Multimedia Wireless Transmission with Network Coding

Dong Nguyen*, Thinh Nguyen*, Xue Yang†

*School of EECS, Oregon State University
Corvallis, OR 97331, USA
Email:{nguyendo, thinhq}@eecs.oregonstate.edu
†Communication Technology Lab
Intel Corporation
Email:xue.yang@intel.com

***Abstract*— In recent years, Wireless Local Area Networks (WLAN) have become the premier choice for many homes and enterprises. WiMAX (Worldwide Interoperability for Microwave Access) has also emerged as the wireless standard that aims to deliver data over long distances, and can potentially provide wireless broadband access as an alternative to the wired cable and DSL networks. Parallel with the surge of wireless networks is the explosive growth of multimedia applications. Therefore, it is important to explore efficient methods for delivering multimedia data in such wireless settings. In this paper, we propose a network coding based scheduling policy to be used at WLAN-like Access Point (AP) or at a WiMAX-like broadcast station that optimizes the multimedia transmission in both broadcast and unicast settings. In particular, the contributions of this paper include (a) a framework for increasing the bandwidth efficiency of broadcast and unicast sessions in a wireless network based on network coding techniques and (b) an optimized scheduling algorithm based on the Markov Decision Process (MDP) to maximize the quality of multimedia applications. Simulations and theoretical results demonstrate the advantages of our approach over the conventional techniques.**

I. Introduction

Although there has been a tremendous growth in multimedia applications over the Internet, packet loss, delay, and time-varying bandwidth of the Internet have hindered many high quality multimedia applications. These problems manifest more so in wireless networks, which often exhibit higher loss rate and lower bandwidth. Arguably, all these problems will disappear if bandwidth is infinite or is well provisioned. Unfortunately, the current *best effort* Internet and wireless networks provide neither. Even if the Internet is redesigned entirely to provide proper bandwidth provision mechanisms, doing so may introduce scalability and complexity issues, resulting in degraded performance. Thus, many *above network layer* approaches to multimedia streaming over the Internet and wireless networks have been proposed to deal with packet loss, delay, and time-varying bandwidth, ranging from transport protocols and packet scheduling algorithms [5][2] to source and channel coding techniques [15][21].

That said, for a number of video streaming applications, their bandwidth requirements are sufficiently small that even without employing sophisticated techniques, a few of these applications can run concurrently over the existing wireless standards (IEEE 802.11(b) and (g)). On the other hand, these standards may not be able to support multimedia applications with much larger bandwidth requirement, e.g., DVD quality video streaming applications. Despite of the fact that wireless bandwidth has been increasing significantly, from a theoretical limit of 11Mbps for 802.11b to 54 Mbps for 802.11g, and to 540 Mbps for 802.11n, there has always been high bandwidth demand resulting from new applications. For example, many wireless devices and multimedia applications have been developed, ranging from MP3 streaming on wireless-ready iPods to video conferencing via laptops. In the near future, IPTV and Video on Demand (VoD) applications will rely on wireless network to deliver high quality videos from the Internet to any TV set or computers at home through a wireless AP. In addition to home networks, WiMAX (Worldwide Interoperability for Microwave Access) has emerged as the wireless standard that aims to deliver wireless data over long distances, and can potentially provide wireless broadband access as an alternative to cable and DSL. With the wireless broadband access, there will be potentially more users (homes), leading to higher bandwidth demand and greater bandwidth constraint. Therefore, it is imperative that an efficient bandwidth sharing/competing scheme among the wireless applications should be employed to enable each application meeting its bandwidth and delay requirements.

Parallel to the advances of wireless technologies is the development of network coding paradigm, which allows a source to disseminate information to multiple destinations efficiently for a given network topology. In a traditional *forward and store* network, packets are forwarded hop-by-hop, unmodified from a source to a destination. On the other hand, network coding techniques allow an intermediate node to combine the data from different input links before sending the combined data on its output links. For many problems such as multicast and broadcast, using appropriate encoding schemes at each intermediate nodes (typically linear combination of input data) can achieve the network capacity. Although the original network coding problem is formulated in the

context of a wireline network, recently, it has been used to reduce the energy consumption and increase the capacity in wireless ad hoc networks. For example, in [6], Fragouli *et al.* provided an overview of network coding and its applications in wireless networks. Wu *et al.* also showed how network coding can be used to improve the capacity of information exchange in a wireless ad hoc network [24].

Following many successes on applying network coding techniques to wireless ad hoc networks, this paper proposes a new network coding technique to improve the overall bandwidth efficiency while optimizing multiple concurrent multimedia applications with heterogeneous requirements in a WLAN/WiMAX network. In particular, we propose a network coding based scheduling policy to be used at a WLAN-like Access Point (AP) or at a WiMAX-like broadcast station that optimizes the multimedia transmission in both broadcast and unicast settings. Throughout this paper, we will use the terms WiMAX and WLAN to refer to the modified WiMAX and WLAN to support our proposed scheduling policy. The contributions of this paper include (a) a framework for increasing the bandwidth efficiency of broadcast and unicast sessions in a wireless network based on network coding techniques and (b) an optimized scheduling algorithm based on the Markov Decision Process (MDP) to maximize the quality of multimedia applications. We first discuss some preliminaries in Section II. In Section III, we present a basic network coding based retransmission scheme that improves the bandwidth efficiency of broadcast and unicast sessions in a one-hop wireless network. Next, we present network coding based scheduling policy using MDP that optimizes multiple concurrent flows under bandwidth constraint. Simulation results and discussions are provided in Section IV. Finally, we conclude with a few remarks in Section V.

II. Preliminaries

To aid our subsequent discussions, in this section, we now present a brief introduction to multimedia streaming, MDP, and network coding for wireless networks.

A. Multimedia Streaming

Multimedia streaming over best-effort, packet-switched networks is challenging due to a number of factors such as high bit rates, delay, and loss sensitivity. As such, many approaches have been proposed ranging from network protocols to source and channel coding techniques. From channel coding perspective, Forward Error Correction (FEC) techniques have been proposed to increase reliability at the expense of bandwidth expansion [12][15][3][13]. From source coding perspective, error-resilient coding techniques have been explored to allow the quality of a video to be degraded gracefully in lossy environments [22][18][19]. In addition, layered video coding techniques have been proposed to deal with heterogeneity and time-varying nature of the Internet by adapting its bit rate to the available bandwidth [7][11][17]. A layered video bit stream is organized into a number of layers with the base layer being the most important layer in terms of the visual quality contribution. Other layers are used to enhance the video quality with more layers resulting in higher video quality.

Depending on the coding techniques, a video bit stream is composed of different types of bits. Each type generally contributes a different amount of enhancement toward the final reconstructed video quality. Furthermore, some bits are only useful when other bits are present. This property leads to much research on the packet scheduling algorithms that choose which packets to send at which times, in order to produce in the best possible reconstructed video quality under insufficient network resources. Notably, the rate-distortion, MDP-based optimization approach to packet scheduling has produced many fruitful results in the past several years [4][2][9]. The main idea of this approach is that using the observations at every single step, the scheduling algorithm chooses the best action to perform (e.g., whether to send a packet or not and which packet to send) in order to maximize the *expected* video quality under limited network resources. The optimal sequence of actions during duration of interest is the solution to the MDP problem which can be efficiently solved in many settings. We now provide a brief introduction to MDP.

B. Markov Decision Process

Let us consider a decision maker or a controller who, at every time step, is in charge of making a decision or choosing an action, which can influence the evolution of a probabilistic system. Assuming that the state of the system evolves in discrete time steps, then the goal of the controller is to choose a sequence of actions that maximizes some cumulative system performance metrics (rewards) at the end of some finite or infinite number of time steps. Since the system states and the performance metrics depend on the chosen action at every time step, it is wise for the controller to consider the future states and the associated rewards in the decision making process at the present state. Finding the optimal sequence of actions is the solution to the MDP problem.

That said, an abstract MDP represents a dynamic system and is specified by a finite set of states S, representing the possible states of the system, a set of control actions A, a transition probability P, and a reward function r. The transition probability specifies the dynamics of the system, and gives the probability $P(s'|s,a)$ of transitioning to state s' after taking action a in state s. The dynamics are Markovian in the sense that the probability of the next state s' depends only on the current state s and action a, and not on any previous history. The reward function assigns a real number to the current state s and the action a taken in that state, so that $r(s,a)$ represents the immediate reward of being in state s and taking action a. A policy π is a mapping from states to actions, which defines a controller that takes actions as specified by the policy. We assume that time is discrete and that the control policy selects one action at each time step. Every policy

π is associated with a value function V^π such that $V^\pi(s)$ gives the expected cumulative reward achieved by π when starting in state s. The solution to a MDP problem is an optimal policy that maximizes the expected cumulative reward over any finite or infinite number of time steps.

When a MDP ends in a finite number of time steps N, we call it a finite-horizon MDP. Mathematically, let us denote S as a set of all the finite states and A a set of all possible actions. Let d_t denote a decision rule, prescribing a procedure for action selection in each state at a specified time step t. In other words, decision rules are functions $d_t : S \rightarrow A$, which specify the choice of action when the system occupies state s at time step t. For each $s \in S$, $d_t(s) = a_t \in A$. A policy $\pi = (d_1, d_2, d_3, ...d_N)$ is a sequence of actions at every time step. Thus, a sample path of the system evolution is a sequence of states and actions pairs $\omega = (s_1, a_1, s_2, ..., a_{N-1}, s_N)$ and the probability of this sample path under policy π is:

$$\begin{aligned} P^\pi(\omega) &= P(s_1, a_1, ..., a_{N-1}, s_N) \\ &= P_1(s_1)p_1(s_2|s_1, a_1)...p(s_N|s_{N-1}, a_{N-1}), \end{aligned} \quad (1)$$

where $P_1(s_1)$ and $p_t(s_{t+1}|s_t, a_t)$ specify the initial distribution of the states and the transition probability from state s_t to s_{t+1} when action a_t is taken. Let $V(\omega)$ denote a real-valued function of ω, then the expected value of V under policy π is:

$$\begin{aligned} E^\pi\{V\} &= \sum_{\omega \in (S \times A)^N} V(\omega)P^\pi(\omega) \quad (2) \\ &= \sum_{v \in R^1} vP^\pi(\omega : V(\omega) = v). \quad (3) \end{aligned}$$

Now, let $r_t(s, a)$ denote an immediate reward of taking action in a in state s at time t, then the total reward corresponding a sample path $\omega = (s_1, a_1, s_2, a_2, ...a_{N-1}, s_N)$ is:

$$V(s_1, a_1, ..., s_N) = \sum_{t=1}^{N-1} r_t(s_t, a_t) + r_N(s_N), \quad (4)$$

where $r_N(s_N)$ is the optional reward in the final state. Even though there is no action taken in the final state, one can assign an immediate reward. Therefore, given a distribution of the states $P(s)$ and policy π, the expected total reward of a MDP is:

$$\begin{aligned} \overline{V}^\pi &= E^\pi\{V(\omega)\} \\ &= E^\pi\left\{\sum_{t=1}^{N-1} r_t(s_t, a_t) + r_N(s_N)\right\}, \quad (5) \end{aligned}$$

In a typical scenario, it is useful to find the expected total reward given a starting state $s_1 = s$. We may denote this expected reward as:

$$\overline{V}_s^\pi = E_s^\pi\left\{\sum_{t=1}^{N-1} r_t(s_t, a_t) + r_N(s_N)\right\}, \quad (6)$$

where the expectation is taken over a new distribution of the sample paths $P^\pi(\omega)$ as:

$$\begin{aligned} P^\pi(\omega) &= P(s, a_1, ..., a_{N-1}, s_N) \\ &= p_1(s_2|s, a_1)...p(s_N|s_{N-1}, a_{N-1}), \quad (7) \end{aligned}$$

as a result of setting $P(s_1 = s) = 1$ in (1).

Typically, $\overline{V}_s^\pi$ can be efficiently computed using dynamic programming by denoting U_t^π as the total expected reward obtained by using policy π from the time t, $t+1$, ... $N-1$. Thus, for $t < N$, we have

$$U_t^\pi(s_t) = E_{s_t}^\pi\left\{\sum_{n=t}^{N-1} r_n(s_n, a_n)) + r_N(s_N)\right\}. \quad (8)$$

Clearly,

$$\overline{V}_s^\pi = U_1^\pi(s) \quad (9)$$

Now, one can compute $U_1^\pi(s)$ using the following recursive equation:

$$\begin{aligned} U_t^\pi(s_t) &= r_t(s_t, a_t) + E_{s_t}^\pi\left\{U_{t+1}^\pi(s_{t+1})\right\} \\ &= r_t(s_t, a_t) + \sum_{j \in S} p(j|s_t, a_t)U_{t+1}^\pi(j). \quad (10) \end{aligned}$$

This equation is intuitive as it indicates that the expected *remaining* reward is equal to the sum of the immediate reward and the expected *remaining* reward of the next state.

Based on (10), it can be shown that the optimal policy $\pi^* = (d^*(s_1), d^*(s_2), ..., d^*(s_N))$ that maximizes the expected reward $\overline{V}_s^\pi$ can be solved using the following Backward Induction Algorithm [16]:

1) Set $t = N$ and $U_N^*(s_N) = r_N(s_N)$ for all $s_N \in S$,
2) Substitute $t-1$ for t and compute $U_t^*(s_t)$ for each $s_t \in S$ by

$$U_t^*(s_t) = \max_{a \in A}\left\{r_t(s_t, a) + \sum_{j \in S} p(j|s_t, a)U_{t+1}^*(j)\right\}.$$

Set

$$d^*(s_t) = \arg\max_{a \in A}\left\{r_t(s_t, a) + \sum_{j \in S} p(j|s_t, a)U_{t+1}^*(j)\right\}.$$

3) If $t = 1$, stop. Otherwise return to step 2.

Having presenting a MDP formulation and an efficient solution using Backward Induction Algorithm, we note that solving a typical MDP problem involving two tasks: modeling and selection of solution tools. In the modeling task, a particular real-world problem is translated into an abstract MDP problem. This involves modeling the states, the actions, the immediate rewards, the transition probabilities, and the desired objective [1]. This modeling process can be hard, and often require domain experts. Even when the states are well modeled, computing the transition probabilities or selecting an appropriate reward function to fit a particular purpose can be difficult. In other cases, representing the system states accurately may require a large state and action spaces, making it hard to solve a MDP in practice. Thus, it is necessary for one to use appropriate approximate algorithms in order to solve the problem in a reasonable amount of time. Readers

[1]The desire objective does not have to the sum of all the immediate rewards. A popular reward is the discount reward where the future reward is weighted less than the current reward.

may refer to [8][16] for the details on Markov Decision Process.

In Section III, we present a MDP model used to describe our network coding based multimedia streaming via a wireless AP. But first, we provide a short introduction on network coding for wireless networks.

C. Network Coding for Wireless Networks

The original network coding problem is formulated for wireline networks by Ahlswede *et al.*[1] which shows that maximum broadcast capacity in a network can be achieved by appropriate mixing of data at the intermediate nodes. Recently, network coding approaches to wireless network have also been investigated for reducing the energy consumption and increasing the capacity. A short overview of network coding and its applications in wireless networks can be found in [6]. Many of network coding techniques for wireless networks focus on improving the capacity of information exchange in a wireless ad hoc network [24][25][10][23] using XOR operations, a form of network coding. In a XOR scheme, two nodes R_1 and R_2 are assumed to exchange their information with each other through a node . Specifically, a packet a sent by node R_1 to node R_2 is relayed by node R. Similarly, packet b sent by node R_2 to node R_1 is relayed by node R. As a result, node R has both packets a and b. Traditionally, node R has to perform two transmissions: one for relaying packet a and one for relaying packet b. On the other hand, when using the network coding scheme with XOR operations, the total number of transmissions can be reduced as follows. Since node R_1 already has packet a, and node R_2 already has packet b, node R can simply broadcast a single packet $a \oplus b$ to both nodes R_1 and R_2. Upon receiving this packet, node R_1 can obtain packet b as $b = a \oplus (a \oplus b)$. Node R_2 can also recover packet a as $b = b \oplus (a \oplus b)$. Thus, with one broadcast from $R3$, both $R1$ and $R2$ can receive the desired packet. Figure 1(a) shows the traditional method with 4 transmissions while Figure 1(b) shows the network coding approach with the number of transmissions reduced to 3.

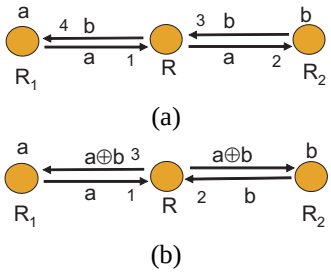

Fig. 1. *(a) Traditional wireless information exchange transmission in an ad hoc network; (b) Wireless information exchange transmission with network coding.*

III. Wireless Streaming via An Access Point

A. Model and Assumption

We now describe the broadcast and unicast models in WLAN and WiMAX networks. Based on this model, we show how network coding and MDP can be used to increase the bandwidth efficiency while optimizing the concurrent applications based on their requirements. In particular, we are interested in designing a packet scheduling algorithm running at a WLAN AP or WiMAX broadcast station that optimizes multiple concurrent wireless applications as shown in Figure 2. In this setting, we assume that data of different applications flow mostly in one direction, from the AP to the users. A user can be a home receiving data via a WiMAX base station, or an 802.11x device receiving data via a WLAN AP. These applications may have different bandwidth, delay, and loss requirements. For example, video streaming applications may require higher bandwidth and shorter delay than that of file downloading applications. However for clarity in this paper, we present an optimized packet scheduling algorithm, designed exclusively for video broadcast and unicast flows from the AP to one or more receivers. The objective of the algorithm is to maximize the visual quality of videos received at the receivers under a certain bandwidth constraint.

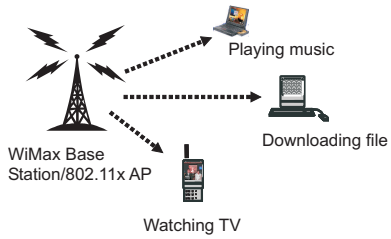

Fig. 2. *A typical setting for Internet applications via an 802.11 AP or WiMAX broadcast station. Each application has a different bandwidth, loss, and delay requirements.*

We make the following assumptions for our model:

1) There are K receivers R_1, R_2, ..., R_K.
2) The AP has a set $\Omega = \{l_1, l_2, ..., l_M\}$ of M packets that needed to be delivered to the receivers after some time slots N. In a broadcast setting, all the receivers request all M packets, while in a unicast setting, each receiver would request different subsets of Ω. In a semi-broadcast setting, there are two or more receivers requesting the same subset of Ω.
3) There is a limit on the total number of time slots N used to transmit these M packets. After N time slots, the AP moves to the next batch of M packets, regardless of whether or not all M packets have been successfully received at the intended receivers. Thus, the quality of the media streams at the receivers might be reduced by some amount proportional to the number and the type of the unavailable packets. This mechanism essentially imposes a bandwidth requirement on the streamed video.
4) Any receiver can cache packets transmitted from the AP to other receivers, even though those packets are not directly useful to itself. We assume that appropriate encryption is employed to provide privacy of a receiver.

5) Data is assumed to be sent in packets, and each packet is sent in a time slot of fixed duration.
6) The AP is assumed to know which packet from which receiver is lost. This can be accomplished through the use of positive and negative acknowledgments (ACK/NAKs). If an ACK or NAK is lost, the corresponding data packet is also considered lost.
7) The distribution of packet loss at a receiver R_i follows the Bernoulli distribution with parameter p_i. This model is clearly insufficient to describe many real-world scenarios. However, it is only intended to capture the essence of the problem. One can develop a more accurate model, albeit complicate analysis. In [14], we provide an analysis for a slightly more accurate model that reflects the correlated loss among the receivers.

Before formulating our packet scheduling algorithm as a MDP, let us consider the advantage of using network coding in the WiMAX/802.11x settings.

B. Wireless Transmission With Network Coding

We consider a broadcast scenario. For simplicity, suppose 2 packets a and b are broadcasted from a wireless AP to 2 receivers R_1 and R_2. Let us first examine a traditional (non-network coding) broadcast scheme in which, packets are sent in time slots. In a 802.11x network, if a packet is received correctly, the AP should receive an ACK within an appropriate amount of time after the data packet is sent. Otherwise, the data packet is considered lost, and must be retransmitted. Using this scheme, a packet loss at any receiver will require the AP to retransmit that packet. If two distinct lost packets at two different receivers, the AP will need at least two retransmissions as seen in Figure 3(a). In the first time slot, packet a is lost at R_1, but is received correctly at R_2. Therefore, in the second time slot, the AP rebroadcasts packet a, and R_1 receives the packet a correctly at this time. In the third time slot, the AP sends packet b which is lost at R_2, but is received correctly at R_1. In the fourth time slot, the AP rebroadcasts packet b, and R_2 receives b correctly this time. Using this scheme, a total of 4 transmissions are required to transmit both packets a and b to receivers R_1 and R_2. We now consider a network coding technique

Time slots	1	2	3	4
Packet sent	a	a	b	b
Events at R1,R2	Lost at R1, received at R2	Received at R1	Lost at R2, received at R1	Received at R2

(a)

Time slot	1	2	3
Packet sent	a	b	a⊕b
Events at R1,R2	Lost at R1, received at R2	lost at R2, received at R1	Received at both R1, R2

(b)

Fig. 3. *(a) Traditional wireless broadcast requiring a total of 4 transmissions to successfully transmit two packets to two users; (b) Wireless broadcast with network coding requiring only 3 transmissions.*

that requires only one retransmission to recover two lost packets at both receivers. Unlike the traditional scheme, using the network coding scheme, the AP does not retransmit the lost packet a at R_1 immediately. Instead, the AP continues to broadcast the next packet until there is a lost packet b at receiver R_2. At this time, the AP broadcasts the new packet $(a \oplus b)$ to both receivers. If R_1 has packet b but not a, and R_2 has packet a but not b, then both receivers will be able to reconstruct their missing packets by simply XORing the packet they have, with the packet $(a \oplus b)$. As shown in Figure 3(b), R_1 reconstructs a as $b \oplus (a \oplus b)$ and R_2 reconstructs b as $a \oplus (a \oplus b)$. Therefore, one retransmission from the AP will enable both receivers to correctly reconstruct their lost packets. In general, this network coding scheme can outperform the traditional scheme substantially when the loss patterns between many receivers are uncorrelated.

It is straightforward to extend the network coding technique to unicast setting. Assume that R_1 wants to receive packet a while R_2 wants to receive packet b. Clearly, if R_1 is willing to cache packet b intended for R_2, and R_2 is willing to cache packet a intended for R_1, then the two unicast sessions are now equivalent to a single broadcast session in the previous example. Similarly, when there are N receivers that want to receive N different packets, a receiver may want to cache everyone else's data in order to use network coding for higher bandwidth efficiency.

The key to improving bandwidth efficiency in one-hop wireless network is the efficient generation of XOR packets to enable all the receivers to recover their lost packets quickly. If the packet loss rate is low, the AP has fewer opportunities to rebroadcast the XOR packets of distinct lost packets at different receivers. Therefore, there is not much relative advantage in bandwidth efficiency of using network coding. At one extremity, if there is no lost packet from any receiver, then there is no opportunity to rebroadcast any XOR packet. Thus the performances of traditional and network coding schemes are identical. On the other hand, when the loss rates at the receivers are large, there are more opportunities for generating XOR packets. As a result, the bandwidth efficiency of network coding scheme is much higher than that of the traditional one.

One constraint with network coding scheme is that for higher bandwidth efficiency, longer delay of some packets may be necessary to allow packet losses to occur at other receivers, leading to more opportunities for the AP to generate the XOR packets. Therefore, this approach might not be acceptable for certain multimedia streaming applications where every packet has a playback deadline. Thus, the AP must consider the trade-off between the delay and bandwidth efficiency based on the application requirements.

The simple two receiver example above shows the essential advantage of network coding without providing a general algorithm for combining and resending the lost packets. Given K receivers with different packet loss rates, we are interested in a scheduling algorithm that determines which packets at which times to send, in order

to maximize a certain objective. In the next section, we find the optimal scheduling algorithms for both broadcast and unicast problems using the MDP framework.

C. Optimal MDP Based Packet Scheduling

As mentioned in Section II-B, solving a MDP problem typically involves two phases: modeling and solution tool selection. The modeling phase involves translating our wireless broadcast and unicast problems into abstract MDPs. In particular, we are going to define the set of the states S, the set of actions A, the immediate reward $r(s_t, a_t)$, derive the transition probabilities $P(s_{t+1}|s_t, a_t)$ and the cumulative reward, e.g., objective for the wireless broadcast and unicast settings.

Our packet scheduling algorithm works as follows. At every time step, the AP sends a packet, and waits for an ACK message. If a receiver receives a packet, an ACK is sent back immediately, similar to the 802.11x protocol. If no ACK is received within a specified time frame, the data packet is considered lost. The AP then can choose to send a new packet, retransmit a lost packet, or transmit a XOR packet. We now proceed to model our packet scheduling algorithm as a MDP of finite horizon N where N is the maximum number of allowable time slots to transmit M packets.

State Representation. At any given time slot, receiver R_i possesses a subset of packets belonged to Ω, including the packets that are intended for other receivers. This subset can be represented by an M-bit vector r_i as $(b_1^i, b_2^i, ..., b_M^i)^T$ where $b \in \{0, 1\}$. $b_i = 1$ indicates the presence of packet l_i at R_i, while $b_i = 0$ indicates otherwise. Since there are K receivers, a system configuration or state s can be represented by an $M \times K$ matrix with binary entries as:

$$s = \begin{pmatrix} b_1^1 & b_1^2 & ... & b_1^K \\ b_2^1 & b_2^2 & ... & b_2^K \\ ... & ... & ... & ... \\ b_M^1 & b_M^2 & ... & b_M^K \end{pmatrix} \tag{11}$$

Thus, there are $2^{M \times K}$ possible states.

Action Representation. At any given time slot, the AP can (a) broadcast any $l_i \in \Omega$, (b) broadcast any XOR packet resulting from XORing the distinct lost packets from different receivers, and (c) broadcast nothing. This implies that the maximum number of possible actions J at any time step:

$$J = M + \sum_{i=2}^{L} \begin{pmatrix} L \\ i \end{pmatrix} + 1, \tag{12}$$

where L denotes the number of packets which are lost at one or more receivers. The maximum number of lost packets L is M, however this case is extremely rare for large M.

Transition Probability. Given the Bernoulli model with parameter p_i for packet loss at each receiver R_i, it is straightforward to compute the transition probability $P(s_{t+1} = s'|s_t = s, a_t = a)$. For example, consider broadcasting two packets to two receivers, i.e., $K = 2$ and $M = 2$. Let us denote

$$s = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad s' = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

Suppose at time t, the system is in the state s, i.e., R_1 has packet l_1 and R_2 has packet l_2, then choosing action a = "send l_1" in state s will move the system to state s' with probability:

$$P(s_{t+1} = s'|s_t = s, a_t = a) = 0 \tag{13}$$

while choosing action a' = "send l_2" will move the system to state s' with probability:

$$P(s_{t+1} = s'|s_t = s, a_t = a') = 1 - p_1 \tag{14}$$

Reward Modeling. The immediate reward $r(s, a)$ for each state and action pair must be chosen such that the sum of these immediate rewards models our objective accurately. Since our objective is to optimize the quality of multimedia streaming applications, we model the immediate rewards as the sum of the reduction in distortion for one or more receivers upon receiving a particular packet. Thus, maximizing the overall reward is equivalent to minimizing the overall distortion for all the receivers' applications under a particular bandwidth constraint.

In our setting, taking an action a in a state s does not provide an explicit immediate reward. Rather, we know the explicit reward amount $r(s', s)$ when the system moves from state s to state s'. For example, given a detail profile of a layered video, we know the amount of distortion reduction contributed by each layer. If state s indicates that a receiver has layers 1 and 2, and state s' indicate the a receiver has layers 1, 2, and 3, then moving from state s to state s' would reward us an amount $r(s', s)$ equal to the distortion reduction contributed by layer 3. Since we know the transition probability between states under an action a, we can compute $r(s, a)$ as the expected immediate reward by taking action a as :

$$r(s, a) = \sum_{j \in S} P(j|s, a) r(j, s) \tag{15}$$

Remarks on state and action space. As presented, the number of states and actions can be exponentially large under certain settings. Thus, one needs to use approximated algorithms. Such approximated algorithms however, are beyond the scope of this paper. Instead, we focus on solving the MDP exactly using the Backward Induction Algorithm when the number of states and actions are sufficiently small to be tractable.

Example. We now present a simple example showing MDP formulations for broadcast and unicast settings with two receivers and two packets. For the state space, there would be a total of 16 states with each state s represented by:

$$s = \begin{pmatrix} b_1^1 & b_1^2 \\ b_2^1 & b_2^2 \end{pmatrix}.$$

$b_j^i = 1$ represents that receiver R_i has packet l_j, and $b_j^i = 0$ represents otherwise.

	States	1 00 00	2 01 00	3 10 00	4 11 00	5 00 01	6 01 01	7 10 01	8 11 01	9 00 10	10 01 10	11 10 10	12 11 10	13 00 11	14 01 11	15 10 11	16 11 11
1	00 00	p_1p_2	p_1* $(1\text{-}p_2)$	$(1\text{-}p_1)$ *p_2	$(1\text{-}p_1)$ *$(1\text{-}p_2)$	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	p_1	0	1- p_1	0	0	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	p_2	$1\text{-}p_2$	0	0	0	0	0	0	0	0	0	0	0	0
4	11 00	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
5	00 01	0	0	0	0	p_1p_2	p_1* $(1\text{-}p_2)$	$(1\text{-}p_1)$ *p_2	$(1\text{-}p_1)$ *$(1\text{-}p_2)$	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	p_1	0	1- p_1	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	p_2	$1\text{-}p_2$	0	0	0	0	0	0	0	0
8	11 01	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
9	00 10	0	0	0	0	0	0	0	0	p_1p_2	p_1* $(1\text{-}p_2)$	$(1\text{-}p_1)$ *p_2	$(1\text{-}p_1)$ *$(1\text{-}p_2)$	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	p_1	0	1- p_1	0	0	0	0
11	10 10	0	0	0	0	0	0	0	0	0	0	p_2	$1\text{-}p_2$	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	p_1p_2	p_1* $(1\text{-}p_2)$	$(1\text{-}p_1)$ *p_2	$(1\text{-}p_1)$ *$(1\text{-}p_2)$
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	p_1	0	1- p_1
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	p_2	$1\text{-}p_2$
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fig. 4. *Transition probability matrix associated with action "sending packet l_1".*

	States	1 00 00	2 01 00	3 10 00	4 11 00	5 00 01	6 01 01	7 10 01	8 11 01	9 00 10	10 01 10	11 10 10	12 11 10	13 00 11	14 01 11	15 10 11	16 11 11
1	00 00	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	p_2	0	0	0	$1\text{-}p_2$	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	p_1	0	0	0	0	0	0	0	$1\text{-}p_1$	0	0	0	0	0
4	11 00	0	0	0	p_1p_2	0	0	0	p_1* $(1\text{-}p_2)$	0	0	0	$(1\text{-}p_1)$ *p_2	0	0	0	$(1\text{-}p_1)$ *$(1\text{-}p_2$
5	00 01	0	0	0	0	p_2	$1\text{-}p_2$	0	0	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	p_1*p_2	p_1* $(1\text{-}p_2)$	0	0	0	0	0	0	$(1\text{-}p_1)$ *p_2	$(1\text{-}p_1)$ *$(1\text{-}p_2)$
8	11 01	0	0	0	0	0	0	0	p_1	0	0	0	0	0	0	0	1- p_1
9	00 10	0	0	0	0	0	0	0	0	p_1	0	1- p_1	0	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	p_1*p_2	0	$(1\text{-}p_1)$ *p_2	0	p_1* $(1\text{-}p_2)$	0	$(1\text{-}p_1)$ *$(1\text{-}p_2)$
11	10 10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	p_2	0	0	0	$1\text{-}p_2$
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	p_1*p_2	p_1* $(1\text{-}p_2)$	$(1\text{-}p_1)$ *p_2	$(1\text{-}p_1)$ *$(1\text{-}p_2)$
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	p_1	0	1- p_1
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	p_2	$1\text{-}p_2$
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fig. 5. *Transition probability matrix associated with action "sending packet $l_1 \oplus l_2$".*

As for the action space, at any time step, the AP can perform one of the four actions: send l_1, send l_2, send $l_1 \oplus l_2$, and send nothing.

To compute the transition probabilities, let us denote p_1 and p_2 as the packet loss probabilities at R_1 and R_2, respectively. For each action, there is an associated transition probability matrix. In this example, we show two transition probability matrices due to taking actions "sending l_1" and "sending $l_1 \oplus l_2$", respectively. The transition probability matrix for taking actions "sending l_2" and "not sending anything" can be similarly computed.

First let us consider the transition probability matrix for taking action "sending l_1". This is shown in Figure 4. An entry in row i and column j denotes the transition probability from state i to state j under action "sending l_1". For example, the probability of transition from state 1 to state 4 when sending packet l_1 is $(1-p_1)(1-p_2)$. The reason is as follows. Since state 1 denotes neither receivers have packets l_1, and the state 4 denotes both receivers have packets l_1, to transition from state 1 to state 4 by sending packet l_1, both receivers must have correctly received l_1, and the probability of this event equals to $(1-p_1)(1-p_2)$. Similarly, other transition probabilities for different states can be computed by using the packet loss probabilities at each receiver. Let us now consider the transition probability matrix for taking action "sending $l_1 \oplus l_2$". This action is interesting as one transmission by the AP can help two receivers to simultaneously recover two distinct lost packets. Consider a transition from state 10 to state 16 in Figure 5. In state 10, R_1 has l_2 but not l_1 while R_2 has l_1 but not l_2. If the AP sends packet $l_1 \oplus l_2$, and the packet is successfully received at both receivers, then both R_1 and R_2 will now obtain $l_1 = l_2 \oplus (l_1 \oplus l_2)$ and $l_2 = l_1 \oplus (l_1 \oplus l_2)$, respectively. The probability of this event is then equal to $(1-p_1)(1-p_2)$. Other probability entries can be calculated in a similar manner. Now for each action, there is an associated reward matrix. Let us denote r_{ij} as the immediate reward of R_i upon receiving packet l_j. It can be seen that for broadcast setting, $r_{11} = r_{21}$ and $r_{12} = r_{22}$ since both receivers want packets l_1 and l_2. For unicast setting, we assume that R_1 wants only l_1 while R_2 wants only l_2, thus $r_{12} = 0$ and $r_{21} = 0$. Given this definition, we can express the reward matrix for both unicast and broadcast settings when sending l_1 as shown in Figure 6. For example, the immediate reward when transitioning from state 1 to state 4 under action "sending l_1" is $r_{11} + r_{21}$. The reason is that the reward in state 1 is zero, and by transition to the state 4, both receivers receive l_1 and l_2. Thus, the immediate reward should be equal to the sum of the individual rewards. In the broadcast and unicast settings, this sum equals to $2r_{11}$ and r_{11}, respectively. Similarly, we can write down the reward matrix for sending $l_1 \oplus l_2$ as shown in Figure 7.

IV. Simulation Results

In this section, we present some simple simulation results to demonstrate the advantages our MDP based network coding scheduling algorithm. We assume that the AP has two layered video sequences: Akiyo and Foreman which are to be sent to two receivers R_1 and R_2. Each layered video sequence is assumed to have four layers: the base layer, the enhancement layers 1, 2, and 3. We further assume that there is a dedicated bandwidth for the base layers, and that the base layers are never lost. This assumption is made to ensure that the quality of any video is reasonable at a receiver, and it is not critical. thus only three (enhancement) layers of each video are considered in our simulations. We now equate the layers

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	States	00 00	01 00	10 00	11 00	00 01	01 01	10 01	11 01	00 10	01 10	10 10	11 10	00 11	01 11	10 11	11 11
1	00 00	0	r_{21}	r_{11}	r_{11}+r_{21}	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	0	0	r_{11}	0	0	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	0	r_{21}	0	0	0	0	0	0	0	0	0	0	0	0
4	11 00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	00 01	0	0	0	0	0	r_{21}	r_{11}	r_{21}+r_{11}	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	0	0	r_{11}	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	0	r_{21}	0	0	0	0	0	0	0	0
8	11 01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	00 10	0	0	0	0	0	0	0	0	0	r_{21}	r_{11}	r_{11}+r_{21}	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	0	0	r_{11}	0	0	0	0
11	10 10	0	0	0	0	0	0	0	0	0	0	0	r_{21}	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}	r_{11}	r_{11}+r_{21}
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{11}
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 6. *Reward matrix associated with action "sending packet l_1".*

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	States	00 00	01 00	10 00	11 00	00 01	01 01	10 01	11 01	00 10	01 10	10 10	11 10	00 11	01 11	10 11	11 11
1	00 00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	0	0	0	0	r_{22}	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	0	0	0	0	0	0	0	0	r_{12}	0	0	0	0	0
4	11 00	0	0	0	0	0	0	0	r_{22}	0	0	0	r_{12}	0	0	0	r_{12}+r_{22}
5	00 01	0	0	0	0	0	r_{21}	0	0	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	0	r_{21}	0	0	0	0	0	0	r_{12}	r_{21}+r_{12}
8	11 01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{12}
9	00 10	0	0	0	0	0	0	0	0	0	0	r_{11}	0	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	0	0	r_{11}	0	r_{22}	0	r_{11}+ r_{22}
11	10 10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{22}
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}	r_{11}	r_{21}
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{11}
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 7. *Reward matrix associated with action "sending packet $l_1 \oplus l_2$".*

to packets, i.e., three layers from Akiyo and three layers from Foreman sequences will be denoted as packets l_1^A, l_2^A, and l_3^A and l_1^F, l_2^F, and l_3^F respectively. Associated with each packet (layer) is a reward or a distortion reduction amount in terms of MSE. In particular, we use the distortion reduction $r_1^A = 20.23$, $r_2^A = 13.06$, $r_3^A = 12.19$, $r_1^F = 14.67$, $r_2^F = 10.60$, and $r_2^F = 6.85$ as provided in [20]. Given that we only have N time slots to send these packets, the objective of our MDP based network coding scheduling algorithm is to maximize the total distortion reduction for both video sequences.

We will compare the performances of four algorithms: The greedy algorithm, the MDP non-network coding algorithm, the MDP network coding algorithm, and the network coding only algorithm. In the greedy algorithm, the AP sends packets starting from the packet with the largest distortion reduction to the one with the lowest distortion reduction, i.e., following the order r_1^A, r_1^F, r_2^A, r_3^A, r_2^F, and r_2^F. Each packet is sent until either it is received correctly at the intended receiver(s) or the number of transmissions exceeds N. After N time slots, regardless of whether or not the AP successfully sends all 6 packets, it moves to the next 6 packets (layers) of the next frame. For the MDP non-network coding algorithm, the scheduler is the optimal MDP policy resulted from using our MDP framework with the action space that does not include action "sending XOR packets". In contrast, the action space of the MDP network coding algorithm includes action "sending XOR packets". In the network coding only algorithm, the AP also sends packets in a greedy manner, i.e., starting from the largest distortion reduction to the one with the lowest distortion reduction but the AP will perform network coding instantly if there is an opportunity. Specifically, the AP keeps sending a packet until at least one of the receivers receive it correctly before sending the next packet. If there is a pair of distinct lost packets for two receivers, it immediately XORs them and broadcasts the XOR packet to two receivers. A lost XOR packet at one receiver will be used to XOR with another lost packet from the other receiver if there is a network coding opportunity in subsequent transmissions. Note that, for unicast, not all pairs of lost packets are used for combining (XOR-ing). For example, if packets a and b need to be unicasted to R_1 and R_2, respectively. If a is received correctly at R_1 but incorrectly at R_2, and b is received correctly at R_2 but incorrectly at R_1, then the unicast transmissions of a and b have been finished and sending XOR packet $a \oplus b$ is not necessary. In the algorithm with MDP, the scheduler will not perform network coding in this situation due to zero-reward gain. After the transmission and network coding phase, if there are still some lost packets at one user, the AP will perform a retransmission phase for this user until there is no lost packet left or the total number of transmissions exceeds N.

Figure 8 shows the total distortion reduction as a function of packet loss probability at R_1 in the broadcast setting for four algorithms. The packet loss rate of R_2 is kept constant at 15% and the total transmission opportunities (maximum number of time steps) $N = 10$. The higher distortion reduction amount results in better average video qualities. As seen, the MDP with network coding algorithm performs the best, followed by the network coding only algorithm, then the MDP non-network coding algorithm, and then the greedy algorithm. For a fixed $N = 10$, the increase of the loss rate of R_1 results in a decrease in throughput. With a small throughput, it is becoming critical to schedule the packets optimally to maximize the video qualities at the receivers. Note that in a broadcast setting with two receivers, the maximum distortion reduction is equal

to $2(r_1^A + r_2^A + r_3^A + r_1^F + r_2^F + r_3^F) = 155.2$ since each receiver receives a maximum reward or distortion reduction equal to $r_1^A + r_2^A + r_3^A + r_1^F + r_2^F + r_3^F = 77.6$. When the loss rate at R_1 is 5% and $N = 10$, the AP has many opportunities to successfully transmit all the packets to both receivers, resulting in approximately the maximum distortion reduction as shown in Figure 8.

For the same broadcasting setting, we fix the loss rates $p_1 = 0.1$ and $p_2 = 0.2$, and vary N. Figure 9 shows the rewards as a function of N for four algorithms. As N increases, there are more opportunities to retransmit the lost packets, thus the performances of all four algorithms also increase. Again, the MDP with network coding algorithm performs the best, followed by other algorithms.

Interestingly, the network coding only algorithm, although transmitting packets in a greedy manner combined with network coding, does not perform as well as the MDP with network coding algorithm as seen in Figure 8. In Figure 9, as N is getting smaller ($N \leq 10$), the performance of the network coding only algorithm shows to be degraded significantly, even much lower than those of all other algorithms. This can be explained by the fact that the network coding only algorithm tries to transmit a packet until at least one receivers gets it correctly before transmitting the next packet. By doing so, the scheduler is expecting a network coding opportunity after some next transmissions. While this mechanism may provide some bandwidth gain, there can be a case that there is no network coding opportunities or no time available for transmitting network coding packets before number of transmissions exceeds N. Therefore, all the packets that depend on the lost packets, which have been received correctly, are useless when the lost packet cannot be recovered in time. However, when N is large, there are more time slots available for transmitting network coding packets, thus the performance of the network coding only algorithm can be slightly better than those of the greedy and MDP without network coding algorithms.

We now consider the unicast setting in which R_1 wants to receive Akiyo sequence only while R_2 wants to receive Foreman sequence only. Figure 10 shows the distortion reduction as a function of the loss rates at R_1 and R_2 for $N = 10$. Clearly, as the loss rates increase, the performances of all four algorithms decrease. The MDP network coding algorithm still performs the best while the network coding only algorithm performs the worst. The large performance gap between the network coding only algorithm and others is due to the same reason as explained previously for the broadcast setting. That the scheduler attempts to create pairs of distinct lost packets for network coding and does not consider the dependence among packets results in a degraded performance. Note that in the unicast setting with two receivers, the maximum distortion reduction equals to $r_1^A + r_2^A + r_3^A + r_1^F + r_2^F + r_3^F = 77.6$ since each receiver only cares about its video sequence. This maximum distortion reduction is achieved when the loss rates of both receivers are less than 5% and $N = 10$ which gives

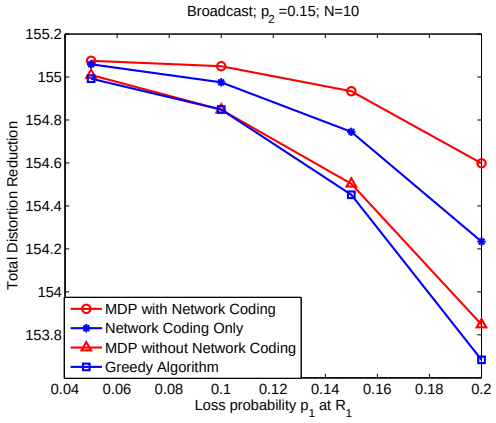

Fig. 8. *Distortion reduction versus loss probabilities for broadcast scenario.*

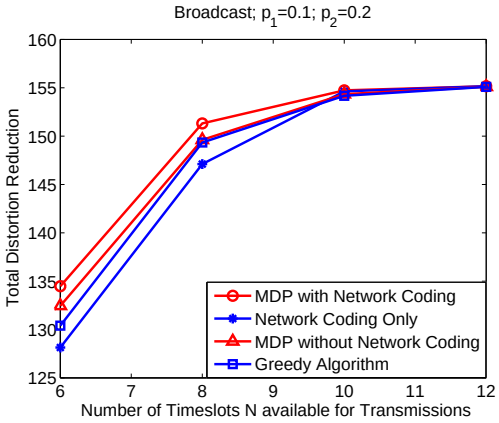

Fig. 9. *Distortion reduction versus transmission opportunities (N) for broadcast scenario.*

enough time for the AP to successfully retransmit all the lost packets.

For the same unicast setting, we now fix the loss rates and vary N. As predicted in Figures 11, larger N provides better performance due to larger available bandwidth. Again, the MDP network coding algorithm leads to largest reward or best video qualities. The performance of the greedy algorithm is very close to that of the MDP non-network coding algorithm as shown in both figures. This is plausible as the MDP always selects the packet with largest distortion reduction to send to its intended receiver, resulting in a similar performance to that of the greedy algorithm.

V. Conclusions

In this paper, we have proposed a network coding based scheduling policy at a WLAN-like Access Point (AP) or at a WiMAX-like broadcast station that optimizes the multimedia transmission in both broadcast and unicast settings. In particular, our contributions include (a) a framework for increasing the bandwidth efficiency of broadcast and unicast sessions in a wireless network based on network coding techniques and (b) an optimized scheduling algorithm based on the Markov Decision Process (MDP) to maximize the quality of multimedia applications. The results demonstrate the advantages of our approach over

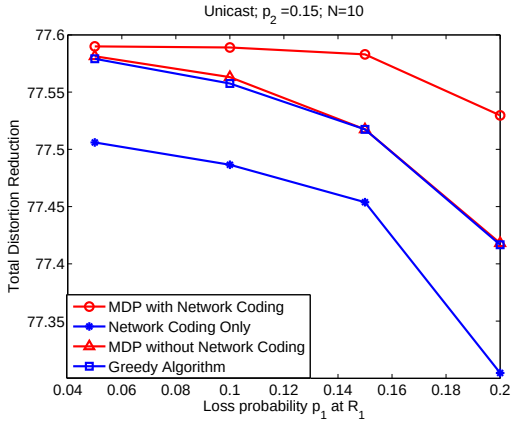

Fig. 10. *Distortion reduction versus loss probabilities for unicast scenario.*

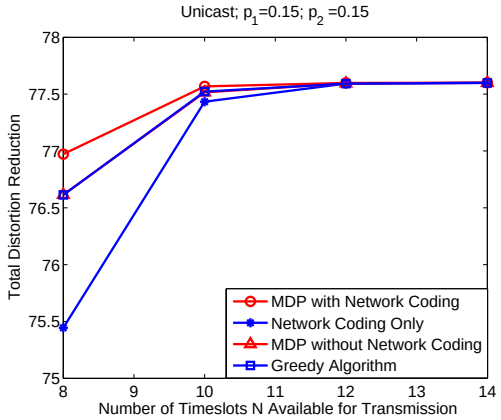

Fig. 11. *Distortion reduction versus transmission opportunities (N) for unicast scenario.*

the traditional techniques.

REFERENCES

[1] R. Ahlswede, N. Cai, R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, pp. 1204–1216, July 2000.

[2] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Transactions on Multimedia*, vol. 8, no. 2, 2006.

[3] P. Chou, A. Mohr, A. Wang, and S. Mehrotra, "Error control for receiver-driven layered multicast of audio and video," *IEEE Transactions on Multimedia*, vol. 3, pp. 108–22, March 2001.

[4] P. Chou and A. Sehgal, "Rate-distortion optimized for receiver-driven streaming over best effort networks," in *Packet Video Workshop*, April 2002.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast application," in *Architectures and Protocols for Computer Communication*, October 2000, pp. 43–56.

[6] C. Fragouli, J. L. Boudec, and J. Widmer, "Network coding: An instant primer," in *Technical Report, TR2005010, EPFL*, 2005.

[7] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Signal Processing: Image communication*, vol. 15, pp. 77–94, 1999.

[8] R. Howard, *Dynamic Programming and Markov Decision Processes*. MIT Press, 1960.

[9] M. Kalman and B. Girod, "Techniques for improved rate-distortion optimized video streaming," *ST Journal of Research-Networked Media*, vol. 2, no. 1, 2004.

[10] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "Xors in the air: Practical wireless network coding," in *SIGCOMM*, 2006.

[11] W. Li, "Overview of fine granularity scalability in mpeg-4 video standard," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 301–317, 2001.

[12] H. Ma and M. E. Zarki, "Broadcast/multicast mpeg-2 video over wireless channels using header redundancy fec strategies," in *Proceedings of The International Society for Optical Engineering (SPIE)*, vol. 3528, November 1998, pp. 69–80.

[13] A. Mohr, E. Riskin, and R. Ladner, "Unequal loss protection: Graceful degradation over packet erasure channels through forward error correction," *IEEE Journal on Selected Areas in Communication*, vol. 18, pp. 819–828, April 2000.

[14] D. Nguyen, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," in *Third Workshop on Network Coding, Theory, and Applications*, January 2007.

[15] T. Nguyen and A. Zakhor, "Multiple sender distributed video streaming," *IEEE Transactions on Multimedia*, vol. 6, no. 2, pp. 315–326, April 2004.

[16] M. L. Puterman, *Markov Decision Processes Discrete Stochastic Dynamic Programming*. John Wiley and Sons, Inc., New York, NY, 1994.

[17] H. Radha, M. van der Schaar, and Y. Chen, "The mpeg-4 fine-grained scalable video coding method for multimedia streaming over ip," *IEEE Trans. on Multimedia*, vol. 3, pp. 53–68, March 2001.

[18] G. D. L. Reyes, A. Reibman, S. Chang, and J. Chuang, "Error-resilient transcoding for video over wireless channels," *IEEE Transactions on Multimedia*, vol. 18, pp. 1063–1074, june 2000.

[19] J. Robinson and Y. Shu, "Zerotree pattern coding of motion picture residues for error resilient transmission of video sequences," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1099–1110, June 2000.

[20] X. Sun, F. Wu, S. Li, W. Gao, and Y. Zhang, "Macroblock-based progressive fine granularity scalable (pfgs) video coding with flexible temporal-snr scalabilities," in *IEEE Internation Conference on Image Processing*, vol. 2, 2001, pp. 1025–1028.

[21] W. Tan and A. Zakhor, "Error control for video multicast using hierarchical fec," in *Proceedings of 6th International Conference on Image Processing*, vol. 1, October 1999, pp. 401–405.

[22] ——, "Real time internet video using error resilient scalable compression and tcp friendly transport protocol," *IEEE Transactions on Multimedia*, vol. 1, pp. 172–186, june 1999.

[23] J. Widmer, C. Fragouli, and J.-Y. L. Boudec, "Low-complexity energy-efficient broadcasting in wireless ad-hoc networks using network coding," in *Proc. Workshop on Network Coding, Theory, and Applications*, Apr. 2005.

[24] Y. Wu, P. A. Chou, and S. Kung, "Information exchange in wireless networks with network coding and physical-layer broadcast," in *Technical Report MSR-TR-2004-78, Microsoft Research*, Aug. 2004.

[25] ——, "Minimum energy multicast in mobile ad hoc networks using network coding," in *IEEE Information Theory Workshop*, Oct. 2004.