2023 IEEE/CIC International Conference on Communications in China (ICCC Workshops) | 979-8-3503-4540-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCCWorkshops57813.2023.10233707

Reduced Complexity SCL Decoding of U-UV Codes Through List Pruning

Wenhao Chen†, Li Chen†, Huazi Zhang‡
†School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China, 510006
‡Hangzhou Research Center, Huawei Technologies Co. Ltd., Hangzhou, China, 310052
Email: chenwh85@mail2.sysu.edu.cn, chenli55@mail.sysu.edu.cn, zhanghuazi@huawei.com

***Abstract*—U-UV structural coding with algebraic component codes can provide competent error-correction performance in the short-to-medium length regime. Constituted by BCH component codes and its ordered statistics decoding (OSD), the successive cancellation list (SCL) decoding of U-UV codes can outperform that of cyclic redundancy check (CRC)-polar codes. However, this list decoding complexity becomes formidable as the decoding output list size L increases. This paper proposes the list pruning techniques for reducing the SCL complexity. It eliminates the unpromising decoding paths, in an effort to reduce redundant decoding operations. We show that this can be realized through either estimating the *a posterior* probability (APP) of a decoding path, or estimating the dynamic performance loss. Subsequently, two list decoding schemes are proposed for U-UV codes. Our simulation results show that they can both effectively reduce the SCL decoding complexity of U-UV codes with marginal performance loss.**

***Index Terms*—List pruning, reduced complexity, successive cancellation list decoding, U-UV codes**

I. Introduction

Future communications require ultra low-latency information recovery, for which the short-to-medium length channel codes will play an important role. However, the capacity-approaching feature of modern codes, such as turbo codes [1], low-density parity-check (LDPC) codes [2] and polar codes [3], is realized based on a large codeword length. They inevitably inherit a large decoding latency. In the short-to-medium length regime, BCH codes, tail-biting convolutional codes, cyclic redundancy check (CRC)-polar codes, and the more recent polarization adjusted convolutional (PAC) codes [4] are known to be the competent coding schemes [5]. U-UV structural codes were recently proposed as another competent candidate [6]. It is constructed by a number of small algebraic component codes through the $(\mathrm{U}\,|\,\mathrm{U}+\mathrm{V})$ structure, where the U codes and V codes are component codes. The $(\mathrm{U}\,|\,\mathrm{U}+\mathrm{V})$ structure is also known as the Plotkin structure [7]. By specifying the component codes, this structural code is also seen as the generalized concatenated codes (GCC) [8]–[10], where the inner codes are polar codes. This structural coding also results in a number of subchannels with polarized capacities. Note the channel polarization is realized when most of the subchannel capacities reach either 1 or 0. Consequently, the component code rates can be designed based on the subchannel capacities. It has been shown in [6] that employing the BCH component codes and its ordered statistics decoding (OSD) [11], the successive cancellation list (SCL) decoding of U-UV codes can substantially outperform that of CRC-polar codes.

The SCL decoding complexity of U-UV codes is dominated by the component codes' OSD. Its list decoding feature also implies a high decoding complexity [6]. The low complexity OSD variants can be employed to reduce the component code decoding complexity, e.g., the box-and-match algorithm [12] [8] and the hybrid OSD [13] can both be employed. On the other hand, avoiding the expansion of the unpromising SCL decoding paths is another approach to realize the complexity reduction. This so called path pruning technique has been explored in SCL decoding of polar codes [14]–[18].

In this paper, the list pruning techniques are proposed to reduce the SCL decoding complexity for U-UV codes. Eliminating the unpromising SCL decoding paths can result in a considerable complexity reduction. We show that this can be realized through either estimating the *a posterior* probability (APP) of a decoding path, or estimating the dynamic performance loss. Subsequently, two list pruning schemes are introduced for U-UV codes. Performance loss of the pruning schemes is analyzed, which sheds light on the determination of the pruning thresholds. Our simulation results show that the proposed pruning schemes can both effectively reduce the SCL decoding complexity of U-UV codes with a marginal performance loss.

II. Preliminaries

This section introduces the U-UV code construction using BCH component codes. The SCL decoding of the U-UV codes is also introduced.

A. U-UV Code Construction

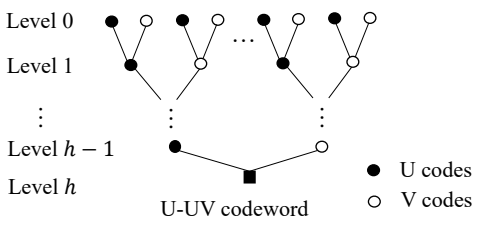

Fig. 1. Recursive construction of an h-level U-UV code.

Let U code and V code be two linear block codes of length n with dimensions k_{U} and k_{V}, respectively. A U-UV code of

length $2n$ and dimension $k_{\mathrm{U}} + k_{\mathrm{V}}$ can be constructed by [7]

$$\{(\boldsymbol{c}_{\mathrm{U}} \,|\, \boldsymbol{c}_{\mathrm{U}} + \boldsymbol{c}_{\mathrm{V}});\ \boldsymbol{c}_{\mathrm{U}} \in \mathbb{C}_{\mathrm{U}} \text{ and } \boldsymbol{c}_{\mathrm{V}} \in \mathbb{C}_{\mathrm{V}}\}, \tag{1}$$

where $\mathbb{C}_{\mathrm{U}}$ and $\mathbb{C}_{\mathrm{V}}$ denote the codebooks of the U code and the V code, respectively, and $\boldsymbol{c}_{\mathrm{U}}$ and $\boldsymbol{c}_{\mathrm{V}}$ are their codewords. In this work, the above U code and V code are two binary primitive BCH codes. This construction can be extended recursively by involving more BCH component codes, resulting in a U-UV code with a larger level of construction. Fig. 1 illustrates the construction of an h-level U-UV code. At level 0, there are $\gamma = 2^h$ BCH component codes of length n. This h-level construction yields a U-UV codeword of length $N = \gamma n$.

This U-UV code construction results in γ subchannels with polarized capacities at level 0, which convey the component codes. By specifying the component codes, the U-UV code can also be seen as a GCC with inner polar codes [8]–[10]. Fig. 2 shows the GCC interpretation. In particular, let $\mathbb{C}^{(i)}$ and $\boldsymbol{c}^{(i)} = (c_0^{(i)}, c_1^{(i)}, ..., c_{n-1}^{(i)})$ denote the ith BCH component code and its codeword, respectively, where $i = 0, 1, ..., \gamma - 1$. The inner codes are n polar codes of length γ. Input of the jth polar encoder will be the jth codeword symbol of all BCH component codes, i.e., $\boldsymbol{c}'_j = (c_j^{(0)}, c_j^{(1)}, ..., c_j^{(\gamma-1)})$, where $j = 0, 1, ..., n-1$. The U-UV codeword $\boldsymbol{v}$ is obtained by cascading the output of the n polar encoders, i.e., $\boldsymbol{v} = (v_0, v_1, ..., v_{\gamma-1}, ..., v_{N-\gamma}, v_{N-\gamma+1}, ..., v_{N-1})$.

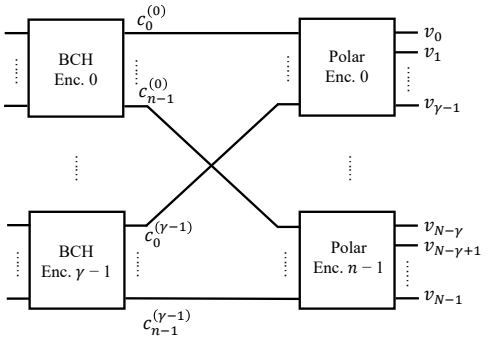

Fig. 2. GCC interpretation of a U-UV code.

B. SCL Decoding of U-UV Codes

The SCL decoding of U-UV codes can also be illustrated under the GCC paradigm of Fig. 2. Assume that a U-UV codeword $\boldsymbol{v}$ of length N is transmitted over the additive white Gaussian noise (AWGN) channel using BPSK. Let $\boldsymbol{y} = (y_0, y_1, ..., y_{N-1})$ denote the received symbol sequence and $\boldsymbol{\mathcal{L}} = (\mathcal{L}_0, \mathcal{L}_1, ..., \mathcal{L}_{N-1})$ denote the corresponding log-likelihood ratio (LLR) sequence with entries defined as

$$\mathcal{L}_s = \ln \frac{P(y_s \,|\, v_s = 0)}{P(y_s \,|\, v_s = 1)}. \tag{2}$$

where $s = 0, 1, ..., N-1$. In the decoding, the LLR sequence will be equally partitioned into n subsequences, which are the input of the n successive cancellation (SC) decoders. The SC decoders can function in parallel and estimate the LLR of their input symbols [3]. Once the ith input symbol LLRs of all inner polar codes have been produced, they are collected to form an LLR sequence $\boldsymbol{\mathcal{L}}^{(i)} = (\mathcal{L}_0^{(i)}, \mathcal{L}_1^{(i)}, ..., \mathcal{L}_{n-1}^{(i)})$, which is the decoder input of the ith component code. The BCH component codes are decoded by OSD successively. To assess the reliability of the BCH codeword estimations, the correlation distance between the input symbol LLR sequence $\boldsymbol{\mathcal{L}}^{(i)}$ and the ith BCH codeword estimation $\hat{\boldsymbol{c}}^{(i)} = (\hat{c}_0^{(i)}, \hat{c}_1^{(i)}, ..., \hat{c}_{n-1}^{(i)})$ is defined as

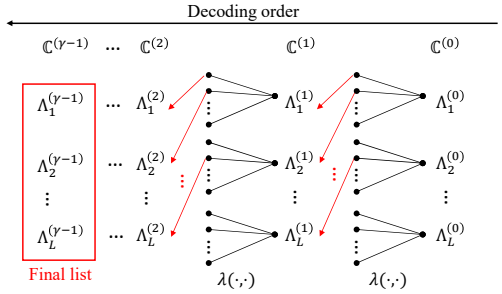

Fig. 3. Path expansion of SCL decoding.

$$\lambda(\boldsymbol{\mathcal{L}}^{(i)}, \hat{\boldsymbol{c}}^{(i)}) = \sum_{j:(1-2\hat{c}_j^{(i)})\cdot\mathcal{L}_j^{(i)}<0} \left|\mathcal{L}_j^{(i)}\right|. \tag{3}$$

A smaller correlation distance indicates the decoding estimation is more reliable. In SCL decoding of a U-UV code, the L most reliable estimations will be preserved. Such estimations will be fed back to the n inner SC decoders to estimate their subsequent input symbol LLRs. The decoding path expansion can be illustrated by the SCL decoding tree as shown in Fig. 3. In the tree, each layer corresponds to a BCH component code and nodes of the layer represent its estimations. Path expansion will be performed each time after a BCH component code is decoded. Each existing path can emancipate into multiple different paths. The number of decoding paths naturally grows exponentially as more component codes are decoded. In order to curb the also growing decoding complexity, after a component codes is decoded, only the L most reliable expanded paths will be preserved. For this, the decoding path metric can be defined based on the accumulated correlation distance (ACD) as [6]

$$\begin{aligned}\Lambda^{(i)} &= \sum_{i'=0}^{i} \lambda(\hat{\boldsymbol{c}}^{(i')}, \boldsymbol{\mathcal{L}}^{(i')}) \\ &= \sum_{i'=0}^{i} \left(\sum_{j:(1-2\hat{c}_j^{(i')})\cdot\mathcal{L}_j^{(i')}<0} \left|\mathcal{L}_j^{(i')}\right| \right).\end{aligned} \tag{4}$$

It indicates the reliability of a decoding path that reaches layer i as shown in Fig. 3. Similarly, a smaller $\Lambda^{(i)}$ indicates the corresponding decoding path is more reliable. Further let

$$\mathfrak{L}^{(i)} = \{\Lambda_l^{(i)} \mid l = 1, 2, ..., L\} \tag{5}$$

denote the set of the L smallest ACDs at layer i and they are ordered as

$$\Lambda_1^{(i)} \le \Lambda_2^{(i)} \le \cdots \le \Lambda_L^{(i)}. \tag{6}$$

Note that each $\Lambda_l^{(i)}$ in $\mathfrak{L}^{(i)}$ uniquely corresponds to a candidate path that starts from an estimation of $\hat{\boldsymbol{c}}^{(0)}$. Hence, $\mathfrak{L}^{(i)}$ also denotes the decoding output list at layer i. After decoding $\mathbb{C}^{(i)}$, the corresponding BCH codeword estimation $\hat{\boldsymbol{c}}^{(i)}$ will be fed back into the n SC decoders to further update the decoding LLR. The SC decoders and OSD decoders exchange their decoding output until all BCH component codes are decoded. At the end, the U-UV codeword estimation that corresponds to the smallest path metric $\Lambda_1^{(\gamma-1)}$ will be chosen as the decoding output $\hat{\boldsymbol{v}}$.

III. Pruning Based on Normalized Accumulated APP

This section reinterprets the decoding path metric of the U-UV codes from the perspective of APP. The normalized APP of the candidate path is further introduced as the probability of the path being the correct path. Subsequently, the list pruning scheme based on normalized accumulated APP is proposed.

A. Normalized Decoding Path APP

The SCL decoding path metric, which is defined as the ACD as in (4), can be utilized to determine the APP of the path. In decoding a U-UV code, the estimations from the OSD decoders will be fed back into the n SC decoders. Hence, the path metric $\Lambda^{(i)}$ of (4) can be seen as the sum of the path metrics of n SC decoding paths. Based on the OSD estimations, the path metric of the jth SC decoder at layer i can be calculated as [19]

$$\begin{aligned} M(\hat{\boldsymbol{c}}_j'^{(i)}) &= -\ln P(\hat{\boldsymbol{c}}_j'^{(i)} \,|\, \boldsymbol{y}_j) \\ &= \sum_{i'=0}^{i} \ln(1+e^{-(1-2\hat{c}_j^{(i')})\mathcal{L}_j^{(i')}}) \\ &\approx \sum_{i':\,(1-2\hat{c}_j^{(i')})\mathcal{L}_j^{(i')}<0} |\mathcal{L}_j^{(i')}|, \end{aligned} \tag{7}$$

where $\hat{\boldsymbol{c}}_j'^{(i)} = (\hat{c}_j^{(0)}, \hat{c}_j^{(1)}, ..., \hat{c}_j^{(i)})$ and $\boldsymbol{y}_j = (y_{j\gamma}, y_{j\gamma+1}, ..., y_{j\gamma+\gamma-1})$. Hence, the decoding path metric defined in (4) can be further written as

$$\begin{aligned} \Lambda^{(i)} &= \sum_{j=0}^{n-1} M(\hat{\boldsymbol{c}}_j'^{(i)}) \\ &= -\sum_{j=0}^{n-1} \ln P(\hat{\boldsymbol{c}}_j'^{(i)} \,|\, \boldsymbol{y}_j) \\ &= -\ln\left(\prod_{j=0}^{n-1} P(\hat{\boldsymbol{c}}_j'^{(i)} \,|\, \boldsymbol{y}_j)\right) \\ &= -\ln P(\hat{\boldsymbol{u}}^{(i)}|\boldsymbol{y}), \end{aligned} \tag{8}$$

where $\hat{\boldsymbol{u}}^{(i)} = (\hat{\boldsymbol{c}}^{(0)}, \hat{\boldsymbol{c}}^{(1)}, ..., \hat{\boldsymbol{c}}^{(i)})$ is the corresponding codeword estimation of the decoding path. Let $\hat{\boldsymbol{u}}_l^{(i)}$ denote the lth candidate path at layer i. The APP of the decoding path $\hat{\boldsymbol{u}}_l^{(i)}$ can be approximated by

$$P(\hat{\boldsymbol{u}}_l^{(i)}|\boldsymbol{y}) = e^{-\Lambda_l^{(i)}}. \tag{9}$$

By assuming that the correct decoding path is still preserved in the list $\mathfrak{L}^{(i)}$, the probability of $\hat{\boldsymbol{u}}_l^{(i)}$ being the correct path can be estimated by the normalized APP as

$$\begin{aligned} P_l^{(i)} &= \frac{P(\hat{\boldsymbol{u}}_l^{(i)} \,|\, \boldsymbol{y})}{\sum_{l'=1}^{L} P(\hat{\boldsymbol{u}}_{l'}^{(i)} \,|\, \boldsymbol{y})} \\ &= \frac{e^{-\Lambda_l^{(i)}}}{\sum_{l'=1}^{L} e^{-\Lambda_{l'}^{(i)}}}. \end{aligned} \tag{10}$$

The decoding path with a small normalized APP indicates it is not likely to be the correct path. Hence, it can be pruned. This normalized APP can also be approximately regarded as the performance loss that is incurred by pruning the path $\hat{\boldsymbol{u}}_l^{(i)}$. Based on (6), it can be seen that

$$P_1^{(i)} \geq P_2^{(i)} \geq \cdots \geq P_L^{(i)}. \tag{11}$$

B. Pruning Scheme I

Unlike the pruning scheme of [14] that only considers the most reliable candidate path metric as the pruning benchmark, the above introduced normalized decoding path APP is utilized. It considers all the decoding paths in the output list $\mathfrak{L}^{(i)}$.

Let $\mathcal{T}_l^{(i)}$ denote the normalized accumulated APP of the lth candidate path at layer i. With the output list $\mathfrak{L}^{(i)}$, $\mathcal{T}_l^{(i)}$ is defined as

$$\mathcal{T}_l^{(i)} = \sum_{l'=1}^{l} P_{l'}^{(i)}, \tag{12}$$

and $0 < \mathcal{T}_l^{(i)} \leq 1$. It further indicates the probability of the correct path being contained in the l most reliable candidates of $\mathfrak{L}^{(i)}$. When $\mathcal{T}_l^{(i)}$ is sufficiently large, the correct path would have a high probability of being contained in these l candidate paths. Consequently, the candidate paths with ordered indices greater than l can be pruned. This analysis leads to the following pruning scheme.

Pruning Scheme I: Let t denote the pruning threshold of normalized accumulated APP, and $0 \leq t < 1$. With the output list $\mathfrak{L}^{(i)}$, let $l_{\min}^{(i)}$ denote the smallest index of the candidate path that satisfies

$$l_{\min}^{(i)} = \arg\min\{\, l \mid \mathcal{T}_l^{(i)} \geq 1-t\}. \tag{13}$$

The candidate paths with the index $l > l_{\min}^{(i)}$ will be pruned from $\mathfrak{L}^{(i)}$, yielding an updated output list as

$$\tilde{\mathfrak{L}}^{(i)} = \{\Lambda_l^{(i)} \mid \Lambda_l^{(i)} \in \mathfrak{L}^{(i)}, l \leq l_{\min}^{(i)}\}, \tag{14}$$

and $1 \leq |\tilde{\mathfrak{L}}^{(i)}| \leq L$. Note that the list pruning does not need to be applied after decoding the last component code $\mathbb{C}^{(\gamma-1)}$.

Performance loss of the above pruning scheme I is further analyzed, which sheds light of the determination of the threshold t. Let $\mathcal{E}$ denote the event that the correct path is preserved in the final list but it does not have the smallest path metric. Further let $\mathcal{C}$ denote the event that the correct path is lost during the decoding, where its complementary event is denoted by $\mathcal{C}^c$. With the decoding output list size L and the

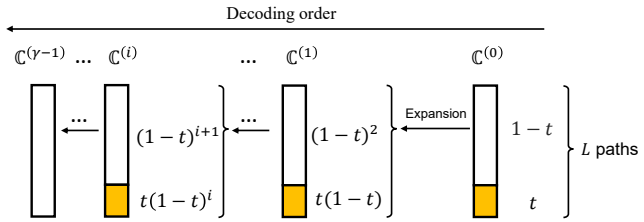

Fig. 4. List pruning based on normalized accumulated APP.

pruning threshold t, the decoding error probability $P_{\mathrm{e}}(L,t)$ can be determined by

$$P_{\mathrm{e}}(L,t) = P_{L,t}(\mathcal{E}|\mathcal{C}^c)P_{L,t}(\mathcal{C}^c) + P_{L,t}(\mathcal{C}), \tag{15}$$

where $P_{L,t}(\cdot)$ denotes the probability of a decoding event under the decoding parameters of L and t. Furthermore, let $\mathcal{C}_i$ denote the event that the correct path is not lost until decoding $\mathbb{C}^{(i)}$. With γ component codes, $P_{L,t}(\mathcal{C})$ can be further decomposed as

$$\begin{aligned} P_{L,t}(\mathcal{C}) &= \sum_{i=0}^{\gamma-1} P_{L,t}(\mathcal{C}_i) \\ &= \sum_{i=0}^{\gamma-1} [P_L(\mathcal{M}_i) + P_{L,t}(\mathcal{N}_i|\mathcal{M}_i^c)P_L(\mathcal{M}_i^c)], \end{aligned} \tag{16}$$

where $\mathcal{M}_i$ and $\mathcal{N}_i$ denote the events that the correct path is not preserved in $\mathfrak{L}^{(i)}$ and $\tilde{\mathfrak{L}}^{(i)}$, respectively, and $P_L(\cdot)$ is the probability of a decoding event which is only determined by parameter L. When $t=0$, this pruning scheme is equivalent to the initial SCL decoding without list pruning, and $P_{L,0}(\mathcal{N}_i|\mathcal{M}_i^c) = 0$. Hence, the decoding performance loss can be upper bounded by

$$\begin{aligned} P_{\mathrm{loss}} &= P_e(L,t) - P_e(L,0) \\ &= P_{L,t}(\mathcal{E}|\mathcal{C}^c)P_{L,t}(\mathcal{C}^c) - P_{L,0}(\mathcal{E}|\mathcal{C}^c)P_{L,0}(\mathcal{C}^c) \\ &\quad + \sum_{i=0}^{\gamma-1} P_{L,t}(\mathcal{N}_i|\mathcal{M}_i^c)P_L(\mathcal{M}_i^c) \\ &\le \sum_{i=0}^{\gamma-1} P_{L,t}(\mathcal{N}_i|\mathcal{M}_i^c)P_L(\mathcal{M}_i^c) \\ &\le \sum_{i=0}^{\gamma-1} P_{L,t}(\mathcal{N}_i|\mathcal{M}_i^c). \end{aligned} \tag{17}$$

Note that the first inequality comes from the fact that $P_{L,t}(\mathcal{C}^c) \le P_{L,0}(\mathcal{C}^c)$ and $P_{L,t}(\mathcal{E}|\mathcal{C}^c) \approx P_{L,0}(\mathcal{E}|\mathcal{C}^c)$. Hence, $P_{L,t}(\mathcal{E}|\mathcal{C}^c)P_{L,t}(\mathcal{C}^c) - P_{L,0}(\mathcal{E}|\mathcal{C}^c)P_{L,0}(\mathcal{C}^c) \le 0$.

Fig. 4 illustrates the list pruning process, in which the yellow bars denote the pruned decoding paths and the white bars denote the reserved paths in $\tilde{\mathfrak{L}}^{(i)}$. Assume that the correct path is contained in $\mathfrak{L}^{(0)}$, i.e., $P_L(\mathcal{M}_0^c) = 1$. At layer 0, we have $P_{L,t}(\mathcal{N}_0^c|\mathcal{M}_0^c) = 1-t$ and $P_{L,t}(\mathcal{N}_0|\mathcal{M}_0^c) = t$. When $i > 0$,

$$\begin{aligned} P_{L,t}(\mathcal{N}_i^c|\mathcal{M}_i^c) &\approx (1-t)\cdot P_{L,t}(\mathcal{N}_{i-1}^c|\mathcal{M}_{i-1}^c) \\ &= (1-t)^{i+1}, \end{aligned} \tag{18}$$

and $P_{L,t}(\mathcal{N}_i|\mathcal{M}_i^c)$ can be approximately estimated by

$$\begin{aligned} P_{L,t}(\mathcal{N}_i|\mathcal{M}_i^c) &\approx t\cdot P_{L,t}(\mathcal{N}_{i-1}^c|\mathcal{M}_{i-1}^c) \\ &= t(1-t)^i. \end{aligned} \tag{19}$$

Since the list pruning does not need to be applied at the last layer, $P_{L,t}(\mathcal{N}_{\gamma-1}|\mathcal{M}_{\gamma-1}^c) = 0$. Hence, based on (17), the total performance loss can be further estimated by

$$P_{\mathrm{loss}} \le \sum_{i=0}^{\gamma-2} t(1-t)^i. \tag{20}$$

Given a tolerable performance loss as P_{loss}, the pruning threshold t can be determined by (20).

IV. Pruning Based on Performance Loss Estimation

The normalized APP of (10) can be approximately seen as the performance loss that is caused by pruning the path [16]. The list pruning can in return be performed based on a dynamic threshold for controlling the performance loss.

A. Performance Loss Estimation

Let λ_i denote the variable of the correlation distance that is produced by the OSD of $\mathbb{C}^{(i)}$. Assuming the branch metric λ_i is Gaussian distributed with $\lambda_i \sim \mathcal{N}(\mu_{\lambda_i}, \sigma_{\lambda_i}^2)$ [20] [21], its lower bound Υ_i can be estimated through

$$\frac{1}{\sqrt{2\pi}\sigma_{\lambda_i}} \int_{\Upsilon_i}^{\infty} e^{-\frac{(\lambda_i-\mu_{\lambda_i})^2}{2\sigma_{\lambda_i}^2}} \mathrm{d}\lambda_i = 1-\theta, \tag{21}$$

where $\theta \in (0,1)$. Therefore, the correlation distance lower bound Υ_i of $\mathbb{C}^{(i)}$ can be obtained ahead of its decoding.

At layer i, we consider that only the pruned paths which can be preserved in $\mathfrak{L}^{(i+1)}$ will cause performance loss. Let $\Gamma^{(i)}$ denote the index set of the pruned paths at layer i whose path metric lower bounds at layer $i+1$ are smaller than the largest path metric in $\mathfrak{L}^{(i+1)}$, i.e.,

$$\Gamma^{(i)} = \{l \,|\, \Lambda_l^{(i)} + \Upsilon_{i+1} \le \Lambda_L^{(i+1)}, l \ge \zeta_{\min}^{(i)}\}, \tag{22}$$

where $\zeta_{\min}^{(i)}$ denotes the smallest index of the pruned paths at layer i. It is assumed that only the pruned paths with indices in $\Gamma^{(i)}$ would cause a performance loss. Hence, the performance loss that is introduced at layer i can be estimated based on the normalized APP of the pruned paths, i.e.,

$$P_{\mathrm{loss}}^{(i)} \le \sum_{l\in\Gamma^{(i)}} P_l^{(i)}. \tag{23}$$

B. Pruning Scheme II

Based on the above performance loss estimation, we can prune the decoding paths that would cause an estiamted performance loss. This list pruning scheme is presented as the follows.

Pruning Scheme II: Given a pruning threshold P_{tol} as a tolerable performance loss that is introduced by list pruning.

With $\mathfrak{L}^{(i)}$, the path index $\zeta_{\min}^{(i)}$ should satisfy

$$\sum_{l=\zeta_{\min}^{(i)}}^{L} P_l^{(i)} \leq P_{\text{tol}} - \sum_{i'=0}^{i-1} P_{\text{loss}}^{(i')}. \tag{24}$$

Then, the candidate paths with index $l \geq \zeta_{\min}^{(i)}$ will be pruned, yielding a set of the decoding paths to be pruned that is denoted by their path metrics as

$$\mathfrak{L}_p^{(i)} = \{\Lambda_l^{(i)} \,|\, l \geq \zeta_{\min}^{(i)}\}. \tag{25}$$

Compared with the list pruning scheme for polar codes of [16], this proposed scheme for U-UV codes utilizes the probability distribution function about the correlation distance of OSD output to derive the path metric lower bound. It also simplifies the update of performance loss estimation, resulting in a reduced complexity SCL decoding for U-UV codes.

V. Simulation Results

In this section, decoding performance of the two proposed list pruning schemes will be analyzed via simulation. Their decoding complexity will also be compared with the original SCL decoding. The simulated 3-level (504, 288) U-UV code is formed by the (63, 57), (63, 57), (63, 51), (63, 30), (63, 51), (63, 24), (63, 18) and (63, 0) binary primitive BCH codes. The OSD orders of the BCH codes are chosen to enable their near ML decoding performances. In particular, $\tau = 1, 1, 1, 3, 1, 3, 3$ and 0 for the BCH codes, respectively. The parameters of the Gaussian distribution in (21), i.e., μ_{λ_i} and $\sigma^2_{\lambda_i}$, are calculated according to [21]. Moreover, the correlation distance lower bounds Υ_i are obatained with $\theta = 10^{-4}$.

A. Decoding Performance

Fig. 5 shows the frame error rate (FER) performance of the (504, 288) U-UV code using different list pruning schemes. The SCL decoding functions with $L = 16$. We also provide performance of the list pruning scheme [16] in decoding the U-UV code as a comparison benchmark. Its path metric lower bound is also determined based on Υ_i as in (21). Note that the thresholds P_{tol} of the pruning scheme II and [16] are set the same as P_{loss} of the pruning scheme I as in (17). Fig. 5 shows that under a same performance loss estimation, the proposed scheme II performs similarly as the scheme of [16], and the proposed scheme I can yield a better performance than the other two with $t = 1 \times 10^{-2}$. However, their performance gap becomes smaller as the performance loss estimation reduces.

Fig. 6 further compares the FER performance deterioration of the two proposed pruning schemes. It shows that in comparison with the performance loss estimation, pruning scheme II performs close to the estimation. Moreover, it can be seen that the performance loss estimation is more accurate at the low signal noise ratio (SNR) regime. Our research has observed that at the high SNR regime, the correct decoding path has an extremely small path metric. Consequently, the normalized APPs of the other candidate paths are marginal, leading to the actual performance loss much smaller than estimation. Note

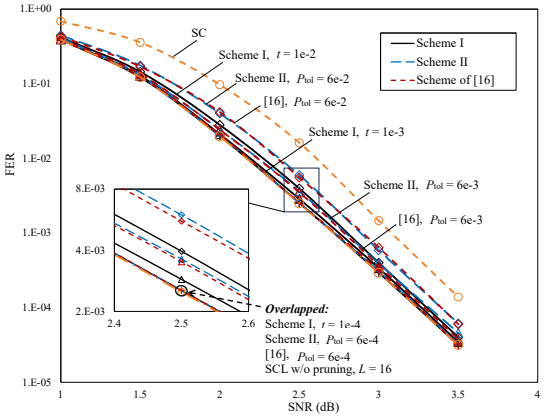

Fig. 5. SCL decoding performance of the (504, 288) U-UV code with $L = 16$.

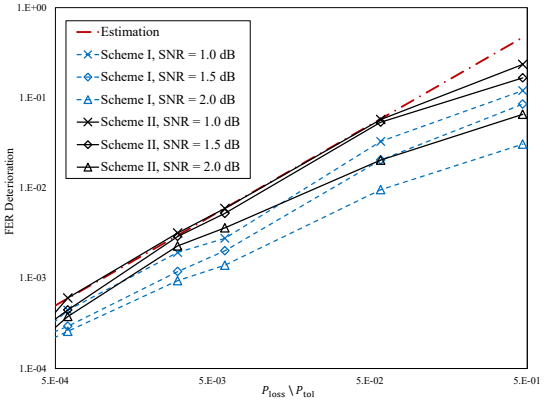

Fig. 6. FER performance deterioration comparison of the proposed list pruning schemes.

that the FER performance deterioration of the pruning scheme of [16] is almost the same as that of pruning scheme II.

B. Decoding Complexity

Table I shows the average decoding output list size for SCL decoding the (504, 288) U-UV code under different pruning schemes with $L = 16$. Fig. 7 further compares the average computational complexity of above mentioned pruning schemes, which are measured as the amount of floating point operations (FLOPs) and binary operations (BOPs) in decoding a codeword, respectively. Our simulation results show that both of the proposed pruning schemes can significantly reduce the SCL decoding complexity. At the high SNR regime, the complexity can be reduced by $80\% \sim 90\%$ over the initial SCL decoding [6]. The amount of BOPs converges to that of SC decoding. However, since the extra APP calculation is required, the amount of FLOPs cannot converge to that of SC decoding. In particular, with $t = 1 \times 10^{-2}$ and $P_{\text{tol}} = 6 \times 10^{-2}$, pruning scheme I produces a similar decoding complexity as the pruning scheme of [16] at the high SNR regime. But it can yield a slightly better performance. Furthermore, the average decoding output list size of pruning scheme II is similar to that of [16]. Both Figs. 5 and 7 show that it also yields a similar decoding performance, but with lower complexity than the scheme of [16]. This is due to pruning scheme II

TABLE I
AVERAGE LIST SIZE FOR DECODING THE (504,288) U-UV CODE UNDER DIFFERENT PRUNING SCHEMES WITH $L = 16$

SNR	Pruning Scheme I		Pruning Scheme II		Scheme of [16]	
	t = 1e-3	t = 1e-4	P_{tol} = 6e-3	P_{tol} = 6e-4	P_{tol} = 6e-3	P_{tol} = 6e-4
1.0	10.05	12.06	10.47	12.52	10.32	12.26
1.5	4.74	6.79	5.57	7.63	5.22	7.25
2.0	1.87	2.61	2.19	3.14	2.13	2.95
2.5	1.10	1.24	1.21	1.41	1.21	1.39
3.0	1.01	1.02	1.04	1.07	1.04	1.07
3.5	1.00	1.00	1.01	1.01	1.01	1.01

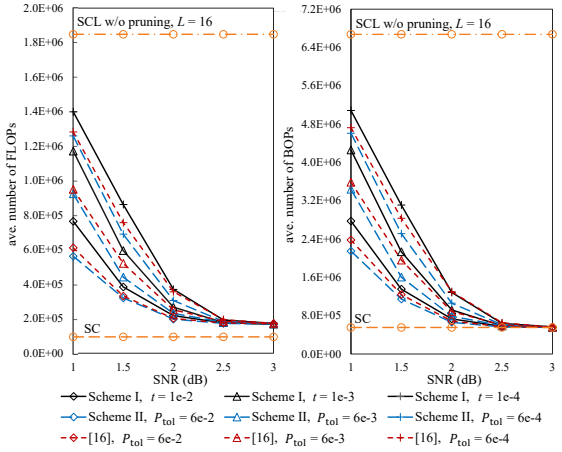

Fig. 7. Decoding complexity of (504, 288) U-UV code under different decoding schemes.

simplifies the update of performance loss estimation. It should also be pointed out that this simplification leads to a lower storage complexity than that of [16], since it needs to store less information of the pruned paths. When the performance loss estimation is sufficiently small, both of the proposed pruning schemes can reduce the decoding complexity with negligible performance loss.

VI. CONCLUSION

In this paper, two list pruning schemes have been proposed for SCL decoding of U-UV codes. With the decoding APP of the candidate paths, pruning scheme I eliminates the unpromising candidate paths based on a predetermined threshold of the normalized accumulated APPs. Pruning scheme II eliminates the redundant candidate paths based on dynamic performance loss estimation. Performance loss of the pruning schemes has been analyzed. Our simulation results have shown that the proposed pruning schemes can efficiently reduce the SCL decoding complexity of U-UV codes with negligible performance loss.

ACKNOWLEDGMENTS

This work is sponsored by the National Natural Science Foundation of China (NSFC) with project ID 62071498.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE Int. Conf. Commun.*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] R. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[3] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[4] ——, "From sequential decoding to channel polarization and back again," arXiv:1908.09594, 2019.

[5] M. C. Coskun *et al.*, "Efficient error-correcting codes in the short blocklength regime," *Phys. Commun.*, vol. 34, no. JUN., pp. 66–79, 2019.

[6] J. Cheng and L. Chen, "BCH based U-UV codes and its decoding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Melbourne, Victoria, Australia, Jul. 2021, pp. 1433–1438.

[7] M. Plotkin, "Binary codes with specified minimum distance," *IRE Trans. Inf. Theory*, vol. 6, no. 4, pp. 445–450, 1960.

[8] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, 2012.

[9] H. Mahdavifar *et al.*, "Performance limits and practical decoding of interleaved Reed-Solomon polar concatenated codes," *IEEE Trans. Commun.*, vol. 62, no. 5, pp. 1406–1417, 2014.

[10] H. Saber and I. Marsland, "Design of generalized concatenated codes based on polar codes with very short outer codes," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3103–3115, 2017.

[11] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inf. Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.

[12] A. Valembois and M. Fossorier, "Box and match techniques applied to soft-decision decoding," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 796–810, 2004.

[13] D. N. Bailon *et al.*, "Concatenated codes based on the Plotkin construction and their soft-input decoding," *IEEE Trans. Commun.*, vol. 70, no. 5, pp. 2939–2950, 2022.

[14] K. Chen, K. Niu, and J. Lin, "A reduced-complexity successive cancellation list decoding of polar codes," in *Proc. IEEE 77th Veh. Technol. Conf. (VTC Spring)*, Dresden, Germany, Jun. 2013, pp. 1–5.

[15] Z. Zhang *et al.*, "A split-reduced successive cancellation list decoder for polar codes," *IEEE J. Select. Areas Commun.*, vol. 34, no. 2, pp. 292–302, Feb, 2016.

[16] K. Chen *et al.*, "Reduce the complexity of list decoding of polar codes by tree-pruning," *IEEE Commun. Lett.*, vol. 20, no. 2, pp. 204–207, 2016.

[17] J. Chen *et al.*, "Low-complexity list successive-cancellation decoding of polar codes using list pruning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Washington, USA, Dec. 2016, pp. 1–6.

[18] M. Rowshan and E. Viterbo, "Stepped list decoding for polar codes," in *Proc. IEEE 10th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Hong Kong, Dec. 2018, pp. 1–5.

[19] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, "LLR-based successive cancellation list decoding of polar codes," *IEEE Trans. Signal Process.*, vol. 63, no. 19, pp. 5165–5179, 2015.

[20] C. Yue *et al.*, "A revisit to ordered statistics decoding: Distance distribution and decoding rules," *IEEE Trans. Inf. Theory*, vol. 67, no. 7, pp. 4288–4337, 2021.

[21] F. Wang *et al.*, "Adjustable ordered statistic decoder for short block length code towards URLLC," in *Proc. 13th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Changsha, China, Oct. 2021, pp. 1–5.