Delay-Power-Rate-Distortion Optimization of Video Representations for Dynamic Adaptive Streaming

Chenglin Li, *Member, IEEE,* Laura Toni, *Member, IEEE,* Junni Zou, *Member, IEEE,* Hongkai Xiong, *Senior Member, IEEE,* and Pascal Frossard, *Senior Member, IEEE*

***Abstract*—Dynamic adaptive streaming addresses user heterogeneity by providing multiple encoded representations at different rates and/or resolutions for the same video content. For delay sensitive applications, such as live streaming, there is however a stringent requirement on the encoding delay, and usually the encoding power (or rate) budget is also limited by the computational (or storage) capacity of the server. It is therefore important, yet challenging, to optimally select the source coding parameters for each encoded representation in order to minimize the resource consumption while maintaining a high quality of experience for the users. To address this, we propose an optimization framework with an optimal representation selection problem for delay, power, and rate constrained adaptive video streaming. Then, by the optimal selection of source coding parameters for each selected representation, we maximize the overall expected user satisfaction, subject not only to the encoding rate constraint, but also to the delay and power constraints at the server. We formulate the proposed optimization problem as an integer linear program (ILP) formulation to provide the performance upper bound, and as a submodular maximization problem with two knapsack constraints to develop a practically feasible algorithm. Simulation results show that the proposed weighted rate and power cost benefit greedy algorithm is able to achieve a near-optimal performance with very low time complexity. In addition, it can strike the best tradeoff both between the rate and power cost, and between the algorithm's performance and the delay requirements proposed by delay sensitive applications.**

***Index Terms*—Dynamic adaptive video streaming, representation selection, delay-power-rate-distortion, live video, submodular function maximization.**

I. Introduction

With the rapid development and ever-increasing popularity of mobile devices, users are now capable of requesting and playing video content anywhere and at any time. Accordingly, the management of video streaming services has recently become a much more complex task due to the growing heterogeneity of user population in terms of demands for specialized video contents, devices used to display, and access network capacity. Dynamic adaptive streaming over HTTP (DASH) has been proposed as an effective solution to address heterogeneity and improve the overall user satisfaction by offering several representations (versions) of the same video content to the different clients [1]. As illustrated in Fig. 1, each representation is encoded with a pre-defined bitrate and/or resolution by the DASH server. The users will then select the representation that better addresses their requirements and the network conditions. Upon request, streams containing the desired representations based on the client-side rate adaptation algorithms are then delivered to the users over certain network architectures, such as the content delivery network (CDN).

While most of the research community focuses on the client-side rate adaptation schemes for smoothly downloading pre-encoded representations, little work has been done to address the representation selection problem at the server with considerations of the video encoding delay or power consumption. This representation selection problem becomes more crucial for delay sensitive applications, e.g., live video streaming. In live streaming [2], for example, there is a stringent requirement on the encoding delay of all the representations, which requires the frame encoding time to be less than or equal to the frame interval. In addition, the video encoding process is generally quite demanding in terms of the computational complexity, which is related to both the encoding delay and the power consumption [3]. Although the server is usually assumed to be very powerful, there still exists a physical limit in reality. For example, as the total number of encoders available in the DASH server is constrained and the maximum clock frequency of the CPUs within each encoder is limited, the encoding process for all the representations of all the video streams should be limited by a maximum power budget (i.e., the total CPU capacities at the DASH server) [2]. As a result, the power limitations of the DASH server are definitely a critical issue in live streaming applications.

Previous server-side representation selection schemes, such as [4], have demonstrated the gain of the rate-distortion optimization in the representation selection for different video types. Due to the diverse content characteristics, it is beneficial to tune the source coding parameters to both the types of videos and the users' conditions. These works are rate-distortion efficient, capable of achieving the best overall video quality with the minimum cost of total encoding bitrate. However, they neglect the cost of encoding delay and power consumption, which nevertheless becomes a key component in delay sensitive applications. From the perspective of the source coding, the impact of delay and power consumption constraints on the rate-distortion behavior is as follows. Ideally, an efficient video compression is preferred to greatly reduce the encoding bitrate while maintaining the same video quality. However, the efficient video compression often requires high computational complexity at the video encoder, which in turn results in long delay and large power consumption of the encoder. Such schemes however spend a large amount of encoding time or power consumption to achieve only a slightly better improvement in the rate-distortion performance of each encoded representation, which might furthermore lead to an outage of the streaming service due to unacceptable latency or the violation of the total power budget.

With the delay requirement of the live video applications and the limited power and rate resources, the DASH server cannot encode as many representations as possible to individually respond to each user's request. Instead, the system resources should be judiciously distributed between the different videos in order to maximize the overall system performance. It is therefore worth investigating an effective selection of the optimal representations encoded for each video with the corresponding encoder parameters, in order to better support the users' requirements and yet to be sustainable with the delay sensitive applications.

We therefore propose in this paper to develop a server-side optimization framework for the adaptive video representation selection in

C. Li and P. Frossard are with the Signal Processing Laboratory (LTS4), École Polytechnique Fédérale de Lausanne (EPFL), Lausanne 1015, Switzerland (e-mail: chenglin.li@epfl.ch, pascal.frossard@epfl.ch).

L. Toni is with the Electrical and Electronic Department, University College London (UCL), London WC1E 7JE, U.K. (e-mail: l.toni@ucl.ac.uk).

J. Zou is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China (e-mail: zou-jn@cs.sjtu.edu.cn).

H. Xiong is with the Department of Electronic Engineering, Shanghai Jiao Tong University, China (e-mail: xionghongkai@sjtu.edu.cn).

delay sensitive streaming with limited resources in terms of storage (or bottleneck link capacity) and power consumption. Specifically, we formulate a representation selection optimization problem for delay sensitive DASH streaming with proper consideration of the delay-power-rate-distortion (d-P-R-D) properties of representations from different videos, under the encoding delay, power and rate constraints. This representation selection problem is then re-formulated as an integer linear program (ILP). The proposed ILP could lead to the optimal tradeoff between the delay-power-rate-distortion resource constraints and thus provide a performance upper bound for the server-side representation selection. However, it is NP-hard and thus too time consuming to be a practical solution for delay-sensitive streaming. In order to greatly reduce the execution time, we further convert the original optimization problem to an equivalent set function optimization problem, which is shown to be a submodular maximization problem subject to two knapsack constraints. A weighted rate and power cost benefit greedy algorithm is developed in order to obtain a practical yet approximate solution with low computational complexity and near-optimal performance. Overall, the contribution of this paper can be summarized as follows.

1) We formulate a novel representation selection optimization framework to find the best set of encoded representations that maximizes the expected video distortion reduction for users under encoding delay, power and rate constraints. We further propose an ILP formulation to provide the performance upper bound for the system design of the server-side DASH representation selection.
2) In order to reduce the additional execution time of the representation selection algorithm in practice, we convert the original optimization problem to an equivalent set function optimization problem and show its submodularity. By using the diminishing return property of submodular functions, we develop a weighted cost benefit greedy algorithm for the representation selection, which has polynomial computational complexity and offers close-to-optimal performance (approximation ratio shown to be above 90% under different simulation settings in Section VI).
3) We conduct extensive simulations under different system settings. The simulation results show that the proposed algorithm can scale very well with the size of the system. It strikes the best tradeoff both between the rate and power cost, and between the algorithm's performance in terms of the average distortion reduction per user and the delay aspects, such as the algorithm computation time and the per-frame encoding time requirements in delay sensitive applications.

The rest of this paper is organized as follows. Section II reviews the related works in literature. In Section III, we introduce the notations and the d-P-R-D models that are used throughout this paper. In Section IV, we propose an optimization framework and formulate a general optimization problem for the representation selection in DASH encoding subject to encoding delay, power and rate constraints. To obtain the practical algorithms with low time complexity, in Section V, we transform the general representation selection optimization problem to an equivalent set function optimization problem, which is further proved to be a submodular maximization problem over two knapsack constraints. We describe a practical approximation algorithm to solve this problem with close to optimal performance. Section VI presents the experimental results, and evaluates the gains of the proposed algorithm compared to existing algorithms. The concluding remarks are given in Section VII.

II. Related Works

Different works have been proposed recently to optimize the multiple representation selection for dynamic adaptive streaming [2], [4]–[10]. Most of these research efforts focus on the client-side adaptation algorithms in order to guarantee the quality of experience (QoE) of users for given encoded representations at the server, such as the live streaming rate adaptation method to support a smooth presentation while maintaining a small buffer size [5], the application layer probe-and-adapt rate adaptation approach driven by an estimate of the network dynamics [6], and the online rate adaptation algorithm in order to minimize the re-buffering phases [7]. Although Thang *et al.* [5] highlight the importance of the server-side representation set optimization and show that the preparation of representation sets may affect the behaviors of some client-side adaptation methods, they do not propose any optimization based guideline on such representation selection.

The server-side representation optimization has been investigated very recently in [9], where a joint transcoding and caching allocation scheme in media cloud is proposed to minimize the total operational cost of delivering on-demand adaptive video streaming. In [4], the optimal representation set selection problem of adaptive streaming under the encoding rate constraint of the DASH server is proposed as an integer linear program (ILP), revealing the best coding parameter in terms of the bitrate and resolution for each representation. In [10], the optimized representations obtained by solving this ILP are further investigated and validated in a practical scenario, by generating a 24-hour streaming scenario based on YouTube traces and device statistics for Hulu and Netflix. These two works are rate-distortion efficient, capable of achieving the best overall video quality with the minimum cost of total encoding bitrate. However, they neglect the cost of encoding delay and power consumption, which nevertheless becomes a key component in delay sensitive applications. For live video streams, the authors in [2] propose another ILP formulation by considering the computation resource constraint. The ILP model in [2] is based on the dataset obtained by extensive transcoding operations of the target videos, which means that the finite ground set of the available representations is pre-encoded with known video qualities, bitrates and resolutions. However, this assumption is not feasible in practical live streaming applications where there is no pre-encoded representation set. Instead, we have to address a rate control problem, which determines on the fly the source coding parameters (e.g., the search range, the quantization step size) to achieve the desired bitrate of each target representation. Another limitation of the above works is that these ILP problems are NP-hard. In practice, even with the latest optimization tools such as the IBM ILOG CPLEX [11], they require exponential computational complexity to achieve optimal solutions. Therefore, a very long execution time will be consumed for larger system settings, which introduces an intolerant initial delay and greatly degrades the QoE of users. In dynamic setups, worse yet, the computation and storage resources are usually time-varying, which requires the system to dynamically scale its capacity to reduce the resource consumption while still respecting the encoding delay requirement imposed by live streaming. To this end, the works in [12]–[14] discuss and investigate the dynamic resource provisioning problem for encoding online videos.

Rate control schemes, on the other hand, aim at providing a good quality for the encoded video under a given rate constraint, by appropriate selections of the source coding parameters. To this end, many works have been conducted to analyze the complexity, rate and distortion performance of the hybrid video encoders [15]–[19]. In the rate-distortion model of [15], both the source coding rate and distortion of a hybrid video coder with block based coding are revealed to be closely related to the video statistics and the quantization step size, and derived as functions of the standard deviation of the transformed residuals under the assumption that these transformed residuals follow a Laplacian distribution. He *et al.* [16]

summarize the encoding complexity of the H.263 video encoder as three modules (motion estimation, precoding and entropy coding), and derive a power-rate-distortion model to analyze the relationship among these three factors. For the more advanced H.264/AVC video encoders that use the tree-structured motion compensation with seven inter-modes, the work in [17] proposes a delay-rate-distortion model for both IPPPP and hierarchical-B coding modes. In [18], the analytical framework for delay-power-rate-distortion modeling of the hybrid video encoder is proposed and derived as a function of source coding parameters (specifically, the search range in motion estimation and the quantization step size). On the basis of the proposed analytic model, a source rate control scheme is further formulated to achieve the minimum encoding distortion for single video representation under the constraints of maximum encoding delay, rate, and power consumption. This model is also applied in the end-to-end wireless video communication system to develop an optimization based rate control scheme that aims at minimizing the end-to-end distortion (including both video encoding distortion and the transmission distortion) subject to the transmission rate and delay constraints [19]. For the single-source, multiple destination video communication over the lossy Internet, a forward error correction packet allocation and scheduling framework is proposed in [20] to trade the transmission delay for the video distortion.

It should be noted that, however, all the aforementioned rate control schemes (e.g., [18], [19]) are dedicated to the single video case, where we only need to determine one pair of the optimal source coding parameters for one encoded representation subject to the resource constraint at the encoder. In other words, all of the encoder's resources, including rate and power, are solely used for encoding one single representation. There is no encoding process of another representation from the same or a different video, which will compete for such limited resources at the encoder. Due to the failure in coping with the fairness and resource competition issue among the multiple representations, they cannot be straightforwardly extended to the DASH scenario with multiple coexisting videos, each of which is further encoded into multiple representations. In fact, the multiple related representations from the same or different videos will compete for the shared rate and complexity resources at the DASH server. However, it is still unclear how to optimally allocate the rate and power resources among different videos, and how to choose the optimal source coding parameters for each specific representation.

In summary, the previous works are limited for delay sensitive DASH streaming since they are either time consuming or not optimized over the rate/power resource allocations. Therefore, we propose an optimization framework for DASH representation selection with limited delay, power and rate resources, and develop accordingly an efficient algorithm that is able to achieve near-optimal performance with very low computational complexity. In general, the differences and novelty of this work can be summarized as: 1) joint consideration of the delay, power and rate constraints at the server; 2) a representation selection problem integrated with the rate control scheme; and 3) a practically efficient approximation algorithm with low computational complexity and theoretical approximation guarantee.

III. DELAY-POWER-RATE-DISTORTION MODEL FOR VIDEO ENCODING

In this section, we introduce the notations and the delay-power-rate-distortion model for general video encoders, which will later be used for characterizing the corresponding behavior of each single encoded representation.

In [18], [21], the models of source coding delay, power, rate and distortion have been derived for IPPPP coding mode in H.264/AVC.

TABLE I
MAIN NOTATIONS.

Symbol	**Definition**
$\mathcal{F} = \{1, 2, \ldots, F\}$	The set of F video streams.
$\mathcal{M} = \{1, 2, \ldots, M\}$	The set of M representations for each video stream.
$\mathcal{N} = \{1, 2, \ldots, N\}$	The set of N users.
σ	The standard deviation of the transformed residuals in motion estimation.
$\lambda_{f,m}$	The search range in motion estimation of the m-th representation of video $f \in \mathcal{F}$.
$Q_{f,m}$	The quantization step size of the m-th representation of video $f \in \mathcal{F}$.
Λ	The search rage set containing all the possible search range values.
$\mathcal{Q}$	The quantization step size set including all the available quantization step sizes.
$D_f(\lambda_{f,m}, Q_{f,m})$	The source coding distortion of the m-th representation of video $f \in \mathcal{F}$.
$R_f(\lambda_{f,m}, Q_{f,m})$	The source coding rate of the m-th representation of video $f \in \mathcal{F}$.
$C_f(\lambda_{f,m}, Q_{f,m})$	The CPU load in clock frequency for encoding the m-th representation of video $f \in \mathcal{F}$.
$P_f(\lambda_{f,m}, Q_{f,m})$	The CPU power consumption for encoding the m-th representation of video $f \in \mathcal{F}$.
d_f	The time (delay) needed to encode one frame of video $f \in \mathcal{F}$.
$R_{\max}$	The maximum total encoding rate constrained by the storage capacity of the server or the bottleneck link's transmission rate of the network.
$C_{\max}$	The maximum CPU load of the server.
ΔT	The desired time interval for encoding one video frame.
B_n	The downlink bandwidth of user $n \in \mathcal{N}$.
$\mathcal{E} = \{e_{f,m} \vert \forall f, m\}$	The finite ground set of representations, where $e_{f,m}$ denotes the encoding of the m-th representation of video f.
$\mathcal{A}$	The encoding decision set $\mathcal{A} \subseteq \mathcal{E}$ with each element $e_{f,m} \in \mathcal{A}$ indicating the actual encoding of the m-th representation of video f.
ρ_f^n	The probability of user n requesting video file f.
$\bar{D}_n(\mathcal{A})$	The expected average video distortion reduction for user n based on the encoding decision set $\mathcal{A}$.

Under the assumption that the transformed residuals in the motion estimation (ME) module follow an i.i.d. zero-mean Laplacian distribution [15], [22], both the source rate and distortion of an inter-coded P-frame are derived as functions of the standard deviation σ of the transformed residuals and the quantization step size Q. Specifically, for a video stream $f \in \mathcal{F}$, the source rate is approximated by the entropy of the quantized transformed residuals, and the source distortion is only incurred by the quantization error, as follows:

$$R_f(L, Q) = -P_0 \log_2 P_0 + (1 - P_0) \Bigg[\frac{LQ \log_2 e}{1 - e^{-LQ}} - \log_2(1 - e^{-LQ}) - LQ\gamma \log_2 e + 1 \Bigg], \tag{1}$$

$$D_f(L, Q) = \frac{LQe^{\gamma LQ}(2 + LQ - 2\gamma LQ) + 2 - 2e^{LQ}}{L^2(1 - e^{LQ})}, \tag{2}$$

where $L = \sqrt{2}/\sigma$ is the Laplace parameter that is one-to-one mapping of σ; γQ represents the rounding offset and γ is a parameter between $(0, 1)$, such as $1/6$ for H.264/AVC inter- frame coding [15]; $P_0 = 1 - e^{-LQ(1-\gamma)}$ is the probability of quantized transform coefficient being zero. For a specific video $f \in \mathcal{F}$, the standard deviation σ can be well fitted by a closed form function of the search range λ in motion estimation and the quantization step size Q [18], as:

$$\sigma_f(\lambda, Q) = a_{f,1} \cdot e^{-a_{f,2} \cdot \lambda} + a_{f,3} + a_{f,4} \cdot Q, \tag{3}$$

where $a_{f,1}$-$a_{f,4}$ are empirical parameters dependent on the encoded video sequence f as well as on the encoding structure. As shown in [19], in order to have a better fitting result, the whole set of the empirical values with different configurations of λ and Q should be used to determine these four parameters. To reduce the complexity in practice, since the function form of $\sigma_f(\lambda, Q)$ is already known and only four fitting parameters are unknown, we could choose a much smaller subset of empirical values with only a few configurations of λ and Q as the training set and obtain the standard deviation model in Eq. (3). Then, integrating $L = \sqrt{2}/\sigma_f(\lambda, Q)$ into Eqs. (1) and (2), both the source coding rate and distortion of video f can be further expressed as functions of λ and Q, i.e., $R_f(\lambda, Q)$ and $D_f(\lambda, Q)$, respectively.

On the other hand, since motion estimation (ME) takes up the majority of the total encoding time, the encoding complexity can be approximated by the ME complexity. Specifically, the ME complexity is derived as the total number of CPU clock cycles consumed by its SAD (sum of absolute difference) operations in ME. Thus, for the single-reference prediction case where only one reference frame is used for motion estimation of the current frame, the CPU load in clock frequency for encoding a specific video $f \in \mathcal{F}$ can also be expressed as a function of λ and Q, as follows:

$$C_f(\lambda, Q) = \frac{K(2\lambda + 1)^2 \cdot \eta_f(Q) \cdot c_0}{d_f}, \tag{4}$$

where K is the total number of Macroblocks (MBs) in a frame; $(2\lambda + 1)^2 \cdot \eta_f(Q)$ is the total number of SAD operations in the two dimensional search area for each MB, $(2\lambda+1)^2$ is the theoretical total number of SAD operations in the search, and $\eta_f(Q)$ is an empirical and video content dependant parameter that denotes the ratio of the actual number of SAD operations in the practical video codec to the theoretical total number of SAD operations; c_0 is the number of clock cycles needed for one SAD operation over a given CPU; d_f denotes the desired encoding delay of video f, i.e., the time required to encode one video frame.

In essence, it is the encoding complexity that depends both on the video file and the target representation. Specifically, in the encoding complexity model in Eq. (4), the complexity to encode one video frame is expressed as the total number of the CPU clock cycles $K(2\lambda+1)^2 \cdot \eta_f(Q) \cdot c_0$, which depends on the video f and the source coding parameter pair (λ, Q) of the target representation. On the other hand, the encoding complexity can be also viewed as the product of the encoding time (delay) and the CPU load in clock frequency. Therefore, according to different application scenarios, we can either fix the CPU load in clock frequency at a constant value C_{CLK} and set the encoding delay as a tunable parameter $d_f(\lambda, Q)$, e.g., for the single video encoder with given CPU as in Ref. [18]. Or, we can fix the encoding delay at a desired value d_f and allocate the total CPU load of the server $C_{\max}$ among different target representations $C_f(\lambda, Q)$, which is the case in this work.

By using the dynamic voltage scaling model to control the power consumption of the microprocessor [23], [24], the CPU load in clock frequency $C_f(\lambda, Q)$ can be further related to the CPU power consumption:

$$P_f(\lambda, Q) = \kappa \cdot [C_f(\lambda, Q)]^3, \tag{5}$$

where κ is a constant in the dynamic voltage scaling model and determined by both the supply voltage and the effective switched capacitance of the circuits [25]. It can been seen from Eq. (5) that, for a given dynamic voltage scaling model with known constant κ, there exists a one-to-one mapping between the CPU clock frequency load $C_f(\lambda, Q)$ and the CPU power consumption $P_f(\lambda, Q)$. Therefore, throughout this paper, these two terms will be interchangeably used to represent the power consumption level of encoding video f with source coding parameter pair (λ, Q).

IV. Framework and Optimization Problem Formulation

In this section, we propose an optimization framework and formulate a general optimization problem for representation selection, subject to encoding delay, power and rate constraints. We then formulate the optimization problem as an integer linear program, which is generally NP-hard.

A. Framework

As illustrated in Fig. 1, we assume that F live video streams, denoted as the set $\mathcal{F} = \{1, 2, \ldots, F\}$, have to be processed by the DASH system. Any video $f \in \mathcal{F}$ can be encoded into at most M representations by the multiple parallel encoders at the DASH server. After encoding, all the encoded representations are made available at the HTTP server for adaptive streaming. Through the CDN, N users subscribe to the video service and watch desired video contents with diverse network and user behaviors. By extracting the first several frames whenever a scene change occurs [18], the delay-power-rate-distortion model of Eqs. (1)-(5) can be explicitly derived for each video stream. The practical derivation process of the d-P-R-D model is as follows. According to [18], the source rate model in Eq. (1), the source distortion model in Eq. (2), the encoding complexity model (revealing the relationship between the CPU load and the encoding time) in Eq. (4), and the encoding power model in Eq. (5) are all general models independent of the video content, while only the standard deviation model $\sigma_f(\lambda, Q)$ in Eq. (3) and the parameter $\eta_f(Q)$ in Eq. (4) are specific to the video content. Therefore, for each video stream, we can extract the first several frames whenever a scene change occurs in order to determine the video content dependent models $\sigma_f(\lambda, Q)$ and $\eta_f(Q)$. Once these two video content dependent models are known, the d-P-R-D model in Eqs. (1)-(5) is also derived.

These d-P-R-D models of different live video streams will then be used by the representation selection module to guide the encoding process in the parallel encoders, through providing the desired bitrate of each representation for each video by setting the optimal encoder parameters. Here, the representation selection module not only addresses the general problem of the number of representations needed to be encoded for each video and their average encoding rate, but also specifies explicitly by using what encoder parameters each individual encoder could achieve the desired rate for the selected representations.

In practice, there are several stringent requirements that constrain the representations encoded at the DASH server. For example, in order to enable delay sensitive streaming without incurring additional delay accumulated over frames, there is a stringent upper limit for the frame encoding time. In addition, the sum of bitrates of all encoded representations may be constrained by the server's storage capacity or the bottleneck link of the network, while the total encoding power consumption is also limited by the total number of encoders and the maximum CPU load of each encoder. Therefore, the proposed representation selection module needs to be carefully optimized, which will be described in detail in the following.

B. Problem Formulation

In accordance with the d-P-R-D models of Section III, we denote by $\mathcal{M} = \Lambda \times \mathcal{Q}$ the set of $M = |\mathcal{M}|$ possible representations. Each

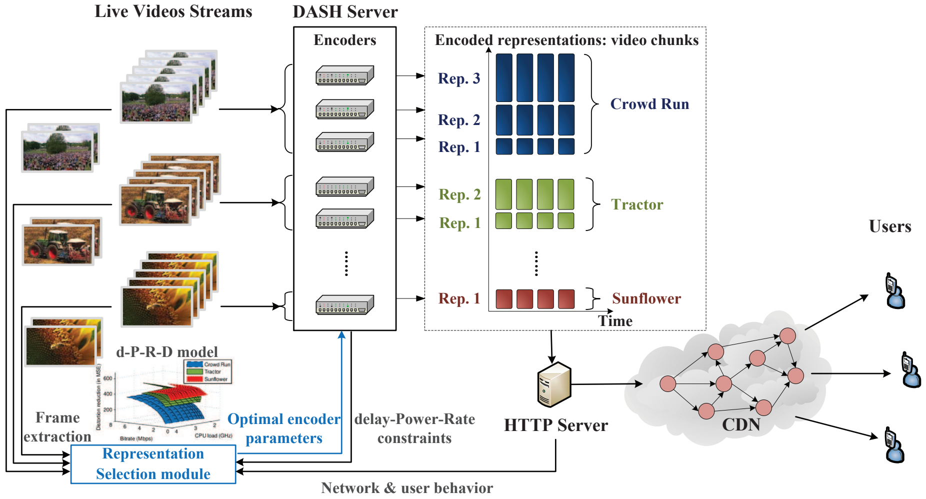


Fig. 1. Example of the delay sensitive dynamic adaptive streaming system, and framework of the proposed optimal representation selection scheme.

element in $\mathcal{M}$ corresponds to a specific source coding parameter pair (λ, Q) with $\lambda \in \Lambda$ and $Q \in \mathcal{Q}$, where Λ is the search range set containing all the possible search range values (e.g., if the maximum search range is 16, then $\Lambda = \{1, 2, \ldots, 16\}$) and $\mathcal{Q}$ denotes the quantization step size set including all the available quantization step sizes (e.g., all the quantization step sizes corresponding to QP from 0 to 51 in H.264/AVC video encoder). Without loss of generality, we sort the representation set $\mathcal{M}$ in an decreasing order of the encoding bitrate, i.e., $R_f(\lambda_{f,i}, Q_{f,i}) > R_f(\lambda_{f,j}, Q_{f,j}), \forall i, j \in \mathcal{M}$ and $1 \leq i < j \leq M$.

The optimal representation selection problem for resource constrained DASH streaming can be summarized as follows. For a given set of source video streams, with a given video popularity distribution, and for given users' downlink bandwidth, the problem consists of deciding the encoded representations for each video (i.e., the number of representations and the average bitrate of each representation) and the corresponding source coding parameters for each representation such that the total system utility in terms of the aggregate users' satisfaction is maximized, subject to the encoding delay, bitrate and power constraints at the DASH server. Mathematically, such a problem can be formulated as:

$$\textbf{P1:}\ \arg \max_{\{(\boldsymbol{\lambda},\boldsymbol{Q})\},M_f} \sum_{f=1}^{F} \sum_{m=1}^{M_f} N \cdot \phi_{f,m} \cdot U_f(\lambda_{f,m}, Q_{f,m}), \tag{6a}$$

s.t.

$$\sum_{f=1}^{F} \sum_{m=1}^{M_f} R_f(\lambda_{f,m}, Q_{f,m}) \leq R_{\max}, \tag{6b}$$

$$\sum_{f=1}^{F} \sum_{m=1}^{M_f} C_f(\lambda_{f,m}, Q_{f,m}) \leq C_{\max}, \tag{6c}$$

$$d_f \leq \Delta T,\ \forall f \in \mathcal{F}, \tag{6d}$$

$$(\lambda_{f,m}, Q_{f,m}) \in \mathcal{M},\ \forall f \in \mathcal{F}, \forall m = \{1, 2, \ldots, M_f\}, \tag{6e}$$

$$M_f \leq M,\ \forall f \in \mathcal{F}. \tag{6f}$$

In the optimization problem **P1**, the objective in Eq. (6a) is to maximize the aggregate expected utility function for all users, where $U_f(\lambda_{f,m}, Q_{f,m})$ represents the utility function after encoding the representation of video f with source coding parameter pair $(\lambda_{f,m}, Q_{f,m})$, N denotes the total number of users, $\phi_{f,m}$ is the probability of users watching the m-th representation of video stream f and thus $N \cdot \phi_{f,m}$ represents accordingly the number of users. The decision variables are the source coding parameter pair $(\lambda_{f,m}, Q_{f,m})$ for the m-th representation of video stream f, and M_f that corresponds to the number of actually encoded representations for video stream f. The constraint (6b) specifies that the sum of bitrates of all representations does not exceed the maximum transmission rate constrained by either the storage capacity of the server or the bottleneck link of the network. The constraint (6c) is the power consumption constraint ensuring that the overall CPU load in clock frequency consumed to encode all representations is limited by the server's maximum CPU load C_{max}. The constraint (6d) is the encoding delay requirement that states that the encoding time for one video frame should not exceed the desired time interval. For example, when ΔT is set to the frame interval (i.e., the reciprocal of the frame rate), it becomes the live video encoding constraint. The constraints (6e) and (6f) define the feasible region of the decision variables, respectively, specifying that the feasible source coding parameter pair $(\lambda_{f,m}, Q_{f,m})$ should be an element of the possible representation set $\mathcal{M} = \Lambda \times \mathcal{Q}$, and the number of video f's representations should not exceed the total number of possible representations M.

In this paper, we mainly focus on the server-side representation selection for live adaptive video streams. Therefore, the corresponding optimal representation selection problem **P1** in Eq. (6) is mainly constrained by the limited rate and power resources at the server side. For example, the constraint in Eq. (6b) specifies the maximum value of the sum of encoding bitrates of all target representations, $R_{\max}$. The physical meaning of $R_{\max}$ could be either the storage capacity of the server's buffer where the violation of constraint (6b) would cause some representations to overflow and thus to be unavailable for transmission to the users, or the bottleneck link capacity of the network that specifies the maximum information flow allowed to be transmitted from the server to the users. The network traffic incurred by video streaming would determine which representation of a video is downloaded and watched by users upon their requests for that video. This factor is thus considered in the objective function in Eq. (6a) and reflected by the probability $N \cdot \phi_{f,m}$. Here, N denotes the total number of users, $\phi_{f,m}$ is the probability of users watching the m-th representation of video file f and thus $N \cdot \phi_{f,m}$ represents accordingly the number of users. When the network traffic is limited, users usually tend to reduce the requested bitrate in order to cope with the congestion, which causes the increment of $N \cdot \phi_{f,m}$ for larger values of m and vice versa.

In the formulation of the optimization problem **P1**, the delay and power constraints cannot be introduced as a straightforward extension of the traditional rate-distortion optimized representation selection problem [4], since for a given encoding delay, different quantities (the utility function related to distortion, the rate and the power) are coupled through the choices of the source coding parameter pair (λ, Q). Therefore, it is nontrivial to investigate the selection of the optimal representations encoded for each video with the corresponding encoder parameters, under the delay, rate and power constraints. However, it can be seen that for a given probability distribution $\phi_{f,m}$, the optimal number of representations for each video with the corresponding source coding parameter pairs can be obtained by solving **P1**. On the other hand, since a user will only choose to watch a video representation with lower encoding bitrate than its download link's bandwidth, the probability distribution $\phi_{f,m}$ is highly dependent on the source coding parameter pairs and is thus unknown unless the source coding parameter sets are determined. Therefore, the practical algorithm is hindered by this chicken and egg dilemma in problem **P1**. To address this issue, in the next subsection, we will re-formulate problem **P1** as an integer linear program based on certain prior information about the users.

C. Integer Linear Programming Approach

We first denote $\mathcal{N} = \{1, 2, \ldots, N\}$ as the set of N users. Each user $n \in \mathcal{N}$ requests a video f with probability ρ_f^n and downloads a representation of the requested video from the server with downlink bandwidth B_n, which therefore specifies the largest bitrate of a representation that could be downloaded by user n. In the following, we introduce two sets of binary decision variables:

$$\alpha_{f,m}^n = \begin{cases} 1, & \text{if user } n \text{ selects the } m\text{-th} \\ & \text{representation for video } f; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

$$\beta_{f,m} = \begin{cases} 1, & \text{if the server encodes the } m\text{-th} \\ & \text{representation of video } f; \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

Therefore, we have $N \cdot \phi_{f,m} = \sum_{n=1}^{N} \rho_f^n \cdot \alpha_{f,m}^n$, and problem **P1** can be equivalently converted to the following ILP:

$$\textbf{P2:}\ \arg\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \sum_{f=1}^{F}\sum_{m=1}^{M}\sum_{n=1}^{N} \rho_f^n \cdot \alpha_{f,m}^n \cdot \Big[D_{\max} - D_f(\lambda_{f,m}, Q_{f,m})\Big], \tag{9a}$$

s.t.

$$\sum_{f=1}^{F}\sum_{m=1}^{M} \beta_{f,m} \cdot R_f(\lambda_{f,m}, Q_{f,m}) \le R_{\max}, \tag{9b}$$

$$\sum_{f=1}^{F}\sum_{m=1}^{M} \beta_{f,m} \cdot C_f(\lambda_{f,m}, Q_{f,m}) \le C_{\max}, \tag{9c}$$

$$d_f \le \Delta T,\ \forall f \in \mathcal{F}, \tag{9d}$$

$$\alpha_{f,m}^n \le \beta_{f,m},\ \forall n \in \mathcal{N}, \forall f \in \mathcal{F}, \forall m \in \mathcal{M}, \tag{9e}$$

$$\alpha_{f,m}^n \cdot R_f(\lambda_{f,m}, Q_{f,m}) \le B_n,\ \forall n \in \mathcal{N}, \forall f \in \mathcal{F}, \forall m \in \mathcal{M}, \tag{9f}$$

$$\sum_{m=1}^{M} \alpha_{f,m}^n \le 1, \forall n \in \mathcal{N}, \forall f \in \mathcal{F}, \tag{9g}$$

$$\alpha_{f,m}^n \in \{0,1\},\ \forall n \in \mathcal{N}, \forall f \in \mathcal{F}, \forall m \in \mathcal{M}, \tag{9h}$$

$$\beta_{f,m} \in \{0,1\},\ \forall f \in \mathcal{F}, \forall m \in \mathcal{M}. \tag{9i}$$

In the ILP problem **P2**, the objective function and the first three constraints are equivalent to those in the original problem **P1**, where we define the reconstructed video distortion reduction (or video quality improvement) after decoding the m-th representation of video f as the utility function, i.e., $U_f(\lambda_{f,m}, Q_{f,m}) = D_{\max} - D_f(\lambda_{f,m}, Q_{f,m})$. Specifically, D_{max} represents a constant maximal distortion when no video is decoded and thus $[D_{max} - D_f(\lambda_{f,m}, Q_{f,m})]$ denotes the distortion reduction after successful decoding of the representation with coding parameter pair $(\lambda_{f,m}, Q_{f,m})$. The constraint (9e) sets up a consistent relationship between the decision variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, ensuring that the representation selected by a user is already encoded and available at the server. The constraint (9f) specifies the possible representations of all video streams that can be supported by user n's download link capacity B_n. The constraint (9g) ensures that at most one representation of a video f is selected by a user n.

The optimal solution of the ILP problem **P2** can be obtained by the generic solver IBM ILOG CPLEX [11], using a branch-and-cut search. The branch-and-cut procedure manages a search tree consisting of nodes, each of which represents a relaxed LP subproblem to be solved. It then involves running a branch and bound algorithm to create two new nodes from a parent node, and adding additional cutting planes to tighten the LP relaxations and reduce the number of branches required to solve the original ILP. In general, the branch-and-cut search requires exponential computational complexity to achieve the optimal solution. Therefore, the ILP problem **P2** is NP-hard. Specifically, it can be observed that the cardinality of the decision variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is NFM and FM, respectively. By using the branch and bound method for the binary decision variables, in the worst case, the number of nodes observed by the CPLEX solver would be upper bounded by $2^{NFM} \times 2^{FM}$ and at each node the solver needs to solve a relaxed LP problem with the SIMPLEX method. This corresponds to an exponential computational complexity $O(2^{F \cdot 2M \cdot 2N})$ and thus incurs an incredibly long execution time when the problem scale becomes large.

To reduce the actual execution time in practical large scale problem, we can terminate the branch-and-cut procedure earlier than a completed proof of optimality, e.g., by setting an error bound (relative optimality tolerance) or a time limit. Although the relative optimality tolerance can guarantee a near-optimal solution within a certain percentage of the optimal solution, in the worst case, the number of nodes on the search tree is still an exponential function of the cardinality of the decision variables, which still indicates exponential time complexity. On the other hand, if we set the time limit as an acceptable value (e.g., several seconds), it is likely that the CPLEX solver would only obtain a poor objective value since only a small subset of nodes are searched and processed.

V. Equivalent Submodular Maximization Problem and Algorithm Design

In order to efficiently cope with the difficulties of solving the original problems **P1** and **P2**, in this section, we convert the general optimization formulation **P1** to an equivalent set function optimization problem. We prove that it is a submodular maximization problem over independence constraints. By utilizing the diminishing return characteristics of the submodular functions, we finally develop new practically efficient algorithms with polynomial computational time complexity and theoretical approximation guarantees.

A. Equivalent Problem Formulation as a Set Function Optimization

First, the finite ground set of representations in the original problem **P1** can be written as:

$$\begin{aligned}\mathcal{E} &= \{e_{f,m} | \forall f \in \mathcal{F},\ \forall m \in \mathcal{M}\} \\ &= \{e_{1,1}, \cdots, e_{1,M}, \ldots, e_{f,m}, \ldots, e_{F,1}, \ldots, e_{F,M}\}.\end{aligned} \tag{10}$$

In Eq. (10), a specific element $e_{f,m}$ exists if the m-th representation is selected to be encoded for a video stream f. Therefore, the ground set $\mathcal{E}$ denotes the full set of all representations of all video streams that are encoded by the DASH server. By integrating Eq. (4), the encoding delay constraint in Eq. (6d) can be rewritten as:

$$\frac{K(2\lambda_{f,m}+1)^2\cdot\eta_f(Q_{f,m})\cdot c_0}{C_f(\lambda_{f,m},Q_{f,m})}\leq\Delta T,\ \forall f\in\mathcal{F},\ \forall m\in\mathcal{M}. \tag{11}$$

Therefore, the feasible region of the allocated CPU load for encoding the m-th representation of video stream f can be denoted as:

$$C_f(\lambda_{f,m},Q_{f,m})\geq\frac{K(2\lambda_{f,m}+1)^2\cdot\eta_f(Q_{f,m})\cdot c_0}{\Delta T},\quad \forall f\in\mathcal{F},\ \forall m\in\mathcal{M}. \tag{12}$$

As long as $C_f(\lambda_{f,m},Q_{f,m})$ lies within the feasible region defined by Eq. (12), the encoding delay for any representation $e_{f,m}\in\mathcal{E}$ would not violate the live encoding constraints in Eq. (6d). When the power (CPU load) related constraint in Eq. (6c) is further taken into account, the optimal solution would be achieved with the minimum CPU load consumed for each representation, i.e., $[K(2\lambda_{f,m}+1)^2\cdot\eta_f(Q_{f,m})\cdot c_0]/\Delta T$. In other words, all the optimal representations should be encoded with the maximum encoding time $d_f=\Delta T$.

For the users, let Ω_n denote the set of representations of all video streams that can be supported by user n's download link capacity B_n, i.e.,

$$\Omega_n=\{e_{f,m}\in\mathcal{E}|R_f(\lambda_{f,m},Q_{f,m})\leq B_n,\quad \forall f\in\mathcal{F},\ \forall m\in\mathcal{M}\}\subseteq\mathcal{E}. \tag{13}$$

Define a specific DASH encoding decision set $\mathcal{A}\subseteq\mathcal{E}$ with each element $e_{f,m}\in\mathcal{A}$ indicating the actual encoding of the m-th representation for video f. Then, based on $\mathcal{A}$, the expected average reduction in video distortion for user n can be derived as:

$$\bar{D}_n(\mathcal{A})=\sum_{f=1}^{F}\sum_{m=1}^{M}\left[\prod_{j=1}^{m-1}(1-\mathbf{1}_{e_{f,j}\in(\mathcal{A}\cap\Omega_n)})\right]\cdot\mathbf{1}_{e_{f,m}\in(\mathcal{A}\cap\Omega_n)}\cdot\rho_f^n\cdot\Big[D_{\max}-D_f(\lambda_{f,m},Q_{f,m})\Big], \tag{14}$$

where ρ_f^n is the probability of user n requesting video stream f, and $\mathbf{1}|_{x\in\mathcal{X}}$ is an indicator function, the value of which is 1 if $x\in\mathcal{X}$ and 0 otherwise.

Therefore, the original optimization problem **P1** can be reformulated as a constrained set function optimization problem, as follows:

$$\textbf{P3:}\ \arg\max_{\mathcal{A}\subseteq\mathcal{E}}\quad D(\mathcal{A})=\sum_{n=1}^{N}\bar{D}_n(\mathcal{A}), \tag{15a}$$

s.t.

$$\mathcal{A}\in\mathcal{I}_R=\Bigg\{\mathcal{A}'\subseteq\mathcal{E}\Bigg|\sum_{f=1}^{F}\sum_{m=1}^{M}\mathbf{1}|_{e_{f,m}\in\mathcal{A}'}\cdot R_f(\lambda_{f,m},Q_{f,m})\leq R_{\max}\Bigg\}, \tag{15b}$$

$$\mathcal{A}\in\mathcal{I}_C=\Bigg\{\mathcal{A}'\subseteq\mathcal{E}\Bigg|\sum_{f=1}^{F}\sum_{m=1}^{M}\mathbf{1}|_{e_{f,m}\in\mathcal{A}'}\cdot C_f(\lambda_{f,m},Q_{f,m})\leq C_{\max}\Bigg\}, \tag{15c}$$

$$C_f(\lambda_{f,m},Q_{f,m})=\frac{K(2\lambda_{f,m}+1)^2\cdot\eta_f(Q_{f,m})\cdot c_0}{\Delta T},\quad \forall f\in\mathcal{F},\ \forall m\in\mathcal{M}. \tag{15d}$$

Comparing the original problem **P1** with the equivalent set function optimization formulation **P3**, it can be seen that the objective function and the first three constraints in problem **P1** are transformed to Eqs. (15a)-(15d) in problem **P3**, respectively, while the available source coding parameter constraint in Eq. (6e) in problem **P1** is expressed as the representation set $\mathcal{M}=\Lambda\times\mathcal{Q}$. It should be noted that in the reformulated ILP problem **P2** and its equivalent submodular maximization problem **P3**, the network traffic is reflected by the users' download link capacity constraint (9f) and the set of representations supported by the user's download link capacity Ω_n in Eq. (13), respectively. Here, a simple assumption is that we have certain prior information about the users, i.e., the downlink bandwidth B_n of any user n, and user n can choose to download a representation only if its bitrate does not exceed B_n. However, taking into account some more complicated network architectures and transmission/routing schemes is beyond the scope of this paper, and will be investigated in our future work.

We show in the next subsection that the equivalent optimization problem **P3** is a maximization problem of a submodular function over general independence constraints, the structure of which can be further utilized to develop a computationally efficient solution with provable approximation gaps.

B. Proof of Submodularity

We show now that the problem **P3** is submodular. We first review and include the definition of independence systems, and submodular functions according to [26]–[28], respectively.

Definition 1. *Independence system: A pair $\mathcal{P}=(\mathcal{E},\mathcal{I})$, where $\mathcal{E}$ is a finite ground set and $\mathcal{I}$ is a collection of subsets of $\mathcal{E}$, is an independence system if and only if it satisfies the following axioms:*
(I1) $\mathcal{I}$ is nonempty, and $\emptyset\in\mathcal{I}$.
(I2) If $\mathcal{X}\subseteq\mathcal{Y}$ and $\mathcal{Y}\in\mathcal{I}$, then $\mathcal{X}\in\mathcal{I}$.

Definition 2. *Submodularity: Let $\mathcal{E}$ be a finite ground set, and a set function $g:2^{\mathcal{E}}\to\mathbb{R}$ is submodular if and only if for any sets $\mathcal{X}\subseteq\mathcal{Y}\subseteq\mathcal{E}$ and for any $e\in(\mathcal{Y}\setminus\mathcal{X})$, we have*

$$g(\mathcal{X})+g(\mathcal{Y})\geq g(\mathcal{X}\cup\mathcal{Y})+g(\mathcal{X}\cap\mathcal{Y}), \tag{16}$$

or equivalently

$$g(\mathcal{X}\cup\{e\})-g(\mathcal{X})\geq g(\mathcal{Y}\cup\{e\})-g(\mathcal{Y}), \tag{17}$$

which captures the diminishing return characteristics such that the benefit of adding a new element into the set would decrease as the set becomes larger.

Then, we prove for the problem **P3** that the constraints form an independence system and the objective function is monotone submodular.

Proposition 1. *The DASH server encoding rate and power constraints in Eq. (15b) and Eq. (15c), respectively, form an independence system on the ground set $\mathcal{E}$ as defined in Eq. (10).*

Proof: Here, we only provide the justification that the total encoding rate constraint in Eq. (15b) is an independence system. The proof of the total encoding power constraint in Eq. (15c) can be obtained in a similar way.

From the definition of $\mathcal{I}_R$, it is obvious that it is not empty and $\emptyset$ is an element of $\mathcal{I}_R$. For any $\mathcal{X}\subseteq\mathcal{Y}$, the total encoding rate based on $\mathcal{X}$ would be smaller than or equal to that based on $\mathcal{Y}$. If $\mathcal{Y}\in\mathcal{I}_R$, then the total encoding rate based on $\mathcal{Y}$ would not exceed $R_{\max}$, which in turn indicates that the total encoding rate based on $\mathcal{X}$ does not exceed $R_{\max}$ and $\mathcal{X}\in\mathcal{I}_R$. It is thus checked that both axioms (I1) and (I2) in Definition 1 are satisfied for $\mathcal{I}_R$, and the total encoding rate constraint in Eq. (15b) forms an independence system $(\mathcal{E}, \mathcal{I}_R)$. ■

Proposition 2. *The objective function in Eq. (15a) is a monotone submodular function over the ground set $\mathcal{E}$ as defined in Eq. (10).*

Proof: According to the property of monotonicity and submodularity, the summation over a set of monotone submodular functions is also monotone submodular. Thus, to prove the monotone submodularity of $\sum_{n=1}^{N} \bar{D}_n(\mathcal{A})$, we only need to prove that the set function $\bar{D}_n(\mathcal{A})$ is monotone submodular for every user $n \in \mathcal{N}$.

1) Monotonicity: For any $\mathcal{X} \subseteq \mathcal{E}$ and any $e_{f,m} \in \mathcal{E} \setminus \mathcal{X}$, we have $\bar{D}_n(\mathcal{X} \cup \{e_{f,m}\}) \geq \bar{D}_n(\mathcal{X})$, since encoding and providing a new representation at the DASH server will at least not degrade the aggregate video quality (i.e., the average video distortion reduction will not decrease). Therefore, for any two placement sets $\mathcal{X} \subseteq \mathcal{Y} \subseteq \mathcal{E}$, we have $\bar{D}_n(\mathcal{Y}) \geq \bar{D}_n(\mathcal{X})$, which indicates that the objective function in Eq. (15a) is monotone non-decreasing.

2) Submodularity: Consider any two DASH encoding decision sets $\mathcal{X} \subseteq \mathcal{Y} \subseteq \mathcal{E}$, and suppose adding a new element $e_{f,m} \in \mathcal{E} \backslash \mathcal{Y}$ to both sets. If $e_{f,m} \notin \Omega_n$, then $e_{f,m}$ is not feasible and for both sets the marginal values of adding $e_{f,m}$ is zero. If $e_{f,m} \in \Omega_n$, we consider the following two cases.

i) There exists $e_{f,y} \in (\mathcal{Y} \cap \Omega_n)$ with $y \leq m$, i.e., based on the encoding decision set $\mathcal{Y}$ user n downloads a better or equal quality representation y of video stream f from the DASH server. In this case, it can be derived from Eq. (14) that $\bar{D}_n(\mathcal{Y} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{Y}) = 0$. On the other hand, due to the monotonicity, for the decision set $\mathcal{X}$ we always have $\bar{D}_n(\mathcal{X} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{X}) \geq 0$. Therefore, the relationship of both marginal values is given by $\bar{D}_n(\mathcal{Y} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{Y}) \leq \bar{D}_n(\mathcal{X} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{X})$.

ii) There exists $e_{f,y} \in (\mathcal{Y} \cap \Omega_n)$ with $y > m$, i.e., based on the encoding decision set $\mathcal{Y}$ user n downloads a worse quality representation y of video stream f from the DASH server. In this case, it can be derived from Eq. (14) that $\bar{D}_n(\mathcal{Y} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{Y}) = \rho_f^n[D_f(\lambda_{f,y}, Q_{f,y}) - D_f(\lambda_{f,m}, Q_{f,m})]$. On the other hand, for the encoding decision set $\mathcal{X}$, since $\mathcal{X} \subseteq \mathcal{Y}$, user n can only download representation x of video stream f with $x \geq y$. Thus, the resulting marginal value is $\bar{D}_n(\mathcal{X} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{X}) = \rho_f^n[D_f(\lambda_{f,x}, Q_{f,x}) - D_f(\lambda_{f,m}, Q_{f,m})]$. Since $x \geq y$, we have $R_f(\lambda_{f,y}, Q_{\lambda,y}) \geq R_f(\lambda_{f,x}, Q_{f,x})$ and thus $D_f(\lambda_{f,y}, Q_{\lambda,y}) \leq D_f(\lambda_{f,x}, Q_{f,x})$. Therefore, the relationship of both marginal values is given by $\bar{D}_n(\mathcal{Y} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{Y}) \leq \bar{D}_n(\mathcal{X} \cup \{e_{f,m}\}) - \bar{D}_n(\mathcal{X})$.

For both cases, the marginal value decreases as the set becomes larger, which satisfies Eq. (17) in Definition 2. Hence, the submodularity is proved. ■

In Proposition 2, we have justified that Eq. (15a) is a monotone submodular function. Further observing the encoding rate and power constraints in Eqs. (15b) and (15c), each element $e_{f,m} \in \mathcal{A}$ has non-uniform rate and power cost of $R_f(\lambda_{f,m}, Q_{f,m})$ and $C_f(\lambda_{f,m}, Q_{f,m})$, while the DASH server has the encoding bitrate and CPU load budget of R_{max} and C_{max}, respectively. These two constraints can be viewed as two knapsack constraints on the finite ground set $\mathcal{E}$. Therefore, the optimization problem **P3** is a submodular maximization problem subject to two knapsack constraints. Such a problem is generally NP-hard and requires exponential computational complexity to reach the optimum by either integer linear programming or other optimization methods [29]. But submodularity ensures that the greedy algorithm provides an effective approximation to the optimal solution of this NP-hard problem.

C. Approximation Algorithm

To efficiently solve the constrained submodular maximization problem in Eq. (15) with polynomial time complexity and theoretical approximation guarantees, we develop an (ω, k)-weighted cost benefit (WCB) greedy algorithm [30]. The two system parameters, $\omega \in [0, 1]$ and $k = 0, 1, 2, \ldots$, specify the relative weight between the rate and the power cost and the size of the initial set, respectively. Specifically, the proposed (ω, k)-WCB greedy algorithm considers all feasible initial sets $\mathcal{A}^0 \subseteq \mathcal{E}$ of cardinality k. Starting from any initial set $\mathcal{A}^0$, at step t, the weighted cost benefit greedy procedure iteratively searches over the remaining set $\mathcal{E}^{t-1} \setminus \mathcal{A}^{t-1}$ and inserts into the partial solution $\mathcal{A}^{t-1}$ an element according to Eqs. (18)-(20), until the remaining set reduces to an empty set. In other words, this procedure adds at each iteration an element that maximizes the weighted marginal benefit $D(\mathcal{A}^{t-1} \cup \{e_{f,m}\}) - D(\mathcal{A}^{t-1})$ and cost $R_f(\lambda_{f,m}, Q_{f,m}), C_f(\lambda_{f,m}, Q_{f,m})$ ratio among all elements still affordable with the remaining rate and power budget until no more element can be added. The proposed (ω, k)-WCB greedy algorithm then enumerates all initial sets $\mathcal{A}^0 \subseteq \mathcal{E}$ of cardinality k, augments each of them following the cost benefit greedy procedure, and selects the initial set achieving the largest value of the objective function $D(\mathcal{A}) = \sum_{n \in \mathcal{N}} \bar{D}_n(\mathcal{A})$ and determines its solution set as the final encoded representation set $\mathcal{A}$. We finally note that in some extreme cases, the algorithm reduces to be pure rate cost benefit when $\omega = 1$ and pure power cost benefit when $\omega = 0$. The complete algorithm is described in Algorithm 1.

Algorithm 1 (ω, k)-weighted cost benefit greedy algorithm

```
For all initial sets $\mathcal{A}^0 \subseteq \mathcal{E}$ such that $|\mathcal{A}^0| = k$, implement the following weighted cost benefit greedy procedure.
  Initialization:
    1) Set $\mathcal{E}^0 = \mathcal{E}$ and $t = 1$.
  Greedy Search Iteration: (at step $t = 1, 2, 3, \ldots$)
    1) Given a partial solution $\mathcal{A}^{t-1}$, find
       $e_{f_t,m_t} = \arg \max_{e_{f,m} \in \mathcal{E}^{t-1} \setminus \mathcal{A}^{t-1}} \omega \cdot \frac{D(\mathcal{A}^{t-1} \cup \{e_{f,m}\}) - D(\mathcal{A}^{t-1})}{R_f(\lambda_{f,m}, Q_{f,m})} + (1-\omega) \cdot \frac{D(\mathcal{A}^{t-1} \cup \{e_{f,m}\}) - D(\mathcal{A}^{t-1})}{C_f(\lambda_{f,m}, Q_{f,m})}$.   (18)
  Update and Determination:
    1) Set $\mathcal{A}^t = \mathcal{A}^{t-1} \cup \{e_{f_t,m_t}\}$, and $\mathcal{E}^t = \mathcal{E}^{t-1}$, if
       $\sum_{f=1}^{F} \sum_{m=1}^{M} \mathbf{1}|_{e_{f,m} \in (\mathcal{A}^{t-1} \cup \{e_{f_t,m_t}\})} \cdot R_f(\lambda_{f,m}, Q_{f,m}) \leq R_{\max}$,   (19)
       and
       $\sum_{f=1}^{F} \sum_{m=1}^{M} \mathbf{1}|_{e_{f,m} \in (\mathcal{A}^{t-1} \cup \{e_{f_t,m_t}\})} \cdot C_f(\lambda_{f,m}, Q_{f,m}) \leq C_{\max}$;   (20)
       otherwise, set $\mathcal{A}^t = \mathcal{A}^{t-1}$, and $\mathcal{E}^t = \mathcal{E}^{t-1} \setminus \{e_{f_t,m_t}\}$.
    2) If $\mathcal{E}^t \setminus \mathcal{A}^t \neq \emptyset$, set $t = t + 1$ and return to the greedy search iteration; otherwise, stop the iteration.
The solution is obtained and output as $\mathcal{A}$, which has the largest value of the objective function $D(\mathcal{A}) = \sum_{n \in \mathcal{N}} \bar{D}_n(\mathcal{A})$ over all the possible choices of the initial sets $\mathcal{A}^0 \subseteq \mathcal{E}$.
```

The proposed algorithm can be implemented in the representation selection module in Fig. 1. Afterwards, if there is no dramatic change of the source videos or the network conditions, it is only necessary to run the proposed algorithm periodically with a relatively long period (e.g., tens or hundreds of minutes) in order to adapt to possible changes in the system; otherwise, the proposed algorithm will be re-implemented whenever a dramatic change occurs. In terms of computational complexity, the running time of the proposed algorithm is $O((FM)^{k+1}N)$, indicating a polynomial time complexity and a very short additional implementation delay. As the value of k increases, the running time of the proposed algorithm becomes longer while the performance improves. As shown in [30], when $k \geq 3$ and in the case of one active knapsack constraint, the theoretical worst-case performance guarantee of the cost benefit algorithm is $1 - 1/e$, i.e., its solution achieves at least the ratio $1 - 1/e \approx 0.632$ of the optimal objective value. Although there is no such theoretical guarantee for the case when both knapsack constraints are active, as will be shown in the simulation results in Section VI, the algorithm's performance approximation ratio is generally above 0.9 in practice.

VI. Experimental Results

In this section, we evaluate the performance of the DASH representation selection optimization framework, and derive simple guidelines for effective content production in adaptive streaming systems under different simulation settings.

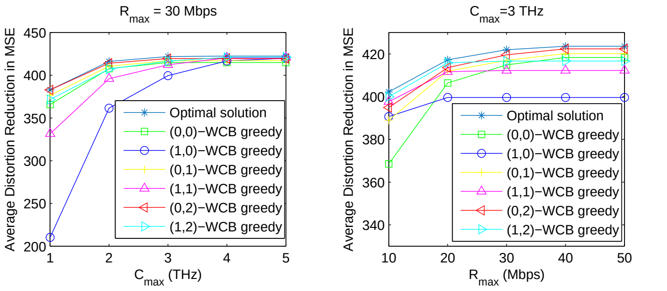
A. Simulation Settings

We implement the proposed framework on a 48-processor server with 252 GB of RAM using Linux 3.1 kernel, where each processor is an Intel Xeon CPU E5-2680 at a clock frequency of 2.50 GHz. We suppose that there are $N = 10$ users and their download bandwidth B_n is randomly distributed in the rate range of $[1, 10]$ Mbps. Three test video sequences ($F = 3$, *Crowd Run*, *Tractor*, and *Sunflower*) with 1080p resolution (1920×1080) [31] are selected as the source video streams to be encoded at the DASH server. These three test video sequences correspond to different content types, i.e., dense object motion for *Crowd Run* sequence, camera movement and medium object motion for *Tractor* sequence, and small object motion for *Sunflower* sequence, respectively. Typically, the distortion decreases faster with the rate and the CPU load when the video content has larger complexity. We assume that the encoding time of each video frame is limited by $\Delta T = 0.03$ s, and the constant maximal distortion is set as $D_{\max} = 500$. At a frame rate of 30 fps, we further encode each video sequence f into $M = 63$ representations with the coding parameter pair $(\lambda_{f,m}, Q_{f,m}) \in \Lambda \times \mathcal{Q}$, where $\Lambda = \{2, 6, 10\}$ and the corresponding QP value ranges between 30 and 50. We further assume that the popularity of the three sequences follows a Zipf distribution with parameter 0.56 [32], i.e., the requesting probabilities of *Crowd Run*, *Tractor*, and *Sunflower* sequences are 0.45, 0.31, and 0.24, respectively [1].

B. Simulation results of the Proposed Algorithm

In this subsection, we illustrate and analyze the simulation results of the proposed (ω, k)-WCB greedy algorithm under different maximum bitrate and power (CPU load) constraints, and investigate the impact of the algorithm parameters ω and k on the overall performance. The optimal solution of the ILP **P2** obtained by the generic solver IBM ILOG CPLEX [11] is also given as the benchmark.

In Fig. 2(a), we set the maximum bitrate capacity at the server to $R_{max} = 30$ Mbps, vary the value of the maximum CPU load C_{max}, and illustrate the average distortion reduction per user under different parameter settings of the proposed (ω, k)-WCB greedy algorithm. The optimal solution of the ILP **P2** obtained by the IBM ILOG CPLEX solver [11] using a branch and bound method with a very high (i.e., exponential) time complexity $O(2^{F \cdot 2M \cdot 2N})$ is given as a performance upper bound. It confirms that the proposed algorithm achieves a good approximation performance but with a lower (i.e., polynomial) time complexity $O((FM)^{k+1}N)$. Two observations can



(a) (b)

Fig. 2. (a) Given $R_{\max} = 30$ Mbps, average distortion reduction per user vs. maximum CPU load constraint $C_{\max}$; and (b) given $C_{\max} = 3$ THz, average distortion reduction per user vs. maximum encoding bitrate constraint $R_{\max}$.

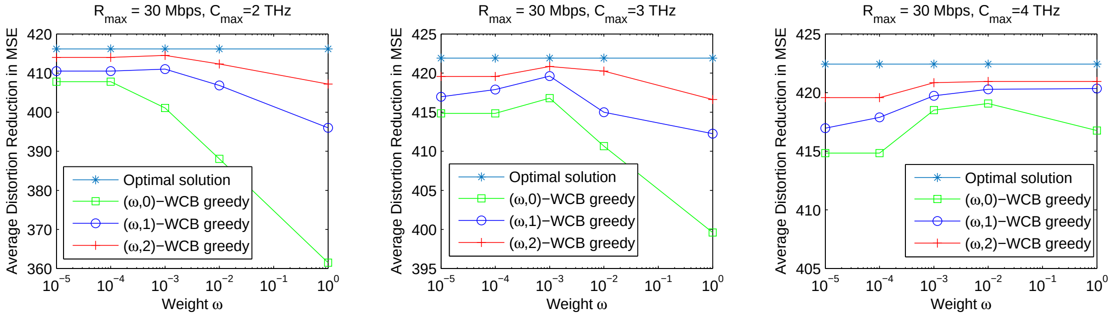
be made from the curves in Fig. 2(a). Given a weight ω, enlarging the size of the initial set k from 0 to 2 incurs higher average distortion reduction per user for all values of $C_{\max}$, but the computational complexity also increases from $O(FMN)$ to $O((FM)^2N)$. On the other hand, when k is fixed, the algorithm performance is affected by the values of the maximum CPU load $C_{\max}$ and the relative cost weight ω. Obviously, the average distortion reduction per user improves if we increase the maximum CPU load C_{max} at the server. In addition, it can also be seen that when the maximum CPU load is small (e.g., $C_{\max} = 1$ THz), the algorithm with the minimum weight $\omega = 0$ (power cost benefit, e.g., 1-approximation ratio for $k = 2$) outperforms the weight assignment of $\omega = 1$ (rate cost benefit, e.g., 0.971-approximation ratio for $k = 2$), and vice versa. The reason is as follows. For small $C_{\max}$, the power (CPU load) becomes a scarcer resource compared to the rate, which causes the CPU load constraint to be active while the encoding bitrate constraint remains inactive. In this case, the power cost benefit greedy algorithm that adds an element maximizing the marginal benefit and power cost ratio at each iteration step would achieve better performance.

The maximum CPU load at the server is then fixed at $C_{\max} = 3$ THz, while the value of maximum encoding bitrate varies from 10 Mbps to 50 Mbps. In this case, it can be seen in Fig. 2(b) that for the same initial set seize k, the two curves corresponding to $\omega = 0$ and $\omega = 1$ intersect at a certain point of maximum encoding bitrate. To the left of this intersecting point, the encoding bitrate is a scarcer resource and the rate cost benefit greedy algorithm with $\omega = 1$ would achieve a better performance, and vice versa.

Then, the average distortion reduction per user versus weight ω is shown in Fig. 3 for the cases of $R_{max} = 30$ Mbps, $C_{max} = 2, 3, 4$ THz, and $k = 0, 1, 2$, respectively, when both the encoding bitrate and CPU load constraints become active. Again, for a given value of ω, larger k indicates higher average distortion reduction. In addition, for all values of k, there exists an optimal weight (e.g., $\omega^* = 0.001$ in Fig. 3(b)) achieving the peak average distortion reduction (0.988, 0.995, and 0.998-approximation ratio for $k = 0$, 1 and 2), which indicates the best tradeoff between the rate and power cost when both resources are limited. Through comparison of Figs. 3(a)-3(c), it can be concluded that such optimal weight value ω^* is affected by the allocation of R_{max} and C_{max}. Since the maximum encoding bitrate constraint R_{max} is fixed in Fig. 3, ω^* would become larger with the increment of the maximum CPU load C_{max}. We show the average distortion reduction per user versus weight ω curves for the cases of $C_{max} = 3$ THz and $R_{max} = 20, 30, 40$ Mbps in Fig. 4, where the similar conclusion can be drawn.

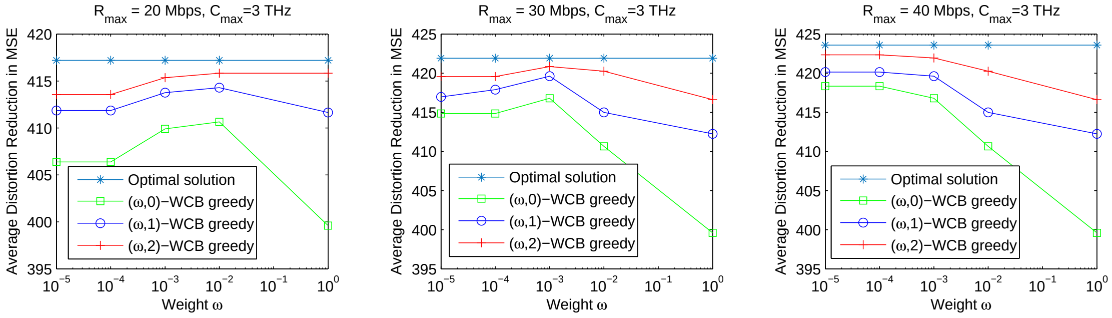
In terms of system design, the messages that can be concluded from the above observations of the proposed (ω, k)-WCB greedy algorithm are in the following. 1) The size of the initial size k adjusts the

[1] Please not that this popularity distribution is chosen as an illustrative example. The proposed algorithm can be applied to any other popularity distribution, which is also experimentally justified in Table IV in Section VI-D.



(a) (b) (c)

Fig. 3. Given $R_{\max} = 30$ Mbps, average distortion reduction per user vs. weight ω when maximum CPU load constraint $C_{\max}$ is set to (a) 2 THz, (b) 3 THz, and (c) 4 THz.



(a) (b) (c)

Fig. 4. Given $C_{\max} = 3$ THz, average distortion reduction per user vs. weight ω when maximum encoding bitrate constraint $R_{\max}$ is set to (a) 20 Mbps, (b) 30 Mbps, and (c) 40 Mbps.

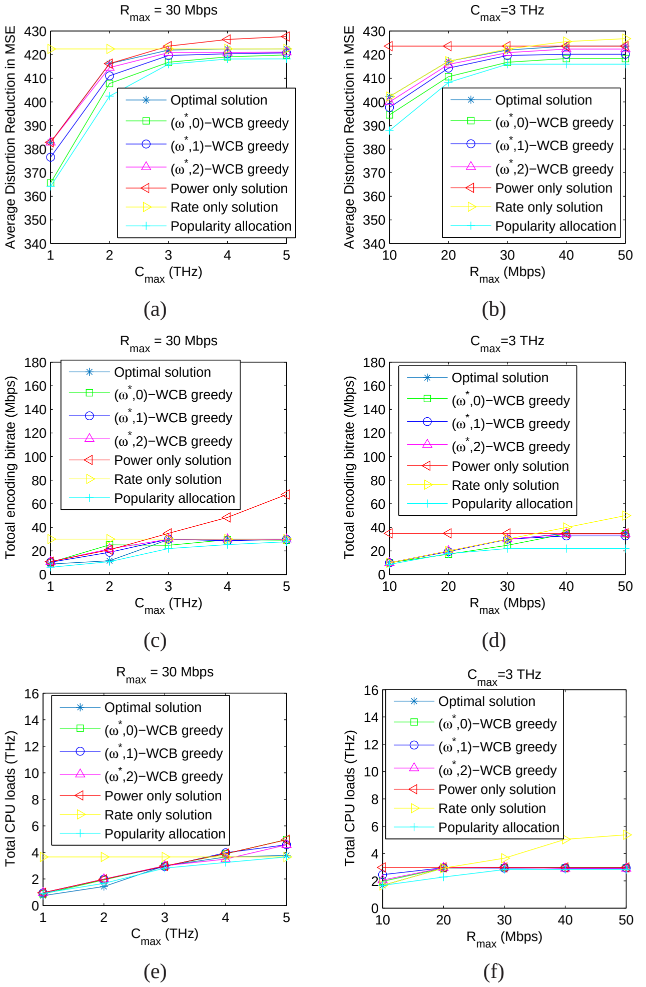
tradeoff between the average distortion reduction performance and the computational complexity. A larger number of k improves the algorithm's performance, but at the cost of a longer execution time. 2) The relative weight ω controls the tradeoff between the rate and power cost. Comparing the rate and power resources, when the rate resource is scarcer, a larger weight value should be allocated to make the proposed algorithm more rate efficient; and vice versa.

C. Performance Comparison

In this subsection, the performance of the proposed (ω, k)-WCB greedy algorithm is compared with the other four baseline schemes: 1) the optimal solution of the ILP **P2** solved by the generic solver IBM ILOG CPLEX [11], which provides a performance upper bound; 2) the power only solution, i.e., the solution of the ILP **P2** without the maximum encoding bitrate constraint; 3) the rate only solution, i.e., the solution of the ILP **P2** without the maximum power (CPU load) constraint; and 4) the popularity based allocation algorithm, which allocates both the encoding bitrate and the encoding CPU load budgets for videos in proportion to their popularity, and then greedily adds encoded representations for each video until either the maximum bitrate or the maximum CPU load of that video is reached.

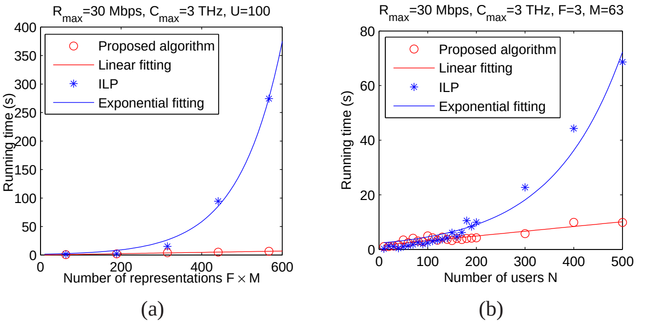
The relationship between these baseline schemes and the existing works on server-side DASH representation selection is as follows. Fundamentally, the ILP formulation proposed in [2] can be viewed as a special case of problem **P2** without the maximum encoding delay and bitrate constraints. Therefore, the corresponding algorithm performance is upper bounded by the baseline scheme 2). On the other hand, Ref. [10] validates the optimized representations obtained by solving the ILP in [4] in a practical scenario, by generating a 24-hour streaming scenario based on YouTube traces and device statistics for Hulu and Netflix. Since the ILP formulation proposed in [4] and [10] can be viewed as a special case of problem **P2** without the maximum encoding delay and power constraints, its performance is upper bounded by the baseline scheme 3). In addition, there are two remarks. First, the ILP formulations in [2], [4] and [10] do not include the rate control consideration. To make a "fair" comparison, we assume that the ground set of all possible representations is already pre-encoded with known bitrates, qualities and power consumptions when solving these ILPs. Second, in practice, the algorithm running time is another performance metric that has the same or even greater importance than the average distortion reduction per user. The computational complexity of the baseline schemes 1)-3) is all exponential since they all have to solve a large scale ILP. We will show later the advantage of the proposed algorithm over the ILP solution in terms of the algorithm running time.

First, we fix the constraint of the maximum encoding bitrate at the server as $R_{max} = 30$ Mbps, vary the value of the maximum CPU load C_{max} from 1 THz to 5 THz, and show the comparison of the average distortion reduction per user, the actual total encoding bitrate, and the actual total encoding CPU load achieved by different algorithms in Figs. 5(a), 5(c), and 5(e), respectively. Compared with the optimal solution, the proposed (ω, k)-WCB greedy algorithm with the optimal weight $\omega = \omega^*$ can achieve a good approximation performance for the representation selection at the DASH server in terms of the largest average distortion reduction per user, while both the maximum encoding bitrate and the maximum encoding CPU load constraints are satisfied. For all different values of C_{max}, for example, the proposed $(\omega^*, 0)$-WCB greedy algorithm can achieve a 0.955-approximation ratio. Since the optimal distortion reduction per user in MSE is around 400, this approximation ratio means a near-optimal performance that is only less than 20 lower in MSE than the optimal one. When k is enlarged to 2, this worst case approximation ratio would be improved to 0.993, which indicates a very good approximation of the optimal solution. It can be further



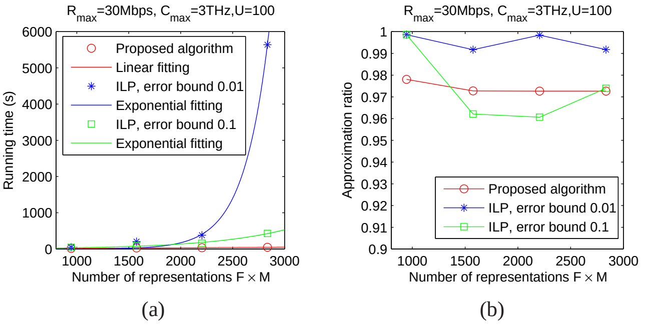
(a) (b) (c) (d) (e) (f)

Fig. 5. (a) Average distortion reduction per user, (c) total encoding bitrate, and (e) total encoding CPU load vs. maximum CPU load constraint $C_{\max}$ curves when maximum encoding bitrate $R_{\max}$ is fixed at 30 Mbps; and (b) average distortion reduction per user, (d) total encoding bitrate, and (f) total encoding CPU load vs. maximum encoding bitrate $R_{\max}$ when maximum CPU load constraint $C_{\max}$ is fixed at 3 THz.



(a) (b)

Fig. 6. Running time vs. (a)number of representations and (b) number of users.



(a) (b)

Fig. 7. (a) Running time and (b) approximation ratio vs. number of representations, under different error bound settings for the ILP solver.

seen that the proposed (ω^*, k)-WCB greedy algorithm outperforms the popularity based allocation algorithm. The reason is that, in addition to the popularity, the video content information is also a very important factor in accordance with which both the encoding bitrate and CPU load budgets should be properly allocated among different videos. When C_{max} is small (e.g., 1 and 2 THz) and becomes the only active constraint, the power only solution (the solution of the ILP **P2** without maximum encoding bitrate constraint) achieves similar average distortion reduction per user to the optimal solution of the ILP **P2**. In this case, even though there is still some encoding bitrate budget remaining for more video representations, the actual total encoding CPU load of the ILP **P2** with/without maximum encoding bitrate constraint reaches the maximum CPU load C_{max}, which prevents from encoding any additional representation due to the lack of CPU capacity. On the other hand, when C_{max} is larger (e.g., 3, 4, and 5 THz) and the bitrate becomes a scarcer resource, the power only solution outperforms the optimal solution of the ILP **P2**. However, it should be noted that the total encoding bitrate exceeds the maximum encoding bitrate constraint $R_{max} = 30$ Mbps. The rate only solution (the optimization based representation selection algorithm in [4]) achieves a stable average distortion reduction per user for different values of C_{max}, since the maximum CPU load constraint is not taken into account. When C_{max} is large and R_{max} becomes the only active constraint, its performance is similar to the optimal solution of the ILP **P2**. However, when C_{max} is reduced, since it is not power optimized, the total CPU load consumed by such representation selection algorithm exceeds the maximum affordable CPU load C_{max}, i.e., its solution is infeasible in practical power constrained system design. Similar observation can be made from Figs. 5(b), 5(d), and 5(f), where the maximum CPU load at the server is fixed at $C_{\max} = 3$ THz, while the value of maximum encoding bitrate varies from 10 Mbps to 50 Mbps.

In order to gain further insight into the difference between the algorithms, in Table II, we list the comparison of the representation selection results, in terms of the coding parameter pair $(\lambda_{f,m}, Q_{f,m})$ and the corresponding encoding bitrate $R_{f,m}$ and CPU load $C_{f,m}$, when $R_{\max} = 30$ Mbps and $C_{\max} = 3$ THz. It can be seen that the representations selected by the proposed $(\omega^*, 2)$-WCB greedy algorithm do not deviate much from the optimal representations of the ILP **P2**, while a 0.998-approximation ration is achieved with both the maximum encoding bitrate and CPU load constraints satisfied. The fundamental reason why the proposed $(\omega^*, 2)$-WCB greedy algorithm outperforms the popularity based allocation algorithm is the following. In addition to the consideration of the video popularity and the bandwidth distribution of different users, the representation selections for different videos can be further adapted by the proposed algorithm according to the video content information. For video sequences with small motion (e.g., *Sunflower*), the proposed algorithm only encodes one basic representation with a relatively small bitrate at the DASH server, while for video sequences with larger motion (e.g., *Crowd Run*), a much greater number of representations with various bitrate allocations are encoded in order to gain larger distortion reduction. For the rate only solution in [4] without the maximum encoding CPU load constraint, almost all the selected representations are encoded with larger search ranges λ such that a smaller encoding bitrate is required for the same distortion reduction but at the cost of a

TABLE II
REPRESENTATION SELECTIONS OF DIFFERENT ALGORITHMS WITH GIVEN $R_{\max} = 30$ MBPS AND $C_{\max} = 3$ THZ.

Algorithm	Video f	Rep. ID m	$\lambda_{f,m}$	$QP_{f,m}$	$R_{f,m}$ (Mbps)	$C_{f,m}$ (THz)
Optimum $\sum R_{f,m} = 29.6$ Mbps $\sum C_{f,m} = 2.99$ THz	*Crowd Run*	1	2	38	6.82	0.247
	Crowd Run	2	6	40	4.76	0.305
	Crowd Run	3	6	42	3.39	0.300
	Crowd Run	4	6	43	2.91	0.297
	Crowd Run	5	6	44	2.42	0.293
	CrowdRun	6	10	45	2.06	0.414
	Tractor	1	10	38	3.19	0.422
	Tractor	2	10	42	2.04	0.415
	Sunflower	1	6	32	2.07	0.299
$(\omega^*, 2)$-WCB greedy $\sum R_{f,m} = 29.9$ Mbps $\sum C_{f,m} = 3.00$ THz	*Crowd Run*	1	6	38	6.48	0.312
	Crowd Run	2	2	40	5.04	0.239
	Crowd Run	3	6	42	3.39	0.300
	Crowd Run	4	6	44	2.42	0.293
	Crowd Run	5	6	45	2.07	0.290
	Crowd Run	6	2	48	1.32	0.213
	Tractor	1	6	37	4.03	0.302
	Tractor	2	10	42	2.04	0.415
	Tractor	3	2	50	1.12	0.215
	Sunflower	1	10	32	2.02	0.418
Power only solution $\sum R_{f,m} = 34.9$ Mbps $\sum C_{f,m} = 2.98$ THz	*Crowd Run*	1	2	38	6.82	0.247
	Crowd Run	2	2	40	5.04	0.239
	Crowd Run	3	6	42	3.39	0.300
	Crowd Run	4	6	43	2.91	0.297
	Crowd Run	5	6	44	2.42	0.293
	Crowd Run	6	6	45	2.07	0.290
	Tractor	1	6	35	5.24	0.308
	Tractor	2	6	40	2.91	0.297
	Tractor	3	10	42	2.04	0.415
	Sunflower	1	6	32	2.07	0.299
Rate only solution $\sum R_{f,m} = 30.0$ Mbps $\sum C_{f,m} = 3.66$ THz	*Crowd Run*	1	10	38	6.45	0.431
	Crowd Run	2	10	41	4.05	0.423
	Crowd Run	3	10	43	2.89	0.418
	Crowd Run	4	6	44	2.42	0.293
	CrowdRun	5	10	45	2.06	0.414
	Tractor	1	10	34	5.43	0.429
	Tractor	2	10	39	2.85	0.421
	Tractor	3	10	42	2.04	0.415
	Sunflower	1	10	33	1.79	0.416
Popularity allocation $\sum R_{f,m} = 21.9$ Mbps $\sum C_{f,m} = 2.82$ THz	*Crowd Run*	1	10	40	4.74	0.426
	Crowd Run	2	10	43	2.89	0.418
	Crowd Run	3	10	45	2.06	0.414
	Tractor	1	10	34	5.43	0.429
	Tractor	2	10	42	2.04	0.415
	Sunflower	1	6	30	2.75	0.304
	Sunflower	2	10	32	2.02	0.418

much larger power consumption. By doing so, the maximum CPU load constraint is violated. Similarly, the power only solution without maximum encoding bitrate constraint allocates more total encoding bitrate than the maximum budget R_{max}.

The algorithm running time is another performance metric which has the same or even greater importance than the average distortion reduction per user. In Fig. 6, we compare the actual running time of the proposed $(\omega^*, 0)$-WCB greedy algorithm and the optimal solution of the ILP **P2** solved by the generic solver IBM ILOG CPLEX [11], and show the impact of the number of representations $F \times M$ and the number of users N on the running time. Through the fitted curves in Fig. 6, the previous theoretical analysis of the computational complexity is well justified. That is, the ILP solution has a very high exponential time complexity $O(2^{F \cdot 2M \cdot 2N})$, while the proposed $(\omega^*, 0)$-WCB greedy algorithm achieves a linear time complexity $O(FMN)$. In other words, the proposed algorithm has a much lower increasing rate and scales better than the ILP solution. Considering a practical video streaming system with a large number of videos, representations and users, the long waiting time for the IBM ILOG CPLEX solver to obtain the optimal representation selection is intolerant and thus infeasible in practice. In contrast, the proposed algorithm is suitable for such delay sensitive applications since it is capable of achieving a near-optimal solution within a short period of time.

Fig. 7(a) illustrates the comparison of the running time versus the number of representations achieved by the proposed algorithm, and the generic solver IBM ILOG CPLEX [11] under different settings of error bounds (relative optimality tolerances). In Fig. 7(b), we accordingly show the algorithm performance after spending the corresponding running time, in terms of the approximation ratio to the optimal solution. The running time of the generic solver can be greatly reduced by enlarging the relative optimality tolerance (from 0.01 to 0.1 in Fig. 7(a)), which comes at the cost of the reduction of approximation ratio (the green curve is generally below the blue curve in Fig. 7(b)). However, the curves of the running time versus the number of representations illustrate that even by setting an error bound, the computational complexity of the generic solver is still exponential. In contrast, the proposed algorithm can achieve a comparable approximation ratio (mostly larger than the green curve in Fig. 7(b)), while the running time is linear with the number of representations and significantly shorter than the generic solver under different error bound settings.

TABLE III
INDICES AND NAMES OF THE TEST VIDEO SEQUENCES.

Index f	1	2	3	4	5	6	7	8
Video	*Crowd Run*	*Tractor*	*Sunflower*	*Aspen*	*Blue Sky*	*Controlled Burn*	*Dinner*	*Ducks Take Off*
Index f	9	10	11	12	13	14	15	
Video	*In To Tree*	*Life*	*Old Town Cross*	*Park Joy*	*Riverbed*	*Station2*	*Touchdown Pass*	

D. Performance Evaluation for Larger System Settings

Finally, we conduct simulations for larger scale settings. In total, $F = 15$ test video sequences with 1080p resolution (1920×1080), available at [31], are selected as the source video streams to be encoded at the DASH server. They correspond to different motion and video types (such as, sport, documentary, cartoon and movie). The indices and names of these video sequences are listed in Table III. The encoding time of each video frame is still limited by $\Delta T = 0.03$ s, and the constant maximal distortion $D_{\max} = 500$. At a frame rate of 30 fps, each video sequence f is encoded into $M = 63$ representations with the coding parameter pair $(\lambda_{f,m}, Q_{f,m}) \in \Lambda \times \mathcal{Q}$, where $\Lambda = \{2, 6, 10\}$ and the corresponding QP value ranges between 30 and 50. For the video popularity, we investigate three different popularity distributions, i.e., the Zipf distribution with parameter 0.96 and 0.56, and the uniform distribution. The number of users is also enlarged to $N = 100$, where each user's download bandwidth B_n is randomly distributed in the range of $[1, 10]$ Mbps.

In Table IV, we compare the average video quality in PSNR obtained by different representation selection algorithms under the three different popularity distributions, when $R_{\max} = 250$ Mbps and $C_{\max} = 25$ THz. Although the system settings scale with a larger number of videos and users, it is again verified that for all popularity distributions the proposed (ω, k)-WCB greedy algorithm outperforms the popularity based allocation algorithm and achieves a higher PSNR value. This PSNR performance is very close to the performance upper bound guided by the optimal solution of the ILP **P2** that is solved by the CPLEX [11], but the actual running time is much shorter. On the other hand, the power only solution without maximum encoding bitrate constraint and the rate only solution in [4] without maximum CPU load constraint would achieve a PSNR value at least no worse than the optimal solution of the ILP **P2**. However, these two schemes are either not rate-efficient or not power-efficient, in the sense that they actually need to consume more bitrate or CPU load resources than the server can afford in order to achieve only slight performance improvement. Therefore, the proposed algorithm is suitable for delay sensitive DASH streaming, since it could strike a tradeoff between the algorithm's performance and running time while satisfying the delay, rate and power constraints at the server.

In practice, the results shown in Fig. 5 and Tables II and IV could further provide some design guidelines for selecting the representations with corresponding encoder parameters, as follows. 1) In a typical delay sensitive streaming scenario, the rate and power (CPU load) allocation among videos is not only dependent on the popularity distribution, but also affected by the video content information. For the same video type, straightforwardly, a larger amount of rate or power budget needs to be allocated for more popular videos. While for different video types, a larger amount of rate or power budget needs to be allocated for videos with larger motion or more complex content. 2) The number of representations and the corresponding encoder parameters per video should also be content-aware: a larger number of representations with more QP configurations needs to be dedicated to videos with larger motion or more complex content. 3) When the rate resource is scarcer than the power resource, a larger search range λ should be selected for each representation in order to reduce the encoding bitrate while achieving the same video distortion but with larger power consumption; and vice versa. Overall, the proposed algorithm complies well with these design guidelines and scales well with the size of the system. Since it could further strike the optimal tradeoff both between the rate and power cost, and between the algorithm's performance in terms of the average distortion reduction per user and the delay requirements, it is therefore useful for practical system design.

TABLE IV
COMPARISON OF AVERAGE VIDEO QUALITY IN PSNR UNDER DIFFERENT POPULARITY DISTRIBUTIONS.

Algorithm	Zipf distribution (parameter 0.96)	Zipf distribution (parameter 0.56)	Uniform
Optimum	30.34	31.07	31.65
$(\omega^*, 1)$-WCB greedy	30.23	30.93	31.49
$(\omega^*, 0)$-WCB greedy	30.21	30.91	31.46
Power only solution	30.34	31.07	31.65
Rate only solution	30.37	31.11	31.68
Popularity allocation	29.87	30.63	31.15

VII. CONCLUSION

This paper has studied an encoding delay, rate and power constrained representation selection problem for delay sensitive DASH streaming in order to maximize the expected aggregate video distortion reduction. Based on this optimization problem, we have provided an ILP formulation to achieve the performance upper bound but with exponential time complexity, and an equivalent constrained submodular maximization that is used to develop an approximate algorithm with polynomial time complexity. Simulation results have justified that the proposed weighted rate and power cost benefit greedy algorithm could achieve a near-optimal performance without introducing a long additional computation delay, which is therefore suitable for delay sensitive video streaming. Our future work will study the online adaptation algorithms for dynamic resource provisioning in the server-side representation selection when taking into account the dynamics of networks and users, and the power consumption of the mobile devices [33] while transcoding the received DASH streams to support device-to-device communication.

REFERENCES

[1] T. Stockhammer, "Dynamic adaptive streaming over HTTP: standards and design principles," in *Proc. ACM MMSys*, 2011, pp. 133–144.

[2] R. Aparicio-Pardo, K. Pires, A. Blanc, and G. Simon, "Transcoding live adaptive video streams at a massive scale in the cloud," in *Proc. ACM MMSys*, 2015, pp. 49–60.

[3] L. Su, Y. Lu, F. Wu, S. Li, and W. Gao, "Complexity-constrained H. 264 video encoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 4, pp. 477–490, Apr. 2009.

[4] L. Toni, R. Aparicio-Pardo, K. Pires, G. Simon, A. Blanc, and P. Frossard, "Optimal selection of adaptive streaming representations," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 11, no. 2s, p. 43, Feb. 2015.

[5] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, "An evaluation of bitrate adaptation methods for HTTP live streaming," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 693–705, Apr. 2014.

[6] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran, “Probe and adapt: Rate adaptation for HTTP video streaming at scale,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, Apr. 2014.

[7] Z. Li, A. C. Begen, J. Gahm, Y. Shan, B. Osler, and D. Oran, “Streaming video over HTTP with consistent quality,” in *Proc. ACM MMSys*, 2014, pp. 248–258.

[8] W. Zhang, Y. Wen, Z. Chen, and A. Khisti, “QoE-driven cache management for HTTP adaptive bit rate streaming over wireless networks,” *IEEE Transactions on Multimedia*, vol. 15, no. 6, pp. 1431–1445, Oct. 2013.

[9] Y. Jin, Y. Wen, and C. Westphal, “Optimal transcoding and caching for adaptive streaming in media cloud: An analytical approach,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, pp. 1914–1925, Dec. 2015.

[10] C. Kreuzberger, B. Rainer, H. Hellwagner, L. Toni, and P. Frossard, “A comparative study of DASH representation sets using real user characteristics,” in *Proc. ACM NOSSDAV*, 2016.

[11] IBM, “ILOG CPLEX optimization studio.” [Online]. Available: http://is.gd/3GGOFp

[12] Y. Wen, X. Zhu, J. J. Rodrigues, and C. W. Chen, “Cloud mobile media: Reflections and outlook,” *IEEE Transactions on Multimedia*, vol. 16, no. 4, pp. 885–902, Jun. 2014.

[13] W. Zhang, Y. Wen, J. Cai, and D. Wu, “Toward transcoding as a service in a multimedia cloud: Energy-efficient job-dispatching algorithm,” *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2002–2012, Jun. 2014.

[14] G. Gao, Y. Wen, and C. Westphal, “Dynamic resource provisioning with QoS guarantee for video transcoding in online video sharing service,” in *Proc. ACM Multimedia Conference*, 2016, pp. 868–877.

[15] X. Li, N. Oertel, A. Hutter, and A. Kaup, “Laplace distribution based Lagrangian rate distortion optimization for hybrid video coding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 2, pp. 193–205, Feb. 2009.

[16] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, “Power-rate-distortion analysis for wireless video communication under energy constraints,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 5, pp. 645–658, May 2005.

[17] Q. Chen and D. Wu, “Delay-rate-distortion model for real-time video communication,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 8, pp. 1376–1394, Aug. 2015.

[18] C. Li, D. Wu, and H. Xiong, “Delay-power-rate-distortion model for wireless video communication under delay and energy constraints,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 7, pp. 1170–1183, Jul. 2014.

[19] C. Li, H. Xiong, and D. Wu, “Delay–rate–distortion optimized rate control for end-to-end video communication over wireless channels,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 10, pp. 1665–1681, Oct. 2015.

[20] J. Wu, B. Cheng, C. Yuen, N.-M. Cheung, and J. Chen, “Trading delay for distortion in one-way video communication over the internet,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 711–723, Apr. 2016.

[21] Z. Chen and D. Wu, “Rate-distortion optimized cross-layer rate control in wireless video communication,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 3, pp. 352–365, Mar. 2012.

[22] H. M. Hang and J. J. Chen, “Source model for transform video coder and its application. I. Fundamental theory,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 287–298, Apr. 1997.

[23] R. Min, T. Furrer, and A. Chandrakasan, “Dynamic voltage scaling techniques for distributed microsensor networks,” in *Proc. IEEE Computer Society Workshop on VLSI*, 2000, pp. 43–46.

[24] J. R. Lorch, A. J. Smith *et al.*, “Improving dynamic voltage scaling algorithms with PACE,” *ACM SIGMETRICS Performance Evaluation Review*, vol. 29, no. 1, pp. 50–61, Jun. 2001.

[25] T. D. Burd and R. W. Brodersen, “Processor design for portable systems,” *The Journal of VLSI Signal Processing*, vol. 13, no. 2, pp. 203–221, 1996.

[26] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, “An analysis of approximations for maximizing submodular set functions-I,” *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[27] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, “An analysis of approximations for maximizing submodular set functions-II,” *Mathematical Programming Studies*, vol. 8, pp. 73–87, 1978.

[28] P. R. Goundan and A. S. Schulz, “Revisiting the greedy approach to submodular set function maximization,” *Optimization online*, pp. 1–25, 2007.

[29] A. Krause and D. Golovin, “Submodular function maximization,” *Tractability: Practical Approaches to Hard Problems*, vol. 3, p. 19, 2012.

[30] M. Sviridenko, “A note on maximizing a submodular set function subject to a knapsack constraint,” *Operations Research Letters*, vol. 32, no. 1, pp. 41–43, 2004.

[31] “Xiph.org video test media.” [Online]. Available: http://media.xiph.org/video/derf/

[32] M. Zink, K. Suh, Y. Gu, and J. Kurose, “Characteristics of youtube network traffic at a campus network–measurements, models, and implications,” *Computer Networks*, vol. 53, no. 4, pp. 501–514, 2009.

[33] J. Wu, C. Yuen, B. Cheng, M. Wang, and J. Chen, “Energy-minimized multipath video transport to mobile devices in heterogeneous wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1160–1178, May 2016.