Decoding Polar Codes with Reinforcement Learning

Nghia Doan*, Seyyed Ali Hashemi†, Warren J. Gross*
*Department of Electrical and Computer Engineering, McGill University, Canada
†Department of Electrical Engineering, Stanford University, USA
nghia.doan@mail.mcgill.ca, ahashemi@stanford.edu, warren.gross@mcgill.ca

***Abstract*—In this paper we address the problem of selecting factor-graph permutations of polar codes under belief propagation (BP) decoding to significantly improve the error-correction performance of the code. In particular, we formalize the factor-graph permutation selection as the multi-armed bandit problem in reinforcement learning and propose a decoder that acts like an online-learning agent that learns to select the good factor-graph permutations during the course of decoding. We use state-of-the-art algorithms for the multi-armed bandit problem and show that for a 5G polar codes of length 128 with 64 information bits, the proposed decoder has an error-correction performance gain of around 0.125 dB at the target frame error rate of 10^{-4}, when compared to the approach that randomly selects the factor-graph permutations.**

***Index Terms*—5G, polar codes, belief propagation, factor-graph permutations, machine learning, reinforcement learning.**

I. Introduction

Polar codes are a breakthrough in the field of channel coding as they can achieve the capacity of any binary symmetric channel with efficient encoding and decoding algorithms [1]. Successive cancellation (SC) and belief propagation (BP) decoding algorithms were introduced in [1] to decode polar codes. Although SC decoding can provide a low-complexity implementation, its serial nature prevents the decoder to reach a high decoding throughput. Furthermore, the error-correction performance of SC decoding for short to moderate polar codes does not satisfy the requirements of the fifth generation of cellular mobile communications (5G) standard. To improve the error-correction performance of SC decoding, SC list (SCL) decoding was introduced in [2] and it was shown that SCL can provide a significant error-correction performance improvement if it is concatenated with a cyclic redundancy check (CRC). Based on this observation, polar codes have been selected to be used in the enhanced mobile broadband (eMBB) control channel of 5G together with a CRC [3].

Unlike SC-based decoders, the iterative message passing process of BP decoding can be executed in parallel, hence enabling the decoder to reach high decoding throughput. However, the conventional BP decoding algorithm suffers from poor error-correction performance. It has been shown that if polar codes are concatenated with a CRC, the error-correction performance of them under BP decoding can be significantly improved by exploiting the extrinsic information between the factor graphs of polar codes and the CRC [4], [5]. In addition, by using multiple independent permutations of the factor-graph of polar codes, the error-correction performance of them under BP decoding is significantly improved [5]–[9]. However, the selection of good factor-graph permutations for polar codes that result in a correctly decoded codeword given a specific channel output realization remains an open research problem.

In this paper, we first formalize the selection of factor-graph permutations of polar codes under the CRC-aided (CA) BP (CABP) decoder in [4] as a multi-armed bandit problem in reinforcement learning (RL). We then utilize state-of-the-art algorithms designed for the multi-armed bandit problem to select the factor-graph permutations of polar codes that work best under CABP decoding. Unlike existing approaches, such as using genetic algorithm [5] or Monte Carlo-based methods [8], [9], in which the mechanism for the selection of factor-graph permutations requires off-line training, the proposed approach treats the CABP-based decoding of polar codes as an online-learning agent that learns to select the good factor-graph permutations during the course of decoding. We show that for a 5G polar code of length 128 with 64 information bits and concatenated with a 16-bit 5G CRC, the proposed RL-aided CABP (RL-CABP) decoding algorithm has an error-correction performance gain of around 0.125 dB, at the target frame error rate (FER) of 10^{-4}, compared to the approach that selects the factor-graph permutations of polar codes randomly.

The remainder of the paper is as follows. Section II provides background on polar codes and BP-based decoding algorithms. Section III summarizes the multi-armed bandit problem and its state-of-the-art algorithms. Section IV introduces the proposed decoding algorithm, followed by the experimental results provided in Section V. Finally, concluding remarks are drawn in Section VI.

II. Polar Codes

A. Polar Encoding

A polar code $\mathcal{P}(N, K)$ of length N with K information bits is constructed by applying a linear transformation to the binary message word $\boldsymbol{u} = \{u_0, u_1, \ldots, u_{N-1}\}$ as $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{G}^{\otimes n}$ where $\boldsymbol{x} = \{x_0, x_1, \ldots, x_{N-1}\}$ is the codeword, $\boldsymbol{G}^{\otimes n}$ is the n-th Kronecker power of the polarizing matrix $\boldsymbol{G} = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$, and $n = \log_2 N$. The vector $\boldsymbol{u}$ contains a set $\mathcal{I}$ of K information bit indices and a set $\mathcal{I}^c$ of $N - K$ frozen bit indices. The positions of the frozen bits are known to the encoder and the decoder and their values are set to 0. The codeword $\boldsymbol{x}$ is then modulated and sent through the channel. In this paper, binary phase-shift keying (BPSK) modulation and additive white Gaussian noise (AWGN) channel model are considered. Therefore, the soft vector of the transmitted codeword received by the decoder is written as $\boldsymbol{y} = (\mathbf{1} - 2\boldsymbol{x}) + \boldsymbol{z}$, where $\mathbf{1}$ is an

all-one vector of size N, and $\boldsymbol{z} \in \mathbb{R}^N$ is a Gaussian noise vector with variance σ^2 and zero mean. In the log-likelihood ratio (LLR) domain, the LLR vector of the transmitted codeword is given as $\boldsymbol{L} = \ln \frac{\Pr(\boldsymbol{x}=0|\boldsymbol{y})}{\Pr(\boldsymbol{x}=1|\boldsymbol{y})} = \frac{2\boldsymbol{y}}{\sigma^2}$.

B. Belief Propagation Decoding of Polar Codes

Fig. 1a illustrates BP decoding on a factor graph representation of $\mathcal{P}(8,5)$. The messages are iteratively propagated through the processing elements (PEs) [10]. An update iteration starts with a right-to-left message pass that propagates the LLR values from the channel stage (right-most stage), to the information bit stage (left-most stage), and ends with the left-to-right message pass occurring in the reverse order. Fig. 1b shows a PE with its corresponding messages, where $r_{s,i}$ denotes a left-to-right message, and $l_{s,i}$ denotes a right-to-left message of the i-th bit index at stage s. The update rule for the right-to-left messages of a PE is [10]

$$\begin{cases} l_{s,i} & = f(l_{s+1,i}, r_{s,i+2^s} + l_{s,i+2^s}), \\ l_{s,i+2^s} & = f(l_{s+1,i}, r_{s,i}) + l_{s+1,i+2^s}, \end{cases} \tag{1}$$

and for the left-to-right messages is

$$\begin{cases} r_{s+1,i} & = f(r_{s,i}, l_{s+1,i+2^s} + r_{s,i+2^s}), \\ r_{s+1,i+2^s} & = f(r_{s,i}, l_{s+1,i}) + r_{s,i+2^s}, \end{cases} \tag{2}$$

where $f(.)$ is the scaled min-sum function [11]:

$$f(x,y) = 0.9375 \times \operatorname{sgn}(x)\operatorname{sgn}(y)\min(|x|,|y|). \tag{3}$$

The BP decoding performs a predetermined $I_{\max}$ iterations where the messages are propagated through all PEs in accordance with (1) and (2). The LLR values at stage 0, denoted as $\boldsymbol{r}_0$, are initialized as

$$r_{0,i} = \begin{cases} 0, & \text{if } i \in \mathcal{I}, \\ +\infty, & \text{if } i \in \mathcal{I}^c, \end{cases} \tag{4}$$

and the LLR values at stage n, denoted as $\boldsymbol{l}_n$, are initialized as $\boldsymbol{l}_n = \boldsymbol{L}$. In addition, all the other left-to-right and right-to-left messages of the PEs at the first iteration are set to 0. After running $I_{\max}$ iterations, the decoder makes a hard decision on the LLR values of the i-th bit at the information bit stage to obtain the estimated message word as

$$\hat{u}_i = \begin{cases} 0, & \text{if } r_{0,i} + l_{0,i} \geq 0, \\ 1, & \text{otherwise.} \end{cases} \tag{5}$$

In this paper we consider the case where a CRC is concatenated to the polar code as in the 5G standard. After running BP decoding on the factor-graph of polar codes for $I_{\min}$ iterations ($0 < I_{\min} < I_{\max}$), a CRC verification is performed to early-terminate the decoding process. In addition, the factor-graph of CRC is utilized to further improve the error-correction performance of polar codes under BP decoding in a way that the extrinsic information of the factor-graphs of CRC and polar codes is exchanged by running BP decoding on both factor-graphs after the $I_{\min}$-th iteration [4]. We refer to this algorithm as CABP decoding.

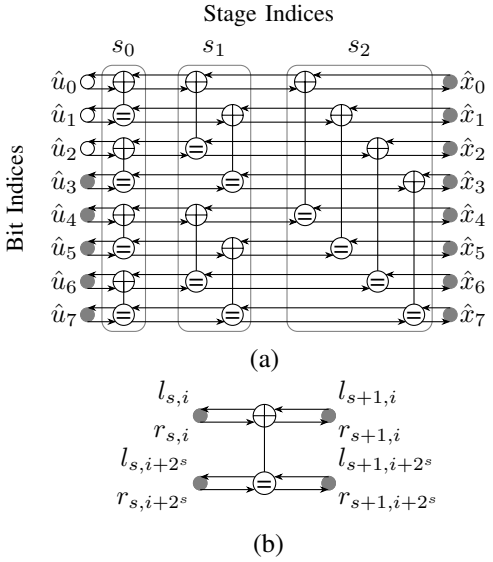

Fig. 1: (a) Factor-graph representation of $\mathcal{P}(8,5)$ with $\mathcal{I}^c = \{0,1,2\}$, (b) a PE for BP decoding.

C. Decoding Polar Codes on Factor-Graph Permutations

The error-correction performance of polar codes under different decoding algorithms can significantly improve if the decoding is performed independently on multiple factor-graph permutations [5]–[9]. A factor-graph permutation, denoted as π_p $(0 \leq p < n!)$, is constructed by permuting the PE stages of the polar codes factor graph [6]. For instance, Fig. 1a shows the original factor graph of $\mathcal{P}(8,5)$, denoted as $\pi_0 = \{s_0, s_1, s_2\}$. Permuting the PEs in stage s_1 and s_2 in Fig. 1a forms another factor-graph permutation, $\pi_1 = \{s_0, s_2, s_1\}$. It was shown that there is a one-to-one mapping between the factor-graph permutation and the bit-index permutation of the original factor-graph [8]. Thus, the decoding of polar codes on their permuted factor graphs can be performed by running the decoder on the permuted bit-indices of the original factor graph. This keeps the architecture of the decoder unchanged [8].

In this paper, given π_p and $\boldsymbol{L}$, we use the technique presented in [8] to form the corresponding permuted bit-indices of the channel LLR values, $\boldsymbol{L}_{\pi_p}$. We then apply CABP decoding on $\boldsymbol{L}_{\pi_p}$ using the original factor-graph. Note that the permuted soft messages of the information bit stage $\boldsymbol{l}_{0_{\pi_p}}$ is permuted back to $\boldsymbol{l}_0$ before running BP decoding on the CRC factor-graph. Given $\boldsymbol{L}$ and π_p, we consider CABP decoding as a function and denote its output as $\hat{\boldsymbol{u}} = \mathrm{CABP}(\boldsymbol{L}, \pi_p)$. In addition, throughout this paper, we refer to π_0 as the permutation corresponding to the original factor-graph.

III. Multi-Armed Bandit Problem

A multi-armed bandit problem, or a k-armed bandit problem $(k > 1)$, is an RL problem where an agent has to repeatedly make a choice among k different actions (options). After each action is performed, the agent receives a numerical reward that is drawn from a distribution that depends on the selected action. The agent's objective is to maximize

the expected cumulative rewards over a time period [12]. Let $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$ be the set of actions and $q^*(a_j)$ $(1 \leq j \leq k)$ be the corresponding expected reward of an action a_j. $q^*(a_j)$ is called the value function and its value is unknown to the agent. In this paper, we consider three state-of-the-art algorithms designed for the multi-armed bandit problem, namely, ε-greedy, upper confidence bound (UCB), and Thompson sampling (TS).

A. ε-Greedy and UCB Algorithms

Let n_{a_j} be the number of times that an action a_j is selected up to the t-th time step. If a_j is selected at the t-th time step, n_{a_j} is updated as $n_{a_j} := n_{a_j} + 1$ [12]. Then, the value function $q^*(a_j)$ is estimated as Q_{a_j} in accordance with $Q_{a_j} := Q_{a_j} + \frac{1}{n_{a_j}}\left[R_t - Q_{a_j}\right]$, where R_t is the reward received by selecting action a_j at the t-th time step [12]. Initially, Q_{a_j} and a_j are set to 0 $(\forall j, 1 \leq j \leq k)$. Given the estimated expected rewards Q_{a_j}, an exploitation occurs when the agent selects an action that has the largest expected reward value [12]. On the other hand, an exploration occurs when the agent selects any action that does not have the largest expected reward value [12].

Let a_{j^*} be the action selected by the agent at the t-th time step. Under the ε-greedy algorithm a_{j^*} is selected as [12]

$$a_{j^*} = \begin{cases} \underset{\forall a_j}{\arg\max}\, Q_{a_j} & \text{with probability } 1-\varepsilon, \\ a_{\text{random}} & \text{with probability } \varepsilon, \end{cases} \tag{6}$$

where a_{random} is a random action drawn i.i.d. from $\mathcal{A}$. On the other hand, under the UCB algorithm a_{j^*} is selected as

$$a_{j^*} = \underset{\forall a_j}{\arg\max} \left[Q_{a_j} + c\sqrt{\frac{\ln t}{n_{a_j}}} \right], \tag{7}$$

where $n_{a_j} \neq 0$ and $c \in \mathbb{R}^+$. If $n_{a_j} = 0$, a_j is considered as an exploitation action. Note that ε and c control the degree of exploration of the ε-greedy and UCB algorithms, respectively.

B. Thompson Sampling

Instead of estimating the expected reward value $q^*(a_j)$ as in the ε-greedy and UCB algorithms, the TS algorithm directly estimates the distribution of the reward value associated with each action. In this paper, as $R_t \in \{0, 1\}$ a Beta distribution is used to estimate the reward's distribution [13]. A Beta distribution has two shape parameters: $\alpha, \beta \in \mathbb{R}^+$, and a different set of shape parameters is used for each action. We denote a random sampling from the estimated reward distribution of the j-th action as $\upsilon_{a_j} = \text{Beta}(\alpha_{a_j}, \beta_{a_j})$. At the t-th time step, the TS algorithm first draws a random sample from each of the estimated reward distributions. The agent then selects the action a_{j^*} as $a_{j^*} = \arg\max_{\forall a_j} \upsilon_{a_j}$. The shape parameters corresponding to the selected action a_{j^*} are then updated as $\alpha_{a_{j^*}} := \alpha_{a_{j^*}} + R_t$ and $\beta_{a_{j^*}} := \beta_{a_{j^*}} + R_t$ [13]. Initially, $\alpha_{a_j} = \beta_{a_j} = 1$ $(\forall j, 1 \leq j \leq k)$ [13].

IV. Selection of Factor-Graph Permutations with Reinforcement Learning

This section first formalizes the selection of factor-graph permutations for polar decoding as a k-armed bandit problem. It then introduces the proposed decoding method that utilizes the multi-armed bandit algorithms in Section III to select the factor-graph permutations under CABP decoding.

A. Problem Formulation

Under BP decoding of polar codes, the original factor-graph permutation π_0 is empirically observed to have the best error-correction performance compared to other factor-graph permutations [6]. However, there are cases that a specific channel output realization, which cannot be decoded using the original factor-graph permutation, can be decoded using another factor-graph permutation [6]. As the number of permutations, $n!$, is large, running BP decoding on all of the permutations is not possible in real applications. Instead, the decoding is performed on a small set of M factor-graph permutations, including the original factor-graph permutation [5]–[8].

Let an action $a_j \in \mathcal{A}$ $(1 \leq j \leq k)$ be a random selection of $M-1$ $(M > 1)$ factor-graph permutations that do not include the original factor-graph permutation. Consider the CRC verification is not successful when CABP decoding is performed on the original factor-graph permutation π_0. The proposed decoder then selects an action a_j from the set $\mathcal{A}$. If one of the factor-graph permutations in a_j results in a successful CRC verification, a reward of 1 is given to the decoder. Otherwise, if none of the permutations in a_j results in a successful CRC verification under CABP decoding, a reward of 0 is given to the decoder. Therefore, among k sets of predefined factor-graph permutations, i.e. k different actions, the proposed decoding algorithm decides which set of factor-graph permutations maximizes the reward during the course of decoding. The selection of factor-graph permutations for CABP decoding can thus be formalized as a k-armed bandit problem as defined in Section III.

B. Reinforcement Learning-Aided CABP

The proposed decoding algorithm starts with the construction of $\mathcal{A}$, the set of k different actions, which is outlined in Algorithm 1. Each action $a_j \in \mathcal{A}$ contains $M-1$ random factor-graph permutations. Formally, $a_j = \{\pi_{j,1}, \pi_{j,2}, \cdots, \pi_{j,M-1}\}$, $\pi_{j,t} \neq \pi_0 \ \forall j, t$, where $1 \leq j \leq k$, and $1 \leq t \leq M-1$. A random factor-graph permutation is formed by randomly permuting the PE stages of the original factor graph π_0, which is obtained by the RandShuffle function in Algorithm 1. The number of all possible actions is

$$k_{\max} = \binom{n! - 1}{M - 1} = \frac{(n! - 1)!}{(M-1)!(n! - M)!}, \tag{8}$$

which is generally intractable for practical values of n and M. Therefore, only the subset $\mathcal{A}$ of all the possible actions is considered. In fact, $\mathcal{A}$ is constructed by randomly sampling from the complete set of actions as shown in Algorithm 1.

Algorithm 1: Forming the action set

Input : n, k, M
Output: $\mathcal{A}$

// Define the original permutation
$\pi_0 \leftarrow \{s_0, s_1, \cdots, s_{n-1}\}$
// Select $M-1$ random permutations for each action
$\mathcal{A} \leftarrow \emptyset$
for $j \leftarrow 1$ **to** k **do**
  $a_j \leftarrow \emptyset$
  for $t \leftarrow 1$ **to** $M-1$ **do**
    $\pi_{j,t} \leftarrow \text{RandShuffle}(\pi_0)$
    $a_j \leftarrow a_j \cup \pi_{j,t}$
  $\mathcal{A} \leftarrow \mathcal{A} \cup a_j$
return $\mathcal{A}$

Note that after $\mathcal{A}$ is formed, the set of actions in $\mathcal{A}$ remains unchanged during the course of decoding.

Algorithm 2 outlines the proposed RL-CABP decoding algorithm, given the predefined set of actions $\mathcal{A}$ constructed in Algorithm 1. The proposed RL-CABP decoder first initializes the parameters of the multi-armed bandit algorithm depending on its type, which is defined by the parameter Algo in Algorithm 2. If Algo indicates the ε-greedy or UCB algorithms, the parameters of the multi-armed bandit algorithm are initialized as $Q_{a_j} = n_{a_j} = 0 \ \forall j$, $1 \leq j \leq k$. If the TS algorithm is used, the set of parameters is initialized as $\alpha_{a_j} = \beta_{a_j} = 1 \ \forall j$, $1 \leq j \leq k$. Note that the initialization process is only carried out once in the course of decoding.

Then, the proposed RL-CABP decoding applies CABP decoding over the original factor-graph permutation π_0. If the CRC verification, which is obtained by the VerifyCRC function in Algorithm 2 is successful, the proposed decoder outputs the estimated message word $\hat{\boldsymbol{u}}$ and the decoding process is terminated. Otherwise, the RL-CABP decoder selects an action a_{j^*} from $\mathcal{A}$, which contains a set of $M-1$ random factor-graph permutations as described in Algorithm 1. Depending on the type of the algorithm, the function SelectAction implements the selection criteria of the considered multi-armed bandit algorithms as introduced in Section III. Note that the SelectAction function can be performed in parallel with the first CABP decoding attempt as there is no dependency between them. Therefore, the selected action a_{j^*} can be obtained in advance without adding a latency overhead to the proposed decoding algorithm. Moreover, if the first CABP decoding attempt over π_0 is successful, the selected action a_{j^*} is discarded.

If the first CABP decoding attempt fails in the proposed RL-CABP decoding algorithm, additional CABP decoding attempts are sequentially carried over the factor-graph permutations specified by a_{j^*}. As soon as the CRC verification is successful after CABP decoding on one of the factor-graph permutations in a_{j^*}, a reward of 1 is given to a_{j^*}, and the

Algorithm 2: RL-CABP Decoding

Input : $\boldsymbol{L}, \mathcal{A}, k, M, \text{Algo}$
Output: $\hat{\boldsymbol{u}}$

// Initialize the bandit parameters
InitBandit(k, Algo)
// Apply CABP decoding on π_0
$\hat{\boldsymbol{u}} \leftarrow \text{CABP}(\boldsymbol{L}, \pi_0)$
$isCorrect_{\pi_0} \leftarrow \text{VerifyCRC}(\hat{\boldsymbol{u}})$
// Select an action in advance
$a_{j^*} \leftarrow \text{SelectAction}(\mathcal{A}, \text{Algo})$
// If applicable, apply CABP decoding on the permutations specified by a_{j^*}
if $(isCorrect_{\pi_0} = 0)$ **then**
  $isCorrect_{a_{j^*}} \leftarrow 0$
  for $t \leftarrow 1$ **to** $M-1$ **do**
    $\hat{\boldsymbol{u}} \leftarrow \text{CABP}(\boldsymbol{L}, \pi_{j^*,t})$
    $isCorrect_{a_{j^*}} \leftarrow \text{VerifyCRC}(\hat{\boldsymbol{u}})$
    if $(isCorrect_{a_{j^*}} = 1)$ **then**
      break
  // Update the bandit parameters associated with a_{j^*}
  $R_t \leftarrow isCorrect_{a_{j^*}}$
  UpdateBandit$(R_t, a_{j^*}, \text{Algo})$
return $\hat{\boldsymbol{u}}$

decoding outputs the estimated message word that satisfies the CRC verification. On the other hand, if running CABP on all of the permutations in a_{j^*} does not result in a successful CRC test, a reward of 0 is given to a_{j^*} and the decoding is declared unsuccessful. Finally, after each action selection, the parameters associated with the selected action a_{j^*} are updated using the UpdateBandit function. Note that the parameter update process is based on the received reward and the type of the mutli-armed bandit algorithm as provided in Section III.

V. Experimental Results

In this section, the performance of various multi-armed bandit algorithms used by the proposed RL-CABP decoding is numerically evaluated. In addition, the error-correction performance of the proposed RL-CABP decoding in terms of FER is compared with that of other polar decoding techniques. A complexity comparison of different multi-armed bandit algorithms in the proposed RL-CABP decoding is also given. We use $\mathcal{P}(128, 64)$ selected for the eMBB control channel of the 5G standard [3]. Furthermore, the polar code is concatenated with a CRC of length 16, which is also used in 5G [3]. The total number of factor-graph permutations used by all BP-based decoders is set to 7. We set $I_{\max} = 100$ and $I_{\min} = 50$ for all BP-based decoding algorithms.

Fig. 2 illustrates the dependence of the average reward on the parameters in ε-greedy and UCB algorithms for $\mathcal{P}(128, 64)$. The simulation is carried out at $E_b/N_0 = 3.0$ dB and we set $k = 500$ for all multi-armed bandit algorithms.

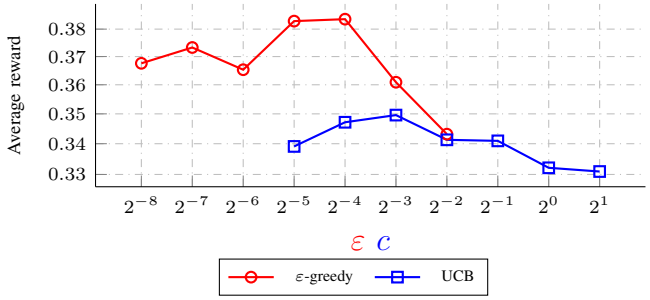

Fig. 2: A parameter study of the ε-greedy and UCB algorithms. The average reward is obtained for the first 10000 time steps with $k = 500$ at $E_b/N_0 = 3.0$ dB.

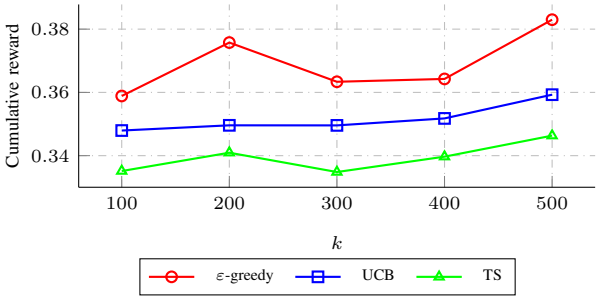

Fig. 3: The impact of k on the performance of different multi-armed bandit algorithms used by RL-CABP decoding for $\mathcal{P}(128, 64)$, obtained for the first 10000 time steps.

In this figure, the average reward of the first 10000 time steps received by the RL-CABP decoder is plotted against the parameter value. Note that a time step is increased by 1 when the multi-armed bandit algorithm is required for the action selection, i.e., when CABP decoding has failed on the original factor-graph permutation π_0. As seen from Fig. 2, at $\varepsilon = 2^{-4}$ and $c = 2^{-3}$, RL-CABP decoding has the highest average reward value for ε-greedy and UCB algorithms, respectively. The TS algorithm does not require parameter tuning since α and β parameters associated with each action are optimized during the decoding process.

Fig. 3 illustrates the performance of multi-armed bandit algorithms used by RL-CABP decoding with different values of k. This simulation is also carried out at $E_b/N_0 = 3.0$ dB. We set $\varepsilon = 2^{-4}$ for the ε-greedy algorithm and $c = 2^{-3}$ for the UCB algorithm as those configurations provide the best performance in Fig. 2. It can be observed that for all the bandit algorithms, $k = 500$ provides the largest cumulative reward after the first 10000 time steps. Thus, we set $k = 500$ for the rest of the paper.

Fig. 4 illustrates the average cumulative reward over the first 10000 time steps for all the multi-armed bandit algorithms. The simulation is performed at $E_b/N_0 = 3.0$ dB with $k = 500$, $\varepsilon = 2^{-4}$, and $c = 2^{-3}$. It can be seen that the ε-greedy algorithm has the best performance in terms of the average cumulative reward. In addition, the UCB algorithm performs slightly better than the TS algorithm. Note that the spikes in the early part of the curves are caused by the small

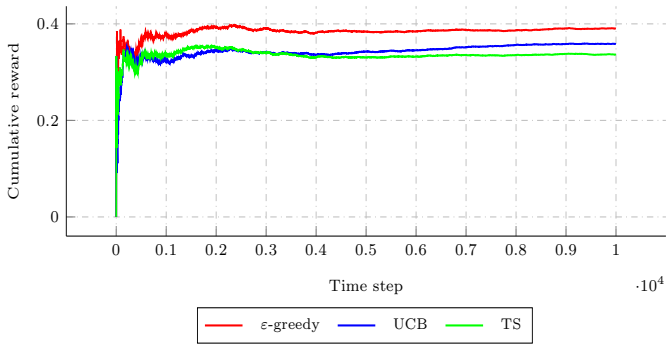

Fig. 4: Performance comparison of various multi-armed bandit algorithms used by RL-CABP decoding. The simulation is obtained at $E_b/N_0 = 3.0$ dB with $k = 500$, $\varepsilon = 2^{-4}$, and $c = 2^{-3}$.

value of the time step, which makes the calculation of the average cumulative reward unreliable at the initial phases of the algorithm.

Fig. 5 compares the FER of different factor-graph permutation selection schemes under the CABP decoding algorithm. In this figure, CABP denotes the CABP decoding algorithm performed only on the original factor-graph permutation. CP-CABP and RP-CABP denote the cyclically-shifted and random factor-graph permutations selection schemes proposed in [6] and [7], respectively. Note that as there are $n = 7$ cyclically-shifted permutations for $\mathcal{P}(128, 64)$, we set the number of additional random permutations used by RP-CABP to 6, and $M = 7$ for the proposed RL-CABP decoder for a fair comparison. It can be seen that the proposed RL-CABP decoder under various multi-armed bandit algorithms has a similar FER performance. When compared with CP-CABP and RP-CABP, an error-correction performance gain of at least 0.125 dB is obtained at the target FER of 10^{-4}. In addition, an FER gain of around 0.62 dB is obtained when the proposed RL-CABP decoding algorithm is compared with the baseline CABP decoder at the FER of 10^{-4}.

Fig. 6 compares the error-correction performance of the proposed RL-CABP decoding with BP decoding and CA-SCL decoding of polar codes. In Fig. 6, CA-SCLL indicates the CA-SCL decoder with a list size of L. It can be observed that at the target FER of 10^{-4}, the FER performance of the proposed RL-CABP decoder is around 0.92 dB better than that of the BP decoding algorithm in [11]. At the same target FER, CA-SCL4 provides a better error-correction performance in comparison with the proposed RL-CABP decoder. However, compared with CA-SCL2 at the same target FER, the proposed decoder has a performance gain of around 0.12 dB, under different multi-armed bandit algorithms.

Table I shows the maximum number of computations required by various permutation selection schemes used in Fig. 5. Among all the multi-armed bandit algorithms, the ε-greedy algorithm in general has the lowest computational complexity. This is because the TS algorithm requires a sampling process for k different Beta distributions, which in general requires higher computational complexity than applying an

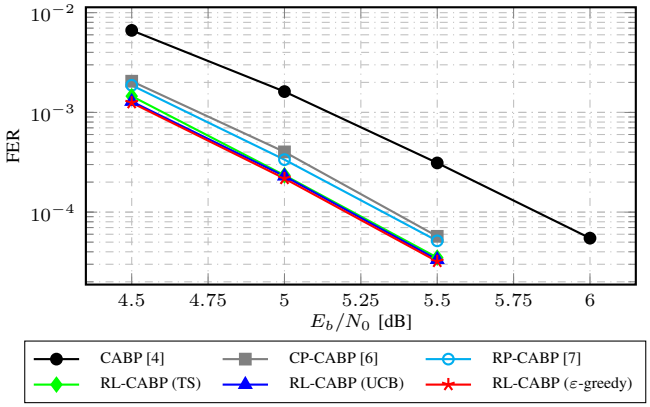

Fig. 5: Error-correction performance of different factor-graph permutation selection schemes for $\mathcal{P}(128, 64)$.

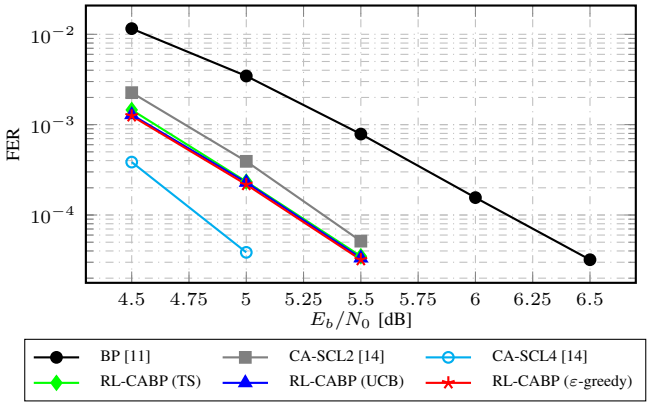

Fig. 6: Error-correction performance of RL-CABP decoding and other decoding algorithms of polar codes.

i.i.d. sampling from the interval of $(0, 1)$ and doing a multiplication as required by the ε-greedy algorithm. In addition, although using the cyclically-shifted factor-graph permutations does not consume any additional complexity for the factor-graph permutation selection, this technique is not applicable when more than n different permutations are required. It can also be observed that the main drawback of the multi-armed bandit algorithms is the sorting operations required to identify the exploitation action. However, as described in Section IV-B, the action selection process can be performed in parallel with the first CABP decoding attempt. Therefore, there is no additional latency overhead. Furthermore, the approaches in [6] and [7] come with the cost of error-correction performance degradation when compared with the proposed RL-CABP decoder as illustrated in Fig. 5.

VI. Conclusions

In this paper, we first showed that the selection of factor-graph permutations for polar decoding can be formalized as a multi-armed bandit problem in RL. We then proposed an RL-CABP decoding algorithm that utilizes the state-of-the-art algorithms for the multi-armed bandit problem to select the factor-graph permutations under CABP decoding of polar codes. We showed that for a 5G polar code of length 128, with 64 information bits and concatenated with a 16-bit 5G CRC, the FER of the proposed decoder is around 0.125 dB better than that of the technique that selects the factor-graph permutations randomly, at the target FER of 10^{-4}. In addition, we showed that there is no additional latency overhead for the selection of factor-graph permutations of the proposed decoder compared with the approach that selects the factor-graph permutations at random.

TABLE I: Computational complexity of different permutation selection schemes in terms of the maximum number of operations performed

Operations	[6]	[7]	ε-greedy	UCB	TS
$+$	0	0	2	$2 + k$	2
$-$	0	0	1	1	0
$\times$	0	0	1	$1+k$	0
$\div$	0	0	0	k	0
$\sqrt{}$	0	0	0	k	0
ln	0	0	0	k	0
Random sampling	0	$M-1$	1	0	k
Sorting	0	0	k	k	k

Acknowledgment

S. A. Hashemi is supported by a Postdoctoral Fellowship from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] 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, July 2009.

[2] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, March 2015.

[3] 3GPP, "Multiplexing and channel coding (Release 10) 3GPP TS 21.101 v10.4.0." Oct. 2018. [Online]. Available: http://www.3gpp.org/ftp/Specs/2018-09/Rel-10/21_series/21101-a40.zip

[4] N. Doan, S. A. Hashemi, E. N. Mambou, T. Tonnellier, and W. J. Gross, "Neural belief propagation decoding of CRC-polar concatenated codes," *IEEE Int. Conf. on Commun.*, pp. 1–6, May 2019.

[5] M. Geiselhart, A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "CRC-aided belief propagation list decoding of polar codes," *IEEE Int. Sym. on Inf. Theory (to appear).*, 2020. [Online]. Available: https://arxiv.org/abs/2001.05303

[6] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding," in *IEEE Int. Symp. on Inf. Theory*, 2009, pp. 1488–1492.

[7] A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "Belief propagation decoding of polar codes on permuted factor graphs," in *IEEE Wireless Commun. and Net. Conf.*, April 2018, pp. 1–6.

[8] N. Doan, S. A. Hashemi, M. Mondelli, and W. J. Gross, "On the decoding of polar codes on permuted factor graphs," *IEEE Global Commun. Conf.*, pp. 1–6, Dec 2018.

[9] Y. Ren, Y. Shen, Z. Zhang, X. You, and C. Zhang, "Efficient belief propagation polar decoder with loop simplification based factor graphs," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2020.

[10] E. Arıkan, "Polar codes: A pipelined implementation," in *Proc. 4th Int. Symp. on Broad. Commun.*, 2010, pp. 11–14.

[11] B. Yuan and K. K. Parhi, "Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders," *IEEE Transactions on Signal Processing*, vol. 62, no. 24, pp. 6496–6506, Dec. 2014.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[13] S. Agrawal and N. Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," in *Conf. on Learning Theory*, 2012, pp. 39–1.

[14] 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, Oct. 2015.