omega_1(\tau_1) - v_1\omega_1(\tau_1') > v_1(\tau_1 - \tau_1')\dot{\omega}_1(\tau_1') \quad (15)$$

$$v_2\omega_2(\tau_2) - v_2\omega_2(\tau_2') > v_2(\tau_2 - \tau_2')\dot{\omega}_2(\tau_2'). \quad (16)$$

Since $v_1\tau_1 + v_2\tau_2 = v_1\tau_1' + v_2\tau_2'$

$$v_1(\tau_1 - \tau_1') = -v_2(\tau_2 - \tau_2') = C > 0. \quad (17)$$

Summing (15) and (16) above, and using (17), we obtain

$$v_1\omega_1(\tau_1) + v_2\omega_2(\tau_2) - v_1\omega_1(\tau_1') - v_2\omega_2(\tau_2') > C(\dot{\omega}_1(\tau_1') - \dot{\omega}_2(\tau_2')).$$

Since $C > 0$ and by assumption $\dot{\omega}_1(\tau_1') - \dot{\omega}_2(\tau_2') > 0$

$$v_1\omega_1(\tau_1) + v_2\omega_2(\tau_2) > v_1\omega_1(\tau_1') + v_2\omega_2(\tau_2').$$

■

*Proof of Lemma 3.4:* We only prove part i). Part ii) can be proved similarly. Let $\tau_k^*, \ldots, \tau_n^*$ be the optimal solution. By definition of a left-critical task, we have $\dot{\omega}_i(\tau_i^*) > \dot{\omega}_{i+1}(\tau_{i+1}^*)$. Since tasks $i$ and $i+1$ are withina single BP, $x_i^* \geq a_{i+1}$. Suppose $x_i^* > a_{i+1}$. Consider a feasible solution $\tau_k', \ldots, \tau_n'$, s.t.

$$\tau_j' = \tau_j^*,\ j \neq i,\ j \neq i+1$$

$$\tau_i' < \tau_i^*,\ \tau_{i+1}' > \tau_{i+1}^*,\ \dot{\omega}_i(\tau_i') > \dot{\omega}_{i+1}(\tau_{i+1}').$$

Note that such a feasible solution always exists as long as $\tau_i'$ and $\tau_{i+1}'$ are arbitrarily close to $\tau_i^*$ and $\tau_{i+1}^*$, respectively. From Lemma 3.3, we obtain

$$v_i\omega_i(\tau_i^*) + v_{i+1}\omega_{i+1}(\tau_{i+1}^*) > v_i\omega_i(\tau_i') + v_{i+1}\omega_i(\tau_{i+1}').$$

Since $\tau_j' = \tau_j^*, j \neq i, j \neq i+1$, using the above inequality, we obtain

$$\sum_{i=k}^{n} v_i\omega_i(\tau_i^*) > \sum_{i=k}^{n} v_i\omega_i(\tau_i')$$

which contradicts the assumption that $\tau_k^*, \ldots, \tau_n^*$ is the optimal solution. Therefore, $x_i^* = a_{i+1}$. ■

*Proof of Lemma 3.5:* i) Since $\omega_m(\tau)$ is strictly convex, continuous, and differentiable, $\dot{\omega}_m(\tau)$ is also continuous. $NE(i, j; t_1, t_2)$ has a feasible solution, because $\dot{\omega}_m(\tau)$ is continuous and monotonically increasing, and $\tau$ can take any value in $(0, \infty)$. Now, suppose there are two different solutions to $NE(i, j; t_1, t_2)$ : $\tau_i(t_1, t_2), \ldots, \tau_j(t_1, t_2)$ and $\tau_i'(t_1, t_2), \ldots, \tau_j'(t_1, t_2)$.

Then, the common derivatives of these two solutions are different. Without loss of generality, we assume $\sigma_{i,j}(t_1,t_2) > \sigma'_{i,j}(t_1,t_2)$, which means $\dot{\omega}_m(\tau_m(t_1,t_2)) > \dot{\omega}_m(\tau'_m(t_1,t_2))$, for any $m$, $i \le m \le j$. By the convexity of $\omega_m(\tau)$, we obtain $\tau_m(t_1,t_2) > \tau'_m(t_1,t_2)$, for any $m$, $i \le m \le j$, and

$$\sum_{m=i}^{j} \tau_m(t_1,t_2)v_m = t_2 - t_1 > \sum_{m=i}^{j} \tau'_m(t_1,t_2)v_m = t_2 - t_1$$

which is a contradiction. Therefore, $NE(i,j;t_1,t_2)$ has a unique solution.

ii) Let $\sigma_{i,j}(t_1,t_2)$, $\sigma_{i,j}(t_3,t_4)$ be the common derivative of $NE(i,j;t_1,t_2)$ and $NE(i,j;t_3,t_4)$, respectively, $0 < \Delta = t_2 - t_1 < \Delta' = t_4 - t_3$. Let $\tau_i(t_1,t_2),\ldots,\tau_j(t_1,t_2)$, $\tau_i(t_3,t_4)$, $\ldots,\tau_j(t_3,t_4)$ be the solution to $NE(i,j;t_1,t_2)$ and $NE(i,j;t_3,t_4)$, respectively. We need to show $\sigma_{i,j}(t_1,t_2) < \sigma_{i,j}(t_3,t_4)$. Suppose $\sigma_{i,j}(t_1,t_2) \ge \sigma_{i,j}(t_3,t_4)$. Then, by definition, we obtain $\dot{\omega}_m(\tau_m(t_1,t_2)) \ge \dot{\omega}_m(\tau_m(t_3,t_4))$, for any $m$, $i \le m \le j$. By the convexity of $\omega_m(\tau)$, $\tau_m(t_1,t_2) \ge \tau_m(t_3,t_4)$, $i \le m \le j$. Therefore, $\sum_{m=i}^{j} \tau_m(t_1,t_2)v_m = t_2 - t_1 = \Delta \ge \sum_{m=i}^{j} \tau_m(t_3,t_4)v_m = t_4 - t_3 = \Delta'$ which contradicts our assumption that $\Delta < \Delta'$. Therefore, the common derivative of $NE(i,j;t_1,t_2)$ is a monotonically increasing function of $\Delta = t_2 - t_1$.

iii) It can be easily checked that $\tau_i(t_1,t_2),\ldots,\tau_p(t_1,t_2)$ and $\tau_{p+1}(t_1,t_2),\ldots,\tau_j(t_1,t_2)$ are the unique solutions to $NE(i,p;t_1,t_1+S_{ip})$ and $NE(p+1,j;t_1+S_{ip},t_2)$, respectively. Therefore, by the definition of $NE(i,j;t_1,t_2)$ and $\sigma_{i,j}(t_1,t_2)$, we have $\sigma_{i,p}(t_1,t_1+S_{ip}) = \sigma_{i,j}(t_1,t_2)$ and $\sigma_{p+1,j}(t_1+S_{ip},t_2) = \sigma_{i,j}(t_1,t_2)$, which implies $\sigma_{i,p}(t_1,t_1+S_{ip}) = \sigma_{p+1,j}(t_1+S_{ip},t_2) = \sigma_{i,j}(t_1,t_2)$.

iv) Let $S_{ip}$ be the same as in iii). By assumption, $c_q \ne c_r \ \forall\, q, r \in \{1,2,3\}, q \ne r$. This implies that $t_3 \ne S_{ip}$; otherwise, $c_q = c_r$. From the monotonicity of $\sigma_{i,j}(t_1,t_2)$ shown in part ii), we obtain $\min(c_1,c_2) < c_3 < \max(c_1,c_2)$. ■

*Proof of Lemma 3.6:* Invoking Lemma 3.4, we obtain $x_r^* = d_r$. Suppose there are left-critical tasks in $\{k,\ldots,r-1\}$ and the closest left-critical task to $r$ is task $l$, $k \le l < r$. Invoking Lemma 3.4, $x_l^* = a_{l+1}$. By assumption

$$\sigma_{k,L_r}(a_k, a_{L_r+1}) \le \sigma_{k,r}(a_k,d_r). \tag{18}$$

Because $l < r$, from (7)

$$\sigma_{k,l}(a_k,a_{l+1}) \le \sigma_{k,L_r}(a_k,a_{L_r+1}). \tag{19}$$

From (18) and (19), the following must be true:

$$\sigma_{k,l}(a_k,a_{l+1}) \le \sigma_{k,r}(a_k,d_r). \tag{20}$$

When the equality holds in (20), from iii) of Lemma 3.5, we obtain

$$\sigma_{l+1,r}(a_{l+1},d_r) = \sigma_{k,l}(a_k,a_{l+1}). \tag{21}$$

When the inequality holds in (20), from iv) of Lemma 3.5, we obtain

$$\sigma_{l+1,r}(a_{l+1},d_r) > \sigma_{k,r}(a_k,d_r).$$

From (20) and the above inequality, we get

$$\sigma_{l+1,r}(a_{l+1},d_r) > \sigma_{k,l}(a_k,a_{l+1}). \tag{22}$$

Combining (21) and (22), we obtain

$$\sigma_{l+1,r}(a_{l+1},d_r) \ge \sigma_{k,l}(a_k,a_{l+1}). \tag{23}$$

Since there is no right-critical or left-critical task in $\{l+1,\ldots,r-1\}$, invoking Lemma 3.1, we get $\dot{\omega}_s(\tau_s^*) = \dot{\omega}_{s+1}(\tau_{s+1}^*) = \sigma_{l+1,r}(a_{l+1},d_r),\ \forall\, s \in \{l+1,\ldots,r-1\}$. From the definition of left-critical tasks, we get

$$\dot{\omega}_l(\tau_l^*) > \dot{\omega}_{l+1}\left(\tau_{l+1}^*\right) = \sigma_{l+1,r}(a_{l+1},d_r). \tag{24}$$

We consider two cases:

Case 1: $k = l$. Then, $\sigma_{k,l}(a_k,a_{l+1}) = \dot{\omega}_l(\tau_l^*)$. Inequalities (23) and (24) contradict each other.

Case 2: $k < l$. We will use a contradiction argument to show that there must exist a right-critical task $m$, $k \le m < l$, that is, $\dot{\omega}_m(\tau_m^*) < \dot{\omega}_{m+1}(\tau_{m+1}^*)$. Suppose such a task $m$ does not exist, i.e.,

$$\dot{\omega}_m(\tau_m^*) \ge \dot{\omega}_{m+1}\left(\tau_{m+1}^*\right),\ k \le m < l. \tag{25}$$

Let

$$y_m = \begin{cases} a_k, & m = k-1 \\ x_m^*, & m = k,\ldots,l. \end{cases}$$

Inequality (25) is equivalent to the following equation:

$$\sigma_{m,m}(y_{m-1},y_m) \ge \sigma_{m+1,m+1}(y_m,y_{m+1}), \quad \text{for } m = k,\ldots,l-1. \tag{26}$$

We will use a recursive proof next:

Step 1: Letting $m = k$ in (26), we have

$$\sigma_{k,k}(y_{k-1},y_k) \ge \sigma_{k+1,k+1}(y_k,y_{k+1}). \tag{27}$$

When the equality holds in (27), invoking part iii) of Lemma 3.5, we get

$$\sigma_{k,k+1}(y_{k-1},y_{k+1}) = \sigma_{k+1,k+1}(y_k,y_{k+1}). \tag{28}$$

When the inequality holds in (27), invoking part iv) of Lemma 3.5, we get

$$\sigma_{k,k+1}(y_{k-1},y_{k+1}) > \sigma_{k+1,k+1}(y_k,y_{k+1}). \tag{29}$$

Combining (28) and (29), we have

$$\sigma_{k,k+1}(y_{k-1},y_{k+1}) \ge \sigma_{k+1,k+1}(y_k,y_{k+1}). \tag{30}$$

Step 2: Letting $m = k+1$ in (26), we have

$$\sigma_{k+1,k+1}(y_k,y_{k+1}) \ge \sigma_{k+2,k+2}(y_{k+1},y_{k+2}).$$

Combining (30) and the above inequality, we obtain

$$\sigma_{k,k+1}(y_{k-1},y_{k+1}) \ge \sigma_{k+2,k+2}(y_{k+1},y_{k+2}).$$

Similar to the derivation of (28)–(30), we can obtain

$$\sigma_{k,k+2}(y_{k-1},y_{k+2}) \ge \sigma_{k+2,k+2}(y_{k+1},y_{k+2}).$$

Repeating the process up to step $l-k$, we obtain

$$\sigma_{k,l}(y_{k-1}, y_l) \geq \sigma_{l,l}(y_{l-1}, y_l).$$

Since task $l$ is left-critical, from the definition of $y_m$ and Lemma 3.4, $y_l = x_l^* = a_{l+1}$, and the above inequality is equivalent to

$$\sigma_{k,l}(a_k, a_{l+1}) \geq \dot{\omega}_l(\tau_l^*). \tag{31}$$

From (31) and (24), we get

$$\sigma_{k,l}(a_k, a_{l+1}) > \sigma_{l+1,r}(a_{l+1}, d_r).$$

Invoking iv) of Lemma 3.5, we obtain

$$\sigma_{k,l}(a_k, a_{l+1}) > \sigma_{k,r}(a_k, d_r)$$

which contradicts (20). Therefore, there must exist a task $m$, $k \leq m < l$, s.t., $\dot{\omega}_m(\tau_m^*) < \dot{\omega}_{m+1}(\tau_{m+1}^*)$. By Definition 1, task $m$ is a right-critical task, which contradicts our assumption that task $r$ is the *first* right-critical task in $\{k, \ldots, n\}$. Therefore, there is no left-critical task before task $r$. ■

*Proof of Lemma 3.7:* We use a contradiction argument to prove the lemma. Suppose there are right-critical tasks before task $R_i$ and the one with the smallest index is task $r$, $k \leq r < R_i$. By assumption, $\sigma_{k,j}(a_k, d_j) \geq \sigma_{k,L_j}(a_k, a_{L_j+1})$, for all $j$, $k < j < i$. When $r > k$, letting $j = r$, we have $\sigma_{k,r}(a_k, d_r) \geq \sigma_{k,L_r}(a_k, a_{L_r+1})$. Then we can invoke Lemma 3.6 to establish that there is no left-critical task in $\{k, \ldots, r-1\}$. Since there is also no right-critical task in $\{k, \ldots, r-1\}$, from Lemma 3.1

$$\dot{\omega}_s(\tau_s^*) = \sigma_{k,r}(a_k, d_r),\ \forall\, s \in \{k, \ldots, r\}. \tag{32}$$

Since $k \leq r < R_i$, from (6), we have

$$\sigma_{k,R_i}(a_k, d_{R_i}) \leq \sigma_{k,r}(a_k, d_r). \tag{33}$$

Then, from ii) of Lemma 3.5

$$\sigma_{k,R_i}\left(a_k, x_{R_i}^*\right) \leq \sigma_{k,R_i}(a_k, d_{R_i}). \tag{34}$$

Combining (33) and (34), we get

$$\sigma_{k,R_i}\left(a_k, x_{R_i}^*\right) \leq \sigma_{k,r}(a_k, d_r). \tag{35}$$

Since $r$ is right-critical, from (32) and Definition 1

$$\sigma_{k,r}(a_k, d_r) = \dot{\omega}_r(\tau_r^*) < \dot{\omega}_{r+1}\left(\tau_{r+1}^*\right). \tag{36}$$

From (35) and (36), there must exist at least one left-critical task in $\{r+1, \ldots, R_i - 1\}$; otherwise, from the definition of a left-critical task in Definition 1 and a simple contradiction argument, we have $\dot{\omega}_s(\tau_s^*) \leq \dot{\omega}_{s+1}(\tau_{s+1}^*), \forall\, s \in \{r+1, \ldots, R_i - 1\}$. Using this result, (36) and a similar method as in obtaining (31), we can get $\sigma_{k,r}(a_k, d_r) < \sigma_{k,R_i}(a_k, x_{R_i}^*)$, which contradicts (35).

Let the left-critical task with the smallest index be $l$. From Lemma 3.4, $x_l^* = a_{l+1}$. Similar to obtaining $\sigma_{k,r}(a_k, d_r) < \sigma_{k,R_i}(a_k, x_{R_i}^*)$, we can obtain

$$\sigma_{k,r}(a_k, d_r) < \sigma_{k,l}(a_k, x_l^*) = \sigma_{k,l}(a_k, a_{l+1}). \tag{37}$$

By assumption, $\sigma_{k,j}(a_k, a_{j+1}) \leq \sigma_{k,R_j}(a_k, d_{R_j})$, and setting $j = l$, we get $\sigma_{k,l}(a_k, a_{l+1}) \leq \sigma_{k,R_l}(a_k, d_{R_l})$. Since, from (6), we have $\sigma_{k,R_l}(a_k, d_{R_l}) \leq \sigma_{k,r}(a_k, d_r)$, combining this and the above inequality, we obtain $\sigma_{k,l}(a_k, a_{l+1}) \leq \sigma_{k,r}(a_k, d_r)$ which contradicts (37) and completes the proof. ■

*Proof of Lemma 3.8:* Suppose task $R_i$ is not right-critical. By assumption, $\sigma_{k,R_i}(a_k, d_{R_i}) < \sigma_{k,i}(a_k, a_{i+1})$. Since $x_{R_i}^* \leq d_{R_i}$, from ii) of Lemma 3.5

$$\sigma_{k,R_i}\left(a_k, x_{R_i}^*\right) \leq \sigma_{k,R_i}(a_k, d_{R_i}). \tag{38}$$

From the two inequalities above, we obtain

$$\sigma_{k,R_i}\left(a_k, x_{R_i}^*\right) < \sigma_{k,i}(a_k, a_{i+1}). \tag{39}$$

Invoking Lemma 3.7, there is no right-critical task before task $R_i$. Next, we use a contradiction argument to show that there must exist at least one right-critical task in $\{R_i, \ldots, i-1\}$. Suppose there is no right-critical task in $\{R_i, \ldots, i-1\}$. Because there is no right-critical task in $\{k, \ldots, i-1\}$, by Definition 1, we have $\dot{\omega}_m(\tau_m^*) \geq \dot{\omega}_{m+1}(\tau_{m+1}^*)$, $m = k, \ldots, i-1$. Let

$$y_m = \begin{cases} a_k, & m = k-1 \\ x_m^*, & m = k, \ldots, i-1 \\ a_{i+1}, & m = i. \end{cases}$$

The above inequality can be rewritten as $\sigma_{m,m}(y_{m-1}, y_m) \geq \sigma_{m+1,m+1}(y_m, y_{m+1})$, $m = k, \ldots, i-1$. Similar to the way of obtaining (31) in proving Lemma 3.6, we obtain $\sigma_{k,R_i}(a_k, x_{R_i}^*) \geq \sigma_{k,i}(a_k, a_{i+1})$, which contradicts (39). We showed there must exist at least one right-critical task in $\{R_i, \ldots, i-1\}$. From the initial contradiction assumption, $R_i$ is not right-critical. When $i = R_i + 1$, the contradiction proof is completed. Next, we consider the case when $i > R_i + 1$. Let $r$ be the closest right-critical task to $R_i$ in $\{R_i, \ldots, i-1\}$. Since there is no right-critical task in $\{k, \ldots, r-1\}$, by Definition 1, $\dot{\omega}_m(\tau_m^*) \geq \dot{\omega}_{m+1}(\tau_{m+1}^*)$, $m = k, \ldots, r-1$. Let

$$y_m = \begin{cases} a_k, & m = k-1 \\ x_m^*, & m = k, \ldots, r. \end{cases}$$

The above inequality can be rewritten as

$$\sigma_{m,m}(y_{m-1}, y_m) \geq \sigma_{m+1,m+1}(y_m, y_{m+1}), m = k, \ldots, r-1.$$

Similar to the way of obtaining (31) in proving Lemma 3.6, we obtain

$$\sigma_{k,R_i}\left(a_k, x_{R_i}^*\right) \geq \sigma_{k,r}(a_k, x_r^*) = \sigma_{k,r}(a_k, d_r).$$

From the above inequality and (38), we obtain

$$\sigma_{k,r}(a_k, d_r) \leq \sigma_{k,R_i}(a_k, d_{R_i})$$

which contradicts the definition of $R_i$ in (4), since $R_i < r < i$. Therefore, task $R_i$ must be right-critical. ■

*Proof of Theorem 3.1:* We only prove part i). Part ii) can be proven similarly. The proof contains several steps: 1) using (8), Lemma 3.6, and setting $j = R_i$, we conclude that there is no left-critical task before $R_i$; 2) using (8), (9), and Lemma 3.7,

we establish that there is no right-critical task before $R_i$; 3) using (8)–(10), and Lemma 3.8, it follows that $R_i$ is a right-critical task; and, finally, 4) combining the results established in the previous steps 1)–3), we can obtain that $R_i$ is the first critical task in $\{k,\ldots,n\}$, and it is right-critical. ■

*Justifications for Assumption 2:* We only justify Part a). Part b) can be justified similarly. Using Shannon's theorem, $\tau_i$ can be represented by the following equation: $\tau_i = 1/(B\log_2(1+(s_iP/N_0)))$, where $B$ is the bandwidth of the channel, $s_i$ is the *task-dependent* channel gain, $P$ is the transmission power, and $N_0$ is the power of the noise. Since $s_iP/N_0 \gg 1$ in typical scenarios, we can omit the 1 above and represent $P$ in terms of $\tau_i$

$$P(\tau_i) = \frac{N_0\left(2^{\frac{1}{B\tau_i}}\right)}{s_i}. \tag{40}$$

We assume that the maximum transmission power of each task is constant $P_{\max}$, and it determines $\tau_{i\_\min}$

$$P_{\max} = \frac{N_0\left(2^{\frac{1}{B\tau_{i\_\min}}}\right)}{s_i}. \tag{41}$$

Because $\omega_i(\tau_i) = P(\tau_i)\tau_i$, we use (40) and (41) to get

$$\dot{\omega}_i(\tau_i) = P(\tau_i)\left(1-\frac{1}{B\tau_i}\right) = \frac{N_0\left(2^{\frac{1}{B\tau_i}}\right)}{s_i}\left(1-\frac{1}{B\tau_i}\right) \tag{42}$$

$$\dot{\omega}_i(\tau_i)|_{\tau_i=\tau_{i\_\min}} = P_{\max}\left(1-\frac{1}{B\tau_{i\_\min}}\right). \tag{43}$$

Using (43) and $\tau_{i\_\min} < \tau_{j\_\min}$, we have $\dot{\omega}_i(\tau_{i\_\min}) < \dot{\omega}_j(\tau_{j\_\min})$. Using (41) and $\tau_{i\_\min} < \tau_{j\_\min}$, we obtain $s_i > s_j$, that is, the channel gain of task $i$ is greater than that of task $j$. Finally, using (42), we get $\dot{\omega}_i(\tau) > \dot{\omega}_j(\tau)$. ■

*Proof of Lemma 3.9:* Let us assume that there exists tasks $\{p,\ldots,q\}$ $(k<p\le q<n)$ in a BP $\{k,\ldots,n\}$ of the optimal sample path of $Q(k,n)$ such that

$$\tau^*_{p-1} \ge \tau_{p-1\_\min},\ \tau^*_{q+1} \ge \tau_{q+1\_\min}$$
$$\tau^*_i < \tau_{\min},\ i=p,\ldots,q. \tag{44}$$

From Assumption 1

$$\dot{\omega}_{p-1}\left(\tau^*_{p-1}\right) \ge \dot{\omega}_{p-1}(\tau_{p-1\_\min})$$
$$\dot{\omega}_{q+1}\left(\tau^*_{q+1}\right) \ge \dot{\omega}_{q+1}(\tau_{q+1\_\min}). \tag{45}$$

Let $z = \arg\min_{i=k,\ldots,n}\tau_{i\_\min}$. Because $\tau_{i\_\min} \ge \tau_{z\_\min}$, we invoke Assumption 2 and get

$$\dot{\omega}_i(\tau_{\min}) \le \dot{\omega}_z(\tau_{\min}),\ i=p,\ldots,q. \tag{46}$$

From Assumption 1 and (44), we have

$$\dot{\omega}_i\left(\tau^*_i\right) < \dot{\omega}_i(\tau_{\min}),\ i=p,\ldots,q. \tag{47}$$

Combining (46) and (47) above, we have

$$\dot{\omega}_i\left(\tau^*_i\right) < \dot{\omega}_z(\tau_{\min}),\ i=p,\ldots,q. \tag{48}$$

Because $\tau_{p-1\_\min} \ge \tau_{\min}$, we invoke Assumption 2 and get

$$\dot{\omega}_{p-1}(\tau_{p-1\_\min}) \ge \dot{\omega}_z(\tau_{\min}). \tag{49}$$

Similarly, we use $\tau_{q+1\_\min} \ge \tau_{\min}$ and Assumption 2 to get

$$\dot{\omega}_{q+1}(\tau_{q+1\_\min}) \ge \dot{\omega}_z(\tau_{\min}). \tag{50}$$

Combining (45), (49), and (50), we have

$$\dot{\omega}_{p-1}\left(\tau^*_{p-1}\right) \ge \dot{\omega}_z(\tau_{\min}) \text{ and } \dot{\omega}_{q+1}\left(\tau^*_{q+1}\right) \ge \dot{\omega}_z(\tau_{\min}). \tag{51}$$

Then, we combine (48) and (51) to get $\dot{\omega}_{p-1}(\tau^*_{p-1}) > \dot{\omega}_i(\tau^*_i)$ and $\dot{\omega}_i(\tau^*_i) < \dot{\omega}_{q+1}(\tau^*_{q+1})$, $i=p,\ldots,q$. From these two inequalities, we have $\dot{\omega}_{p-1}(\tau^*_{p-1}) > \dot{\omega}_p(\tau^*_p)$ and $\dot{\omega}_q(\tau^*_q) < \dot{\omega}_{q+1}(\tau^*_{q+1})$. Invoking Lemma 3.4, we have

$$x^*_{p-1} = a_p \text{ and } x^*_q = d_q. \tag{52}$$

Because $x^*_q = x^*_{p-1} + \sum_{i=p}^q v_i\tau^*_i$, we use (44) and (52) to get $x^*_q - x^*_{p-1} = d_q - a_p = \sum_{i=p}^q v_i\tau^*_i < \sum_{i=p}^q v_i\tau_{\min} \le \sum_{i=p}^q v_i\tau_{i\_\min}$. For any feasible solution of $Q'(k,n)$, we must have $\sum_{i=p}^q v_i\tau'_i \le d_q - a_p \le \sum_{i=p}^q v_i\tau_{i\_\min}$, which implies that there exists at least one $\tau'_i$, $i=p,\ldots,q$, such that $\tau'_i < \tau_{i\_\min}$. This implies that $Q'(k,n)$ is infeasible. ■

*Proof of Theorem 4.1:* Because **P2** is feasible, we have $x'^*_i \le d_i$. To prove the theorem, we only need to show that $\tilde{x}_i \le x'^*_i$ for $i=1,\ldots,N$. We use induction to prove it.

1) When $t=0$, we are at the first decision point $\tilde{x}_0 = a_1$. We have two cases:

Case 1.1) We apply $\tau_{1\_\min}$ to task 1. It is obvious that $\tilde{x}_1 \le x'^*_i$.

Case 1.2) We apply control $\widetilde{\tau}^*_1$ obtained in either Step 1 or Step 2 of Table V to task 1. Without loss of generality, let us assume that $\widetilde{\tau}^*_1$ is obtained from Step 1 of Table V. In this case, $\widetilde{\tau}^*_i$, $i=1,\ldots,h$, are the solution to $\tilde{Q}(1,h)$ and $\widetilde{\tau}^*_i \ge \tau_{i\_\min}$ for all $i$. Therefore, $\widetilde{\tau}^*_i$ is also the solution to problem $\tilde{Q}'(1,h)$. We have two subcases:

Case 1.2.1) When the planning horizon contains the end of a busy period on the optimal sample path, it is trivial that $\widetilde{\tau}^*_1 = \tau'^*_1$. Therefore, $\tilde{x}_1 = x'^*_1 \le d_1$.

Case 1.2.2) We now consider the more interesting case that $d_i \ge \widetilde{a}_{i+1}$, $i=1,\ldots,h$. To compare the RH problem and the offline problem, we now add subscripts to indicate the starting and ending times of each problem. In particular, we use $\tilde{Q}'_{a_1,a_1+H}(1,h)$ to show that the starting transmission time of $\tilde{Q}'(1,h)$ is $a_1$ and the ending transmission time is $a_1+H$. Similarly, we use $Q'_{a_1,x'^*_h}(1,h)$ to show that the starting transmission time of $Q'(1,h)$ is $a_1$ and the ending transmission time is $x'^*_h$. Because $a_{h+1} > a_1+H$ and $d_h \ge a_1+H$, we must have $x'^*_h \ge a_1+H$. Looking at $\tilde{Q}'_{a_1,a_1+H}(1,h)$

and $Q'_{a_1,x'^*_h}(1,h)$, they are exactly the same, except that the ending transmission time of $Q'_{a_1,x'^*_h}(1,h)$ is potentially at a later time. Therefore, the optimal departure time of any task in $Q'_{a_1,x'^*_h}(1,h)$ must not be earlier than that in $\tilde{Q}'_{a_1,a_1+H}(1,h)$, which means $\tilde{x}_1 \le x'^*_1$.

2) Suppose that the RH controller is at decision point $\tilde{x}_t$, and $\tilde{x}_t \le x'^*_t$. We also have two cases:

Case 2.1) We apply $\tau_{t+1\,\min}$ to task $t+1$. It is obvious that $\tilde{x}_{t+1} \le x'^*_{t+1}$.

Case 2.2) We apply control $\widetilde{\tau}^*_{t+1}$ obtained in either Step 1 or Step 2 of Table V to task 1. Without loss of generality, let us assume that $\widetilde{\tau}^*_{t+1}$ is obtained from Step 1 of Table V. In this case, $\widetilde{\tau}^*_i$, $i = t+1,\ldots,h$, are the solution to $\tilde{Q}(t+1,h)$ and $\widetilde{\tau}^*_i \ge \tau_{i\_\min}$ for all $i$. Therefore, $\widetilde{\tau}^*_i$ is also the solution to problem $\tilde{Q}'(t+1,h)$. We consider two subcases:

Case 2.2.1) When the planning horizon contains the end of a busy period on the optimal sample path, that is, there exists task $j \epsilon \{t+1,\ldots,h\}$ s.t. $d_j < \widetilde{a}_{j+1}$, we focus on tasks $\{t+1,\ldots,j\}$. In this case, the controls of these tasks on the RH sample path are the solutions to problem $\tilde{Q}'_{\tilde{x}_t,d_j}(t+1,j)$, and the controls of these tasks on the optimal sample path of **P2** are the solutions to problem $Q'_{x'^*_t,d_j}(t+1,j)$. Looking at $\tilde{Q}'_{\tilde{x}_t,d_j}(t+1,j)$ and $Q'_{x'^*_t,d_j}(t+1,j)$, they are identical, except that the starting transmission time of $\tilde{Q}'_{\tilde{x}_t,d_j}(t+1,j)$ is potentially earlier than that of $Q'_{x'^*_t,d_j}(t+1,j)$. Therefore, the optimal departure time of any task in $\tilde{Q}'_{\tilde{x}_t,d_j}(t+1,j)$ must be no later than that in $Q'_{x'^*_t,d_j}(t+1,j)$, which means $\tilde{x}_{t+1} \le x'^*_{t+1}$.

Case 2.2.2) $d_i \ge \widetilde{a}_{i+1}$, $i = t+1,\ldots,h$. In this case, the controls of tasks $\{t+1,\ldots,h\}$ on the RH sample path are the solutions to problem $\tilde{Q}'_{\tilde{x}_t,\tilde{x}_t+H}(t+1,h)$, and the controls of these tasks on the optimal sample path of **P2** are the solutions to problem $Q'_{x'^*_t,x'^*_h}(t+1,h)$. Because $a_{h+1} > \tilde{x}_t + H$ and $d_h \ge \tilde{x}_t + H$, we must have $x'^*_h \ge \tilde{x}_t + H$. Looking at $\tilde{Q}'_{\tilde{x}_t,\tilde{x}_t+H}(t+1,h)$ and $Q'_{x'^*_t,x'^*_h}(t+1,h)$, they are exactly the same, except that the starting and ending transmission times of $\tilde{Q}'_{\tilde{x}_t,\tilde{x}_t+H}(t+1,h)$ are potentially sooner than those of $Q'_{x'^*_t,x'^*_h}(t+1,h)$. Therefore, the optimal departure time of any task in $\tilde{Q}'_{\tilde{x}_t,\tilde{x}_t+H}(t+1,h)$ must be no later than that in $Q'_{x'^*_t,x'^*_h}(t+1,h)$, which means $\tilde{x}_{t+1} \le x'^*_{t+1}$. ■

## REFERENCES


[1] T. D. Burd, T. A. Pering, A. J. Stratakos, and R. W. Brodersen, "A dynamic voltage scaled microprocessor system," *IEEE J. Solid-State Circuits*, vol. 35, no. 11, pp. 1571–1580, Nov. 2000.

[2] A. E. Gamal, C. Nair, B. Prabhakar, E. Uysal-Biyikoglu, and S. Zahedi, "Energy-efficient scheduling of packet transmissions over wireless networks," in *Proc. IEEE INFOCOM*, New York, USA, 2002, vol. 3, no. 23–27, pp. 1773–1782.

[3] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana, IL, USA: University of Illinois Press, 1949.

[4] B. E. Collins and R. L. Cruz, "Transmission policies for time varying channels with average delay constraints," presented at the Allerton Conf. Commun., Control, Comput., Monticello, IL, USA, 1999.

[5] R. Berry, "Power and delay trade-offs in fading channels," Ph.D. dissertation, Mass. Inst. Technol., Cambridge, MA, USA, 2000.

[6] B. Ata, "Dynamic power control in a wireless static channel subject to a quality of service constraint," *Oper. Res.*, vol. 53, 2005.

[7] B. Ata and K. E. Zachariadis, "Dynamic power control in a fading downlink channel subject to an energy constraint," *Queueing Syst.*, vol. 55, no. 1, pp. 41–69, Jan. 2007.

[8] M. Neely, "Energy optimal control for time varying wireless networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 7, pp. 2915–2934, Jul. 2006.

[9] E. Uysal-Biyikoglu, B. Prabhakar, and A. E. Gamal, "Energy-efficient packet transmission over a wireless link," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 487–499, Aug. 2002.

[10] M. Zafer and E. Modiano, "Minimum energy transmission over a wireless channel with deadline and power constraints," *IEEE Trans. Autom. Control*, vol. 54, no. 12, pp. 2841–2852, Dec. 2009.

[11] M. Zafer and E. Modiano, "Delay-constrained energy efficient data transmission over a wireless fading channel," presented at the IEEE Inf. Theory Appl. Workshop, San Diego, CA, USA, Jan./Feb. 2007.

[12] M. Zafer and E. Modiano, "Optimal rate control for delay-constrained data transmission over a wireless channel," *IEEE Trans. Inf. Theory*, vol. 54, no. 9, pp. 4020–4039, Sep. 2008.

[13] W. Chen, M. Neely, and U. Mitra, "Energy efficient scheduling with individual packet delay constraints: Offline and online results," presented at the IEEE Infocom, Anchorage, AK, USA, May 2007.

[14] W. Chen, U. Mitra, and M. Neely, "Energy-efficient scheduling with individual packet delay constraints over a fading channel," *ACM Wireless Netw.*, vol. 15, pp. 601–618, 2009.

[15] M. Zafer and E. Modiano, "A calculus approach to energy-efficient data transmission with quality-of-service constraints," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 898–911, Jun. 2009.

[16] X. Wang and Z. Li, "Energy-efficient transmissions of bursty data packets with strict deadlines over time-varying wireless channels," *IEEE Trans. Wireless Commun.*, vol. 12, no. 5, pp. 2533–2543, May 2013.

[17] M. I. Poulakis, A. D. Panagopoulos, and P. Canstantinou, "Channel-aware opportunistic transmission scheduling for energy-efficient wireless links," *IEEE Trans. Veh. Technol.*, vol. 62, no. 1, pp. 192–204, Jan. 2013.

[18] Z. Zhou, P. Soldati, H. Zhang, and M. Johansson, "Energy-efficient deadline-constrained maximum reliability forwarding in lossy networks," *IEEE Trans. Wireless Commun.*, vol. 11, no. 10, pp. 3474–3483, Oct. 2012.

[19] F. Shan, J. Luo, W. Wu, M. Li, and X. Shen, "Discrete rate scheduling for packets with individual deadlines in energy harvesting systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 3, pp. 438–451, Mar. 2015.

[20] B. Tomasi and J. C. Preisig, "Energy-efficient transmission strategies for delay constrained traffic with limited feedback," *IEEE Trans. Wireless Commun.*, vol. 14, no. 3, pp. 1369–1379, Mar. 2015.

[21] X. Zhong and C. Xu, "Online energy efficient packet scheduling with delay constraints in wireless networks," presented at the IEEE Infocom, Phoenix, AZ, USA, Apr. 2008.

[22] R. Vaze, "Competitive ratio analysis of online algorithms to minimize packet transmission time in energy harvesting communication system," presented at the IEEE Infocom, Turin, Italy, Apr. 2013.

[23] L. Miao and C. G. Cassandras, "Optimal transmission scheduling for energy-efficient wireless networks," in *Proc. IEEE INFOCOM*, Barcelona, Spain, 2006, pp. 1–11.

[24] A. J. Goldsmith, "The capacity of downlink fading channels with variable rate and power," *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 569–580, Aug. 1997.
[25] R. R. Kompella and A. C. Snoeren, "Practical lazy scheduling in sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sens. Syst.*, Los Angeles, CA, USA, 2003, pp. 280–291.
[26] Y. C. Cho, C. G. Cassandras, and D. L. Pepyne, "Forward decomposition algorithms for optimal control of a class of hybrid systems," *Int. J. Robust Nonlinear Control*, vol. 11, no. 5, pp. 497–513, 2001.
[27] P. Zhang and C. G. Cassandras, "An improved forward algorithm for optimal control of a class of hybrid systems," *IEEE Trans. Autom. Control*, vol. 47, no. 10, pp. 1735–1739, Oct. 2002.
[28] L. Miao and C. G. Cassandras, "Optimality of static control policies in some discrete event systems," *IEEE Trans. Autom. Control*, vol. 50, no. 9, pp. 1427–1431, Sep. 2005.
[29] J. Mao, C. G. Cassandras, and Q. Zhao, "Optimal dynamic voltage scaling in energy-limited nonpreemptive systems with real-time constraints," *IEEE Trans. Mobile Comput.*, vol. 6, no. 6, pp. 678–688, Jun. 2007.
[30] C. G. Cassandras and R. Mookherjee, "Receding horizon optimal control for some stochastic hybrid systems," in *Proc. 42nd IEEE Conf. Decis. Control*, Dec. 2003, vol. 3, pp. 2162–2167.
[31] L. Miao and C. G. Cassandras, "Receding horizon control for a class of discrete-event systems with real-time constraints," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 825–839, May 2007.

**Lei Miao** received the B.S. and M.S. degrees in telecommunication engineering from Northeastern University, Shenyang, Liaoning, China, in 1998 and 2001, respectively, and the Ph.D. degree in electrical engineering from Boston University, Boston, MA, USA, in 2006.

From 2006 to 2009, he was with Nortel Networks, Billerica, MA, USA. From 2009 to 2011, he was Visiting Professor at the University of Cincinnati, Cincinnati, OH, USA. From 2011 to 2014, he was with NuVo Technologies/Legrand North America, Hebron, KY, USA. From 2014 to 2015, he was a faculty member at State University of New York Farmingdale, Farmingdale, NY, USA. Currently, he is Assistant Professor of Mechatronics Engineering at Middle Tennessee State University, Murfreesboro, TN, USA. His research interests include control and optimization for discrete event systems and hybrid systems, with applications in communication networks, wireless networks, and cyberphysical systems.

**Jianfeng Mao** received the B.E. and M.E. degrees in automation science from Tsinghua University, Beijing, China, in 2001 and 2004, respectively, and the Ph.D. degree in systems engineering from Boston University, Boston, MA, USA, in 2009.

His research primarily focuses on modeling and optimization of discrete event and hybrid systems with applications to transportation systems, computer networks, sensor networks, and health-care systems.

**Christos G. Cassandras** (F'96) received the B.S. degree in engineering and applied science from Yale University, New Haven, CT, USA, in 1977, the M.S.E.E. degree from Stanford University, Stanford, CA, USA, in 1978, and the M.S. and Ph.D. degrees in applied mathematics from Harvard University, Cambridge, MA, USA, in 1979 and 1982, respectively.

He was with ITP Boston, Inc., Cambridge, MA, USA, from 1982 to 1984, where he was involved in the design of automated manufacturing systems. From 1984 to 1996, he was a Faculty Member with the Department of Electrical and Computer Engineering, University of Massachusetts Amherst, Amherst, MA, USA. Currently, he is a Distinguished Professor of Engineering with Boston University, Brookline, MA, USA, the Head of the Division of Systems Engineering, and a Professor of Electrical and Computer Engineering. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL from 1998 to 2009. He has authored more than 350 refereed papers in the following research areas, and five books. He specializes in the areas of discrete event and hybrid systems, cooperative control, stochastic optimization, and computer simulation, with applications in computer and sensor networks, manufacturing systems, and transportation systems.

Dr. Cassandras is a member of Phi Beta Kappa and Tau Beta Pi. He is also a Fellow of the International Federation of Automatic Control (IFAC). He was a recipient of several awards, including the 2011 IEEE Control Systems Technology Award, the 2006 Distinguished Member Award of the IEEE Control Systems Society, the 1999 Harold Chestnut Prize (IFAC Best Control Engineering Textbook), a 2011 prize and a 2014 prize for the IBM/IEEE Smarter Planet Challenge competition, the 2014 Engineering Distinguished Scholar Award at Boston University, several honorary professorships, a 1991 Lilly Fellowship, and a 2012 Kern Fellowship. He serves on several editorial boards and has been a Guest Editor for various journals. He was the President of the IEEE Control Systems Society in 2012.