On the Application of Game-Theoretic Mechanism Design for Resource Allocation in Multimedia Systems

Shih-Chung Su and Mihaela van der Schaar, *Senior Member, IEEE*

***Abstract*—In this paper, we study the system-level computational resource allocation problem among multiple multimedia tasks. We consider the multimedia tasks to be autonomous, i.e., they are selfish and behave strategically. We propose a resource allocation framework based on mechanism design to prevent the tasks from behaving strategically and manipulating the available system resources. We apply two mechanisms in the framework and assess their advantages over proportional-share resource allocation algorithms, which are often used in multimedia systems. We show in the simulations that the incorporation of mechanism design for system resource allocation is a promising solution that achieves efficient, fair and robust allocation against manipulation from strategic applications.**

***Index Terms*—Mechanism design, multimedia systems, multimedia tasks, system resource management.**

I. Introduction

With the advent of video streaming, IPTV, video surveillance, etc., multiple multimedia processing tasks, such as processing and compression, need to be executed concurrently and are required to share the available system resources, e.g., the CPU time, local and main memory, and cache, etc. Among them, the CPU time allocation is one of the most widely studied topics [3], [5]–[7]. Hence, we also focus in this paper on the problem of CPU processing time allocation among multimedia tasks, because this represents one of the main resource bottlenecks [14]. The resource allocation solutions studied in this paper, however, can also be applied to other types of system resources.

Early computer systems research addresses the problem of resource allocation using scheduling [4]. Scheduling focuses on determining when to allocate system resources to a specific task. In [4], the authors propose the Earliest-Deadline-First (EDF) policy that allocates resources to the task having the most imminent deadline. However, this policy is inefficient for multimedia applications because it does not take into account the soft deadlines characteristics of multimedia applications and the impact on multimedia quality. Alternatively, the SMART algorithm in [6] extends the EDF scheduling policy by adopting the proportional-share (PS) algorithm to prioritize the tasks based on their importance, as defined by each user. SMART iteratively allocates resources to competing tasks starting from the tasks having the highest priority and the most urgent deadline. However, the priority is computed using predetermined weights. Therefore, the algorithm cannot take advantage of the complexity-scalability of multimedia tasks to achieve different quality-resource tradeoffs.

Subsequent research takes into account the resource allocation's impact on the quality of the multimedia tasks [3], [8]. For instance, in [8], a piecewise linear utility function is introduced to characterize the quality of audio tasks as a function of the CPU utilization. In [3], the proposed resource allocation policy incrementally allocates the CPU processing time to each task proportionally to the user's importance/priority. At the application layer, each task finds the best configuration to maximize its quality for the given amount of CPU processing time. The quality-awareness of the architecture is based on the coarse assumption that the quality of a multimedia task increases linearly with the amount of allocated resources. However, the quality of each multimedia task scales differently depending on the underlying multimedia content and the algorithm (e.g., encoding) parameters.

A limitation of existing solutions on resource allocation in computer systems is that they heavily depend upon the applications declaring their resource demand in a truthful manner, and they lack robustness against strategic manipulation from applications. In this paper, we propose to complement existing solutions for multi-task resource allocation by explicitly considering the individual characteristics and the strategic behaviors of multimedia tasks using mechanism design. Our mechanism-design based framework complements previous research on system resource allocation [3], [5]–[7] by providing quality-aware resource allocation that is robust against manipulation. We model how the allocated resources impact the multimedia quality of the various tasks and propose a global adaptation framework for resource allocation in multimedia systems based on two mechanisms with different properties and overheads: the Vickrey–Clarke–Groves (VCG) mechanism [1] and a recently proposed budget-balanced pricing mechanism (BBPM) [2]. We analyze the features of these two mechanisms in the context of multimedia system resource allocation. Specifically, we study the resulting quality measurements and the associated overheads for the two mechanisms against those of the PS algorithm that is often used for resource allocation [3], [5]–[7].

This paper is organized as follows. In Section II, we introduce the global adaptation framework for system resource

Manuscript received March 30, 2007; revised April 08, 2008. First published October 7, 2008; current version published October 24, 2008. This work was supported by National Science Foundation Award 0509522. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Madjid Merabti.

The authors are with the Electrical Engineering Department, University of California Los Angeles (UCLA), Los Angeles, CA 90095-1594 USA (e-mail: shihchungsu@gmail.com; mihaela@ee.ucla.edu).

Digital Object Identifier 10.1109/TMM.2008.2001366

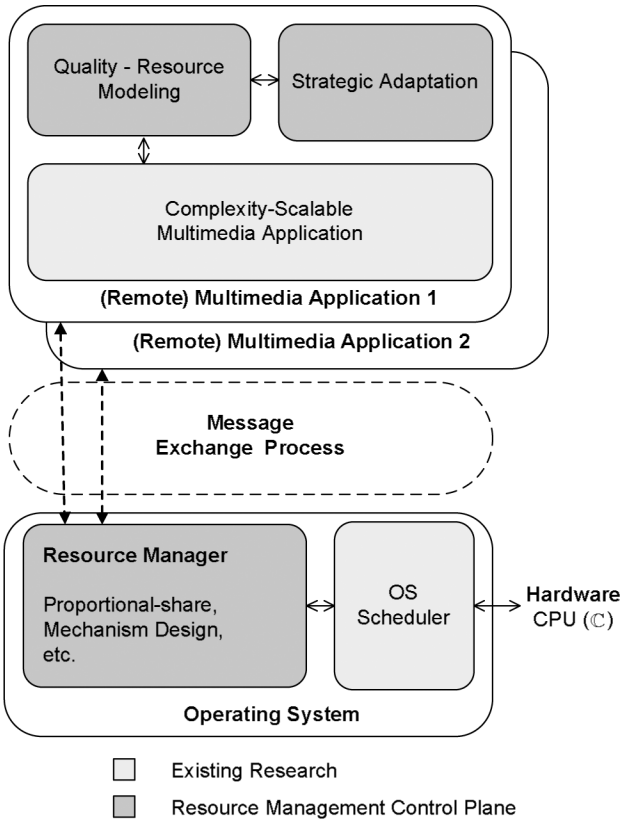

Fig. 1. Proposed global adaptation framework.

allocation and formulate the studied multi-task resource allocation problem. In Section III, we introduce the tasks' resource adaptation strategies, message exchanges between tasks and the system resource manager, and characterize the various allocations for multimedia systems when using the VCG and BBPM mechanisms. In Section IV, we analyze the overhead and optimality gap associated with our proposed global adaptation framework. In Section V, we compare the simulation results for the proposed global adaptation framework against the popular PS policy. The conclusions are presented in Section VI.

II. Multi-User Video Transmission—Problem Formulation and System Description

We propose a global adaptation framework for system resource allocation similar to the one defined in [3]. We enhance this framework by allowing each multimedia task to adopt a quality-complexity (Q-C) model to quantify its quality as a function of the amount of resources allocated to it. Based on its Q-C model, a task negotiates for resources with the system resource manager (RM). Moreover, we adopt mechanism design solutions for the computation of the resource allocation by the RM. In the following sections, we will discuss the reasons why we use mechanism design. First, we present an overview of the proposed framework.

A. Overview of the Framework

Fig. 1 shows the architecture of the global adaptation framework. The framework is a cross-layer resource allocation scheme that coordinates the information between the application layer and hardware layer to facilitate the resource allocation decision. The cross-layer information exchanges are implemented as part of a resource management control plane (RMCP). The RMCP supports two-way information exchanges between the OS layer and the application layer, e.g., the resource demands of tasks and the resource allocation decision by the RM. The RMCP also enables the RM to query the system for information on the available resources and the number of concurrently running tasks.

Since the implementation of the framework is not the focus of this paper, we refer the interested readers to [10] for more details about the implementation of such system software. The global adaptation framework is modeled as follows:

1) Hardware Layer: In this paper, we focus on the allocation of the computational resource in the form of CPU cycles. The maximum number of cycles per second $\mathbb{C}$ that a CPU is capable of providing is determined by the CPU frequency, which we assume to be fixed. We denote c_i to be the i-th task's CPU cycle demand. For generality, we represent the CPU resource demand of task i by normalized CPU utilization α_i, where

$$\alpha_i = \frac{c_i}{\mathbb{C}}. \tag{1}$$

By definition, the normalized cycle value satisfies $0 \leq \alpha_i \leq 1$.

2) Application Layer: We assume there are N video decoding tasks indexed by $i = 1, \ldots, N$ waiting to be serviced. Each decoding task requires some number of CPU cycles for its successful execution. This demand is represented by the normalized CPU utilization α_i, which we refer to as the CPU utilization of task i. In this paper, the utilization is measured by profiling the reference H.264 decoder in [9] with queries to a high-resolution CPU timer through a Windows API [15] to gather the number of cycles required to decode a sequence. The CPU utilization of a video decoding task depends on the video sequence and the decoding parameters, which we abstract by the complexity profile ρ_i. For simplicity, we encapsulate these factors into a joint variable v_i, i.e., $v_i = (\theta_i, \rho_i)$, called the "type" of task i. Each task i can operate at various complexity levels by adjusting its complexity profile (i.e., decoding parameters) and has a quality-complexity (Q-C) model $Q_{v_i}(\alpha_i)$ that specifies the video quality given the type v_i of the task and the CPU utilization α_i [13]. In this paper, the Q-C models are derived using the methodology described in the Appendix. In the RMCP, each task is represented by an agent [18], which negotiates with the RM to determine the resource allocation to the task, and a commensurate complexity profile. For clarity, we refer to the information submitted by agents as "messages" transmitted by tasks in the RMCP.

3) OS Layer: A global adaptation is initiated by the RM whenever there are significant variations in the resource demands, e.g., when a task enters or exits the system. Through the RMCP, the RM and the tasks are involved in a message exchange process, where the content of the messages in the RMCP depends on the employed resource allocation policy. The choice of resource allocation policy depends on the application, e.g., mechanism design or PS. At the end of the message exchange process, the RM determines a resource allocation decision and enforces the allocation through the OS scheduler until the next

TABLE I
RESOURCE ALLOCATION AMONG THREE MULTIMEDIA TASKS WHEN TASK 1 MANIPULATES THE RESOURCE ALLOCATION BY MISREPORTING ITS SEQUENCE TO BE THE MOBILE SEQUENCE

Task - Sequence	Truthful revelation		Manipulation	
	Resource	Quality	Resource	Quality
Task 1 - Foreman	0.300	45.15	0.404	47.57
Task 2 - Coastguard	0.400	41.69	0.362	40.71
Task 3 - Foreman	0.300	45.15	0.234	43.50

invocation of global adaptation. The scheduler can adopt any existing scheduling policy, e.g., EDF. However, we note that scheduling is not the focus of this paper. Finally, tasks are executed with the decoding parameters derived from the complexity profile of their Q-C models.

B. Problem Formulation

The resource allocation problem for the global adaptation framework can be formulated as follows:

$$\begin{aligned}\boldsymbol{\alpha}^* &= \arg\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} U_i(\alpha_i, \tau_{\mu,i}) \\ &= \arg\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} [Q(\alpha_i) + \tau_{\mu,i}] \\ &\text{s.t. } \sum_{i=1}^{N} \alpha_i \le 1, \boldsymbol{\alpha} \in [0,1]^N \end{aligned} \tag{2}$$

where $U_i(\alpha_i, \tau_{\mu,i})$ represents the utility function of task given the allocated resource α_i and the transfer $\tau_{\mu,i}$. The transfer $\tau_{\mu,i}$ depends on the employed mechanism μ and will be introduced in greater detail in Sections III-B and III-C. In our framework, the RM utilizes the transfer to encourage truthful revelation of each task's type. If no transfer is employed, as in traditional resource allocation frameworks, the utility function of task i reduces to its Q-C model, i.e., $U_i(\alpha_i, \tau_{\mu,i}) = Q_{v_i}(\alpha_i)$.

Equation (2) can be solved by the RM without employing any mechanism design concept. However, under traditional quality-aware resource allocation frameworks, tasks can take strategic behaviors and deceive the RM by revealing false information. For example, a task can misreport its video characteristics, e.g., $\tilde{\theta}_i = \text{mobile}$ when $\theta_i = \text{foreman}$, in order to gain more resource than it requires. Table I shows the resource allocation decision among three video decoding tasks (tasks 1–3) when one of the tasks strategically misreports its sequence. We assume task 1 is the strategic task and represents the video decoding task of the Foreman sequence, i.e., $\theta_1 = \text{Foreman}$. However, instead of reporting its true sequence, it reveals itself to be a decoding task of a more complex sequence, i.e., the Mobile sequence $\tilde{\theta}_1 = \text{Mobile}$. To highlight the effect of manipulation, we assume the other tasks declare their true types, i.e., $\theta_2 = \text{Coastguard}$ and $\theta_3 = \text{Foreman}$.

Compared with the resource allocation when all tasks reveal their true types, the result shows that the manipulation by task 1 causes task 2 and 3 to be admitted at a lower quality. Previous quality-aware resource allocation frameworks [3], [8] cannot successfully address this unfairness caused by tasks' manipulation because the RM could not verify the truthfulness of tasks until the actual execution. Even if the RM could detect such strategic behaviors after the execution of tasks, the RM would have to discard the previous resource allocation and re-initiate the global adaptation.

III. MECHANISM DESIGN FOR MULTIMEDIA SYSTEMS

In this section, we introduce two game-theoretic mechanism designs for the global adaptation framework. In Section III-A, we present the desired properties of a mechanism design for the global adaptation framework. In Section III-B, we introduce the VCG mechanism. Section III-C presents the BBPM mechanism. In Section III-D, we discuss the convergence issue associated with the BBPM. Lastly, we discuss the constraints on the Q-C model and propose a piecewise linear approximation of the Q-C model in Section III-E.

A. Desired Mechanism Design Properties

To prevent tasks from untruthfully declaring their resource requirements, mechanism design introduces the concept of taxation, which is referred to as the transfer and denoted by τ_i for task i (the transfer can be tokens or another form of currency adopted by the system to tax the various multimedia tasks). The transfer integrated into each task's utility is designed to penalize the task by increases in its transfer. Since each task aims to maximize its utility, the employment of transfer effectively deters manipulation by tasks. Importantly, the actual value of a transfer depends on the deployed mechanism.

A game-theoretic mechanism is devised by its mechanism designer to achieve certain desired properties at the equilibrium [1]. Specifically, a mechanism designer formulates the mapping function $f_\mu(\cdot)$, the messages of the RM $\boldsymbol{m}^{\text{RM}} = (m_1^{\text{RM}}, \ldots, m_N^{\text{RM}})$, the messages of tasks $\boldsymbol{m}^{\text{task}} = (m_1^{\text{task}}, \ldots, m_N^{\text{task}})$ such that $f_\mu(\boldsymbol{m}^{\text{RM}}, m^{\text{task}}) = (\boldsymbol{\alpha}^*, \boldsymbol{\tau}_\mu^*)$, where $\boldsymbol{\alpha}^* = (\alpha_1^*, \ldots, \alpha_N^*)$ and $\boldsymbol{\tau}_\mu^* = (\tau_{\mu,1}^*, \ldots, \tau_{\mu,N}^*)$ are the allocated resources and transfer, which achieve the properties desired by the designer. In the following subsections, we will discuss the mapping functions in greater details in Section III-B and III-C.

For the resource allocation problem considered in this paper, there are several properties that a mechanism design should possess.

1) Efficiency: In our proposed framework, the RM utilizes mechanism design to solve the resource allocation problem in (2). Hence, the resulting outcome $(\boldsymbol{\alpha}^*, \boldsymbol{\tau}_\mu^*)$ is utility-maximizing, which is called "efficient" [17]. Mathematically, an efficient outcome $(\boldsymbol{\alpha}^*, \boldsymbol{\tau}_\mu^*)$ is the solution to the following problem:

$$(\boldsymbol{\alpha}^*, \boldsymbol{\tau}_\mu^*) = \arg \max_{(\boldsymbol{\alpha}, \boldsymbol{\tau}_\mu) \in S} \left\{ \sum_{i=1}^{N} U_i(\alpha_i, \tau_{\mu,i}) \right\} \tag{3}$$

where $S = \{(\boldsymbol{\alpha}, \boldsymbol{\tau}_\mu) | \sum_{i=1}^{N} \alpha_i \le 1, \sum_{i=1}^{N} \tau_{\mu,i} \le 0, 0 \le \alpha_i \le 1, \forall i \in \{1, \ldots, N\}\}$. The VCG and BBPM mechanisms employed in this paper have been shown to satisfy efficiency [1], [2].

2) Incentive Compatibility: For tasks, incentive compatibility means that truthful revelation of their types is the optimal strategy to maximum their own utilities. It also implies that the local utility-maximization of each task will lead to the global maximization of the system utility. In other words, an incentive compatible mechanism design intrinsically encourages truth-revelation of the types of various tasks. Both the VCG and BBPM mechanisms are known to be incentive compatible [1], [2].

3) Individual Rationality: Individual rationality states that the utilities of tasks joining the global adaptation framework will not be worse off after the allocation, i.e.,

$$\sum_{i=1}^{N} U_i(\alpha_i, \tau_{\mu,i}) \geq 0, (\boldsymbol{\alpha}, \boldsymbol{\tau}_\mu) \in S. \tag{4}$$

Tasks have zero utility if they do not join the resource allocation framework. Hence, (4) can be interpreted as motivation for tasks to voluntarily participate in the resource allocation. Both mechanisms possess the property of individual rationality [1], [2].

In addition to the properties mentioned above, the BBPM is shown to be budget balanced [2]. The property of budget balance states that the net value of the transfer imposed by the RM for all tasks is zero at equilibrium, i.e.,

$$\sum_{i=1}^{N} \tau_{\mu,i} = 0. \tag{5}$$

Therefore, tasks can achieve higher system utility under the BBPM mechanism than the VCG mechanism.

The other difference between the VCG mechanism and the BBPM is the equilibrium concept. The VCG mechanism converges to the so-called dominant strategy equilibrium, which states that for each task i, there exists a message m_i^{task} irrespective of the other tasks' messages $\boldsymbol{m}_{-i}^{\text{task}} = (m_1^{\text{task}}, \ldots, m_{i-1}^{\text{task}}, m_{i+1}^{\text{task}}, \ldots, m_N^{\text{task}})$ such that

$$m_i^{\text{task}} = \arg\max_{m_i^{\text{task}'}} \left\{ U_i \left(\left[f_\mu \left(\boldsymbol{m}^{\text{RM}}, \left(m_i^{\text{task}'}, \boldsymbol{m}_{-i}^{\text{task}} \right) \right) \right]_i \right) \right\}, \quad \forall \boldsymbol{m}_{-i}^{\text{task}} \in \prod_{\substack{j=1 \\ j \neq i}}^{N} M_j \tag{6}$$

where $[f_\mu(m_i, \boldsymbol{m}_{-i})]_i = (\alpha_i^*, \tau_{\mu,i}^*)$ and M_j is the feasible message space for task $j \neq i$. Under the dominant strategy equilibrium, a task does not need to know the messages of the other tasks to maximize its own utility.

The BBPM implements another type of equilibrium concept, the Nash equilibrium. At Nash equilibrium, given a specific equilibrium message m_{-i}^{task} of the other tasks, the message m_i^{task} of task i satisfies

$$m_i^{\text{task}} = \arg\max_{m_i^{\text{task}'}} \left\{ U_i \left(\left[f_\mu \left(\boldsymbol{m}^{\text{RM}}, \left(m_i^{\text{task}'}, \boldsymbol{m}_{-i}^{\text{task}} \right) \right) \right]_i \right) \right\}. \tag{7}$$

Compared with (6), the Nash equilibrium concept is weaker because for each task i, the utility is maximized only under the assumption that the messages m_{-i}^{task} of tasks other than i are unchanged. This implies that a task must know the messages of the other tasks in order to maximize its own utility under Nash equilibrium. However, this does not necessarily suggest that the BBPM will have higher message exchange overhead. The overhead will be further discussed in Section IV-B.

In summary, the properties of efficiency and incentive compatibility suggest that a global adaptation framework based on mechanism design does not require frequent monitoring of the system to prevent strategic behaviors from tasks. The property of individual rationality is essential for the tasks to voluntarily join the global adaptation. Lastly, the budget balance property results in a higher system utility for the BBPM mechanism. Ideally, we would like to have a mechanism design that incorporates all of the above properties. In the next subsections, we discuss how to implement the various mechanisms for multimedia systems.

B. VCG Mechanism

In the VCG mechanism, the tasks and RM are engaged in a single-round message exchange process through the RMCP. The message exchange process of the VCG mechanism consists of the following steps:

Step 1. RMCP Messages of Multimedia Tasks: The message m_i^{task} that each task i submits to the RM contains the parameters of its Q-C model $Q_{v_i}(\alpha_i)$. We note that a task could be designed to reveal false information in an effort to gain as much resources as possible. However, we will show that the transfer defined in the VCG mechanism makes "truth-telling" the best strategy for all the tasks.

Step 2. Allocation Decision: Based on the tasks' RMCP messages, the RM solves the global adaptation resource allocation problem in (2) to determine the optimal allocation $\boldsymbol{\alpha}^*$ that maximizes the sum of tasks' qualities. In the VCG mechanism, the objective is to maximize the system quality. However, we will show later that such maximization also maximizes the system utility. Let $Q_{\tilde{v}_i}(\alpha_i)$ denote the Q-C model based on the revealed type $\tilde{v}_i$ of task i. The optimization problem becomes:

$$\begin{aligned} \boldsymbol{\alpha}^* &= \arg\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} Q_{\tilde{v}_i}(\alpha_i) \\ &= \arg\max_{\boldsymbol{\alpha}} \left[Q_{\tilde{v}_i}(\alpha_i) + \sum_{\substack{j=1 \\ j \neq i}}^{N} Q_{\tilde{v}_j}(\alpha_j) \right] \\ &\text{s.t.} \sum_{i=1}^{N} \alpha_i \leq 1, \boldsymbol{\alpha} \in [0,1]^N. \end{aligned} \tag{8}$$

Step 3. Transfer Computation: Based on the resource allocation decision derived in Step 2, the RM computes the transfer $\tau_{\text{VCG},i}$ for each task i. In the VCG mechanism, the transfer is the RMCP message by the RM, i.e., $m_i^{\text{RM}} = \tau_{\text{VCG},i}$, and is defined to be the total quality loss of all tasks due to the presence of task i in the resource allocation. Let $\boldsymbol{\alpha}_{-i}^*$ be the solution to the optimization problem in (8) without task i participating in the resource allocation. The transfer is computed by:

$$\tau_{\text{VCG},i}^*(\boldsymbol{\alpha}^*) = \sum_{j \neq i} Q_{\tilde{v}_j}(\alpha_j^*) - \max_{\boldsymbol{\alpha}_{-i}} \sum_{\substack{j=1 \\ j \neq i}}^{N} Q_{\tilde{v}_j}(\alpha_j). \tag{9}$$

The first term in (9) calculates the sum of quality of all tasks excluding task i when task i participates in the resource allocation. The second term is the other tasks' aggregated valuation without task i's participation in the allocation. It is clear from (9) that the transfer will always be non-positive. Plugging (9) into (8), the utility of task i is described by

$$\begin{aligned} &U_i(\alpha_i^*, \tau_{\mathrm{VCG},i}) \\ &\quad = Q_{v_i}(\alpha_i^*) + \tau_{\mathrm{VCG},i}^*(\alpha^*) \\ &\quad = \left[Q_{v_i}(\alpha_i^*) + \sum_{\substack{j=1 \\ j\neq i}}^{N} Q_{\tilde{v}_j}(\alpha_j^*) \right] - \max_{\boldsymbol{\alpha_{-i}}} \sum_{\substack{j=1 \\ j\neq i}}^{N} Q_{\tilde{v}_j}(\alpha_j). \end{aligned} \tag{10}$$

Under the VCG mechanism, revealing the true type is the best strategy for task i. This can be seen from (10), which states that task i tries to maximize its utility function by manipulating its revealed type $\tilde{v}_i$. However, the last term in (10) is independent of the strategic behavior of task i. Therefore, the best that a task can hope for by manipulating its type is to maximize the terms in the square bracket in (10), i.e.,

$$\max_{\hat{v}_i} \left[Q_{v_i}(\alpha_i^*) + \sum_{\substack{j=1 \\ j\neq i}}^{N} Q_{\tilde{v}_j}(\alpha_j^*) \right]. \tag{11}$$

Comparing (11) with (8), the best strategy for task i in order to maximize its utility is to reveal its true type v_i, irrespective of other tasks' revealed types $\tilde{v}_{-i}$. In conclusion, the VCG mechanism is utility maximizing even though the objective of the mechanism is to maximize the system quality.

C. The Budget-Balanced Pricing Mechanism

By relaxing the constraint of dominant strategy equilibrium, in [2] a budget-balanced pricing mechanism that is Nash equilibrium incentive compatible is introduced. Unlike the VCG mechanism, the BBPM is budget balanced and can achieve higher utility. In the BBPM, tasks iteratively exchange messages through the RMCP. Moreover, tasks do not directly reveal their true valuations, e.g., the parameters of the Q-C models, to the RM. Instead, their valuations are represented by a single variable p_i, which can be thought of as a pricing. The details of the BBPM are as follows:

Step 1. Initialization: The RM determines an initial resource allocation α_i^0 and initial price p_i^0 for each task. Due to the possible strategic manners of tasks, the best policy for the resource manager is to treat all tasks equally by assigning each of them with the same resource allocation and price, e.g., $\alpha_i^0 = 1/N, p_i^0 = 1/N, \forall i \in \{1, \ldots, N\}$.

Step 2. RMCP Messages of Multimedia Tasks: The RM and the tasks iteratively exchange the pricing information based on the tasks' CPU utilizations. The step is repeated until equilibrium is reached:

2.1 RMCP Messages of the RM. Under the BBPM, the RM conveys two variables, $m_i^{\mathrm{RM}} = (p_{-i}, d_i)$, to each task i as the RMCP message. Let p_i represents task i's valuation of the resource. For the first iteration, we have $\alpha_i = \alpha_i^0$ and $p_i = p_i^0$. The RMCP messages of the RM $m_i^{\mathrm{RM}} = (p_{-i}, d_i)$ is defined by

$$p_{-i} = \frac{1}{N-1} \sum_{\substack{j=1 \\ j\neq i}}^{N} p_j \tag{12}$$

which represents the average price per unit of resource from the other tasks, and

$$d_i = \sum_{\substack{j=1 \\ j\neq i}}^{N} \alpha_j - 1 \tag{13}$$

which is the excess resource demand excluding the demand from task i. The messages from the RM are used by tasks to compute their transfer.

2.2 *RMCP Messages of Multimedia Tasks*. Given the message $m_i^{\mathrm{RM}} = (p_{-i}, d_i)$ transmitted by the RM, each task i maximizes its utility by solving the following resource allocation problem:

$$\begin{aligned} \max_{\alpha_i, p_i} U_i(\alpha_i, \tau_{\mathrm{BBPM},i}) = \max_{\alpha_i, p_i} \{ & Q_{v_i}(\alpha_i) \\ & + \tau_{\mathrm{BBPM},i}\left(\alpha_i, p_i, m_i^{\mathrm{RM}}\right) \} \text{ s.t. } 0 \le \alpha_i \le 1. \end{aligned} \tag{14}$$

The transfer function $\tau_{\mathrm{BBPM},i}$ in (14) is defined by

$$\begin{aligned} \tau_{\mathrm{BBPM},i}(\alpha_i, p_i, m_i^{\mathrm{RM}}) = & -(\alpha_i - \alpha_i^0) p_i \\ & - \left[p_i - p_{-i} \left(1 + \frac{d_i + \alpha_i}{\gamma} \right) - \chi_+(d_i, \alpha_i, \gamma) \right]^2 \end{aligned} \tag{15}$$

where

$$\chi_+(d_i, \alpha_i, \gamma) = \max \left\{ 0, \frac{d_i + \alpha_i}{\gamma} \right\}. \tag{16}$$

The first term in (15) represents the price task i pays/earns for buying/selling $(\alpha_i - \alpha_i^0)$ amount of resource from/to the other users. The second term is the penalty that task i pays due to the mismatch of its price to the average price of the other tasks p_{-i}. The third term $\chi_+(\cdot)$ in (15) is introduced to prevent the solution from reaching an inefficient Nash equilibrium such as i) $p_i = 0$ for all the users and ii) the total demand exceeds the available resource, i.e., $\sum_{i=1}^{N} \alpha_i > 1$. The variable γ in (15) can be thought of as the step size for updating the price. Its influence on the convergence of the mechanism will be discussed in the next subsection.

Plugging (15) into (14), the optimization problem can be written as

$$\begin{aligned} &\max_{\alpha_i, p_i} U_i(\alpha_i, \tau_{\mathrm{BBPM},i}) \\ &\quad = \max_{\alpha_i, p_i} \left\{ Q_{v_i}(\alpha_i) - (\alpha_i - \alpha_i^0) \times p_{-i} \right. \\ &\qquad \left. - \left[p_i - p_{-i} \left(1 + \frac{d_i + \alpha_i}{\gamma} \right) - \chi_+(d_i, \alpha_i, \gamma) \right]^2 \right\}. \end{aligned} \tag{17}$$

Observe that the third term in (17) is non-negative no matter what the values of (α_i, p_i) are. Therefore, the maximization of the utility can only occur when the transfer is minimized, i.e.,

$$p_i = p_{-i}\left(1 + \frac{d_i + \alpha_i}{\gamma}\right) + \chi_+(d_i, \alpha_i, \gamma). \tag{18}$$

Hence, the problem in (17) can be decomposed by first solving the following equation

$$\max_{\alpha_i} \left\{ Q_{v_i}(\alpha_i) - \left(\alpha_i - \alpha_i^0\right) \times p_{-i} \right\} \tag{19}$$

and then determining the price according to (18). At the end of the step, task i submits its message to the RM, i.e., $m_i^{\text{task}} = (\alpha_i, p_i)$.

Step 3. Allocation Decision: After reaching the equilibrium, the RM allocates the resource based on the equilibrium message $(\boldsymbol{\alpha}^*, \boldsymbol{p}^*)$, where $\boldsymbol{\alpha}^* = (\alpha_1^*, \ldots, \alpha_N^*)$ and $\boldsymbol{p}^* = (p_1^*, \ldots, p_N^*)$. At equilibrium, the squared term in (15) becomes zero. Hence, the transfer for task i calculated by the RM is

$$\tau_{BBPM,i}^*(\boldsymbol{\alpha}^*, \boldsymbol{p}^*) = -(\alpha_i^* - \alpha_i^0) \times p_{-i}^*. \tag{20}$$

D. Convergence of the BBPM

The BBPM introduced in Subsection III.C is an iterative algorithm and its convergence depends on the value of the step size γ. From (18), the value of γ is inversely proportional to the amount that the price p_i changes per iteration. Therefore, a small γ may cause the price p_i to oscillate and affect the convergence of the mechanism. On the other hand, if γ is too large, the mechanism will exhibit a slow convergence rate. One solution to satisfy the convergence speed and the stability of the mechanism is to adopt a variable step size scheme, i.e., a small step size is used as the initial value and then changed to a larger value in the subsequent iterations in order to guarantee convergence [21].

In addition to the step size, the initial price p_{-i} that the RM conveys to the tasks also affects the convergence speed. If the gap between p_{-i} and the equilibrium price is large, the message exchange process will take more iterations to reach convergence.

E. Constraint on the Q-C Model and a Piecewise Linear Approximation of the Model

The two mechanisms discussed above impose certain constraints on the Q-C model in order to produce an efficient allocation. The VCG mechanism assumes a quasi-linear utility function in order to satisfy the properties discussed in Section III-A, [1]. On the other hand, the BBPM imposes stricter constraints—strict concavity and differentiability—on the utility function in order to guarantee efficiency [2]. We note that the utility functions defined in (8) and (17) are quasi-linear by definition. Hence, the VCG mechanism will always produce efficient allocation within the scope of discussion in this paper. For the BBPM, since the transfer in (14) is a function of tasks' Q-C model, the constraints of strict concavity and differentiability applies directly to the Q-C model. For strict concavity, it has been shown that the Q-C models of efficient video coders satisfy the property [19], [20]. Differentiability limits the feasible Q-C model to be a continuous function for an efficient allocation.

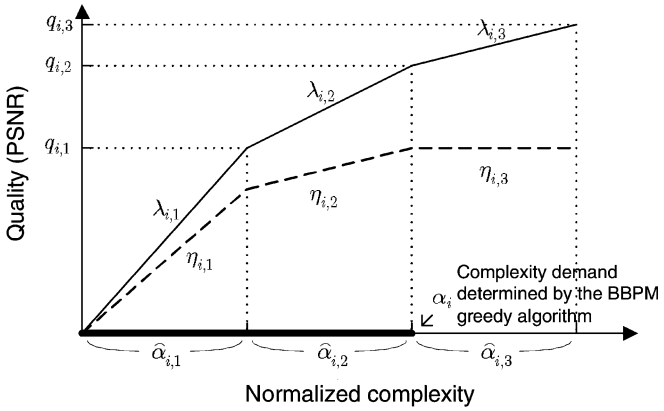

Fig. 2. Example of the piecewise linear Q-C model for task i when the number of segments of the model is $\Omega_i = 3$.

The optimization problems in (8) and (17) can be solved by nonlinear programming techniques. However, the associated computation demand may prohibit the deployment of the proposed framework in real-time applications. By relaxing the constraints on the Q-C model and utilizing a piecewise linear Q-C model, we can devise heuristics that reduce the computation overhead. Specifically, we assume a Ω_i - segment piecewise linear Q-C model as follows:

$$q_i = Q_{v_i}(\alpha_i) = q_{i,k} + \lambda_{i,k}\left(\alpha_i \sum_{j=0}^{k-1} \widehat{\alpha}_{i,j}\right),$$
$$\sum_{j=0}^{k-1} \widehat{\alpha}_{i,j} \le \alpha_i \le \sum_{j=0}^{k} \widehat{\alpha}_{i,j}$$
$$\text{where} \quad \lambda_{i,j} = \frac{q_{i,j} - q_{i,j-1}}{\widehat{\alpha}_{i,j}}, \sum_{j=0}^{\Omega_i} \widehat{\alpha}_{i,j} = 1, q_{i,0} = 0, \widehat{\alpha}_{i,0} = 0. \tag{21}$$

An example of the piecewise linear Q-C model is illustrated in Fig. 2. A piecewise linear Q-C model does not satisfy the strict concavity and differentiability properties. Therefore, the resource allocation based on a piecewise linear model will be suboptimal. We will study the gap to optimality in Section IV-C.

IV. Quantifying the Overheads of the Resource Allocation Mechanisms

A. Computational Overhead

Based on the assumption of the piecewise linear Q-C model proposed in Section III-E, we propose two greedy algorithms that have linear complexity for both mechanisms. First of all, we note that in the piecewise linear Q-C model introduced in Section III-E, the quality gain per unit resource $\lambda_{i,j}$ must be non-increasing with the increment of j, i.e., $\lambda_{i,1} \ge \cdots \ge \lambda_{i,\Omega_i}$ since the empirically-derived Q-C models in the Appendix are concave. To facilitate the VCG greedy algorithm, we sort $\lambda_{i,j}$ in descending order. If $\lambda_{i,j} = \lambda_{i',j'}$, the order is randomly determined. We denote the ordered list of $\lambda_{i,j}$ by $\bar{\lambda}_l$ for $1 \le l \le \Pi$

1. **Initialization:** for $i = 1 : N$, $\alpha_i \leftarrow 0$, $q_{total} \leftarrow 0$.

Sort $\lambda_{i,j}$ in descending order into $\bar{\lambda}_l$.

2. **Repeat:**

3. for $l = 1 : \Pi$

4. if $(q_i < q_{i,\Omega_i})$

5. find: i such that $l \in S(i)$

6. update: $\alpha_i \leftarrow \alpha_i + \min\{\alpha_{left}, \bar{\alpha}_l\}$, $\alpha_{left} \leftarrow 1 - \sum_{i=1}^{N} \alpha_i$

$q_i \leftarrow q_i + \bar{\lambda}_l \min\{\alpha_{left}, \bar{\alpha}_l\}$, $q_{total} \leftarrow q_{total} + q_i$

7. **Until:** $\alpha_{left} \leq 0$ or $(q_i = q_{i,\Omega_i}, \forall i)$

Fig. 3. Greedy algorithm for the VCG mechanism.

where Π is the total number piecewise linear components across all of the tasks, i.e., $\Pi = \sum_{i=1}^{N} \Omega_i$ in our example. We also denote the increase in CPU utilization associated with $\bar{\lambda}_l$ by $\bar{\alpha}_l$. For example, if the descending list of quality gain per unit resource comes from $\{\lambda_{1,1} \geq \lambda_{2,1} \geq \lambda_{1,2}\} \rightarrow \{\bar{\lambda}_1, \bar{\lambda}_2, \bar{\lambda}_3\}$, we have the associated CPU utilization increment indexed by $\{\widehat{\alpha}_{1,1}, \widehat{\alpha}_{2,1}, \widehat{\alpha}_{1,2}\} \rightarrow \{\bar{\alpha}_1, \bar{\alpha}_2, \bar{\alpha}_3\}$.

The VCG greedy algorithm allocates the resource in descending order of the quality gain per unit resource, as specified by $\bar{\lambda}_l$, until there is no resource available or all tasks have reached their maximum quality q_{i,Ω_i}. Let $S(i)$ be the index set of the reordered utility gain per unit time $\bar{\lambda}_l$ of task i. Continuing with the prior example, we have $\{\bar{\lambda}_1, \bar{\lambda}_3\}$ belonging to task 1. This leads to $S(1) = \{1, 3\}$. The greedy algorithm for the optimization problem in (8) is shown in Fig. 3. The algorithm can also be applied to the transfer computation in (9). The computational complexity of the algorithm consists of sorting the quality gain per unit resource and iteratively updating the variables. Since the sorting is the dominant factor and needs only to be done once, the overall computational complexity of the VCG mechanism is $\mathrm{O}(\Pi \log \Pi)$.

For the BBPM, due to the iterative nature it is more difficult to quantify the computational complexity. However, we can study the complexity it takes to solve (8) per message exchange iteration. Plugging in the piecewise linear Q-C model, the optimization in (19) can be reformulated as

$$\begin{aligned}
&\max_{\alpha_i} \left\{ Q_{v_i}(\alpha_i) - (\alpha_i - \alpha_i^0) \times p_{-i} \right\} \\
&= \max_{t_i \in \{1,\ldots,\Omega_i\}} \left\{ \sum_{j=1}^{t_i} \lambda_{i,j} \widehat{\alpha}_{i,j} - \left(\sum_{j=1}^{t_i} \widehat{\alpha}_{i,j} - \alpha_i^0 \right) p_{-i} \right\} \\
&= \max_{t_i \in \{1,\ldots,\Omega_i\}} \left\{ \sum_{j=1}^{t_i} (\lambda_{i,j} - p_{-i}) \widehat{\alpha}_{i,j} + p_{-i} \alpha_i^0 \right\} \\
&= \max_{t_i \in \{1,\ldots,\Omega_i\}} \left\{ \sum_{j=1}^{t_i} \eta_{i,j} \widehat{\alpha}_{i,j} + p_{-i} \alpha_i^0 \right\}. \qquad (22)
\end{aligned}$$

Hence, the maximizing value of t_i depends on the term $\eta_{i,j} = \lambda_{i,j} - p_{-i}$, which represents the quality gain per unit resource revised by p_{-i}. An example of the revised quality gain per unit resource is illustrated in Fig. 3 as the dotted line. Due to the concavity of the Q-C model, $\eta_{i,j}$ decreases as the index j increases. In order to maximize (22), each task i should increase its resource demand until the revised quality gain per unit resource becomes less than or equal to zero, i.e., $\eta_{i,t_i} \leq 0, \forall i$. Based on the above observation, we devise the algorithm in Fig. 4 for the BBPM optimization problem in (14).

1. for $i = 1 : N$

2. compute: $\eta_{i,j} \leftarrow (\lambda_{i,j} - p_{-i}), \forall j$.

3. find: $t_i = \arg\max_j \{\eta_{i,j} > 0\}$

4. update: $\alpha_i \leftarrow \sum_{j=1}^{t_i} \widehat{\alpha}_{i,j}$, $p_i \leftarrow p_{-i}(1 + \frac{d_i + \alpha_i}{\gamma}) + \chi_+(d_i, \alpha_i, \gamma)$

$q_i \leftarrow \sum_{j=1}^{t_i} \lambda_{i,j} \widehat{\alpha}_{i,j}$

Fig. 4. Greedy algorithm for the BBPM.

The computational complexity of the algorithm is $\mathbf{O}(\Pi)$, where $\Pi = \sum_{i=1}^{N} \Omega_i$. However, from the simulation results that will be presented in Section V, the BBPM takes several iterations to reach the equilibrium. Hence, the lower per-iteration complexity for the BBPM does not guarantee that the overall complexity will be less than the VCG mechanism.

B. Message Exchange Overhead

The message exchange overhead introduces additional delay, which is undesirable for real-time applications Therefore, we quantify the message exchange overhead as the number of parameters exchanged by the tasks and the RM per message exchange iteration. In the VCG mechanism, the message of the RM to each task i is the transfer, i.e., $m_i^{\mathrm{RM}} = \tau_{\mathrm{VCG},i}$. The message transmitted by each task i contains the parameters of the piecewise linear Q-C model, i.e., $m_i^{\mathrm{task}} = [\lambda_{i,j}, \widehat{\alpha}_{i,j}]$, $\forall j \in \{1, \ldots, \Omega_i\}$. These messages are only transmitted once. Hence, the number of messages for the VCG mechanism is $(N + 2\Pi)$, where $\Pi = \sum_{i=1}^{N} \Omega_i$. Note that for the VCG mechanism, the message exchange overhead is linear with the number of tasks and the number of parameters of the Q-C model.

For the BBPM, we consider the per-iteration overhead. From Section III-C, the RM conveys to task i the average price and the excess demand of all tasks except for i, i.e., $m_i^{\mathrm{RM}} = (p_{-i}, d_i)$. Task i solves the optimization problem in (17) and transmits the resource demand and its valuation of the resource back to the RM, i.e., $m_i^{\mathrm{task}} = (\alpha_i, p_i)$. Hence, the number of messages per iteration for the BBPM is $(2 + 2) \cdot N$. Note that the message exchange overhead of the BBPM is independent of the employed Q-C model and only depends on the number of iterations to reach the equilibrium and on the number of tasks present in the system.

C. Gap to Optimality

Although the proposed greedy algorithms can be combined with piecewise linear Q-C models to reduce the resource allocation overhead, the resulting optimality gap needs to be considered. To illustrate this tradeoff, we employ a 50-segment piece-

TABLE II
COMPARISON OF THE OPTIMAL ALLOCATION DERIVED BY NONLINEAR PROGRAMMING FOR THE CASE OF CONTINUOUS Q-C MODELS AND THE SUBOPTIMAL ALLOCATION DERIVED BY THE GREEDY ALGORITHM IN FIG. 4 FOR THE CASE OF 50-SEGMENT PIECEWISE LINEAR Q-C MODELS

Task - Sequence	Optimal		Sub-optimal		Gap
	Resource	Quality	Resource	Quality	Quality
Task 1 - Foreman	0.234	43.50	0.240	43.66	+0.16
Task 2 - Mobile	0.404	34.73	0.40	34.61	-0.12
Task 3 - Coastguard	0.362	40.71	0.360	40.67	-0.04

wise linear Q-C model for each task and show the resource allocation decision in Table II. The piecewise linear Q-C models are derived from approximating the continuous Q-C model in the Appendix.

Compared with the optimal allocation derived by nonlinear programming and continuous Q-C models, the quality gap is less than 0.2 dB for each task, which is not noticeable by a human observer. We only show one allocation decision for the greedy algorithms since both algorithms produce the same allocation. This justifies the utilization of a smooth piecewise linear Q-C model to reduce the computation overhead for solving the optimization problems in (8) and (17) while producing near-optimal solutions. Hence, we only show the resource allocation results derived by employing 50-segment piecewise linear Q-C models and the greedy algorithms henceforward.

TABLE III
COMPARISON OF THE RESOURCE ALLOCATIONS AND THE RESULTING QUALITY MEASUREMENTS BETWEEN THE PROPOSED GLOBAL ADAPTATION FRAMEWORK AND THE CONVENTIONAL PS ALGORITHM

Task - Type	PS Allocation	Proposed Global Adaptation
Task 1 - Resource	0.15	0.24
Task 2 - Resource	0.60	0.40
Task 3 - Resource	0.15	0.36
Task 1 - Quality	41.25	43.66
Task 2 - Quality	40.56	34.61
Task 3 - Quality	34.39	40.67
System - Quality	116.20	118.94

V. SIMULATION RESULTS

In this section, we provide simulation results to explicitly quantify the performance of multimedia tasks under the proposed resource allocation framework. In Section V-A, we compare the resulting quality for the various resource allocation schemes and show the degradation in performance upon the arrivals of extra tasks in the system. We discuss the overhead incurred by the application of mechanism design in the global adaptation framework in Subsection V.B.

For the simulation, we utilize the H.264 coder in [9] to obtain the continuous Q-C models for Foreman, Mobile, and Coastguard sequences as described in the Appendix. Based on the continuous Q-C model, we derive piecewise linear Q-C models in order to utilize the greedy algorithms discussed in Section IV-A. Unless otherwise specified, we denote task 1, 2 and 3 to represent the decoding task of Foreman, Mobile, and Coastguard sequences henceforward.

A. Comparison of Resource Allocation Schemes

First, we compare the resource allocation decision of the global adaptation framework proposed in this paper and that of the proportional-share (PS) algorithm. We choose the PS algorithm because it is pertinent to the resource allocation problem discussed in this paper. Besides, the PS algorithm is widely adopted in various resource allocation literatures [3], [5]–[7]. For the experiment, we also utilize the EDF scheduling policy similar to [3], [6]. We assume there are three tasks (task 1–3) being serviced in the system. Each task has a minimum CPU utilization that is defined to be the CPU utilization that achieves 30 dB PSNR for a given sequence. We choose the value of 30 dB because it is recognized as a threshold for acceptable video quality [21]. For the Q-C models derived in the Appendix, task 2 requires the most resources while task 1 and 3 already satisfy the 30 dB PSNR criterion at a comparatively low CPU utilization at 0.1. Hence, the minimum utilization for the three tasks are chosen to be (0.1, 0.3, 0.1). Based on this knowledge, the weightings for the PS algorithm are determined to be (0.15, 0.7, 0.15). Table III shows the resulting resource allocations and the quality for individual tasks by applying the PS algorithm and the VCG mechanism in the proposed global adaptation framework. We do not show the result of BBPM because it produces the same allocation decision as the VCG mechanism.

By employing mechanism design, the system quality receives nearly 3 dB PSNR gain over the resource allocation by the PS algorithm. The main problem with the PS algorithm is that the resource allocation relies heavily on the weightings choice. Even if the Q-C model of each task is known, it is still possible to have a resource allocation that is far from the optimal allocation, as demonstrated by the result in Table III. Besides, PS-based algorithms do not have the ability to deal with the strategic behavior of tasks.

Next, we examine the quality impact on existing tasks when new tasks join the global adaptation framework. The setup is described as follows. We assume at time 0 the RM makes the resource allocation decisions for three tasks. The duration of each task is set to be 10 s. At the fifth second, two new tasks join the global adaptation framework. Table IV shows the initial results

TABLE IV
RESOURCE ALLOCATION AND QUALITY MEASUREMENTS WHEN THE SYSTEM INITIALLY RUNS THREE TASKS AND TWO ADDITIONAL TASKS JOIN AFTER 5 S

Task - Sequence	Time = 0 seconds				Time = 5 seconds			
	PS		Proposed		PS		Proposed	
	Resource	Quality	Resource	Quality	Resource	Quality	Resource	Quality
Task 1 - Coastguard	0.333	39.05	0.400	41.69	0.200	36.04	0.301	39.08
Task 2 - Foreman	0.333	45.16	0.300	45.15	0.200	42.61	0.133	40.77
Task 3 - Foreman	0.333	45.16	0.300	45.15	0.200	42.61	0.133	40.77
Task 4 - Coastguard	-	-	-	-	0.200	36.04	0.301	39.08
Task 5 - Foreman	-	-	-	-	0.200	36.04	0.133	40.77

TABLE V
NUMBER OF ITERATIONS TAKEN TO PRODUCE THE OUTCOME FOR THE VCG, BBPM MECHANISMS AND THE PS ALGORITHM. BBPM-NP DENOTES THE NUMBER OF ITERATIONS TAKEN BY UTILIZING NONLINEAR PROGRAMMING FOR THE CASE OF CONTINUOUS Q-C MODELS. BBPM-GREEDY REPRESENTS THE NUMBER OF ITERATIONS OBTAINED BY THE PROPOSED GREEDY ALGORITHM IN FIG. 4 UNDER THE ASSUMPTION OF PIECEWISE LINEAR Q-C MODELS

Algorithm	Number of Iterations
BBPM – NP	18
BBPM – Greedy	11
Proportional-share	1
VCG	1

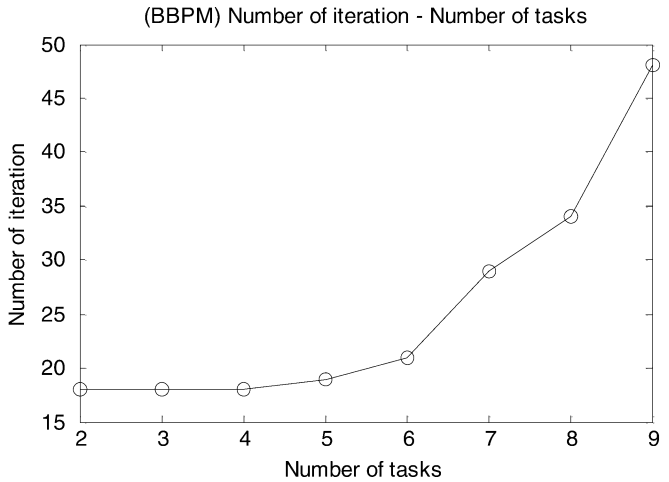

Fig. 5. Number of iterations taken to produce the outcome versus the number of tasks participating in the global adaptation.

and the updated results at time 5 s for the VCG mechanism and the PS algorithm.

The result shows that mechanism design achieves better fairness among tasks. The mechanism-design based framework assigns fewer resources to decoding tasks of the Foreman sequence (task 2, 3, 5) since the quality impact for these tasks is lesser than that for the decoding tasks of the Coastguard sequence (task 1,4). Therefore, we conclude that the mechanism-design based global adaptation framework scales with the number of tasks and achieves graceful quality degradation for the various already admitted tasks.

B. Quantification of Mechanism's Overhead

In this experiment, we assume that all tasks report true types and follow the same setup in the previous subsection to assess the overhead incurred by the application of mechanism design. Since the actual time taken to produce the outcome highly depends on the implementation of an algorithm, we measure the number of iterations taken for the computation of the outcome. The results presented in Table V shows that the BBPM takes the most iterations to reach convergence. However, the proposed greedy algorithm for BBPM can be used to reduce the overhead of the resource allocation.

We further examine the convergence rate of the BBPM against the number of tasks presented in the global adaptation framework. For the experiment, we concurrently run nine video decoding tasks of the Coastguard sequence and measure the number of iterations taken by the BBPM mechanism. We choose the Coastguard sequence because it is the least complexity-demanding task. This enables us to run nine tasks simultaneously without having any of the tasks being rejected due to the lack of resources. Fig. 5 illustrates the result. It shows that there exists a non-linear relationship between the number of iterations taken to reach the convergence and the number of tasks presented in the system.

From Subsection III.B, the per-iteration computational overhead of the VCG and the BBPM mechanisms are $\mathbf{O}(\Pi \log \Pi)$ and $\mathbf{O}(\Pi)$ respectively under the assumption of a piecewise linear Q-C model. Compared with the $\mathbf{O}(N)$ computational overhead of the PS algorithm, the two mechanisms have higher per-iteration computational overhead.

Finally, the message exchange overhead should also be considered since it could be a dominant factor in a computer network setup with transmission delays. For the PS algorithms, the RM only needs to determine the weightings of tasks based on their minimum CPU utilizations as defined in Subsection V.A. Hence, the PS algorithm has the least message exchange overhead, with the number of parameters being N. The message

exchange overhead of the VCG mechanism depends on how complex the employed Q-C model is, e.g., the number of segments of a piecewise linear Q-C model. For the piecewise-linear model employed in the paper, the message exchange overhead is $(100+1)\cdot N$. Similarly, the message exchange overhead of the BBPM, for this particular experiment, is $15\cdot 4N$.

In summary, our simulation results show that the two mechanisms considered in this paper produce efficient resource allocation decisions that maximize the system utility. We categorize the overhead introduced by the two mechanisms into computational overhead and the message exchange overhead. For the BBPM mechanism, our simulation result shows that the number of iterations taken to reach equilibrium grows significantly when there are more than six tasks in the system. The non-linear relationship between the number of tasks and the number of iterations could limit the application of the BBPM mechanism only to small systems that handle fewer tasks. In contrast, the VCG mechanism does not have such limitation. However, its message exchange overhead grows linearly with the number of parameters of the Q-C model. Therefore, the choice of the two mechanisms depends on the system specifications and requirements. Although both mechanisms have higher computational and message exchange overhead, we note that the overhead of mechanism-design based solutions is acceptable since the global adaptation is not invoked frequently.

VI. Conclusion

In this paper, we study the problem of system-level resource allocation in multitask multimedia systems, while explicitly considering the strategic behaviors of tasks. We propose a global adaptation framework based on mechanism design to deter the tasks from manipulating the resource allocation. By modeling the quality-complexity functions of multimedia tasks, the proposed framework takes into account the unique characteristics of individual tasks. We study how two popular mechanisms can be efficiently integrated in existing global adaptation frameworks for resource allocation. Compared with the PS algorithm, the simulation results show that the resulting resource allocation decision is both utility/quality maximizing and robust against task manipulation. In addition, we also investigate practical issues associated with implementing mechanism design for system resource allocation, such as the computational overhead and the message exchange overhead incurred. We conclude that the properties of efficiency, fairness and robustness against manipulation of mechanism design justify the additional overhead introduced by the adoption of mechanism design in the global adaptation framework. For the problem considered in this paper, we conclude that the VCG mechanism is a better solution due to its simplicity and faster convergence property. The framework proposed in the paper can be extended to support multiprocessor systems or to include energy constraints. These topics and the implementation of the proposed framework are part of our future research.

Appendix

For the derivation of the Q-C models, we utilize the H.264/AVC joint model reference coder in [9] throughout the paper. However, we note that the methodology can be applied to forthcoming coder standards as well to obtain a similar Q-C model.

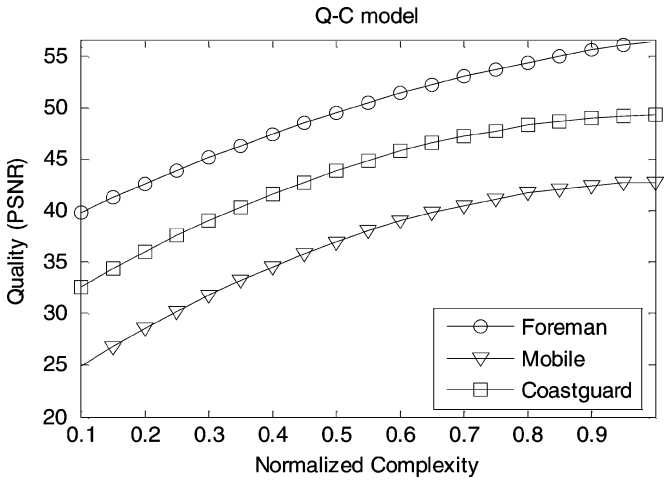

Fig. 6. Plot of the Q-C models for Foreman, Mobile, and Coastguard sequences.

TABLE VI
Set of Parameters Used for the Derivation of the Q-C Models

Bit-rate r_i	Quantization parameter:
	{27,29,31,33,35,37,39}
GOP structure g_i	Period of I-frame:{1,3,7,15,31,63} Number of B-frame:{0,1,3,5,7,9,11}

First, we represent the CPU utilization for task i by α_i, which is the normalized CPU cycles per second required to decode a particular sequence. The quality is represented by the average PSNR q_i in the unit of dB. It is well known that the CPU utilization of a video task depends on the type v_i, which consists of the characteristics of a sequence θ_i and varies according to the complexity profile $\boldsymbol{\rho}_i$. Similarly, the type also influences the PSNR quality measurement. We leverage the scalable nature of video decoding tasks to depict their Q-C functions. Specifically, each video decoding task i is decoded using a set of complexity profiles ρ_i. The resulting CPU utilization and quality measurements $(\alpha_i(\rho_i), q_i(\rho_i))$ constitutes the achievable Q-C set for a task given its set of complexity profile ρ_i. Similar to the operational rate-distortion models [11], we obtain the Q-C model by least-mean-square curve fitting the hull of the operational Q-C set.

For illustration, we applied the above methodology to obtain the Q-C models for Foreman, Mobile, and Coastguard sequences. The complexity profile consists of the bit-rat r_i and the group-of-pictures (GOP) structure g_i, i.e., $\rho_i(v_i) = (r_i, g_i)$. Table VI shows the collection of parameters of the complexity profile. The sequences are decoded at CIF resolution and 30 Hz frame-rate with 300 frames decoded. Fig. 6 plots the derived Q-C models based on the parameters listed in Table VI.

From Fig. 6, the derived Q-C models are concave. They are also monotonically increasing. We assume that every point on

the Q-C model is achievable. This assumption is reasonable because for H.264/AVC-based coders, the techniques of data partitioning and hierarchical B frames can be used to create various layers.

ACKNOWLEDGMENT

The authors would like to thank the associate editor and all the reviewers for the comments, which significantly helped us improve our manuscript.

REFERENCES

[1] A. Mas-Colell, M. Whinston, and J. Green, *Microeconomic Theory*. New York: Oxford Univ. Press, 1995.

[2] T. Stoenescu and J. Ledyard, "A pricing mechanism which implements in nash equilibria a rate allocation problem in networks," in *45th IEEE Conf. on Dec. and Control*, Dec. 2006, pp. 1270–1277, "A Pricing Mechanism which Implements a Network Rate Allocation Problem in Nash Equilibria," Submitted to the IEEE/ACM Transaction on Networking.

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

[4] C. L. Liu and J. W. Layland, "Scheduling algorithm for multiprogramming in a hard real-time environment," in *IEEE Real-Time System Symp.*, 1987.

[5] C. A. Waldspurger and W. E. Weihl, "Lottery scheduling: Flexible proportional-share resource management," in *Proc. First Symp. Operating Systems Design and Implementation*, Nov. 1994, pp. 1–11.

[6] J. Nieh and M. S. Lam, "The design, implementation and evaluation of smart: A scheduler for multimedia applications," in *Proc. 16th ACM Symp. on Operating Systems Principles*, Oct. 1997, pp. 184–197.

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

[8] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek, "A resource allocation model for QoS management," in *Proc. 18th IEEE Real-Time Systems Symp.*, Dec. 1997, pp. 298–307.

[9] H.264/AVC Joint Model Reference [Online]. Available: http://iphome.hhi.de/suehring/tml

[10] K. Nahrstedt, H. Chu, and S. Narayan, "QoS-aware resource management for distributed multimedia applications," *J. High-Speed Networking*, vol. 8, no. 3–4, pp. 227–255, 1998, IOS Press.

[11] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 23–50, Nov. 1998.

[12] A. Silberschatz, P. B. Galvin, and G. Gagne, *Operating System Concepts*, 7th ed. Hoboken, NJ: Wiley, 2004.

[13] J. R. Ohm, "Advances in scalable video coding," *Proc. IEEE*, vol. 93, no. 1, pp. 42–56, 2005.

[14] 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, July 2003.

[15] S. Browne, J. Dongarra, N. Garner, K. London, and P. Mucci, "A scalable cross-platform infrastructure for application performance tuning using hardware counters," in *Proc. Supercomputing 2000*, Nov. 2000, p. 42.

[16] T. Plagemann, V. Goebel, P. Halvorsen, and O. Anshus, "Operating system support for multimedia systems," *Comput. Commun. J.*, vol. 23, no. 3, pp. 267–289, Feb. 2000.

[17] M. Jackson, *Mechanism Theory*. Oxford, U.K.: Optimization and Operations Research in the Encyclopedia of Life Support Systems, 2003.

[18] N. Nisan, "Algorithms for selfish agents," in *Proc. 16th Symp. on Theoretical Aspects of Computer Science*, Mar. 1999.

[19] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: Tools, performance, and complexity," *IEEE Circuits Syst. Mag.*, vol. 4, no. 1, pp. 7–28, 2004.

[20] H. M. Radha, M. van der Schaar, and Y. Chen, "MPEG-4 fine-grain scalable video coding method for multimedia streaming over IP," *IEEE Trans. Multimedia*, vol. 3, no. 1, pp. 53–68, Mar. 2001.

[21] J. Pons, M. P. Malumbres, and R. Garcia, "Scaled-MJPEG: A symmetric video compression method for low bit rates," *IASTED Applied Informatics 2000*, pp. 302–308, 2000.

[22] R. H. Kwong and E. W. Johnston, "A variable step size LMS algorithm," *IEEE Trans. Signal Process.*, vol. 40, no. 7, pp. 1633–1642, July 1992.

Shih-Chung Su received the B.S degree in electrical engineering from National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., in 2003, and the M.S. degree in electrical engineering from University of California, Los Angeles, in 2007.

He is currently a Member of the Technical Staff at Broadlogic, Inc., San Jose, CA. His research interests are multimedia communication and multimedia systems.

Mihaela van der Schaar (SM'04) is currently an Associate Professor in the Electrical Engineering Department at University of California, Los Angeles. She has 30 granted U.S. patents. Her research interests are in multimedia communications, networking, processing and systems.

Dr. van der Schaar received the NSF Career Award in 2004, the Best Paper Award from IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (T-CSVT) in 2005, the Okawa Foundation Award in 2006, the IBM Faculty Award in 2005, 2007 and 2008, and the Most Cited Paper Award from *EURASIP: Image Communications* journal in 2006. She was an associate editor for IEEE TRANSACTIONS ON MULTIMEDIA, IEEE SIGNAL PROCESSING LETTERS, IEEE T-CSVT, IEEE *Signal Processing Magazine*, etc. She holds three ISO awards. Her research interests are in multimedia communications, networking, processing and systems.