On Burst Transmission Scheduling in Mobile TV Broadcast Networks

Mohamed Hefeeda, *Member, IEEE*, and Cheng-Hsin Hsu, *Student Member, IEEE*

***Abstract*—In mobile TV broadcast networks, the base station broadcasts TV channels in bursts such that mobile devices can receive a burst of traffic and then turn off their radio frequency circuits till the next burst in order to save energy. To achieve this energy saving without scarifying streaming quality, the base station must carefully construct the burst schedule for all TV channels. This is called the burst scheduling problem. In this paper, we prove that the burst scheduling problem for TV channels with arbitrary bit rates is NP-complete. We then propose a practical simplification of the general problem, which allows TV channels to be classified into multiple classes, and the bit rates of the classes have power of two increments, e.g., 100, 200, and 400 kbps. Using this practical simplification, we propose an optimal and efficient burst scheduling algorithm. We present theoretical analysis, simulation, and actual implementation in a mobile TV testbed to demonstrate the optimality, practicality, and efficiency of the proposed algorithm.**

***Index Terms*—Burst scheduling, Digital Video Broadcast-Handheld (DVB-H), energy saving, mobile multimedia, mobile TV, video broadcast networks, wireless video streaming.**

I. Introduction

MOBILE TV allows users to watch their favorite TV shows on small handheld devices while traveling. It, therefore, extends the prime-time viewing of users and provides more business opportunities for content providers. Mobile TV has already been deployed in parts of Europe and Asia and in pilot-testing in several locations in North and South America [1]. This rapid adoption is fueled by the desire of users for multimedia content and by the technological advances in wireless mobile devices, such as personal digital assistants (PDAs), smart cellular phones, and mobile media players. Many of these devices have evolved to almost full-fledged mobile computers with high-resolution displays, fast network links, large memory and storage space, and fast processors. Therefore, multimedia content can be rendered on most of these mobile devices, which further stimulates the user demands for more content and better quality. A common issue in all mobile devices is the limited energy supply since they are battery-powered. Thus, minimizing the energy consumption in mobile TV networks is indeed a critical problem for the success and wide adoption of such systems.

We consider energy optimization in mobile TV networks in which a base station concurrently broadcasts multiple digital TV channels to mobile devices over a common wireless medium. Examples of such systems include the Digital Video Broadcast-Handheld (DVB-H) [2]–[4] and Forward Link Only technology (MediaFLO) [5] networks. In these systems, the base station broadcasts TV channels in *bursts* with bit rates much higher than the encoding rates of the video streams. Thus, mobile devices can receive a burst of traffic and then turn off their radio frequency (RF) circuits till the next burst. This is referred to as *time slicing* [3], and it is illustrated in Fig. 1 for one TV channel. Common standards for mobile TV networks, such as DVB-H and MediaFLO, *dictate* using energy-saving schemes to increase the viewing time on mobile devices. While time slicing leads to energy conservation, burst transmission schedules, which specify the burst start times and sizes, must be carefully composed to guarantee service quality and proper functioning of the system. This is because of a number of reasons. First, since mobile TV receivers have limited receiving buffer capacity, arbitrary burst schedules can result in buffer over- and underflow instances that cause play-out glitches and degrade viewing experience. Second, as several TV channels share the same broadcast medium, burst schedules must not have any burst conflicts, which occur when two or more bursts intersect with each other in time. Third, turning on RF circuits is not instantaneous as it takes some time to search for and lock on RF signals. This imposes overhead on energy consumption, and this overhead must be considered in constructing burst schedules. Fourth, burst schedules directly impact the channel switching delay, which is the average time a user waits before s/he starts viewing a selected channel when a change of channel is requested by the user. Long and variable channel switching delays are annoying to users and could turn them away from the mobile TV service. This is because many users quickly flip through several TV channels before they decide on watching a specific one.

Current practices of computing burst schedules are rather *ad hoc*. For example, the heuristic method proposed in the standard documents [6, p. 66] provides schedules for only one TV channel. This heuristic simply allocates a new burst only after the data of its preceding burst is consumed by the player at the receiver. This cannot be generalized to multiple TV channels with *different* bit rates because the computed schedule may have burst conflicts and may result in buffer under/overflow instances. Thus, many mobile TV deployments resort to encoding all TV channels at the same bit rate in order to use the heuristic

Manuscript received October 24, 2008; revised May 18, 2009; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor D. M. Chiu. First published October 30, 2009; current version published April 16, 2010. This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

The authors are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: mhefeeda@cs.sfu.ca).

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/TNET.2009.2030326

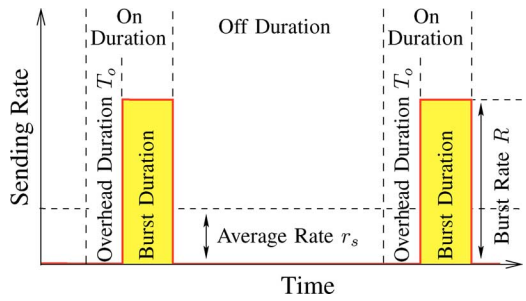


Fig. 1. Time slicing in mobile TV networks to save energy.

burst scheduling technique in the standard. This is clearly inefficient and may yield large quality variations among TV channels carrying different types of multimedia content, which is the typical case in practice. For example, encoding a high-motion soccer game requires a much higher bit rate than encoding a low-motion talk show. If we encode all TV channels at the same high bit rate, some channels may unnecessarily be allocated more bandwidth than they require, and this extra bandwidth yields only marginal or no visual quality improvement. Thus, the expensive wireless bandwidth of the broadcast network could be wasted. On the other hand, if we encode all TV channels at the same low or moderate bit rate, not all channels will have good visual quality, which could be annoying to users of a *commercial* service. To the best of our knowledge, there exist no systematic ways in the literature that fully address the burst scheduling problem for mobile TV channels at different bit rates, despite the tremendous potential of mobile TV networks.

The contributions of this paper can be summarized as follows.

- We formulate the burst scheduling problem in mobile TV networks, and we show that this problem is NP-complete for TV channels with arbitrary bit rates. This is presented in Section III.
- We propose a practical simplification of the general problem, which allows TV channels to be classified into multiple classes, and each class has a different bit rate. The bit rate of class c, r_c, can take any value in the form of $r_c = 2^i \times r_1$, where $i \in \{0, 1, 2, 3, \ldots\}$, and r_1 is the bit rate of the lowest class. r_1 can take any arbitrary bit rate. For example, the bit rates 800, 400, 200, and 100 kbps could make four different classes for encoding sports events, movies, low-motion episodes, and talk shows, respectively. This classification of TV channels also enables the operators of mobile TV networks to offer differentiated services: higher bit rate classes can broadcast premium services for higher fees. This service differentiation is not possible with the current burst scheduling schemes. Using the above simplification, we develop an optimal (in terms of energy consumption) burst scheduling algorithm, which is quite efficient: its time complexity is $O(S \log S)$, where S is the total number of TV channels. The new algorithm is presented in Section IV.
- We implement our algorithm in a real mobile TV testbed and demonstrate its practicality and effectiveness in saving energy. We also conduct simulation experiments to analyze the performance of our algorithm under a wide range of parameters. In addition, we empirically demonstrate the importance of using different bit rates for various video sequences. We do this by encoding several video sequences with different complexities and analyzing their rate-distortion characteristics. These implementation and simulation experiments are presented in Section V.

The following section presents a brief background on different methods for streaming video to mobile devices, places our work within the big picture in the area, and summarizes the related works in the literature.

II. Background and Related Work

A. Background and the Big Picture

Delivering TV programs to mobile devices such as cell phones and PDAs can be done over wireless cellular networks or over dedicated broadcast networks. Traditional cellular networks support unicast, which does not efficiently utilize network resources, especially in urban areas where there are many users interested in the same content. To cope with this problem, extensions to support multicast and broadcast have been proposed for cellular networks. For example, the 3G Partnership Project (3GPP) has defined an integrated multicast and broadcast extension, called Multimedia Broadcast Multicast Service (MBMS) [7], for Universal Mobile Telecommunications System (UMTS). Video streaming in cellular networks is outside the scope of this paper. We focus on dedicated broadcast networks, which have the potential to serve TV content to a large number of subscribers. We note that dedicated broadcast networks usually employ cellular networks to enable user interaction with some TV programs, but not to transmit video.

There are several example systems and standards for dedicated video broadcast networks, including Terrestrial-Digital Multimedia Broadcasting (T-DMB) [8], Integrated Services Digital Broadcasting-Terrestrial (ISDB-T) [9], MediaFLO [5], and DVB-H [3], [10]. A brief overview of each follows. T-DMB [8] is an extension for the Digital Audio Broadcast (DAB) standard [11] to add video broadcast services to the high-quality audio services offered by DAB. The extension includes both source coding, such as using MPEG-4/AVC encoding, and channel coding, such as employing Reed–Solomon code. The development of T-DMB is supported by the South Korean government, and T-DMB is the first commercial mobile video broadcast service. In addition to South Korea, several European countries may deploy T-DMB as they already have equipments of and experiences with DAB systems. ISDB-T [9] is a digital video broadcast standard defined in Japan, which is not only for fixed video receivers, but also for mobile receivers. ISDB-T divides its band into 13 segments, where 12 of them are used for broadcasting HDTV and one is for broadcasting to mobile devices.

MediaFLO [5] is a video broadcast system developed by Qualcomm and the FLO forum [12]. MediaFLO is designed from scratch for video broadcast services to mobile devices. The details of the design are not public. In contrast, DVB-H [3], [10] is an open international standard [4]. Among the above dedicated broadcast networks, only DVB-H and MediaFLO try to minimize the energy consumption of mobile devices by periodically turning their RF circuits off [5], [13]. Since the

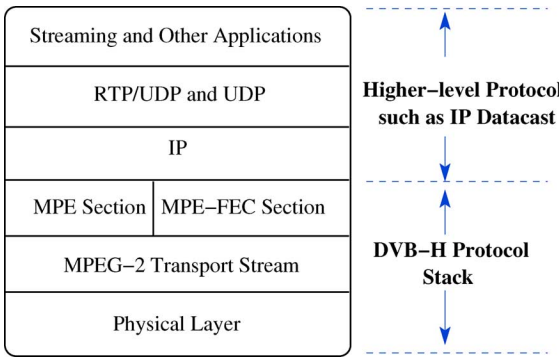


Fig. 2. The protocol stack of mobile TV networks using the DVB-H standard.

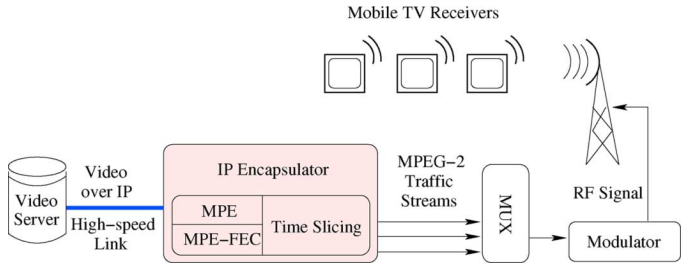


Fig. 3. The main components of mobile TV broadcasting systems. Our work optimizes the time slicing component, which in turn minimizes the energy consumption of mobile devices by making their RF circuits sleep for the longest period.

goal of this paper is to optimize energy consumption for mobile devices, we use the open DVB-H standard in our discussion throughout the paper. Nonetheless, in our problem formulation and solution, we abstract away the specific details of the DVB-H standard. Therefore, our solution is also applicable to the MediaFLO system and other wideband video broadcast networks that may be developed in the future.

We now present an overview of the DVB-H system, defining some concepts that will be used later in the paper. DVB-H is an extension to the Digital Video Broadcast-Terrestrial (DVB-T) standard [14] tailored for mobile devices. DVB-H standard defines protocols below the network layer and uses IP as the interface with higher-layer protocols such as UDP and RTP. Standards such as IP Datacast [2], [3] complement DVB-H by defining higher-level protocols for a complete end-to-end solution. Fig. 2 illustrates the protocol stack of video streaming over DVB-H networks. DVB-H uses a physical layer compatible with the DVB-T, which employs orthogonal frequency division multiplexing (OFDM) modulation. DVB-H encapsulates IP packets using Multi-Protocol Encapsulation (MPE) sections to form MPEG-2 transport streams. Thus, data from a specific TV channel form a sequence of MPEs. MPEs are optionally FEC-protected before being transmitted over the wireless medium. To save energy of mobile devices, MPEs belonging to a given TV channel are transmitted in *bursts* with a bit rate much higher than the video stream itself. Fig. 3 illustrates the main components of a DVB-H system. The goal of this paper is to determine the optimal burst sizes and their transmission times for multiple TV channels in order to minimize the energy consumption of mobile devices. Therefore, our solution would be used in the Time Slicing part of the IP encapsulator in Fig. 3.

We note that receivers in mobile TV *broadcast* networks have separate RF circuit and antenna for processing TV signals, other than the circuits for receiving and making phone calls. Our work focuses only on optimizing the energy saving for TV signal receivers. In addition, because of the one-way nature of broadcast networks, feedback channels from numerous receivers to the base station are not practical. Thus, many of the energy-saving techniques designed for video streaming to the general wireless devices are not applicable to mobile TV networks. For example, the throttling technique proposed in [15], which enables a wireless receiver to indirectly control the sending pattern of an Internet streaming server, requires a feedback channel from the receiver to the server, which may not be possible in mobile TV networks.

B. Related Work

A number of works have studied energy saving in mobile TV networks. The authors of [16] and [6] estimate the effectiveness of the time slicing technique for given burst schedules. Both works indicate that time slicing enables mobile TV receivers to turn off their RF circuits for a significant fraction of the time. These two works do not solve the burst scheduling problem—they only compute the achieved energy saving for given *predetermined* burst schedules. In contrast, we formulate and solve the burst scheduling problem for different channel bit rates. To the best of our knowledge, there exist no other burst scheduling algorithms that can accommodate TV channels at *different* bit rates in the literature.

The authors of [17] propose an energy-saving strategy by not receiving some MPE-FEC sections once the received sections can successfully reconstruct the data. Skipping a few MPE-FEC sections means that mobile TV receivers can turn off their RF circuits earlier, which leads to additional energy saving compared to receiving all MPE-FEC sections regardless of whether they are necessary. The authors of [18] consider mobile TV receivers with an auxiliary short-range wireless interface and construct a cooperative network among several receivers over this short-range wireless network. Mobile TV receivers share received IP packets over this short-range network, so each mobile TV receiver only receives a small fraction of IP packets directly from the DVB-H network. This allows receivers to reduce the frequency of turning on their RF circuits. Assuming sending/receiving IP packets through the short-range network is more energy-efficient than receiving DVB-H sections, this cooperative strategy can reduce energy consumption. The proposals in [17] and [18] are orthogonal and complementary to our work, as they reside in the mobile devices themselves and try to achieve additional energy saving on top of that achieved by time slicing. In contrast, our algorithm is to be implemented in the base station broadcasting TV channels to mobile devices.

Optimizing mobile TV networks from aspects other than energy saving has also been studied in the literature, including the radio performance optimization in [19], the frame refresh delay reduction in [20]–[24], and the video quality optimization in [22]. The author of [19] studies DVB-H radio performance using simulations. The DVB-H system parameters are

TABLE I
LIST OF SYMBOLS USED IN THE PAPER

Symbol	Description
b (bits)	Link-layer receiver buffer size
d (sec)	Average channel switching delay
γ	Average energy saving over all channels
p (sec)	Recurring frame duration
r_s (bps)	Bit rate of TV channel s
R (bps)	Bandwidth of the wireless medium
S	Number of TV channels
T_o (sec)	Overhead period

classified into three sets: physical layer, time slicing module, and MPE-FEC module, while interaction among parameters in different sets are outlined. The simulation results in [19] indicate that extending the time interleaving depth in MPE-FEC to at least 100 ms results in good radio link performance, while increasing it beyond 300 ms yields no further improvements. Extending the time interleaving depth, however, increases burst durations and imposes negative impact on power consumption.

The frame refresh delay refers to the time period between receiving the first bit of a new video stream and reaching the next random access point, typically an intracoded frame, of that video. Shorter frame refresh delays lead to shorter channel switching delays, and thus more responsive systems. To reduce frame refresh delays, the authors of [20] propose to periodically add redundant intracoded frames into video streams coded by H.264/AVC [25]. By frequently adding low-quality intracoded redundant frames into a video stream, more random access points are created, which in turn reduces the refresh delay. Instead of sending low-quality intracoded frames over dedicated channels, intracoded frames can also be transmitted at the beginning of bursts to shorten frame refresh delays [21]–[24].

The authors of [22] propose a rate control algorithm for broadcasting multiple video channels using DVB-H systems. The objective of this rate control algorithm is to determine the best quantization parameter (QP) for individual TV channels for equalizing the video quality across all TV channels. The algorithm consists of two components: a rate controller based on fuzzy logic and a heuristic quality controller that determines QP values based on historical rate-distortion (R-D) curves. The QP values are then sent back to the video encoder for rate adaptation.

None of the aforementioned works presents burst scheduling algorithms. In fact, unlike our burst scheduling problem that is in the link layer, the radio performance optimization lies in the physical layer, and both frame fresh delay reduction and video quality optimization fall in the application layer. Therefore, all these works are complementary to our work on maximizing energy saving.

III. PROBLEM STATEMENT AND HARDNESS

In this section, we formulate the burst scheduling problem and show that it is NP-complete. We list all symbols used in the paper in Table I for quick reference.

We consider mobile TV networks in which a base station concurrently broadcasts S digital TV channels to clients with handheld devices over a wireless medium with bandwidth R kbps. Each TV channel s, $1 \leq s \leq S$, has a bit rate r_s kbps, which

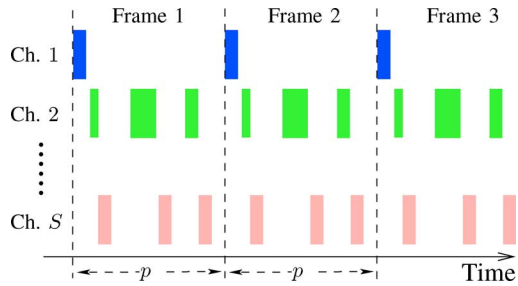


Fig. 4. The burst scheduling problem in mobile TV networks.

is typically much less than R. The base station broadcasts each TV channel in *bursts* at bit rate R kbps. After receiving a burst of data, the RF circuits are switched off until the time of the next burst, which is computed by the base station and included in the header fields of the current burst. The RF receivers of mobile TV receivers must be open slightly before the burst time because it takes some time to wake up and synchronize the circuitry before it can start receiving data. This time is called the overhead duration and is denoted by T_o. With current technology, T_o is in the range of 50–250 ms [3], [6]. The energy saving achieved by mobile devices receiving TV channel s is denoted by γ_s, and it is calculated as the ratio of time the channel is in off mode to the total time [6], [16]. We define the system-wide energy saving metric over all TV channels as $\gamma = \left(\sum_{s=1}^{S} \gamma_s\right)/S$. The energy saving as well as the burst scheduling itself are typically performed on a recurring time window called a frame. The length of each frame is a system parameter and is denoted by p. With these definitions, we state the burst scheduling problem as follows.

Problem 1 (Burst Scheduling in Mobile TV Systems): Given S TV channels of different bit rates to be simultaneously broadcast to mobile devices. Each TV channel is broadcast as bursts of data to save the energy of mobile devices. Find the optimal transmission schedule for bursts of all TV channels to maximize the system-wide energy saving metric γ. The transmission schedule must specify the number of bursts for each TV channel in a frame as well as the start and end times for each burst. The schedule cannot have burst collisions, which happen when two or more bursts have nonempty intersection in time. In addition, given a link-layer receiver buffer size b, the schedule must ensure that there are no receiver buffer violations for any channel. A buffer violation occurs when the receiver has either no data in the buffer to pass on to the decoder for playout (buffer underflow) or has no space to store data during a burst transmission (buffer overflow).

Fig. 4 illustrates a simple example for the burst scheduling problem. Notice that the bursts have different sizes, are disjoint in time, and are repeated in successive frames. In addition, there can be multiple bursts for a TV channel in each recurring frame to ensure that there are no buffer under/overflow instances. To illustrate the receiver buffer dynamics for a valid solution of the burst scheduling problem, we demonstrate in Fig. 5 the buffer level as a function of time. This is shown for a receiver of an arbitrary TV channel s with two bursts in each frame. We make two observations on this figure. First, during a burst, the buffer level increases with a rate (slope of the line) of $R - r_s$, which

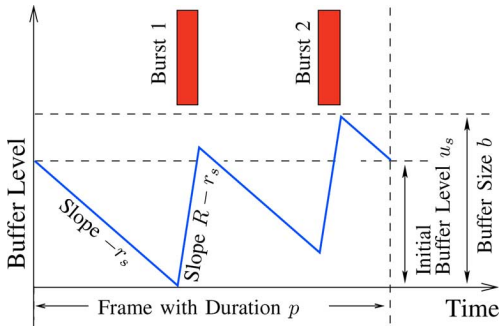


Fig. 5. The dynamics of the receiver buffer during in successive frames.

is much larger than the consumption rate of $-r_s$ when there is no burst. Second, the frame starts with an initial buffer level (denoted by u_s) and ends at the same buffer level. Clearly, this is a requirement for any valid burst scheduling solution, otherwise there is no guarantee that the buffer will not suffer from over- or under-flow instances.

Remark: Various Buffers in Streaming Systems: It is important to note that in video streaming systems, there are typically multiple buffers in different layers, and these buffers are coordinated using signaling mechanisms of the protocols in the corresponding layers. For example, in the application layer, an H.264/AVC video encoder can use the buffering model described in the standard document [26, Annex. C] to specify the buffer requirements for the decoder in order to have a smooth playout of the encoded video. The DVB-H standard, on the other hand, defines a buffering model for the physical, link, and application layers [27, Sec. 5.3]. As we mentioned in Section II-A, our burst scheduling problem belongs to the time slicer in the link layer. Thus, the receiver buffer mentioned in Problem 1 and throughout the paper is the link-layer buffer, which is also known as the time slicing buffer [28, Sec. 9.4].

The following theorem shows that the burst scheduling problem is NP-complete.

Theorem 1 (Burst Scheduling): The burst scheduling problem stated in Problem 1 is NP-complete.

Proof: We first show that the problem of maximizing energy saving (Problem 1) is the same as the problem of minimizing the total number of bursts in each frame for a given frame length p. To maximize energy saving γ, we have to minimize the RF circuit on-time for receivers. Notice that the RF circuit on-time can be divided into two parts: burst and overhead durations as illustrated in Fig. 1. The burst duration represents the time in which mobile devices receive the video data. Since we consider steady burst schedules, where the number of received bits is equal to the number of consumed bits in each frame for all mobile devices, the burst duration must be constant across all feasible burst schedules. Notice also that each burst incurs a fixed overhead T_o. Therefore, minimizing the RF circuit on-time is equivalent to minimizing total number of bursts in each frame, because it minimizes the total overhead.

Next, we reduce the NP-complete problem of task sequencing with release times and deadlines [29, p. 236] to the problem of minimizing the total number of bursts in each frame. The task sequencing problem consists of T tasks, where each task $t = 1, 2, \ldots, T$ is released at time x_t with length y_t and deadline z_t. The problem is to determine whether there is a single machine (nonpreemptive) schedule that meets all constraints of release times and deadlines. For any task sequencing problem, we set up a burst scheduling problem as follows. We let $S = T$ and map every task to a TV channel. We let $p = p^*$ be the optimum frame length, which will be derived in the next subsection. We choose an arbitrary burst bit rate R. For any TV channel s ($s = 1, 2, \ldots, S$), we let channel bit rate $r_s = Ry_s/p^*$ to balance the number of received bits and the number of consumed bits. We set the initial buffer level $u_s = (z_s - y_s)r_s$, which guarantees that a burst with length y_s will be scheduled (and finished) *before* the deadline z_s, or mobile devices will run out of data for playout (underflow). We let the receiver buffer size $b_s = u_s + y_s R - (x_s + y_s)r_s - \epsilon$, where $\epsilon > 0$ is an arbitrary small number. Selecting such a b_s guarantees that a burst with length y_s will be scheduled *after* the release time x_s, or mobile devices will run out of buffer space (overflow).

Clearly, we can set up the burst scheduling problem in polynomial time. Furthermore, solving the burst scheduling problem leads to the solution of the task sequencing problem because the minimum total number of bursts is equal to S if and only if there is a nonpreemptive schedule that satisfies the constraints on release times and deadlines of the task sequencing problem. Thus, the burst scheduling problem is NP-hard. Finally, determining whether a given burst schedule meets the collision-free and buffer violation-free requirements, i.e., a valid solution for Problem 1, takes polynomial time. Hence, the burst scheduling problem is NP-complete. ■

We notice that this theorem might seem counter-intuitive at a first glance because the burst scheduling problem looks somewhat similar to preemptive machine scheduling problems. However, there is a fundamental difference between our burst scheduling problem and various machine scheduling problems: most of the machine scheduling problems consider *costless* preemption model [30]. In contrast, our burst scheduling problem adopts *costly* preemption model, as our problem aims at minimizing the total number of bursts in a frame, which is essentially the number of preemptions. Therefore, the algorithms developed for various machine scheduling problems are not applicable to our burst scheduling problems (see [30] for a comprehensive list of machine scheduling problems). The costly preemption model has only been considered in a few works [31]–[34]. The authors of [32] and [33] partially cope with preemption costs by adding constraints to limit the number of preemptions. The authors of [31] solve the problem of minimizing the weighted sum of the total task flow time and the preemption penalty, where the weight is heuristically chosen. The author of [34] considers the problem of minimizing weighted completion time and task makespan under a given preemption cost. Unlike these problems, our burst scheduling problem solely uses the preemption cost as the objective function and does not allow any late task, which renders the algorithms proposed in [31]–[34] inapplicable to our problem.

IV. Burst Scheduling Algorithm

In the previous section, we proved that the general burst scheduling problem is NP-complete. In this section, we present

P2OPT

1. Compute optimal frame length p^*
2. Allocate a leaf node l for every channel s, **let** $l.ch = s$
3. Push all leaf nodes to a priority queue P with key r_s/r_1
4. **while true**{
5. **let** m_1 = pop_min(P), m_2 = pop_min(P);
6. **if** m_2 is **null** or m_1.key $<$ m_2.key { // no sibling
7. **if** m_2 is not **null** push(P, m_2); // return m_2 back to P
8. Allocate a dummy node m_2, where m_2.key = m_1.key
9. }
10. Create an internal node n with children n.left & n.right
11. **let** n.left = m_1, n.right = m_2;
12. **let** n.key = m_1.key + m_2.key;
13. push(P, n); // insert this internal node
14. **if** n.key $\geq R/r_1$ **break**;
15. }
16. **if** $|P| > 1$ **return** $\varnothing$; // no feasible schedule
17. **let** $\mathbb{T} = \varnothing$; // start composing schedule $\mathbb{T}$
18. Traverse the tree from root down, annotate each node with an *offset* with the value of the reverse bit pattern from root till this node
19. **foreach** leaf node l {
20. **for** i = 0 to l.key - 1 {
21. start = offset + $i \times (R/r_1)/l$.key;
22. // add a burst to channel l.ch at time start $\times\ p^* r_1/R$
23. insertBurst($\mathbb{T}$, start $\times\ p^* r_1/R$, l.ch);
24. }
25. }
26. **return** $\mathbb{T}$;

Fig. 6. An efficient algorithm to solve the burst scheduling problem.

an algorithm that optimally and efficiently solves this problem under a certain assumption that usually holds in practice. To simplify the presentation, we first describe an overview of the algorithm and an illustrative example. We then analyze the algorithm and prove its correctness, optimality, and efficiency. Then, we analyze the tradeoff between the achieved energy saving and the channel switching delay. Finally, we discuss several practical issues of the proposed algorithm.

A. Overview of the Algorithm

We propose an optimal algorithm for the burst scheduling problem when the bit rate of a TV channel s, $1 \leq s \leq S$, is given by $r_s = 2^i \times r_1$ for any i, where $i \in \{0, 1, 2, 3, \ldots\}$, and r_1 can be any arbitrary bit rate. As mentioned in Section I, the TV channels can be divided into classes, where each class contains similar-type multimedia content encoded at the same bit rate. Without loss of generality, we assume that the bit rates of the S channels are ordered such that $r_1 \leq r_2 \leq \cdots \leq r_S$. If otherwise, a relabeling based on the bit rates is applied. We also assume that the bandwidth of the wireless medium satisfies $R = 2^k \times r_1$, where k is a positive integer. We present in Fig. 6 an optimal algorithm for solving the burst scheduling problem in this case.

The basic idea of our algorithm is as follows. The algorithm first computes the optimal value for the frame length p^* (we derive p^* in Theorem 3). It then divides p^* into bursts of equal size $p^* r_1$ bits. Thus, there are $(p^* R)/(p^* r_1) = R/r_1$ bursts in each frame. Then, each TV channel is allocated a number of bursts proportional to its bit rate. That is, TV channel s, $1 < s \leq S$, is allocated r_s/r_1 bursts and TV channel 1 is allocated only one burst in each frame. Moreover, bursts of TV channel s are equally spaced within the frame, with interburst distance of $p^*/(r_s/r_1)$ s. This ensures that there will be no underflow instances in the receiver buffer because the consumption rate of the data in the buffer for TV channel s is r_s bps and the burst size is $p^* r_1$ bits. Since the optimal frame length can be written as $p^* = b/r_1$, the size of each burst is $p^* r_1 = b$, which is no larger than the receiver buffer size b. This ensures that there is no buffer overflow instances. Finally, bursts of different TV channels are arranged such that they do not intersect in time, that is, the resulting schedule is conflict-free.

To achieve the above steps in a systematic way, the pseudocode in Fig. 6 works as follows. It builds a binary tree bottom-up. Leaf nodes representing TV channels are created first, where the leaf node of TV channel s is annotated with the value r_s/r_1. The algorithm uses this value as the key and inserts all leaf nodes into a priority queue. This priority queue is implemented as a binary heap to efficiently find the node with the smallest key. The algorithm then repeatedly merges the two nodes that have the least key values into a new internal node. This new internal node has a key value equivalent to the sum of the key values of its children. This is done by popping the smallest two values from the heap and then pushing the newly created node into it. The merging of nodes continues till the tree has a height of $\log(R/r_1)$. The last merged node becomes the root of the binary tree. Note that if the wireless medium is fully utilized by the TV channels, i.e., $\sum_{s=1}^{S} r_s = R$, the computed bursts of the different TV channels will completely fill the frame p. If otherwise (i.e., $< 100\%$ utilization), the wireless medium will have to be idle during some periods within the frame. The algorithm represents these idle periods as dummy nodes in the tree.

Once the binary tree is created, the algorithm constructs the burst schedule. It allocates to each TV channel a number of bursts that is equal to its key value. In order to ensure conflict-free schedule, the algorithm computes the start time for each burst as follows. For each leaf node representing a TV channel, the algorithm traverses the tree top-down. During the traversal, each node is assigned the reverse bit pattern from the root to this node, where the right branch has the bit 1 and the left branch has the bit 0. The bit pattern for a leaf node encodes the number of bursts and their start times for the TV channel corresponding to that node. For example, in a tree of depth 3, the bit pattern 010 means that the TV channel is assigned only the second burst in a frame of eight bursts. For leaf nodes at levels less than the depth of the tree, the bit pattern is padded with one or more "x" to the left. For example, if a leaf node has the bit pattern x01 in a tree of depth 3, this means that the node is at level 2 from the root. It also means that this node should be assigned the two bursts: 001 and 101. Notice that the first burst assigned to any TV channel is equal to the numeric value of its bit pattern, with all "x" bits set to zero. The algorithm computes this value and refers to it as the offset. The algorithm then computes successive bursts relative to this offset.

B. Illustrative Example

Consider four TV channels distributed over three different classes: two channels in class I with $r_1 = r_2 = 256$ kbps,

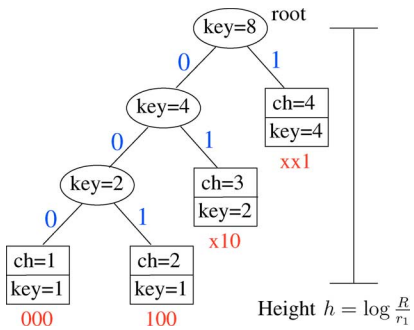


Fig. 7. An illustrative example for the P2OPT algorithm.

one in class II with $r_3 = 512$ kbps, and one in class III with $r_4 = 1024$ kbps. Let the wireless medium bandwidth $R = 2048$ kbps and the receiver buffer $b = 1$ Mb. As explained later, p^* is given by $p^* = b/r_1 = 4$ s. The algorithm divides each frame into $R/r_1 = 8$ bursts, and assigns 1, 1, 2, 4 bursts to channels 1, 2, 3, 4, respectively. The algorithm constructs a binary tree bottom-up as shown in Fig. 7. Four leaf nodes are created, each representing a TV channel and having a key value equal to the number of bursts that should be allocated to that channel. Notice that the leaf nodes are logically placed at different levels based on their key values. Then, the algorithm recursively merges nodes with the same key values till it creates the root node with key value $8 = R/r_1$. Then, the algorithm constructs the schedule by traversing the tree from the root down to assign bit patterns to leaf nodes, which are shown in Fig. 7. Using these bit patterns, the offset for each node is computed and the bursts are assigned. The resulting burst schedule is

$$\begin{aligned}\mathbb{T} =&\{(0.0,1),(0.5,4),(1.0,3),(1.5,4),\\ &(2.0,2),(2.5,4),(3.0,3),(3.5,4)\}\end{aligned}$$

where the first element in the parentheses is the burst start time, and the second element indicates the TV channel.

Notice that we present a rather simple example for illustration, and the P2OPT algorithm is quite flexible on the bit rates of individual TV channels. In fact, network operators can broadcast each TV channel s ($2 \le s \le S$) at an average bit rate $r_s = 2^i \times r_1$ for *any* i, where $i \in \{0, 1, 2, 3, \ldots\}$. For instance, by setting $i = 0$ for all TV channels, network operators can compute the optimal broadcast schedule for TV channels with uniform bit rate. Therefore, the P2OPT algorithm provides a systematic way to construct optimal burst schedules for mobile TV networks that broadcast channels at uniform bit rates, which is currently a common practice.

C. Analysis of the Algorithm

We show the correctness, efficiency, and optimality of our algorithm in the following two theorems.

Theorem 2 (Correctness and Efficiency): The P2OPT algorithm in Fig. 6 returns a conflict-free schedule with no buffer under/overflow instances, if one exists. It has a worst-case time complexity of $O(S \log S)$, where S is the number of TV channels.

Proof: We prove the correctness part in two steps. First, observe that a burst schedule produced by P2OPT is conflict-free because the algorithm assigns each TV channel a unique bit pattern that specifies the allocated bursts to that TV channel. Moreover, the bit pattern is padded with zero or more "x" to the left, which guarantees that TV channel s is assigned r_s/r_1 equally spaced bursts. Hence, if P2OPT returns a schedule, this schedule is conflict-free with no buffer under/overflow instances.

Second, we prove that if P2OPT fails to return a schedule, there exists no feasible schedule for the given TV channels. P2OPT only merges nodes at the same binary tree level, and nodes at lower levels have strictly smaller key values than nodes at higher levels. Therefore, P2OPT merges all nodes from bottom-up and creates at most one dummy node at every level. Moreover, P2OPT uses line 12 to ensure the key value of each node indicates how many time slots are consumed by itself (for a leaf node) or by all leaf nodes in its subtree (for an internal node). P2OPT returns no feasible solution for a given problem if a full binary tree with height $h = \log(R/r_1)$ is built and $|P| > 1$ in line 16. Let z be the first merged node with key value R/r_1, which is the last merged node before returning from line 16. Since $|P| > 1$, we let $w \neq z$ be an arbitrary node in P. w must have key value no less than $\frac{1}{2}R/r_1$, otherwise w would have been merged before the children of z. We account for the number of time slots consumed by real (non-dummy) leaf nodes in subtrees beneath w and z. w resides either at the same or higher level than z (case I) or at a level lower than z (case II). In case I, we know that $w.\text{key} \ge z.\text{key} = R/r_1$ and w is a real leaf node because z is the first merged node at its level. Since P2OPT guarantees that at most one dummy leaf node exists at each level, the total time slots occupied by dummy leaf nodes in z's subtree cannot exceed

$$\sum_{i=0}^{\log R/r_1 - 1} 2^i = R/r_1 - 1$$

time slots. This shows that w and z consume at least

$$2\frac{R}{r_1} - \left(\frac{R}{r_1} - 1\right) = R/r_1 + 1$$

time slots. Since $R/r_1 + 1$ exceeds the total number of available time slots R/r_1, there exists no feasible schedule. Case II can be shown in a similar way and is omitted for brevity.

For time complexity, P2OPT can be efficiently implemented using a binary heap, which can be initialized in time $O(S)$. Notice that we have at most $\log R/r_1$ dummy leaf nodes because there is at most one dummy leaf node for each level. The while loop in lines 4–15 iterates at most $O(S + \log R/r_1) = O(S)$ times because $\log R/r_1$ can be considered as a constant for practical encoding bit rates. Since each iteration takes $O(\log S)$ steps, the while loop takes $O(S \log S)$ steps. Constructing the burst schedule in lines 17–25 takes $O(S)$ steps since the tree has up to $2S$ nodes. Thus, the time complexity of P2OPT is $O(S \log S)$. ■

The following theorem shows that the P2OPT algorithm produces optimal burst schedules in terms of maximizing energy saving.

Theorem 3 (Optimality): The frame duration $p^* = b/r_1$, where b is the receiver buffer size, computed by P2OPT maximizes the average energy saving γ over all TV channels.

Proof: We leverage the fact that we established in the proof of Theorem 1: The problem of maximizing energy saving is equivalent to the problem of minimizing the total number of bursts in each frame for a given frame length p. To maximize energy saving γ, we have to minimize the ratio of RF circuit on-time to the frame length. We again divide RF circuit on-time into burst and overhead durations. Notice that the ratio of burst duration to frame length is constant for any feasible schedule because each TV channel is broadcast at a given bit rate. Since each burst incurs overhead T_o s, following the definition of γ, our burst scheduling problem is reduced to the problem of minimizing the total number of bursts normalized to the frame length p.

We first prove by contradiction that any frame length $p < p^*$ cannot result in higher energy saving. Assume a feasible schedule $\mathbb{T}_1$ results in higher energy saving with frame length p_1 than schedule $\mathbb{T}^*$ with frame length p^*, which is produced by P2OPT, where $p_1 < p^*$. To outperform $\mathbb{T}^*$, $\mathbb{T}_1$ must have fewer number of bursts than $\mathbb{T}^*$, i.e., $|\mathbb{T}_1| \leq |\mathbb{T}^*|$. However, $\mathbb{T}_1$ must lead to some buffer overflow because fewer number of bursts is equivalent to longer bursts, but P2OPT has fully utilized (filled up) the receiver buffer b for all bursts. This contradicts the assumption that $\mathbb{T}_1$ is a feasible schedule.

We next consider $p > p^*$. Assume $\mathbb{T}_2$ is a feasible schedule that results in better energy saving with frame length p_2, where $p_2 > p^*$. Let $k = \lceil p_2/p^* \rceil$. Define $\mathbb{T}'$ by repeating $\mathbb{T}^*$, which is produced by P2OPT, k times. Since burst schedule $\mathbb{T}'$ also fills up receiver buffer in all bursts, following the same argument as above, we show that $\mathbb{T}_2$ must lead to some buffer overflow. This contradicts the assumption that $\mathbb{T}_2$ is a feasible schedule. ■

D. Tradeoff Between Energy Saving and Switching Delay

We first compute the average energy saving γ^*. Since channel s has an interburst duration of b/r_s and the burst length is b/R, the average energy saving is given by

$$\gamma^* = \sum_{s=1}^{S} \Big(1 - r_s\Big(1/R + T_o/b\Big)\Big)\Big/S = 1 - \sum_{s=1}^{S} r_s \frac{1/R + T_o/b}{S}. \quad (1)$$

Next, we analyze another important metric in mobile TV networks: the channel switching delay d, which is the time a user waits before s/he starts viewing a selected channel when a change of channel is requested by her/him. Channel switching delay is composed of several parts in which the frame refresh delay and time slicing delay are the two dominating contributors [21], [22]. In Section II, we defined the frame fresh delay and described several works in the literature on minimizing it. The frame refresh delay is controlled in the application layer and is orthogonal to our burst scheduling problem. The time slicing delay refers to the time period between locking on to a mobile TV channel and reaching the first burst of that TV channel. Since time slicing delay is a *by-product* of the time-slicing-based energy-saving scheme, we only consider time slicing delay in this paper. We assume all other parts of channel switching delays are constant as they are outside of the scope of this paper. The average channel switching delay resulted by the optimal schedule γ^* is then given by

$$d^* = \Big(\sum_{s=1}^{S} b/2r_s\Big)\Big/S = \Big(\sum_{s=1}^{S} 1/r_s\Big)\Big(b/2S\Big). \quad (2)$$

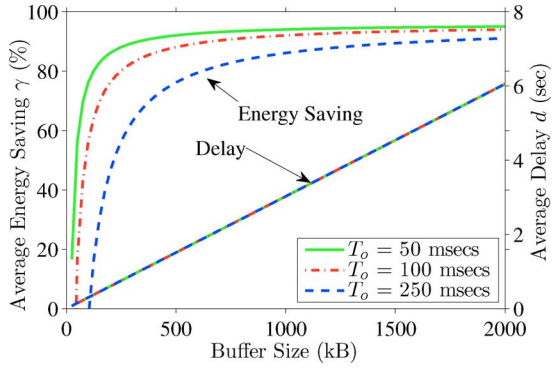


Fig. 8. The tradeoff between energy saving and channel switching delay.

Notice that there is a tradeoff between γ and d, and the main control parameter is the buffer size b. To examine this tradeoff, we present an illustrative example. We consider a broadcast service with $R = 10$ Mbps and 25 TV channels equally distributed in five classes of heterogeneous channel bit rates (i.e., 1024, 512, 256, 128, and 64 kbps). We plot the energy saving and channel switching delay under different overhead durations in Fig. 8. This figure shows that larger buffer sizes and smaller overhead durations lead to higher energy savings *and* also to higher channel switching delays.

E. Discussion

We discuss several practical issues related to the burst scheduling problem and our proposed P2OPT algorithm. First, we do not restrict the coding of videos to the exact power-of-two bit rates. We do not even restrict them to be constant-bit-rate (CBR)-coded. What we assume is that the network operator will broadcast the video over the closet power-of-two bandwidth, and smoothing buffers at the base station will be used to regulate the traffic. Variable-bit-rate (VBR) streams achieve higher coding efficiency compared to CBR-coded ones, and users of recent video coding standards, such as H.264/AVC, no longer use strict CBR coding [35]. Instead, users usually specify the network channel bit rate and smoothing buffer parameters at encoding time, and the video coders perform rate control to ensure the resulting VBR-coded streams can be transmitted over the CBR network channel without any buffer under/overflow instances. Therefore, our algorithm allows video streams to be VBR-coded and transmitted over CBR network channels using smoothing buffers, which is currently a common practice.

Second, we need to recompute the burst schedule many times during the broadcast in order to cope with the broadcast service dynamics. For example, each TV channel frequently changes programs (say each 30 min). Clearly, different programs may have different video characteristics, and thus we need to reschedule whenever a change of program occurs on any TV channel. Given that there could be 20–50 TV channels concurrently broadcast over the same wireless medium, there

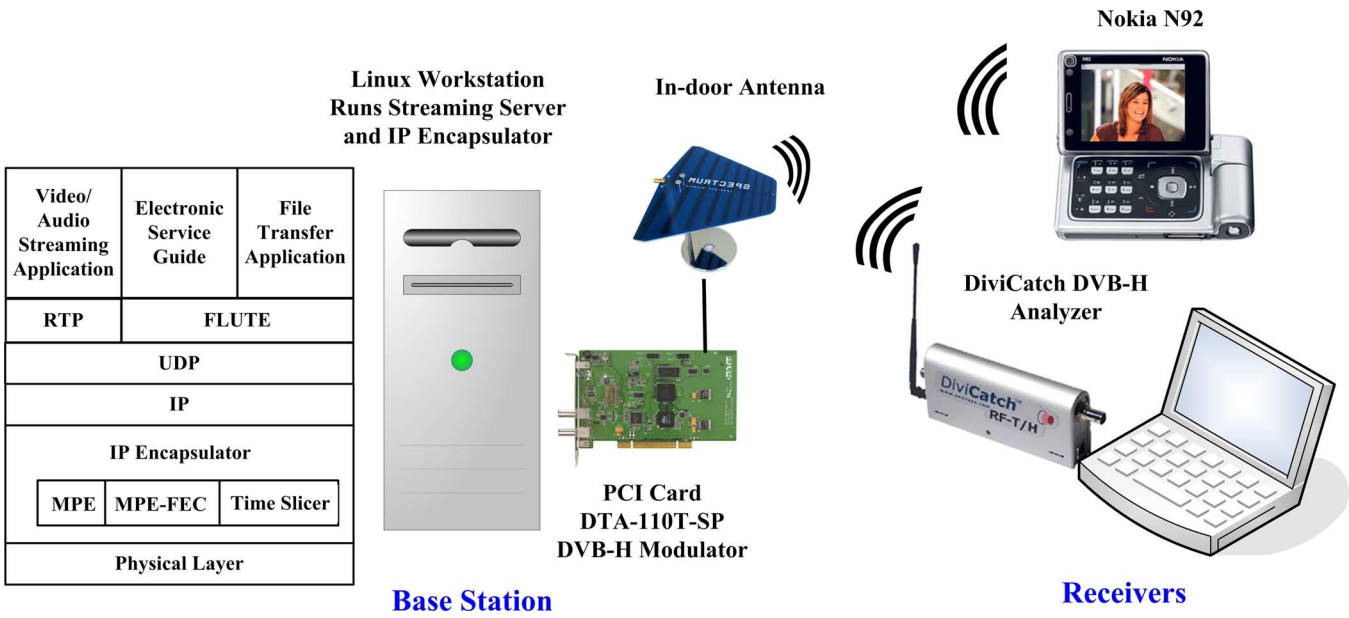


Fig. 9. Setup of the mobile TV (DVB-H) testbed.

will be a very high-level of dynamic changes that must be considered in real time by the burst scheduling algorithm. In addition, even during one TV program on a single TV channel, there are typically many commercial ads. Each commercial ad is a different video clip with different characteristics and bit rate. Furthermore, during a single commercial period, ads (video clips) change very fast, on the order of 30–60 s. Each change *on any TV channel* triggers rescheduling. Hence, the efficiency of the proposed P2OPT scheduling algorithm is important. A brute-force approach to compute burst schedules for 20–50 channels may take prohibitively long time and may not even be doable at all, as the complexity is exponential in the total number of bursts for all channels.

Third, in broadcast networks, video streams are typically obtained from multiple sources, e.g., from news, movies, and sports channels. It is also common that the operators of broadcast networks decode and then reencode video streams to customize them for their networks and meet the limitations on the bandwidth of the allocated wireless spectrum. In addition, network operators have the option to broadcast video streams at different qualities. For example, sports channels can be broadcast at higher quality because they are watched by many users. Our proposed algorithm allows this differentiation in quality.

Finally, our P2OPT algorithm constructs schedules to minimize the energy consumption of mobile devices and to maximize the viewing time of users. In contrast, there are heuristic methods used in practice to construct burst schedules. For example, in Nokia's Mobile Broadcast Solution (MBS) [36], [37], network operators specify an interburst time period ΔT s and a burst size b_s kb for each TV channel s. The base station then schedules a burst every ΔT s for each TV channel s, where the burst size is b_s kb. In such a base station, network operators have to manually choose ΔT and b_s to form a burst schedule that leads to no burst collisions and no buffer violations. This task is time-consuming and error-prone. More importantly, the resulting schedules do not lead to optimum energy savings.

V. Evaluation and Analysis

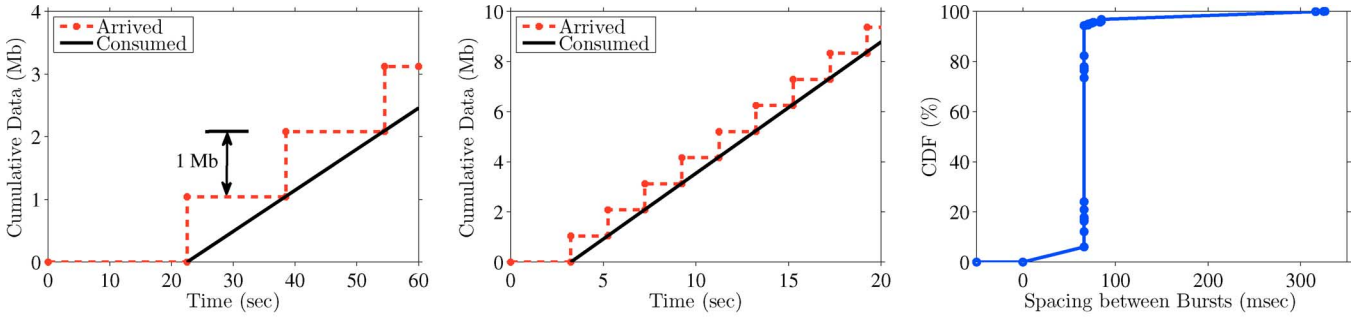
We evaluate the proposed P2OPT algorithm using actual implementation in a mobile TV testbed as well as simulations. We also analyze the limitations of the current practice of assigning the same bit rate for all TV channels, and we experimentally show that our power-of-two bit rate increments solution can result in better viewing experience by reducing quality variation among all TV channels.

A. Setup of the Mobile TV Testbed

We have set up a testbed for DVB-H networks, which provides a realistic platform for analyzing our burst scheduling algorithm. As illustrated in Fig. 9, the testbed has two parts: base station and receivers, which are described below.

The base station is a Linux box in which we installed the RF signal modulator available from [38]. This modulator implements the physical layer of the protocol stack and transmits DVB-H standard-compliant signals via an indoor antenna. We choose an open-source IP encapsulator for the DVB-H systems [39], which encapsulates IP packets of video streams into transport streams. The original IP encapsulator only supports burst schedules with uniform interburst distance for all TV channels. We have redesigned the time slicing module to be well-structured with defined interfaces in order to facilitate implementing and comparing different burst scheduling algorithms. We have implemented our P2OPT burst scheduling algorithm in this IP encapsulator.

We use Nokia N92 and N96 devices as mobile TV receivers. Although these devices help in assessing the visual quality of videos, they do not provide detailed logging functions of the low-level signals, which are needed to evaluate different scheduling algorithms. To address this issue, we added the DVB-H Analyzer available from [40] to the testbed. This analyzer is attached to a PC via a USB port. The analyzer records traffic streams and provides a detailed information on the RF signals



(a) Channel 1, 64 kbps (b) Channel 6, 512 kbps (c) Bursts of all channels

Fig. 10. Experimental validation of the P2OPT algorithm: (a) Channel 1, 64 kbps; (b) Channel 6, 512 kbps; (c) bursts of all channels. (a) and (b) show no over/under flow instances, and (c) shows no burst conflicts.

and DVB-H channels. It also comes with a visualization software that can run on a PC for analysis.

We configure the modulator to use a 5-MHz radio channel with quadrature phase-shift keying (QPSK) modulation. This leads to 5.445-Mbps air medium bandwidth according to the DVB-H standard [6]. We concurrently broadcast nine TV channels using our P2OPT algorithm for 10 min. These TV channels are classified into four classes: two channels at 64 kbps, three channels at 256 kbps, two channels at 512 kbps, and two channels at 1024 kbps. The receiver buffer size is 1 Mb. For each of these TV channels, we set up a streaming server on the base station to send 1-kB IP packets at the specified bit rate. To conduct statistically meaningful performance analysis, we collect detailed event logs from the base station. The logs contain the start and end times (in ms) of broadcasting every burst of data and its size.

We develop software utilities to analyze the logs for three performance metrics: bit rate, time spacing between successive bursts, and energy saving. We compute the bit rate for each TV channel by considering the start times of two consecutive bursts and the burst size. We use the bit rate to verify that our burst scheduling algorithm leads to no buffer under/overflow instances. We compute the time spacing between bursts by first sorting bursts of all TV channels based on their start times. Then, we sequentially compute the time spacing between the start time of a burst and the end time of its immediate, previous burst. We use the time spacing to verify that there are no burst conflicts, as a positive time spacing indicates bursts do not intersect with each other. We compute the energy saving for each TV channel as the ratio between the RF circuit on time and off time. We assume the overhead duration $T_o = 100$ ms.

B. Experimental Results

Experimental Validation of P2OPT Correctness: We first validate the correctness of our P2OPT algorithm from the actual testbed implementation. Fig. 10 demonstrates the correctness of the P2OPT algorithm. In Fig. 10(a) and (b), we plot the cumulative data received by receivers of two sample channels as the time progresses (other results are similar). We also show the consumed data with the time. To account for the worst case, we assume that the receiver starts consuming (playing back) the video data immediately after receiving a burst. The two figures clearly show that there are: 1) no buffer underflow instances as the consumption line never crosses the staircase curve representing arrived data; and 2) no buffer overflow instances as the distance between the data arrival and consumption curves never exceeds the buffer size (1 Mb). Notice that Fig. 10(a) and (b) show only short time periods for clarity, but since these short periods cover multiple frame periods and the burst scheduling is identical in successive frames, the results are the same for the whole streaming period.

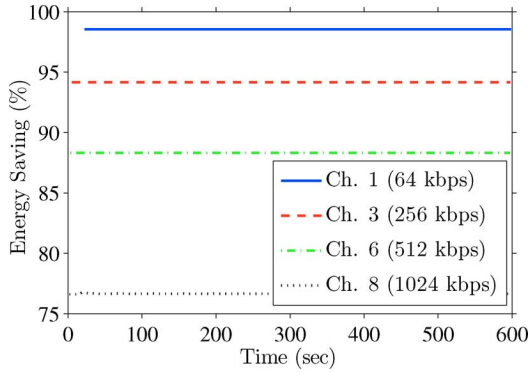


Fig. 11. Energy saving achieved by our P2OPT algorithm for individual TV channels.

In order to show that there are no burst conflicts, we plot the cumulative distribution function (CDF) of the time spacing between successive bursts in Fig. 10(c). This CDF curve is computed from all bursts of all TV channels broadcast during the experiment period (10 min). Negative time spacing would indicate that two bursts are intersecting in time, i.e., burst conflict. This figure clearly shows that our P2OPT algorithm results in no burst conflicts.

Energy Saving Achieved by P2OPT: Next, we report the energy saving achieved by our algorithm. Fig. 11 presents the energy saving of each TV channel. We observe that the energy saving for low-bit-rate TV channels can be as high as 99%, while it is only 76% for high-bit-rate TV channels. This dramatic difference emphasizes the importance of broadcasting TV channels at heterogeneous bit rates: To maximize energy saving on mobile devices, a TV channel should be sent at the lowest bit rate that fulfills its minimum quality requirement.

Optimality of P2OPT: Last, we verify that the energy saving achieved by our P2OPT algorithm is indeed optimal. To do this, we compute the absolute maximum energy saving that can be

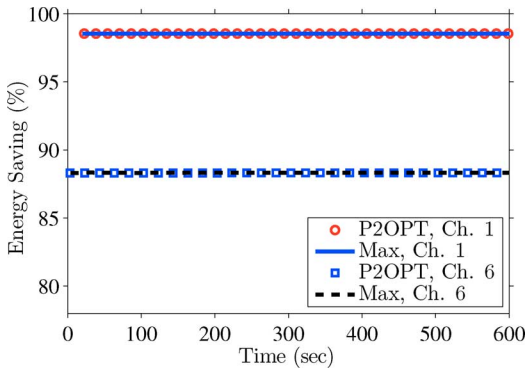


Fig. 12. Optimality of P2OPT: comparing the energy saving achieved by P2OPT against the absolute maximum saving.

achieved by any algorithm for a given TV channel. We compute this maximum by making the base station broadcast only the given TV channel. In this case, the base station easily maximizes the energy saving by allocating the largest burst that can fill the receiver's buffer. The RF circuit of the receiver is then turned off till the data of this burst is consumed. We repeat this experiment nine times, once for each considered TV channel, and we compute the maximum possible energy saving. We then run our algorithm to compute the burst schedule for the nine TV channels, and we make the base station broadcast all of them *concurrently*. We compute the energy saving for each TV channel. Sample results for channels 1 and 6 are presented in Fig. 12; all other results are similar. The figure confirms the optimality of the P2OPT algorithm in terms of energy saving.

C. Simulation Analysis of P2OPT

To evaluate our algorithm under wider ranges of parameters, we have implemented a simulator for mobile TV networks. The simulator captures the important aspects of mobile TV networks that are relevant to the burst scheduling problem, and it abstracts away details such as sending program guide to mobile devices and FEC protection on video packets, which are orthogonal to burst scheduling algorithms.

We simulate a mobile TV network with a wireless medium bandwidth $R = 6.4$ Mbps. We set the receiver buffer size $b = 2$ Mb and the overhead duration $T_o = 100$ ms. We broadcast multiple TV channels using the optimal burst schedules computed by P2OPT. We vary the number of TV channels to achieve different target bandwidth utilization values. We consider bandwidth utilization from 30% to 100% to cover most practical scenarios and to validate the scalability of our P2OPT algorithm. We randomly construct burst scheduling problems for each bandwidth utilization value by selecting TV channel bit rates from 50, 100, 200, 400, and 800 kbps so that the total bit rate does not exceed the bandwidth utilization value. We solve the burst scheduling problems using our P2OPT algorithm and measure the following metrics: energy saving, channel switching delay, and running time. We repeat each experiment 100 times and report the minimum, mean, and maximum values of each performance metric. We run our simulation on a commodity PC running Linux.

Fig. 13 summarizes the performance of the P2OPT algorithm. Fig. 13(a) shows that our algorithm constantly leads to high energy saving—more than 92%. This means that mobile devices can turn off their RF circuits for 92% of the time, which increases battery life and watch time. Fig. 13(b) implies that the channel switching delay is less than 5 s on average. Fig. 13(c) reports the running time of our P2OPT algorithm. This figure shows that our algorithm is very efficient: It terminates in less 30 ms on average under all considered bandwidth utilization levels. Most importantly, Fig. 13(c) implies that our P2OPT algorithm is scalable and can be employed in fully loaded mobile TV networks.

D. Power-of-Two Versus Uniform Bit Rates

We experimentally quantify the potential benefits of classifying TV channels into multiple classes with heterogeneous bit rates. Sending all TV channels at the same bit rate can lead to underutilization of the wireless medium and/or degraded video quality. This is because different video streams need to be encoded at bit rates proportional to their content complexity. For example, to achieve an acceptable video quality, encoding a sports event such as a football game requires higher bit rate than encoding a talk show. In addition to content complexity, video frame rates and display dimensions also have significant impacts on the TV channel bit rates for achieving a target video quality.

To verify the above intuition and quantify its impacts, we encode three video sequences—Foreman, Mobile, and Soccer—at five different frame rates and three screen resolutions. Foreman has a talking person with camera movements, Mobile contains many spatial details, and Soccer features fast motion. Table II lists all considered, 10-s-long, video sequences. We encode each of these sequences into a video stream using the H.264 reference coder [41] with typical configurations used in previous works [42], [43]. To derive the R-D curves, we encode each video sequence at six sampling bit rates from 32 to 1024 kbps. We then decode each coded stream and compute the average video quality quantified by the peak signal-to-noise ratio (PSNR). Fig. 14 shows four sample R-D curves among all considered video sequences. This figure indicates that encoding all video sequences at a uniform bit rate results in significant quality variation. For example, at 512 kbps, the quality difference among different sequences can be as high as 20 dB. Serving all TV channels at a uniform bit rate, therefore, leads to huge quality variations among channels, which degrades user experience especially when they switch channels.

To show that proper selection of bit rates can reduce the quality variation, we consider the following rate allocation problem for multiple concurrent TV channels: Given the R-D characteristic of each TV channel, determine the best power-of-two coding rates for individual TV channels that minimize the video quality variation. In particular, we consider the 10 video sequences listed in Table II and the six coding rates used before and compute the best rate allocation using exhaustive search. We then compare the resulting quality variation against the quality variations if we were broadcasting all TV channels at any of the uniform bit rates. We note that we did not solve this rate allocation problem with more efficient

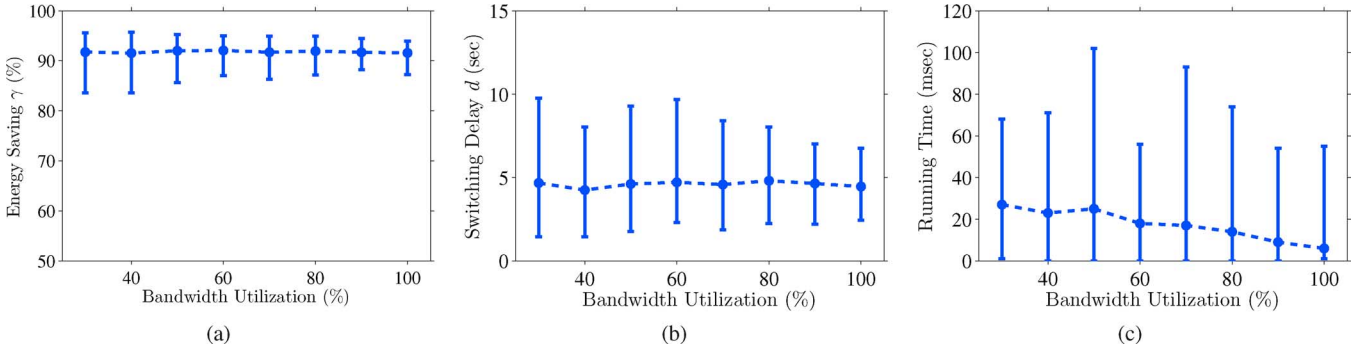


Fig. 13. Performance of the P2OPT algorithm: (a) energy saving, (b) channel switching delay, and (c) running time.

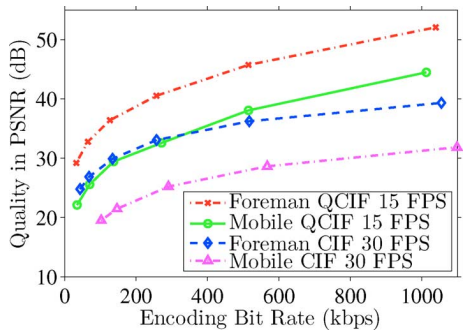


Fig. 14. Sample R-D curves for considered video sequences.

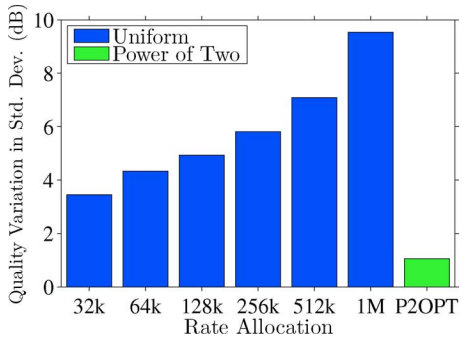


Fig. 15. Comparison of quality variation for uniform and power-of-two bit rates.

TABLE II
LIST OF VIDEO SEQUENCES

Seq.	Res.	FPS	Seq.	Res.	FPS
Foreman	QCIF	7.5	Mobile	QCIF	7.5
Foreman	QCIF	15	Mobile	QCIF	15
Foreman	CIF	15	Mobile	CIF	15
Foreman	CIF	30	Mobile	CIF	30
Soccer	4CIF	30	Soccer	4CIF	60

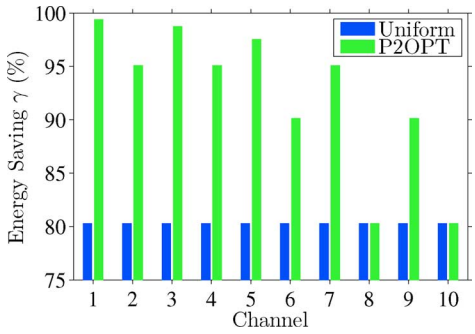


Fig. 16. Comparison of the energy saving achieved by uniform and P2OPT.

algorithms because we only want to quantify the potential benefits of using our burst scheduling algorithm.

Fig. 15 reports the quality variation produced by the optimal power-of-two and uniform bit rate allocations. The quality variation is computed as the standard deviation of the PSNR values of the 10 video sequences when they are encoded at the specified bit rate. This figure shows that broadcasting TV channels at uniform bit rates can lead to high quality variations, up to almost 10 dB in terms of standard deviation. In contrast, broadcasting TV channels at power-of-two bit rates reduces video quality variation: A standard deviation less than 1 dB can be achieved with only six possible encoding rates. As low video quality variation is desirable, service providers who use our P2OPT algorithm to broadcast TV channels at heterogeneous bit rates can provide better service quality and higher user satisfaction, which in turn will increase their revenue.

Finally, we study advantages of using the P2OPT algorithm other than lower quality variation. We consider a network operator who needs to broadcast the aforementioned 10 video sequences at video quality no less than the basic quality of 30 dB in PSNR. If uniform encoding rate is employed, all video sequences are encoded at 1 Mbps to achieve this basic quality. In contrast, using the P2OPT algorithm, video sequences with fewer details/motions can be encoded at as low as 32 kbps while still achieving the basic quality. Consider a mobile TV network with a wireless medium bandwidth $R = 10.556$ Mbps, the network load with uniform encoding rate is $1024 * 10/10556 = 94.73\%$, and the load with P2OPT algorithm is $4096/1056 = 38.50\%$. Clearly, using P2OPT algorithm enables network operators to reduce the network load and increase the number of concurrently broadcast TV channels. Next, we plot the energy saving of individual TV channels in Fig. 16. This figure shows that the P2OPT algorithm allows mobile devices to save more energy. More precisely, broadcasting the video sequences at a uniform encoding rate results in 80.30% energy saving for all TV channels, while broadcasting them using the P2OPT algorithm leads to 92.18% energy saving on average. In summary, in addition to lower quality variation, the P2OPT algorithm enables network operators to broadcast more TV channels and allows mobile devices to save more energy.

VI. Conclusion and Future Work

In this paper, we formulated the energy optimization problem in mobile TV systems. In these systems, a base station broadcasts TV channels in bursts with bit rates much higher than the encoding rates of the video streams. This enables mobile devices to receive a burst of traffic and then turn off their radio frequency circuits till the next burst in order to save energy. We showed that this burst scheduling problem is NP-complete when TV channels have arbitrary bit rates. We then studied a more practical variation of the general problem that enables the use of different classes for video streams, where each class has a different bit rate. The bit rate of class c, r_c, can take any value in the form of $r_c = 2^i \times r_1$, where $i \in \{0, 1, 2, 3, \ldots\}$, and r_1 is the bit rate of the lowest class. r_1 can take any arbitrary bit rate. This classification of TV channels enables the operators of mobile TV networks to offer differentiated services: Higher bit rate classes can broadcast premium services for higher fees. We also showed that this classification can result in better viewing experience by reducing quality variation among all TV channels. We did this by encoding various video sequences with diverse content complexities and empirically analyzing their rate-distortion characteristics.

We proposed an optimal burst scheduling algorithm that runs in $O(S \log S)$ time, where S is the number of TV channels. We proved the correctness and optimality of the proposed algorithm. We derived closed-form equations for the energy saving achieved by our algorithm and the resulting channel switching delay. We numerically analyzed theses equations to demonstrate the existence of a tradeoff between energy saving and channel switching delay, and to show that this tradeoff can be controlled by the receiver's buffer size. In addition, we evaluated the performance of the proposed algorithm using simulation, and we showed that it achieves an average energy saving of more than 92%. Finally, we implemented the proposed algorithm in an actual mobile TV testbed. We analyzed several logs from the testbed and empirically showed that our algorithm can run in real-time, results in no buffer under/overflow instances and no burst conflicts, and yields optimal energy savings.

The work in this paper can be extended in several directions. For example, we are currently designing approximation algorithms for the general burst scheduling problem with arbitrary bit rates.

References

[1] Digital Video Broadcasting – Handheld (DVB-H) Home Page, 2008 [Online]. Available: http://www.dvb-h.org/

[2] G. May, "The IP Datacast system—Overview and mobility aspects," in *Proc. IEEE ISCE*, Reading, U.K., Sep. 2004, pp. 509–514.

[3] M. Kornfeld and G. May, "DVB-H and IP Datacast—Broadcast to handheld devices," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pt. 2, pp. 161–170, Mar. 2007.

[4] *Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H)*, Standard EN 302 304 Ver. 1.1.1, European Telecommunications Standards Institute (ETSI), Nov. 2004.

[5] "FLO technology overview," 2009 [Online]. Available: http://www.mediaflo.com/news/pdf/tech_overview.pdf

[6] *Digital Video Broadcasting (DVB); DVB-H Implementation Guidelines*, Standard EN 102 377 Ver. 1.3.1, European Telecommunications Standards Institute (ETSI), May 2007.

[7] F. Hartung, U. Horn, J. Huschke, M. Kampmann, T. Lohmar, and M. Lundevall, "Delivery of broadcast services in 3G networks," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pt. 2, pp. 188–199, Mar. 2007.

[8] S. Cho, G. Lee, B. Bae, K. Yang, C. Ahn, S. Lee, and C. Ahn, "System and services of terrestrial digital multimedia broadcasting (T-DMB)," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pt. 2, pp. 171–178, Mar. 2007.

[9] M. Takada and M. Saito, "Transmission system for ISDB-T," *Proc. IEEE*, vol. 94, no. 1, pp. 251–256, Jan. 2006.

[10] G. Faria, J. Henriksson, E. Stare, and P. Talmola, "DVB-H: Digital broadcast services to handheld devices," *Proc. IEEE*, vol. 94, no. 1, pp. 194–209, Jan. 2006.

[11] *Radio Broadcasting Systems: Digital Audio Broadcasting (DAB) to Mobile, Portable and Fixed Receivers*, Standard EN 300 401 Ver. 1.3.3, European Telecommunications Standards Institute (ETSI), May 2001.

[12] FLO forum home page, 2008 [Online]. Available: http://www.floforum.org/

[13] "DVB-H global mobile TV: FAQ," 2008 [Online]. Available: http://dvb-h.org/faq.htm

[14] *Digital Video Broadcasting (DVB); Framing Structure, Channel Coding and Modulation for Digital Terrestrial Television*, Standard EN 300 744 Ver. 1.5.1, European Telecommunications Standards Institute (ETSI), Jun. 2004.

[15] E. Tan, L. Guo, S. Chen, and X. Zhang, "PSM-throttling: Minimizing energy consumption for bulk data communications in WLANs," in *Proc. IEEE ICNP*, Beijing, China, Oct. 2007, pp. 123–132.

[16] X. Yang, Y. Song, T. Owens, J. Cosmas, and T. Itagaki, "Performance analysis of time slicing in DVB-H," in *Proc. Joint IST Workshop Mobile Future SympoTIC*, Bratislava, Slovakia, Oct. 2004, pp. 183–186.

[17] E. Balaguer, F. Fitzek, O. Olsen, and M. Gade, "Performance evaluation of power saving strategies for DVB-H services using adaptive MPE-FEC decoding," in *Proc. IEEE PIMRC*, Berlin, Germany, Sep. 2005, pp. 2221–2226.

[18] Q. Zhang, F. Fitzek, and M. Katz, "Cooperative power saving strategies for IP-services supported over DVB-H networks," in *Proc. IEEE WCNC*, Hong Kong, China, Mar. 2007, pp. 4107–4111.

[19] M. Kornfeld, "Optimizing the DVB-H time interleaving scheme on the link layer for high quality mobile broadcasting reception," in *Proc. IEEE ISCE*, Dallas, TX, Jun. 2007, pp. 1–6.

[20] V. Vadakital, M. Hannuksela, and M. Gabbouj, "Time-interleaved simulcast and redundant intra picture insertion for reducing tune-in delay in DVB-H," in *Proc. PV*, Lausanne, Switzerland, Nov. 2007, pp. 123–132.

[21] M. Rezaei, M. Hannuksela, and M. Gabbouj, "Tune-in time reduction in video streaming over DVB-H," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pt. 2, pp. 320–328, Mar. 2007.

[22] M. Rezaei, I. Bouazizi, and M. Gabbouj, "Joint video coding and statistical multiplexing for broadcasting over DVB-H channels," *IEEE Trans. Multimedia*, vol. 10, pp. 1455–1464, Dec. 2008.

[23] M. Rezaei, M. Hannuksela, and M. Gabbouj, "Video encoding and splicing for tune-in time reduction in IP datacasting (IPDC) over DVB-H," in *Proc. IEEE ICME*, Toronto, ON, Canada, Jul. 2006, pp. 601–604.

[24] M. Rezaei, M. Hannuksela, and M. Gabbouj, "Video splicing and fuzzy rate control in IP multi-protocol encapsulator for tune-in time reduction in IP datacasting (IPDC) over DVB-H," in *Proc. IEEE ICIP*, Atlanta, GA, Oct. 2006, pp. 3041–3044.

[25] T. Wiegand, G. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[26] "Advanced video coding for generic audiovisual services," Joint Video Team, 2005, ITU-T Rec. H.264 & ISO/IEC 14496–10 AVC.

[27] *Digital Video Broadcasting (DVB); IP Datacast Over DVB-H: Content Delivery Protocol*, Standard EN 102 472 Ver. 1.1.1, European Telecommunications Standards Institute (ETSI), Nov. 2006.

[28] *Digital Video Broadcasting (DVB); DVB Specification for Data Broadcasting*, Standard EN 301 192 Ver. 1.4.1, European Telecommunications Standards Institute (ETSI), Jun. 2004.

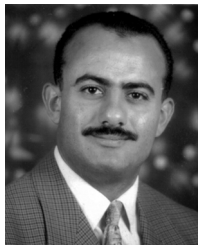
[29] M. Garey and D. Johnson, *Computers and Intractability: A Guide to The Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[30] P. Brucker, *Scheduling Algorithms*, 4th ed. New York: Springer, 2004.

[31] Y. Bartal, S. Leonardi, and G. S. R. Sitters, "On the value of preemption in scheduling," in *Proc. Workshop APPROX*, Barcelona, Spain, Aug. 2006, pp. 39–48.

[32] O. Braun and G. Schmidt, "Parallel processor scheduling with limited number of preemptions," *SIAM J. Comput.*, vol. 32, no. 3, pp. 671–680, 2003.

[33] R. Motwani, S. Phillips, and E. Torng, "Nonclairvoyant scheduling," *Theor. Comput. Sci.*, vol. 130, no. 1, pp. 17–47, Aug. 1994.
[34] U. Schwiegeishohn, "Preemptive weighted completion time scheduling of parallel jobs," in *Proc. ESA*, Barcelona, Spain, Sep. 1996, pp. 39–51.
[35] P. Chou, "Streaming media on demand and live broadcast," in *Multimedia Over IP and Wireless Networks*, M. van der Schaar and P. Chou, Eds. New York: Academic, 2007, ch. 14, pp. 453–502.
[36] "Nokia mobile broadcast solution," Feb. 2009 [Online]. Available: http://www.mobiletv.nokia.com/solutions/mbs/
[37] Private communication with Nokia's engineers managing mobile TV base stations, Dec. 2008.
[38] Dektec DTA-110T PCI modulator, 2008 [Online]. Available: http://www.dektec.com/Products/DTA-110T/
[39] "FATCAPS project," 2008 [Online]. Available: http://amuse.ftw.at/downloads/encapsulator
[40] "Divi catch RF-T/H transport stream analyzer," 2008 [Online]. Available: http://www.enensys.com/
[41] "Joint scalable video model reference software. ver. JSVM 14.0, Joint Video Team, May 2008.
[42] P. Lambert, W. de Neve, P. de Neve, I. Moerman, P. Demeester, and R. van de Walle, "Rate-distortion performance of H.264/AVC compared to state-of-the-art video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 1, pp. 134–140, Jan. 2006.
[43] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 688–703, Jul. 2003.



Mohamed Hefeeda (S'01–M'04) received the B.Sc. and M.Sc. degrees from Mansoura University, Mansoura, Egypt, in 1997 and 1994, respectively, and the Ph.D. degree from Purdue University, West Lafayette, IN, in 2004.

He is an Assistant Professor with the School of Computing Science, Simon Fraser University, Surrey, BC, Canada, where he leads the Network Systems Lab. His research interests include multimedia networking over wired and wireless networks, peer-to-peer systems, network security, and wireless sensor networks. He has served on more than 20 technical program committees of major conferences in his research areas, including ACM Multimedia, ACM Multimedia Systems, ACM/SPIE Multimedia Computing and Networking (MMCN), IEEE Conference on Network Protocols (ICNP), and IEEE Conference on Communications (ICC). He is an Associate Editor of the *International Journal of Advanced Media and Communication* and the Guest Editor of that journal's special issue on high-quality multimedia streaming in P2P environments. His research on optimizing the performance of mobile multimedia systems has been featured in the *ACM Tech News* (July 1, 2009), CTV British Columbia News (June 26, 2009), and Simon Fraser University's News (May 28, 2009).

Dr. Hefeeda is a Member of the ACM Special Interest Groups on Data Communications (SIGCOMM) and Multimedia (SIGMM). His paper on the hardness of optimally broadcasting multiple video streams with different bit rates won the Best Paper Award in the IEEE Innovations 2008 conference. In addition to publications, he and his students develop actual systems, such as PROMISE, pCache, svcAuth, pCDN, and mobile TV testbed, and contribute the source code to the research community. The mobile TV testbed software developed by his group won the Best Technical Demo Award in the ACM Multimedia 2008 conference. The pCDN (Peer-assisted Content Distribution Network) system is cosponsored by the Canadian Broadcasting Corporation (CBC) and is currently being used to distribute high-quality multimedia content for some of the CBC's Internet streaming services in efficient and cost-effective manner.

Cheng-Hsin Hsu (S'09) received the B.Sc. and M.Sc. degrees from National Chung-Cheng University, Min-Hsiung, Taiwan, in 2000 and 1996, respectively, and the M.Eng. degree from the University of Maryland, College Park in 2003.

He is working toward the Ph.D. degree in the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. His research interests are in the area of multimedia networking, broadcast networks, wireless networks, and peer-to-peer networks.