Sliding-Window BATS Code for Scalable Video Multicasting over Erasure Networks

Xiaoran Wang, Jinbei Zhang, and Kechao Cai
School of Electronic and Communication Engineering, Sun Yat-sen University, Shenzhen, China
Emails: wangxr77@mail2.sysu.edu.cn, zhjinbei@mail.sysu.edu.cn, caikch3@mail.sysu.edu.cn

***Abstract*—Scalable video coding (SVC) is a promising technology for video transmission, which can provide different qualities of videos under different network channels. An Expanding-Window scheme based on BATS code (EW-BATS) was proposed to transmit SVC content over erasure networks [1]. When the number of receivers becomes large, EW-BATS may send important packets with a large number of times, which incurs futile redundancy. In this paper, we propose a Sliding Window BATS (SW-BATS) code to reduce the futile redundancy. By fixing the length of sliding window, the number of times packets covered by windows does not vary with the number of receivers, which can significantly reduce the futile redundancy. On the other hand, since the packets are covered by windows with a same number of times, it is a challenge for SW-BATS code to provide UEP for SVC. Thus, we characterize a joint constraint on window selection distribution and degree distributions to guarantee the decoding priority. The asymptotic analysis is carried out using And-Or tree analysis, in which the decoding performance of one class is only coupled to two adjacent classes, thus the approximate analysis on the performance could be more accurate. Furthermore, when the number of input packets is finite, we modify the asymptotic analysis by introducing a new term to characterize the impact of multiple factors to transmission overhead, such as the number of input packets and channel status. Simulation results show that the proposed SW-BATS code scheme can achieve a lower transmission overhead under a same decoding requirement.**

***Index Terms*—BATS Codes, sliding window, unequal error protection, erasure networks, decoding performance**

I. Introduction

Video transmission plays an important role in network traffic and its importance continues to grow. To ensure the smoothness of video playing, Scalable Video Coding (SVC) [2] is proposed, which divides video content into a basement layer and multiple enhancement layers. In this manner, SVC can adjust data transmission rate according to the channel status and retain reconstruction speed with varying qualities. While transmitting the SVC content over multi-hop networks, the packet loss rate may increase exponentially with the number of hops. [3] points out that the network transmission rate can reach the channel capacity by applying suitable network coding scheme. Therefore, many network coding schemes have been proposed to enhance transmission efficiency.

Random Linear Network Coding (RLNC) approach [4] [5] can greatly improve the transmission reliability and reach the channel capacity theoretically. During transmission, the intermediate nodes receive coded packets and performs random linear combination of them to generate new coded packets to send. When the number of input packets is small, RLNC performs well. However, when the number of packets is large, the decoding process brings a high computation complexity since RLNC utilizes Gaussian elimination for decoding at the receiver. In addition, the intermediate node needs to cache all the coded packets, which requires high cache capacity and coding overhead when the number of input packets is large.

Another coding scheme called Fountain Code [6]–[9] utilizes a low complexity *Belief Propagation* (BP) decoding algorithm based on the fact that coded packets follow specific degree distribution. However, for multi-hop networks, the recoding process at intermediate nodes changes the original degree of the packet, for which the packets received by receiver may hardly satisfy the requirements of BP decoding. To solve this problem, the intermediate node can perform retransmission or recoding after decoding on the received packets to ensure the network capacity, which increases the transmission delay or the computation complexity respectively.

Batched Sparse (BATS) code combines the advantages of two coding schemes mentioned above [10] [11]. A BATS code consists of an outer code and an inner code over a finite field. The outer code is a matrix extension of fountain code, which is applied to the source and generates batches of coded packets. The inner code is a random linear network code, applied to the intermediate nodes. When batches are transmitted over network, the batch degree remains unchanged while the intermediate node performs RLNC on the received batch. Therefore, when decoding at the receiver, the packets can be decoded by BP algorithm with low complexity. In addition, since RLNC is performed within the batch, only the packets in the same batch need to be cached, which reduces the caching requirement and coding overhead. To measure the decoding performance of BATS code, [12] provides an analytical approach of the decoding performance while the number of input packets tends to infinity. When the number of input packets is finite, [13] analyzes the decoding performance by deriving the distribution of stopping time of BP decoder.

The work in [1] applies BATS code for scalable video content over erasure multicast networks, where each user may have a channel of distinct quality. An Expanding-Window (EW) BATS code scheme is proposed to serve receivers with different requests by grouping the input packets into overlapped expanding windows according to their importance levels. The more important packet is covered by window for more times. Therefore, the EW-BATS code can provide Un-

ISBN 978-3-903176-65-2 © 2024 IFIP

equal Error Protection (UEP). When the number of receivers is small, the EW-BATS code performs well as the number of input packets increases asymptotically. However, when the number of receivers increases or the number of input packets decreases, the performance of EW-BATS code may deviate. The main reason for the deviation is that important packets are covered by windows multiple times and will be repeatedly sent with a large number of times. Thus, EW-BATS code may incur futile redundancy.

In this paper, we propose a new Sliding Window (SW) BATS code scheme to reduce the futile redundancy. Our major contributions are listed as follows.

1) With the introduced sliding windows, each packet will be covered by windows with a same number of times, thus decreasing the futile redundancy. When the number of input packets tends to infinity, we apply and extend the And-Or tree analysis to obtain the asymptotic decoding performance. For the theoretical analysis, since the decoding performance of one class is only coupled to two adjacent classes, the approximate analysis on the performance could be more accurate.
2) Since the important packets cannot obtain priority from a same number of overlapping windows, we characterize a joint constraint on window selection distribution and degree distributions, such that important packets can be preferably encoded and decoded. By doing so, the UEP can be guaranteed.
3) When the number of input packets is finite, the asymptotic analysis does not hold, and the approximation term in existing works could be loose [1] [14]. We then design a novel new term $f^{(i)}$ which takes into account the factors such as the number of packets requested by receivers and the channel status. This term will gradually decrease with the increasing number of input packets and will be large when the size of class is small. We note that the new design of this approximation term can be of independent interest for future works.

Finally, extensive simulations show the superior performance of SW-BATS code, especially when the number of receivers is large or the number of input packets is small.

The rest of this paper is organized as follows. The network model is represented in section II. The SW-BATS code scheme and its parameter design are provided in section III. In section IV, some simulation results are given to demonstrate the performance of proposed SW-BATS code. Finally, we conclude a summary of our results in section V.

II. System Model

The system model is shown in Fig. 1. Assume that the scalable video content at source node s is split into K input packets of length T, which are then encoded and sent to n receivers $t_1, t_2, \ldots, t_n$ over a multi-hop erasure network. To reduce the coding overhead and the encoding/decoding complexity, the coded packets are split into batches of size M, within which linear operations are performed during transmission.

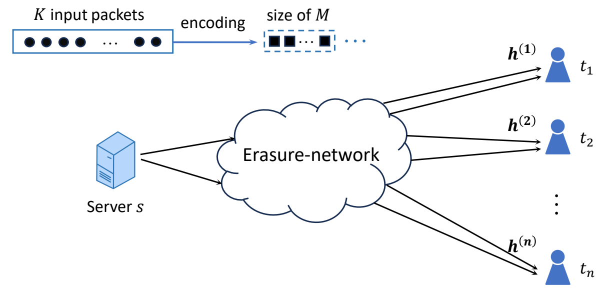


Fig. 1. System model.

The channel status between s and receiver t_i can be characterized by channel rank distribution $\mathbf{h}^{(\mathbf{i})} = [h_0^{(i)}, \ldots, h_M^{(i)}]$, where $h_r^{(i)}$ represents the probability that r independent packets remains after a batch of M coded packets is transmitted to receiver t_i. Denote $\rho_i = \frac{1}{M}\Sigma_{r=0}^{M} r h_r^{(i)}$ as the normalized throughput. Without loss of generality, assume that $\rho_1 \leq \rho_2 \leq \cdots \leq \rho_n$. Since the channel status of each receiver is heterogeneous, to better utilize the channel, receivers can request videos with distinct qualities.

According to the properties of SVC, the K input packets can be divided into one basement class and multiple enhancement classes, for which we denote the division by a distribution $\mathbf{\Pi} = [\Pi_1, \ldots, \Pi_{n_1}]$, where n_1 is the number of classes and Π_i represents proportion of the number of i-th class packets to total input packets. Assume that the smaller the index, the more important the class is. For the more important class, the UEP should be provided to guarantee the decoding priority at receiver. Without loss of generality, we assume $n_1 = n$. Since the n receivers have distinct requirements, it is reasonable to assume that receiver t_i only interests in the first i importance classes, where $i = 1, 2, \ldots, n$.

To judge whether a receiver has successfully decoded requested data, we denote decoding objective as η, which represents that the request data is considered to be decoded successfully if the receiver has recovered $(1 - \eta)$ fraction of requested data, where $\eta \ll 1$.

The BATS code scheme is to encoded input packets into a fix number (i.e., M) of coded packets at a time called a batch. Assume that column vector b_k represents the k-th input packet, where $1 \leq k \leq K$. All the data waiting for being encoded is denoted as $\mathbf{B}$ by juxtaposing the vectors of all packets. The operations involved are performed under the finite base field $GF(2^q)$. An i-th batch can be generated by following steps.

1) Choose a degree d_i by sampling from a predetermined degree distribution $\mathbf{\Psi} = [\Psi_1, \ldots, \Psi_D]$, where Ψ_j denotes the probability of choosing degree j and $1 \leq j \leq D$.
2) Select d_i packets from $\mathbf{B}$ uniformly at random and form $\mathbf{B_i}$ by juxtaposing vectors of these d_i packets.
3) Generate a $d_i \times M$ totally random matrix $\mathbf{G_i}$ called *generator matrix*.

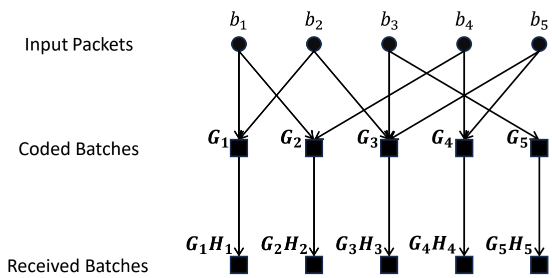


Fig. 2. Tanner graph for the inner and the outer code of a BATS code. Nodes in the first row are variable nodes, representing input packets. Nodes in the second and third rows are check nodes, representing batches.

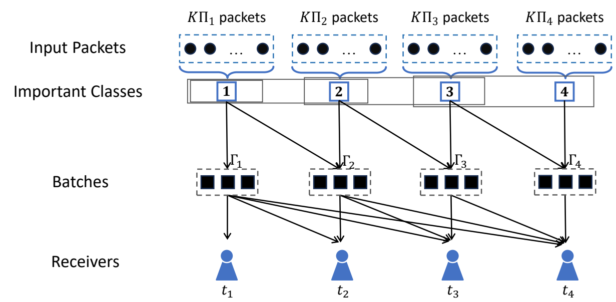


Fig. 3. Encoding process of SW-BATS code scheme.

4) Use $\mathbf{G_i}$ to encode $\mathbf{B_i}$. The encoded batch $\mathbf{X_i}$ is obtained, i.e., $\mathbf{X_i} = \mathbf{B_i}\mathbf{G_i}$.

After encoding process, the batch is sent by source node to receiver. During transmission, two linear operations will occur. One is the packet loss occurred in channels, and the other is that the intermediate node will perform RLNC on the received batch. All packet loss and RLNC encoding operations experienced during the period from the source to the receiver are uniformly represented by a matrix $\mathbf{H_i}$, which is called transfer matrix. The final received batch is $\mathbf{Y_i}$, where $\mathbf{Y_i} = \mathbf{X_i}\mathbf{H_i}$. Fig. 2 illustrates an example of encoding process.

At destination, BP decoding algorithm is utilized to decode the received batch. Batch $\mathbf{Y_i}$ is *decodable* if and only if $rk(\mathbf{G_i}\mathbf{H_i}) = d_i$, which implies that received batch retains information of all input packets encoded into this batch. The decoding process is as follows: first, choose a decodable batch i and recover d_i packets in $\mathbf{B_i}$ by solving the linear equation $\mathbf{Y_i}$=$\mathbf{B_i}\mathbf{G_i}\mathbf{H_i}$; then, substitute the values of these decoded input packets for the undecodable batches; repeat this decoding-substitution procedure until there are no more decodable batches.

Assume that the receivers successfully decode all the required packets after generating m batches of coded packets. The transmission overhead can be denoted by $\epsilon = \frac{mM-K}{K}$. The objective of this paper is to design a coding scheme that minimizes the transmission overhead while satisfying the request of each receiver.

III. Sliding Window BATS Code

In this section, we propose a new scalable video transmission scheme called Sliding Window (SW) BATS code.

A. SW-BATS code Scheme Design

Based on the content division $\mathbf{\Pi}$, the input packets are divided into n overlapping windows according to their importance levels. Set window size w as the number of importance classes included. An i-th window includes $\max(1, i-w+1)$-th to i-th classes, $i = 1, 2, \dots, n$. Intuitively, a shorter window overlap leads to less redundancy. Hence, we assume $w = 2$[1], i.e., the length of first window is set to 1, the length of other windows is set to 2. Let $\overline{\mathbf{\Pi}} = [\overline{\Pi}_1, \dots, \overline{\Pi}_n]$ denote the size of windows, i.e.,

$$\overline{\Pi}_i = \begin{cases} \Pi_i, & i = 1, \\ \Pi_i + \Pi_{i-1}, & i = 2, \dots, n. \end{cases}$$

Fig. 3 illustrates an example of encoding progress when $n = 4$. During outer coding, the i-th window is selected according to the probability Γ_i, $i = 1, 2, \dots, n$. The window selection probability distribution $\mathbf{\Gamma} = [\Gamma_1, \Gamma_2, \dots, \Gamma_n]$ is predetermined and will be designed in the next subsection. Then the packets in the chosen i-th window are encoded and transmitted based on the BATS code with the degree distribution $\mathbf{\Psi^{(i)}}$. $\mathbf{\Psi^{(i)}} = [\Psi_1^{(i)}, \dots, \Psi_{D_i}^{(i)}]$ denotes the degree distribution of the i-th window, where $\Psi_j^{(i)}$ represents the probability of choosing j packets in the i-th window to encode a batch and D_i is the maximum degree of i-th window. After encoding, the batch is sent to receivers. During transmission, the intermediate nodes will perform RLNC within the batch. Server s will repeat encoding operations until each receiver can recover its request.

The goal of this paper is to minimize the transmission overhead while ensuring the receivers can successfully decode the required packets, which is related to optimizing window selection distribution $\mathbf{\Gamma}$ and degree distributions $\{\mathbf{\Psi}^{(i)}, i = 1, 2, \dots, n\}$ and will be discussed in next subsection.

For SW-BATS code scheme, we utilize overlapping window to improve the decoding performance of receiver through the correlation of windows, but do not provide UEP through overlapping windows. To guarantee the UEP, we propose a joint constraint on $\mathbf{\Gamma}$ and $\{\mathbf{\Psi}^{(i)}, i = 1, 2, \dots, n\}$, such that the important packets can be preferably encoded and decoded, which will also be illustrated in next subsection.

B. Window Selection and Degree Distributions Optimizations

To ensure the decoding performance of each receiver, we first illustrate the measure of decoding performance.

The And-Or tree analysis method [15] is introduced to construct a tree that indicates the decoding relations between input packets and received batches. [12] has made an extension of [15] to analyze the decoding performance of classic BATS code. In this subsection, we will apply and extend the And-Or

[1]We note that the scheme and analysis in this paper can be readily extended to settings when $w \geq 3$.

tree analysis to SW-BATS code scheme. For ease of discussion, we refer to input packets as variable nodes and batches as check nodes, which are also classified by importance classes and the window indexes respectively.

We first derive the probability of variable nodes and check nodes being adjacent. For an i-th type variable node, the probability of being adjacent with a check node is equal to the probability of being selected when encoding a batch. It is obvious that an i-th type variable node can be selected for encoding only when the $(i-1)$-th window or the i-th window are selected. Therefore, the probability of an i-th type variable node being adjacent to a random check node is

$$p_i = \frac{\Gamma_i \mu_i}{K\overline{\Pi}_i} + \frac{\Gamma_{i+1}\mu_{i+1}}{K\overline{\Pi}_{i+1}},$$

where $\mu_i = \sum_{d=1}^{D_i} d\Psi_d^{(i)}$ denotes the expected degree of the i-th window. Let $\Gamma_{n+1} = 0$, such that the expression for p_i is applicable to $i = 1, 2, \ldots, n$. Thus, the probability that an i-th type variable node is adjacent to j check nodes is

$$\Lambda_j^{(i)} = \binom{m}{j} p_i^j (1-p_i)^{m-j}, 1 \le j \le m,$$

where $\mathbf{\Lambda^{(i)}}$ follows a binomial distribution. On the other hand, since each batch is encoded by packets, the probability of an i-th type check node being adjacent to a random variable node is equal to 1. The probability of an i-th type check node being adjacent to j variable nodes is $\Psi_j^{(i)}$, which denotes that there are j input packets used to encode this batch.

Assume that the distance of two adjacent nodes is 1. For i-th type variable nodes, an And-Or tree $G_{2l-1,i}$ can be constructed by selecting an i-th type variable node and its $(2l-1)$-neighborhood subgraph, which is shown as follows:

1) The root of the tree is an i-th type variable node, which is at level 0.
2) Each node at an even level is a variable node, called Or-node, and each node at an odd level is a check node, called And-node.
3) For $j \ge 1$, a non-leaf i-th type Or-node has $j-1$ children at the next lower level with probability $\alpha_{i,j}$, where $\alpha_{i,j} = \Lambda_j^{(i)} \cdot \frac{j}{\theta_i}$ and $\theta_i = mp_i$ denotes the expected number of And-nodes that is adjacent to an i-th type Or-node.
4) For $j \ge 1$, a non-leaf i-th type And-node has $j-1$ children at the next lower level with probability $\beta_{i,j}$, where $\beta_{i,j} = \Psi_j^{(i)} \cdot \frac{j}{\mu_i}$.
5) An i-th type Or-node can only have $\{i, i+1\}$-th type And-node as children with the probability of $\{p_i^{(i)}, p_{i+1}^{(i)}\}$, where $p_j^{(i)} = \frac{\Gamma_j \mu_j / K\overline{\Pi}_j}{p_i}, j = i, i+1$.
6) An i-th type And-node can only have $\{i-1, i\}$-th type Or-node as children with the probability of $\{q_{i-1}^{(i)}, q_i^{(i)}\}$, where $q_j^{(i)} = \frac{\Pi_j}{\overline{\Pi}_i}, j = i-1, i$.

With our constructed And-Or tree $GT_{2l-1,i}$, we can iteratively solve the undecodable probability of each node from bottom up similar to [12]. Then, the asymptotical decoding performance can be obtained as follows.

Lemma 1: For $l > 1, 1 \le i \le n$, under the channel status $\mathbf{h} = [h_0, h_1, \ldots, h_M]$, let $y_{l,i}$ denote the undecodable probability of an i-th type variable node after l iterations. We have

$$\begin{aligned} y_{l,i} = \exp\Bigg\{ -\frac{1+\epsilon}{M}\sum_{r=1}^{M} h_r \Big[\frac{\Gamma_i}{\overline{\Pi}_i} \sum_{d=1}^{D_i} d\Psi_d^{(i)} \\ \sum_{s=0}^{r-1} \xi(d, s, \frac{\Pi_{i-1}}{\overline{\Pi}_i} y_{l-1,i-1} + \frac{\Pi_i}{\overline{\Pi}_i} y_{l-1,i}) + \frac{\Gamma_{i+1}}{\overline{\Pi}_{i+1}} \sum_{d=1}^{D_{i+1}} d \\ \Psi_d^{(i+1)} \sum_{s=0}^{r-1} \xi(d, s, \frac{\Pi_{i+1}}{\overline{\Pi}_{i+1}} y_{l-1,i+1} + \frac{\Pi_i}{\overline{\Pi}_{i+1}} y_{l-1,i}) \Big] \Bigg\}, \end{aligned} \tag{1}$$

where $\xi(d, s, x) = \binom{d-1}{s} x^s (1-x)^{d-1-s}$.

Note that the decoding performance of each class is interrelated, since the variable nodes and check nodes in our scheme are classified into multiple classes. This is also the main difference between (1) and the expression of y_i in [12].

Given SW-BATS $(\mathbf{\Gamma}, \mathbf{\Psi}^{(1)}, \mathbf{\Psi}^{(2)}, \ldots, \mathbf{\Psi}^{(n)})$, with And-Or tree analysis, a receiver with the channel status $\mathbf{h}$ can decode up to $(1-\eta)$ fraction of i-th importance class if the undecodable probability of i-th type variable node satisfies $\lim_{l\to\infty} y_{l,i} \le \eta$. In other words, during iteration, the value of $y_{l,i}$ is monotonically decreasing and eventually converges to a value that less than or equal to η. Therefore, the receiver can successfully decode i-th class if and only if there is $y_{l,i} \le y_{l-1,i}$ when $\eta \le y_{l-1,i} \le 1, l = 2, 3, \ldots$.

However, in (1), the undecodable probability of i-th type variable node is coupled with two neighboring types. To analyze the decoding performance of i-th class seperatedly, we first decouple the undecodable probability. Based on the weighted BATS codes [16], we assume that the decoding probability of an variable node is positively correlated with its probability of being selected at encoding, i.e., assume

$$\log(y_{l,j}) = \frac{p_i}{p_j}\log(y_{l,i}), 1 \le i, j \le n.$$

Let $\lambda_{i,j} = \frac{p_i}{p_j}$, then we have

$$y_{l,j} = (y_{l,i})^{\lambda_{i,j}}, \tag{2}$$

where

$$\lambda_{i,j} \approx \frac{\Gamma_i/\overline{\Pi}_i + \Gamma_{i+1}/\overline{\Pi}_{i+1}}{\Gamma_j/\overline{\Pi}_j + \Gamma_{j+1}/\overline{\Pi}_{j+1}}, 1 \le i, j \le n,$$

in which the approximation is obtained by ignoring the difference of expected degree of each window.

Since the undecodable probabilities are decoupled, (1) can be expressed as the function of $y_{l-1,i}$, i.e., $y_{l,i} = f(y_{l-1,i})$. Therefore, to guarantee the decoding performance of receiver t_i, under the channel station $\mathbf{h^{(i)}}$, the inequality $f(y_{l-1,i}) \le y_{l-1,i}$ must be hold while $\eta \le y_{l-1,i} \le 1$. Perform a logarithmic transformation on both sides of the inequality and replace $y_{l-1,i}$ with the variable x. Then we can obtain the inequality (3) to guarantee the decoding performance of i-th class at receiver t_i, which poses an inequality constraint on

$\boldsymbol{\Gamma}$ and $\{\boldsymbol{\Psi}^{(i)}, i = 1, 2, \ldots, n\}$. Let $D_1 = D_2 = \ldots, D_n = \lceil M/\eta \rceil - 1$. With (1) and (2), we have

$$\sum_{d=1}^{D} d \sum_{r=1}^{M} h_r^{(i)} \Big[\frac{\Gamma_i}{\overline{\overline{\Pi}}_i} \Psi_d^{(i)} \sum_{s=0}^{r-1} \xi(d, s, \frac{\Pi_{i-1}}{\overline{\overline{\Pi}}_i} x^{\lambda_{i,i-1}} + \frac{\Pi_i}{\overline{\overline{\Pi}}_i} x) + \frac{\Gamma_{i+1}}{\overline{\overline{\Pi}}_{i+1}} \Psi_d^{(i+1)} \sum_{s=0}^{r-1} \xi(d, s, \frac{\Pi_i}{\overline{\overline{\Pi}}_{i+1}} x + \frac{\Pi_{i+1}}{\overline{\overline{\Pi}}_{i+1}} x^{\lambda_{i,i+1}}) \Big] + \theta \ln(x) \geq 0, \tag{3}$$

where $\eta \leq x \leq 1$. $\theta = \frac{M}{1+\epsilon}$ denotes the effective coding rate, which represents the ratio of the number of input packets and batches sent. The objective of this paper is to minimize ϵ, which is equal to maximize θ with (3) hold.

Compared to [1], from (3), the decoding performance analysis of the i-th class involves only two adjacent windows, which makes the number of inter-related classes smaller during decoupling and improves the accuracy of analysis.

Since each class is covered by windows for a same number of times, we cannot obtain UEP from the overlapping windows. Therefore, *Lemma 2* is proposed to characterize a joint constraint on $\boldsymbol{\Gamma}$ and $\{\boldsymbol{\Psi}^{(i)}, i = 1, 2, \ldots, n\}$, from which UEP can be guaranteed.

Lemma 2: For any receiver t_i, if the following inequality holds, i.e.,

$$\frac{\sum_{d=1}^{D} d\Gamma_i \Psi_d^{(i)} f(d, r, x)}{\overline{\overline{\Pi}}_i} \geq \frac{\sum_{d=1}^{D} d\Gamma_{i+2} \Psi_d^{(i+2)} f(d, r, x)}{\overline{\overline{\Pi}}_{i+2}}, \tag{4}$$

where $i = 1, 2, \ldots, n-2$, $\eta \leq x \leq 1$ and

$$f(d, r, x) = \begin{cases} 1, r = 0, \\ \sum_{s=0}^{r-1} \xi(d, s, x), r = 1, 2, \ldots, M, \end{cases}$$

we have $y_{l,1} \leq \cdots \leq y_{i,i}$ for receiver t_i regardless the value of l, where $1 \leq i \leq n$.

When *Lemma 2* holds for a receiver t_i, as long as the least important class of its request can be recovered, the more important classes can be recovered as well. If there is a sudden error occurred in a channel and the receiver is unable to decode all request data with the expected transmission overhead, the more important data is decoded at receiver with priority due to UEP, which allows the receiver to obtain the maximum amount of information.

Based on (3) and (4), we are ready to present the maximization problem of θ as follow, where the window selection distribution and degree distributions are to be optimized.

$$\begin{aligned} \mathbf{P1}: & \max_{\boldsymbol{\Gamma}, \boldsymbol{\Psi}^{(1)}, \ldots, \boldsymbol{\Psi}^{(n)}} \theta \\ s.t.\ & (3), \eta \leq x \leq 1, i = 1, \ldots, n; \\ & (4), \eta \leq x \leq 1, i = 1, \ldots, n-2, r = 1, \ldots, M; \\ & \sum_{i=1}^{n} \Gamma_i = 1, \Gamma_i > 0, i = 1, \ldots, n; \\ & \sum_{d=1}^{D} \Psi_d^{(i)} = 1, \Psi_d^{(i)} \geq 0, i = 1, \ldots, n, d = 1, \ldots, D. \end{aligned} \tag{5}$$

Note that $\theta = \frac{M}{1+\epsilon}$, thus maximizes θ is equal to minimize ϵ.

Since Γ_i is multiplied by $\boldsymbol{\Psi}^{(\mathbf{i})}$ and is also located at the exponential position of x in (3) and (4), **P1** is not a linear optimization problem, which is difficult to solve. To simplify **P1**, some assumptions are made as follows.

1) Let $\hat{\Psi}_d^{(i)} = \Gamma_i \Psi_d^{(i)}$, $i = 1, 2, \ldots, n, d = 1, 2, \ldots, D$. Then the exponential number of $\hat{\Psi}_d^{(i)}$ is all 1.
2) Since $\lambda_{i,j}$ is correlated with $\boldsymbol{\Gamma}$, we choose to assign initial value to $\lambda_{i,j}$. Ideally, if the packets of i-th class can be successfully received by t_i, they can also be received by t_{i+1} to t_n. Therefore, the number of coded packets encoded by each window is $\frac{K\overline{\overline{\Pi}}_i}{\rho_i}$ respectively. The initial window selection distribution can be denoted as $\Gamma_i^{(0)} = \frac{\overline{\overline{\Pi}}_i/\rho_i}{\sum_{j=1}^{n} \overline{\overline{\Pi}}_j/\rho_j}$, $i = 1, 2, \ldots, n$. Then $\lambda_{i,j}$ can be computed as a constant $\hat{\lambda}_{i,j}$.
3) Discretize x into a series of points in the interval $[\eta, 1]$.

Note that the $\lambda_{i,j}$ is different with [1], which could be more closer to the optimal result since we set $\Gamma_i^{(0)}$ as the optimal value of SW-BATS code under ideal scenario.

After simplification, we can obtain a linear optimization problem **P2**.

$$\mathbf{P2}: \max_{\hat{\boldsymbol{\Psi}}^{(1)}, \ldots, \hat{\boldsymbol{\Psi}}^{(n)}} \theta$$

$$s.t. \sum_{d=1}^{D} d \sum_{r=1}^{M} h_r^{(i)} \Big[\frac{\hat{\Psi}_d^{(i)}}{\overline{\overline{\Pi}}_i} \sum_{s=0}^{r-1} \xi(d, s, \frac{\Pi_{i-1}}{\overline{\overline{\Pi}}_i} x^{\hat{\lambda}_{i,i-1}} + \frac{\Pi_i}{\overline{\overline{\Pi}}_i} x) + \frac{\hat{\Psi}_d^{(i+1)}}{\overline{\overline{\Pi}}_{i+1}} \sum_{s=0}^{r-1} \xi(d, s, \frac{\Pi_i}{\overline{\overline{\Pi}}_{i+1}} x + \frac{\Pi_{i+1}}{\overline{\overline{\Pi}}_{i+1}}) \Big] + \theta \ln(x) \geq 0,$$
$$\eta \leq x \leq 1, i = 1, \ldots, n; \tag{6a}$$

$$\frac{\sum_{d=1}^{D} d\hat{\Psi}_d^{(i)} f(d, r, x)}{\overline{\overline{\Pi}}_i} \geq \frac{\sum_{d=1}^{D} d\hat{\Psi}_d^{(i+2)} f(d, r, x)}{\overline{\overline{\Pi}}_{i+2}},$$
$$\eta \leq x \leq 1, i = 1, \ldots, n-2, r = 1, \ldots, M; \tag{6b}$$

$$\sum_{i=1}^{n} \sum_{d=1}^{D} \hat{\Psi}_d^{(i)} = 1; \tag{6c}$$

$$\hat{\Psi}_d^{(i)} >= 0, i = 1, \ldots, n, d = 1, \ldots, D. \tag{6d}$$

Assume that receiver t_i only receives batches encoded by first i windows, i.e., $y_{l,i+1} = 1$ for receiver t_i, $1 \leq i \leq n$. Therefore, it is reasonable to set $\hat{\lambda}_{i,i+1} = 0$ for receiver t_i.

In addition, the value of $\hat{\lambda}_{i,j}$ in **P2** is not the optimal value, which may influence the result of θ. To obtain the optimal solution of **P2**, we first solve **P2** with $\hat{\lambda}_{i,j}$, and then update the value of $\hat{\lambda}_{i,j}$ to resolve it. Repeat the process until $\boldsymbol{\Gamma}$ converges, then we can obtain the optimal result.

C. Finite-length Analysis

When the number of packets to be transmitted tends to infinity, the optimal solution of **P2** can be obtained via iteration. However, in practical applications, the number of packets is usually finite, for which the practical decoding performance will deviate from the asymptotic case. The main reason is that the degree distributions obtained in the asymptotic analysis may be scattered, i.e., the degree obtained by sampling from these degree distributions is likely to be larger than the number of input packets. Thus, one needs to increase the probability of a smaller degree being selected.

In [1], an additional term was added in the optimization problem, which shows good performance when the number of packets is small. However, when the number of packets increases, the adding term quickly decreases to 0, resulting in the performance deviation. In this paper, we redesign this new term, denoted as $f^{(i)}$ for receiver t_i. Intuitively, the larger the number of packets that can be decoded at receiver, the larger θ is, i.e., θ is positively correlated with the number of packets that can be decoded. In addition, the effective coding rate decreases as the decoded portion increases. Hence, we assume $f^{(i)} = \frac{c(i)}{K\sum_{j=1}^{i}\Pi_j}x^{c_1}$, where $c(i) = K^{c_2}M\rho_i\frac{\overline{\Pi}_n}{\overline{\Pi}_i}$, c_1, c_2 are parameters that can be tuned. When K tends to infinity, $f^{(i)}$ tends to zero and θ tends to asymptotic performance. Our design is different with the design of [1] in two main points:

1) We increase the impact of the number of packets to $\mathbf{\Gamma}$ and $\{\mathbf{\Psi}^{(i)}, i = 1, 2, \ldots, n\}$, which makes it more representative of their impact on transmission overhead and is shown in simulation.
2) Compared to [1], we consider the effect of window size on transmission overhead, i.e., $\frac{\overline{\Pi}_n}{\overline{\Pi}_i}$ in $f^{(i)}$. Intuitively, the smaller the window size is, the more it offsets to asymptotic performance.

Adding $f^{(i)}$ to the asymptotic optimization problem **P2**, we can obtain **P3** for finite-length decoding performance.

$$
\begin{aligned}
\mathbf{P3}: &\max_{\hat{\mathbf{\Psi}}^{(\mathbf{1})},\ldots,\hat{\mathbf{\Psi}}^{(\mathbf{n})}} \theta \\
s.t. &\sum_{d=1}^{D} d \sum_{r=1}^{M} h_r^{(i)} \Big[\frac{\hat{\Psi}_d^{(i)}}{\overline{\Pi}_i}\sum_{s=0}^{r-1}\xi(d,s,\frac{\Pi_{i-1}}{\overline{\Pi}_i}x^{\hat{\lambda}_{i,i-1}} + \frac{\Pi_i}{\overline{\Pi}_i}x) + \\
&\qquad \frac{\hat{\Psi}_d^{(i+1)}}{\overline{\Pi}_{i+1}}\sum_{s=0}^{r-1}\xi(d,s,\frac{\Pi_i}{\overline{\Pi}_{i+1}}x + \frac{\Pi_{i+1}}{\overline{\Pi}_{i+1}})\Big] + \\
&\qquad \theta[\ln(x) - f^{(i)}] \geq 0, \eta \leq x \leq 1, i = 1, \ldots, n; \\
&(6b) - (6d).
\end{aligned} \tag{7}
$$

Note that similar to [1], **P3** is not exact optimal. However, it will be shown in the simulations that, our design of this additional term will lead to a good performance.

IV. Numerical Results

In this section, we simulate and analyze the performance of the proposed SW-BATS code scheme.

A. Parameter Settings

All simulations are carried out under the $GF(2^8)$ base field. Assume the batch size $M = 8$. For the channel status simulation of each receiver t_i, we uniformly and randomly choose a value in the interval $[0, 0.5]$ to represent its channel loss and the corresponding channel rank distribution follows a binomial distribution. In order to make channel rank distributions not "strictly ordered" [1], we set $\mathbf{h}^{(\mathbf{1})} = [0.4, 0, 0, 0, 0.1, 0.1, 0.1, 0.15, 0.15]$. For example, if $n = 4$, the throughput of all channels is $\rho_1 = 0.46875$, $\rho_2 = 0.625$, $\rho_3 = 0.75$, $\rho_4 = 0.875$. Set the decoding objective $\eta = 0.01$.

B. Asymptotic Performance

In this subsection, we compare the asymptotic performance of the proposed SW-BATS code and the EW-BATS code [1]. The decoding performance is represented by the transmission overhead ϵ, which varies with n. Since the content division also has an impact on decoding performance, we simulate three cases where the content division is uniformly distributed, uniformly increasing and uniformly decreasing, respectively. When $n = 4$, we simulate the content division as $[\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$, $[\frac{1}{8}, \frac{5}{24}, \frac{7}{24}, \frac{3}{8}]$ and $[\frac{3}{8}, \frac{7}{24}, \frac{5}{24}, \frac{1}{8}]$ for these three cases.

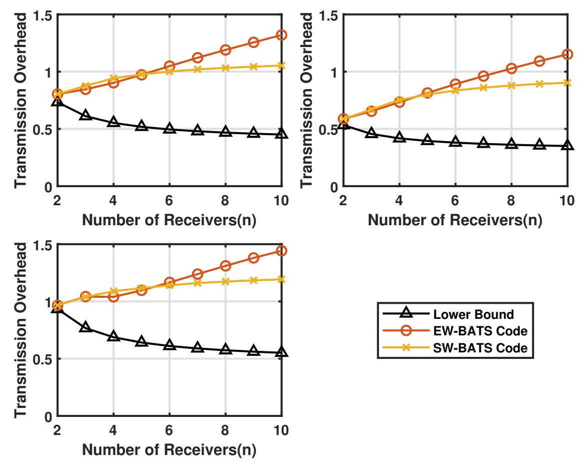


Fig. 4. Comparison of asymptotic performance.

In Fig. 4, each subfigure corresponds to a content division, which is mentioned above. In each subfigure, the *Lower Bound* indicates the transmission overhead incurred in the ideal scenario, where the i-th coded packets sent by the source based on the i-th channel status can be successfully decoded by the receivers t_i to t_n; the *EW-BATS Code* line represents the transmission overhead obtained by solving the linear optimization problem **P2** in [1], and the *SW-BATS Code* line is obtained by solving **P2** in section III. Simulation result shows that when $n = 2$, the transmission overhead of SW-BATS code is equal to that of EW-BATS code, which agrees with the theoretical analysis. When the number of receivers n is larger than 5, the SW-BATS code scheme performs better than the EW-BATS code scheme as the number of input packets approaches infinity. The performance gain reaches at least 17.27% when $n = 10$ in Fig. 4. We also note that, if the number of input packets is limited, the SW-BATS code scheme could also outperform EW-BATS code when $n \leq 5$, which is discussed in the next subsection.

In order to verify whether the decoding priority of the importance classes is guaranteed at each receiver, we simulate the decoding performance of each class when $n = 4$. We take transmission overhead ϵ as an independent variable and observe the decoding performance of each class at each receiver when the transmission overhead varies. The simulation result is shown in Fig. 5.

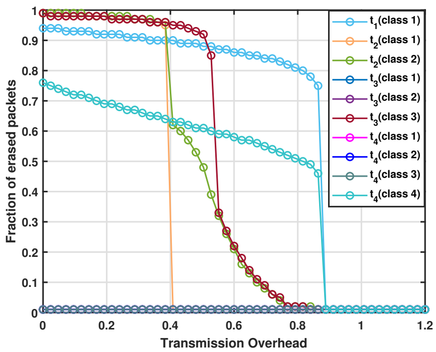


Fig. 5. Variation of asymptotic decoding performance in relation to transmission overhead.

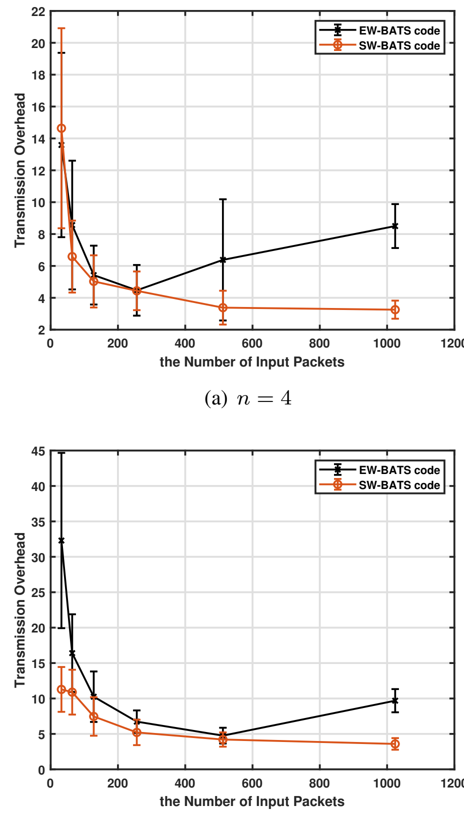
Simulation result shows that the more important class has a smaller fraction of erased packets at the same receiver under the same condition of transmission overhead, which reflects the decoding priority of important classes.

C. Finite-length Performance

The transmission overhead will deviate from the asymptotic performance when the number of input packets is limited. To obtain the finite-length performance of EW-BATS code and SW-BATS code, we first design the optimal parameters for each scheme. Then we simulate actual communication process and apply coding scheme by using parameters solved above, through which we can obtain multiple samples of transmission. With the samples, we can analyze the trend of the decoding performance with the number of input packets.

In this simulation, we simulate the finite-length performance of EW-BATS code by solving **P3** proposed in [1] as a control experiment for our scheme. For SW-BATS code, we solve **P3** mentioned in III-C, where $c_1 = 0.25$, $c_2 = 0.25$. We simulate the transmission overhead when the number of receivers is 4 and 6 with 2^k input packets respectively, $k = 5, 6, \ldots, 10$. Repeat each simulation 50 times and plot the mean and standard deviation in Fig. 6.

In Fig. 6, the performance difference of EW-BATS code and SW-BATS code is shown as the line *EW-BATS code* and the line *SW-BATS code*, from which we can find that both coding schemes have high overhead when the number of input packets K is small. In addition, the standard deviation of the transmission overhead is also large, since a relatively small number of samples may make the probability-based window selection approach lead to a large redundancy. Nevertheless, the average value of transmission overhead for SW-BATS code scheme is generally smaller than that of EW-BATS code scheme. As the number of input packets increases, the transmission overhead of SW-BATS code scheme decreases exponentially. However, when $n = 4$, different from SW-BATS



(a) $n = 4$

(b) $n = 6$

Fig. 6. Comparison of finite-length performance.

code, the transmission overhead of EW-BATS increases when the number of input packets is large than 256, which implies that the finite-length analysis of EW-BATS code scheme could be loose when the number of packets is large. Therefore, as the number of input packets increases, the performance gap between the two schemes first decreases and then increases. In addition, as the number of receivers increases, the benefit brought by SW-BATS code increases generally. For example, when $n = 4$ and $K = 64$, the average overhead of SW-BATS code scheme reduces by about 23.09% compared to EW-BATS code. When $n = 6$, the average overhead can reduce by about 33.59%, which implies that the larger the number of receiver is, the greater the gain brought by SW-BATS code.

In addition, for the variation of the standard deviation of the two schemes, we find that the standard deviation of SW-BATS code scheme is always smaller, which indicates that the performance of SW-BATS code scheme is more stable.

D. Scalable Video Multicasting

In this subsection, we explore the performance of SW-BATS code scheme when the source s is multicasting H.264 SVC coded video stream. Similar to [1], we use Stefan video sequence, which has a spatial resolution of 352×288 and a temporal resolution of 30 fps. The video sequence is first segmented into groups of frames (GOFs), each of which

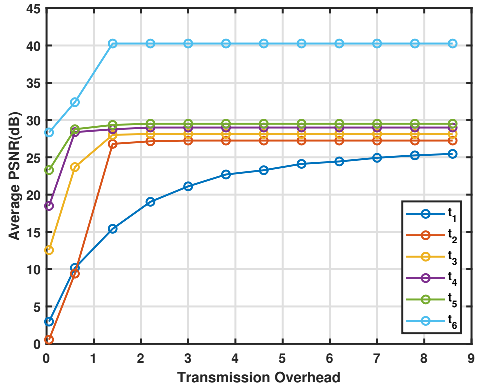


Fig. 7. The achievable PSNR by EW-BATS code.

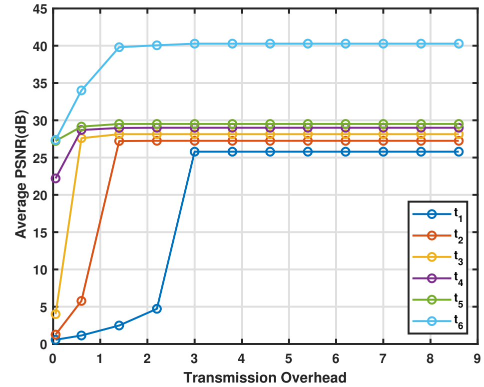


Fig. 8. The achievable PSNR by SW-BATS code.

contains 16 frames. Then, each GOF is seemed as a source block and encoded into one base layer and 14 enhancement layers, which contains $K = 3800$ packets of 50 bytes.

Assume that there are 6 receivers request for video sequence of different qualities. The request of each receiver and the corresponding average Y-PSNR are given in Table I.

TABLE I
REQUESTS OF DIFFERENT QUALITIES AND THE CORRESPONDING Y-SNR.

Index	Request	Size	$\sum_{j=1}^{i} \Pi_j$	Y-PSNR[dB]
1	BL only	400	0.105	25.79
2	BL + 1 ELs	700	0.185	27.25
3	BL + 2 ELs	875	0.23	28.14
4	BL + 3 ELs	1155	0.305	29.00
5	BL + 4 ELs	1550	0.4	29.51
6	BL + All ELs	3800	1	40.28

We use EW-BATS code and SW-BATS code schemes to encode the input packets after SVC classification, the simulation results of which are shown in Fig. 7 and Fig. 8 respectively. We choose average PSNR as performance metric to measure the video quality after reconstruction at each receiver. Fig. 7 shows that while the transmission overhead of EW-BATS code is nearly 9, all receivers can reconstruct the video with their expected qualities. Compared to EW-BATS code, SW-BATS code scheme can achieve the same requirements with a smaller overhead, as shown in Fig. 8.

V. CONCLUSION

In this paper, we propose a SW-BATS code scheme for the scalable video content multicasting, which may outperform EW-BATS code when the number of receivers is large or the number of input packets is small. For future work, it would be interesting to explore how to design precoding or recoding scheme that can further reduce the transmission overhead.

ACKNOWLEDGMENTS

This work was supported in part by National Key R&D Program of China under Grant 2022YFB2902700, NSF China (Grant No. 62471505, 62071501, 62202508), Shenzhen Science and Technology Program (Grant JCYJ20220818102011023, 20220817094427001, ZDSYS20210623091807023).

REFERENCES

[1] X. Xu, Y. Zeng, Y. L. Guan, and L. Yuan, "Expanding-window bats code for scalable video multicasting over erasure networks," *IEEE Transactions on Multimedia*, vol. 20, no. 2, pp. 271–281, 2018.
[2] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, 2007.
[3] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.
[4] S.-Y. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.
[5] T. Ho, M. Medard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.
[6] J. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1528–1540, 2002.
[7] M. Luby, "LT codes," in *the 43rd Annual IEEE Symposium on Foundations of Computer Science.*, Vancouver, BC, Canada, Nov. 2002.
[8] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.
[9] P. Cataldi, M. Grangetto, T. Tillo, E. Magli, and G. Olmo, "Sliding-window Raptor Codes for Efficient Scalable Wireless Video Broadcasting With Unequal Loss Protection," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1491–1503, 2010.
[10] S. Yang and R. W. Yeung, "Coding for a network coded fountain," in *2011 IEEE International Symposium on Information Theory Proceedings*, St. Petersburg, Russia, Jul. 2011.
[11] ——, "Batched sparse codes," *IEEE Transactions on Information Theory*, vol. 60, no. 9, pp. 5322–5346, 2014.
[12] S. Yang and Q. Zhou, "Tree analysis of bats codes," *IEEE Communications Letters*, vol. 20, no. 1, pp. 37–40, 2016.
[13] T.-C. Ng and S. Yang, "Finite-length analysis of BATS codes," in *2013 International Symposium on Network Coding (NetCod)*, Calgary, AB, Canada, Jun. 2013.
[14] X. Xu, Y. L. Guan, Y. Zeng, and C.-C. Chui, "Quasi-universal bats code," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3497–3501, 2017.
[15] M. G. Luby, M. Mitzenmacher, and M. A. Shokrollahi, "Analysis of random processes via And-Or tree evaluation," in *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, ser. SODA '98, USA, Jan. 1998.
[16] X. Xu, Y. Zeng, Y. L. Guan, and L. Yuan, "BATS code with unequal error protection," in *2016 IEEE International Conference on Communication Systems (ICCS)*, Shenzhen, China, Dec. 2016.