# A Bargaining Theoretic Approach to Quality-Fair System Resource Allocation for Multiple Decoding Tasks

Nicholas H. Mastronarde, *Student Member, IEEE*, and Mihaela van der Schaar, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a new resource allocation framework for multimedia systems that perform multiple simultaneous video decoding tasks. We jointly consider the available system resources (e.g., processor cycles) and the video decoding task's characteristics such as the sequence's content, the bit-rate, and the group of pictures (GOP) structure, in order to determine a fair and optimal resource allocation. To this end, we derive a quality-complexity model that determines the quality [in terms of peak signal-to-noise ratio (PSNR)] that a task can achieve given a certain system resource allocation. We use these quality-complexity models to determine a quality-fair and Pareto-optimal resource allocation using the Kalai–Smorodinski Bargaining Solution (KSBS) from axiomatic bargaining theory. The KSBS explicitly considers the resulting multimedia quality when performing a resource allocation and distributes quality-domain penalties proportional to the difference between each video decoding task's maximum and minimum quality requirements. We compare the KSBS with other fairness policies in the literature and find that, because it explicitly considers multimedia quality, it provides significantly fairer resource allocations in terms of the resulting PSNR compared with policies that operate solely in the resource domain. To weight the quality impact of the resource allocations to the different decoding tasks depending on application-specific requirements or user preferences, we generalize the existing KSBS solution by introducing bargaining powers based on each video sequence's motion and texture characteristics.**

***Index Terms*—Bargaining powers, complexity scalability, Kalai–Smorodinski bargaining solution, multimedia systems, quality-complexity models, system resource management, video decoding complexity.**

## I. Introduction

DECODING multiple video streams simultaneously is necessary in a variety of applications, including multipoint video conferencing, video surveillance from multiple cameras, and picture-in-picture. In recent years, such applications are becoming more and more important across different platforms such as desktop PCs, laptops, PDAs, cellular phones, and portable music players. These devices, however, have a wide range of processing capabilities, and power, energy, and resource constraints that are not ideal for latency-sensitive and computationally intensive video decoding tasks. Compounding this problem, these applications may require multiple such tasks to run simultaneously on one resource-constrained processor. In this scenario, the need for fair system resource allocation becomes important to ensure that all video decoding tasks attain reasonable quality. Video decoding applications allow for flexible tradeoffs between output quality and resource usage that can be exploited in order to meet real-time latency constraints when resources are scarce (e.g., when many tasks compete for processor time or when few tasks require a long lifetime on a battery-powered device). Therefore, as discussed in [1], system resource allocation for video decoding is not a problem that can only be addressed at the OS or hardware level. Instead, it is a cross-layer problem involving the application, operating system and hardware because it requires joint consideration of available resources (e.g., processor cycles) and a video decoding task's characteristics such as the sequence's content, the bit-rate, and the GOP structure, all of which determine different levels of complexity and, concomitantly, video decoding quality.

In this paper, we jointly consider the system resource constraints and the quality–complexity tradeoffs available to each video decoding task in order to determine a fair and optimal resource allocation. Unlike conventional fair resource allocation schemes, where system resources are distributed based on resource-domain metrics, we deploy a bargaining-theoretic policy that distributes limited resources fairly and optimally with respect to a video quality metric (i.e., PSNR). Specifically, we deploy the Kalai–Smorodinski Bargaining Solution (KSBS) [18], which distributes penalties proportional to the difference between a task's maximum *achievable* quality and its minimum *required* quality. Using a proposed quality-complexity model, the quality-domain bargaining solution is then converted into the quality-fair resource allocation.

Our contributions are as follows.

- We propose a new general complexity traffic modeling framework for multimedia tasks. We characterize the traffic with five parameters that together we designate as a task's *complexity specification* (CSPEC).
- To enable the complexity scalable admission control, where multiple video tasks can be admitted simultaneously, we introduce *complexity strategies* that encapsulate the complexity- and quality-scalable adaptation points that a video decoding task can select to match its workload to the available resources. Each complexity strategy has a corresponding CSPEC, and we label the aggregation of all CSPEC adaptation points as a task's *scalable* CSPEC. Each task uses its scalable CSPEC to negotiate for its fair

Manuscript received October 28, 2006; revised April 14, 2007. This work was supported by the National Science Foundation under Career Award CCF-0541453 and Grant CNS-0509522 and by grants from UC Micro. This paper was recommended by Associate Editor Z. He.

The authors are with the Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095-1594 USA (e-mail: nhmastro@ee.ucla.edu; mihaela@ee.ucla.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/TCSVT.2008.918440

share of resources with the resource manager. We derive a model that allows us to determine the quality–complexity tradeoffs for different video sequences.

- We generalize existing bargaining-theoretic allocation policies to determine a quality-fair resource allocation for video decoding tasks sharing a single resource constrained processor. Additionally, we compare the proposed solution to other resource allocation solutions in the literature.

The remainder of this paper is organized as follows. In Section II, we define a task's CSPEC and scalable CSPEC and show how various complexity strategies for decoding result in different complexity levels. In Section III, we introduce some resource management preliminaries and summarize several existing fairness policies for resource allocation that are implemented in the resource-domain. In Section IV, we define the quality–complexity function and illustrate the properties of the resulting feasible quality sets. We then formulate the bargaining problem and define the KSBS. We then generalize the KSBS for our problem and we summarize the steps required to implement the proposed bargaining-based resource allocation system. In Section V, we present our experimental results, and we conclude in Section VI.

## II. Adaptive Workload Modeling

### A. Complexity Scalability

A majority of state-of-the-art video coders can create bitstreams that can be decoded at different quality and complexity levels. In this paper, we focus on MCTF-based temporal layers. However, our approach is general and not limited by the particular layering/partitioning choice or video coder, as will be shown in some of our illustrative examples. An MCTF structure with a Haar 5/3 temporal decomposition [3] and $T$ temporal levels, for example, can be partitioned into $T$ layers (i.e., we can decode $l = 1, \ldots, T$ layers). We use the notation $\mathrm{H}_{\tau,\kappa}$ and $L_{\tau,\kappa}$ to indicate the $\kappa$th H and $L$ frame of temporal level $\tau$, respectively, where $1 \leq \tau \leq T$ and $0 \leq \kappa < 2^{T-\tau}$. By convention, we denote the original encoded frames $A_{\tau,\kappa}$ as belonging to temporal level $\tau = 0$. Table I shows the decoding deadlines for the frames of an MCTF structure with $T = 4$ when it is partitioned into $l =$ 1, 2, 3, and 4 temporal layers. In Table I, $t_0$ denotes the display deadline of the pair of original frames $\{A_0, A_1\}$ and the deadline of each subsequent pair is $2\Delta$ s after the previous pair's deadline. The value of $\Delta$ depends on the encoded frame-rate and, for our experiments, is set as $\Delta = 1/30 = 0.0333$ s to correspond to the 30-Hz frame-rate of the original encoded sequences. We note that the above analysis is related to the temporal decomposition used (5/3 Haar filter) and can differ for other temporal decomposition structures [3].

To illustrate several key properties of the video decoding workload traffic that make it challenging to characterize and model, example traffic for a variety of scenarios is shown in Fig. 1. We will describe each plot in Fig. 1 individually.

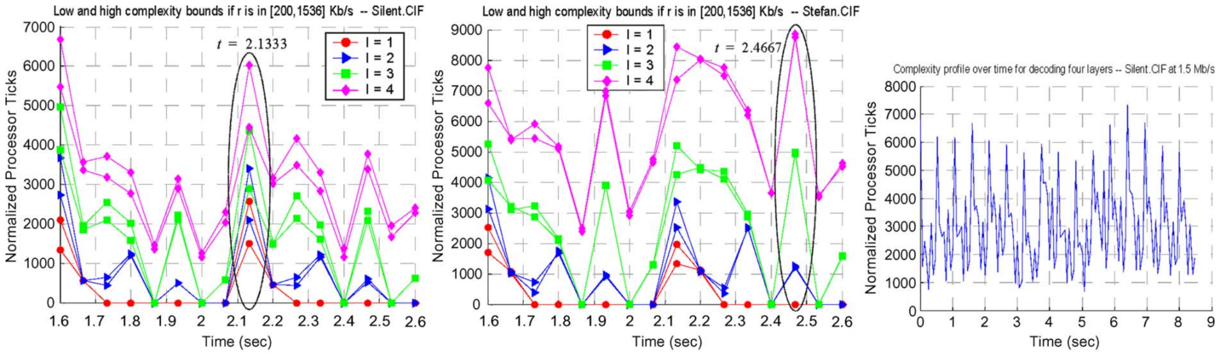
TABLE I
Display Deadlines for Decoding Frames Partitioned Into $T = 4$ Temporal Layers Using 5/3 Haar Filter. Notice That Decoding $l$ Layers Requires Decoding the Frames in Columns $1, \ldots, l - 1$ as Well as the Frames in Column $l$

| Display Deadline | Number of Decoded Layers ($l$ ) | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| $t_0$ | $L_{4,0}$, $\mathrm{H}_{4,0}$, $L_{3,0}$ | $\mathrm{H}_{3,0}$, $L_{2,0}$ | $\mathrm{H}_{2,0}$, $L_{1,0}$ | $\mathrm{H}_{1,0}$, $A_0$, $A_1$ |
| $t_0$ + 2Δ | $L_{3,1}$ | - | $L_{1,1}$, $L_{1,2}$ | $\mathrm{H}_{1,1}$, $A_2$, $A_3$ |
| $t_0$ + 4Δ | - | $\mathrm{H}_{3,1}$ | $\mathrm{H}_{1,2}$, $A_4$, $A_5$ | $\mathrm{H}_{1,2}$, $A_4$, $A_5$ |
| $t_0$ + 6Δ | - | $L_{2,1}$, $L_{2,2}$ | $\mathrm{H}_{2,2}$ | $\mathrm{H}_{1,3}$, $A_6$, $A_7$ |
| $t_0$ + 8Δ | - | - | - | $\mathrm{H}_{1,4}$, $A_8$, $A_9$ |
| $t_0$ + 10Δ | - | $L_{2,3}$ | $\mathrm{H}_{2,3}$, $L_{1,5}$, $L_{1,6}$ | $\mathrm{H}_{1,5}$, $A_{10}$, $A_{11}$ |
| $t_0$ + 12Δ | - | - | - | $\mathrm{H}_{1,6}$, $A_{12}$, $A_{13}$ |
| $t_0$ + 14Δ | - | - | $L_{1,7}$ | $\mathrm{H}_{1,7}$, $A_{14}$, $A_{15}$ |

To measure the video decoding workload characteristics in a variety of scenarios, we profiled a C++ implementation of the $t + 2D$ MCTF coder in [3] with system calls that query the internal processor counter to obtain high-resolution timing measurements. All measurements in this paper are obtained from the decoder implemented in C++ and executed on a Dell Pentium IV system running Microsoft Windows XP safe mode with only a command prompt to ensure that context switches and interrupting processes are kept at a minimum. We note that, in Fig. 1(a)–(c), a data point at time $t_0 + 2n\Delta$, $n \in \mathbb{Z}$, represents the number of normalized processor cycles required during the interval of time $(t_0 + 2(n-1)\Delta, t_0 + 2n\Delta]$ to decode the appropriate set of frames before their decoding/display deadline at $t_0 + 2n\Delta$. The next paragraphs illustrate several important properties of the workload traffic.

- *Workload depends on the sequence characteristics*: To illustrate the impact of varying video source characteristics[1] $\xi$ on the decoding workload across a GOP, Fig. 1(a) and (b) show the upper and lower bounds on the workload traffic for GOPs four and five (frames 49–80) of the *Silent* and *Stefan* sequences, respectively, at CIF resolution, encoded frame-rates of 30 Hz, and with rate $r \in [200, 1536]$ kb/s. Comparing Fig. 1(b) and Fig. 1(a), it is clear that decoding *Stefan* is significantly more computationally complex than decoding *Silent* for most choices of $l$. This observation is congruent with the intuition that *Stefan*'s intense motion characteristics should generally yield heavier peak and mean computational workloads at the CPU. Hence, the complexity depends on a particular video source's characteristics $\xi$.
- *Workload is highly time-varying*: From Table I, it is clear that the number of frames that must be decoded at each deadline is not distributed evenly over the duration of the GOP. For example, when decoding all the layers, only 3 frames require decoding or reconstruction during the time-interval $(t_0 + 6\Delta, t_0 + 8\Delta]$ while 10 frames are decoded or reconstructed during the time-interval $(t_0 - 2\Delta, t_0]$. The time-varying workload curves in Fig. 1(a) and (b) reflect this unbalanced workload distribution within a GOP. Fig. 1(c), on the other hand, shows how decoding workloads are also typically time-varying across several GOPs. This is the consequence of changes in motion and texture characteristics over the duration of the video sequence.
- *Workload is rate-dependent*: Fig. 1(a) and (b) also illustrates how rate affects the decoding complexity. Notice that adjusting the rate can have a significant impact on the complexity, particularly at the first decoding deadline in a GOP where the base-layer frames (i.e., $L_{T,0}$, $\mathrm{H}_{T,0}$) containing

[1] Source characteristics include, e.g., high motion, detailed textures, and object occlusion.



(a) (b) (c)

Fig. 1. (a), (b) Typical complexity profile for decoding $l = 1, 2, 3$, or 4 temporal layers (with $T = 4$ temporal level MCTF structure) of one GOP of the *Silent* and *Stefan* sequences, respectively. Includes upper and lower bounds on the decoding complexity when decoding $l = 1, 2, 3$, or 4 layers with rate $r \in [200, 1536]$ kb/s. Notice that the $y$-axis scales are different for each sequence. (c) Typical complexity profile over many GOPs when decoding $l = 4$ layers of the *Silent* sequence.

most of the texture information are decoded. For example, in Fig. 1(a), at the circled complexity measurements at $t = 2.1333$ s (i.e., the first decoding deadline for GOP 5), adjusting the rate $r \in [200, 1536]$ causes the upper-bound complexity resulting from decoding $l$ layers (for $l = 1$, 2) to be *above* the lower-bound complexity for decoding $l + 1$ layers. In other cases, when minimal residual texture information is decoded, rate-independent motion compensation operations dominate the complexity. In these cases, adjusting $r$ insignificantly influences the complexity (see the circled complexity measurement at $t = 2.4667$ s in Fig. 1(b)). Nevertheless, such significant peak workload variation induced by adjusting the rate $r$ in the former case illustrates how adapting the rate can increase a task's chances for admission into a system with limited resources.

- *Decoding different layers leads to complexity scalability*: Fig. 1(a) and (b) also illustrates that significantly reduced workloads in terms of both peak and mean resource requirements can be achieved by decoding less layers. The wide range of complexity scalability enabled by decoding various layers is important in an admission control scenario where a task may not be allocated any processor resources if it cannot adapt its workload to the resources available to it.

As mentioned earlier, the workload distribution within a GOP depends on the video encoding parameters, including the number of temporal levels $T$ and the choice of filter used in the temporal decomposition. It has been shown [8] that based on the video source characteristics, $\xi$, it may be desirable to chose different encoding parameters that yield different levels of complexity-scalability. Hence, our choice of the number of decoded layers $l$ and rate $r$ are dependent on $\xi$ because, depending on a task's video sequence characteristics, different $l$ and $r$ pairs may be required to meet the same resource constraints. For notational simplicity, we do not explicitly indicate this dependence.

Based on the above observations pertaining to the time-varying decoding workloads, a workload traffic model that captures all of these characteristics and can provide quality and latency guarantees through an admission control process is highly desirable. We present such a model in the next subsection. Note that, while the above observations were made for one particular coder, similar observations can be made for other coders such as H.264/AVC and MPEG-4 using the same methodology.

### B. Characterizing Video Decoding Workload Traffic

In order to capture a video decoding task's time-varying and bursty resource requirements (see Section II-A), we model the decoding workload traffic with a twin leaky bucket. We assume that the task decodes $l$ layers at rate $r$. The important model parameters are the Peak Workload $P(l,r)$ (cycles/second), Mean Workload $\rho(l,r)$ (cycles/second), Maximum Burst Size $\sigma(l,r)$ (cycles), and Delay $d(l,r)$ (seconds). $d(l,r)$ is set based on the application requirements or user preferences. The remaining parameters can be determined using offline modeling, training, or profiling, followed by real-time classification [2], [7], [9]. We would like to develop a model that considers both the Peak Workload and the Mean Workload for two reasons. First, considering just the Peak Workload $P(l,r)$ results in over conservative worst-case complexity estimates that inefficiently use the CPU bandwidth. Conversely, considering only the Mean Workload $\rho(l,r)$ under allocates CPU bandwidth during time intervals where the workload exceeds the Mean Workload and therefore results in missed decoding deadlines and, consequently, frame drops. By enforcing the (small) Delay $d(l,r)$ on all display deadlines (i.e., the display deadlines in Table I become $t_0 + 2\Delta n + d(l,r)$ for $n = 0, \ldots, 7$) the bursty workload can be smoothed to reduce the peak computational complexity. This delay parameter was first introduced in [10] in order to reduce the peak computational capacity required by a device to decode a particular bit-stream. In this paper, we expand on the concept by also exploiting complexity-scalability which, given a fixed Delay parameter, allows a task to adapt its complexity to match available resources.

Based on a twin leaky bucket analysis, the CPU Bandwidth Demand for decoding $l$ layers at rate $r$ with Delay $d(l,r)$ is

$$g(l,r) = \frac{P(l,r)}{1 + d(l,r)\left[P(l,r) - \rho(l,r)\right]\sigma^{-1}(l,r)}. \quad (1)$$

Together, the token bucket parameters and the CPU Bandwidth Demand determine the task's *Complexity Specification* (CSPEC) denoted as the set $\mathbf{g}(l,r) =$

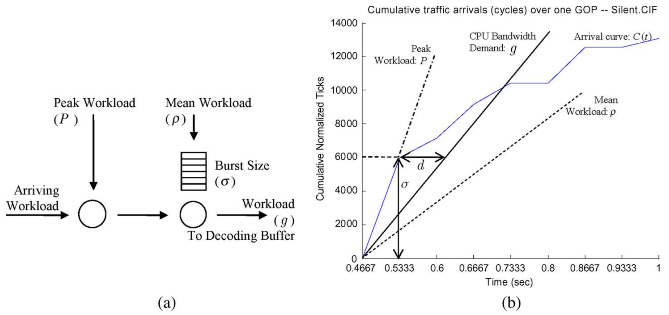


Fig. 2. (a) Twin leaky bucket modeling of workload traffic. (b) Arrival curve for one GOP of the *Silent* sequence for decoding $l = 3$ layers with $T = 4$ and calculation of CPU Bandwidth Demand.

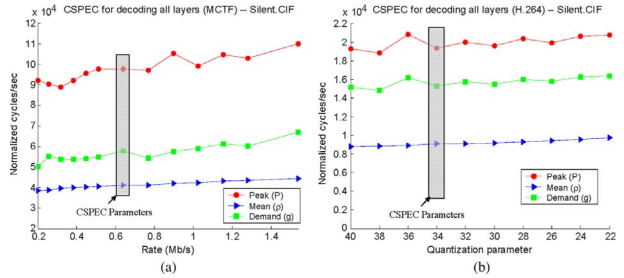


Fig. 3. (a) CSPEC parameters for MCTF based decoder: 16 GOP normalized computation workload for decoding $l = 4$ layers of the *Silent* sequence at 13 rates between 200 kb/s and 1.5 Mb/s with Delay $d(l,r) = 2\Delta$. (b) CSPEC parameters for H.264–based decoder: 128 frame normalized computation workload for decoding all layers of the *Silent* sequence with quantization parameters between 22 and 40 with Delay $d(l,r) = 0.5\Delta$. GOP structure used for H.264–based measurements has one I-frame for every 3 P-frames, and 3 B-frames for each I/P-frame.

$\{g(l,r), P(l,r), \rho(l,r), d(l,r)\}$, where the Maximum Burst Size $\sigma(l,r)$ is omitted because it can be determined as

$$\sigma(l,r) = 2\Delta \cdot P(l,r) \text{ cycles}. \tag{2}$$

Intuitively, $\sigma(l,r)$ can be expressed as in (2) because it corresponds to the maximum processor workload during any $2\Delta$ second time interval. We note that the CPU Bandwidth Demand $g(l,r)$ in (1), corresponding to the smoothed workload, is the parameter we use for resource allocation and admission control because it resolves the aforementioned issues with the Peak and Mean Workloads. Fig. 2(a) illustrates how the first and second token buckets regulate the arrival of workload traffic to the decoding buffer. Fig. 2(b) details how the various CSPEC parameters are determined using the cumulative workload traffic arrival curve $C(t)$. Note that, in Fig. 2(b), $P$, $\rho$, and $g$ are the slopes of the respective lines.

Example values for $P(l,r)$, $\rho(l,r)$, and $g(l,r)$ for the workload traffic of the first 16 GOPs of the *Silent* sequence are shown in Fig. 3 for 13 rates between 200 kb/s and 1.5 Mb/s, $l = 4$, and Delay $d(l,r) = 2\Delta$. Using the data in Fig. 3, the Maximum Burst Size can be determined using (2). The thin vertical rectangle labeled “CSPEC parameters” in Fig. 3 is used to emphasize that the enclosed parameters constitute the CSPEC $\mathbf{g}(l,r)$ associated with a particular vector $[l,r][= (4, 640 \text{ kb/s})$, in Fig. 3(a)]. Importantly, this CSPEC definition can be used to characterize the decoding workloads of other coders. For example, Fig. 3(b) shows the CSPEC parameters for the *Silent* sequence determined using an H.264/AVC-based coder [2] with quantization parameters between 22 and 40 and with Delay $d(l,r) = 0.5\Delta$. The CSPEC can also be adapted to characterize video encoding workloads.

### C. Complexity Strategies and the Scalable Complexity Specification

Depending on the sequence characteristics, $\xi_i$, task $i$ might deploy different decoding strategies in order to adapt its CSPEC and negotiate with the *Resource Manager* for its fair share of resources. Note that each video coder and task can implement its own decoding strategy set. Here, for illustration purposes, we define $\mathbf{a}_i = [l_i, r_i] \in \mathcal{A}_i$, $i = 1, \ldots, M$ as a *complexity strategy* vector in the feasible set of complexity strategies for task $i$, where $\mathcal{A}_i = \mathcal{A}_i^{\text{LAYER}} \times \mathcal{A}_i^{\text{RATE}}$ and $\mathcal{A}_i^{\text{LAYER}} = \{l_i^1, \ldots, l_i^{N_i^{\text{LAYER}}}\}$ and $\mathcal{A}_i^{\text{RATE}} = \{r_i^1, \ldots, r_i^{N_i^{\text{RATE}}}\}$ denote the decoding strategy spaces enabling spatio-temporal decoding tradeoffs and bit-rate adaptations (corresponding to the SNR scalability), respectively. We denote the cardinalities of the strategy spaces as $N_i^{\text{LAYER}} = |\mathcal{A}_i^{\text{LAYER}}|$ and $N_i^{\text{RATE}} = |\mathcal{A}_i^{\text{RATE}}|$. Note that the feasible complexity strategies and decoding strategy spaces for the $i$th task depend on its video source characteristics, $\xi_i$, for the reasons described in Section II-A. For notational simplicity, we do not explicitly indicate this dependence. By selecting the complexity strategy vector $\mathbf{a}_i = [l_i, r_i] \in \mathcal{A}_i$, a task operates at one of the $N_i = N_i^{\text{LAYER}} \cdot N_i^{\text{RATE}} = |\mathcal{A}_i|$ feasible complexity levels that define its *Scalable* CSPEC. Formally, we define the $i$th task’s Scalable CSPEC as the set

$$\begin{aligned} \mathcal{G}_i(\mathcal{A}_i) &= \{\mathbf{g}_i(\mathbf{a}_i) | \mathbf{a}_i \in \mathcal{A}_i\} \\ &= \{\{g(\mathbf{a}_i), P(\mathbf{a}_i), \rho(\mathbf{a}_i), d(\mathbf{a}_i)\} \, | \mathbf{a}_i \in \mathcal{A}_i\} \end{aligned} \tag{3}$$

where $g_i(\mathbf{a}_i)$, rewritten using the complexity strategy notation, is determined by (1) using the corresponding Peak Workload $P_i(\mathbf{a}_i)$, Mean Workload $\rho_i(\mathbf{a}_i)$, Maximum Burst Size $\sigma_i(\mathbf{a}_i)$ determined by (2), and the Delay $d_i(\mathbf{a}_i)$, which relaxes the decoding/display deadlines.

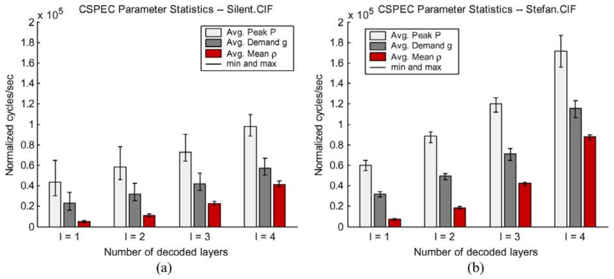
Fig. 4(a) and (b) illustrates example Scalable CSPEC statistics for the first 256 frames of the *Silent* and *Stefan* sequences, respectively (CIF resolution, 30-Hz encoded frame-rate). The Delay $d(\mathbf{a}_i)$ is set to $2\Delta$. The example operating points, $\mathbf{g}_i(\mathbf{a}_i) \in \mathcal{G}_i(\mathcal{A}_i)$, from which the statistics are gathered are defined by the $N_i = 52$ complexity strategies $\mathbf{a}_i = [l_i, r_i] \in \mathcal{A}_i$ with the vector components $l_i$ and $r_i$ in their respective strategy spaces

$$\mathcal{A}_i^{\text{LAYER}} = \{1,2,3,4\}, N_i^{\text{LAYER}} = 4 \tag{4}$$

$$\begin{aligned} \mathcal{A}_i^{\text{RATE}} &= \{200, 256, 320, 384, 448, 512, 640, 768, \\ &\qquad 896, 1024, 1152, 1280, 1536\}\text{kb/s} \\ N_i^{\text{RATE}} &= 13. \end{aligned} \tag{5}$$

The solid bars in Fig. 4(a) and (b) are the averaged value of the corresponding parameter (i.e., $P(\mathbf{a}_i)$, $g(\mathbf{a}_i)$, or $\rho(\mathbf{a}_i)$ over 13 measurements taken for bit-rates between 200 kb/s and 1.5 Mb/s. The error bars show the maximum and minimum value of the corresponding parameter for the 13 measurements.

The solid bars in Fig. 4(a) and (b) illustrate that, by selecting $l_i \in \mathcal{A}_i^{\text{LAYER}}$ as defined in (4), a task’s CPU Bandwidth Demand $g(\mathbf{a}_i)$ can be scaled to below half of its maximum.



(a) (b)

Fig. 4. CSPEC parameter statistics for various complexity strategies based on 16 GOP normalized computation workload with Delay $d(l, r) = 2\Delta$, Peak CPU Bandwidth $P(l, r)$, Mean CPU Bandwidth $\rho(l, r)$, and CPU Bandwidth Demand $g(l, r)$ for decoding $l = 1, 2, 3$, or 4 layers. The solid bars are the averaged value of the corresponding parameter over 13 measurements taken for bit-rates between 200 kb/s and 1.5 Mb/s. The error bars show the maximum and minimum value of the corresponding parameter for the 13 measurements. (a) *Silent* sequence. (b) *Stefan* sequence.

Additionally, observing the error bars, it is clear that adjusting $r_i \in \mathcal{A}_i^{\mathrm{RATE}}$ as defined in (5) yields finer complexity scalability by approximately 10%–40% (depending on the sequence) of the maximum CPU Bandwidth Demand for a fixed value of $l_i$. Clearly, the feasible set of complexity strategies defines a wide range of complexity scalability. Therefore, complexity strategies are essential when tasks negotiate for limited system resources with the resource manager. Moreover, the Delay $d(\mathbf{a}_i) = 2\Delta$ significantly reduces the processing rate required to meet all decoding and display deadlines. Specifically, compared with the Peak CPU Bandwidth $P(\mathbf{a}_i)$ that is required to meet all task deadlines when $d(\mathbf{a}_i) = 0$, the CPU Bandwidth Demand $g(\mathbf{a}_i)$ for $d(\mathbf{a}_i) = 2\Delta$ is lower by ~30% on average and at most by nearly 40%. In scenarios where a device's limited processing capacity preclude the admission of one or more tasks with high peak requirements, even small delays can dramatically improve the number of admitted tasks and each task's quality. We note that the Delay $d(\mathbf{a}_i)$ can be increased further to achieve greater reduction in the CPU Bandwidth Demand, however, this requires larger memory buffers [10].

In a scenario where task $i$ is given a static resource allocation $c_i$ (in processor cycles), we wish to determine the optimal complexity strategy that maximizes the task's quality under the complexity constraint. Formally, the optimal complexity strategy is determined as

$$\mathbf{a}_i^* = \arg\max_{\mathbf{a}_i \in \mathcal{A}_i} Q_i\left(\mathbf{g}(\mathbf{a}_i)\right) : g(a_i) \leq \frac{c_i}{t}, g(a_i) \in \mathbf{g}(\mathbf{a}_i) \quad (6)$$

where $Q_i$ is the $i$th task's quality that we will define in (17) in Section IV-A and $t$ is the duration of time during which task $i$ may consume resources $c_i$.

The strategy spaces defined in (4) and (5) serve only as examples. Another layer partitioning may have more or less layer strategies in $\mathcal{A}_i^{\mathrm{LAYER}}$ which enable finer or coarser complexity scalability. Similarly, more or less rates can be included in $\mathcal{A}_i^{\mathrm{RATE}}$. Importantly, this methodology can be applied to other coders such as H.264/AVC and MPEG-4. Notice that, unlike in a bandwidth-constrained network scenario where $r_i$ is chosen to match the channel bandwidth, we adjust $r_i$ as a dimension of our complexity strategy framework because it influences the resources required for decoding [2]. A joint rate-distortion-complexity framework, however, can be used to adapt to both network bandwidth constraints as well as system resource constraints [7], [9], but this is beyond the scope of this paper.

## III. Fair Allocation in the Resource Domain

### A. *Resource Management Preliminaries*

We consider $M$ real-time video decoding tasks that are to be allocated resources on a single CPU. These tasks are competing for the available CPU resources $\mathcal{R}$, which in our system is the amount of processor cycles that can be allocated to the tasks ($\mathcal{R} \in \mathbb{R}_+$). We assume that a *Resource Manager* (RM) is in place to divide the available resources among the different tasks by using a fairness policy denoted by $\boldsymbol{\mathcal{F}}$. Possible implementations of the RM can be found in [1]. The resources are allocated by the RM for time intervals with a granularity specified by the service interval $t_{\mathrm{SI}}$ and a parameter $w \in \mathbb{Z}_+$, which we describe later. During one service interval, each admitted task is guaranteed CPU time. In general, the service interval $t_{\mathrm{SI}}$ depends on the GOP structure and the delay deadlines of the various tasks. $t_{\mathrm{SI}}$ can be calculated as in [11].

In general, due to overheads, resources cannot be reallocated every $t_{\mathrm{SI}}$ seconds (for the proposed bargaining-based resource allocation system, we describe the resource allocation overheads in Section IV-G). The aforementioned parameter $w$ is introduced to ameliorate this problem by allowing us to decrease the frequency of the resource allocation. Specifically, we define the super service interval $t_{\mathrm{SI}}^w = w \cdot t_{\mathrm{SI}}$ which determines the global resource allocation frequency. Depending on the available resources $\mathcal{R}(t_{\mathrm{SI}}^w)$, the number of active tasks $M$, and the video sequence characteristics $\xi_i$ $(i = 1, \ldots, M)$, a task can deploy a different complexity strategy for every $t_{\mathrm{SI}}^w$.

To enable resource allocation for the upcoming super service interval with duration $t_{\mathrm{SI}}^w$ seconds, the tasks need to provide the RM their external information $\Psi_i$ which depends on the deployed resource allocation scheme and may include information about the task's feasible complexity strategies, scalable CSPEC, decoding deadlines, or quality. Denoting the possible external information as the set $\boldsymbol{\Psi}$, the RM will decide in real-time the nonnegative allocation $\mathbf{X} = (X_1, \ldots, X_M)$ using fairness policy $\boldsymbol{\mathcal{F}}$ as

$$\boldsymbol{\mathcal{F}} : \boldsymbol{\Psi} \rightarrow \mathbb{R}_+^M, \boldsymbol{\mathcal{F}}(\Psi_1, \ldots, \Psi_M) = \mathbf{X} = (X_1, \ldots, X_M). \quad (7)$$

Depending on whether the fairness policy is implemented in the resource domain or the quality domain, the allocation point $\mathbf{X} \in \mathbb{R}_+^M$ will take on a different meaning, specifically

$$\mathbf{X} = (X_1, \ldots, X_M) = \begin{cases} (c_1, \ldots, c_M), & \text{if the policy is implemented in the resource domain} \\ (Q_1, \ldots, Q_M), & \text{if the policy is implemented in the utility domain.} \end{cases} \quad (8)$$

$\mathbf{X} = (c_1, \ldots, c_M)$ $(0 \leq c_i \leq \mathcal{R}(t_{\mathrm{SI}}^w)$ and $\sum_{i=1}^M c_i \leq \mathcal{R}(t_{\mathrm{SI}}^w))$ is the resource-domain allocation where $c_i$ is the number of processor cycles allocated to task $i$ for the super service interval $t_{\mathrm{SI}}^w$. $\mathbf{X} = (Q_1, \ldots, Q_M)$ $(Q_i \geq 0,\ i = 1, \ldots, M)$ is the quality-domain solution where $Q_i$ is the $i$th task's quality (see Section IV-A for the definition of quality and Section IV-D for how the quality-domain solution is converted to the resource allocation). Given the resource allocations $(c_1, \ldots, c_M)$ for the super service interval, each task selects the optimal complexity strategy $\mathbf{a}_i^*$ defined in (6). Finally, the super service interval $t_{\mathrm{SI}}^w$

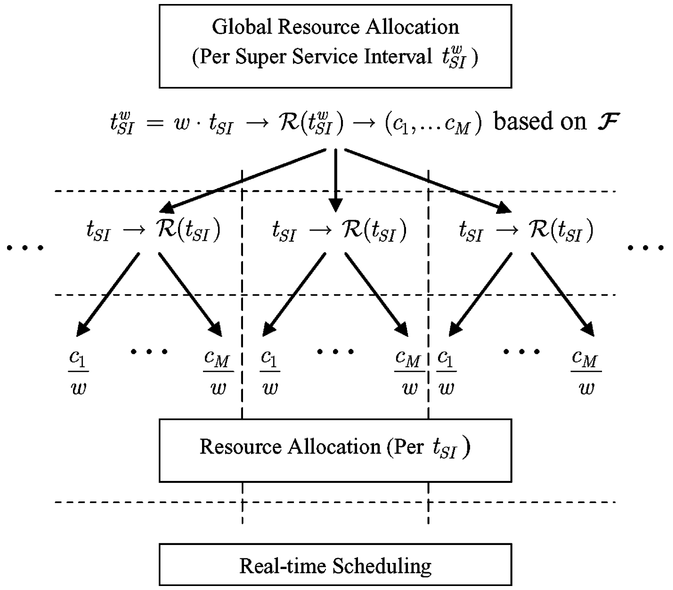

Global Resource Allocation
(Per Super Service Interval $t^w_{SI}$)
$t^w_{SI} = w \cdot t_{SI} \rightarrow \mathcal{R}(t^w_{SI}) \rightarrow (c_1, \ldots c_M)$ based on $\mathcal{F}$
$t_{SI} \rightarrow \mathcal{R}(t_{SI})$
$t_{SI} \rightarrow \mathcal{R}(t_{SI})$
$t_{SI} \rightarrow \mathcal{R}(t_{SI})$
$\frac{c_1}{w}$ … $\frac{c_M}{w}$
$\frac{c_1}{w}$ … $\frac{c_M}{w}$
$\frac{c_1}{w}$ … $\frac{c_M}{w}$
Resource Allocation (Per $t_{SI}$)
Real-time Scheduling

Fig. 5. Relation between global resource allocation per super service interval $t^w_{\mathrm{SI}}$, resource allocation per service interval $t_{\mathrm{SI}}$ and real-time scheduling.

is partitioned into $w$ service intervals with duration $t_{\mathrm{SI}}$ during which each task is allocated resources $c_i/w$.

Fig. 5 illustrates the relationship between the global resource allocation $(c_1, \ldots, c_M)$ for the super service interval $t^w_{\mathrm{SI}}$, the available resources $\mathcal{R}(t_{\mathrm{SI}})$ during each service interval $t_{\mathrm{SI}}$, and real-time system scheduling for the $M$ video tasks. System scheduling is not the focus of the paper, however, any existing real-time scheduling disciplines such as rate-monotonic, deadline-monotonic, earliest deadline first, and least slack [4], [5], [12]–[14] can be applied within each $t_{\mathrm{SI}}$.

### B. Existing Fairness Policies

Here, using the framework introduced in Section III-A, we describe a few existing fairness policies for system resource management as benchmarks against which the quality-domain system resource management policy proposed in Section IV can be compared. In general, the resource-domain fairness policies attempt to allocate resources equitably to each task.

*1) Generalized Processor Sharing (GPS):* The conventional GPS fairness strategy allocates resources proportionally to a task's required resources [15]. In our resource allocation scenario, processor resources can be allocated proportionally to each task's maximum CPU Bandwidth Demand, $g_{i,\max}$. The resource allocations $c_i$ must meet the following condition:

$$\frac{c_1}{g_{1,\max}} = \ldots = \frac{c_M}{g_{M,\max}}, \ \sum_{i=1}^{M} c_i = \mathcal{R}\left(t^w_{\mathrm{SI}}\right) \tag{9}$$

with $g_{i,\max} = \max\{g_i(\mathbf{a}_i) | g_i(\mathbf{a}_i) \in \mathbf{g}_i(\mathbf{a}_i)$ and $\mathbf{g}_i(\mathbf{a}_i) \in \mathcal{G}_i(\mathcal{A}_i)\}$.

In order for the RM to apply this resource management scheme, it must gather from each task $i$ its external information $\Psi_i = (g_{i,\max})$. Hence, the resource allocation by the GPS policy is expressed as

$$\boldsymbol{\mathcal{F}}_{\mathrm{GPS}}(\Psi_1, \ldots, \Psi_M) = (c_1^*, \ldots, c_M^*) \tag{10}$$

where the resource allocation $(c_1^*, \ldots, c_M^*)$ must satisfy (9).

*2) Equal Resource Allocation (ERA):* Using the ERA fairness policy, processor resources are allocated equally to each task. Hence, the ERA policy can be written as

$$\begin{aligned}\boldsymbol{\mathcal{F}}_{\mathrm{ERA}}(\Psi_1, \ldots, \Psi_M) &= (c_1^*, \ldots, c_M^*) \\ &= \left(\frac{\mathcal{R}\left(t^w_{\mathrm{SI}}\right)}{M}, \ldots, \frac{\mathcal{R}\left(t^w_{\mathrm{SI}}\right)}{M}\right)\end{aligned} \tag{11}$$

where $\Psi_i = \mathbf{0}$ for all $i$ because the RM requires no information from the tasks to determine the ERA.

*3) Weighted Max-Min (WMM):* In [1], a WMM fairness policy is used to fairly allocate resources to multiple video decoding tasks. The $i$th task's CPU utilization in terms of the resource allocation $c_i$ is defined as [1]

$$J_i = \frac{c_i}{\mathcal{R}\left(t^w_{\mathrm{SI}}\right)}, \text{ subject to } \sum_{i=1}^{M} J_i \leq 1. \tag{12}$$

The minimum CPU utilization is defined as

$$\begin{aligned}J_i^{\min} = \min \Bigg\{ &\frac{g_i(\mathbf{a}_i) \cdot t^w_{\mathrm{SI}}}{\mathcal{R}\left(t^w_{\mathrm{SI}}\right)} | g_i(\mathbf{a}_i) \in \mathbf{g}_i(\mathbf{a}_i) \\ &\text{and } \mathbf{g}_i(\mathbf{a}_i) \in \mathcal{G}_i(\mathcal{A}_i) \Bigg\}\end{aligned} \tag{13}$$

and the maximum CPU utilization as

$$\begin{aligned}J_i^{\max} = \max \Bigg\{ &\frac{g_i(\mathbf{a}_i) \cdot t^w_{\mathrm{SI}}}{\mathcal{R}\left(t^w_{\mathrm{SI}}\right)} | g_i(\mathbf{a}_i) \in \mathbf{g}_i(\mathbf{a}_i) \\ &\text{and } \mathbf{g}_i(\mathbf{a}_i) \in \mathcal{G}_i(\mathcal{A}_i) \Bigg\}.\end{aligned} \tag{14}$$

Given the user-defined weights $\phi_i$, the WMM resource allocation algorithm is as follows [1].

1) The resource manager allocates to each task $i$ ($1 \leq i \leq M$) its minimum CPU utilization $J_i^{\min}$, i.e., $J_i = J_i^{\min}$. If $J_i < J_i^{\max}$, then task $i$ is made an element of the unsatisfied task set $\mathbf{I} = \{i | 1 \leq i \leq M$ and $J_i < J_i^{\max}\}$.
2) For each task $i \in \mathbf{I}$, increment the CPU utilization by $(\phi_i / \sum_{i \in \mathbf{I}} \phi_i)(1 - \sum_{i=1}^{M} J_i)$. If $J_i > J_i^{\max}$, set $J_i = J_i^{\max}$ and remove task $i$ from $\mathbf{I}$.
3) Repeat step 2) if $\sum_{i=1}^{M} J_i \neq 1$ and $\mathbf{I} \neq \emptyset$. Otherwise, allocate resources $c_i^* = J_i \cdot \mathcal{R}(t^w_{\mathrm{SI}})$ to each task $i$.

Hence, the WMM policy can be written as

$$\begin{aligned}\boldsymbol{\mathcal{F}}_{\mathrm{WMM}}(\Psi_1, \ldots, \Psi_M) &= (c_1^*, \ldots, c_M^*) \\ &= \left(J_1 \cdot \mathcal{R}\left(t^w_{\mathrm{SI}}\right), \ldots, J_M \cdot \mathcal{R}\left(t^w_{\mathrm{SI}}\right)\right)\end{aligned} \tag{15}$$

where $J_i$ is the $i$th task's CPU utilization determined by the WMM resource allocation algorithm and $\Psi_i = (J_i^{\min}, J_i^{\max}, \phi_i)$.

## IV. Fair Resource Allocation Solution in the Quality Domain

The fairness policies described in Section III-B allocate CPU resources without considering the quality impact. For multimedia applications, however, explicitly considering the video quality is essential because when a solution is fair in the resource domain it does not guarantee a quality-fair solution. To overcome the shortcomings of the resource-domain resource allocation policies for multimedia applications, we propose using a bargaining-theoretic approach based on the KSBS for fair and optimal (in a Pareto sense) resource allocation in the quality domain. To determine the KSBS, we must first define a quality-complexity model to relate a task's resource consumption to its achieved video quality. We propose such a model in the following subsection.

### A. *Quality Definition and Model*

In previous sections, we have described that a quality–complexity tradeoff is made when choosing one complexity strategy instead of another. In order to quantify this tradeoff, we introduce our quality-complexity model in this section. In multimedia applications, the video quality depends on the video sequence characteristics, encoding parameters, and deployed multimedia algorithms (e.g., H.264/AVC or MPEG-2) and can be expressed as a function of complexity. The quality-complexity (QC) model introduced in this paper is denoted as $Q(\mathbf{g})$, where $\mathbf{g}$ is the CSPEC and $Q(\cdot)$ is the PSNR which is a frequently used measure of video quality. Given a set of complexity strategies $\mathcal{A}_i$ and the corresponding scalable CSPEC $\mathcal{G}_i(\mathcal{A}_i)$, we define the $i$th task's operational quality set as

$$\boldsymbol{\mathcal{U}}_i\left(\mathcal{G}_i(\mathcal{A}_i)\right) = \left\{u_i\left(\mathbf{g}_i(\mathbf{a}_i)\right) | \mathbf{g}_i(\mathbf{a}_i) \in \mathcal{G}_i(\mathcal{A}_i)\right\} \tag{16}$$

where $u_i(\mathbf{g}_i(\mathbf{a}_i))$ is the measured or estimated average PSNR (i.e., quality) that user $i$ achieves when deploying complexity strategy $\mathbf{a}_i$ with the corresponding CSPEC $\mathbf{g}_i(\mathbf{a}_i)$. In other words, $\mathcal{U}_i(\mathcal{G}_i(\mathcal{A}_i))$ contains all of the quality points achievable by task $i$ given its set of complexity strategies $\mathcal{A}_i$. Similar to conventional operational rate-distortion models, we fit the QC model to the convex hull of the operational quality set because we assume that $Q_i(\mathbf{g}_i)$ is monotonically increasing with increased resource consumption (i.e., for a fixed delay $d_i \in \mathbf{g}_i$, if $g_i \in \mathbf{g}_i$ increases, then the quality also increases). This assumption is reasonable because we would not waste resources by allocating them to a task that will gain nothing from them in terms of quality. $Q_i(\mathbf{g}_i)$ may also be interpreted as the quality corresponding to the achievable minimum distortion defined in [19].

Using an empirical curve fitting approach, similar to successful approaches used in rate-distortion theory for modeling the average distortion-rate performance of video coders, we found that a good QC model for the $t+2D$ version of the MCTF video coder in [3] is

$$Q(\mathbf{g}) = m_1 \exp(m_2 \cdot g) + m_3, g \in \mathbf{g}, g_{\max} \geq g \geq g_{\min}, g_{\min} > 0 \tag{17}$$

where $m_1, m_2 \in \mathbb{R}_+$ and $m_3 \in \mathbb{R}$ are the QC model parameters that depend on the sequence characteristics $\xi$, the Scalable CSPEC $\mathcal{G}_i(\mathcal{A}_i)$, and the operational quality set $\mathcal{U}_i(\mathcal{G}_i(\mathcal{A}_i))$. Additionally, $g_{\min}$ and $g_{\max}$ are the minimum and maximum required resources to decode a scalable bit-stream, respectively, and $Q(\cdot)$ is the PSNR of the luminance channel. Example QC model parameters for the *Stefan*, *Silent*, *Foreman*, and *Coastguard* sequences at CIF resolution, determined using least-mean square fitting techniques and the measured PSNR of the sequence's luminance components, are shown in Table II along with the corresponding mean square error (MSE) of the model fit.

TABLE II
Quality-Complexity Model Parameters [See (17)] and MSE of the Model Fit for the *Silent*, *Stefan*, *Coastguard*, and *Foreman* Sequences at CIF Resolution and With a 30-Hz Encoded Frame-Rate

| Model Parameter | Video Sequence | | | |
|---|---|---|---|---|
| | Silent | Stefan | Coastguard | Foreman |
| $m_1$ | 43.98 | 0.7022 | 4.63 | 8.72 |
| $m_2$ | 5.38e-6 | 2.63e-5 | 1.17e-5 | 1.24e-5 |
| $m_3$ | -20.60 | 17.14 | 16.01 | 9.24 |
| MSE | 0.02 | 0.09 | 0.03 | 0.33 |

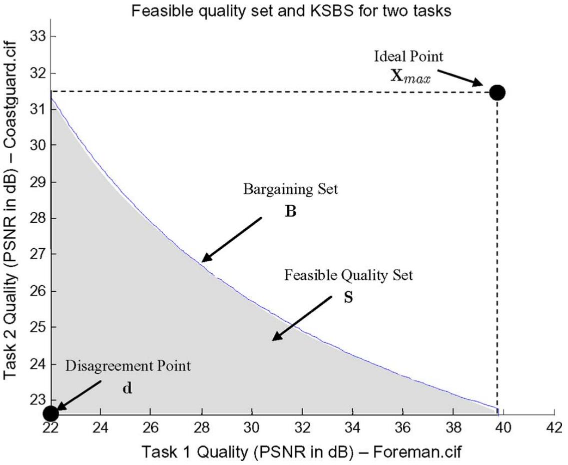


Fig. 6. Nonconvex feasible quality set for two decoding tasks: *Foreman* (task 1) and *Coastguard* (task 2).

Note that (17) is just one example of a QC model. Other coders may use different QC models. Additionally, the QC model could be based on the Mean CPU Bandwidth requirement $\rho$ or the Peak CPU Bandwidth requirement $P$ or a statistical model [1], [6] depending on the application requirements. Importantly, if a complexity-scalable coder is used, we can construct a set of feasible complexity strategies $\mathcal{A}_i$ such that any point on $Q(\mathbf{g})$ is achievable.

### B. *Feasible Quality Set*

The feasible quality set $\mathbf{S}$ for $M$ tasks contains all of the achievable quality points in $\mathbb{R}_+^M$ given the total system resource constraint $\mathcal{R}(t_{\mathrm{SI}}^w)$. Formally, the feasible quality set is defined as follows.

*Definition 1: Feasible Quality Set:*

$$\mathbf{S} = \left\{ (Q_1(\mathbf{g}_1), \ldots, Q_M(\mathbf{g}_M)) \,| \right. \\ \left. \sum_{i=1}^{M} g_i \cdot t_{\mathrm{SI}}^w \leq \mathcal{R}\left(t_{\mathrm{SI}}^w\right) \text{ and } g_i \in \mathbf{g}_i \right\}. \tag{18}$$

The feasible quality set's properties constrain the set of solutions that we may apply to determine a quality-fair resource allocation. Observing Fig. 6, it is clear that the feasible quality set $\mathbf{S}$ (the shaded region in the figure) is nonconvex. The nonconvexity of the feasible quality set is a result of the exponential form of each task's QC function [see (17)]. Due to the nonconvexity of $\mathbf{S}$, conventional Lagrangian optimization techniques which require convexity are not applicable because using convex relaxation of the original feasible quality set may result in video quality degradation, especially in cases where operational QC points are sparse. Moreover, the solutions are often very complex [1].

The feasible quality set $\mathbf{S}$, however, is $\mathbf{d}$-comprehensive, which is a requirement of the KSBS [18]. In the following definition, we let $\mathbf{x} \leq \mathbf{y}$ denote component-wise inequality for vector comparison.

*Definition 2: $\mathbf{d}$-Comprehensive:* A $\mathbf{d}$-comprehensive set $\mathbf{S} \in \mathbb{R}^M$ is one for which given a point $\mathbf{d} \in \mathbb{R}^M$, if $\mathbf{d} \leq \mathbf{x} \leq \mathbf{y}$ and $\mathbf{y} \in \mathbf{S}$, then $\mathbf{x} \in \mathbf{S}$.

Clearly, $\mathbf{S}$ is $\mathbf{d}$-comprehensive because (17) is a monotonically increasing function of the resource allocation.

### C. Problem Formulation

Here, we describe the objective and constraints of the quality-fair resource allocation.

*System Resource Constraint:* The CPU has a total resource constraint $\mathcal{R}$. The resource considered is in units of processor cycles. Each application $i$ $(i = 1, \ldots, M)$ is allocated resources $c_i$ such that

$$\sum_{i=1}^{M} c_i \leq \mathcal{R}. \tag{19}$$

*Application Quality Constraint:* Application $i$ $(i = 1, \ldots, M)$ requires at least a minimum quality denoted by $X_i^{\min}$. Additionally, application $i$ has a maximum (best) achievable quality $X_i^{\max}$ that corresponds to deploying the complexity strategy that requires the most resources. In this paper, quality is defined as the average decoded PSNR. Note that if the application quality constraint cannot be met for a particular application (i.e., the achievable quality is not higher than its minimum required quality), then that application will not be included in the resource allocation and the task will not be performed. In summary, letting $Q_i(c_i)$ denote the quality achieved by the $i$th application when it is allocated resources $c_i$, the application quality constraint can be written as

$$X_i^{\min} \leq Q_i(c_i) \leq X_i^{\max},\ \forall i \in \{1, \ldots, M\}. \tag{20}$$

*Fairness Criterion:* We define fairness for autonomous applications sharing the same CPU resources to be a proportional drop in quality for all applications. Formally, such a fair allocation satisfies the following fairness criterion:

$$\frac{Q_i(c_i) - X_i^{\min}}{X_i^{\max} - X_i^{\min}} = \frac{Q_j(c_j) - X_j^{\min}}{X_j^{\max} - X_j^{\min}}, \forall i, j \in \{1, \ldots, M\}. \tag{21}$$

*Problem Formulation:* We desire to divide the CPU resources $\mathcal{R}$ among the $M$ applications such that the fairness criterion is satisfied, the application quality constraint is met, and the system resource constraint is met with equality such that no system resources are left unused. The quality-fair resource allocation problem is formulated as follows:

$$\begin{gathered}\text{find } \mathbf{c} \in \partial\mathbf{C} \\ s.t.\ \partial\mathbf{C} = \left\{(c_1, \ldots, c_M) | \sum_{i=1}^{M} c_i = \mathcal{R}\right\} \\ X_i^{\min} \leq Q_i(c_i) \leq X_i^{\max},\ \forall i \in \{1, \ldots, M\} \\ \frac{Q_i(c_i) - X_i^{\min}}{X_i^{\max} - X_i^{\min}} = \frac{Q_j(c_j) - X_j^{\min}}{X_j^{\max} - X_j^{\min}},\ \forall i, j \in \{1, \ldots, M\}.\end{gathered} \tag{22}$$

We note that the solution to the above problem is unique because the feasible quality set is $\mathbf{d}$-comprehensive. In this paper, in order to solve (22), we deploy the KSBS from axiomatic bargaining theory. The KSBS formulates and solves the resource allocation problem in (22) in the quality domain by explicitly considering the quality impact on the various applications sharing the same system resources.

### D. KSBS: A Fair Scheduling Solution in the Quality Domain

Using the KSBS, the resource manager tries to determine the resource allocation by first selecting a fair and optimal quality-domain solution and then mapping this solution into the resource domain solution described in (22). Here, we describe the concept of a bargaining problem and then present the six axioms that comprise the KSBS. Subsequently, in the following subsection, we describe how the quality-domain bargaining solution is mapped back into the resource domain.

The bargaining problem is expressed as a pair $(\mathbf{S}, \mathbf{d})$. $\mathbf{S}$ represents the feasible quality set defined in (18) and $\mathbf{d}$ is a disagreement point corresponding to the minimum acceptable qualities of all the tasks. $\mathbf{d}$ is formally defined below in definition 3. Since the resource manager tries to allocate the resources in an optimal manner, the selected quality point needs to be in the *Pareto Optimal* quality set (Pareto optimality is defined below in Definition 5). Note that, in this section, the notation $X_i$ represents the $i$th task's quality and is used interchangeably with $Q_i(\cdot)$.

*Definition 3: Disagreement Point:* The point $\mathbf{d} = (X_1^{\min}, \ldots, X_M^{\min}) \in \mathbf{S}$ is the disagreement point if $\mathbf{d} = (X_1^{\min}, \ldots, X_M^{\min}) = (\min_{\mathbf{X} \in \mathbf{S}} X_1, \ldots, \min_{\mathbf{X} \in \mathbf{S}} X_M)$ [18].

The coordinates of the disagreement point, $X_i^{\min}$ $(i = 1, \ldots, M)$, are the qualities obtained when the resource manager assigns task $i$ resources $c_i = g_i^{\min}(\mathbf{a}_i) = \min\{g_i(\mathbf{a}_i) \in \mathbf{g}_i(\mathbf{a}_i) | \mathbf{g}_i(\mathbf{a}_i) \in \mathcal{G}_i(\mathcal{A}_i)\}$, i.e., $X_i^{\min} = Q_i(\mathbf{g}_i^{\min}(\mathbf{a}_i))$. $X_i^{\min}$ can be defined by the task prior to revealing its external information $\Psi_i$ to the RM. $X_i^{\min}$ may also be interpreted as the minimum quality expected by task $i$ for collaborating in the resource allocation or, as the quality the task expects to gain if it does not collaborate at all (i.e., does not want to participate in the resource allocation and wants only to receive best-effort service).

*Definition 4: Ideal Point:* The point $\mathbf{X}_{\max} = (X_1^{\max}, \ldots, X_M^{\max}) \in \mathbf{S}$ is the ideal point if $\mathbf{X}_{\max} = (X_1^{\max}, \ldots, X_M^{\max}) =(\max_{\mathbf{X} \in \mathbf{S}, \mathbf{X} \geq \mathbf{d}} X_1, \ldots, \max_{\mathbf{X} \in \mathbf{S}, \mathbf{X} \geq \mathbf{d}} X_M)$ [18], that is, the ideal point is the point where every task achieves its maximum quality. In general, the ideal point cannot be achieved because there are not enough resources to allocate to each task in order for them to all obtain their maximum quality.

*Definition 5: Pareto Optimality:* The quality point $(X_1, \ldots, X_M) \in \mathbf{S}$ is *Pareto optimal* if for each $(X_1', \ldots, X_M') \in \mathbf{S}$ and $(X_1', \ldots, X_M') \geq (X_1, \ldots, X_M)$, then $(X_1', \ldots, X_M') = (X_1, \ldots, X_M)$ [18].

Pareto optimality in the quality domain implies that the resource-domain allocation satisfies the condition $\sum_{i=1}^{M} c_i = \mathcal{R}(t_{\mathrm{SI}}^{w})$. This is highly desirable because no resources are wasted and no task can increase its quality without decreasing the quality of another task. If some resources are left unused, however, then one or more tasks can increase their resource allocation (and quality) without decreasing any other task's resource allocation (and quality).

Generally, a bargaining solution is a function $F : (\mathbf{S}, \mathbf{d}) \rightarrow \mathbb{R}^M$ with the property that $F(\mathbf{S}, \mathbf{d}) \in \mathbf{S}$. The KSBS gives a unique and fair Pareto optimal solution that fulfills the following axioms. In the following, we let $\mathbf{x} \leq \mathbf{y}$ denote component-wise inequality for vector comparison.

*Definition 6: KSBS:* $\mathbf{X}^* = F(\mathbf{S}, \mathbf{d})$ is said to be a KSBS in $\mathbf{S}$ for the disagreement point $\mathbf{d}$, if the following six axioms are satisfied [18]:

1) *Individual Rationality*: $\mathbf{X}^* \geq \mathbf{d}$.

2) *Feasibility*: $\mathbf{X}^* \in \mathbf{S}$.

3) *Pareto Optimality*: $\mathbf{X}^*$ is Pareto optimal.

Axioms 1)–3) define the *bargaining set* $\mathbf{B} = \{\mathbf{X} \in \mathbf{S} | \mathbf{Y} > \mathbf{X} \Rightarrow \mathbf{Y} \notin \mathbf{S}\}$, which is the set of all individually rational and Pareto-optimal quality points. Thus, the KSBS is located in the bargaining set. Note that the bargaining set $\mathbf{B}$ contains all of the quality points that correspond to the set of resource allocations $\partial\mathbf{C}$ defined in (22).

4) *Individual Monotonicity*: Given another feasible quality set $\mathbf{S}^*$, if $\mathbf{S}' \supset \mathbf{S}$, $\mathbf{d} = \mathbf{d}'$, and $\max_{\mathbf{X} \in \mathbf{S}, \mathbf{X} \geq \mathbf{d}} X_k = \max_{\mathbf{X}' \in \mathbf{S}', \mathbf{X}' \geq \mathbf{d}'} X'_k$, $\forall k \in \{1, \ldots, M\} \setminus \{i\}$, then $[F(\mathbf{S}', \mathbf{d}')]_i \geq [F(\mathbf{S}, \mathbf{d})]_i$.

Axiom 4) states that increasing the bargaining set size in a direction favorable to task $i$ always benefits task $i$. In other words, Axiom 4) provides incentive for a task to deploy the optimal complexity strategy in (6) so that the resources allocated to the task by the RM are optimally used to maximize the tasks quality.

To illustrate how the feasible quality set changes when one task deploys a subset of its available complexity strategies, consider the two user example in Fig. 7(a) and (b) where task 1 consists of decoding *Silent* and task 2 consists of decoding *Stefan*. Fig. 7(a) shows the feasible quality set in a scenario where both tasks deploy the complexity strategy set corresponding to the cross-product of (4) and (5). Fig. 7(b) shows the feasible quality set in a scenario where task 1 deploys at random only one quarter of its complexity strategy set while task 2 deploys the same strategy set as in the former scenario. In both cases, task 2 has a maximum achievable quality $X_2^{\max} = 32.26$ dB; hence, its maximum achievable quality is not penalized or rewarded by the complexity strategies that task 1 chooses to reveal to the RM. Task 1, however, loses ~4 dB in maximum achievable quality in the second scenario compared to the first because it does not deploy the optimal complexity strategy given the available resources [see (b)]. Moreover, the KSBS results in task 1 achieving ~2 dB less in the second scenario (i.e., $X_1^* = 34.13$ dB) compared with the first (i.e., $X_1^* = 35.90$ dB). Note that the low quality (<30 dB) achieved by task 2 in both scenarios may be caused by decoding at a low temporal resolution and not necessarily by poor image quality. Using (23) (defined below in Section IV-E) to determine the resource allocation corresponding to the quality point $\mathbf{X}^*$, we determine that both scenarios yield nearly the same resource allocation despite the significantly different qualities. Specifically, in scenario 1, 31.38% and 68.62% of the available resources are allocated to task 1 and task 2, respectively; in scenario 2, 30.81% and 69.19% of the available resources are allocated to task 1 and 2, respectively. Hence, because the task's complexity strategies directly impact how much quality the task can derive from the available resources, each task has incentive to deploy its optimal complexity strategy[(see (6)] to maximize its quality.

5) *Independence of Linear Transformations*: For any linear scale transformation $\varphi$, $\varphi(F(\mathbf{S}, \mathbf{d})) = F(\varphi(\mathbf{S}), \varphi(\mathbf{d}))$.

Axiom 5) states that the bargaining solution does not change if the quality function and disagreement point are scaled by a linear transformation.

6) *Symmetry*: If $\mathbf{S}$ is invariant under all exchanges of tasks, $F_i(\mathbf{S}, \mathbf{d}) = F_j(\mathbf{S}, \mathbf{d})$ for all possible tasks $i, j$.

Finally, Axiom 6) is a special case of the equal penalty drop fairness criteria imposed by the KSBS. It implies that if tasks have the same disagreement points and the same maximum achievable quality, then they will have the same quality allocation and therefore incur equal drops in quality. Together, axioms 4, 5, and 6 are called "axioms of fairness."

### E. Resource Allocation Using the KSBS

Let $F$ be the KSBS defined by the six axioms described in Section IV-D and let $\mathbf{Q} = (Q_1(\cdot), \ldots, Q_M(\cdot))$ be a set of quality functions. We can express the system resource allocation as

$$\begin{aligned}\boldsymbol{\mathcal{F}}_{\mathrm{KSBS}}(\Psi_1, \ldots, \Psi_M, \mathbf{C}) &= (\mathbf{Q}^{-1} \circ F \circ \mathbf{Q})(\Psi_1, \ldots, \Psi_M, \mathbf{C}) \\ &= (c_1^*, \ldots, c_M^*) \end{aligned} \tag{23}$$

where a composite function of $f$ and $h$ is denoted as $(f \circ h)(x) = f(h(x))$ and $\mathbf{C} = \{(c_1, \ldots, c_M) | \sum_{i=1}^{M} c_i \leq \boldsymbol{\mathcal{R}}\}$. Note that $\partial\mathbf{C}$, defined in (22), is the subset of $\mathbf{C}$ for which there is equality in the system resource constraint [see (19)]. The KSBS requires that the resource manager gathers the external information $\Psi_i = (Q_i(\cdot), X_i^{\min})$ from each task $i = 1, \ldots, M$ (i.e., each task's QC model and minimum required quality).

Given a set of resource allocations $\mathbf{C}$ and the received external information defining the QC models, (23) first forms the bargaining problem pair $(\mathbf{S}, \mathbf{d})$ and then finds the KSBS $\mathbf{X}^* = F(\mathbf{S}, \mathbf{d})$. Finally, (23) determines the fair and Pareto-optimal resource allocation $(c_1^*, \ldots, c_M^*)$ using the inverse of the quality function (i.e., the inverse of (17)). In the next subsection, we describe how to determine the KSBS $\mathbf{X}^* = F(\mathbf{S}, \mathbf{d})$.

### F. Generalized KSBS

The conventional KSBS for $M$ tasks satisfies

$$\mathbf{X}^* = F(\mathbf{S}, \mathbf{d}) = \mathbf{d} + \lambda_{\max}(\mathbf{X}_{\max} - \mathbf{d}) \tag{24}$$

where $\lambda_{\max} = \max_{\lambda}\{\lambda | \mathbf{d} + \lambda(\mathbf{X}_{\max} - \mathbf{d}) \in \mathbf{S}\}$. Hence, the conventional KSBS is the intersection between the bargaining set $\mathbf{B} \subseteq \mathbf{S}$ and the line $\mathbf{L}$ joining the disagreement point and the ideal point defined by

$$\mathbf{L} = \left\{ \mathbf{X} | \frac{X_1}{X_1^{\max}} = \cdots = \frac{X_M}{X_M^{\max}}, X_i > 0, \forall i \right\}. \tag{25}$$

Note that the constraint that $X_i > 0$ is a consequence of the translated disagreement point being the origin of the quality space (i.e., $\mathbf{d} = \mathbf{0}$). A generalization of the KSBS can be en-

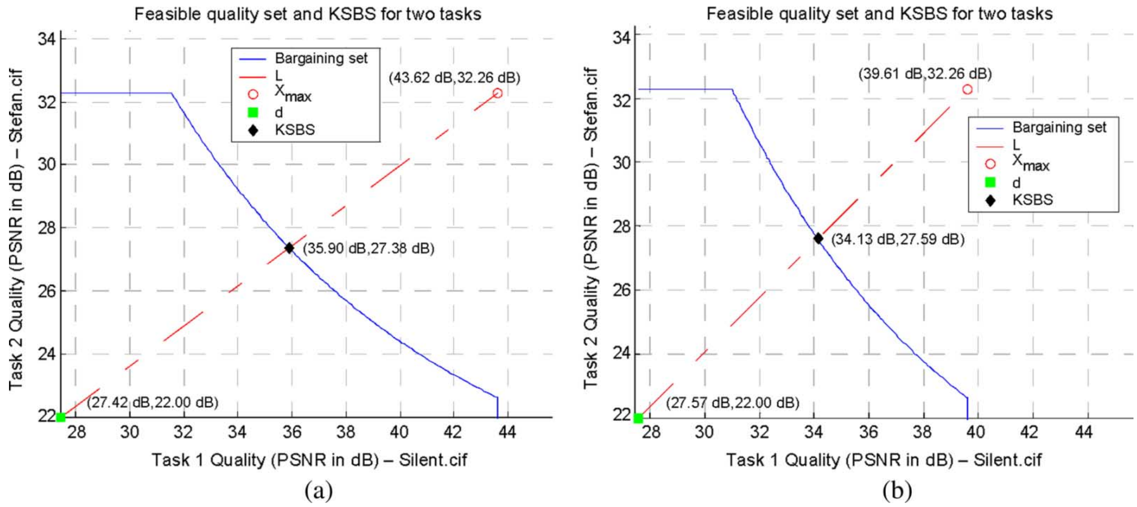


Fig. 7. Examples of achievable quality sets and the associated KSBS for two decoding tasks. Task 1 and task 2 decode CIF video sequences *Silent* and *Stefan*, respectively, under a resource constraint. The $x$-axis of both plots is the quality (PSNR in dB) of task 1, and the $y$-axis of both plots is the quality of task 2. (a) Scenario where both tasks deploy all of their complexity strategies. (b) Scenario where task 1 deploys at random one quarter of its complexity strategies and task 2, again, deploys all of its complexity strategies.

visaged by assigning each task $i$ different bargaining powers, $\alpha_i$, which scale the ideal point and lead to unique weighted bargaining solutions in the Pareto optimal bargaining set $\mathbf{B}$. The generalized KSBS, however, requires that the disagreement point coincides with the origin of the quality space (i.e., $\mathbf{d} = \mathbf{0}$). Therefore, to determine the generalized KSBS, we have to first translate the feasible quality set $\mathbf{S}$ to the set $\tilde{\mathbf{S}} = \mathbf{S} - \mathbf{d}$ such that the translated disagreement point $\tilde{\mathbf{d}} = \mathbf{d} - \mathbf{d} = \mathbf{0}$ coincides with the origin of the quality space. Similarly, the ideal point is translated to $\tilde{\mathbf{X}}_{\max} = \mathbf{X}_{\max} - \mathbf{d}$. We then form the line $\tilde{\mathbf{L}}'$ joining the translated disagreement point and the translated and scaled ideal point defined by

$$\tilde{\mathbf{L}}' = \left\{ \tilde{\mathbf{X}} | \frac{\tilde{X}_1}{\alpha_1 \tilde{X}_1^{\max}} = \cdots = \frac{\tilde{X}_M}{\alpha_M \tilde{X}_M^{\max}}, \tilde{X}_i > 0, \sum_{i=1}^{M} \alpha_i = 1, \alpha_i \geq 0, \forall i \right\} \quad (26)$$

where $\alpha_i$ denotes the $i$th task's bargaining power. When $\alpha_1 = \ldots = \alpha_M$, the generalized KSBS becomes the conventional KSBS. As in (25), the constraint that $\tilde{X}_i > 0$ is due to the fact that $\tilde{\mathbf{d}} = \mathbf{0}$. The translated generalized KSBS $\tilde{\mathbf{X}}^* = F(\tilde{\mathbf{S}}, \tilde{\mathbf{d}})$ is the intersection of $\tilde{\mathbf{L}}'$ with the translated Pareto-optimal bargaining set $\tilde{\mathbf{B}} = \mathbf{B} - \mathbf{d}$.

We solve for $\tilde{\mathbf{X}}^*$ using a low-complexity numerical method based on a bisection search with a predetermined tolerance $\varepsilon$, which can be adjusted to limit the overall complexity of the resource allocation. The complexity of this search is on the order of $\log_2(\varepsilon) \cdot M$. Finally, the generalized KSBS is determined as $\mathbf{X}^* = \tilde{\mathbf{X}}^* + \mathbf{d}$. Fig. 7(a) and (b) illustrates how the line $\mathbf{L}' = \tilde{\mathbf{L}}' + \mathbf{d}$, determined by the disagreement point $\mathbf{d}$ and the ideal point $\mathbf{X}_{\max}$, intersects the bargaining set $\mathbf{B} = \tilde{\mathbf{B}} + \mathbf{d}$ to obtain the KSBS for two different two task scenarios.

We note that the bargaining powers provide a flexible resource management tool for making quality tradeoffs between different tasks. This is particularly important for multimedia applications because the priorities of different tasks may change over time depending on the sequence's content characteristics [16]. Examples of the affect of bargaining powers are given in Section V.

### G. Proposed Bargaining-Based Resource Allocation System

Here, we describe the steps that comprise our bargaining-based resource allocation system and discuss the complexity overheads associated with each step. All of the following steps for determining the KSBS are illustrated in Fig. 8.

Step 1) ***Session initialization stage***: The RM determines the available system resources $\mathcal{R}(t_{\mathrm{SI}}^{w})$ for the current super service interval $t_{\mathrm{SI}}^{w}$ and conveys this information to tasks $i \in \{1, \ldots, M\}$. This step is performed online, however, it incurs negligible complexity overhead.

Step 2) ***Determine the Scalable CSPEC***: Given the sets of complexity strategies $\mathcal{A}_i$, $i \in \{1, \ldots, M\}$, we determine the Scalable CSPEC $\mathcal{G}_i(\mathcal{A}_i)$ defined in (3). The Scalable CSPEC can be determined using offline modeling, training, or profiling, followed by real-time classification [2], [7], [9]. The offline methods incur no overheads during the resource allocation. Only a small overhead is incurred by real-time classification.

Step 3) ***Determine the quality-complexity models/external information***: The QC models $Q_i(\cdot)$ ($i \in \{1, \ldots, M\}$) defined in (17) are determined offline (similar to [7]) as described in Section IV-A. Therefore, no overhead is incurred during the online resource allocation. These QC models comprise each task's external information that must be revealed to the RM.

Step 4) ***Determine the KSBS***: For the set of $M$ tasks, we find the quality-domain allocation $\mathbf{X}^* = (X_1^*, \ldots, X_M^*)$

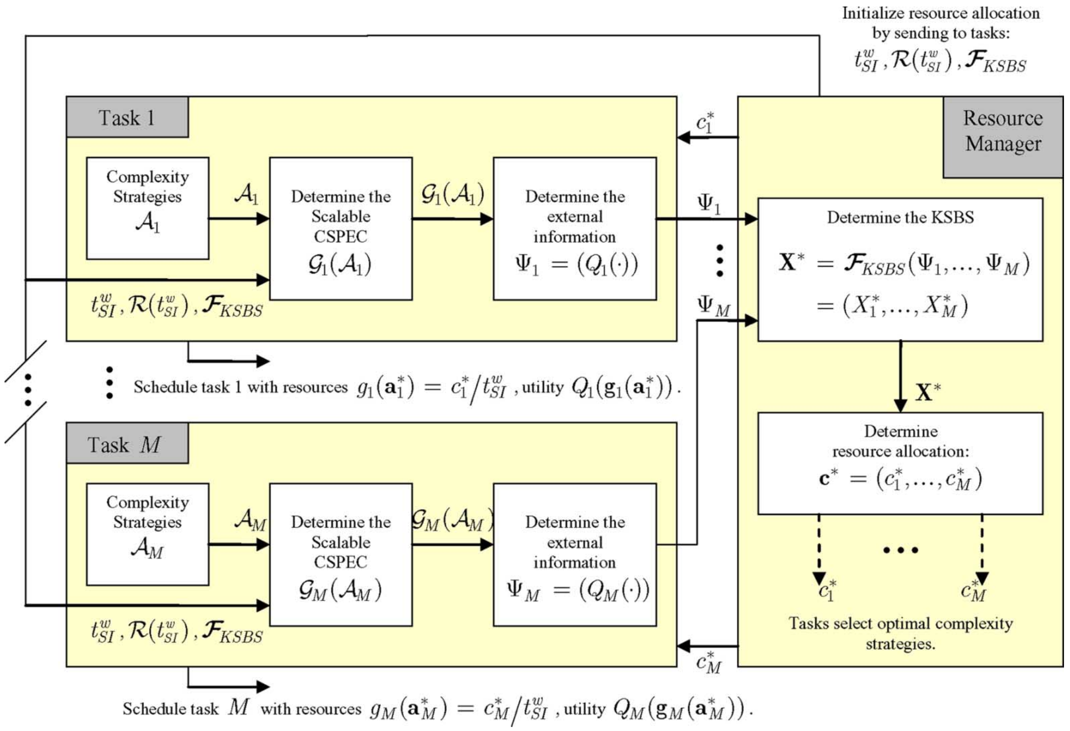

Initialize resource allocation by sending to tasks:
$t_{SI}^{w}, \mathcal{R}(t_{SI}^{w}), \mathcal{F}_{KSBS}$
Task 1
Complexity Strategies $\mathcal{A}_1$
$\mathcal{A}_1$
Determine the Scalable CSPEC $\mathcal{G}_1(\mathcal{A}_1)$
$\mathcal{G}_1(\mathcal{A}_1)$
Determine the external information $\Psi_1 = (Q_1(\cdot))$
$t_{SI}^{w}, \mathcal{R}(t_{SI}^{w}), \mathcal{F}_{KSBS}$
$c_1^*$
$\Psi_1$
$\Psi_M$
Resource Manager
Determine the KSBS
$\mathbf{X}^* = \mathcal{F}_{KSBS}(\Psi_1, \ldots, \Psi_M) = (X_1^*, \ldots, X_M^*)$
$\mathbf{X}^*$
Schedule task 1 with resources $g_1(\mathbf{a}_1^*) = c_1^*/t_{SI}^{w}$, utility $Q_1(\mathbf{g}_1(\mathbf{a}_1^*))$.
Task $M$
Complexity Strategies $\mathcal{A}_M$
$\mathcal{A}_M$
Determine the Scalable CSPEC $\mathcal{G}_M(\mathcal{A}_M)$
$\mathcal{G}_M(\mathcal{A}_M)$
Determine the external information $\Psi_M = (Q_M(\cdot))$
$t_{SI}^{w}, \mathcal{R}(t_{SI}^{w}), \mathcal{F}_{KSBS}$
Determine resource allocation: $\mathbf{c}^* = (c_1^*, \ldots, c_M^*)$
$c_1^*$
$c_M^*$
Tasks select optimal complexity strategies.
$c_M^*$
Schedule task $M$ with resources $g_M(\mathbf{a}_M^*) = c_M^*/t_{SI}^{w}$, utility $Q_M(\mathbf{g}_M(\mathbf{a}_M^*))$.

Fig. 8. Resource allocation system diagram showing the exchange of information between the $M$ video tasks and the RM which determines a fair resource allocation based on the imposed fairness policy.

determined by the KSBS (see (24)). The KSBS is solved online using a low-complexity numerical method based on a bisection search with a predetermined tolerance $\varepsilon$, which can be adjusted to limit the overall complexity of the resource allocation. The complexity of this search is $O(|\log_2(\varepsilon)| \cdot M)$.

Step 5) ***Determine the quality-fair resource allocation***: The quality-domain allocation $\mathbf{X}^* = (X_1^*, \ldots, X_M^*)$ determined by the KSBS is then converted to the quality-fair resource allocation $c^* = (c_1^*, \ldots, c_M^*)$ using (23). The step incurs negligible computational overhead since it only requires calculating the inverse of (17) for each of the $M$ tasks.

Step 6) ***Select the optimal complexity strategy***: Given the quality-fair resource allocations $c_i^*$ $(i \in \{1, \ldots, M\})$ each task selects its optimal complexity strategy $\mathbf{a}_i^*$ defined in (6). This step is performed online, however, if we assume that each task's complexity strategies are sorted (offline) in descending or ascending order of complexity, the optimal complexity strategy can be determined using a low complexity binary search. For a set of $M$ tasks that each have $K$ complexity strategies, the complexity of the binary search is $O(M \cdot \log_2(K))$. This complexity is limited for a relatively small number of tasks and complexity strategies. However, if a system designer wants to limit the complexity of this step, a maximum number of complexity strategies per task can be imposed. The number of these strategies can be the same for all tasks, or can be larger for more important tasks/users or tasks that require a large complexity for their execution.

Step 7) ***Real-time scheduling***: Each task is scheduled using any real-time scheduling policy in the literature [4], [5], [12]–[14], consumes resources $g_i(\mathbf{a}_i^*) = c_i^*/t_{\mathrm{SI}}^{w}$, and achieves quality $Q_i(\mathbf{g}_i(\mathbf{a}_i^*)), g_i(\mathbf{a}_i^*) \in \mathbf{g}_i(\mathbf{a}_i^*)$.

## V. Experiments

Here, we first evaluate the CSPEC as an admission control tool for multimedia systems and, specifically, video decoding. We then compare the performance of the KSBS against other system resource allocation fairness policies. Finally, we give examples of the effect of motion and texture based bargaining powers on the KSBS.

### A. Evaluation of the CSPEC as an Admission Control Tool

In Section II-C, we illustrated how adding the delay parameter $d$ to a task's deadlines significantly reduced the task's CPU bandwidth requirement. Here, we evaluate the use of the CSPEC's CPU bandwidth demand parameter $g$ (defined in Section II-B), used for admission control and complexity prediction, against an existing solution in the literature [1], [6] and also against the mean bandwidth requirement.

Let any frames belonging to the equivalent display deadlines within each GOP belong to a particular deadline/priority class $n$. For example, if $t_\chi$ is the first deadline of the $\chi$-th GOP, then all frames with their deadlines at $t_\chi + 2n\Delta$, for $n = 0, \ldots, 7$ (as in Table I) belong to the $n$th deadline/priority class. Smaller

TABLE III
DEADLINE MISS PERCENTAGE FOR DIFFERENT DEADLINE/PRIORITY CLASSES FOR TWO BANDWIDTH ALLOCATION STRATEGIES BASED ON EITHER MEETING 95% OF ALL DEADLINES OR ONLY PROVIDING THE MEAN CPU BANDWIDTH REQUIRED BY A TASK. COMPARISON OF PSNRS FOR DIFFERENT RESOURCE ALLOCATIONS

| Sequence | Allocated CPU Bandwidth | Deadline / Priority Class Deadline Miss % | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| *Silent* | 95% | 43.75 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Mean | 100.0 | 68.75 | 62.50 | 50.00 | 0 | 81.25 | 0 | 0 |
| *Stefan* | 95% | 43.75 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Mean | 100.0 | 87.50 | 75.00 | 0 | 0 | 87.50 | 0 | 0 |

| Sequence | PSNR (Proposed) | PSNR (95%) | PSNR (Mean) |
|---|---|---|---|
| *Silent* | 38.25 dB | 37.14 dB | 31.37 dB |
| *Stefan* | 35.03 dB | 32.06 dB | 26.46 dB |

TABLE IV
COMPARISON OF DIFFERENT FAIRNESS POLICIES. A NEGATIVE "QUALITY INCREASE FACTOR" MEANS THAT THE TASK ACHIEVES LESS QUALITY THAN THE MINIMUM QUALITY THAT IT DEMANDS WHEN THE KSBS IS DEPLOYED AS THE FAIRNESS POLICY

| | Ideal Point $\mathbf{X}_{max}$ (dB)<br>(*Foreman, Coastguard, Stefan, Silent*) | Disagreement Point $\mathbf{d}$ (dB)<br>(*Foreman, Coastguard, Stefan, Silent*) | Quality Increase Factor<br>$100 \times \frac{Q_i(c_i) - X_i^{min}}{X_i^{max} - X_i^{min}}$ |
|---|---|---|---|
| | (39.8 dB, 37.9 dB, 32.3 dB, 45.2 dB) | (25.0 dB, 25.0 dB, 25.0 dB, 25.0 dB) | |
| Fairness Policy | Quality $Q(c_i)$ (dB)<br>(*Foreman, Coastguard, Stefan, Silent*) | Resource $\propto c_i$ (%)<br>(*Foreman, Coastguard, Stefan, Silent*) | (*Foreman, Coastguard, Stefan, Silent*) |
| ERA | (31.3 dB, 27.1 dB, 22.1 dB, 45.0 dB) | (25.00 %, 25.00 %, 25.00 %, 25.00 %) | (42.31, 16.37, -40.05, 98.85) |
| GPS | (30.4 dB, 29.9 dB, 23.3 dB, 36.9 dB) | (23.95 %, 31.42 %, 27.81 %, 16.82 %) | (36.63, 38.02, -23.40, 59.08) |
| WMM | (32.0 dB, 27.6 dB, 22.6 dB, 41.7 dB) | (25.84 %, 26.16 %, 26.16 %, 21.85 %) | (47.04, 19.92, -33.64, 82.86) |
| KSBS | (30.2 dB, 29.5 dB, 27.5 dB, 32.1 dB) | (23.62 %, 30.59 %, 34.51 %, 11.28 %) | (34.94, 34.91, 34.96 , 34.96) |

values of $n$ correspond to frames of higher priority because future frames depend on them. In an H.264/AVC-based coder, for example, I-frames can be classified as having $n = 0$, P-frames $n = 1$, and B-frames $n = 2$. Table III illustrates the distribution of missed deadlines in a priority class for two cases. first, the CPU bandwidth assigned is statistically determined in order to meet 95% of the video decoding task's deadlines [1], [6]. Second, the CPU bandwidth is assigned as the task's Mean Bandwidth requirement $\rho$. In the latter case, many deadlines are missed across several classes. The mean bandwidth only ensures that over the duration of the super service interval $t_{\text{SI}}^{w}$ the task receives enough CPU cycles to complete, however, this does not guarantee that the instantaneous deadline-to-deadline bandwidth requirements are satisfied. The former case has many less missed deadlines, however, they all occur in the highest priority class which can adversely affect the quality of subsequent frames. The bottom half of Table III shows the quality impact of the Mean Bandwidth (labeled "Mean"), 95% deadline (labeled "95%"), and CPU Bandwidth Demand based (labeled "Proposed") resource allocations. Based on the PSNRs in Table III, it is clear that allocating resources to meet an arbitrary percent (e.g., 95%) of a task's deadlines significantly impacts the PSNR. In the top half of Table III, we do not include the case when the CPU bandwidth demand $g$ is assigned to a task, and each priority class's deadlines are increased to $t_\chi + 2n\Delta + d$, because no deadlines are missed. This additional delay, however, incurs a small penalty in memory requirements due to increased buffering. An analysis of the buffering overheads and peak CPU bandwidth savings when using the CPU bandwidth demand $g$ can be found in [10].

## B. Simulation Setup

Experiments are done via simulation. The Scalable CSPEC data for the *Stefan*, *Silent*, *Coastguard*, and *Foreman* sequences (CIF resolution, 30-Hz encoded frame-rate) is collected offline. We extract the CSPEC data to correspond to frames 1–256 for each sequence (i.e., the Peak workload, Mean workload, Delay, and CPU Bandwidth Demand are constant for the 256 frames). Additionally, each sequence's QC model parameters shown in Table II are determined. Lastly, we assume that each task is able to choose the optimal complexity strategy given its resource allocation [the optimal strategy is defined in (6)].

## C. Comparison of the KSBS With Other Fairness Policies

Here, we compare the KSBS from Section IV-D with the resource allocation policies in Section III-B. We perform the resource allocation for all four experimental sequences. Table IV illustrates the resulting quality and resource allocation for each fairness policy, the coordinates of the disagreement point $\mathbf{d}$ and the ideal point $\mathbf{X}_{\max}$, and the Quality Increase Factor (QIF) defined relative to $\mathbf{d}$ and $\mathbf{X}_{\max}$ as follows:

$$100 \times \frac{Q_i(c_i) - X_i^{\min}}{X_i^{\max} - X_i^{\min}}. \tag{27}$$

For the KSBS, $(Q_i(c_i) - X_i^{\min})/(X_i^{\max} - X_i^{\min}) = (Q_j(c_j) - X_j^{\min})(/X_j^{\max} - X_j^{\min}), \forall i, j \in \{1, \ldots, M\}$ as required by the fairness criterion defined in (21).

We use the metric in (27) to compare the different resource allocation policies because it captures the quality requirements for

TABLE V
EXAMPLE BARGAINING POWERS

| Bargaining Power | Sequence | | | |
|---|---|---|---|---|
| | *Foreman* | *Coastguard* | *Stefan* | *Silent* |
| MV bit-rate $\propto p_{MV}$ (bits/sec) | 62997<br>$(\alpha = 0.195)$ | 46483<br>$(\alpha = 0.144)$ | 185034<br>$(\alpha = 0.572)$ | 29086<br>$(\alpha = 0.089)$ |
| $p_{NZ}$ | 0.35<br>$(\alpha = 0.276)$ | 0.29<br>$(\alpha = 0.228)$ | 0.37<br>$(\alpha = 0.291)$ | 0.26<br>$(\alpha = 0.205)$ |

TABLE VI
KSBS WITH BARGAINING POWERS

| Fairness Policy | Resource $\propto c_i$ (%)<br>(*Foreman, Coastguard, Stefan, Silent*) | Quality $Q(c_i)$ (dB)<br>(*Foreman, Coastguard, Stefan, Silent*) |
|---|---|---|
| KSBS<br>$(\alpha \propto p_{MV})$ | (24.36 %, 28.83 %, 40.14 %, 6.66 %) | (30.75 dB, 28.70 dB, 33.28 dB, 28.32 dB) |
| KSBS<br>$(\alpha \propto p_{NZ})$ | (24.77 %, 30.34 %, 35.25 %, 9.64 %) | (31.08 dB, 29.39 dB, 28.15 dB, 30.70 dB) |

each task. Specifically, a QIF of 0 indicates that a task achieves its minimum desired quality; higher (positive) values of the QIF indicate that the task achieves a higher quality. A QIF of 100 indicates that a task achieves its maximum desired quality, and a negative QIF indicates that a task achieves below its minimum required quality.

Table IV clearly illustrates how the resource domain fairness policies fail to provide quality-fair resource allocations:

The results for the ERA policy demonstrate how an equal resource allocation does not guarantee equal qualities in the quality domain. For example, *Silent* receives 25% of the resources which corresponds to an imperceptibly distorted 45.0 dB PSNR, however, the 25% resource allocation to *Stefan* achieves only a 22.1 dB PSNR which is well below the task's minimum acceptable quality when using the KSBS (hence the negative QIF), and is intolerably distorted.

The GPS policy has significant variation in the quality degradation for each task. Most notably, *Silent* has a QIF of 59.08 while *Stefan* has a QIF of $-23.40$, which is negative because the resource domain fairness policies cannot guarantee a minimum quality.

The WMM policy (with equal weights) provides fairly equitable resource allocations to each task and, out of the GPS, WMM, and KSBS policies, has the maximum minimum resource allocation. The consequence of this, however, is that the task that required the least resources (*Silent*) received an unnecessary boost in quality to 41.7 dB. This large windfall for the *Silent sequence*, in turn, penalized *Stefan* significantly by starving it of necessary resources and decreasing its QIF to $-33.64\%$ corresponding to an achieved PSNR of 22.6 dB.

Finally, the KSBS provides almost equitable QIFs as it is designed to do based on the fairness criterion in (21). In fact, the QIFs for the KSBS only differ because of the nonzero tolerance used in the numerical bisection search for the Pareto optimal and fair quality allocation. Since the KSBS operates in the quality domain, it allocates significantly less resources to *Silent* compared to other fairness policies. Consequently, the resources that aren't allocated to *Silent* can be allocated to *Stefan*, which requires the most resources to achieve a fixed quality. The KSBS is effective because it explicitly considers the quality impact of the resource allocations.

### D. *Effect of Bargaining Powers*

Here, we illustrate how bargaining powers change the KSBS. Table V shows example bargaining power values for each of the four experimental sequences based on the motion vector (MV) bit-rates (which are proportional to the average number of motion vectors per pixel $p_{\mathrm{MV}}$) and $p_{\mathrm{NZ}}$ [2], [7], [9] which is the fraction of nonzero transform coefficients (taken from the sequences decoded at 384 kb/s). Table VI illustrates the KSBS resulting from these sets of bargaining powers. We make the following observations.

- *Motion-related bargaining powers*: When the bargaining powers $\alpha_i$ are proportional to $p_{i,\mathrm{MV}}$ the KSBS favors the tasks with high motion characteristics. In this case, *Stefan* is favored and it achieves a 33.28 dB PSNR. Since the bargaining powers are widely varying among the four experimental sequences, the quality allocation is also widely varying. Notably, the *Silent* sequence, which always achieved the highest PSNR in the KSBS without bargaining powers, now achieves the lowest PSNR (28.32 dB).
- *Texture-related bargaining powers*: When the bargaining powers $\alpha_i$ are proportional to $p_{i,\mathrm{NZ}}$ the KSBS favors the tasks with more nonzero transform coefficients. Since $p_{i,\mathrm{NZ}}$ is proportional to the average video bit-rate [17], these bargaining powers are similar for our set of test sequences, however, the high frequency textures of the crowd in the background of *Stefan* result in it having a higher bargaining power than the other sequences. *Silent*, on the other hand, has comparatively lower frequency textures (particularly in the residual error frames) and has a correspondingly lower bargaining power. Hence, compared to the KSBS without bargaining powers, *Stefan* improves by almost 1 dB while *Silent* drops over 1 dB.

We note that many other bargaining powers based on the sequence's content characteristics can be deployed [16].

## VI. CONCLUSION

In this paper, we propose a new system resource allocation framework for multimedia systems that perform multiple simultaneous video decoding tasks. We jointly consider the available system resources (e.g., processor cycles) and a video decoding task's characteristics such as the sequence's content, the bit-rate, and the GOP structure, in order to determine a quality-fair and Pareto optimal resource allocation using the KSBS from axiomatic bargaining theory. To drive the resource allocation, we characterize the video decoding workload using a twin leaky bucket traffic model from which we determine the task's CPU bandwidth demand given its latency/delay constraints. Using the CPU bandwidth demand and offline measurements, we derive a quality-complexity model that quantifies the tradeoffs between the video quality and the actual system complexity (e.g., processing time on a specific processor). We compare the KSBS to other fairness policies in the literature and find that because it explicitly considers multimedia quality it provides significantly fairer resource allocations than the other policies that operate solely in the resource domain.

## REFERENCES

[1] W. Yuan, K. Nahrstedt, S. V. Adve, D. L. Jones, and R. H. Kravets, "GRACE-1: Cross-layer adaptation for multimedia quality and battery energy," *IEEE Trans. Mobile Computing*, vol. 5, no. 7, pp. 799–815, Jul. 2006.

[2] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 704–716, Jul. 2003.

[3] J. R. Ohm, M. van der Schaar, and J. W. Woods, "Interframe wavelet coding—Motion picture representation for universal scalability," *Signal Process.: Image Commun.*, vol. 19, no. 9, pp. 877–908, Oct. 2004.

[4] D. K. Y. Yau and S. S. Lam, "Adaptive rate-controlled scheduling for multimedia applications," *IEEE/ACM Trans. Netw.*, vol. 5, no. 4, pp. 475–488, Aug. 1997.

[5] P. Goyal, X. Guo, and H. M. Vin, "A hierarchical CPU scheduler for multimedia operating systems," in *Proc. Usenix 2nd Symp. OS Design and Implementation*, 1996, pp. 107–122.

[6] W. Yuan and K. Nahrstedt, "Energy-efficient CPU scheduling for multimedia applications," *ACM Trans. Computer Syst.*, vol. 24, no. 3, pp. 292–331, Aug. 2006.

[7] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 5, pp. 645–658, May 2005.

[8] D. Turaga, M. van der Schaar, and B. Pesquet-Popescu, "Complexity scalable motion compensated wavelet video encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 8, pp. 982–993, Aug. 2005.

[9] M. van der Schaar and Y. Andreopoulos, "Rate-distortion-complexity modeling for network and receiver aware adaptation," *IEEE Trans. Multimedia*, vol. 7, no. 3, pp. 471–479, Jun. 2005.

[10] S. L. Regunathan, P. A. Chou, and J. Ribas-Corbera, "A generalized video complexity verifier for flexible decoding," in *Proc. Int. Conf. Image Process.*, Sep. 14–17, 2003, vol. 3, pp. III-289–III-92, vol. 2.

[11] A. Grilo, M. Macedo, and M. Nunes, "A scheduling algorithm for QoS support in IEEE802.11 networks," *IEEE Wireless Commun.*, vol. 10, no. 3, pp. 36–43, Jun. 2003.

[12] S. Lui, R. Rajkumar, and S. S. Sathaye, "Generalized rate-monotonic scheduling theory: A framework for developing real-time systems," *Proc. IEEE*, vol. 82, no. 1, pp. 68–82, Jan. 1994.

[13] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *J. ACM*, vol. 20, no. 1, pp. 46–61, Jan. 1973.

[14] N. C. Audsley, A. Burns, M. F. Richardson, and A. J. Wellings, "Hard real-time scheduling: The deadline monotonic approach," in *Proc. 8th IEEE Workshop Real-Time Operating Syst.*, Dec. 1989, pp. 166–171.

[15] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case," *IEEE/ACM Trans. Netw.*, vol. 1, no. 3, pp. 344–357, Jun. 1993.

[16] Y. Wang, M. van der Schaar, S. F. Chang, and A. Loui, "Content-based optimal MDA operation prediction for scalable video coding systems using subjective quality evaluation," *IEEE Trans. Circuits Syst. Video Technol.* , vol. 15, Special Issue on Analysis and Understanding for Video Adaptation, no. 10, Oct. 2005.

[17] Z. He and S. K. Mitra, "A unified rate-distortion analysis framework for transform coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 12, pp. 1221–1236, Dec. 2001.

[18] E. Kalai and M. Smorodinski, "Other solutions to Nash's bargaining problem," *Econometrica*, vol. 43, no. 3, pp. 513–518, May 1975.

[19] Z. He and D. Wu, "Resource allocation and performance analysis of wireless video sensors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 5, pp. 590–599, May 2006.

**Nicholas H. Mastronarde** (S'05) received the M.S. degree in electrical engineering at the University of California, Davis. in 2006. He is currently working toward the Ph.D. degree at the Electrical Engineering Department, University of California, Los Angeles (UCLA).

His research interests are in multimedia systems, multimedia complexity modeling, and multimedia system software design.

**Mihaela van der Schaar** (SM'00) received both the M.S. and Ph.D. degrees from Eindhoven University of Technology, Eindhoven, The Netherlands, in 1996 and 2001, respectively.

She is an Associate Professor of Electrical Engineering with the University of California, Los Angeles (UCLA), where she leads the Multimedia Communications and Systems Laboratory. She has been an active participant and chair in the ISO Motion Picture Expert Group (MPEG) international standard since 1999, to which she made more than 50 contributions and for which she received three ISO recognition awards. She is also co-editor (with P. Chou) of the book *Multimedia Over IP and Wireless Networks: Compression, Networking, and Systems* (Academic, 2007). She holds 30 granted U.S. patents.

Dr. van der Schaar was the recipient of a National Science Foundation Career Award in 2004, the IBM Faculty Award in 2005 and 2007, the Okawa Foundation Award in 2006, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY Best Paper Award in 2005, and the Most Cited Paper Award from *EURASIP Journal Signal Processing: Image Communication* between the years 2004–2006. She has served as an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA and *SPIE Electronic Imaging Journal* and is currently an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE SIGNAL PROCESSING LETTERS, and *IEEE Signal Processing Magazine e-Newsletter*.