SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

May 2023, Vol. 66 152301:1–152301:12
https://doi.org/10.1007/s11432-022-3498-8

A fast parallel SC-Fano decoding algorithm for PAC codes

Yunzhi WU, Li LI* & Pingzhi FAN

Information Coding & Transmission Key Lab of Sichuan Province, Communications & Sensor Networks for Modern Transportation International Cooperatation Research Centre of China, Southwest Jiaotong University, Chengdu 610031, China

Received 13 January 2022/Revised 8 April 2022/Accepted 23 May 2022/Published online 20 April 2023

Abstract In order to effectively reduce the "decoding latency" of successive cancellation decoding aided Fano (SC-Fano) decoder employed in polarization-adjusted convolutional (PAC) codes, a fast parallel SC-Fano decoding algorithm is designed, which leverages multiple individual decoding components. Critical challenges arise from how to appropriately initialize the thresholds of different SC-Fano decoding components and how will these decoding components cooperate with each other. After appropriately overcoming these challenges, our design is capable of halving the "decoding latency" without incurring any noticeable frame error rate (FER) performance degradation, meanwhile, the overall computational complexity is normally constrained below one and a half times of that required by conventional SC-Fano decoders in simulated typical configurations.

Keywords PAC codes, parallel decoding, Fano decoding, complexity

Citation Wu Y Z, Li L, Fan P Z. A fast parallel SC-Fano decoding algorithm for PAC codes. Sci China Inf Sci, 2023, 66(5): 152301, https://doi.org/10.1007/s11432-022-3498-8

1 Introduction

Polar codes proposed by Arikan [1] can provably achieve the capacity of binary-input memoryless channels with the aid of successive cancellation (SC) decoding. Then, in order to overcome the "error propagation" problem of SC decoding, the well-known successive cancellation list (SCL) decoding was proposed in [2]. The performance gain of SCL with respect to SC is obtained at the cost of requiring a higher computational complexity and a larger buffer size. However, regretfully, SCL decoding cannot compensate for the poor Hamming distance property of polar codes and still suffers performance loss. This deficiency was overcome by concatenating cyclic redundancy check (CRC) codes or more generalized parity check (PC) codes with polar codes. It results in CRC-aided SCL decoding [3] or PC-aided SCL decoding [4]. The latter has been chosen by 5G NR to protect the control signaling [5].

The evolution of polar code encoding methods paralleled the development of the above-mentioned decoding algorithms. To be concrete, in order to improve the Hamming distance property, the Reed-Muller (RM)-polar code was proposed in [6]. Later, it was proved by the same authors in [7] that the number of polar codewords, which incurs the minimum Hamming distance could be reduced by an upper-triangular matrix based pre-transform of the polar code generator. Then, a polarization-adjusted convolutional (PAC) code was proposed in [8], which employs an RM rate profile and concatenates an outer convolutional code with an inner polar code. In [8], sequential decoding was advocated; e.g., the successive cancellation decoding aided Fano (SC-Fano) decoder that first proposed in [9] was employed again for PAC codes. Benefiting from the improved Hamming distance profile and the advanced SC-Fano decoding, the frame error rate (FER) of PAC codes almost approaches the dispersion approximation reported in [10]; e.g., the gap is less than 0.2 dB at the target FER of 10^{-4}. However, while considering SC-Fano decoding aided PAC codes, their computational complexity required in the low signal to noise ratio (SNR) region is very high and significantly exceeds that required in moderate or high SNR regions. More

* Corresponding author (email: ll5e08@home.swjtu.edu.cn)

seriously, this low-SNR aggravated computational complexity strongly fluctuates and may dramatically overflow its statistic average. In this case, an unpredictable decoding latency takes place, which prevents PAC codes from being used in latency sensitive applications, such as some specific Internet of Things (IoT) networks and the next generation of Wi-Fi networks. Hence, how to effectively mitigate the computational complexity of PAC codes, especially how to reduce their decoding latency [11][1)] becomes an important issue.

In order to overcome the above-stated deficiency of SC-Fano decoding aided PAC codes, some new decoding algorithms were proposed in the past two years. For example, Moradi [12] attempted to mitigate this high computational complexity by only adjusting the metric calculation function of the Fano decoder. Later, Rowshan et al. [13] discussed the feasibility of replacing the previous sequential decoding by its counterpart of list decoding. The computational complexity was effectively reduced, but on the other hand, the performance degradation was also considerable. Then, Yao and Rowshan [14, 15] further developed the list decoding algorithm for PAC codes. From our perspective, their developments could be regarded as invoking local parallel decoding architectures into the PAC decoder for reducing the decoding latency. However, their parallel decoders are only involved in the forward movements along a single code tree, hence their algorithms are still far away from a fully paralleled architecture and do not effectively constrain the computational complexity although they can decrease the decoding latency.

In this paper, our major contributions are as follows.

(1) It is revealed that the tradeoff between complexity and error-correction performance of SC-Fano decoding is not only related to path metric bias and threshold spacing but also impacted by the initialization and upper bound of the threshold. Based on this fact, a parallel SC-Fano decoding algorithm is proposed to boost the decoding speed.

(2) In our parallel SC-Fano decoding algorithm, the initial threshold list for different decoding components could be determined by Monte Carlo based method, and then a more convenient density evolution based alternative is proposed. The interaction mechanism among decoding components is also investigated.

(3) It is shown that, after parallel processing, the decoding latency is almost halved without noticeable degradation of FER performance. Benefiting from the early stop mechanism in our parallel processing, its overall computational complexity will not linearly increase with the number of individual decoding components; actually, it can be constrained below one and a half times of the complexity required by conventional SC-Fano deconders.

The rest of this paper is organized as follows. Section 2 briefly introduces PAC codes and conventional Fano decoder. In Section 3, the schematic and major steps of our parallel SC-Fano decoding are provided. Section 4 discusses two methods for calculating the initial threshold list of parallel SC-Fano decoder. Simulation results are provided and analyzed in Section 5. Finally, we conclude in Section 6.

In this paper, the bold lowercase letter denotes vector. $\boldsymbol{x}_i^j$ is used to denote subvector $[x_i, \ldots, x_j]$, where x_i denotes the ith element of $\boldsymbol{x}$. The bold capital letter denotes a matrix. An element in the ith row and the jth column of a matrix $\boldsymbol{X}$ is defined as $X[i, j]$. $\boldsymbol{x}_{\mathcal{A}}$ denotes a sub-vector of $\boldsymbol{x}$, whose elements are indicated by the index set $\mathcal{A}$.

2 Preliminaries

2.1 General flowchart of PAC codes

A PAC code can be specified by four parameters of $(N, K, \mathcal{A}, \boldsymbol{g})$, where $N = 2^n$ and K represent the codeword length and the number of carried information bits, respectively. The set $\mathcal{A} \subseteq \{0, 1, \ldots, N-1\}$ consists of indices of every information bit, and hence $\|\mathcal{A}\| = K$. It is chosen according to RM rate profile herein, which is also one of the recommended methods in [8]. $\boldsymbol{g} = [g_0, \ldots, g_m]$ is the generator polynomial of a rate one convolutional code. Figure 1 illustrates the general PAC encoding and decoding flowchart. First, the input data bit sequence $\boldsymbol{d} = [d_0, \ldots, d_{K-1}] \in \mathbb{F}_2^K$ is expanded to $\boldsymbol{v} = [v_0, \ldots, v_{N-1}] \in \mathbb{F}_2^N$ by assigning $\boldsymbol{d}$ to $\boldsymbol{v}_{\mathcal{A}}$ and padding zero bits to $\boldsymbol{v}_{\mathcal{A}^c}$, where $\boldsymbol{v}_{\mathcal{A}} = \{v_i \mid i \in \mathcal{A}\}$ and $\mathcal{A}^c$ denotes the

1) For a single sequential decoder, its decoding latency is in accordance with its computational complexity. However, while employing a parallel decoding algorithm, the decoding latency could be significantly reduced by invoking multiple independent processors, in contrast, the overall computational complexity is not improved or even becomes worse. A well-known paradigm is the paralleled LDPC decoding [11].

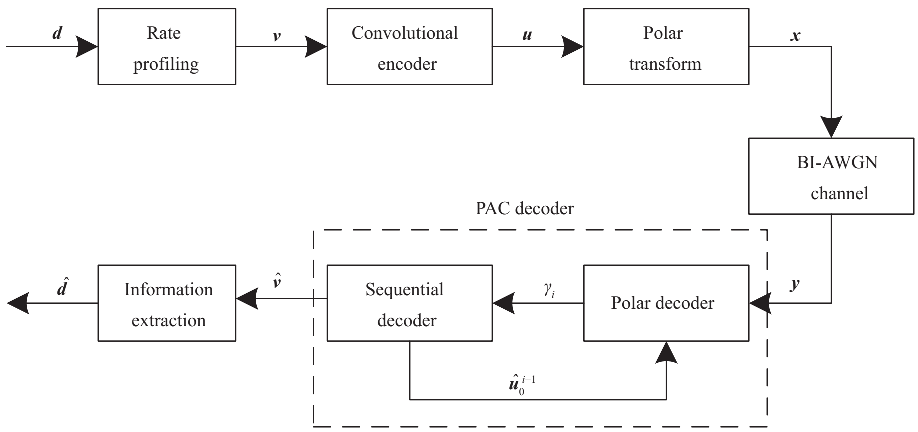

Figure 1 Flowchart of PAC encoding and decoding procedure.

complementary of $\mathcal{A}$. Then, the convolutional encoding is formulated by

$$u_i = \sum_{j=0}^{m} g_j v_{i-j}, \tag{1}$$

where $v_{i-j} = 0$ for $i - j < 0$. After polar encoding, the PAC codeword $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{P}^n$ is obtained, where $\boldsymbol{P}^n$ is an n-th Kronecker power of $\boldsymbol{P} = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$. The PAC codeword $\boldsymbol{x}$ is transmitted through a binary-input additive white Gaussian noise (BI-AWGN) channel and received as the signal $\boldsymbol{y}$. Normally, sequential decoding is employed to decode $\boldsymbol{y}$ as advocated in [8,12]. Finally, the estimation of data bits $\hat{\boldsymbol{d}}$ is extracted from the output of sequential decoder according to $\hat{\boldsymbol{d}} = \hat{\boldsymbol{v}}_{\mathcal{A}}$.

2.2 PAC decoding

PAC decoding could be realized by combining Fano decoding with SC decoding, i.e., the SC-Fano decoding mentioned before. Particularly, we may refer to the PAC decoding algorithm reported in [12, 16] as the conventional PAC decoding algorithm.

Fano decoding [17] is a score function based depth-first tree search algorithm. Its decoding path will move forward from current tree node to one of the children nodes whose path metric (PM) Γ is the highest and exceeds a running threshold T. Otherwise, the decoding path tries to move backward to one of the parent nodes that satisfies $\Gamma \geqslant T$. In a forward move, if $\Gamma \geqslant T + \Delta$ is also satisfied, the dynamic running threshold will further increase to $T + \Delta$. In a backward attempt, if no parent node could meet the requirement of $\Gamma \geqslant T$, the running threshold has to degrade to $T - \Delta$ and then the decoding path tries the forward move again by comparing PM with the updated threshold of $T - \Delta$.

In conventional PAC decoding, the partial PM $\Gamma[j]$ accumulates the branch metric (BM) of every node pertaining to current decoding path that spans from root node to the jth node, i.e., $\Gamma[j] = \sum_{i=0}^{j} \gamma(\hat{u}_i; \boldsymbol{y}, \hat{\boldsymbol{u}}_0^{i-1})$. Hence, PM Γ of a complete decoding path is obtained by assigning $N-1$ to j. Then, the BM of ith node is given by [12]

$$\gamma(\hat{u}_i; \boldsymbol{y}, \hat{\boldsymbol{u}}_0^{i-1}) = \begin{cases} 1 - \log_2(1 + \mathrm{e}^{-L(u_i)}) - b_i, & \hat{u}_i = 0, \\ 1 - \log_2(1 + \mathrm{e}^{L(u_i)}) - b_i, & \hat{u}_i = 1, \end{cases} \tag{2}$$

where $L(u_i)$ is the log-likelihood ratio (LLR) of ith input bit to polar transform model. It could be calculated and fed back by the SC decoder.

In PAC decoding, diverse tradeoffs between decoding complexity and error-correction performance could be achieved by adjusting its threshold spacing parameter Δ. Assign $\Delta = 2$ was recommended in [12], which almost minimizes FER and still remains a relatively low complexity. Furthermore, the complexity and FER performance of conventional PAC decoding are also sensitive to the value of bias b_i that is involved in (2). Assign the cutoff rate of ith polarized bit channel, namely $E_0(1, W_N^{(i)})$ to b_i was recommended in [12].

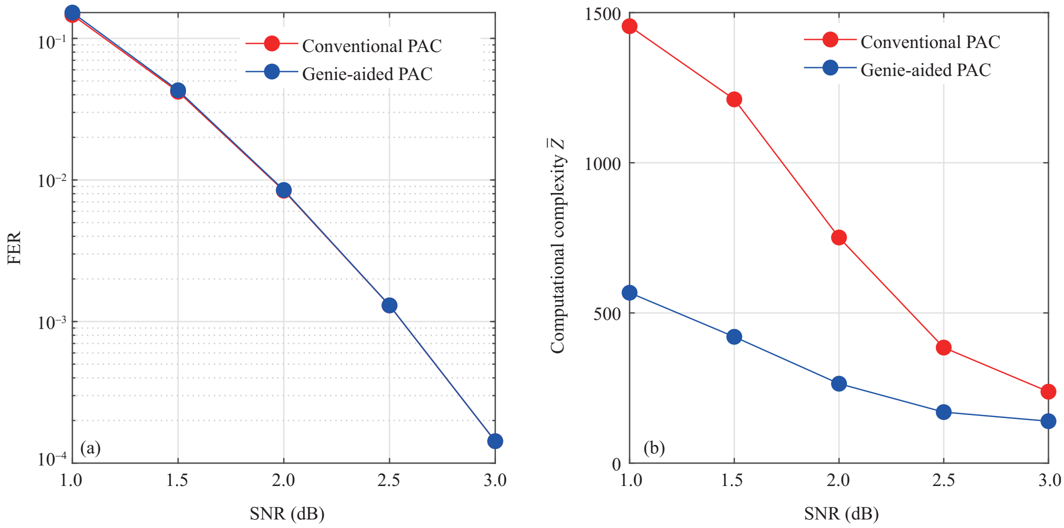

Figure 2 FER performance and computational complexity of PAC decoding, $N = 128$, $K = 64$.

3 Parallel PAC decoding

3.1 Motivation of parallel decoding

The decoding path selected by conventional PAC decoding is referred to as a near-optimal decoding path, since its performance has got extremely close to the dispersion approximation. During PAC decoding, the running threshold T will gradually converge from its initial value $\mathring{T}$ to the PM of near-optimal decoding path, namely $\dot{\Gamma}$. In order to guarantee a satisfactory error-correction performance, normally, $\mathring{T}$ is set to a relatively large value. However, a too large discrepancy between $\mathring{T}$ and $\dot{\Gamma}$ may incur too many unnecessary backward moves and updates of running threshold in the Fano decoding procedure. It aggravates the high decoding latency problem of Fano decoding.

In order to verify the impact of $\mathring{T}$, Figure 2 is generated. A number of M expanded data sequence $\boldsymbol{v}$ are transmitted. Every received signal $\boldsymbol{y}$ will be decoded twice. Firstly, $\boldsymbol{y}$ is decoded by a conventional PAC decoder, whose threshold is initialized to $\mathring{T} = 0$. Once a complete decoding path is determined, we record the associate value of $\dot{\Gamma}$ and its components $\dot{\Gamma}[j]$, $j = 0, 1, \ldots, N-1$. Then, we decode $\boldsymbol{y}$ by a conventional PAC decoder again. In contrast, this time, its initial threshold is specified as $\mathring{T} = \lfloor \frac{\Gamma_{\min}}{\Delta} \rfloor \Delta$, $\Gamma_{\min} = \min_{0 \leqslant j \leqslant N-1} \dot{\Gamma}[j]$ and its running threshold is upper bounded by $\mathring{T}$. Since the a priori knowledge of $\dot{\Gamma}[j], 0 \leqslant j \leqslant N-1$ is exploited by the second PAC decoder, it is referred to as a genie-aided PAC decoder.

Furthermore, in line with [16], the total number of nodes visited by the forward moves until the conventional PAC decoder determines a complete decoding path is denoted as a variable Z. The average of Z over M simulation trials, namely $\bar{Z}$ appropriately measures the computational complexity of conventional PAC decoder. The computational complexity of genie-aided PAC decoding could be measured in the same way, which is denoted by $\bar{Z}_g$. Particularly, in genie-aided PAC decoding, the initial threshold for some received signals may be set to the same value of $\mathring{t}$. The complexity required by genie-aided PAC decoding to decode these particular signals is denoted by $\bar{Z}_{g|\mathring{t}}$.

It is evidenced in Figure 2 that an appropriate initial threshold $\mathring{T}$, which also acts as the upper bound of running threshold could significantly reduce the computational complexity of conventional PAC decoding without any noticeable degradation of FER performance. The essential reason is that in genie-aided PAC decoding, its initial threshold $\mathring{T}$ matches the practical channel condition well. In general, a better channel quality should be associated with a higher $\mathring{T}$ and vice versa. However, genie-aided PAC decoding is not a practical solution, because we cannot obtain its preferred $\Gamma_{\min}$ before executing the conventional PAC decoding once. A feasible alternative is that we simultaneously employ a number of different initial thresholds. Hence, as long as the number is sufficiently large and their values are appropriately distributed, at least one or even some of them could match the practical channel condition and will result in a fast decoding process. This consideration directly inspires us to a parallel decoding design for PAC codes. As a benefit, the high decoding latency of conventional PAC decoder could be effectively mitigated, although

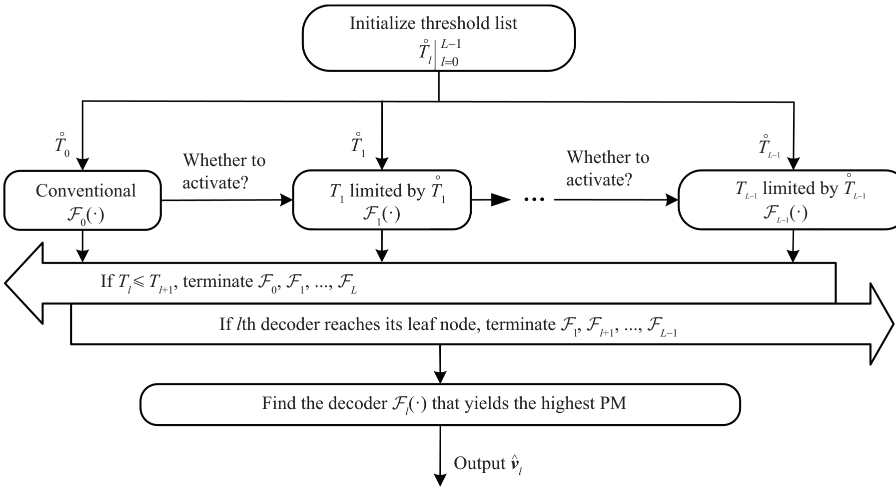

Figure 3 Schematic of parallel PAC decoding.

the overall computational complexity will somewhat increase.

3.2 Realization of parallel decoding

The proposed parallel PAC decoder consists of L individual PAC decoders. Each of them is defined as a function $\mathcal{F}_l(\cdot), 0 \leqslant l \leqslant L-1$. In general, the partial PMs $\Gamma_l[j]\big|_{j=0}^{i-1}$, the temporary estimation of expanded data $\hat{\boldsymbol{v}}_l[j]\big|_{j=0}^{i-1}$, the running threshold T_l, as well as the received signal sequence $\boldsymbol{y}$ constitute the inputs at the ith node of $\mathcal{F}_l(\cdot)$. During each decoding step, $\mathcal{F}_l(\cdot)$ generates $\Gamma_l[j]\big|_{j=0}^{i}$ and $\hat{\boldsymbol{v}}_l[j]\big|_{j=0}^{i}$ in the case of making a forward move, and further records its current arrival node i_l, as well as updates T_l. In the case of making a backward move, $\mathcal{F}_l(\cdot)$ additionally verifies whether it is the first time to make such a movement. If yes, a specific label b_l is set to be true. Pay attention to that $\mathcal{F}_0(\cdot)$ is just a conventional PAC decoder, which is the same as that introduced in Section 2, but the running thresholds T_l of other PAC decoders $\mathcal{F}_{l\neq 0}(\cdot)$ will be upper bounded by their initial values $\mathring{T}_l$. For convenience, the matrix of partial PMs and that of estimated data sequences generated by every individual PAC decoder are aggregated in $\boldsymbol{\Gamma}$ and $\boldsymbol{V}$, respectively, where $\boldsymbol{\Gamma}[l,j] = \Gamma_l[j]$ and $\hat{\boldsymbol{V}}[l,j] = \hat{\boldsymbol{v}}_l[j]$. The state a_l is used to indicate $\mathcal{F}_l(\cdot)$ is activated or terminated.

The parallel PAC decoding schematic is depicted in Figure 3, which consists of these main steps.

(1) Initialize the thresholds of all the individual PAC decoders. The set $\mathring{T}_l\big|_{l=0}^{L-1}$ actually dominates the balance between decoding latency and FER performance. A superior solution is provided in Subsection 3.2. Without loss of generality, $\mathring{T}_l\big|_{l=0}^{L-1}$ are arranged in a descending order, i.e., $\mathring{T}_l \geqslant \mathring{T}_{l+1}, \forall l$.

(2) Activate one more PAC decoder. If $\mathcal{F}_l(\cdot)$ makes a backward move in the first time, it implies that a mismatch between the running threshold[2)] and current channel quality perhaps occurs. Hence it is better to further activate the next PAC decoder $\mathcal{F}_{l+1}(\cdot)$, whose running threshold is initialized to a lower level. In order to accelerate the decoding in $\mathcal{F}_{l+1}(\cdot)$, the temporary decoding states of $\mathcal{F}_{l+1}(\cdot)$ are synchronized to that of $\mathcal{F}_l(\cdot)$. These operations are summarized as lines 9–13 in Algorithm 1.

(3) Early termination. As stated early, a PAC decoder configured with a higher initial threshold $\mathring{T}_l$ is employed to match a better channel condition and vice versa. Hence, if the running threshold T_l of $\mathcal{F}_l(\cdot)$ has degraded below $\mathring{T}_{l+1}$, it implies $\mathring{T}_l\big|_{l=0}^{l}$ employed by the preceding decoders $\mathcal{F}_l(\cdot)\big|_{l=0}^{l}$ are prone to overestimate the channel quality and should be terminated for saving computational complexity. This operation is called backward termination and shown on lines 14–16 in Algorithm 1. On the other hand, if $\mathcal{F}_l(\cdot)$ has reached a leaf node on its code tree; i.e., it has obtained a complete estimation $\hat{\boldsymbol{v}}_l$, the later decoders $\mathcal{F}_l(\cdot)\big|_{l+1}^{L-1}$ should also be terminated for saving computational complexity, because their estimations are more error-prone than $\hat{\boldsymbol{v}}_l$. This operation is called as forward termination and shown on lines 17–19 in Algorithm 1.

(4) Fusion of decisions. In the proposed parallel PAC decoding, it is possible that more than one individual PAC decoders successfully generate their complete estimations. In this case, $\hat{\boldsymbol{v}}_l$ that has the highest PM is outputted as the final decision.

2) It still equals $\mathring{T}_l$ at this moment.

Algorithm 1 Parallel PAC decoding

Require: $\boldsymbol{y}$, L, N, $\mathring{T}_l \big|_{l=0}^{L-1}$;
Ensure: $\hat{\boldsymbol{v}}$;
Initialize all the variables $\{a_l, i_l, b_l, T_l\} \big|_{l=0}^{L-1}$ to zero;
$\boldsymbol{\Gamma} \leftarrow \text{zeros}(L, N)$, $\boldsymbol{V} \leftarrow \text{zeros}(L, N)$;
$a_0 \leftarrow 1$, $T_0 \leftarrow \mathring{T}_0$; //Activate conventional PAC decoder.
while $\sum_{l=0}^{L-1} a_l \neq 0$ **do**
  for $l = 0$; $l < L$; $l++$ **do**
    if $a_l = 1$ **then**
      $[\boldsymbol{\Gamma}[l,:], \boldsymbol{V}[l,:], i_l, T_l, b_l] \leftarrow \mathcal{F}_l(\cdot)$;
    end if
    if $b_l = \text{true}$ & $l \leqslant L-2$ **then**
      $a_{l+1} \leftarrow 1, T_{l+1} \leftarrow \mathring{T}_{l+1}$; //Activate $(l+1)$th PAC.
      $\big[\boldsymbol{\Gamma}[l+1,:], \boldsymbol{V}[l+1,:], i_{l+1}\big] = \big[\boldsymbol{\Gamma}[l,:], \boldsymbol{V}[l,:], i_l\big]$; //Accelerate $(l+1)$st PAC decoder.
      $b_l \leftarrow \text{false}$;
    end if
    if $T_l \leqslant \mathring{T}_{l+1}$ **then**
      $\big[a_0, a_1, \ldots, a_l\big] \leftarrow \boldsymbol{0}$; //Backward termination.
    end if
    if $i_l = N-1$ **then**
      $\big[a_l, a_{l+1}, \ldots, a_{L-1}\big] \leftarrow \boldsymbol{0}$; //Forward termination.
    end if
  end for
end while
$l \leftarrow \arg\max_{0 \leqslant l \leqslant L-1} \boldsymbol{\Gamma}[l, N-1]$; //Best solution in L PAC decoders.
return $\hat{\boldsymbol{v}} \leftarrow \boldsymbol{V}[l,:]$.

4 Threshold list determination for parallel PAC decoding

4.1 Monte Carlo simulation based method

Since $\mathcal{F}_0(\cdot)$ is a conventional PAC decoder, we have $\mathring{T}_0 = 0$. Bear in mind that $\mathring{T}_l \big|_{l=0}^{L-1}$ has a descending order; hence, any initial threshold is constraint to the range of $[\mathring{T}_{L-1}, 0]$. According to the second main step of parallel PAC decoding, $\mathring{T}_{L-1}$ should satisfy that no significant computational complexity deduction of $\mathcal{F}_{L-1}(\cdot)$ could be observed after replacing $\mathring{T}_{L-1}$ by a smaller value. On the other hand, it is better to uniform the computational complexity deduction effect of each initial threshold $\mathring{T}_l$. For any given PAC code $(N, K, \mathcal{A}, \boldsymbol{g})$ and the number of individual decodes L, Monte Carlo simulation based method could tackle the above mentioned problems. In more details, an incremental complexity deduction function is defined as

$$\mathcal{D}(t) = \sum_{\mathring{T}=t}^{0} (\bar{Z} - \bar{Z}_{g|\mathring{T}}), \tag{3}$$

where the increasing bias of $\mathring{T}$ is Δ. Assume $\mathcal{D}(t)$ approaches saturation in the range of $t \leqslant t^*$. Accordingly, let d^* denote the value of $\mathcal{D}(t^*)$. Hence, the Monte Carlo simulation based initial thresholds are given by

$$\mathring{T}_l = \text{round}\left[\frac{\mathcal{D}^{-1}(l \times \frac{d^*}{L})}{\Delta}\right] \times \Delta, \quad 0 \leqslant l \leqslant L-1, \tag{4}$$

where round$(\cdot)$ calculates the closest integer. Since a nonlinear operation of round$(\cdot)$ is employed in (4), it is possible that a pair of adjacent l is projected onto the same level of $\mathring{T}_l$. Once this particular case happens, we can reduce the number of L by one.

An example of the proposed Monte Carlo simulation based threshold initialization is illustrated in Figure 4, where $L = 6$ and SNR $= 1.5$ dB are assumed. It is shown in Figure 4 that no considerable complexity deduction could be obtained after t becomes less than -24. The value of $\mathcal{D}(-24)$ that equals 790 already gets close to the maximum incremental complexity deduction, hence it could play the role of d^*. Then, the operation $\mathcal{D}^{-1}(l \times \frac{d^*}{L}), 0 \leqslant l \leqslant L-1$ in (4) is equivalent to quantize the overall incremental complexity deduction by $L = 6$ levels and project them onto the threshold space. These operations are denoted by red points and dashed red arrows in Figure 4. Finally, the initial threshold list obtained in this example is $\mathring{T}_l|_{l=0}^{4} = \{0, -8, -10, -12, -14\}$, where L has been autonomously deducted by 1, since we encounter "round$[\frac{\mathcal{D}^{-1}(4 \times \frac{d^*}{L})}{\Delta}]$ = round$[\frac{\mathcal{D}^{-1}(5 \times \frac{d^*}{L})}{\Delta}]$" in this example.

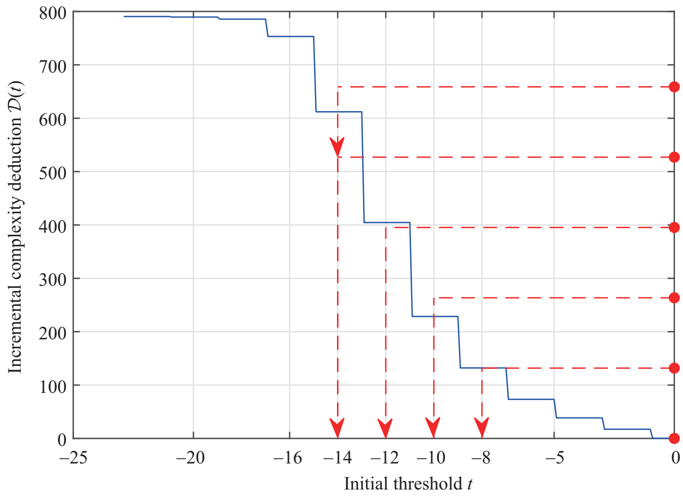

Figure 4 Incremental complexity deduction, when $L = 6$, SNR = 1.5 dB.

4.2 Density evolution based method

Monte Carlo simulation based threshold initialization improves the performance of parallel PAC decoder as demonstrated later in Section 5. However, for every specific configuration of $(N, K, \mathcal{A}, \mathbf{g})$, L and SNR, its incremental complexity deduction function in (3) has to be tested again and the projection of quantized complexity levels onto threshold space has to be implemented manually. These inconveniences prevent its applicability. Hence, a density evolution (DE) based threshold initialization is further proposed herein.

As shown in Figure 1, after the polar transform, the PAC codeword is given by $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{P}_n$. Assume the LLR value of the ith input bit u_i that is generated by polar decoder obeys to a normal distribution of mean μ_i and variance σ_i^2, i.e., $L(u_i) \sim \mathcal{N}(\mu_i, \sigma_i^2)$. Then, on the condition that an all-zero data sequence $\boldsymbol{u}$ is transmitted, the LLR mean μ_i and variance σ_i^2 can be evaluated according to the classical Gaussian approximation (GA) algorithm [18]. Accordingly, while only employing the polar decoder in Figure 1, the error probability of u_i could be evaluated by

$$P_e(u_i) = P(L(u_i) < 0) = 1 - Q\left(\frac{-\mu_i}{\sigma_i}\right) = Q\left(\frac{\mu_i}{\sigma_i}\right). \tag{5}$$

Based on (5), the coordinates of every information bit in $\mathcal{A}$ could be arranged in a descending order of their error probabilities; i.e., the index of the information bit that incurs the largest $P_e(u_i)$ will become the first element in the reordered set $\mathcal{S}$, and so force. The event that a number of J most error-prone information bits are simultaneously decoded incorrectly by the SC polar decoder has a probability of $\Pi_{j=0}^{J-1} P_e(u_i)$, where $i = \mathcal{S}[j]$. Let ϵ denote a target burst error probability. If $\Pi_{j=0}^{J} P_e(u_i) \geqslant \epsilon$ holds, it implies all the information bits whose indices fall under the subset of $\mathcal{S}[0:J-1]$ shall be paid particular attentions in the SC-Fano decoding, because their branch metrics fed back from SC polar decoder tend to degrade the cumulative path metric of SC-Fano decoder and finally result in a very small complete path metric $\Gamma[N-1]$. Hence, we use $\mathcal{B}$ to represent this particular subset of $\mathcal{S}[0:J-1]$. Then, we claim that the low path metric value of $\Gamma[N-1]$ associated with $\mathcal{B}$[3], namely Γ_{low} could play a similar role to t^* that used in Monte Carlo based threshold initialization. Hence, similarly to (4), the density evolution based initial thresholds could be given by

$$\mathring{T}_l = \text{round}\left[\frac{l}{L} \times \frac{\Gamma_{\text{low}}}{\Delta}\right] \times \Delta, \qquad 0 \leqslant l \leqslant L-1. \tag{6}$$

In the rest of this subsection, we will discuss the evaluation of Γ_{low}. The branch metric of ith node $\gamma(\hat{u}_i; \boldsymbol{y}, \hat{\boldsymbol{u}}_0^{i-1})$ has been formulated in (2). It is rewritten as γ_i herein for simplicity. As mentioned before, in Gaussian Approximation of density evolution, it is assumed that $L(u_i) \sim \mathcal{N}(\mu_i, \sigma_i^2)$. Then, it is clearly

3) It means $\Gamma[N-1]$ is evaluated based on the assumption that all the information bits in $\mathcal{B}$ are incorrectly decoded in SC-Fano decoder, while the other information bits are correctly decoded.

in (2) that γ_i is a function of $L(u_i)$. Hence, if we regard γ_i as a random variable, its probability density function (PDF) $p(\gamma_i)$ could be derived from the PDF of $L(u_i)$, which is given by

$$p(\gamma_i) = \begin{cases} \frac{2^{1-b_i-\gamma_i}\ln 2}{2^{1-b_i-\gamma_i}-1}\frac{1}{\sqrt{2\pi}\sigma_i}\mathrm{e}^{-\frac{(\ln(2^{1-b_i-\gamma_i}-1)+\mu_i)^2}{2\sigma_i^2}}, & \hat{u}_i = 0, \\ \frac{2^{1-b_i-\gamma_i}\ln 2}{2^{1-b_i-\gamma_i}-1}\frac{1}{\sqrt{2\pi}\sigma_i}\mathrm{e}^{-\frac{(\ln(2^{1-b_i-\gamma_i}-1)-\mu_i)^2}{2\sigma_i^2}}, & \hat{u}_i = 1. \end{cases} \tag{7}$$

According to (2) again, the legitimate range of γ_i should be $\gamma_i \in (-\infty, 1-b_i)$.

If u_i is correctly decoded in SC-Fano decoder, i.e., $\hat{u}_i = 0$, the expectation of γ_i is given by

$$\begin{aligned} E(\gamma_i|\hat{u}_i = 0) &= \int_{-\infty}^{1-b_i} \gamma_i p(\gamma_i)\mathrm{d}\gamma_i \\ &= 1 - b_i - \int_{-\infty}^{+\infty} \frac{\log_2(\mathrm{e}^{2\sqrt{\mu_i}t-\mu_i}+1)}{\sqrt{\pi}}\mathrm{e}^{-t^2}\mathrm{d}t. \end{aligned} \tag{8}$$

Otherwise, the expectation of γ_i is given by

$$E(\gamma_i|\hat{u}_i = 1) = 1 - b_i - \int_{-\infty}^{+\infty} \frac{\log_2(\mathrm{e}^{2\sqrt{\mu_i}t+\mu_i}+1)}{\sqrt{\pi}}\mathrm{e}^{-t^2}\mathrm{d}t. \tag{9}$$

According to our definition of Γ_{low}, it could be approximated as

$$\Gamma_{\text{low}} = \sum_{i=0, i\notin\mathcal{B}}^{N-1} E(\gamma_i|\hat{u}_i = 0) + \sum_{i\in\mathcal{B}} E(\gamma_i|\hat{u}_i = 1). \tag{10}$$

The above stated density evolution based threshold initialization is summarized in Algorithm 2.

Algorithm 2 DE based threshold initialization

Require: SNR, N, $\mathcal{A}$, L, Δ, ϵ;
Ensure: $\mathring{T}_l\big|_{l=0}^{L-1}$;
$\{P_e(u_i), \mu_i | 0 \leqslant i \leqslant N-1\} \leftarrow GA(\text{SNR}, N)$; //Use Gaussian approximation.
$[\sim, \mathcal{S}] \leftarrow$sort $[\{P_e(u_i) | i \in \mathcal{A}\}$, "descend"];
$j = 0$, $P_e = 1$, $\mathcal{B} = \emptyset$;
while $P_e > \epsilon$ **do**
  $i \leftarrow \mathcal{S}[j]$, $P_e = P_e \times P_e(u_i)$, $\mathcal{B}[j] = \mathcal{S}[j]$;
  $j = j + 1$;
end while
for $i = 0$; $i <= N-1$; $i++$ **do**
  $E(\gamma_i|\hat{u}_i = 0) = 1 - b_i - \int_{-\infty}^{+\infty} \frac{\log_2[\exp(2\sqrt{\mu_i}t-\mu_i)+1]}{\sqrt{\pi}}\mathrm{e}^{-t^2}\mathrm{d}t$;
  $E(\gamma_i|\hat{u}_i = 1) = 1 - b_i - \int_{-\infty}^{+\infty} \frac{\log_2[\exp(2\sqrt{\mu_i}t+\mu_i)+1]}{\sqrt{\pi}}\mathrm{e}^{-t^2}\mathrm{d}t$;
end for
$\Gamma_{\text{low}} = \sum_{i=0,i\notin\mathcal{B}}^{N-1} E(\gamma_i|\hat{u}_i = 0) + \sum_{i\in\mathcal{B}} E(\gamma_i|\hat{u}_i = 1)$;
for $l = 0$; $l <= L-1$; $l++$ **do**
  $\mathring{T}_l = \text{round}(\frac{l}{L} \times \frac{\Gamma_{\text{low}}}{\Delta}) \times \Delta$;
end for
return $\mathring{T}_l\big|_{l=0}^{L-1}$.

4.3 Monte Carlo (MC) vs. density evolution (DE) threshold initialization

The initial threshold lists obtained by Monte Carlo simulation based method and that by density evolution based method are compared in Table 1, where $L = 6$, $N = 128$, $K = 64$ are considered, while SNR grows from 1 to 3 dB.

5 Simulation results and analysis

5.1 Numerical calculation of decoding latency

As stated at the end of Section 1, the proposed parallel SC-Fano decoding algorithm aims at boosting the decoding speed (or alternatively, the decoding throughput) of PAC codes. Correspondingly, a specific

Table 1 Initial threshold list selected by different algorithms

SNR (dB)	Initial threshold list	
	Monte Carlo	Density evolution
1	$0, -4, -6, -8, -10$	$0, -2, -4, -6, -8, -10$
1.5	$0, -8, -10, -12, -14$	$0, -4, -8, -10, -14, -18$
2	$0, -10, -14, -16, -18, -20$	$0, -4, -8, -12, -16, -20$
2.5	$0, -8, -14, -18, -22, -26$	$0, -4, -8, -12, -16, -20$
3	$0, -4, -8, -14, -18, -26$	$0, -6, -12, -20, -26, -32$

metric that is termed as decoding latency is invoked herein for quantifying this improvement. In more details, the decoding latency of lth PAC decoder $\mathcal{F}_l(\cdot)$ illustrated in Figure 3 is defined as

$$\phi_l = \begin{cases} \psi_l + Z_l, & a_l = 1, \\ 0, & a_l = 0, \end{cases} \tag{11}$$

where $a_l = 1$ indicates that $\mathcal{F}_l(\cdot)$ has been activated during parallel PAC decoding; hence its decoding latency should be taken into account. Then, ψ_l evaluates the waiting time of $\mathcal{F}_l(\cdot)$ until it is activated. According to the second main step of our parallel PAC decoding, which is addressed in Subsection 3.2 and summarized on lines 9–14 in Algorithm 1, ψ_l could be recursively calculated as

$$\psi_l = \psi_{l-1} + \overrightarrow{Z}_{(l-1)}, \tag{12}$$

where ψ_{l-1} is the waiting time required by the preceding PAC decoder $\mathcal{F}_{l-1}(\cdot)$. $\overrightarrow{Z}_{(l-1)}$ is the number of code tree nodes visited by the forward moves of $\mathcal{F}_{l-1}(\cdot)$ until it activates $\mathcal{F}_l(\cdot)$. Obviously, we have $\psi_0 = 0$. Finally, Z_l in (11) evaluates the running time of $\mathcal{F}_l(\cdot)$, which is determined by the decoding computational complexity of $\mathcal{F}_l(\cdot)$, i.e., the number of nodes visited by the forward moves of $\mathcal{F}_l(\cdot)$ until it is terminated.

According to the above definitions, the decoding latency and computational complexity of the entire parallel PAC decoder could be evaluated by

$$\phi_{\mathrm{PAR}} = \max_{0 \leqslant l \leqslant L-1} \phi_l, \tag{13}$$

$$Z_{\mathrm{PAR}} = \sum_{l=0}^{L-1} Z_l. \tag{14}$$

5.2 Experiments

In our simulations, without loss of generality, the Reed-Muller rate profile is employed to construct $\mathcal{A}$. Then, the parameters of $g = [1011011]$, $N = 128$, $K = 64$, $\Delta = 2$, as well as $L = 6$ are employed, unless otherwise stated. Particularly, the computational complexity required to calculate the initial threshold list is not taken into account, since that calculation could be off-line.

Firstly, the FER performance of different PAC decoders is compared in Figure 5(a), where the SC-Fano decoder proposed in [12] is referred to as "conventional PAC" decoder, while the parallel PAC decoder employing MC based threshold initialization and that employing DE based threshold initialization are referred to as "Parallel PAC: MC" and "Parallel PAC: DE", respectively. It is evidenced in Figure 5(a) that both the MC-based and the DE-based parallel PAC decoders remain the same FER performance as the conventional PAC decoder. No noticeable performance loss is observed.

The computational complexity and decoding latency performance of above-mentioned "conventional PAC" decoder, "Parallel PAC: MC" decoder, as well as "Parallel PAC: DE" decoder are compared in Figures 5(b) and (c). More attention should be paid to the low SNR region, e.g., SNR $\leqslant 1.5$ dB, because the sequential decoding algorithms will suffer a significantly higher computational complexity in low SNR region than that in high SNR region. It is evidenced in Figure 5(b) that during the critical region of SNR $\leqslant 1.5$ dB, the computational complexity required by the proposed parallel PAC decoders (either MC based or DE based) is higher than that required by conventional PAC decoder, but their additional complexity will not exceed a half of conventional PAC decoder; even a number of 6 individual PAC decoding components are deployed in our parallel PAC decoders. It means the computational complexity

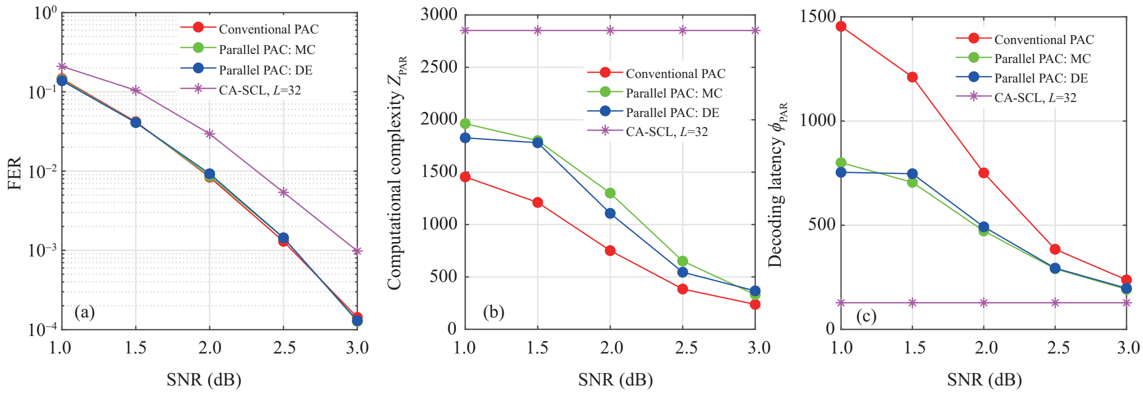

Figure 5 Comparison among different PAC decoding algorithms. (a) FER performance; (b) computational complexity; (c) decoding latency.

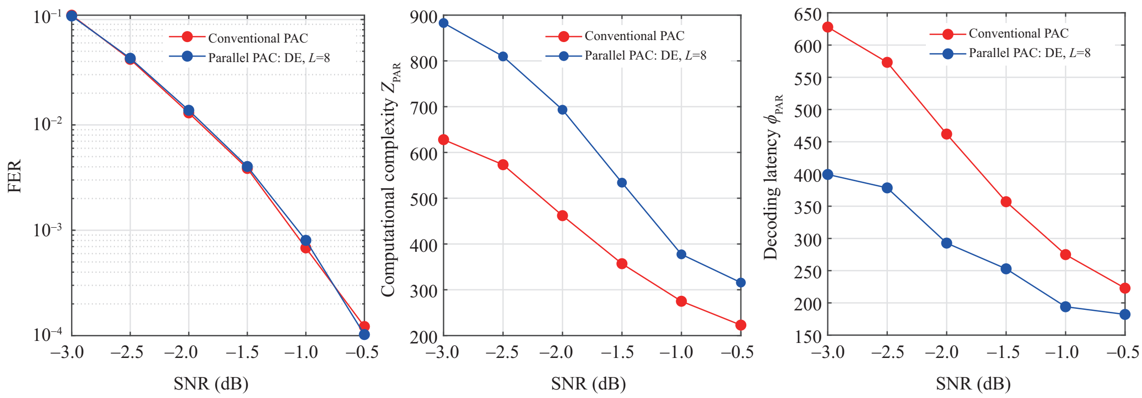

Figure 6 Performance of parallel PAC decoder for the PAC code ($N = 128, K = 29$). (a) FER performance; (b) computational complexity; (c) decoding latency.

of our parallel PAC decoder will not linearly grow with the number of L, which benefits from our adaptive activation strategy that the lth individual PAC decoder $\mathcal{F}_l(\cdot)$ is activated only after its proceeding decoder $\mathcal{F}_{l-1}(\cdot)$ has made a backward movement along the code tree; then a pair of early termination criterion is invoked in.

However, as demonstrated in Figure 5(c), the proposed parallel PAC decoders (either MC based or DE based) are capable of halving the decoding latency of the conventional PAC decoder in the critical low SNR region of SNR $\leqslant 1.5$ dB. Apparently, this property is attractive for short packet and low latency communications.

Furthermore, the relevant performances of classical CA-SCL decoding algorithm using a list size of 32 and 8 CRC bits are also provided in Figure 5. It is clearly observed in Figure 5 that although the CA-SCL algorithm achieves a lower decoding latency, it incurs a significant loss of coding gain (around 0.4 dB) and a higher computational complexity. Bear in mind that, in code design area, normally, the power gain is more important than decoding latency deduction. This is also the motivation to propose PAC codes in [8] on the background that the CA-SCL algorithm has existed for many years. Hence, our design guideline shall be to reduce the decoding latency but without loss of coding gain.

The FER, computational complexity and decoding latency performance of the PAC code ($N = 128, K = 29$) and the PAC code ($N = 256, K = 128$) [19] with the aid of our parallel PAC decoder are demonstrated in Figures 6 and 7, respectively, where our parallel PAC decoder employs the DE-based method and $L = 8$. According to these comparisons between the proposed decoder and the conventional PAC decoder, it is clear that our proposal has the decoding latency again without loss of any noticeable FER performance degradation. Its penalty of imposing higher computational complexity is also effectively constrained, which will not exceed one and a half times of that required by the conventional PAC decoder. This phenomenon is in line with that observed in Figure 5. It implies that the proposed parallel PAC decoder will adapt to different code configurations well.

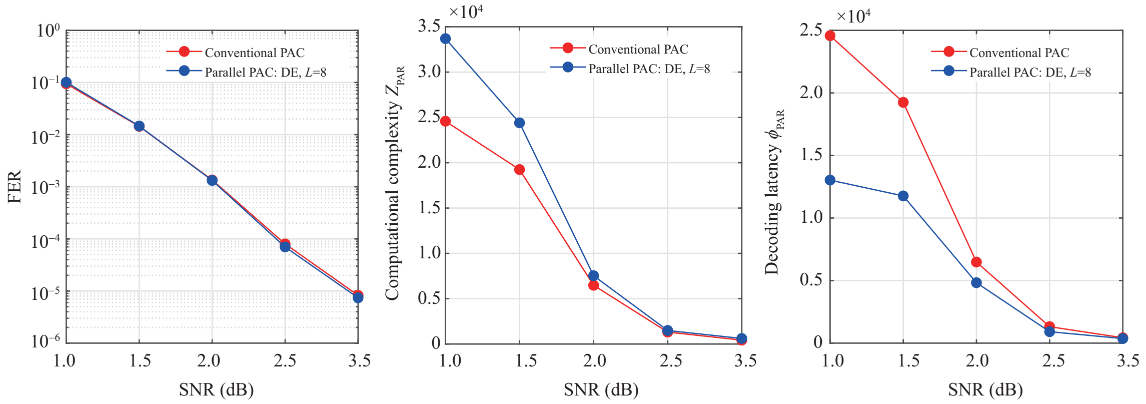

Figure 7 Performance of parallel PAC decoder for the PAC code ($N = 256, K = 128$). (a) FER performance; (b) computational complexity; (c) decoding latency.

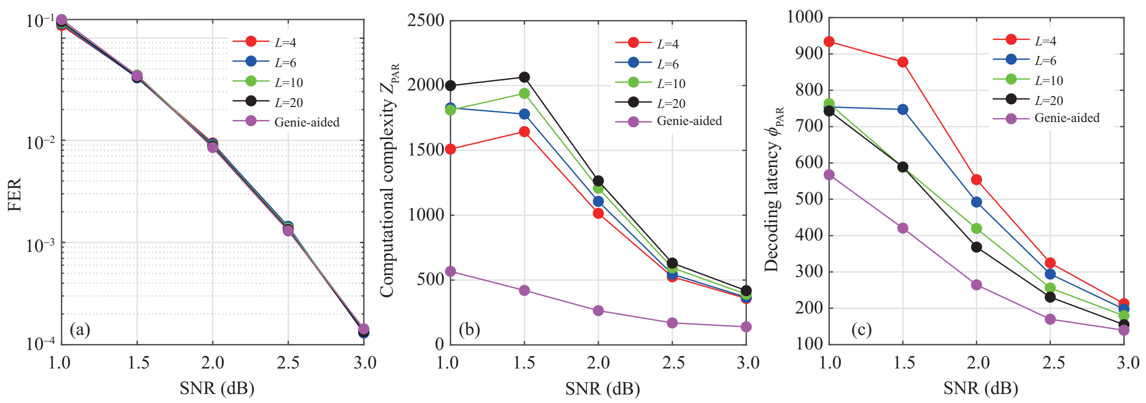

Figure 8 Effect of L on parallel PAC decoder. (a) FER performance; (b) computational complexity; (c) decoding latency.

Table 2 Average number of activated PAC decoding components ζ

L	SNR (dB)				
	1	1.5	2	2.5	3
4	3.27	2.95	2.61	2.34	2.06
6	4.54	3.95	3.36	2.84	2.37
10	4.62	6.04	4.92	3.90	2.96

The impact of the number of individual PAC decoding components L on the parallel PAC decoder is investigated in Figure 8, where the DE-based threshold initialization is employed. It is clear that adding more individual PAC decoding components will effectively reduce the decoding latency, however, meanwhile, a higher computational complexity is also required. Particularly, as observed in Figure 8(c) that the decoding latency of parallel PAC decoder has got closed to that of genie-aided PAC decoder after $L \geqslant 10$. Another penalty of employing multiple individual SC-Fano decoding components in our parallel PAC decoder is increasing the storage requirement compared with a conventional SC-Fano decoder. According to the proposed decoding architecture shown in Figure 3 and decoding method summarized in Algorithm 1, the storage requirement of the proposed parallel PAC decoder shall be approximate ζ times of that required by the conventional SC-Fano decoder, where ζ is the average number of individual SC-Fano decoding components activated during our parallel decoding procedure. The performance of ζ with respect to different SNR configurations while relying on a $(N = 128, K = 64)$ PAC code is shown in Table 2. As verified in Table 2, benefiting from our adaptive activation and early termination strategy again, the extra storage requirement is effectively constrained, especially in high SNR regions. Hence, the number of L shall be appropriately chosen by comprehensively considering the trade-off among computational complexity, decoding latency, FER performance and storage requirement.

6 Conclusion

With the aid of SC-Fano decoding, the FER performance of the recently proposed PAC codes almost approaches the dispersion approximation bound in the case of using short codeword lengths. However, the existing decoder still suffers a high computational complexity in the low SNR region owing to its sequential decoding style. Especially, in some worst cases, the decoding complexity significantly exceeds its average, as a result, the decoding latency becomes unpredictable. In order to overcome this critical deficiency of current PAC codes, a parallel PAC decoding algorithm was proposed in this paper, where the threshold initialization method, the collaboration between multiple individual PAC decoding components, and the early stop criteria were carefully designed. The simulation results demonstrate that the proposed parallel PAC decoder almost has the decoding latency of the conventional PAC decoder, and could tend to the genie-aided bound by combining more and more PAC decoders. Meanwhile, the entire computational complexity is effectively constrained and will not significantly grow with the number of individual PAC decoding components.

Acknowledgements This work was supported by National Key R&D Project (Grant No. 2018YFB1801104), National Natural Science Foundation of China (Grant Nos. 62071394, 62020106001), 111 Project (Grant No. 111-2-14), and Sichuan Science and Technology Program (Grant No.2020YFH0011).

References

1 Arikan E. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. IEEE Trans Inform Theor, 2009, 55: 3051–3073
2 Tal I, Vardy A. List decoding of polar codes. In: Proceedings of 2011 IEEE International Symposium on Information Theory Proceedings, St. Petersburg, 2011. 1–5
3 Niu K, Chen K. CRC-aided decoding of polar codes. IEEE Commun Lett, 2012, 16: 1668–1671
4 Wang T, Qu D, Jiang T. Parity-check-concatenated polar codes. IEEE Commun Lett, 2016, 20: 2342–2345
5 Zhang H, Li R, Wang J, et al. Parity-check polar coding for 5G and beyond. In: Proceedings of 2018 IEEE International Conference on Communications (ICC), Kansas City, 2018. 1–7
6 Li B, Shen H, Tse D. A RM-polar codes. 2014. ArXiv:1407.5483
7 Li B, Zhang H, Gu J. On pre-transformed polar codes. 2019. ArXiv:1912.06359
8 Arıkan E. From sequential decoding to channel polarization and back again. 2019. ArXiv:1908.09594
9 Jeong M O, Hong S N. SC-Fano decoding of polar codes. IEEE Access, 2019, 7: 81682–81690
10 Polyanskiy Y, Poor H V, Verdu S. Channel coding rate in the finite blocklength regime. IEEE Trans Inform Theor, 2010, 56: 2307–2359
11 Fossorier M P C, Mihaljevic M, Imai H. Reduced complexity iterative decoding of low-density parity check codes based on belief propagation. IEEE Trans Commun, 1999, 47: 673–680
12 Moradi M. On sequential decoding metric function of polarization-adjusted convolutional (PAC) codes. IEEE Trans Commun, 2021, 69: 7913–7922
13 Rowshan M, Burg A, Viterbo E. Polarization-adjusted convolutional (PAC) codes: sequential decoding vs list decoding. IEEE Trans Veh Technol, 2021, 70: 1434–1447
14 Yao H, Fazeli A, Vardy A. List decoding of Arıkan's PAC codes. In: Proceedings of IEEE International Symposium on Information Theory (ISIT), Los Angeles, 2020. 443–448
15 Rowshan M, Viterbo E. List Viterbi decoding of PAC codes. IEEE Trans Veh Technol, 2021, 70: 2428–2435
16 Moradi M, Mozammel A, Qin K, et al. Performance and complexity of sequential decoding of PAC codes. 2020. ArXiv:2012.04990
17 Fano R. A heuristic discussion of probabilistic decoding. IEEE Trans Inform Theor, 1963, 9: 64–74
18 Trifonov P. Efficient design and decoding of polar codes. IEEE Trans Commun, 2012, 60: 3221–3227
19 Moradi M, Mozammel A. A Monte-Carlo based construction of polarization-adjusted convolutional (PAC) codes. 2021. ArXiv:2106.08118