On-Line Learning and Optimization for Wireless Video Transmission

Yu Zhang, *Student Member, IEEE*, Fangwen Fu, *Student Member, IEEE*, and Mihaela van der Schaar, *Fellow, IEEE*

***Abstract*—In this paper, we address the problem of how to optimize the cross-layer transmission policy for delay-sensitive video streaming over slow-varying flat-fading wireless channels on-line, at transmission time, when the environment dynamics are unknown. We first formulate the cross-layer optimization using a systematic layered Markov decision process (MDP) framework, which complies with the layered architecture of the OSI stack. Subsequently, considering the unknown dynamics of the video sources and underlying wireless channels, we propose a layered real-time dynamic programming (LRTDP) algorithm, which requires no *a priori* knowledge about the source and network dynamics. LRTDP allows each layer to *learn* the dynamics on-the-fly, and adjusts its policy autonomously, based on their experienced dynamics as well as limited message exchanges with other layers. Unlike existing cross-layer methods, LRTDP optimizes the cross-layer policy in a layered and on-line fashion, exhibits a low computational complexity, requires limited message exchanges among layers, and is capable to adapt on-the-fly to the experienced environment dynamics. Finally, we prove that LRTDP converges to the optimal cross-layer policy asymptotically. Our numerical experiments show that LRTDP provides comparable performance to the idealized optimal cross-layer solutions based on complete knowledge.**

***Index Terms*—Layered Markov decision process, layered real-time dynamic programming, on-line learning, wireless video transmission.**

I. Introduction

VIDEO transmission over error-prone wireless networks is challenging due to a number of factors, including the high bit-rate requirements and hard delay constraints imposed by the video traffic, as well as the time-varying environmental dynamics experienced by wireless users (e.g., video source characteristics, wireless channel conditions, end-user experience and requirements, etc. [1], [2]). This paper focuses on determining the optimal cross-layer transmission policy for an individual wireless user (i.e., a transmitter-receiver pair) streaming video traffic over a single-hop wireless network in a dynamic, unknown environment. Specifically, we consider how, depending on the experienced time-varying environment, we can optimally tradeoff the received video quality and transmission energy for real-time video transmission, by judiciously and jointly adapting the transmission strategies at various layers of the protocol stack based on the experienced wireless environment. The cross-layer transmission strategies can include[1] adaptive power control [35] as well as modulation and coding schemes [21] at the physical (PHY) layer, automatic repeated request [22] at the media access control (MAC) layer, and priority scheduling [5] at the application (APP) layer.

Cross-layer optimization has been proposed to solve the abovementioned problem [2]–[5], which can significantly improve the user's experienced performance by jointly optimizing the protocol parameters across various layers of the protocol stack. However, most existing cross-layer algorithms still exhibit several limitations.

Centralized: Most of the algorithms optimize the cross-layer transmission strategies in a centralized fashion, as discussed in [1] and [23]. A middleware or system-level monitor serves as the centralized optimizer, which estimates the resource availability and environmental dynamics, optimizes the cross-layer strategies and implements the optimal strategies for real-time data transmission, which requires to access each layer's internal protocols and private data. Such approaches can determine global optimal cross-layer policy. However, they violate the layered network architecture, as individual layers lose their ability to control their own protocols and algorithms. Moreover, such cross-layer optimization solutions have very high complexity and require heavy message exchanges (i.e., communication overhead) among the participating layers.

Myopic: Another common limitation of existing cross-layer algorithms is that they focus on maximizing the instant utility, without considering the impact of the user's current action on its long-term performance [24], [25]. In wireless multimedia applications, such myopic strategy design can result in unacceptable deterioration in long-term multimedia quality due to the heterogeneous characteristics of the media traffic. Therefore, cross-layer strategies need to be optimized in a foresighted way by considering the effect of current actions on the future performance.

Off-Line Optimization: The cross-layer optimization problem was formulated within the framework of discrete-time MDP proposed in [6], in which the wireless user no longer passively adapts the transmission policy to its current experienced dynamics; instead, each layer actively selects actions to account for, as well as influence, the future dynamics it will experience in order to achieve optimal long-term performance over time, even if this requires sacrificing immediate benefits. At each layer, conventional off-line dynamic programming

Manuscript received July 06, 2009; accepted January 24, 2010. Date of publication March 18, 2010; date of current version May 14, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Andrea Cavallaro.

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

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

Digital Object Identifier 10.1109/TSP.2010.2046040

[1]Since we consider delay-sensitive video transmission, we assume that UDP was used at the transport layer.

(DP) methods (e.g., value iteration, policy iteration, etc.) are employed to autonomously determine the optimal actions by exchanging only limited information with other layers. However, these DP algorithms iteratively search the optimal transmission policy by fully sweeping the entire state space of each layer. The optimal transmission policy is able to be deployed only after the MDP problem has been solved, which often incurs unacceptable delay and is unsuitable for delay-sensitive video transmission. Moreover, and even more importantly, the dynamic wireless environment, such as the time-varying traffic characteristic and channel conditions are often difficult to characterize *a priori*, before transmission time. In this case, the off-line DP methods cannot be performed, due to their requirements on complete *a priori* knowledge of the environment dynamics [2], and are therefore impractical for real-time applications, such as wireless video streaming.

To address the above problems, we propose an on-line approach called layered real-time dynamic programming (LRTDP) to solve the MDP-based cross-layer optimization problem, which have the following characteristics:

1) Layered: LRTDP computes the cross-layer transmission policy in a layered fashion, with the objective function optimized by different layers in a distributed manner;
2) Foresight: LRTDP maximizes the user's long-term utility, instead of the one-shot reward;
3) On-line: LRTDP takes advantage of indirect on-line learning method [13] to autonomously learn the experienced dynamics and adjusts the cross-layer transmission policy in real-time;
4) Limited information: LRTDP does not require any *priori*i knowledge of the system and environment.
5) Low complexity: LRTDP has low complexity and limited inter-layer message exchanges, which decrease the operation delay and is thus important for real-time applications.

Summarizing, we make the following contributions in this paper.

- We propose a novel layered cross-layer learning algorithm for wireless video transmission.
- We prove that LRTDP converges to the optimal cross-layer policy that maximizes the user's long-term utility during the video streaming process.
- We employ extensive experiments to compare LRTDP with alternative cross-layer methods, and show that LRTDP delivers similar or better performance with lower operation cost.

The remainder of this paper is organized as follows. In Section II, we introduce the considered problem of video transmission over a single wireless link and formulate it as a formal MDP-based cross-layer optimization problem. In Section III, we present briefly the layered off-line DP method and discuss our proposed LRTDP method under the layered MDP framework. Section IV presents the experiment results and Section V concludes the paper.

II. System Model and Cross-Layer Problem Formulation

This paper considers real-time video transmission over a single-hop slow-varying flat fading channel. In this section, we model the wireless user as a system where three layers (i.e., APP, MAC, PHY) participate in the cross-layer optimization, as depicted in Fig. 1. The specific details of the illustrative model adopted in this paper can be found in Appendix A, but the methodology proposed in this paper can be also adopted for other models at the various layers of the OSI stack. (This is why we provide the model's specific details within the Appendix.) For clarity, we use subscripts APP, MAC, and PHY to represent quantities related to the corresponding layers, respectively. We focus on the transmission strategy adaptation at these three layers[2] to optimize the received video quality under transmission energy constraints. As mentioned in [6], this system can be modeled as an MDP defined across the layers. For better illustration, we assume the system is time-slotted, with the slot length of ΔT, as in [3], [32], and [33]. The wireless user makes decision at the beginning of every time slot.

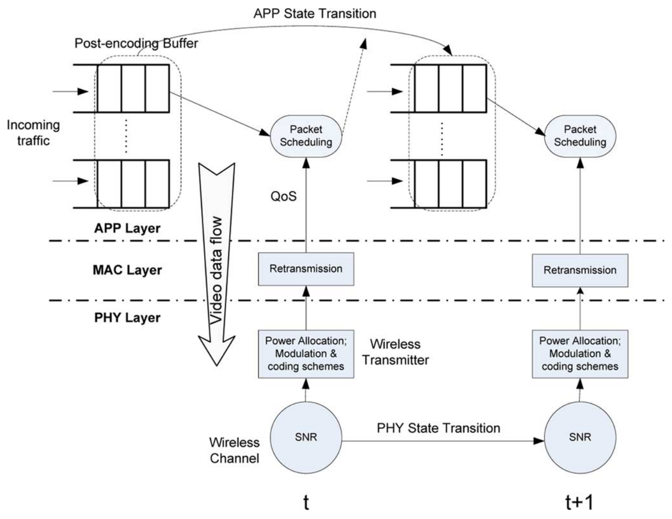


Fig. 1. Real-time video streaming process.

A. PHY Layer Model

The wireless channel experienced by the user at PHY is modeled as a discrete-time Rayleigh-fading additive white Gaussian noise channel [9]. We assume that the channel coherence time T_{coh} is larger than ΔT [3], [4] such that the signal-to-noise ratio (SNR) is constant within each time slot. The SNR at each time slot is defined as PHY's state s_{PHY} to represent the channel condition.

To enable our cross-layer framework to comply with the layered architecture of protocol stack, we divide the processing actions at each layer into two categories [6]: the external action a which controls the state transition at this layer, and the internal action b determining the QoS provided to the upper layers. PHY's external action $a_{\mathrm{PHY}} \in A_{\mathrm{PHY}}$ is the power allocation at the current time slot, where A_{PHY} is the set of applicable power allocation schemes. Several works [9], [31] have proposed finite-state Markov chain (FSMC) to model the transition of channel SNR across time slots. In this paper, we further extend this Markovian SNR model into an MDP, in which the state transition at any time slot t is determined

[2]Since we focus on the single-hop wireless transmission, we do not need to consider the routing at the network layer and congestion control at the transport layer.

not only by the current state s^t_{PHY}, but also by the current power allocation a^t_{PHY} with transition probability specified as $p_{\mathrm{PHY}}(s^{t+1}_{\mathrm{PHY}}|s^t_{\mathrm{PHY}}, a^t_{\mathrm{PHY}})$. The details of this transition probability structure can be found in Section A of Appendix A. Given the channel state, the wireless user is able to adapt its modulation and channel coding scheme, which is taken as PHY's internal action $b_{\mathrm{PHY}} \in B_{\mathrm{PHY}}$, where B_{PHY} denotes the set of applicable modulation and channel coding schemes.

PHY's quality of service (QoS) Z_{PHY} describes services that PHY can provide to the upper layers. It is jointly determined by s_{PHY} and b_{PHY}, and are comprised of three elements: i) the packet error rate e_{PHY}; ii) the data throughput v_{PHY}; and iii) the cost of transmitting one packet ω_{PHY}.

B. MAC Layer Model

At MAC, the channel access is based on time division multiplexing access (TDMA), which is commonly used in 802.11a PCF and 802.11e HCCA [5]. In TDMA-based channel access, MAC requests spectrum access through some polling-based mechanism, in which the available transmission time within one time slot is divided by some Central Spectrum Moderator (CSM) among competing wireless users [18]. The allocation scheme deployed by CSM can be (i) a static allocation, where the CSM is polling the various wireless users for a fixed fraction of every time slot, based on the pre-negotiated traffic specification and resource requirement of users [18]; or (ii) a dynamic allocation, where the time allocation to each user changes within every time slot, based on the time-varying channel condition, quality and resource requirements of users [19]. To keep our analysis simple due to space limitations, we assume that a static allocation scheme is deployed, i.e., the amount of transmission time allocated to each user within each time slot is predetermined and fixed based on *a priori* negotiation methods such as those described in [5]. Therefore, MAC's state s_{MAC}, defined as the transmission opportunity (TXOP) duration t_{TXOP} within each time slot, remains constant. It should be noted, however, that with only simple extensions, our results can also work in dynamic resource allocation scenarios.

MAC's external action a_{MAC} determines t_{TXOP} for each wireless user. This action can be the resource requirement or the price which the user is willing to pay for the spectrum resources. Under the static allocation scheme, the wireless user does not have to negotiate and compete for the resources with other users and hence, a_{MAC} is non-adaptive, similarly to the state s_{MAC}. During its TXOP, the user can perform error control algorithms such as Automatic Repeat-reQuest (ARQ) to improve the QoS provided to the upper layers, similar to those used in 802.11 wireless networks [5]. MAC's internal action b_{MAC} is defined as the user's retransmission limit. The QoS Z_{MAC} at MAC is thus jointly determined by $s_{\mathrm{MAC}}, b_{\mathrm{MAC}}$, and the QoS level Z_{PHY} from PHY, with details can be found in Section B of Appendix A.

C. APP Layer Model

At APP, the transmitter receives video data from the encoder and schedules buffered packets deciding which of them should be transmitted. As in [30], APP's state s_{APP} within each time slot is characterized by the amount of incoming traffic s_{TRA} and the amount of buffered packets s_{BUF}, with state space correspondingly represented as $S_{\mathrm{APP}} = S_{\mathrm{TRA}} \times S_{\mathrm{BUF}}$. The external action $a_{\mathrm{APP}} \in A_{\mathrm{APP}}$ is defined as the packet scheduling algorithm, and there is no internal action at APP since it is the highest layer in our model and not required to provide QoS for any upper layer.

The incoming traffic depends on the characteristics of the transmitted video sequence, as well as the source coding algorithm which is assumed to be fixed here. As shown in [8], the temporal correlation of the traffic can be captured by an FSMC, with state transition probability $p_{\mathrm{TRA}}(s^{t+1}_{\mathrm{TRA}}|s^t_{\mathrm{TRA}})$. The transition of the output buffer state s_{BUF} is jointly determined by the buffer occupancy, the incoming traffic, and the packet scheduling, with transition probability $p_{\mathrm{BUF}}(s^{t+1}_{\mathrm{BUF}}|s^t_{\mathrm{BUF}}, s^t_{\mathrm{TRA}}, a^t_{\mathrm{APP}})$. APP's state transition is consequently given as: $p_{\mathrm{APP}}(s^{t+1}_{\mathrm{APP}}|s^t_{\mathrm{APP}}, a^t_{\mathrm{APP}}) = p_{\mathrm{TRA}}p_{\mathrm{BUF}}$. The detailed structure of the transition probability can be found in Section C of Appendix A.

D. System Utility Function

With the above model, we can define the state of the system as $\boldsymbol{s} = (s_{\mathrm{APP}}, s_{\mathrm{MAC}}, s_{\mathrm{PHY}})$, which can be further simplified as $\boldsymbol{s} = (s_{\mathrm{APP}}, s_{\mathrm{PHY}})$, since MAC's state is fixed and will not affect the optimization of the transmission policy. Similarly, the external and internal actions of the system are $\boldsymbol{a} = (a_{\mathrm{APP}}, a_{\mathrm{PHY}})$ and $\boldsymbol{b} = (b_{\mathrm{MAC}}, b_{\mathrm{PHY}})$, respectively.

Since each layer has been modeled as an FSMC or MDP, it is obvious that the system is also an MDP, with the state transition probability given by: $p(\boldsymbol{s}^{t+1}|\boldsymbol{s}^t, \boldsymbol{a}^t) = p_{\mathrm{APP}}p_{\mathrm{PHY}}$.

At each time slot, the video quality is determined by $s_{\mathrm{APP}}, a_{\mathrm{APP}}, Z_{\mathrm{MAC}}$ and denoted by $q(s_{\mathrm{APP}}, a_{\mathrm{APP}}, Z_{\mathrm{MAC}})$. At the same time, performing external actions at each layer l incurs a cost[3] $c_l(s_l, a_l)$, and the internal action cost has already been considered in $\omega_l(s_l, b_l, Z_{l-1})$. Therefore, we use a weighted sum of video quality and operation cost to define the overall system utility at each time slot, as

$$U^t(\boldsymbol{s}^t, \boldsymbol{a}^t, \boldsymbol{b}^t) = q^t\left(s^t_{\mathrm{APP}}, a^t_{\mathrm{APP}}, Z^t_{\mathrm{MAC}}\right) - \sum_l \lambda^a_l c^t_l\left(s^t_l, a^t_l\right) - \lambda^b \omega^t_{\mathrm{APP}}\left(s^t_{\mathrm{APP}}, Z^t_{\mathrm{MAC}}\right) \quad (1)$$

where λ^a_l and λ^b are positive Lagrangian parameters to trade off the video quality and operation cost[4]. They are selected based on energy constraints [29]. Given that MAC is a fixed layer in this paper, whose state and external actions remain constant, (1) can be further simplified as

$$\begin{aligned} U^t(\boldsymbol{s}^t, \boldsymbol{a}^t, \boldsymbol{b}^t) = q^t&\left(s^t_{\mathrm{APP}}, a^t_{\mathrm{APP}}, Z^t_{\mathrm{MAC}}\right) \\ &- \lambda^b \omega^t_{\mathrm{APP}}\left(s^t_{\mathrm{APP}}, Z^t_{\mathrm{MAC}}\right) \\ &- \lambda^a_{\mathrm{PHY}} c^t_{\mathrm{PHY}}\left(s^t_{\mathrm{PHY}}, a^t_{\mathrm{PHY}}\right) \\ &- \lambda^a_{\mathrm{APP}} c^t_{\mathrm{APP}}\left(s^t_{\mathrm{APP}}, a^t_{\mathrm{APP}}\right). \end{aligned} \quad (2)$$

[3]In this paper, the cost represents the resource consumed by performing the external actions (e.g., RF power consumed during data transmission at PHY, competition bids at MAC, and etc.)

[4]The Lagrangian parameters can be determined based on the resource budgets available for the wireless user [36] or by the network coordinator to efficiently utilize the network resources [37]. In our manuscript, the optimal Lagrangian multipliers have been numerically determined using the bisection method and we focus on the internal and external action selections. We could also use stochastic subgradient-based resource price update as in [38] to iteratively update them.

Considering the impact of current actions on the future system evolution, the wireless user aims to find the optimal actions such that the cumulative long-term system utility is maximized, i.e.

$$V(\boldsymbol{s}) = \mathbb{E}_\pi \left\{ \sum_{t=0}^{\infty} \mu^t U^t(\boldsymbol{s}^t, \boldsymbol{a}^t, \boldsymbol{b}^t) \right\} \tag{3}$$

where π is the transmission policy, $\boldsymbol{s}^0 = \boldsymbol{s}$ and μ is a discount factor within [0, 1). When $\mu = 0$, the problem becomes a optimal myopic decision problem, where the user only considers maximizing the immediate utility received within the current time slot. Note that μ should be less than 1 because: i) for delay-sensitive video applications, the data needs to be sent out as soon as possible to avoid missing delay deadlines and hence, utilities closer to "present" are considered to be more important; ii) the undiscounted sum of utilities is not guaranteed to be finite and to converge if $\mu = 1$ [12].

III. Real-Time DP Solution

A. Layered Synchronous DP Solution

The optimal cross-layer policy for video transmission formulated as an MDP in Section II can be found using centralized dynamic programming methods (e.g., value iteration, policy iteration, etc. [12]) to maximize the discounted long-term utility that can be received starting from any state $\boldsymbol{s}$

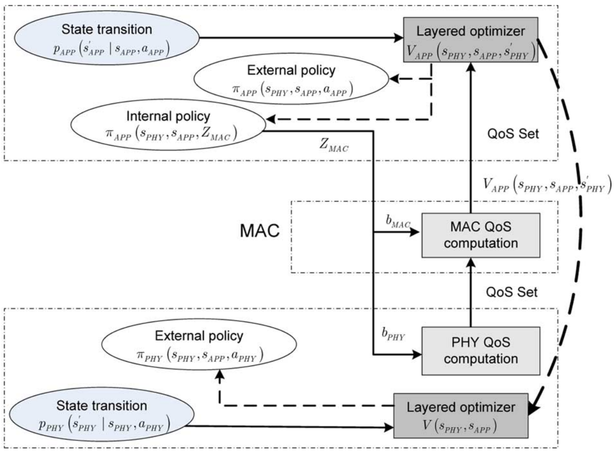
$$\max_{a,b} \left\{ U(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{b}) + \mu \sum_{s' \in \mathcal{S}} p(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a}) V(\boldsymbol{s}') \right\}. \tag{4}$$

However, such centralized methods require layers to share information about their states, actions, utilities, and dynamics, which not only violates the layered network architecture, but also leads to large message exchange overhead. To adhere to the current layered network architecture, we propose a layered decomposition of the DP operator (4) similar to [6], to allow each layer to update its own external and internal policies independently, according to the information exchanged with other layers. For the streaming application, the decomposition is shown in Appendix B, where we further prove its equivalence to the centralized approach. In the rest of this paper, this layered optimization method is referred to as layered synchronous dynamic programming (LSDP), since the state-value functions of all states are updated synchronously within each round of its iterations.

The operation and message exchange of LSDP within each time slot is summarized in Fig. 2.

B. Why Is a Real-Time Solution Needed

LSDP provides a systematic way to solve the cross-layer optimization problem and guarantees an optimal solution. It iteratively optimizes APP's scheduling policy and PHY's transmission policy *off-line*, that is, the optimal cross-layer transmission policy is obtained before the real data is transmitted. This feature introduces additional latency which may not suitable for the real-time video transmission. Furthermore, in real-time video transmission system, the experienced environments are often changed drastically over time which cannot be characterized *a priori* and, hence, impedes the implementation of LSDP. The disadvantages of LSDP are summarized below.



APP
State transition
$p_{APP}(s'_{APP} \mid s_{APP}, a_{APP})$
Layered optimizer
$V_{APP}(s_{PHY}, s_{APP}, s'_{PHY})$
External policy
$\pi_{APP}(s_{PHY}, s_{APP}, a_{APP})$
Internal policy
$\pi_{APP}(s_{PHY}, s_{APP}, Z_{MAC})$
Z_{MAC}
QoS Set
$V_{APP}(s_{PHY}, s_{APP}, s'_{PHY})$
MAC
b_{MAC}
MAC QoS computation
QoS Set
PHY
External policy
$\pi_{PHY}(s_{PHY}, s_{APP}, a_{PHY})$
b_{PHY}
PHY QoS computation
State transition
$p_{PHY}(s'_{PHY} \mid s_{PHY}, a_{PHY})$
Layered optimizer
$V(s_{PHY}, s_{APP})$

Fig. 2. The operation and message exchange in LSDP.

First, LSDP requires perfect knowledge of the environment as well as the response of the considered video transmission system to environmental changes. In our problem, this knowledge includes the transition probability of both the video traffic at the APP layer and the wireless channel condition at PHY layer. In real-time video transmission systems, it is often difficult to obtain this knowledge because: i) it is costly to build up the transition probability profile sequence, e.g., using repeated training or Monte Carlo simulations [8]; ii) different video sequence characteristics (e.g., high motion and low motion) will lead to different statistical properties and, hence, the statistic model learned from one video sequence cannot be applied to another sequence [8].

Second, the complexity of the offline computation is $O(|S_{\mathrm{PHY}}|^2 |S_{\mathrm{APP}}|\,|A_{\mathrm{APP}}|\,|A_{\mathrm{PHY}}|\,|B_{\mathrm{MAC}}|\,|B_{\mathrm{PHY}}|)$. In video transmission, the cardinality of $|S_{\mathrm{APP}}|$ is large, usually dominant to all the other terms, which will result in long operation latency.

Finally and most importantly, LSDP cannot adapt its cross-layer transmission policy on-the-fly to the system and environment dynamics, which is important for delay-sensitive applications. In real-time video transmission, any part of the system and the environment may vary over time (e.g., the source traffic, the wireless channel condition, the end user's experience and interaction). As LSDP is run off-line, the knowledge its computation relies on (i.e., the utility profile, the statistic model for the system) is stationary, which is often not true in practice. The performance of LSDP will severely degrade whenever the environment model is inaccurate.

C. Why is a Real-Time Solution Needed

To overcome the shortcomings of LSDP, we propose a novel LRTDP method, which is performed *on-line* in order to adapt the cross-layer transmission strategies to the unknown dynamics in both the source traffic and experienced wireless channel. In LRTDP, each layer learns from its own experience with on-line estimation and cooperatively adapts to the system dynamics to maximize the user's utility in a time-varying environment. The on-line adaptation makes LRTDP more realistic to be

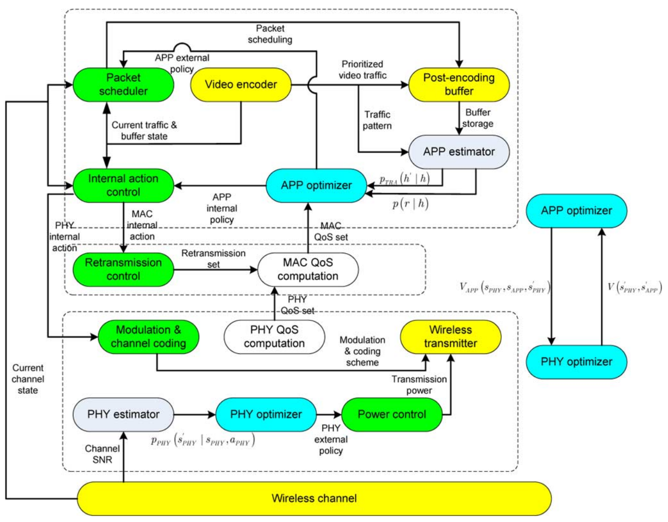


Fig. 3. The system diagram of our streaming system.

implemented in cross-layer optimization of delay-sensitive video streaming applications, such that the delay constrained data can be delivered in time.

The key idea of LRTDP is to incrementally improve the optimal scheduling and transmission policies on-line during the data transmission. This is implemented by intertwining the processes of on-line estimation, policy update, and decision making together. The complete system diagram of LRTDP is shown in Fig. 3, in which the yellow parts are the modules on video data path, the gray parts are on-line estimation modules, the white parts are QoS computation modules, the blue parts are policy update modules, and the green parts are decision making and control modules. The working procedure of LRTDP is summarized in Table V and its specific details are summarized in the following sections.

1) On-Line Estimation: The first operation within time slot t is to update the user's knowledge about system dynamics using the observation history. In the transmission process, it is easy to learn the structure of the instant utility received by the user through some training video sequence, and therefore we only need to estimate on the state transition probabilities which capture the system dynamics. We use system identification methods which are usually referred to as "indirect methods" as compared to "direct methods" like reinforcement learning [13].

In our considered cross-layer problem, there are three categories of unknown distributions which influence the state transition probability: the transition probability of the incoming video traffic $p_{\mathrm{TRA}}(h'|h)$, the distribution on the number of incoming packets $p(r|h)$, and the channel transition probability $p_{\mathrm{PHY}}(s'_{\mathrm{PHY}}|s_{\mathrm{PHY}}, a_{\mathrm{PHY}})$. These distributions have different expressions, but the rules of updating them are identical. Hence, we use uniform variables s and a to denote the state and action, and $p(s'|s,a)$ as the distribution. At the beginning of the streaming process, there is zero knowledge about $p(s'|s,a)$ and hence, the distribution is initialized to be uniform as $p^1(s'|s,a) = 1/|S|$ where $|S|$ is the cardinality of the set S. As the process evolves, the state-visiting history accumulates and we can approximate the state transition probability based on the visiting history. Let $n^t(s,a)$ denote the number of visits to state s with action a performed up to time slot $t-1$, and $n^t(s'|s,a)$ be the number of subsequent transition to state s', then the one-step transition probability $p^t(s'|s,a)$ is updated as

$$p^t(s'|s,a) = \frac{n^t(s'|s,a)}{n^t(s,a)}. \tag{5}$$

The law of large numbers ensures that the estimation $p^t(s'|s,a)$ converges to the true value $p(s'|s,a)$ as long as the state space is communicating (i.e., each state can be accessed from any other state through a finite path) and the whole streaming process is ergodic. In the later part, we will show that the convergence of the transition probability estimation is a sufficient condition of LRTDP's convergence.

2) Policy Update: In [12], the partial ordering over different policies in an MDP has been defined through the state-value function as

$$\begin{aligned}
&\pi \geq \pi' \\
&\text{iff } \mathbb{E}_{\pi}\left\{\sum_{t=0}^{\infty} \mu^t U^t\left(\boldsymbol{s}^t, \boldsymbol{a}^t, \boldsymbol{b}^t\right)\right\} \\
&\quad \geq \mathbb{E}_{\pi'}\left\{\sum_{t=0}^{\infty} \mu^t U^t(\boldsymbol{s}^t, \boldsymbol{a}^t, \boldsymbol{b}^t)\right\}, \qquad \text{for all } \boldsymbol{s} \in S.
\end{aligned} \tag{6}$$

It has been further shown in [12] that there is always at least one optimal policy that is better or equal to all other policies. The state-value function corresponding to the optimal policy is called the optimal state-value function, denoted as $V^*(\boldsymbol{s})$. It is easy to show that $V^*(\boldsymbol{s})$ is unique and any policy which achieves $V^*(\boldsymbol{s})$ is the optimal policy. Therefore, once the state-value function $V(\boldsymbol{s})$ converges to $V^*(\boldsymbol{s})$, the policy corresponding to $V(\boldsymbol{s})$ also approaches to one of the optimal policies.

Similar to the state transition probability, the wireless user has no knowledge about $V(\boldsymbol{s})$ and must update it on-line, during the streaming process. Let $\boldsymbol{s}^t = (s^t_{\mathrm{PHY}}, s^t_{\mathrm{APP}})$ and $V^t(\boldsymbol{s}^t)$ denote the system state and the corresponding latest estimate of state-value function at time slot t, respectively, LRTDP also uses the idea of value iteration to optimize $V(\boldsymbol{s})$ by solving the optimization problem

$$\max_{a,b}\left\{U(\boldsymbol{s},\boldsymbol{a},\boldsymbol{b}) + \mu \sum_{\boldsymbol{s}'\in S} p^t(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a})V(\boldsymbol{s}')\right\}. \tag{7}$$

It should be noticed that LRTDP only has the estimate of transition probability and, hence, $p(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a})$ should be replaced by $p^t(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a})$ during implementation. Using the same decomposition rule as in (29) and (30), LRTDP also decomposes (7) and distributes the optimization task to all layers, as shown in Table I. The operation of the policy update in LRTDP within one time slot (i.e., starting from time t and ending at time $t+1$) is as follows, which is also shown in Fig. 3:

Step 1. PHY computes its QoS set $\mathcal{Z}^t_{\mathrm{PHY}}(s^t_{\mathrm{PHY}})$ (i.e., the set of possible QoS which it can provide to the upper layers) as in Section II-A based on its current state s^t_{PHY} and its internal action set, then transmits it upwards to MAC.

Step 2. MAC computes its QoS set $\mathcal{Z}^t_{\mathrm{MAC}}(s^t_{\mathrm{PHY}})$ as (24), and further forwards it to APP.

TABLE I
THE SUBVALUE FUNCTION (I.E., DP OPERATOR) AT EACH LAYER IN LRTDP

	LRTDP operator
APP	$V_{APP}^{t}\left(s_{PHY}^{t}, s_{APP}^{t}, a_{PHY}\right) = \max\limits_{\substack{a_{APP}\in A_{APP}\\ Z_{MAC}\in \mathcal{Z}_{MAC}}} \left[\begin{array}{l} q\left(s_{APP}^{t}, a_{APP}, Z_{MAC}\right) - \lambda^{b}\omega_{APP}\left(s_{APP}^{t}, Z_{MAC}\right) - \lambda_{APP}^{a} c_{APP}\left(s_{APP}, a_{APP}\right) \\ +\mu \sum\limits_{\substack{s_{PHY}^{'}\in S_{PHY}\\ s_{APP}^{'}\in S_{APP}}} p_{PHY}^{t}\left(s_{PHY}^{'} \mid s_{PHY}^{t}, a_{PHY}\right) p_{APP}^{t}\left(s_{APP}^{'} \mid s_{APP}^{t}, a_{APP}\right) V^{t}\left(s_{PHY}^{'}, s_{APP}^{'}\right) \end{array}\right]$
PHY	$V^{t+1}\left(s_{PHY}^{t}, s_{APP}^{t}\right) = \max\limits_{a_{PHY}\in A_{PHY}} \left[-\lambda_{PHY}^{a} c_{PHY}\left(s_{PHY}^{t}, a_{PHY}\right) + V_{APP}^{t}\left(s_{PHY}^{t}, s_{APP}^{t}, a_{PHY}\right)\right]$

Step 3. As PHY's action $a_{\rm PHY}$ has not been determined yet, APP has to update its PHY-dependent policy $\pi_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, A_{\rm PHY})$, which is defined as $\{\pi_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, a_{\rm PHY}) | a_{\rm PHY} \in A_{\rm PHY}\}$, using Bellman-backup operations [12] for every possible $a_{\rm PHY}$, where $\pi_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, a_{\rm PHY})$ is the policy assuming that $a_{\rm PHY}$ is executed at PHY at this time slot, and the subvalue function $V_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, a_{\rm PHY})$ is the corresponding expected long-term utility.

Step 4. APP transmits the maximized subvalue function set $V_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, A_{\rm PHY})$, similarly defined as $\{V_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, a_{\rm PHY}) | a_{\rm PHY} \in A_{\rm PHY}\}$, downwards to MAC. Since there is no optimization taking place at MAC, it further passes $V_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, A_{\rm PHY})$ to PHY.

Step 5. PHY optimizes its subvalue function $V^{t+1}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$, and PHY's policy $\pi_{\rm PHY}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}) = a_{\rm PHY}(\boldsymbol{s}_{\rm PHY}^{t}, s_{\rm APP}^{t})$ is updated.

Step 6. The state-value function of $\boldsymbol{s}^{t}$ is updated as $V^{t+1}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$.

Step 7. PHY acknowledges its updated policy $a_{\rm PHY}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$ to APP, and APP updates its policy for the current state as

$$\pi_{\rm APP}^{t}\left(s_{\rm PHY}^{t}, s_{\rm APP}^{t}\right) = \left\{\begin{array}{l} a_{\rm APP}\left(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, a_{\rm PHY}\left(s_{\rm PHY}^{t}, s_{\rm APP}^{t}\right)\right), \\ Z_{\rm MAC}\left(s_{\rm PHY}^{t}, s_{\rm APP}^{t}, a_{\rm PHY}\left(s_{\rm PHY}^{t}, s_{\rm APP}^{t}\right)\right) \end{array}\right\}. \quad (8)$$

3) Decision Making: The final step in LRTDP is to control the streaming process by applying the transmission and scheduling actions $a_{\rm PHY}^{t}$ and $\{a_{\rm APP}^{t}, Z_{\rm MAC}^{t}\}$ according to the latest policy, i.e., decision making.

An intuitive way of decision making is to use the *greedy* strategy, i.e., the user always chooses actions stored in the current policy, which maximize the expected utility starting from $\boldsymbol{s}^{t}$. However, such a greedy strategy for external actions is prone to generating loops on the state-visiting path during the streaming process, i.e., some system states are visited frequently while some other states are not visited at all. As we will show in the next section, it is necessary for every state in the state space to be visited in order to ensure the convergence of the policy updates. Hence, for QoS (i.e., internal actions) $Z_{\rm MAC}^{t}$, we always use the greedy strategy to maximize the long-term utility; yet, for the external action $\boldsymbol{a}^{t} = \{a_{\rm PHY}^{t}, a_{\rm APP}^{t}\}$, we choose the greedy action with a probability of $1-\varepsilon^{t}(\boldsymbol{s}^{t})$, and leave a small probability $\varepsilon^{t}(\boldsymbol{s}^{t})$ for the remaining actions to be randomly picked up. Such a strategy is called an ε-greedy policy. In the following section, we will show that it is necessary for the user to take such a small probability to explore other external actions rather than solely utilizing the greedy one.

The decision making of LRTDP is performed as follows (its processing and information flow can be found in Fig. 3:

Step 1. PHY selects its action $a_{\rm PHY}^{t}$ using the ε-greedy strategy. With probability $1-\varepsilon^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t}) = 1-\varepsilon/n^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$, it selects the greedy action as $a_{\rm PHY}^{t} = \pi_{\rm PHY}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$; with probability $\varepsilon^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$, it selects a random action otherwise. $n^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$ represents the number of visits to $(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$ until time slot t, and ε is a small value called the *exploration factor*.

Step 2. According to $\pi_{\rm APP}^{t}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$, APP selects the greedy QoS $Z_{\rm MAC}^{t} = Z_{\rm MAC}(s_{\rm PHY}^{t}, s_{\rm APP}^{t})$, and uses the ε-greedy strategy to select the external action $a_{\rm APP}^{t}$.

Step 3. APP delivers $Z_{\rm MAC}^{t}$ downwards to MAC and PHY for them selecting their corresponding internal actions $b_{\rm PHY}^{t}$ and $b_{\rm MAC}^{t}$.

4) Complexity and Interlayer Message Exchange: In this section, we discuss the computation complexity and message exchange of LSDP and LRTDP. In order to evaluate the complexity of LSDP, we must first look at the complexity of subvalue iteration at each layer. For a fixed state $(s_{\rm PHY}, s_{\rm APP})$, the subvalue iteration at APP defined in (41) has complexity $O(|S_{\rm PHY}|\,|A_{\rm APP}|\,|Z_{\rm MAC}|)$, which equals to $O(|S_{\rm PHY}|\,|A_{\rm APP}|\,|B_{\rm MAC}|\,|B_{\rm PHY}|)$, and the subvalue iteration at PHY defined in (40) has complexity $O(|A_{\rm PHY}|)$. Hence, the total complexity of one iteration (i.e., sweeping the whole state space once) of LSDP can be expressed as $O(|S_{\rm PHY}|^{2}|S_{\rm APP}|\,|A_{\rm APP}|\,|A_{\rm PHY}|\,|B_{\rm MAC}|\,|B_{\rm PHY}|)$.

For LRTDP, as the policy update is time slotted, we consider the complexity within one time slot, which is referred to as one round of iteration here as well. Similar to LSDP, the complexity of one round in LRTDP is $O(|A_{\rm APP}|\,|A_{\rm PHY}|^{2}|B_{\rm MAC}|\,|B_{\rm PHY}|)$. The major difference between the complexity of LSDP and LRTDP is the size of APP state space $|S_{\rm APP}|$, which depends on the chosen encoding parameters, but which is usually larger than 10^{8} for video application when we consider the prioritized classification of

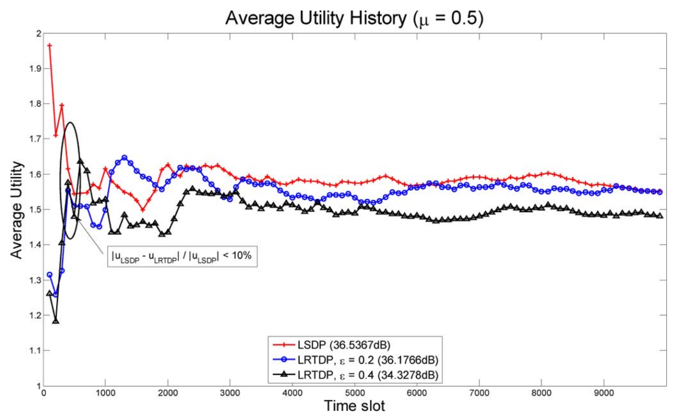


Fig. 4. The performance comparison between LSDP and LRTDP.

video data as in Appendix A. However, the number of iterations which LRTDP takes to achieve a similar performance as LSDP are relatively few. Fig. 4 compares the performances of LSDP and LRTDP with the same discounted factor μ. From this figure, we note that after around 400 time slots (indicated by the circle), i.e., 800 ms as the length of one time slot $\Delta T = 2$ ms, LRTDP catches up with the performance of LSDP (i.e., the relative difference is less than 10%). This means that, to obtain a similar performance, LRTDP only requires to visit in the order of 10^3 states, while LSDP requires in the order of 10^8 states, which will lead to significant differences among the algorithms in their run-time performance.

In terms of message exchange, LSDP requires to report the QoS of PHY and MAC to the upper layers, which has a communication cost of $O(|B_{\mathrm{PHY}}|\,|B_{\mathrm{MAC}}|)$. The downward transmission of subvalue functions takes a communication cost of $O(|S_{\mathrm{PHY}}|)$. Hence, the total communication cost of one iteration of LSDP is roughly $O(|S_{\mathrm{PHY}}|^2|S_{\mathrm{APP}}|)$. Similar to the computation complexity, LRTDP also requires significantly fewer inter-layer message exchanges to achieve a performance close to LSDP.

5) The Convergence Analysis of LRTDP: In this section, we prove that LRTDP converges to the optimal policy at each layer.

First, we prove that the state-value function converges to the optimum in LRTDP. Proposition 1 shows that, if the state transition probability is known, then the subvalue function at each layer converges to their optimum if ε-greedy exploration strategy is applied. Afterwards, we extend this conclusion in Proposition 2 showing that when the state transition probability is unknown, the convergence results of Proposition 1 still hold for the LRTDP's on-line estimation. Subsequently, we finalize our analysis in Proposition 3 showing that the policy generated by LRTDP converges to the optimum at each layer as well.

Proposition 1: With state transition probability known, LRTDP converges to the optimal state-value function with ε-greedy exploration strategy as long as the state space $\boldsymbol{S}$ is finite and communicating. The optimal state-value function is unique and satisfies the following Bellman's equations:

$$V^*(\boldsymbol{s}) = \max_{a,b}\left\{U(\boldsymbol{s},\boldsymbol{a},\boldsymbol{b}) + \mu\sum_{s'\in\boldsymbol{S}} p(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a})V^*(\boldsymbol{s})\right\}$$

$$= \max_{a,\boldsymbol{Z}}\left\{U(\boldsymbol{s},\boldsymbol{a},Z) + \mu\sum_{s'\in\boldsymbol{S}} p(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a})V^*(\boldsymbol{s}')\right\} \quad (9)$$

Proof: The proof can be found in Appendix C.

Proposition 1 proves the convergence of LRTDP under the assumption that the state transition probability is known. This proposition can be further extended to the case in which the state transition probability is unknown but can be estimated as in Section III-C-1). This extension is presented next.

Proposition 2: With indirect on-line estimation, the state-value function converges to the optimal state-value function when the transition probabilities are unknown.

Proof: The proof can be found in Appendix D.

With state-value function $V(\boldsymbol{s})$ converging, it is straightforward to show the convergence of LRTDP's policy.

Proposition 3: LRTDP's policy converges to the optimum at each layer.

Proof: Proposition 1 and 2 have shown that LRTDP converges to the optimal state-value function $V^*(\boldsymbol{s})$ with on-line estimation and ε-greedy exploration strategy, i.e.

$$\lim_{t\to\infty} V^t_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}) = V^*_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}) \quad (10)$$

and

$$\lim_{t\to\infty} V^{t+1}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = V^*(s_{\mathrm{PHY}}, s_{\mathrm{APP}}). \quad (11)$$

Since the policy is generated as

$$\pi^t_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}) = \arg\max_{\substack{a_{\mathrm{APP}}\in A_{\mathrm{APP}}\\ z_{\mathrm{MAC}}\in\mathcal{Z}_{\mathrm{MAC}}}} V^t_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}) \quad (12)$$

and

$$\pi^t_{\mathrm{PHY}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = \arg\max_{a_{\mathrm{PHY}}\in A_{\mathrm{PHY}}} V^{t+1}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}), \quad (13)$$

their value at the time limit are

$$\begin{aligned}\lim_{t\to\infty} \pi^t_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}) &= \lim_{t\to\infty}\arg\max_{\substack{a_{\mathrm{APP}}\in A_{\mathrm{APP}}\\ z_{\mathrm{MAC}}\in\mathcal{Z}_{\mathrm{MAC}}}} V^t_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}})\\ &= \arg\max_{\substack{a_{\mathrm{APP}}\in A_{\mathrm{APP}}\\ z_{\mathrm{MAC}}\in\mathcal{Z}_{\mathrm{MAC}}}} \lim_{t\to\infty} V^t_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}})\\ &= \arg\max_{\substack{a_{\mathrm{APP}}\in A_{\mathrm{APP}}\\ z_{\mathrm{MAC}}\in\mathcal{Z}_{\mathrm{MAC}}}} V^*_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}})\\ &= \pi^*_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}) \end{aligned} \quad (14)$$

and

$$\begin{aligned}\lim_{t\to\infty} \pi^t_{\mathrm{PHY}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) &= \lim_{t\to\infty}\arg\max_{a_{\mathrm{PHY}}\in A_{\mathrm{PHY}}} V^{t+1}(s_{\mathrm{PHY}}, s_{\mathrm{APP}})\\ &= \arg\max_{a_{\mathrm{PHY}}\in A_{\mathrm{PHY}}} \lim_{t\to\infty} V^{t+1}(s_{\mathrm{PHY}}, s_{\mathrm{APP}})\\ &= \arg\max_{a_{\mathrm{PHY}}\in A_{\mathrm{PHY}}} V^*(s_{\mathrm{PHY}}, s_{\mathrm{APP}})\\ &= \pi^*_{\mathrm{PHY}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}). \end{aligned} \quad (15)$$

TABLE II
EXPERIMENT PARAMETERS

Layer	Parameter	Value
APP Layer (APP)	Packet length	$L = 1000\ bytes$
	Frame interval	$T_{frame} = 1/f_s = 20ms$
	Lagrangian multiplier	$\lambda^b = 0.3,\ \lambda^a_{PHY} = 0.2,\ \lambda^a_{APP} = 0.35$
	Number of priority class	$H = 45$
	Maximum GOP length	$M = 15$
MAC Layer (MAC)	Maximum retransmission limit	$Rt_{\max} = 5$
	Duration of TXOP	$t_{TXOP} = 500\mu s$
	Length of time slot	$\Delta T = 2ms$
Physical Layer (PHY)	Received SNR threshold	$\widehat{\Gamma} \in [0, 0.4, 0.8, \cdots, 4]\, dB$
	Power allocation	$a_{PHY} \in [0.5, 1.0, 1.5, 2.0]\, mW$
	Modulation level	$BPSK, QPSK, 16QAM, 64QAM$
	FEC rate	½, ⅔, ¾

Notice that APP's policy is a composite mapping from PHY's policy and APP's PHY-dependent policy, i.e.

$$\pi^t_{\text{APP}}(s_{\text{PHY}}, s_{\text{APP}}) = \pi^t_{\text{APP}}\left(s_{\text{PHY}}, s_{\text{APP}}, \pi^t_{\text{PHY}}(s_{\text{PHY}}, s_{\text{APP}})\right). \quad (16)$$

We further have the convergence of APP's policy as $\lim_{t\to\infty} \pi^t_{\text{APP}}(s_{\text{PHY}}, s_{\text{APP}}) = \pi^*_{\text{APP}}(s_{\text{PHY}}, s_{\text{APP}})$. Therefore, we have proved the convergence of LRTDP's policy at both PHY and APP layers. ■

IV. NUMERICAL RESULTS

A. Experiment Settings

In this section, we test the performance of the proposed LRTDP method using the illustrative example described in Appendix A, and compare it with the conventional LSDP method described in Appendix B. The video sequence "Foreman" with length of 50 s (CIF resolution, 50 Hz frame-rate) is compressed by an H.264/AVC codec [27] with target bit rate of 1.5 Mbit/s. At PHY, we consider the 802.11a standard, it has 8 operation modes (the specification of different 802.11 modes can be found in Table V), with the symbol rate varies from 333 to 500 KBd/s. Convolutional codes are employed to perform forward error correction (FEC). Table II summarizes the parameters used at each layer.

B. Experiment Results

1) LRTDP vs. Alternative Methods: In this experiment, we compared LRTDP with alternative cross-layer methods.

First, we show that our LRTDP algorithm approaches the optimal performance obtained by LSDP. We use the average utility as the performance metric, which is the average one-step utility received by the user during the period starting from the beginning of streaming to the current moment, i.e.,

$$\bar{U}^t = \frac{\sum_{i=1}^t U^i}{t}. \quad (17)$$

Fig. 5 shows the average utilities obtained by LSDP and LRTDP, with the resulting average PSNR also listed in the legend. LSDP first computes the optimal cross-layer transmission policy off-line, and subsequently applies it for the real-time video transmission. If we assume that LSDP has full knowledge of the system dynamics, this policy results in the optimal performance. Therefore, the performance of LSDP can be served as the upper bound of LRTDP's performance. While implementing LRTDP, there is no *a priori* knowledge assumed about the system dynamics and the wireless user needs to learn these on-line. This explains why the transient performance of LRTDP falls behind LSDP at the beginning of the streaming process. However, LRTDP's on-line learning gradually accumulates its knowledge about system dynamics, and its performance (e.g., LRTDP with $\varepsilon = 0.2$ and $\mu = 0.4$) approaches that of LSDP.

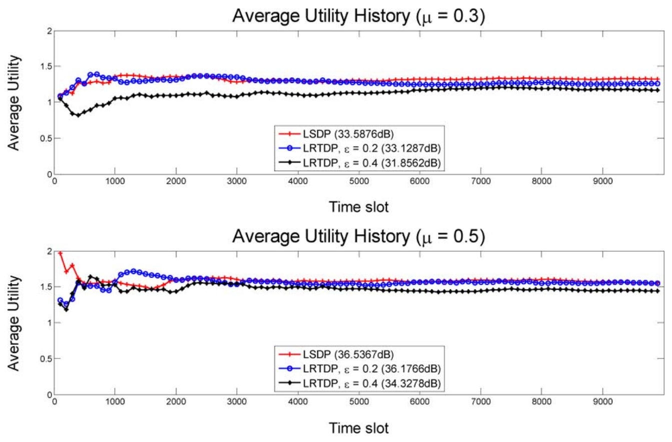


Fig. 5. Average utility obtained using LSDP and LRTDP.

Fig. 6 then compares the average utilities achieved by LRTDP and the approach with only APP layer adaptation [40]. From this figure, we can observe that, over time, the layered RTDP algorithm achieves an average reward of 1.6002 and an average PSNR of 36.1766 dB, while the APP layer adaptation achieves an average reward of 1.3127 and an average PSNR of 34.1743 dB. It shows that the cross-layer optimization improves the system performance by jointly scheduling the actions at different layers, e.g., the adaptation of MAC and PHY's actions to the video traffic condition.

Next, we analyze the effect of the foresighted optimization. Fig. 7 shows the average utilities obtained by LRTDP with my-

TABLE III
THE IMPACT OF DISCOUNT FACTOR ($\varepsilon = 0.2$)

μ	0	0.2	0.4	0.6	0.8	1.0
Average utility	1.1222	1.2083	1.4629	1.6325	1.5822	1.3746
Average PSNR (dB)	32.7345	33.4332	34.4540	36.1323	36.2323	34.6575

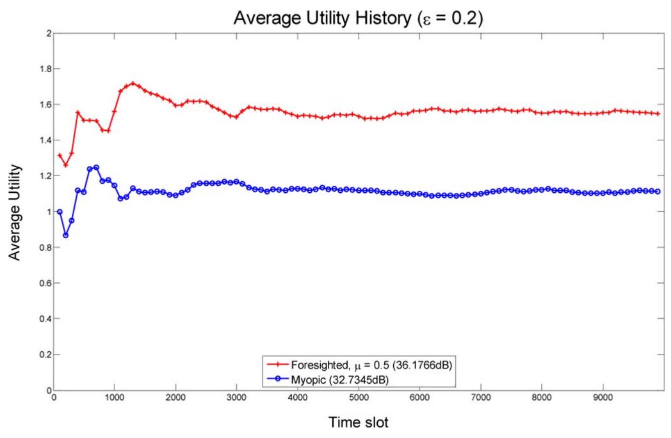


Fig. 6. Average utility obtained using foresighted and myopic LRTDP.

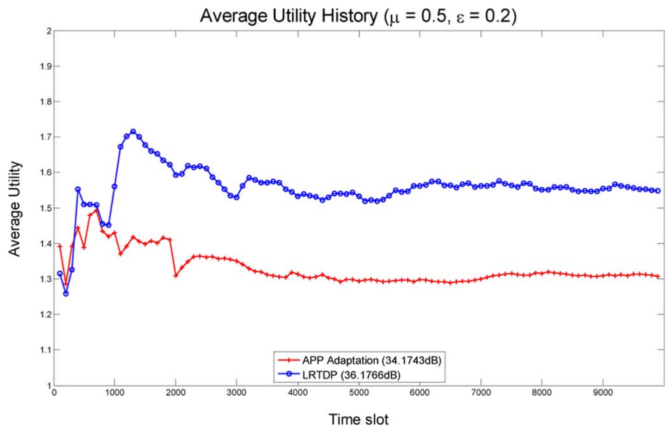


Fig. 7. Average utility obtained using centralized and layered RTDP.

opic optimization (i.e., $\mu = 0$) [39] and foresighted optimization ($\mu = 0.5$). The foresighted LRTDP achieves an average utility of 1.6002 and an average PSNR of 36.1766 dB, while the myopic LRTDP only achieves an average utility of 1.1222 and an average PSNR of 32.7345 dB. A more detailed discussion about the effect of discount factor μ can be found in the next subsection.

2) The Impact of Discount and Exploration Factors: In this section, we further consider the impact of the exploration factor ε and the discount factor μ on LRTDP's performance. From Fig. 5, we notice that LRTDP with a smaller exploration factor and larger discount factor performs better. This is expected, since a smaller exploration factor leads to less exploration and hence, LRTDP can focus on the greedy action which has more immediate effect in increasing the user's utility, and a larger discount factor gives LRTDP more foresight when updating its policy, thereby improving the average utility in the long-term. However, the following discussion shows that the above intuition on ε and μ does not always hold.

The discount factor μ determines the level of "foresight" in making control decisions for the streaming process and hence, it has remarkable influence over the system performance. Table III. summarizes the impact of the discount factor on the average utility achieved in transmitting a 50-s "Foreman" sequence with exploration factor $\varepsilon = 0.2$. An increase on μ does not necessarily lead to an improvement on the average utility as in [17]. A larger discount factor leads to more foresight when making transmission decisions, and therefore improves LRTDP's performance when the policy has reached the optimum, but such an improvement is achieved at the expense of slowing down LRTDP's convergence speed to the optimal policy. As a result, a larger μ does not necessarily perform better than a smaller μ, if we consider the results obtained for only a finite period of time, before the policy arrives at its optimum.

A similar argument exists for the exploration factor ε. It is well known that the policy update of different states in dynamic programming is unbalanced under the greedy exploration strategy [13], which means that, during the streaming process, there are some states which will be visited more frequently than others because the expected utility will be maximized in this way. For example, under heavy video traffic conditions, PHY usually has the tendency to raise its transmission power in order to reach a higher SNR state; on the contrary, under light video traffic conditions, PHY will reduce its transmission power to save energy. Therefore, ε has a similar impact as μ in balancing the user's long-term and short-term benefit when making transmission decisions, as shown in Table IV. Starting from the extreme case when $\varepsilon = 0$ and there is no exploration, the average utility and PSNR received by LRTDP increase along with ε, as exploring different actions other than the greedy one helps the user to visit more states, which thereby helps to accelerate the policy's convergence to optimum. Nevertheless, when ε further increases, the decision making of LRTDP gradually changes from "greedy" (i.e., focusing only on the greedy actions and thus a small set of states) to "random" (i.e. trying different actions and thus visiting and updating a larger set of states simultaneously). These can be considered as two different paths leading to the same optimal policy when time goes to infinity. Given a finite streaming duration, the "random" path with a too large exploration factor will affect the received average utility in a negative manner. Therefore, the values of μ and ε should be carefully selected and adjusted for real applications, which is affected by the experienced environmental dynamics.

3) The Impact of Model Inaccuracy: Since one of the biggest differences between LSDP and LRTDP is the requirement on the *a priori* knowledge about the system dynamics, here we analyze how the performances of these methods change when the state transition probability is not accurately known. To get a better

TABLE IV
THE IMPACT OF EXPLORATION FACTOR ($\mu = 0.5$)

ε	0	0.2	0.4	0.6	0.8	1.0
Average utility	1.5088	1.6002	1.4524	1.0127	0.6248	0.1969
Average PSNR (dB)	33.7345	36.1766	34.3278	31.4209	30.1230	30.0193

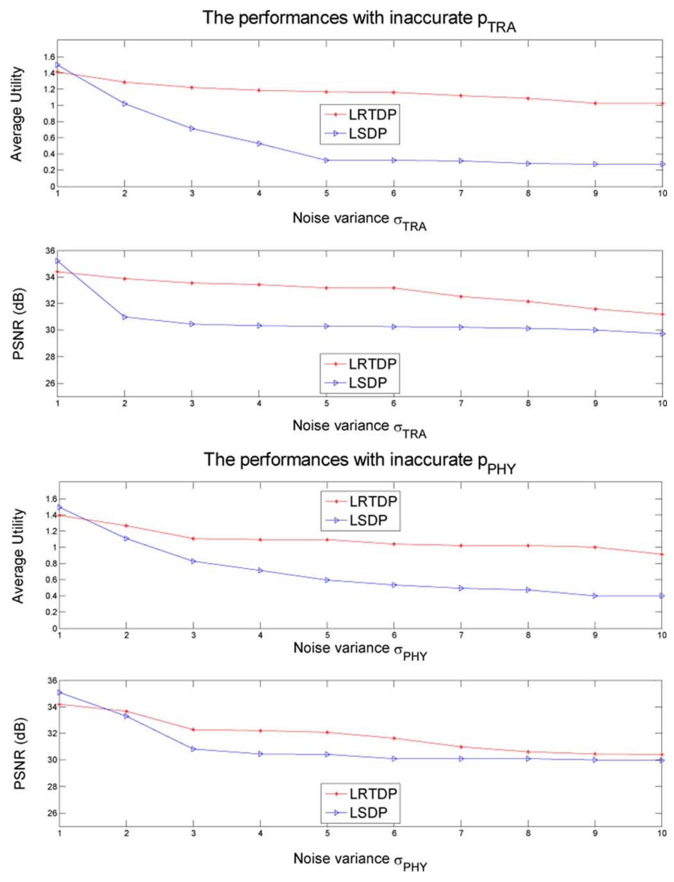


Fig. 8. The average performances along with noise variance ($\varepsilon = 0.2, \mu = 0.5$).

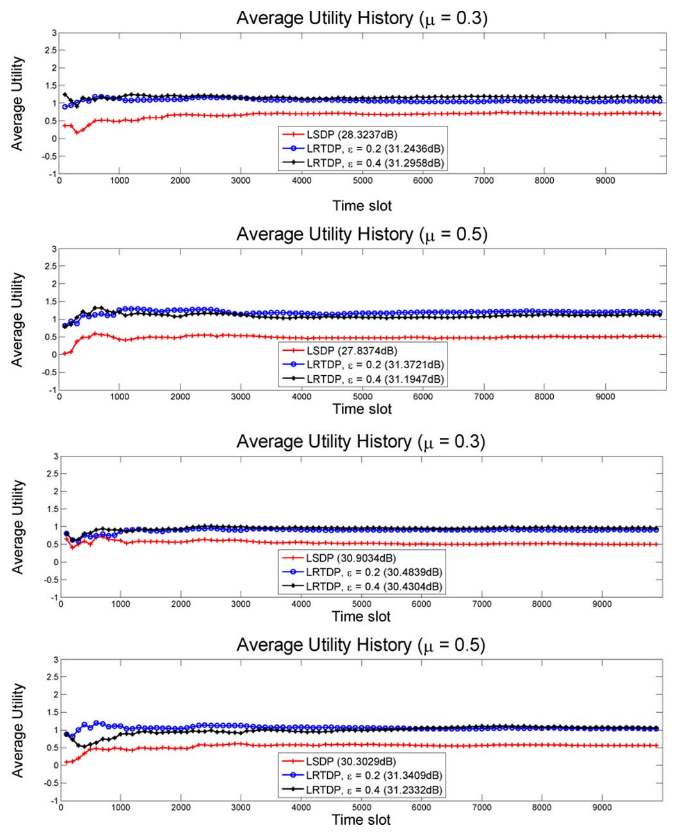


Fig. 9. (a) The influence of p_{TRA} inaccuracy ($\sigma_{\mathrm{TRA}} = 0.3$). (b) The influence of p_{PHY} inaccuracy ($\sigma_{\mathrm{PHY}} = 0.3$).

observation on the impact of each layer, we analyze the two sets of state transition probabilities, p_{TRA} and p_{PHY} separately. p_{TRA} can be represented by an $H \times H$ matrix, where H is the number of priority classes and each entry $p_{\mathrm{TRA}}(i,j)$ stores the transition probability of incoming traffic from frame type i to frame type j. In this experiment, we add to p_{TRA}an $H \times H$ Gaussian noise matrix $N_{\sigma_{\mathrm{TRA}}}$, with variance σ_{TRA}, that is

$$\tilde{p}_{\mathrm{TRA}}(i,:) = \frac{p_{\mathrm{TRA}}(i,:) + N_{\sigma_{\mathrm{TRA}}}(i,:)}{\|p_{\mathrm{TRA}}(i,:) + N_{\sigma_{\mathrm{TRA}}}(i,:)\|_2}, \quad \text{for all } i. \tag{18}$$

p_{PHY} is a $|A_{\mathrm{PHY}}| \times |S_{\mathrm{PHY}}| \times |S_{\mathrm{PHY}}|$ matrix in our example, $p_{\mathrm{PHY}}(i, j_1, j_2)$ represents the transition probability from channel state j_1 to state j_2, when the power allocation i is executed. Thus, we add a Gaussian noise matrix $N_{\sigma_{\mathrm{PHY}}}$with the same size to it, with variance σ_{PHY}

$$\tilde{p}_{\mathrm{PHY}}(i,j_1,:) = \frac{P_{\mathrm{PHY}}(i,j_1,:) + N_{\sigma_{\mathrm{PHY}}}(i,j_1,:)}{\|P_{\mathrm{PHY}}(i,j_1,:) + N_{\sigma_{\mathrm{PHY}}}(i,j_1,:)\|_2}, \quad \text{for all } i. \tag{19}$$

The performances of LSDP and LRTDP with such inaccurate state transition probability are shown in Fig. 8. In both cases, LSDP is most severely influenced with a remarkable drop in both average utility and PSNR. Fig. 9 illustrates how the average performances (measured in average utility as well as PSNR) of both methods vary along with the level of model inaccuracy. Consistent with our intuition, the performance of LRTDP outperforms LSDP when the noise variance keeps increasing. Therefore, in situations where little knowledge of the system dynamics can be obtained, LRTDP is a better solution than LSDP since its performance is more robust and less dependent to the *a priori* knowledge.

V. CONCLUSION

In this paper, we considered the cross-layer optimization of the real-time video transmission of an individual user over a single wireless link, where the environment dynamics are time-varying and unknown to the wireless user. We formulated the cross-layer optimization into a layered Markov decision process, and proposed a novel layered real-time dynamic programming method to adapt the cross-layer transmission strategies to the experienced dynamic environment (including the source dynamics and channel dynamics) which is proved to converge to the optimal policy. The advantages of this approach are as follows: i) it adheres to the OSI structure, with only limited information exchanges among layers; ii) compared to the conventional methods, it has much lower computation complexities and message exchange overheads; iii) more importantly, this method learns about system dynamics on-line, and has no *a priori* knowledge requirement. In the subsequent experiments, we further showed that this method is more robust than the conventional cross-layer optimization methods with complete knowledge in a nonstationary and time-varying wireless environment.

Appendix A
An Illustrative Video Streaming Example

In this section, we provide an illustrative example of states, actions, QoS, state transition probabilities, and system utility functions within the system model for video streaming established in Section II.

A. PHY Layer

1) PHY State and State Transition: We assume that the received signal envelope has a Rayleigh distribution with additive Gaussian noise in a typical multipath propagation environment, the received instantaneous SNR γ is distributed exponentially with the probability density function $p_\gamma(\gamma) = (1/\bar{\gamma}(a_{\mathrm{PHY}}))\exp(-(\gamma/\bar{\gamma}(a_{\mathrm{PHY}})))$ [9], where $\bar{\gamma}(a_{\mathrm{PHY}})$ is the average SNR and is determined by the power allocation.

s_{PHY} is represented by γ within each time slot. Due to the continuity of γ, the cardinality of PHY's state space S_{PHY} is infinite. Hence, we have to quantize the continuous SNR value to make the state space finite. Let $\vec{\Gamma} = [\Gamma_1, \Gamma_2, \ldots, \Gamma_{K+1}]$ be the received SNR thresholds in increasing order with $\Gamma_1 = 0$ and $\Gamma_{K+1} = \infty$, the total SNR space can be partitioned into K intervals. We assume a quantization function $Q(\cdot)$ which maps the received SNR into one of K discrete values, i.e., $s_{\mathrm{PHY}} = Q(\gamma) \in S_{\mathrm{PHY}} = \{\widehat{\Gamma}_1, \widehat{\Gamma}_2, \ldots, \widehat{\Gamma}_K\}$, where $\widehat{\Gamma}_k \in [\Gamma_k, \Gamma_{k+1}]$ is the representative SNR of the interval $[\Gamma_k, \Gamma_{k+1}]$.

As we already assumed that the SNR remains constant within one time slot, the one-step state transition at PHY happens only on the boundary of two successive time slots, and is restricted from a given state to its two adjacent states. It can be approximated as in [9]

$$p_{\mathrm{PHY}}\left(s_{\mathrm{PHY}}^{t+1} \mid s_{\mathrm{PHY}}^{t}, a_{\mathrm{PHY}}^{t}\right) = \begin{cases} \frac{F(\Gamma_{k+1})T_p}{\phi_k}, & s_{\mathrm{PHY}}^{t} = \widehat{\Gamma}_k, \quad s_{\mathrm{PHY}}^{t+1} = \widehat{\Gamma}_{k+1} \\ \frac{F(\Gamma_{k-1})T_p}{\phi_k}, & s_{\mathrm{PHY}}^{t} = \widehat{\Gamma}_k, \quad s_{\mathrm{PHY}}^{t+1} = \widehat{\Gamma}_{k-1} \\ 1 - \frac{F(\Gamma_{k-1})T_p}{\phi_k} - \frac{F(\Gamma_{k+1})T_p}{\phi_k}, & s_{\mathrm{PHY}}^{t+1} = s_{\mathrm{PHY}}^{t} = \widehat{\Gamma}_k \end{cases} \tag{20}$$

where T_p is the transmission time for one packet, $F(\Gamma)$ is the level crossing rate of SNR level Γ for the SNR process, and is expressed as

$$F(\Gamma) = \sqrt{\frac{2\pi\Gamma}{\bar{\gamma}(a_{\mathrm{PHY}})}} f_m \exp\left(-\frac{\Gamma}{\bar{\gamma}(a_{\mathrm{PHY}})}\right) \tag{21}$$

TABLE V
Possible 802.11a Operation Modes

Mode	Modulation	FEC Rate	Gross Rate (Mbit/s)
1	BPSK	½	12
2	BPSK	¾	12
3	QPSK	½	24
4	QPSK	¾	24
5	16-QAM	½	48
6	16-QAM	¾	48
7	64-QAM	⅔	72
8	64-QAM	¾	72

where f_m is the maximum Doppler frequency. The steady-state probability of each state equals

$$\begin{aligned} \phi_k &= \int_{\Gamma_k}^{\Gamma_{k+1}} p_\gamma(\gamma) d\gamma \\ &= \exp\left(-\frac{\Gamma_k}{\bar{\gamma}(a_{\mathrm{PHY}})}\right) - \exp\left(-\frac{\Gamma_{k+1}}{\bar{\gamma}(a_{\mathrm{PHY}})}\right). \end{aligned} \tag{22}$$

2) PHY QoS: In this example, we consider transmission over the 802.11a standard, which can offer bit rates up to 54 Mb/s and is suitable for real-time video transmission over WLAN. The available modulation and coding schemes in 802.11a are given in Table V. The technical details of different modes in 802.11a (net rate, gross rate, code rate, efficiency, etc.) can be found in [34].

Therefore, the selection of modulation and channel coding scheme, i.e., b_{PHY} is equivalent to selecting among the possible 802.11a PHY modes. The transmission rate v_{PHY} is defined as the effective rate which depends on both the modulation level and channel coding rate with specified values available in [34].

The packet loss rate e_{PHY} can be approximated using the sigmoid function as in [10] and [41]: $e_{\mathrm{PHY}} = (1)/(1 + e^{\varsigma(s_{\mathrm{PHY}} - \vartheta)})$, where ς and ϑ are empirical constants determined by the modulation scheme, channel coding, and packet length L, which is assumed up to 1000 bytes in this paper [10].

Thus, the effective data rate within each time slot is $\bar{v}_{\mathrm{PHY}} = v_{\mathrm{PHY}} \times (1 - e_{\mathrm{PHY}})$, which represents the actual number of packets to be transmitted within one time slot, or in other words, the goodput at the PHY layer.

The internal cost ω_{PHY} of transmitting one packet is defined as the energy-rate function [11]: $\omega_{\mathrm{PHY}} = \sigma_B(2^{2v_{\mathrm{PHY}}} - 1)/g$, where σ_B denotes thermal noise and g is the channel gain.

B. MAC Layer

As we used a simplified MAC model, only the internal action b_{MAC} needs to be specified. Here we assume $b_{\mathrm{MAC}} \in B_{\mathrm{MAC}} = \{0, \ldots, Rt_{\max}\}$, where $Rt_{\max}$ is the maximum retransmission limit.

Together with the QoS Z_{PHY} provided by the PHY layer, b_{MAC} and s_{MAC} determines the QoS Z_{MAC} provided for the

TABLE VI
THE WORKING PROCEDURE OF LRTDP

```
Algorithm: Layered Real-Time Dynamic Programming

Initialize $V(\boldsymbol{s}) = 0$ for all $\boldsymbol{s} \in \boldsymbol{\mathcal{S}}$
Initialize $p(s' \mid s, a) = 1/|S|$, for all transition probability sets.
$\boldsymbol{s} = \boldsymbol{s}^{(0)}$, $t = 0$
Repeat
    Update $p^t(s^t \mid s^{t-1}, a^{t-1}) = \frac{n^t(s^t \mid s^{t-1}, a^{t-1})}{n^t(s^{t-1}, a^{t-1})}$ at both APP and PHY
    PHY computes QoS set $\mathcal{Z}^t_{PHY}(s^t_{PHY})$
    MAC computes QoS set $\mathcal{Z}^t_{MAC}(s^t_{PHY})$
    APP updates the set of PHY-dependent policy $\{\pi^t_{APP}(s^t_{PHY}, s^t_{APP}, a_{PHY}) \mid a_{PHY} \in A_{PHY}\}$
    PHY updates its policy for current state $\pi^t_{PHY}(s^t_{PHY}, s^t_{APP}) = a_{PHY}(s^t_{PHY}, s^t_{APP})$
    State value function for current state is updated as $V^{t+1}(s^t_{PHY}, s^t_{APP})$
    APP updates its policy for current state
$\pi^t_{APP}(s^t_{PHY}, s^t_{APP}) = \{a_{APP}(s^t_{PHY}, s^t_{APP}, a_{PHY}(s^t_{PHY}, s^t_{APP})), Z_{MAC}(s^t_{PHY}, s^t_{APP}, a_{PHY}(s^t_{PHY}, s^t_{APP}))\}$
    PHY selects its action $a^t_{PHY} = \pi^t_{PHY}(s^t_{PHY}, s^t_{APP})$ with probability $\varepsilon^t(s^t_{PHY}, s^t_{APP})$
    APP selects the greedy QoS $Z^t_{MAC} = Z_{MAC}(s^t_{PHY}, s^t_{APP})$ and $a^t_{APP}$ with $\varepsilon - greedy$ strategy
    The actions $a^t_{PHY}$, $a^t_{APP}$, $b^t_{PHY}$, $b^t_{MAC}$ are executed
    State transition $s^{t+1} \leftarrow s^t$ at each layer
    Time evolves: $t + 1 \leftarrow t$
```

APP layer, as in [6] (as s_{MAC} is constant, we neglect it in the expression of Z_{MAC})

$$Z_{\text{MAC}}(b_{\text{MAC}}, s_{\text{MAC}}, Z_{\text{PHY}}) = (e_{\text{MAC}}, v_{\text{MAC}}, \omega_{\text{MAC}}) = \left((e_{\text{PHY}})^{b_{\text{MAC}}+1}, \frac{(1 - e_{\text{PHY}})v_{\text{PHY}}}{(1 - (e_{\text{PHY}})^{b_{\text{MAC}}})}, \frac{\left(1 - (e_{\text{PHY}})^{b_{\text{MAC}}}\right)\omega_{\text{PHY}}}{(1 - e_{\text{PHY}})} \right). \tag{23}$$

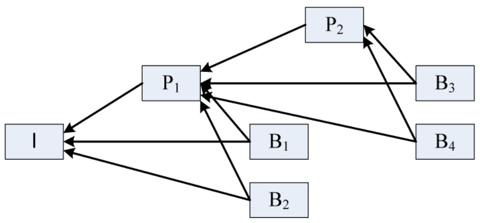


Fig. 10. The DAG of a GOP (IBBPBBP).

C. APP Layer

Similar to [2] and [7], we partition the incoming encoded video packets into H different priority classes and adjust the transmission strategies accordingly for each class. This approach can significantly improve the overall received video quality. The set of priority classes depends on the specific video encoder used at APP. For example, in DCT-based video codecs (e.g., H.264/AVC), video streams are typically compressed into three classes of frames [Intra (I), Predictive (P), and Bidirectionally predictive (B), i.e., $H = 3$]. In this paper, we further consider the interdependency between packets, which can be captured by the Directed Acyclic Graph (DAG), when performing packet prioritization. In addition, we assume that each frame has an activity level taking value from the set $\{\text{High}, \text{Medium}, \text{Low}\}$ [8] in order to capture the variation in activity level (e.g., motion) between scenes.

To simplify our analysis, we assume that the maximum GOP length is bounded by M. Fig. 10 shows the direct acyclic graph (DAG) of a group of (IBBPBBP) frames, which reflects the interdependency between frames typical of an encoding by a standard video coder (e.g., H.264/AVC). In the DAG, each leaf node is considered as a priority class and hence there are 7 priority classes in this GOP. We further define the depth of any class to be its maximum distance to the DAG's root. For example, the depth of the first P frame (P1 for short) in this GOP is 1 and the depth of the fourth B frame (B4 for short) is 4.

Thereby, we can formally assume that a packet from class $h \in S_{\text{TRA}} = \{h_i; i = 1, \ldots, H\}$ can be characterized by the following parameters in our priority classification model:

1) The distortion impact q_h depends on the underlying video characteristics, encoding parameters, etc. It reflects the importance of the packet in terms of quality contribution. Here we assume an additive distortion reduction [15], and

q_h represents the distortion reduction when a packet h is received and successfully decoded at the decoder's side [see (24)–(26), shown at the bottom of the page].

2) The packet lifetime length d_h, which is an integer number with the actual life time to be $d_h \Delta T$ seconds.

The depth l_h as defined earlier.

Moreover, the number of packets generated by one frame of class h is modeled as a random variable r_h with mean $\bar{r}_h$, whose probability mass function is assumed to be only depended on the class type but independent of time [8]. With this traffic model, packets from different GOPs and belonging to the same priority group are treated similarly and the total number of priority classes is $H = 3L$, where 3 is the number of activity levels.

As ΔT is assumed to be much smaller than the frame interval, it is straightforward that the incoming traffic within an individual time slot belongs to the same priority class. The traffic state can be represented as $s_{\text{TRA}} = (h, r_h)$, where h is the priority class that incoming packets belongs to and r_h is the number of incoming packets. It has been shown in [8] that the transition probability $\{p(h^{t+1}|h^t)\}$ only depends on the specific GOP structure and video content.

With the incoming traffic differentiated in different priority classes, the buffer occupancy is represented as $s_{\text{BUF}} = [\kappa_1, \kappa_2, \ldots, \kappa_{d_{\max}}]$, where $\kappa_d = (\kappa_{d,1}, \ldots, \kappa_{d,H})$, and $\kappa_{d,h} (1 \le h \le H)$ represents the number of packets from priority class h who have a remaining lifetime of d time slots, and $d_{\max} = \max_{1 \le h \le H} d_h$ is the largest packet lifetime.

The packet scheduling at APP is defined as $a_{\text{APP}} = [a_1, a_2, \ldots, a_{d_{\max}}]$, where $a_d = (a_{d,1}, \ldots, a_{d,H})$ are the numbers of packets to be transmitted who have a remaining lifetime of d time slots.

The update of s_{BUF} over one time slot is simply performed by deleting the transmitted packets and expired packets, and adding the new incoming packets as follows:

$$\left(\kappa_1^{t+1}, \ldots, \kappa_{d_{\max}}^{t+1}\right) = \left(\kappa_2^t - \Delta\kappa_2^t, \ldots, \kappa_{d_{\max}+1}^t - \Delta\kappa_{d_{\max}+1}^t\right). \quad (27)$$

$\kappa_{d_{\max}+1}^t = 0$ and $\{\Delta\kappa_d^t\}$ denotes the buffer change during time slot t and is computed as

$$\Delta\kappa_d^t = \begin{cases} a_d^t - r_d^t, & d = d_h + 1, \quad 1 \le h \le H \\ -r_d^t, & d = d_{\max} \\ a_d^t, & \text{otherwise} \end{cases} \quad (28)$$

where $r^t = \{r_h^t; 1 \le h \le H\}$ denotes the number of incoming packets within time slot t. The buffer state transition probability is computed as

$$p_{\text{BUF}}\left(s_{\text{BUF}}^{t+1} | s_{\text{BUF}}^t, s_{\text{TRA}}^t, a_{\text{APP}}^t\right) = \begin{cases} p(r^t \mid h^t) & \text{(24) satisfied} \\ 0 & \text{otherwise} \end{cases}. \quad (29)$$

The received video quality of APP is computed as $q(s_{\text{APP}}, a_{\text{APP}}, Z_{\text{MAC}}) = \sum_{d=1}^{d_{\max}} \sum_{h=1}^{H} a_{d,h} q_h$, which is the distortion reduction contributed by the successfully transmitted and decoded packets.

Appendix B
Decomposition of DP Operator

In this Appendix, we decompose our DP operator into sub-value functions which can be updated locally at each layer. Note that MAC only has internal action b_{MAC}, which can be optimized at APP through the selection on Z_{MAC}, therefore we do not consider the optimization at MAC explicitly.

The DP operator in (4) can be rewritten as (24).

Instead of finding the optimal external actions and internal actions (e.g., the QoS level) simultaneously, we can decompose (24) into a two-loop optimization, as in (25) and (26)

$$\begin{aligned} V^{\dagger}(s_{\text{PHY}}, s_{\text{APP}}, a_{\text{PHY}}, a_{\text{APP}}, Z_{\text{MAC}}) &= q(s_{\text{APP}}, a_{\text{APP}}, Z_{\text{MAC}}) - \lambda_{\text{PHY}}^a c_{\text{PHY}}(s_{\text{PHY}}, a_{\text{PHY}}) \\ &\quad - \lambda_{\text{APP}}^a c_{\text{APP}}(s_{\text{APP}}, a_{\text{APP}}) - \lambda^b \omega_{\text{APP}}(s_{\text{APP}}, Z_{\text{MAC}}) \\ &\quad + \mu \sum_{\substack{s'_{\text{PHY}} \in S_{\text{PHY}} \\ s'_{\text{APP}} \in S_{\text{APP}}}} p_{\text{PHY}}(s'_{\text{PHY}} | s_{\text{PHY}}, a_{\text{PHY}}) \\ &\quad \times p_{\text{APP}}(s'_{\text{APP}} | s_{\text{APP}}, a_{\text{APP}}) V(s'_{\text{PHY}}, s'_{\text{APP}}) \end{aligned} \quad (30)$$

$$V(s_{\text{PHY}}, s_{\text{APP}}) = \max_{\substack{a_{\text{PHY}}, a_{\text{APP}} \\ Z_{\text{MAC}}}} \left\{ \begin{array}{l} q(s_{\text{APP}}, a_{\text{APP}}, Z_{\text{MAC}}) - \lambda_{\text{PHY}}^a c_{\text{PHY}}(s_{\text{PHY}}, a_{\text{PHY}}) - \lambda_{\text{APP}}^a c_{\text{APP}}(s_{\text{APP}}, a_{\text{APP}}) \\ -\lambda^b \omega_{\text{APP}}(s_{\text{APP}}, Z_{\text{MAC}}) + \mu \sum_{\substack{s'_{\text{PHY}} \in S_{\text{PHY}} \\ s'_{\text{APP}} \in S_{\text{APP}}}} p_{\text{PHY}}(s'_{\text{PHY}} \mid s_{\text{PHY}}, a_{\text{PHY}}) p_{\text{APP}}(s'_{\text{APP}} \mid s_{\text{APP}}, a_{\text{APP}}) V(s'_{\text{PHY}}, s'_{\text{APP}}) \end{array} \right\} \quad (24)$$

$$V(s_{\text{PHY}}, s_{\text{APP}}) = \max_{a_{\text{PHY}}} \left\{ -\lambda_{\text{PHY}}^a c_{\text{PHY}}(s_{\text{PHY}}, a_{\text{PHY}}) + V_{\text{APP}}(s_{\text{PHY}}, s_{\text{APP}}, a_{\text{PHY}}) \right\} \quad (25)$$

$$V_{\text{APP}}(s_{\text{PHY}}, s_{\text{APP}}, a_{\text{PHY}}) = \max_{a_{\text{APP}}, Z_{\text{MAC}}} \left\{ \begin{array}{l} q(s_{\text{APP}}, a_{\text{APP}}, Z_{\text{MAC}}) - \lambda^b \omega_{\text{APP}}(s_{\text{APP}}, Z_{\text{MAC}}) - \lambda_{\text{APP}}^a c_{\text{APP}}(s_{\text{APP}}, a_{\text{APP}}) \\ +\mu \sum_{\substack{s'_{\text{PHY}} \in S_{\text{PHY}} \\ s'_{\text{APP}} \in S_{\text{APP}}}} p_{\text{PHY}}(s'_{\text{PHY}} \mid s_{\text{PHY}}, a_{\text{PHY}}) p_{\text{APP}}(s'_{\text{APP}} \mid s_{\text{APP}}, a_{\text{APP}}) V(s'_{\text{PHY}}, s'_{\text{APP}}) \end{array} \right\} \quad (26)$$

$$V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = \max_{\substack{a_{\mathrm{PHY}}, a_{\mathrm{APP}} \\ Z_{\mathrm{MAC}}}} \{V^\dagger(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}, a_{\mathrm{APP}}, Z_{\mathrm{MAC}})\} \tag{31}$$

$$V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = \max_{a_{\mathrm{PHY}}} \left\{ \max_{\substack{a_{\mathrm{APP}} \\ Z_{\mathrm{MAC}}}} V^\dagger(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}, a_{\mathrm{APP}}, Z_{\mathrm{MAC}}) \right\}. \tag{32}$$

Equation (26) only includes the selection of APP's action and QoS and, hence, is the subvalue function updated locally at APP, which can be taken as the inner loop of the whole optimization process. When $V_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}})$ for all $a_{\mathrm{PHY}} \in A_{\mathrm{PHY}}$ is updated, its value is reported to PHY, where subvalue function (25) is updated as the outer loop of optimization. In this order, the DP operator in (24) is updated once.

In the following part, we prove the update in (25) and (26) is equivalent to that in (24). To make the proof more clear, we use several shorthand notations as (30), (31), and (32).

Obviously, $V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}})$ equals to (24) which is the result for centralized optimization, and $V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}})$ equals to (25) and (26) for layered optimization.

According to the property of optimization, we have

$$V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) \geq V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}}). \tag{33}$$

Assuming that $(a^C_{\mathrm{PHY}}, a^C_{\mathrm{APP}}, Z^C_{\mathrm{MAC}})$ is the solution for (31), that is

$$V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = V^\dagger\left(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a^C_{\mathrm{PHY}}, a^C_{\mathrm{APP}}, Z^C_{\mathrm{MAC}}\right). \tag{34}$$

It is easy to tell that

$$\max_{\substack{a_{\mathrm{APP}} \\ Z_{\mathrm{MAC}}}} V^\dagger\left(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a^C_{\mathrm{PHY}}\right) \geq V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}}). \tag{35}$$

According to the definition of $V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}})$, we further have

$$V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) \geq \max_{\substack{a_{\mathrm{APP}} \\ Z_{\mathrm{MAC}}}} V^\dagger\left(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a^C_{\mathrm{PHY}}\right) \tag{36}$$

and, hence

$$V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) \geq V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}}). \tag{37}$$

Putting (33) and (37) together, we get our conclusion that

$$V^L(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = V^C(s_{\mathrm{PHY}}, s_{\mathrm{APP}}). \tag{38}$$

In the layered optimization defined by (25) and (26), APP needs to know the transition probability of PHY's states to update its policy, which is usually protocol-dependent and violates the current network structure. Therefore, a further improvement was made in [6] such that each layer does not need the transition probability of lower layers to update its policy, and the updates on both layers are correspondingly changed as (39) and (40), shown at the bottom of the page.

It has been shown in [6] that the update rules in (25) and (26) and in (39) and (40) provide close performances in cross-layer optimization, while the latter one consumes much less information exchange. Therefore, we use (39) and (40) as the update rule for LSDP in the following discussion of this paper.

APPENDIX C
PROOF OF PROPOSITION 1

To prove Proposition 1, we need the following two lemmas: Lemma 1 guarantees that every state in the state space is visited infinitely often with our ε-greedy strategy; Lemma 2 proves the convergence of asynchronously updated sequences.

Lemma 1 (Extended Borel-Cantelli Lemma): Let $\{Y_n\}$ be any process, and $\{A_n\}$ is a process adapting to $\{Y_n\}$, that is, $A_n \in \mathfrak{F}(Y_1, Y_2, \ldots, Y_n)$. Then almost surely

$$\{\omega; \omega \in A_n \text{ infinitely often}\} = \left\{\omega; \sum_{n=1}^{\infty} P(A_{n+1} \,|\, Y_n, \ldots, Y_1) = \infty\right\}. \tag{41}$$

Proof: The proof can be found in [20].

Definition 1: Define a sequence of nonempty sets $\{X(t) \in \mathbb{R}^M, t \geq 0\}$ with

$$\cdots \subset X(t+1) \subset X(t) \subset \cdots \subset X(0) \tag{42}$$

PHY:

$$V(s_{\mathrm{PHY}}, s_{\mathrm{APP}}) = \max_{a_{\mathrm{PHY}}} \left\{ \begin{array}{l} -\lambda^a_{\mathrm{PHY}} c_{\mathrm{PHY}}(s_{\mathrm{PHY}}, a_{\mathrm{PHY}}) \\ +\mu \sum_{s'_{\mathrm{PHY}} \in S_{\mathrm{PHY}}} p_{\mathrm{PHY}}\left(s'_{\mathrm{PHY}} | s_{\mathrm{PHY}}, a_{\mathrm{PHY}}\right) V_{\mathrm{PHY}}\left(s'_{\mathrm{PHY}}\right) \end{array} \right\}. \tag{39}$$

APP:

$$V_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, s'_{\mathrm{PHY}}) = \max_{a_{\mathrm{APP}}, Z_{\mathrm{MAC}}} \left\{ \begin{array}{l} q(s_{\mathrm{APP}}, a_{\mathrm{APP}}, Z_{\mathrm{MAC}}) \\ -\lambda^b \omega_{\mathrm{APP}}(s_{\mathrm{APP}}, Z_{\mathrm{MAC}}) - \lambda^a_{\mathrm{APP}} c_{\mathrm{APP}}(s_{\mathrm{APP}}, a_{\mathrm{APP}}) \\ +\mu \sum_{s'_{\mathrm{APP}} \in S_{\mathrm{APP}}} p_{\mathrm{APP}}(s'_{\mathrm{APP}} | s_{\mathrm{APP}}, a_{\mathrm{APP}}) V(s'_{\mathrm{PHY}}, s'_{\mathrm{APP}}) \end{array} \right\}. \tag{40}$$

which satisfies the following two conditions:

a) (a) *(Synchronous Convergence Condition)* For any $x \in X(t)$, a function $\eta(\cdot)$ maps it into $X(t+1)$, that is, $\eta(x) \in X(t+1), \forall t \in \mathbb{N}$ and $\forall x \in X(t)$.

b) *(Box Condition)* If X can be decomposed as $X(0) = X_1(0) \times X_2(0) \times \cdots \times X_M(0)$, then for every t, there exists sets $\{X_i(t) \subset X_i(0)\}$ such that $X(t) = X_1(t) \times X_2(t) \times \cdots \times X_M(t)$ ■

Lemma 2: (Asynchronous Convergence Theorem [14]) If the Synchronous Convergence and Box Conditions hold for the sequence of nonempty sets $\{X(t)\}$, and the initial solution estimate $x(0) = \{x_1(0), \ldots, x_M(0)\}$ belongs to the set $X(0)$, then every limit point of $\{x(t)\}$ is a fixed point of the function η as in Definition 1.

Proof: The proof can be found in [14]. ■

Proof of Proposition 1: We first show that every state $(s_{\mathrm{PHY}}, s_{\mathrm{APP}})$ is visited infinite times with ε-greedy strategy.

Let the probability of any action $(a_{\mathrm{PHY}}, a_{\mathrm{APP}})$ being chosen at the time slot t be denoted by $p^t(a_{\mathrm{PHY}}, a_{\mathrm{APP}} | s^t_{\mathrm{PHY}}, s^t_{\mathrm{APP}})$. If a state $\boldsymbol{s} = (s_{\mathrm{PHY}}, s_{\mathrm{APP}})$ is visited infinitely often, the total probability of $(a_{\mathrm{PHY}}, a_{\mathrm{APP}})$ being chosen from $(s_{\mathrm{PHY}}, s_{\mathrm{APP}})$ amounts to

$$\begin{aligned}\sum_{i=1}^{\infty} p^{t(s,i)} &\left(a_{\mathrm{PHY}}, a_{\mathrm{APP}} \,\middle|\, s^{t(s,i)}_{\mathrm{PHY}}, s^{t(s,i)}_{\mathrm{APP}}\right) \\ &\geq \sum_{i=1}^{\infty} \frac{\varepsilon^2}{|A_{\mathrm{PHY}}|\,|A_{\mathrm{APP}}|\left(n^{t(s,i)}\left(s^{t(s,i)}_{\mathrm{PHY}}, s^{t(s,i)}_{\mathrm{APP}}\right)\right)} \\ &= \frac{\varepsilon^2}{|A_{\mathrm{PHY}}|\,|A_{\mathrm{APP}}|} \sum_{i=1}^{\infty} \frac{1}{i} = \infty \end{aligned} \tag{43}$$

where $t(s,i)$ represents the time slot when s is visited for the ith time.

Let $Y^t = \{U^t, s^t_{\mathrm{PHY}}, s^t_{\mathrm{APP}}\}$ denote the observation at time slot t and $A^t = \{\omega : a^t(\omega) = (a_{\mathrm{PHY}}, a_{\mathrm{APP}})\}$, it is obvious that $\{A^t\}$ is adapting to the history of $\{Y^t\}$, which makes Lemma 1 applicable here. Together with (43), we draw the conclusion that action $(a_{\mathrm{PHY}}, a_{\mathrm{APP}})$ is executed infinitely often from state $(s_{\mathrm{PHY}}, s_{\mathrm{APP}})$.

Now let $\boldsymbol{S}_\infty$ be the set of states in the state space S which are being visited infinitely often. S_∞ is obviously not an empty set. Assuming there is a state $(s_{\widehat{\mathrm{PHY}}}, S_{\widehat{\mathrm{APP}}}) \notin \boldsymbol{S}_\infty$, and with the assumption of a communicating state space, there exists a state-action pair $(\boldsymbol{s}_{\mathrm{PHY}}, \boldsymbol{s}_{\mathrm{APP}}, a_{\mathrm{PHY}}, a_{\mathrm{APP}})$, which leads to $(S_{\widehat{\mathrm{PHY}}}, S_{\widehat{\mathrm{APP}}})$ with a positive probability $p(S_{\widehat{\mathrm{PHY}}}, S_{\widehat{\mathrm{APP}}} | \boldsymbol{s}_{\mathrm{PHY}}, \boldsymbol{s}_{\mathrm{APP}}, a_{\mathrm{PHY}}, a_{\mathrm{APP}}) > \sigma > 0$. Similar to (43), the total probability of $(S_{\widehat{\mathrm{PHY}}}, S_{\widehat{\mathrm{APP}}})$ being visited following every visit to $(\boldsymbol{s}_{\mathrm{PHY}}, \boldsymbol{s}_{\mathrm{APP}})$ is

$$\sum_{i=1}^{\infty} p\left(\underset{\mathrm{PHY}}{\widehat{S}}, \underset{\mathrm{APP}}{\widehat{S}} | s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}}, a_{\mathrm{APP}}, t(s,i)\right) \geq \sum_{i=1}^{\infty} \sigma = \infty. \tag{44}$$

Again, Lemma 1 yields the conclusion that $(S_{\widehat{\mathrm{PHY}}}, S_{\widehat{\mathrm{APP}}})$ is also visited infinitely often. As S is communicating, the above argument can be extended to the whole state space, and finally we know that all states are visited infinitely often with ε-greedy strategy.

In the next step, we prove that the state-value function update in LRTDP is a contraction mapping.

For any state $\boldsymbol{s} = (s_{\mathrm{PHY}}, s_{\mathrm{APP}})$, rewrite the mappings at different layers in (45) and (46), shown at the bottom of the page.

It is easy to verify that for any two different value functions V and $\tilde{V}$ as in (47) and (48), shown at the top of the next page, where $\|\cdot\|_\infty$ is the maximum norm and I is the all-one vector $(1, 1, \ldots, 1)$. Similarly, we also have

$$F^{(s,a_{\mathrm{PHY}})}(\tilde{V}) \leq F^{(s,a_{\mathrm{PHY}})}(V) + \mu\|V - \tilde{V}\|_\infty \boldsymbol{I}. \tag{49}$$

Therefore

$$\left|F^{(s,a_{\mathrm{PHY}})}(V) - F^{(s,a_{\mathrm{PHY}})}(\tilde{V})\right| \leq \mu \left\|V - \tilde{V}\right\|_\infty \boldsymbol{I}. \tag{50}$$

As (50) holds for any $(V, \tilde{V})$, we have the following contraction condition:

$$\left\|F^{(s,a_{\mathrm{PHY}})}(V) - F^{(s,a_{\mathrm{PHY}})}(\tilde{V})\right\|_\infty \leq \mu \left\|V - \tilde{V}\right\|_\infty. \tag{51}$$

Similarly, it can be shown that

$$\|F^s(V_{\mathrm{APP}}) - F^s(\tilde{V}_{\mathrm{APP}})\|_\infty \leq \|V - \tilde{V}\|_\infty. \tag{52}$$

The contraction property at both layers guarantees the following iteration is a contraction mapping for both V_{APP} and V:

$$\begin{aligned} V_{\mathrm{APP}}(\boldsymbol{s}, a_{\mathrm{PHY}}) &= F^{(s,a_{\mathrm{PHY}})}(V) \\ V(s) &= F^{\boldsymbol{S}}(V_{\mathrm{APP}}). \end{aligned} \tag{53}$$

For V_{APP}, the mapping equals to $V_{\mathrm{APP}}(\boldsymbol{s}, a_{\mathrm{PHY}}) = F^{(s,a_{\mathrm{PHY}})}(F^s(V_{\mathrm{APP}}))$, which is defined as $F_{\mathrm{APP}}(V_{\mathrm{APP}})$; for V, the mapping equals to $V(\boldsymbol{s}) = F^{(\boldsymbol{S},a_{\mathrm{PHY}})}(F^{\boldsymbol{S}}(V)) \triangleq F(V)$.

$$F^{(s,a_{\mathrm{PHY}})}(V) = \max_{\substack{a_{\mathrm{APP}} \in A_{\mathrm{APP}} \\ Z_{\mathrm{MAC}} \in \mathcal{Z}_{\mathrm{MAC}}}} \left[\begin{array}{l} q(s_{\mathrm{APP}}, a_{\mathrm{APP}}, Z_{\mathrm{MAC}}) - \lambda^b \omega_{\mathrm{APP}}(s_{\mathrm{APP}}, Z_{\mathrm{MAC}}) - \lambda^a_{\mathrm{APP}} c_{\mathrm{APP}}(s_{\mathrm{APP}}, a_{\mathrm{APP}}) \\ +\mu \sum_{\substack{s'_{\mathrm{PHY}} \in S_{\mathrm{PHY}} \\ s'_{\mathrm{APP}} \in S_{\mathrm{APP}}}} p_{\mathrm{PHY}}(s'_{\mathrm{PHY}} | s_{\mathrm{PHY}}, a_{\mathrm{PHY}}) p_{\mathrm{APP}}(s'_{\mathrm{APP}} | s_{\mathrm{APP}}, a_{\mathrm{APP}}) V(s'_{\mathrm{PHY}}, s'_{\mathrm{APP}}) \end{array}\right] \tag{45}$$

$$F^s(V_{\mathrm{APP}}) = \max_{a_{\mathrm{PHY}} \in A_{\mathrm{PHY}}} \left[-\lambda^a_{\mathrm{PHY}} c_{\mathrm{PHY}}(s_{\mathrm{PHY}}, a_{\mathrm{PHY}}) + V_{\mathrm{APP}}(s_{\mathrm{PHY}}, s_{\mathrm{APP}}, a_{\mathrm{PHY}})\right]. \tag{46}$$

$$\begin{aligned}
F^{(s,a_{\mathrm{PHY}})}(V) &= \max_{\substack{a_{\mathrm{APP}}\in A_{\mathrm{APP}}\\ Z_{\mathrm{MAC}}\in \mathcal{Z}_{\mathrm{MAC}}}} \left[\begin{array}{c} q(s_{\mathrm{APP}},a_{\mathrm{APP}},Z_{\mathrm{MAC}}) - \lambda^b \omega_{\mathrm{APP}}(s_{\mathrm{APP}},Z_{\mathrm{MAC}}) - \lambda^a_{\mathrm{APP}} c_{\mathrm{APP}}(s_{\mathrm{APP}},a_{\mathrm{APP}}) \\ +\mu \sum_{\substack{s'_{\mathrm{PHY}}\in S_{\mathrm{PHY}}\\ s'_{\mathrm{APP}}\in S_{\mathrm{APP}}}} p_{\mathrm{PHY}}(s'_{\mathrm{PHY}} \mid s_{\mathrm{PHY}},a_{\mathrm{PHY}}) p_{\mathrm{APP}}(s'_{\mathrm{APP}}|s_{\mathrm{APP}},a_{\mathrm{APP}}) V(s'_{\mathrm{PHY}},s'_{\mathrm{APP}}) \end{array}\right] \\
&= F^{(s,a_{\mathrm{PHY}})}(\tilde{V}) + \max_{\substack{a_{\mathrm{APP}}\in A_{\mathrm{APP}}\\ Z_{\mathrm{MAC}}\in \mathcal{Z}_{\mathrm{MAC}}}} \\
&\quad \left[\mu \sum_{\substack{s'_{\mathrm{PHY}}\in S_{\mathrm{PHY}}\\ s'_{\mathrm{APP}}\in S_{\mathrm{APP}}}} p_{\mathrm{PHY}}(s'_{\mathrm{PHY}}|s_{\mathrm{PHY}},a_{\mathrm{PHY}}) p_{\mathrm{APP}}(s'_{\mathrm{APP}}|s_{\mathrm{APP}},a_{\mathrm{APP}}) (\tilde{V}(s'_{\mathrm{PHY}},s'_{\mathrm{APP}}) - V(s'_{\mathrm{PHY}},s'_{\mathrm{APP}}))\right] \\
&\le F^{(s,a_{\mathrm{PHY}})}(\tilde{V}) + \mu \|V-\tilde{V}\|_\infty I
\end{aligned} \tag{47}$$

$$\begin{aligned}
&\left|V^{t(s,i)+1}(s) - V^{*(t(s,i))}(s)\right| = \left|F(V^{t(s,i)}) - F\left(V^{*(t(s,i))}\right)\right| \\
&\le \mu \left\|V^{t(s,i)} - V^{*(t(s,i))}\right\|_\infty \le \mu \left\{\left\|V^{t(s,i)} - V^{*(t(s,i-1))}\right\|_\infty + \left\|V^{*(t(s,i-1))} - V^{*(t(s,i))}\right\|_\infty\right\} \\
&< \mu \left\{\left\|V^{t(s,i)} - V^{*(t(s,i-1))}\right\|_\infty + \delta\right\} \le \mu \left\{\left\|V^{t(s,i-1)+1} - V^{*(t(s,i-1))}\right\|_\infty + \delta\right\} \\
&\cdots \\
&\le \mu^i \left\|V^{t(s,1)+1} - V^{*(t(s,1))}\right\|_\infty + \frac{\mu\left(1-\mu^i\right)}{1-\mu}\delta \triangleq \mu^i \left\|V^{t(s,1)+1} - V^{*(t(s,1))}\right\|_\infty + \delta
\end{aligned} \tag{48}$$

Therefore, both V_{APP} and V converge under the synchronous update defined by (25) and (26). According to Lemma 2, since we guaranteed infinite visits to all states, then the asynchronous update as in Table I also converge to the fixed point (i.e., the optimum).

APPENDIX D
PROOF OF PROPOSITION 2

Proof of Proposition 2: Let $V^{*(t)}_{\mathrm{APP}}$ and $V^{*(t)}$ denotes the optimal subvalue functions based on the estimates of transition probabilities $p^t(s'_{\mathrm{PHY}}|s_{\mathrm{PHY}},a_{\mathrm{PHY}})$ and $p^t(s'_{\mathrm{APP}}|s_{\mathrm{APP}},a_{\mathrm{APP}})$ in time slot t. It is easy to tell that $\lim_{t\to\infty} V^{*(t)}_{\mathrm{APP}} \overset{w.p.1}{\to} V^*_{\mathrm{APP}}$ and $\lim_{t\to\infty} V^{*(t)} \overset{w.p.1}{\to} V^*$. For any small δ, assume $\|V^{*(t)}_{\mathrm{APP}} - V^*_{\mathrm{APP}}\|_\infty < \delta$, for $t > T_{\mathrm{APP}}(\delta)$; and $\|V^{*(t)} - V^*\|_\infty < \delta$, for $t > T(\delta)$.

Similar to the definition in Proposition 1, let $t(\boldsymbol{s}, i)$ denote the time that $\boldsymbol{s}$ is visited for the ith time after $T_{\mathrm{APP}}(\delta)$, then we first prove that the following inequality holds for any state $\boldsymbol{s}$:

$$\begin{aligned}
&\left|V^{t(\boldsymbol{s},i)+1}(\boldsymbol{s}) - V^{*(t(\boldsymbol{s},i))}(\boldsymbol{s})\right| \\
&\quad < \mu^i \left\|V^{t(\boldsymbol{s},1)+1} - V^{*(t(\boldsymbol{s},1))}\right\|_\infty + \delta \\
&\qquad \text{for any small } \delta.
\end{aligned} \tag{54}$$

We use induction to prove it in (48).

Therefore, since $\|V^{t(s,1)+1} - V^{*(t(s,1))}\|_\infty$ is bounded, $\lim_{i\to\infty} \|V^{t(s,i)+1} - V^{*(t(s,i))}\|_\infty < \delta$, for any small δ.

Finally, as $\lim_{t\to\infty} V^{*(t)} \overset{w.p.1}{\to} V^*$, we can draw our conclusion that

$$\lim_{i\to\infty} V^{t(s,i)}(\boldsymbol{s}) \to V^*(\boldsymbol{s}), \quad w.p.1. \tag{55}$$

Similar result can be obtained for V_{APP}. ■

REFERENCES

[1] M. van der Schaar and S. Shankar, "Cross-layer wireless multimedia transmission: Challenges, principles, and new paradigms," *IEEE Wireless Commun. Mag.*, vol. 12, no. 4, Aug. 2005.

[2] *Multimedia Over IP and Wireless Networks: Compression, Networking, and Systems*, M. van der Schaar and P. Chou, Eds. New York: Academic, 2007.

[3] Q. Liu, S. Zhou, and G. B. Giannakis, "Cross-layer combing of adaptive modulation and coding with truncated ARQ over wireless links," *IEEE Trans. Wireless Commun.*, vol. 3, no. 5, pp. 1746–1755, May 2005.

[4] Y. J. Chang, F. T. Chien, and C. C. Kuo, "Cross-layer QoS analysis of opportunistic OFDM-TDMA and OFDMA networks," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 657–666, May 2007.

[5] M. van der Schaar, Y. Andreopoulos, and Z. Hu, "Optimized scalable video streaming over IEEE 802.11 a/e HCCA wireless networks under delay constraints," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 755–768, Jun. 2006.

[6] F. Fu and M. van der Schaar, "A new systematic framework for autonomous cross-layer optimization," *IEEE Trans. Veh. Technol.*, vol. 58, no. 4, pp. 1887–1903, Apr. 2009.

[7] A. Albanese and M. Luby, "PET-priority encoding transmission," in *High-Speed Networking for Multimedia Application*. Boston, MA: Kluwer, 1996.

[8] D. S. Turaga and T. Chen, "Hierarchical modeling of variable bit rate video sources," in *Proc. 11th Packet Video Workshop*, 2001.

[9] Q. Zhang and S. A. Kassam, "Finite-state Markov model for Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 47, no. 11, Nov. 1999.

[10] D. Krishnaswamy, "Network-assisted link adaptation with power control and channel reassignment in wireless networks," in *Proc. 3G Wireless Conf.*, 2002, pp. 165–170.

[11] W. Chen, U. Mitra, and M. J. Neely, "Energy-efficient scheduling with individual packet delay constraints over a fading channel," in *Wireless Networks*. New York: Springer, 2008.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[13] A. G. Barto, S. J. Bradtke, and S. P. Singh, "Learning to act using real-time dynamic programming," in *Artificial Intelligence*. New York: Elsevier, 1995.

[14] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[15] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

[16] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvari, "Convergence results for single-step on-policy reinforcement-learning algorithms," in *Machine Learning*. New York: Springer, 2000.

[17] N. Mastronarde and M. van der Schaar, "Towards a general framework for cross-layer decision making in multimedia systems," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.

[18] IEEE 802.11e/D5.0, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS) Jun. 2003, draft supplement, 802.11e/D5.0.

[19] C. Curescu and S. Nadjm-Tehrani, "Time-aware utility-based resource allocation in wireless networks," *IEEE Trans. Parallel Distrib.*, vol. 16, no. 7, pp. 624–636, May 2005.

[20] L. Breiman, "Probability, classic in applied mathematics," *Soc. Indust. Appl. Math.*, 1992.

[21] A. J. Goldsmith and S. G. Chua, "Adaptive coded modulation for fading channels," *IEEE Commun. Mag.*, vol. 46, pp. 595–602, May 2007.

[22] Y. J. Chang, F. T. Chien, and C. C. Kuo, "Cross-layer QoS analysis of opportunistic OFDM-TDMA and OFDMA networks," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 657–666, May 2007.

[23] V. Srivastava and M. Motani, "Cross-layer design: A survey and the road ahead," *IEEE Commun. Mag.*, vol. 43, no. 12, pp. 112–119, Dec. 2005.

[24] R. Hamzaoui, V. Stankovic, and Z. Xiong, "Optimized error protection of scalable image bit streams," *IEEE Signal Process. Mag.*, vol. 22, no. 6, pp. 91–107, Nov. 2005.

[25] F. Zhai, Y. Eisenberg, and A. K. Katsaggelos, "Joint source-channel coding for video communications," in *Handbook of Image and Video Processing*, 2nd ed. New York: Elsevier, 2000.

[26] D. V. Djonin and V. Krishnamurthy, "Q-Learning algorithms for constrained Markov decision processes with randomized monotone policies: Application to MIMO transmission control," *IEEE Trans. Signal Process.*, pp. 2170–2181, 2007.

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

[28] T. Stockhammer and M. Bystrom, "H.264/AVC data partitioning for mobile video communication," in *Proc. IEEE Int. Conf. Image Process.*, Singapore, Oct. 2004, pp. 545–548.

[29] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge , U.K.: Cambridge Univ. Press, 2004.

[30] T. Holliday, A. Goldsmith, and P. Glynn, "Optimal power control and source-channel coding for delay constrained traffic over wireless channels," in *Proc. IEEE Int. Conf. Commun.*, May 2002, pp. 831–835.

[31] P. Sadeghi, R. Kennedy, P. Rapajic, and R. Shams, "Finite-state Markov modeling of fading channels," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 57–80, Sep. 2008.

[32] X. Wang, Q. Liu, and G. B. Giannakis, "Analyzing and optimizing adaptive modulation coding jointly with ARQ for QoS-guaranteed traffic," *IEEE Trans. Veh. Technol.*, vol. 56, no. 2, Mar. 2007.

[33] Y. J. Chang, F. T. Chien, and C. C. Kuo, "Cross-layer QoS analysis of opportunistic OFDM-TDMA and OFDMA networks," *IEEE J. Select. Areas Commun.*, vol. 25, no. 4, pp. 657–666, May 2007.

[34] *High-Speed Physical Layer in the 5 GHz Band*, IEEE Std., 802.11a-1999, 1999.

[35] V. Kawadia and P. R. Kumar, "Principles and protocols for power control in wireless ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, Jan. 2005.

[36] D. Djonin and V. Krishnamurthy, "MIMO transmission control in fading channels—A constrained Markov decision process formulation with monotone randomized policies," *IEEE Trans. Signal Process.*, vol. 55, no. 10, pp. 5069–5083, Oct. 2007.

[37] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proc. IEEE*, vol. 95, no. 1, pp. 255–312, Jan. 2007.

[38] J. W. Lee, R. R. Mazumdar, and N. B. Shroff, "Non-convex optimization and rate control for multi-class services in the internet," *IEEE/ACM Trans. Netw.*, vol. 13, no. 4, pp. 827–840, Aug. 2005.

[39] E. Maani, P. Pahalawatta, R. Berry, T. N. Pappas, and A. K. Katsaggelos, "Resource allocation for downlink multiuser video transmission over wireless lossy networks," *IEEE Trans. Image Process.*, vol. 17, no. 9, pp. 1663–1671, Sep. 2008.

[40] B. Girod, M. Kalman, Y. Liang, and R. Zhang, "Advances in channel-adaptive video streaming," *Wireless Commun. Mobile Comput.*, vol. 2, no. 6, pp. 549–552, Sep. 2002.

[41] H. P. Shiang and M. van der Schaar, "Multi-user video streaming over multi-hop wireless networks: A distributed, cross-layer approach based on priority queuing," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 770–785, May 2007.

Yu Zhang (S'08) received the Bachelor's and Master's degrees from Tsinghua University, Beijing, China, in 2006 and 2008, respectively.

He is currently working toward the Ph.D. degree with the Department of Electrical Engineering, University of California, Los Angeles.

Fangwen Fu (S'08) received the Bachelor's and Master's degrees from Tsinghua University, Beijing, China, in 2002 and 2005, respectively.

He is currently working toward the Ph.D. degree with the Department of Electrical Engineering, University of California, Los Angeles.

Mihaela van der Schaar (SM'04–F'10) received the Ph.D. degree from Eindhoven University of Technology, The Netherlands, in 2001.

She is now an Associate Professor with the Electrical Engineering Department, University of California, Los Angeles.