Adaptive Scheduling for Multicasting Hard Deadline Constrained Prioritized Data via Network Coding

Tuan T. Tran[1], Hongxiang Li[1], Weiyao Lin[2], Lingjia Liu[3], and Samee U. Khan[4]
[1]Department of Electrical and Computer Engineering,
J.B. Speed School of Engineering, University of Louisville, KY 40292.
[2]Institute of Image Communication and Information Processing,
Shanghai Jiaotong University, China.
[3]Department of Electrical Engineering and Computer Science,
The University of Kansas, Lawrence, KS 66045.
[4]Department of Electrical and Computer Engineering
North Dakota State University, Fargo, ND 58102.
Email: tttran14@louisville.edu

***Abstract*—Network coding offers a promising platform for multicast transmission by approaching its min-cut capacity. However, pushing the network throughput toward this upper bound comes with a sacrifice in delivery delay due to the decoding procedure that requires performing batch of coded packets. Further, in some transmission scenarios where the receivers experience deep fading or unable to collect a full set of the transmitted data, no useful information is recovered. The effect is more severe in the networks where the transmitted information has priority structure with hard deadline constraint due to the limited delivery time and data interdependencies. In this paper, we consider single-hop wireless networks where the transmitter wishes to multicast hard deadline constrained prioritized data to many receivers over lossy channels. We first study the network performance of a variety of transmission techniques, depending on how the transmitter schedules transmission in each time slot. We then propose an adaptive encoding and scheduling technique to maximize the network throughput. To find the optimal transmission scheduling at the presence of the network dynamics, we cast the problem in the framework of Markov Decision Processes (MDP) and use backward induction method to find an optimal solution. We further propose simulation-based algorithm and greedy scheduling technique that obtain high performance with much lower time complexity. Both analytical and simulation results have been provided to corroborate the effectiveness of the proposed techniques.**

***Index Terms*—Multicast, hard deadline constraint, prioritized transmission, adaptive scheduling, network coding.**

I. Introduction

Multimedia networking applications over wireless networks have gained much popularity recently. Unlike other types of traffic, multimedia traffic such as video requires a higher level of quality of service (QoS), e.g., minimum bandwidth and maximum delay tolerance, to meet user satisfactions. However, the current wireless networks such as WiFi were not designed for efficient provisioning of network resources in order to guarantee some specified QoS. To mitigate this lack of infrastructure support, scalable compression techniques are used to allow the sender to dynamically adjust the media bitrate to the current available bandwidth in real time. Notably, scalable video coding (SVC) [14] is a class of techniques that allows a sender to adapt its video bit rate by partitioning a video bit stream into a base layer and several enhancement layers. The receiver is then able to view the video with higher quality when more layers are received. The base layer is the most important layer and must be present in order to have a reasonable video quality. The enhancement layers are organized in a hierarchical fashion such that the first enhancement layer must be present for the second enhancement layer to be useful, and the second enhancement layer must be present for the third enhancement layer to be useful, and so on. Naturally, this scenario leads to the general problem of prioritized transmissions where the goal is to transmit the most useful data under some resource constraints.

To deliver prioritized data to multiple users in wireless networks, multicasting is a well-suited emerging technology. The underlying principle of wireless multicast is to exploit the inherent broadcast nature of a wireless network. That is, a transmitted packet can be intercepted by multiple receivers in the transmission range of the sender [6], [16]. As a result, the required bandwidth and power consumption are substantially reduced compared to the unicast protocols. However, designing an efficient multicast protocol for prioritized transmissions is quite challenging due to the lossy nature of wireless channels, and the heterogeneity in the amount of information correctly received across different receivers. In addition, it needs to tackle the "*bottleneck*" effect of the shared transmission medium among many receivers, i.e., the overall performance is limited by the receiver that has the worst channel condition. With that said, ensuring the QoS for the bottleneck user on par with the others clearly makes the multicast transmission more challenging.

Fortunately, network coding (NC), a recent routing protocol proposed by Ahlswede *et al.* [1], offers a promising platform for multicast transmissions. Indeed, it has been proved that one can approach the multicast capacity by using random linear network coding technique [10], by which the intermediate nodes form output data by linearly combining the input data. A receiver is able to decode the original transmitted data when it receives a full set of independent coded packets. This method is simple, efficient, and decentralized in the sense that the coded packets are independently generated at intermediate nodes.

However, random network coding may come with a sacrifice in delivery delay due to the decoding procedure that requires performing batch of coded packets. In addition, receivers need to collect a full set of the coded packets in order to recover the transmitted data, e.g., by Gaussian elimination. Specially, when only partial set of the coded packets is received, no useful information is recovered. As a result, it substantially decreases the QoS of receivers that are unable to collect enough data during the transmission period. The effect is more severe in transmission scenarios where the transmitted information has priority structure with hard deadline constraint due to the limited delivery time and data interdependencies. Without any careful encoding and scheduling design, the effect can be detrimental.

Therefore, in this paper, we investigate transmission scheduling strategies in wireless networks where the transmitter wishes to multicast hard deadline constrained prioritized data to many receivers over lossy channels. This framework

can be used to model many practical networks such as multimedia streaming in the last-mile networks, or remotely reprogramming the sensors in unreachable fields. In our model, we allow the transmitter to use random network coding (RNC) [10] to encode the incoming data packets before sending them out. The task of the transmitter is to schedule packet transmissions, i.e., selecting which packets in which time slots, in order to maximize the network throughput. Particularly, we propose an adaptive encoding and scheduling technique based on the framework of Markov Decision Processes (MDP) to exploit the feedback information from the receivers. Further, to reduce the time complexity of the standard algorithm that finds the optimal solution to the MDP, we propose simulation-based algorithm and greedy scheduling technique.

Our main contributions in this paper are as follows.

- We study in detail a variety of transmission techniques, depending on different transmission scheduling strategies.
- We then design an adaptive encoding and scheduling technique that maximizes the network throughput based on the framework of MDP.
- We further propose simulation-based algorithm and greedy scheduling technique that approximate an optimal network performance with much lower time complexity.
- We corroborate the effectiveness of the proposed techniques through both theoretical analysis and simulations.

The organization of the paper is as follows. We provide in Section II some preliminaries and related work. In Section III, we describe the system model and problem formulation. In Section IV, we study in detail a variety of transmission techniques and analyze the corresponding network performance. Simulation results and discussions are provided in Section V. Finally, we conclude the paper in Section VI.

II. Preliminaries and Related Work

Formally, the prioritized data refers to the notion that, given L prioritized packets in the decreasing order of importance, $a_1 \geq a_2 \geq \cdots \geq a_L$, to be delivered to a receiver, then the packet a_i is useful to the receiver only if it has received all its interdependent packets a_j with $j < i$ successfully. Those interdependencies are graphically represented in Fig. 1(a), in which a_1 is the most important data packet and it needs to be decoded first in order to the other packets to be decoded. Similarly, Fig. 1(b) represents the dependencies of video frames of a group of pictures (GOP) often used in MPEG standard [7]. As shown, frame B_1 can be decoded only if its dependent frames I and P_1 also have been decoded. Also, frame P_1 can be decoded only if frame I has been decoded. Eventually, as represented in the graph, data packet containing frame I is the most important data packet that needs to be decoded first in order to decode the GOP. That said, when dealing with the prioritized data, receiving more data packets may not necessarily result in higher QoS. This is because received data packets which are missing their interdependencies cannot be decoded. Thus, to compare the performance between two prioritized transmission techniques, we use the effective throughput, i.e., the data that contributes to the improvement of the QoS at receivers.

In the literature, multicasting hard deadline constrained data has been studied in [2], [3]. In these works, the authors considered one-hop wireless networks with random incoming data and proposed scheduling strategies to minimize the system energy. These works focused on deriving energy bounds from theoretical view point and did not consider prioritized data. On the other hand, in the context of streaming of prioritized data over lossy wired network, Chou *et al.* [5] proposed a heuristic algorithm to minimize the rate-distortion of the stream. In another work, Tran *et al.* [15] proposed a class of approximate

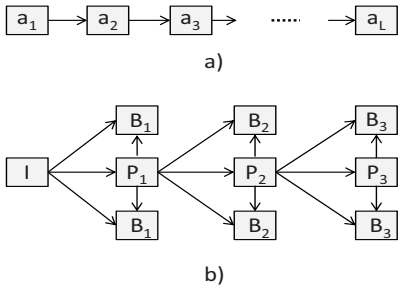

Fig. 1. Examples of data interdependence graphs. (a) Sequential dependencies of prioritized data. (b) Dependencies of video frames of a group of pictures (GOP) using MPEG standard.

algorithms based on the framework of Markov Chain Monte Carlo (MCMC) to maximize the sum throughput of a wireless broadcast network. However, in this work the authors assumed that the transmitter is an oracle which knows the channel conditions of all time slots in advance.

Along in a different avenue, since the pioneering work of Ahlwede [1], NC has received much attention. In particular, in the context of noiseless multiple-source multicast transmission over wired networks, Wu *et al.* [17] proposed a practical communication scheme for a case with two sources. The proposed scheme achieved low complexity and robustness to the network dynamics, but it resulted in a suboptimal performance. In another work, Goel *et al.* [9] derived a lower bound of energy consumed in wireless multicast networks using NC. Further, some adaptive NC schemes have been proposed in wireless networks to increase the network throughput and bandwidth efficiency [11], [12], [13]. However, all these works did not consider prioritized data or focused only on heuristic algorithms to minimize the decoding delay of special scenarios with limited number of receivers. Different from these works, we investigate a general framework by employing adaptive network coding for multicasting prioritized data with hard deadline constraint over error-prone channels.

III. System Model

Consider a single hop wireless network, such as WiFi or WiMAX networks, where the transmitter wishes to multicast hard deadline constrained prioritized data to many receivers. We assume that system operates on a single frequency and the communication channels between the transmitter and receivers are lossy, i.e., each packet transmitted to receiver R_k is subject to an erasure probability ϵ_k. The erasure probabilities are assumed to be independent and identically distributed ($i.i.d.$) across receivers and time. We assume that the system is time-slotted and each slot length corresponds to one packet transmission. At the transmitter, random incoming data packets are accumulated in its buffer before encoding and sending them out. Optimal accumulation strategy is not the focus of this paper. Instead, in this paper we are particularly interested in designing adaptive encoding and scheduling schemes to maximize the network throughput. We assume that the transmitter has L prioritized data packets in the order of importance: $a_1 \geq a_2 \geq \cdots \geq a_L$, where their sequential interdependencies are represented in Fig. 1(a). The transmitter wants to transmit these data packets to K receivers over erasure channels within a deadline $T \geq L$ time slots. Further, we assume that the receivers will inform the transmitter whether a packet has been received successfully or not via one-bit feedback messages. For the sake of clarity, we assume that the feedback messages are instantaneous and reliable. However, the main principle of the framework still holds for the case of unreliable feedback, and one can develop a more accurate model, albeit complicate

analysis. At the receivers, a packet a_j is decoded only if it is received before the deadline, and all of its interdependent packets a_i, $\forall i < j$, also have been received successfully. We assume that the transmitter is able to implement RNC over a large finite field $\mathbb{F}_q$.

Different packet encoding and scheduling schemes may result in different network throughputs. In order to achieve the maximum network throughput, the transmitter needs to carefully encode and schedule data packet in every time slot. Therefore, the main focus of this paper is to design adaptive encoding and scheduling techniques to maximize the network throughput of multicasting hard deadline constrained prioritized data over erasure channels. Before delving into details of the problem, we first provide a metric that will be used throughout the paper as the performance measure to compare different techniques.

Definition 3.1: Assume the transmitter has L prioritized data packets in the decreasing order of importance $a_1 \geq a_2 \geq \cdots \geq a_L$ associated with a deadline of T time slots. Receiver R_k achieves a throughput $\eta_k = m \leq L$ packets if it can recover m consecutive original data packets $\{a_1, a_2, \ldots, a_m\}$ by the deadline T. The average network throughput per time unit across K receivers is defined as

$$\eta \triangleq \lim_{\zeta \to \infty} \frac{1}{\zeta} \sum_{\zeta} \frac{\sum_{k=1}^{K} \eta_k}{KT}, \tag{1}$$

where ζ is the operating time of the network. Using this definition, a transmission technique A is better than technique B if $\eta_A > \eta_B$.

IV. Transmission Techniques: Performance Analysis

In this section, we will investigate in detail a variety of transmission techniques for multicasting hard deadline constrained prioritized data to multiple receivers, depending on how the transmitter schedules transmission in each time slot. In particular, we consider five transmission strategies: Automatic repeat-request, round-robin scheduling, random network coding, and the new adaptive random network coding and greedy scheduling techniques. For all techniques, we assume the transmitter has L prioritized data packets which need to be delivered to K receivers within a hard deadline of T time slots.

A. Automatic Repeat ReQuest (ARQ)

This is a basic approach to deliver data to the receivers over error-prone channels. In this transmission protocol, after sending out a data packet, the transmitter waits for feedback messages, i.e., acknowledgements (ACKs) or negative acknowledgements (NAKs), from the receivers to decide which packet will be transmitted in the next time slot. If there exists at least a packet loss at the receivers, the transmitter resends the lost packet until all receivers receive it correctly. We note that it does not necessarily require all receivers successfully receive the transmitted data packet in the same time slot as the correctly received packet can be stored in the receivers' buffer. Intuitively, the network throughput of the data multicasting using ARQ technique is determined by the "bottleneck" receiver, i.e, the receiver has the worst channel condition. With this observation, we will use the tools of the order statistics to characterize the network performance.

Let X_k be the random variable denoting the number of transmissions that the transmitter needs to attempt to deliver a packet to receiver R_k. Further, define Y as the number of transmissions required to deliver a packet to all receivers, we then can write $Y = \max_{k\in\{1,\ldots,K\}}\{X_k\}$. The probability that the transmitter needs at most n transmissions to deliver a packet to all receivers is given by

$$\begin{aligned}\mathbb{P}(Y \leq n) &= \mathbb{P}\left(\max_{k\in\{1,\ldots,K\}}\{X_k\} \leq n\right) \\ &\overset{(a)}{=} \prod_{k=1}^{K}\left(1-\epsilon_k^n\right),\end{aligned} \tag{2}$$

where (a) follows the result of the geometric distribution with success probability $1-\epsilon_k$. Similarly, we have the probability that the transmitter needs less than n transmissions to deliver a packet to all receivers is given by

$$\mathbb{P}(Y < n) = \prod_{k=1}^{K}\left(1-\epsilon_k^{n-1}\right). \tag{3}$$

From (2) and (3), the expected number of transmissions needed to deliver a packet to all receivers can be written as

$$\begin{aligned}\mathbb{E}[Y] &= \sum_{n=1}^{\infty} n\mathbb{P}(Y=n) \\ &= \sum_{i_1,i_2,\ldots,i_K} \frac{(-1)^{i_1+i_2+\cdots+i_K-1}}{1-\epsilon_1^{i_1}\epsilon_2^{i_2}\ldots\epsilon_K^{i_K}},\end{aligned} \tag{4}$$

where $i_1, i_2, \ldots, i_K \in \{0,1\}, \exists i_j \neq 0$. Therefore, after a long operation, the average network throughput per time slot across K receivers of the ARQ technique is written as

$$\eta = \frac{1}{\mathbb{E}[Y]}. \tag{5}$$

B. Round-robin Scheduling (RRS)

This is a basic time-sharing scheme where the transmitter sends the data packets in a circular order. Particularly, using this technique packet a_i is transmitted in time slots ts_j if $T \equiv j \pmod{L}$. Consider Example 1, for instance, we have $L = 3$ and $T = 4$; thus packet a_1 ($i = 1$) will be transmitted in time slots ts_j where $j = \{1, 4\}$ because $T \equiv j \pmod{L}$.

Let T_i denote the number of time slots used for transmitting data packet a_i. We have that

$$T_i = \begin{cases} \lfloor \frac{T}{L} \rfloor + 1 & \text{if } i \leq T - L\lfloor \frac{T}{L} \rfloor; \\ \lfloor \frac{T}{L} \rfloor & \text{otherwise,} \end{cases} \tag{6}$$

where $\lfloor x \rfloor$ denotes the largest integer not greater than x. The probability that packet a_i is received successfully at receiver R_k is given by

$$\mathbb{P}_k(a_i) = \sum_{j=1}^{T_i} \binom{T_i}{j} \epsilon_k^{T_i-j}(1-\epsilon_k)^j. \tag{7}$$

Let X_i be the random variable representing the event packet a_i can be decoded at receiver R_k after T time slots (this implies that all of its interdependent packets are also received successfully.) Then, we have the probability that packet a_i is decoded at receiver R_k is

$$\begin{aligned}\mathbb{P}_k(X_i) &= \prod_{j=1}^{i} \mathbb{P}_k(a_j) \\ &= \prod_{j=1}^{i} \sum_{m=1}^{T_j} \binom{T_j}{m} \epsilon_k^{T_j-m}(1-\epsilon_k)^m.\end{aligned} \tag{8}$$

Next, we obtain the expected throughput achieved at receiver R_k as

$$\eta_k = \sum_{l=1}^{L} l\mathbb{P}_k(X_l). \quad (9)$$

From (1), (8), (9), and considering the independent received data at each receiver, we obtain the average network throughput as

$$\eta = \frac{1}{KT}\sum_{k=1}^{K}\sum_{l=1}^{L} l \prod_{j=1}^{l}\sum_{m=1}^{T_j}\binom{T_j}{m}\epsilon_k^{T_j-m}(1-\epsilon_k)^m, \quad (10)$$

where T_j is determined from (6).

C. Random Network Coding (RNC)

In this transmission technique, the transmitter uses RNC to generate coded packets and sends them out at every time slot. Particularly, a coded packet is generated as $c_i = \sum_{j=1}^{L}\alpha_{ij}a_j$, where α_{ij} are withdrawn randomly from the finite field $\mathbb{F}_q$. Let X be the random variable denoting the number of coded packets received successfully at receiver R_k. The probability that receiver R_k can recover all L original data packets is given by[1]

$$\mathbb{P}_k(X \geq L) = \sum_{l=L}^{T}\binom{T}{l}(1-\epsilon_k)^l\epsilon_k^{T-l}. \quad (11)$$

We should note that once a receiver collects at least L coded packets, it is able to recover all the original data packets by solving the system of linear equations formed by the received coded packets. Thus, the expected throughput achieved at receiver R_k is

$$\begin{aligned}\eta_k &= L\mathbb{P}_k(X \geq L)\\ &= L\sum_{l=L}^{T}\binom{T}{l}(1-\epsilon_k)^l\epsilon_k^{T-l}. \end{aligned} \quad (12)$$

Similarly as in ARQ technique, from (1), (11), (12), and considering the independent data received at the receivers, the average network throughput of RNC technique is represented as

$$\eta = \frac{L}{KT}\sum_{k=1}^{K}\sum_{l=L}^{T}\binom{T}{l}(1-\epsilon_k)^l\epsilon_k^{T-l}. \quad (13)$$

D. Adaptive Random Network Coding (ARNC)

In this transmission technique, the transmitter exploits the feedback information from the receivers to adaptively encode and schedule coded packet at every time slot to maximize the average network throughput. We assume that at the beginning of each time slot the transmitter receives one-bit feedback from the receivers to indicate whether the previous transmitted packet has been received successfully. Based on that, the transmitter updates the network state, i.e., which receiver has which packets, and decides which packet will be sent out in the next time slot. Fortunately, in this scenario the network dynamics can be modeled as a Markov Decision Process (MDP) in which the transmitter acts as a decision maker to decide which action to take from an action set at every time slot. We specify the network dynamics by a six-tuple $(\mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{r}, T, \gamma)$, where we use boldface letters to refer to vectors or matrices.

[1]Assume the finite field $\mathbb{F}_q$ is large so that all coded packets are independent.

1) State space $\mathbf{S}$: A network state s is defined by a matrix

$$s = \begin{bmatrix} n_{11} & n_{12} & \dots & n_{1L} \\ n_{21} & n_{22} & \dots & n_{2L} \\ \dots & \dots & & \\ n_{K1} & n_{K2} & \dots & n_{KL} \end{bmatrix}, \quad (14)$$

where an element n_{ij} represents the number of coded packet, which is generated from a data batch of j original packets $\{a_1, a_2, \dots, a_j\}$ and received by receiver R_i.
2) The action set $\mathbf{A}$ consists of actions sending coded packets that are generated from different data generations (batches). In particular, there have L generations where generation G_i consists of i consecutive original packets $\{a_1, a_2, \dots, a_i\}$. We then denote a coded packet which is generated from data generation G_i as c^i.
3) The transition distribution $\mathbf{P}(s_{t+1}|s_t, a_t)$ is computed based on the network current state, s_t, action taken, a_t, and packet erasure probability of the channel to each receiver. For example, in the case of two receivers and $L = 2$. If the network current state is $s_t = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$, and the transmitter takes action $a_t =$ "sending a coded packet $c^1 \in G_1$", the probability that the network transits to state $s_{t+1} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ is $P(s_{t+1}|s_t, a_t) = (1-\epsilon_1)\epsilon_2$. This scenario occurs when R_1 and R_2 correspondingly succeeds and fails to receive the transmitted packet.
4) The immediate reward matrix $\mathbf{r}(s, a)$ is computed based on the future-dependent reward function $r(s_{t+1}|s_t, a_t)$ as

$$r(s_t, a_t) = \sum_{s_{t+1}\in\mathbf{S}} r(s_{t+1}|s_t, a_t)P(s_{t+1}|s_t, a_t). \quad (15)$$

If we define the *"terminal reward"* $r(s_T) \triangleq \eta_{s_T}$, i.e., the average network throughput when the network is in state s_T, we then eventually can compute the immediate rewards of all intermediate states. This reward function is designed to reflect whether the receivers are able to decode the transmitted data. Its principle is similar to a delay reward assignment, i.e., the intermediate states, in which the receiver does not have enough data to recover the transmitted data, should have a zero immediate reward.
5) T is the number of stages or time slots associated with the current data batch.
6) $\gamma \in [0, 1)$ is discount factor. When γ is close to zero, the transmitter tends to consider only immediate reward, and γ is close to one, the transmitter prefers future reward with higher weight. The value of γ is adjusted to balance between exploration and exploitation.

A transmission policy is defined as a strategy based on which the transmitter chooses an action corresponding to each state and time slot. Let a non-negative real value function $V_\pi : \mathbf{S} \to \mathbb{R}$ represent the expected reward obtained by following policy π at each state in $\mathbf{S}$. Assume that at the beginning the system is in state s, the expected total reward using any such policy π is defined as

$$V_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{T-1}\gamma^t r(s_t, \pi(s_t)) + \gamma^T r(s_T)|s_0 = s\right], \quad (16)$$

where $\mathbb{E}[.]$ is the expected function. The cumulative reward from time step t to T is recursively computed using backward induction as follows

$$V_\pi(s_t) = r(s_t, \pi(s_t)) + \mathbb{E}\left[\gamma V_\pi(s_{t+1})\right] = r\left(s_t, \pi(s_t)\right) + \gamma \sum_{s_{t+1} \in \mathbf{S}} P(s_{t+1}|s_t, \pi(s_t)) V_\pi(s_{t+1}), \quad (17)$$

where $t = \{T-1, \ldots, 0\}$. We are seeking for an optimal policy π^* in T steps that maximizes the expected cumulative reward. We have that

$$\pi^* = \arg\max_{\pi} \{V_\pi(s_t)\} \quad (18)$$

To find an optimal policy, we can use the backward induction algorithm (BIA). The main principle of this algorithm is that it initializes immediate rewards for all terminal states then computes the expected rewards of the intermediate states backwardly. Specifically, in the time-horizon the algorithm runs from $T-1$ to zero, and in each iteration it computes the immediate reward for each pair of state and action based on the immediate rewards of the future states. The time complexity of the BIA is $\mathcal{O}(T|\mathbf{S}|^2|\mathbf{A}|)$ and it is subject to the curse of dimensionality when dealing with a large space problem. To tackle this problem, we use the simulation-based method [4] to approximate the optimal policy. In particular, our algorithm combines backward induction with simulation-based methods to reduce the time complexity to $\mathcal{O}(\mathbf{T}\Delta|\mathbf{S}||\mathbf{A}|)$ with $\Delta \ll \mathbf{S}$. The pseudocode of the simulation-based backward induction algorithm (SBIA) is summarized in Algorithm 1.

Algorithm 1 : Simulation-based Backward Induction Algorithm (SBIA)

Input: $\mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{r}, T, \gamma, \Delta$.
Output: $\pi^* = \{d(s_1), d(s_2), \ldots, d(s_T)\}$

```
Initialize: t = T, set V*(s_T) := 0 and r(s_T) := η_{s_T} for ∀s_T ∈ S
for t := T − 1 to 0 do
    for each state s_t ∈ S do
        try all actions a_t ∈ A_{s_t} for Δ iterations, and compute
        V̂*(s_t, a_t) = (1/Δ) Σ_Δ [r(s_{t+1}|s_t, a_t) P(s_{t+1}|s_t, a_t)
        +γ V*(s_{t+1})]
        V*(s_t) = max_{a_t ∈ A} { V̂*(s_t, a_t) }
        d(s_t) = arg max_{a_t ∈ A} { V̂*(s_t, a_t) }
    end for
end for
```

Remark 1: At the beginning, the network state is initialized as $s_0 = [\mathbf{0}]$, i.e., matrix with all elements are zero.

Remark 2: So far, we have solved the problem with reliable feedback messages. In practice, however, feedback messages are subject to losses and errors; thus, the transmitter observes only partial of the network states. Fortunately, this case can be formulated as a Hidden Markov Model (HMM) and solved by the same SBIA algorithm with

$$\widehat{V}^*(o_t, a_t) = \frac{1}{\Delta} \sum_{\Delta} \left[r(o_{t+1}|o_t, a_t) P(o_{t+1}|o_t, a_t) + \gamma V^*(o_{t+1})\right], \quad (19)$$

where o_t is the observed state at time step t.

E. Greedy Scheduling Technique (GST)

Intuitively, the GST is of interest because it would have much less time complexity compared to the SBIA algorithm. In this technique, the transmitter seeks for an action that maximizes the network throughput considering only one time step ahead. In particular, based on the feedback information from the receivers, the transmitter updates the network state and selects a coded packet to transmit in the next time slot to maximize the one time step future-reward. The GST technique may not result in a global optimal solution after T steps, but it may yield locally optimal solution that approximates the global optimal solution. To get a concrete sense, consider the example in Section IV-D with the current state of the network $s_t = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. Using GST technique, an optimal action in the next time slot is to send packet $c^1 \in G_1$. This is because no packet has been received, and c^1 is the most important packet that all receivers need to have to be able to decode other packets. Specifically, the optimal action a_t^* taken at time step t to maximize one time step future-reward is expressed as

$$a_t^* = \arg\max_{a_t \in \mathbf{A}} P(s_{t+1}|s_t, a_t) r(s_{t+1}). \quad (20)$$

We can see that the time complexity of the GST algorithm is $\mathcal{O}(T|A|)$. It is much less than that of the BIA and SBIA algorithms, and more importantly, it does not depend on the size of the state space and all actions are computed on the fly.

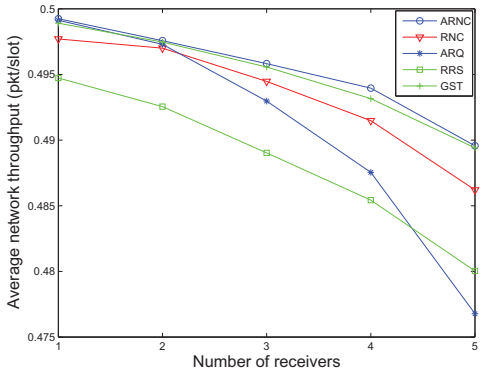

Fig. 2. Network throughput vs. number of receivers.

V. Simulation Results and Discussions

In this section, we illustrate the performance gain of the proposed techniques via simulations. We start with the basic setup.

A. Basic Setup

In our simulation, we assume that each data batch consists of a multimedia frame, e.g., Akiyo [8], prioritized into four interdependent layers in the decreasing order of importance $a_1 \geq a_2 \geq a_3 \geq a_4$, where each layer is assumed can be encapsulated into one packet. To reduce the size of the state space, we categorize the prioritized data into two generations $G_2 = \{a_1, a_2\}$ and $G_4 = \{a_1, a_2, a_3, a_4\}$. Further, we assume that the erasure channels between the transmitter and receivers are independent. The average network throughput is determined as the mean of the average network throughputs across all receivers defined in Eq. (1) over $10,000$ trials.

B. Simulation Results

First, we investigate the impact of network size on the network throughput of different techniques in Fig. 2. In this experiment, we fix the deadline $T = 8$ time slots and vary the number of receivers. To simulate the heterogeneity of channels, we assume that the packet erasure probabilities of the transmission channels are different and set as $10\%, 12\%, 15\%, 17\%$, and

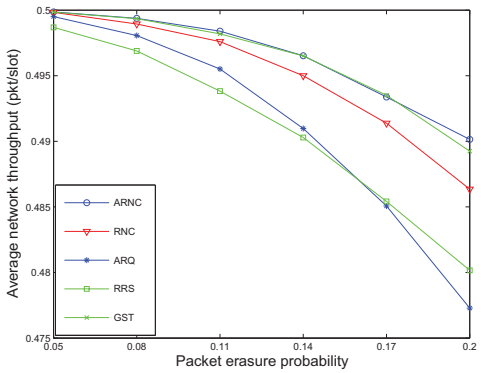

Fig. 3. Network throughput vs. packet erasure probability.

20% (corresponding to $\epsilon_1, \ldots, \epsilon_5$). As expected, the average network throughput of each technique decreases with the increase of number of receivers. This is because of the higher erasure probabilities of the new added receivers and the heterogeneity in the amount of information received across the receivers. Further, we observe that with the higher number of receivers, i.e., $K = 5$, ARQ technique suffers the "bottleneck" effect in which retransmissions duplicate received data at receivers with good channel conditions; consequentially, the network throughput decreases significantly. It is clear that ARNC technique achieves the best performance by employing an adaptive encoding and scheduling transmissions over T time slots, and its performance gain increases substantially in the cases of larger number of receivers. This is because each transmitted packet using ARNC technique brings innovative information to many receivers, leveraging the bottleneck effect. On the other hand, GST technique using one-time-step ahead scheduling significantly reduces the time-complexity while still obtaining a performance very close to that of ARNC.

Next, Fig. 3 depicts the average network throughput of different techniques with respect to the packet erasure probabilities. In this experiment, we fix the number of receivers $K = 5$ and vary the erasure probabilities of the transmission channels. Specifically, we assume that all channels have the same erasure probability ϵ, and we vary ϵ from 5% to 20% with step size of 3%. As seen in Fig. 3, when the erasure probability ϵ is small, all the techniques obtain high network throughput and the performance gains among the techniques are small. Our intuition is that when the erasure probabilities are small, each transmitted packet is received successfully at the receivers with high probability. Consequently, most of the transmitted data can be decoded at the receivers. On the other hand, in the high-erasure regime. i.e., $\epsilon = 20\%$, the network throughput decreases substantially. This is because there are more packet losses due higher erasure probabilities. Further, we observe that the performance gains among the techniques enlarge significantly. In particular, the ARQ technique, that the transmitter may need to retransmit a lost packet many times would duplicate received data at receivers, result in the worst performance. On the other hand, as expected, ARNC technique obtains the best performance while GST approximates that of ARNC with much lower time complexity.

VI. Conclusion

In this paper, we considered transmission scenarios in which the transmitter multicasts hard deadline constrained prioritized data to many receivers over lossy channels. We first investigated the network throughput of a variety of transmission techniques, depending on how the transmitter schedules transmission in each time slot. We further designed an adaptive encoding and scheduling technique based on the framework of MDP to maximize the network throughput. The backward induction algorithm (BIA) was used to find an optimal transmission strategy. To reduce the time complexity of the standard BIA algorithm, we proposed sample-based backward induction algorithm and greedy scheduling technique that have much lower time complexity, but still achieving high performance. Both analytical and simulation results have been provided to support the effectiveness of the proposed techniques.

Acknowledgment

This work is partially supported by the US National Science Foundation (Grant No. 1032567), Kentucky NASA EPSCoR (Grant No. 120469), National Natural Science Foundation of China (Grant No. 61001146), and the Open Project Program of the Chinese National Laboratory of Pattern Recognition (NLPR).

References

[1] R. Ahlswede, N. Cai, R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46:1204–1216, July 2000.

[2] M.M. Butt, K. Kansanen, and R.R. Muller. Hard deadline constrained multiuser scheduling for random arrivals. In *Wireless Communications and Networking Conference (WCNC), 2011 IEEE*, pages 1540 – 1545, 2011.

[3] M.M. Butt, K. Kansanen, and R.R. Muller. Individual packet deadline constrained opportunistic scheduling for a multiuser system. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pages 1–5, 2011.

[4] Hyeong Soo Chang, Michael C. Fu, Jiaqiao Hu, and Steven I. Marcus. *Simulation-based Algorithms for Markov Decision Processes (Communications and Control Engineering)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[5] Philip A. Chou and Zhourong Miao. Rate-distortion optimized streaming of packetized media. *IEEE Transactions on Multimedia*, pages 390–404, 2005.

[6] Carlos de Morais Cordeiro, Hrishikesh Gossain, and Dharma Agrawal. Multicast over wireless mobile ad hoc networks: Present and future directions. *IEEE Network*, 17:52–59, 2003.

[7] Chad Fogg, Didier J. LeGall, Joan L. Mitchell, and William B. Pennebaker. *MPEG video compression standard*. Digital multimedia standards series. Chapman & Hall, 1997.

[8] V. Goebel and T. Plagemann. *Interactive distributed multimedia systems and telecommunication services: The 5th International Workshop (IDMS)*. Springer, 1998.

[9] Ashish Goel and Sanjeev Khanna. On the network coding advantage for wireless multicast in euclidean space. In *Information Processing in Sensor Networks (IPSN)*, pages 64 – 69, 2008.

[10] T. Ho, M. Medard, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006.

[11] Jin Jin and Baochun Li. Adaptive random network coding in WiMAX. In *IEEE International Conference on Communications (ICC)*, pages 2576 – 2580, 2008.

[12] Mahmood K., Kunz T., and Matrawy A. Adaptive random linear network coding with controlled forwarding for wireless broadcast. In *Wireless Days (WD),IFIP*, pages 1 – 5, 2010.

[13] Sadeghi P., Traskov D., and Koetter R. Adaptive network coding for broadcast channels. In *Network Coding, Theory, and Applications (NetCod)*, pages 80 – 85, 2009.

[14] Heiko Schwarz, Detlev Marpe, and Thomas Wieg. Overview of the scalable video coding extension of the h.264/avc standard. In *IEEE Transactions on Circuits and Systems for Video Technology In Circuits and Systems for Video Technology*, pages 1103–1120, 2007.

[15] T. Tran and T. Nguyen. Prioritized wireless transmissions using random linear codes. *IEEE International Symposium on Network Coding (NetCod)*, 2010.

[16] Upkar Varshney. Multicast over wireless networks. *Communications of the ACM*, 45:31–37, December 2002.

[17] Yunnan Wu, Vladimir Stankovic, Zixiang Xiong, and Sun yuan Kung. On practical design for joint distributed source and network coding. In *Network Coding, Theory, and Applications (NetCod)*, 2005.