Polar Sampler: Discrete Gaussian Sampling over the Integers Using Polar Codes

Jiabo Wang and Cong Ling

Imperial College London
j.wang16@imperial.ac.uk, c.ling@imperial.ac.uk

Abstract. Cryptographic constructions based on hard lattice problems have emerged as a front runner for the standardization of post quantum public key cryptography. As the standardization process takes place, optimizing specific parts of proposed schemes becomes a worthwhile endeavor. Gaussian sampling over the integers is one of the fundamental building blocks of latticed-based cryptography. In this work, we propose a new integer Gaussian sampler based on polar codes, dubbed "polar sampler". The polar sampler is asymptotically information theoretically optimum in the sense that the number of uniformly random bits it uses approaches the entropy bound. It also features quasi-linear complexity and constant-time implementation. Our algorithm becomes effective when sufficiently many samples are required at each query to the sampler. Security analysis is given based on the statistical distance, Kullback-Leibler divergence and Rényi divergence. A comparison between the polar sampler and the Knuth-Yao sampler verifies its time efficiency and the memory cost can be further optimized if space-efficient successive-cancellation decoding is adopted.

Keywords: Discrete Gaussian sampling · Polar codes · Integer lattice · Kullback-Leibler divergence · Constant-time implementation.

1 Introduction

Lattice-based cryptography is one of the most promising candidates of cryptosystems in the future post-quantum age. The security of lattice-based primitives is guaranteed by the hardness of worst-case lattice problems, e.g. the Learning With Errors (LWE) problem [26, 15] and Short Integer Solution (SIS) problem [18, 17]. The discrete Gaussian distribution lies at the core of security proofs of these primitives, and it is also one of the fundamental building blocks of practical lattice-based cryptographic applications, e.g. signature schemes, encryption and key exchanges. In general, the security level of these cryptographic applications is closely related to the statistical performance of the discrete Gaussian sampling (DGS) algorithm. From an implementation standpoint, cryptographers also take other qualities of a DGS into consideration including side-channel resistance, computation and storage efficiency. In practice, the tradeoff between these performances is a bottleneck of this problem.

It has been widely assumed that for cryptographic applications with λ bits of security the statistical distance (SD) between the ideal distribution and the approximated one should be roughly $2^{-\lambda}$ such that there is only minor loss in security [8]. Some other measures such as Kullback-Leibler (KL) divergence and Rényi divergence are proved to provide more efficient security analysis than the SD, as they can lower the requirement for precision and reduce the cost of the algorithms in many practical cases [22, 24, 25, 3]. From a practical point of view, the difficulty of DGS lies in the implementation of DGS in cryptographic primitives with constrained resources. Besides the resilience against potential side-channel attacks, a designer looking for the optimal DGS solution to a specific application strikes the balance of memory consumption and running time, precision and efficiency.

There are already a variety of works addressing the application of DGS in lattice-based primitives. Existing techniques include the Bernoulli sampler [7], the cumulative distribution table (CDT) sampler [6], the Knuth-Yao sampler [13], and the discrete Ziggurat sampler [16], etc. Rejection sampling can be used to generate discrete Gaussian samples where one draws an element x from a discrete domain uniformly at random and accepts it with probability proportional to $\exp(-x^2/2\sigma^2)$ where σ is the standard deviation. However, calculating the exponential function requires high-precision computing and sufficient trials are needed before the sampler produces an output. In [21], Peikert suggested to perform binary search through CDT and he adapted it to the signature scheme BLISS [7]. At the first step of BLISS, a discrete Gaussian vector is generated to blind the secret. However, the CDT sampling itself takes 35 percent of the total running time of BLISS [12] and the precomputed CDT requires larger memory especially when a wider distribution is in need to improve the security level.

In [10], Hülsing et al. replaced the discrete Gaussian distribution in Lyubashevsky's signature scheme and BLISS by a rounded Gaussian distribution and proved that these schemes were safe. As the term suggested, a rounded Gaussian distribution is obtained by rounding continuous Gaussian samples which can be efficiently realized by Box-Muller transform [4] in constant time. A convolution method, first proposed in [21], can expand a discrete Gaussian distribution with a small parameter to a wider one. A recent sampler design [19] exploits a base sampler with small parameters to efficiently generate DGS with arbitrary and varying parameters in a convolutional fashion. This application-independent algorithm consists of an online and offline stage, both of which can be carried out in constant time, proving a resilience against timing attack. [30] proposed a constant-time sampler and applied Rényi divergence analysis to their algorithm. [27] proposed a DGS algorithm for lattice signatures using arithmetic coding.

Contribution In this work, we propose a novel algorithm for DGS over the integers using polar codes. Polar codes are the first class of efficiently encodable and decodable codes which provably achieve Shannon's entropy bound in source coding and channel capacity in channel coding, respectively [1, 2]. The power of polar codes stems from the polarization phenomenon: under Arikan's polar transform, information measures of synthesized sources (channels) converge to

either 0 or 1 when coding becomes trivial. Moreover, both encoding and decoding run with $O(N \log N)$ complexity, where N denotes the length of a polar code. Given their attractive performance, polar codes have found a wide range of applications in information theory and communication systems. In particular, they have been standardized for the upcoming fifth-generation (5G) wireless communication networks.

This work tackles the sampling problem from a source coding perspective, namely, sampling can be considered the inverse problem of source coding. In source coding or data compression, one typically encode a block of symbols of a certain distribution into some bits which become uniformly random as the block-length tends to infinity [5]. Since a source code is invertible, inverting this process would produce samples from the desired distribution. When a large number of independent Gaussian samples are required in cryptographic applications, polar codes are well suited because in this case the information source of distribution $D_{\mathbb{Z}^N,c,s}$ is memoryless. Obviously, this technique is not restricted to sampling from the discrete Gaussian distribution, but can be extended to other distributions of interest in cryptography.

The principal contributions of this paper are summarized as follows:

- A novel approach to sample from discrete Gaussian distribution over the integers with multilevel polar codes. Using a binary partition tree, we recursively partition $\mathbb{Z}$ into 2 cosets, 4 cosets, and so on. The number of levels is only logarithmic in s. Each level gives rise to a binary source, which is compressed by a binary polar code. The advantage of this multilevel coding approach is that only binary codes are needed, which allow simpler implementation than nonbinary codes.
- Analysis of approximation errors. Although inverting this multilevel polar code would produce the desired distribution $D_{\mathbb{Z}^N,c,s}$, it is not exactly so. This is because the bits after compression are not exactly uniformly random, so feeding the inverted source code with uniformly random bits will only yield an approximate version of the desired distribution. We derive upper bounds on the closeness of the target discrete Gaussian and its approximation by our polar sampler, in SD and KL divergence.
- Security analysis. To achieve a certain security level in a standard cryptographic scheme with oracle access to a discrete Gaussian distribution, the principle of setting the parameters of our polar sampler is also discussed. In cryptographic applications where the number of queries q to the Gaussian sampler is limited (e.g., $q \leq 2^{64}$ in the NIST specifications of signatures), it is well known that using Rényi divergence yields considerable savings. We also apply Rényi divergence to analyze the security level associated with the proposed polar sampler.

The proposed polar sampler complements and distinguishes from existing discrete Gaussian samplers in the literature. In addition to offering a different approach, it exhibits several salient features:

- Information theoretic optimality. Asymptotically, the polar sampler achieves the entropy bound of the discrete Gaussian distribution. This implies

that it requires minimum resources of random bits to produce the desired distribution.
- Quasi-linear complexity. The multilevel approach to Gaussian sampling enjoys low complexity. Both design and implementation require quasi-linear complexity $O(N \log N)$ in storage and running time. The design can be done at the offline stage, that is, given a target distribution, it is done once and for all. The online stage of implementation requires certain a posteriori probabilities, which can be computed in a successive manner. The polar transform itself can be implemented efficiently in $O(N \log N)$ complexity.
- Constant-time implementation. The polar sampler also admits constant-time implementation, since a polar code has a fixed-length. This compares favorably with other source coding techniques such as Huffman coding whose codewords have variable lengths. Moreover, all the computations required run in constant time. This makes our polar sampler very attractive when dealing with side-channel (e.g., timing) attacks.

Of course, the proposed sampler can be combined with existing "expander" techniques such as convolution if needed. In this paper, we focus on the theoretic design and analysis of polar samplers, whereas various optimization issues (e.g., concrete computational/storage costs, finite precision etc.) are left to future work. Nevertheless, we have found it in experiments that even a prototype implementation significantly outperforms benchmark Knuth-Yao sampling in speed.

Roadmap The roadmap of this paper is given as follows. Section 2 introduces the preliminaries of polar source coding and elucidates its relation to sampling. Section 3 presents our polar sampler in detail. Section 4 gives an analysis on the approximation error of our sampler based on both SD and KL divergence. In Section 5, the security level of our sampler is analysed based on SD and KL divergence, respectively. Section 6 compares the polar sampler with Knuth-Yao regarding the complexity. Section 7 concludes this paper and gives some suggestions on future work.

2 Source Coding versus Sampling

2.1 Notation

We use the notation $x^{1:N}$ as shorthand for a row vector $(x^{(1)}, \cdots, x^{(N)})$ of which the i-th entry is denoted by $x^{(i)}$. Given a vector $x^{1:N}$ and a set $\mathcal{A} \subset \{1, \cdots, N\}$, $x_{\mathcal{A}}$ denotes the subvector of $x^{1:N}$ indexed by $\mathcal{A}$. Capital letters such as U and X are used to denote variables while lowercase letters such as u and x represent a realization of the corresponding variable. Denote by $X \sim P$ a distribution P of X over a countable set $\mathcal{X}$. Then the entropy of X is defined as $H_P(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$. We write $H(X) = H_P(X)$ for brevity if the distribution is clear. Suppose X and Y have a joint distribution $P(X, Y)$. The conditional entropy of X given Y is defined as $H(X|Y) = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log \frac{p(y)}{p(x,y)}$. The logarithm to base 2 is denoted by log while the natural logarithm is denoted by ln.

2.2 Source Polarization

The key idea of polar source coding can be found in [2] where a polar code was proposed to achieve Shannon's source coding bound. Let $(X^{1:N}, Y^{1:N})$ denotes N i.i.d. copies of a memoryless source $(X, Y) \sim P_{X,Y}$, where X takes values over $\mathcal{X} = \{0, 1\}$ while Y takes values over a countable set $\mathcal{Y}$. The two random source X and Y are correlated, and Y is called the side-information[1]. In source coding, the encoder compresses a sequence $X^{1:N}$ into a shorter codeword, such that the decoder can produce an estimation $\hat{X}^{1:N}$ of $X^{1:N}$ using the codeword side information $Y^{1:N}$.

Polar codes are proved to achieve Shannon's source coding bound asymptotically. The source polarization transform from $X^{1:N}$ to $U^{1:N}$ is performed by applying an entropy-preserving circuit to $X^{1:N}$, i.e.,

$$U^{1:N} = X^{1:N} G_N$$

and

$$G_N = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^{\otimes n} B_N,$$

where $^{\otimes n}$ denotes the n-th Kronecker power, and B_N is a bit-reversal permutation [1] of the input vector. Fig. 1 illustrates the source polarization transforms of $X^{1:2}$ and $X^{1:4}$ where $\oplus$ denotes mod-2 sum. This transform preserves the entropy in the sense that

$$H(U^{1:2} \mid Y^{1:2}) = 2H(X \mid Y), \;\; H(U^{1:4} \mid Y^{1:4}) = 4H(X \mid Y).$$

Meanwhile, it also polarizes the entropy in the sense that

$$H(U^{(1)} \mid Y^{1:4}) \geq H(S^{(1)} \mid Y^{1:2}) = H(S^{(2)} \mid Y^{3:4}) \geq H(U^{(2)} \mid Y^{1:4}, U^{(1)}),$$

and

$$\begin{aligned} H(U^{(3)} \mid Y^{1:4}, U^{1:2}) \geq & H(R^{(1)} \mid Y^{1:2}, S^{(1)}) \\ = & H(R^{(2)} \mid Y^{3:4}, S^{(2)}) \geq H(U^{(4)} \mid Y^{1:4}, U^{1:3}). \end{aligned}$$

By applying the construction in Fig. 1 recursively, we can obtain the induced joint distribution of $(U^{1:N}, Y^{1:N})$ as

$$\begin{aligned} P_{U^{1:N},Y^{1:N}} &= \sum_{x^{1:N}} P_{X^{1:N},Y^{1:N}}(x^{1:N}, y^{1:N}) \cdot \mathbb{I}\{u^{1:N} = x^{1:N} G_N\} \\ &= \sum_{x^{1:N}} \Big(\prod_{i=1}^{N} P_{X,Y}(x^{(i)}, y^{(i)})\Big) \cdot \mathbb{I}\{u^{1:N} = x^{1:N} G_N\}, \end{aligned} \tag{1}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function. By computing the conditional entropy $H(U^{(i)} \mid Y^{1:N}, U^{1:i-1})$, we have the following source polarization theorem.

[1] Note that even if there is only one source in the context of this paper, the proposed multilevel code still needs side information (which is basically information coming from lower levels.)

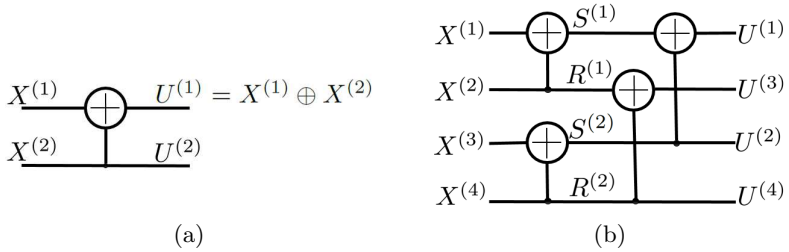


Fig. 1. The source polarization transform [1]: (a) A two-by-two transform (b) A four-by-four transform.

Theorem 1 (Source Polarization [2]). *Let (X, Y) be a source as above. For any $N = 2^n$,$n \geq 1$, let $U^{1:N} = X^{1:N}G_N$. Then, for any $0 < \beta < 0.5$, as $N \to \infty$,*

$$\frac{\left|\left\{i \in [1, N] : H(U^{(i)} \mid Y^{1:N}, U^{1:i-1}) \in (1 - 2^{-N^\beta}, 1]\right\}\right|}{N} \to H(X \mid Y) \quad (2)$$

and

$$\frac{\left|\left\{i \in [1, N] : H(U^{(i)} \mid Y^{1:N}, U^{1:i-1}) \in [0, 2^{-N^\beta})\right\}\right|}{N} \to 1 - H(X \mid Y). \quad (3)$$

Note that in the absence of side information Y, the above theorem still holds by considering Y independent of X.

Definition 1 (Bhattacharyya Parameter [11]). *Let $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ be a pair of random variables where $\mathcal{X} = \{0, 1\} = GF(2)$ and $\mathcal{Y}$ is an arbitrary finite set. Let X and Y follow the joint distribution $P_{XY}(x, y)$. If X is the source to be compressed and Y is the side information, the Bhattacharyya parameter is defined as*

$$\begin{aligned} Z(X|Y) &\equiv 2\sum_y P_Y(y)\sqrt{P_{X|Y}(0|y)P_{X|Y}(1|y)} \quad (4) \\ &= 2\sum_y \sqrt{P_{X,Y}(0, y)P_{X,Y}(1, y)}. \end{aligned}$$

Proposition 1 ([2], Proposition 2).

$$Z(X|Y)^2 \leq H(X|Y) \quad (5)$$
$$H(X|Y) \leq \log(1 + Z(X|Y)). \quad (6)$$

It is implied by Proposition 1 that for a source (X, Y), the parameters $H(U^{(i)} \mid Y^{1:N}, U^{1:i-1})$ and $Z(U^{(i)} \mid Y^{1:N}, U^{1:i-1})$ polarize simultaneously in the sense that $H(U^{(i)} \mid Y^{1:N}, U^{1:i-1})$ approaches 0 (resp. 1) as $Z(U^{(i)} \mid Y^{1:N}, U^{1:i-1})$ approaches 0 (resp. 1).

For $\beta \in (0, 1/2)$ and $\alpha = 2^{-N^\beta}$, the indexes of $U^{1:N}$ can be divided into a low-entropy set

$$\mathcal{L}_{X|Y} = \left\{ i \in [N] : \ Z(U^{(i)} \mid Y^{1:N}, U^{1:i-1}) \in [0, \alpha) \right\} \tag{7}$$

and its complement ${\mathcal{L}^c}_{X|Y}$. Again, in the absence of side information Y, the two sets are defined in the same way by considering Y independent of X and U.

This gives rise to the encoding and decoding scheme described in [2]. Specifically, for a realization of $(X^{1:N}, Y^{1:N}) = (x^{1:N}, y^{1:N})$, the encoder computes $u^{1:N} = x^{1:N} G_N$ and only shares $u_{\mathcal{L}^c_{X|Y}}$ with the decoder. The compression rate is defined as $R = |\mathcal{L}^c_{X|Y}|/N$. The decoder can obtain an estimate $\hat{u}^{1:N}$ of $u^{1:N}$ in a successive manner as

$$\hat{u}^{(i)} = \begin{cases} u^{(i)}, & \text{if } \ i \in \mathcal{L}^c_{X|Y} \\ 0, & \text{if } \ i \in \mathcal{L}_{X|Y} \text{ and } L_N^{(i)}(y^{1:N}, \hat{u}^{1:i-1}) \geq 1 \\ 1, & \text{if } \ i \in \mathcal{L}_{X|Y} \text{ and } L_N^{(i)}(y^{1:N}, \hat{u}^{1:i-1}) < 1, \end{cases} \tag{8}$$

where $L_N^{(i)}(y^{1:N}, \hat{u}^{1:i-1})$ is called the likelihood ratio (LR) defined by

$$L_N^{(i)}(y^{1:N}, \hat{u}^{1:i-1}) = \frac{P(U^{(i)} = 0 | Y^{1:N} = y^{1:N}, U^{1:i-1} = \hat{u}^{1:i-1})}{P(U^{(i)} = 1 | Y^{1:N} = y^{1:N}, U^{1:i-1} = \hat{u}^{1:i-1})}. \tag{9}$$

The probability of block error measured by $P_e = \Pr(\hat{U}^{1:N} \neq U^{1:N}) = \Pr(\hat{U}_{\mathcal{H}^c_{X|Y}} \neq U_{\mathcal{H}^c_{X|Y}})$ is upper-bounded by

$$P_e \leq \sum_{i \in H^c_{X|Y}(N,R)} Z(U^{(i)} | Y^{1:N}, U^{1:i-1}), \tag{10}$$

which can be further formalized as a theorem as follows.

Theorem 2 (An upper bound on error probability [2]). *For any fixed $R > H(X|Y)$ and $\beta < 0.5$, the probability of error for the above polar source coding method is bounded as $P_e = O(2^{-N^\beta})$.*

It implies that any rate $R > H(X|Y)$ is achievable with a vanishing block error probability for sufficiently large N. As N goes to infinity, the polarization process removes the randomness of the low-entropy set almost surely while the other set becomes random. Additionally, the complexity of polar encoding and decoding are both $O(N \log N)$. More details on the complexity of successive-cancellation (SC) decoding can be found in Appendix A.

2.3 From Source Coding to Sampling

Now consider the sampling problem. To produce the above memoryless source X given side information Y, we further define a high-entropy set[2]

$$\mathcal{H}_{X|Y} = \left\{ i \in [N] : \ Z(U^{(i)} \mid Y^{1:N}, U^{1:i-1}) \in (1 - \alpha, 1] \right\}. \tag{11}$$

[2] In [2], ${\mathcal{L}^c}_{X|Y}$ is called the high-entropy set, which is larger than $\mathcal{H}_{X|Y}$.

According to polar source coding, the $U^{(i)}$ for $i \in \mathcal{H}_{X|Y}$ with very high entropy is approximately uniformly distributed and is approximately independent of both $U^{1:i-1}$ and the side information $Y^{1:N}$, while the $U^{(i)}$ for $i \in \mathcal{L}_{X|Y}$ with very low entropy is almost deterministic. Those $U^{(i)}$ for $i \in \mathcal{H}^c_{X|Y} \backslash \mathcal{L}_{X|Y}$ (i.e., $Z(U^{(i)} \mid Y^{1:N}, U^{1:i-1}) \in [\alpha, 1-\alpha]$) are unpolarized. As N goes to infinity, the fraction of unpolarized indexes vanishes and the fraction of high-entropy indexes approaches the entropy of X given Y according to Theorem 1.

Since the polarization transform is reversible, it is expected to produce an approximation $Q_{X^{1:N}}$ of $P_{X^{1:N}}$ by applying the above transform to $U^{1:N}$, i.e. $X^{1:N} = G_N U^{1:N}$. However, the unpolarized set may not be negligible for finite length N, and should be handled with care. More precisely, those $U^{(i)}$ for $i \in \mathcal{H}^c_{X|Y} \backslash \mathcal{L}_{X|Y}$ are not quite uniform; feeding them with uniform bits may cause non-negligible distortion from the target distribution. Therefore, for sampling, $U^{(i)}$s should follow the distribution:

$$U^{(i)} = \begin{cases} \{0,1\} \sim \text{Bernoulli}(0.5), & \text{if } i \in \mathcal{H}_{X|Y} \\ \arg\max_u P_{U^{(i)}|Y^{1:N},U^{1:i-1}}(u|y^{1:N}, u^{1:i-1}), & \text{if } i \in \mathcal{L}_{X|Y} \end{cases} \tag{12}$$

and

$$U^{(i)} = \begin{cases} 0 \;\; w.p. \; P_{U^{(i)}|Y^{1:N},U^{1:i-1}}(0|y^{1:N}, u^{1:i-1}) \\ 1 \;\; w.p. \; P_{U^{(i)}|Y^{1:N},U^{1:i-1}}(1|y^{1:N}, u^{1:i-1}) \end{cases} \text{if } i \in \mathcal{H}^c_{X|Y} \backslash \mathcal{L}_{X|Y} \quad . \tag{13}$$

Fig. 2 shows the difference between source coding and sampling: although the unpolarized set belongs to the compressed codeword in source coding, its bits should be randomized as in (13) in sampling. Denote by $P_{U^{1:N}}$ (resp. $Q_{U^{1:N}}$) the

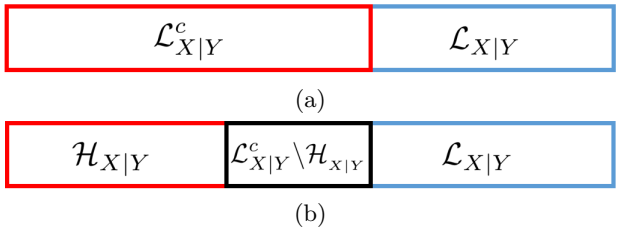


Fig. 2. Polar source coding vs. polar sampling: (a) Subsets of indexes for polar source coding (b) Subsets of indexes for polar sampling. The fraction of $\mathcal{L}^c_{X|Y} \backslash \mathcal{H}_{X|Y}$ vanishes as N goes to infinity.

distribution of $U^{1:N}$ (resp. $U^{1:N}$) induced by the polarization transform, see (1). For some useful metric δ, the distance between $P_{X^{1:N}}$ and $Q_{X^{1:N}}$ is bounded by the data processing inequality

$$\delta(P_{X^{1:N}}, Q_{X^{1:N}}) \leq \delta(P_{U^{1:N}}, Q_{U^{1:N}}).$$

The goodness of a metric sometimes can help reduce the complexity of implementation. To evaluate the performance of our sampling scheme, we will compare different metrics in the sequel, e.g. SD, KL divergence and Rényi divergence.

In order to realize the operations given in (12) and (13), one need to compute $P_{U^{(i)}|Y^{1:N},U^{1:i-1}}$ and define the two sets $\mathcal{H}_{X|Y}$ and $\mathcal{L}_{X|Y}$ by Bhattacharyya parameter $Z(U^{(i)} \mid Y^{1:N}, U^{1:i-1})$. To calculate $P_{U^{(i)}|Y^{1:N},U^{1:i-1}}$ efficiently we can formulate (X, Y) into a binary memoryless symmetric source pair. Denote by W a transition from X to Y with probability $W(y|x) = P_{X,Y}(X, Y)/P_X(x)$. A source pair (X, Y) given as above is said to be symmetric if there exists a permutation $\pi(\cdot)$ such that $W(y|0) = W(\pi(y)|1)$. Some efficient algorithms to find out the two sets $\mathcal{H}_{X|Y}$ and $\mathcal{L}_{X|Y}$ were proposed in [29, 20]. On the other hand, $P_{U^{(i)}|Y^{1:N},U^{1:i-1}}$ can be computed with quasi-linear complexity by SC decoding proposed in [1].

Theorem 3 originally gives the connection between the Bhattacharyya parameters of an asymmetric channel and a symmetric one [11]. Similarly, the Bhattacharyya parameter can be efficiently calculated by the corresponding Bhattacharyya parameter of a binary memoryless symmetric source pair.

Theorem 3 (Connection Between Bhattacharyya Parameters, adapted from [11, 14]). *Denote by (X, Y) a source pair with a joint distribution $P_{X,Y}$ where $X \in \mathcal{X} = \{0, 1\}$ and $Y \in \mathcal{Y}$ for some countable set $\mathcal{Y}$. Let $\tilde{X}$ takes values over $\mathcal{X}$ uniformly and let $\tilde{Y} = (\tilde{X} \oplus X, Y)$. $\tilde{X}$ and $\tilde{Y}$ can naturally be regarded as a BMS source pair with transition probability $\tilde{W}(\tilde{y}|\tilde{x}) = P_{X,Y}(x, y)$ where $\tilde{X} \in \mathcal{X}$ and $\tilde{Y} \in \tilde{\mathcal{Y}} = \mathcal{X} \times \mathcal{Y}$. By performing the above transformation circuit G_N to N i.i.d. copies of X and $\tilde{X}$, i.e., $U^{1:N} = X^{1:N}G_N$ and $\tilde{U}^{1:N} = \tilde{X}^{1:N}G_N$, respectively, we can have two combined transition blocks $W_N : U^{1:N} \to Y^{1:N}$ and $\tilde{W}_N : \tilde{U}^{1:N} \to \tilde{Y}^{1:N}$, respectively. These two combined transitions can be split back into a set of N sub-source pairs $(U^{(i)}, Y^{1:N} \times U^{1:i-1})$ giving rise to N sub-transitions $W_N^{(i)} : \mathcal{X} \to \mathcal{Y}^{1:N} \times \mathcal{X}^{1:i-1}$ and $\tilde{W}_N^{(i)} : \mathcal{X} \to \tilde{\mathcal{Y}}^{1:N} \times \mathcal{X}^{1:i-1}$, respectively. For each sub-transition of W_N and $\tilde{W}_N$, we have the following properties regarding the probability distribution and the Bhattacharyya parameter, i.e.,*

$$P_{U^{1:i},Y^{1:N}}(u^{1:i}, y^{1:N}) = 2^{n-1} P_{\tilde{U}^{1:i-1},\tilde{Y}^{1:N}|\tilde{U}^{(i)}}(u^{1:i-1}, (0, y^{1:N})|u^{(i)}), \tag{14}$$

and

$$Z(U^{(i)}|U^{1:i-1}, Y^{1:N}) = \tilde{Z}(\tilde{U}^{(i)}|\tilde{U}^{1:i-1}, X^{1:N} \oplus \tilde{X}^{1:N}, Y^{1:N}). \tag{15}$$

It is straightforward to see that the Bhattacharyya parameter of the symmetric source pair and asymmetric source pair are the same, and one can calculate $\tilde{Z}(\tilde{U}^{(i)}|\tilde{U}^{1:i-1}, X^{1:N} \oplus \tilde{X}^{1:N}, Y^{1:N})$ with the known technique for symmetric po-

lar codes [29, 20]. Furthermore, we can have

$$\frac{P_{U^{1:i},Y^{1:N}}(u^{1:i-1},0,y^{1:N})}{P_{U^{1:i},Y^{1:N}}(u^{1:i-1},1,y^{1:N})} = \frac{P_{U^{(i)}|Y^{1:N},U^{1:i-1}}(0|y^{1:N},u^{1:i-1})}{P_{U^{(i)}|Y^{1:N},U^{1:i-1}}(1|y^{1:N},u^{1:i-1})} \tag{16}$$

$$= \frac{P_{\tilde{U}^{1:i-1},\tilde{Y}^{1:N}|\tilde{U}^{(i)}}(\tilde{u}^{1:i-1},(0^{1:N},y^{1:N})|0)}{P_{\tilde{U}^{1:i-1},\tilde{Y}^{1:N}|\tilde{U}^{(i)}}(\tilde{u}^{1:i-1},(0^{1:N},y^{1:N})|1)} \tag{17}$$

$$= L_N^{(i)}((0^{1:N},y_1^N),\tilde{u}^{1:i-1}), \tag{18}$$

where $L_N^{(i)}((0^{1:N},y_1^N),\tilde{u}^{1:i-1})$ denotes the posterior probability ratio for a sub-transition $\tilde{W}_N^{(i)}$. Therefore, the posterior probability $P_{U^{(i)}|Y^{1:N},U^{1:i-1}}$ in (12) and (13) can be computed with complexity $O(N\log N)$ by the SC decoding proposed in [1]. In the same fashion, we can compute $P_{U^{(i)}|U^{1:i-1}}$ by considering Y independent of X.

3 Polar Sampling over the Integers

Definition 2. *For any vectors c, x and any $s > 0$, let*

$$\rho_{c,s}(x) = \exp(-\pi\|x-c\|^2/s^2)$$

be a Gaussian function on $\mathbb{R}$ centred at $x = c$ with parameter s. The total mass of the function is $\int_{x\in\mathbb{R}} \rho_{\mathbf{c},s}(x)dx = s$.

Definition 3. *For any $c \in \mathbb{R}$, $s > 0$, define the discrete Gaussian distribution over $\mathbb{Z}$ as*

$$\forall x \in \mathbb{Z},\ D_{\mathbb{Z},c,s}(x) = \rho_{c,s}(x)/\rho_{c,s}(\mathbb{Z})$$

where $\rho_{c,s}(\mathbb{Z}) = \sum_{z\in\mathbb{Z}} \rho_{c,s}(z)$.

In the above definition, the denominator $\rho_{c,s}(\mathbb{Z})$ is for normalization. For convenience, we may omit c for $c = 0$, e.g. $\rho_{0,s}(x) = \rho_s(x)$ and $D_{\mathbb{Z},0,s}(x) = D_{\mathbb{Z},s}(x)$.

Gaussian sampling over integers $\mathbb{Z}$ can be formulated as a multilevel coding problem over a binary partition chain $\mathbb{Z}/2\mathbb{Z}/4\mathbb{Z}/\cdots 2^r\mathbb{Z}/\cdots$ of which each level is labeled by $X_1, X_2, \ldots, X_r, \ldots$ (see Fig. 3). Then the discrete Gaussian distribution over integers $D_{\mathbb{Z},c,s}$ induces a distribution $P_{X_{1:r}}$ whose limit corresponds to $D_{\mathbb{Z},c,s}$ as r goes to infinity. By cutting off the tail of negligible probability, a discrete Gaussian distribution over the integer lattice $\mathbb{Z}$ can be reduced to a distribution over a finite set. An example is $D_{\mathbb{Z},s=3\sqrt{2\pi}}$ for which a constellation of 32 points centred at 0 is somewhat sufficient because the total probability is rather close to 1.

Suppose r levels of partition are employed to approximate $D_{\mathbb{Z},c,s}$. The chain rule of conditional probability and the chain rule of conditional entropy, i.e.

$$P(X_{1:r}) = \prod_{k=1}^{r} P(X_k|X_{1:k-1}), \tag{19}$$

$$H(X_{1:r}) = \sum_{k=1}^{r} H(X_k|X_{1:k-1}), \tag{20}$$

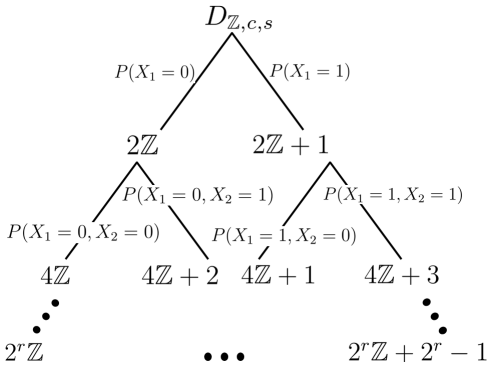


Fig. 3. An r-level binary partition tree of the integer lattice $\mathbb{Z}$.

imply that the Gaussian distribution over the finite constellation can be generated in a level-by-level way. For the k-th level, we can generate the component source X_k by the the scheme proposed in Subsection 2.3 given the samples $x_{1:k-1}$ from lower levels as side information. The sampling problem for each level can be divided into two stages, construction and implementation.

For the first level, we want to generate the component source X_1 in the absence of any side information.

1. *Construction:* By performing the source polarization transformation G_N on N i.i.d. copies of X_1, we obtain an N dimensional vector $U_1^{1:N} = X_1^{1:N} G_N$. For any $\beta \in (0, 1/2)$ and $\alpha = 2^{-N^\beta}$, we formally define two sets $\mathcal{H}_{X_1}$ and $\mathcal{L}_{X_1}$ as
$$\mathcal{H}_{X_1} = \left\{ i \in [N] : \ Z(U_1^{(i)} \mid U_1^{1:i-1}) \in (1-\alpha, 1] \right\} \tag{21}$$
and
$$\mathcal{L}_{X_1} = \left\{ i \in [N] : \ Z(U_1^{(i)} \mid U_1^{1:i-1}) \in [0, \alpha) \right\}. \tag{22}$$
For any $i \in \mathcal{H}_{X_1}$, $U_1^{(i)}$ is approximately uniform and independent of $U_1^{1:i-1}$, while for $i \in \mathcal{L}_{X_1}$, $U_1^{(i)}$ is almost deterministic given the knowledge of $U_1^{1:i-1}$.
2. *Implementation:* After getting the two sets $\mathcal{H}_{X_1}$ and $\mathcal{L}_{X_1}$, we can generate N i.i.d. copies of X_1 by applying the polarization transform circuit to the input vector $U_1^{1:N}$ of which each entry takes a value according to the following rule:
$$U_1^{(i)} = \begin{cases} \text{Bernoulli}(\frac{1}{2}) & \text{if } i \in \mathcal{H}_{X_1} \\ \arg\max_{u_1^{(i)}} P_{U_1^{(i)}|U_1^{1:i-1}}(u_1^{(i)}|u_1^{1:i-1}) & \text{if } i \in \mathcal{L}_{X_1} \end{cases}, \tag{23}$$
and
$$U_1^{(i)} = \begin{cases} 0 \ w.p. \ P_{U_1^{(i)}|U_1^{1:i-1}}(0|u_1^{1:i-1}) \\ 1 \ w.p. \ P_{U_1^{(i)}|U_1^{1:i-1}}(1|u_1^{1:i-1}) \end{cases} \text{if } i \in \mathcal{H}_{X_1}^c \backslash \mathcal{L}_{X_1} . \tag{24}$$

As discussed in Subsection 2.3, by converting the source X_1 without side information to a BMS source pair $(\tilde{X}_1, X_1 \oplus \tilde{X}_1)$, we can calculate the Bhattacharyya parameter $Z(U_1^{(i)} \mid U_1^{1:i-1})$ and the posterior probability $P_{U_1^{(i)}|U_1^{1:i-1}}$ efficiently. Once we have a realization $u_1^{1:N}$ of $U_1^{1:N}$, we can obtain a realization $x_1^{1:N} = u_1^{1:N} G_N$ of $X_1^{1:N}$ and transmit it to the next level for further processing.

For higher levels with $k \in (1, r]$, our task is to generate N i.i.d. copies of source X_k given the side information $x_{1:k-1}^{1:N}$ which were generated at the previous $k-1$ levels.

1. *Construction:* By performing the source polarization transformation G_N on N i.i.d. copies of X_k, we obtain an N dimensional vector $U_k^{1:N} = X_k^{1:N} G_N$. For $\beta \in (0, 1/2)$ and $\alpha = 2^{-N^\beta}$, we define $\mathcal{H}_{X_k|X_{1:k-1}}$ and $\mathcal{L}_{X_k|X_{1:k-1}}$ as

$$\mathcal{H}_{X_k|X_{1:k-1}} = \left\{ i \in [N] : \ Z(U_k^{(i)} \mid X_{1:k-1}^{1:N}, U_k^{1:i-1}) \in (1-\alpha, 1] \right\} \tag{25}$$

and

$$\mathcal{L}_{X_k|X_{1:k-1}} = \left\{ i \in [N] : \ Z(U_k^{(i)} \mid X_{1:k-1}^{1:N}, U_k^{1:i-1}) \in [0, \alpha) \right\}. \tag{26}$$

2. *Implementation:* We can generate N i.i.d. copies of X_k by applying the polarization transformation circuit to the input vector $U_k^{1:N}$ of which each entry takes a value according to the following rule:

$$U_k^{(i)} = \begin{cases} \text{Bernoulli}(\frac{1}{2}) & \text{if } i \in \mathcal{H}_{X_k|X_{1:k-1}} \\ \arg\max_u P_{U^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}(u|x_{1:k-1}^{1:N}, u_k^{1:i-1}) & \text{if } i \in \mathcal{L}_{X_k|X_{1:k-1}} \end{cases} \tag{27}$$

and

$$U_k^{(i)} = \begin{cases} 0 \ w.p. \ P_{U_k^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}(0|x_{1:k-1}^{1:N}, u_k^{1:i-1}) \\ 1 \ w.p. \ P_{U_k^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}(1|x_{1:k-1}^{1:N}, u_k^{1:i-1}) \\ \qquad \text{if } i \in \mathcal{H}^c_{X_k|X_{1:k-1}} \backslash \mathcal{L}_{X_k|X_{1:k-1}}. \end{cases} \tag{28}$$

Once we have a realization $u_k^{1:N}$ of $U_k^{1:N}$, we can obtain a realization $x_k^{1:N} = u_k^{1:N} G_N$ of $X_k^{1:N}$ and transmit it to the next level for further processing. Again, $Z(U_k^{(i)} \mid X_{1:k-1}^{1:N}, U_k^{1:i-1})$ and $P_{U_k^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}$ can be calculated efficiently using Theorem 3. Note that for a target statistical difference between the target distribution and the distribution we generate, the two sets $\mathcal{H}_{X_k|X_{1:k-1}}$ and $\mathcal{L}_{X_k|X_{1:k-1}}$ for each level can be determined offline. By repeating the operations in (27) and (28) from level 2 to level r, we can finally obtain N samples $x^{1:N}$ from $D_{\mathbb{Z},c,s}$, i.e.,

$$x^{1:N} = \sum_{k=1}^{r} 2^{k-1} x_k^{1:N}. \tag{29}$$

Fig. 4 shows how this Gaussian sampler works at each level in terms of construction and implementation. It also shows how to combine the output of each

level. At the construction stage, W designates the probability transition from X_k to $X_{1:k}$. A sub-transition $W_N^{(i)}$ is obtained by combining then spliting N copies of i.i.d. source pair $(X_k, X_{1:k-1})$. At this stage the Bhattacharyya parameters of $W_N^{(i)}$ are calculated to define $\mathcal{H}_{X_k|X_{1:k-1}}$ and $\mathcal{L}_{X_k|X_{1:k-1}}$. At the implementation stage, realizations of $U^{1:N}$ are generated according to the implementation rules (27) and (28). In Appendix C, we give the key functions invoked at the construction and implementation stage. Given the two parameters N and β, the closeness between the target distribution and the one our sampler can produce will be analyzed in the next section.

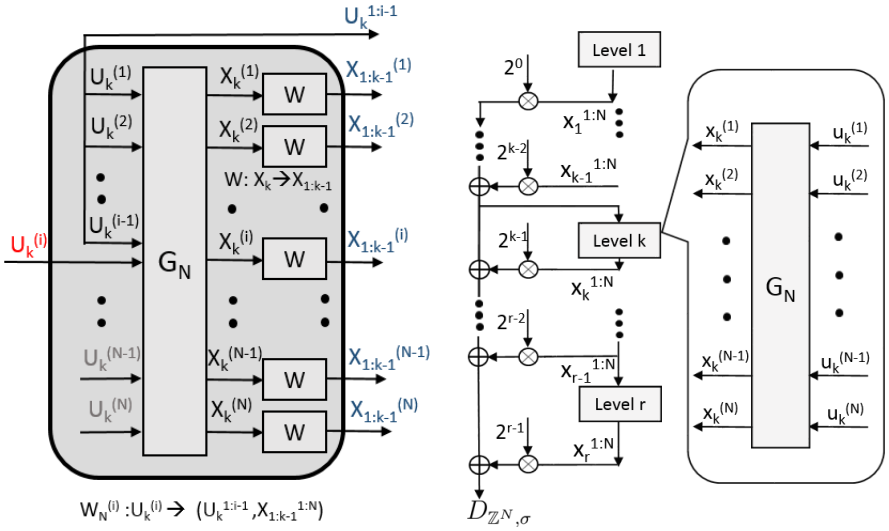


Fig. 4. The construction and implementation of the polar sampler: (a) Construction (can run offline) (b) Implementation (runs online).

4 Closeness Analysis

4.1 Closeness Measures

Definition 4 (Statistical distance). *Denote by P and Q two distributions with countable supports and let $\mathcal{A}$ be the union of supports of P and Q. The SD between P and Q is*

$$\mathbb{V}(P,Q) = \frac{1}{2}\sum_{a\in\mathcal{A}} |P(a) - Q(a)|.$$

Definition 5 (Kullback-Leibler divergence). *Let P and Q be two distributions over a common countable set Ω, and let $\mathcal{A} \subset \Omega$ be the strict support of P ($P(a) > 0$ iff $a \in \mathcal{A}$). The KL divergence D_{KL} of Q from P is defined as:*

$$D_{KL}(P\|Q) = \sum_{a\in\mathcal{A}} P(a) \ln\left(\frac{P(a)}{Q(a)}\right)$$

with the convention that $\ln(x/0) = +\infty$ for any $x > 0$.

4.2 The Approximation Error Model

In a concrete implementation, an ideal DGS is replaced by an approximation. To give a sharp estimation of the accuracy/security of a cryptographic primitive, the closeness between the approximation and the ideal DGS should be measured. Conventionally, the closeness between two distribution P and Q over the same support $\mathcal{A}$ is measured by the SD and KL divergence. In this section, we will derive the upper bounds on the closeness between the ideal DGS and the distribution generated by our sampling scheme in SD and KL divergence, respectively.

The approximation error comes from two sources, the tailcut (finite levels of the partition employed) and the polar source coding. On the one hand, we need to decide how many levels of partition are in need. On the other hand, the error introduced by polar sampling should also be analyzed. Denote by $D_{\mathbb{Z},c,s}$ the target discrete Gaussian distribution and we decide to employ r levels of partition. If polar sampling did not introduce any error, we would generate a distribution $P_{X_{1:r}}$ with a closeness measure $\delta(D_{\mathbb{Z},c,s}, P_{X_{1:r}})$ which is determined only by r for some metric δ. Then, let $Q_{X_{1:r}}(x_{1:r})$ denote the distribution obtained by polar sampling where the error introduced is $\delta(P_{X_{1:r}}, Q_{X_{1:r}})$. By the triangle inequality of SD, we obtain the SD between the ideal discrete Gaussian distribution and ours as follows,

$$\mathbb{V}(D_{\mathbb{Z},c,s}, Q_{X_{1:r}}) \leq \mathbb{V}(D_{\mathbb{Z},c,s}, P_{X_{1:r}}) + \mathbb{V}(P_{X_{1:r}}, Q_{X_{1:r}}). \tag{30}$$

Since the KL divergence does not satisfy the triangle inequality, we will only give $D(D_{\mathbb{Z},c,s}\|P_{X_{1:r}})$ and $D(P_{X_{1:r}}\|Q_{X_{1:r}})$ rather than a total KL divergence $D(D_{\mathbb{Z},c,s}\|Q_{X_{1:r}})$. However, as discussed in [24, Section 3] the lack of symmetry and triangle inequality can be easily handled in KL-based security analysis.

4.3 Approximation Error from Tailcut

Definition 6 (Smoothing Parameter [18]). *For an n-dimensional lattice Λ, and positive real $\epsilon > 0$, we define its smoothing parameter $\eta_\epsilon(\Lambda)$ to be the smallest s such that $\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \epsilon$.*

The smoothing parameter quantifies how large s is sufficient such that $D_{\Lambda,c,s}$ behaves like the continuous Gaussian distribution. It is implied by Definition 6 that for any $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\mathbb{Z})$ of $\mathbb{Z}$ is the smallest s such that $\rho(s\mathbb{Z}) \leq 1 + \epsilon$.

Lemma 1 (Lemma 4.2, [9]). *For any $\epsilon > 0$, any $s > \eta_\epsilon(\mathbb{Z})$, and any $t > 0$,*

$$\Pr_{x \leftarrow D_{\mathbb{Z},c,s}}(|x - c| \geq t \cdot s) \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2e^{-\pi t^2}.$$

Instead of sampling over the full domain of the integer lattice, a distribution tail of negligible probability is cut off in practice. Suppose 2^r samples are left after the tailcut. Let $\mathcal{A} = \mathbb{Z} \cap [-2^{r-1} + c, 2^{r-1} + c)$. The distribution of the finite set is

$$\begin{aligned} D_\gamma(a) &= \frac{\rho_{c,s}(a)}{\sum_{a \in \mathcal{A}} \rho_{c,s}(\mathbb{Z})} \\ &= D_{\mathbb{Z},c,s}(a) / D_{\mathbb{Z},c,s}(\mathcal{A}), \end{aligned}$$

where γ is the probability of the tail. This constellation $\mathcal{A}$ of 2^r points can be represented as a binary partition tree labeled by $X_{1:r}$ in the same way as Fig. 3. In our sampling scheme, we obtain a sample labeled by

$$x^{1:N} = \sum_{k=1}^{r} 2^{k-1} x_k^{1:N}.$$

There exists a one-to-one mapping from $X_{1:r}$ to $\mathcal{A}$. Therefore $P_{X_{1:r}}$ and the tailcut distribution D_γ are exactly the same and we can obtain $\mathbb{V}(D_{\mathbb{Z},c,s}, P_{X_{1:r}})$ (resp. $D_{KL}(D_{\mathbb{Z},c,s} \| P_{X_{1:r}})$) by calculating $\mathbb{V}(D_{\mathbb{Z},c,s}, D_\gamma)$ (resp. $D_{KL}(D_{\mathbb{Z},c,s} \| D_\gamma)$).

SD-Based Analysis Given the tailcut distribution D_γ over the finite constellation $\mathcal{A}$ as above, the SD between $D_{\mathbb{Z},c,s}$ and D_γ is

$$\begin{aligned} \mathbb{V}(D_{\mathbb{Z},c,s}, D_\gamma) &= \frac{1}{2} \sum_{a \notin \mathcal{A}} |D_{\mathbb{Z},c,s}(a) - 0| + \frac{1}{2} \sum_{a \in \mathcal{A}} |D_{\mathbb{Z},c,s}(a) - D_\gamma(a)| \\ &= \frac{1}{2}\gamma + \frac{1}{2} \sum_{a \in \mathcal{A}} \left| D_{\mathbb{Z},c,s}(a) \left(1 - \frac{1}{D_\gamma(\mathcal{A})}\right) \right| \\ &= \frac{1}{2}\gamma + \frac{1}{2}(1-\gamma) \left| 1 - \frac{1}{D_\gamma(\mathcal{A})} \right| \\ &= \frac{1}{2}\gamma + \frac{1}{2}(1-\gamma) \left| 1 - \frac{1}{1-\gamma} \right| \\ &= \gamma. \end{aligned}$$

By Lemma 1, the SD between $D_{\mathbb{Z},c,s}$ and $P_{X_{1:r}}$ is bounded as

$$\begin{aligned} \mathbb{V}(D_{\mathbb{Z},c,s}, P_{X_{1:r}}) &= \mathbb{V}(D_{\mathbb{Z},c,s}, D_\gamma) \\ &\leq \frac{1+\epsilon}{1-\epsilon} \cdot 2e^{-\pi t^2}, \end{aligned} \tag{31}$$

for any $\epsilon > 0$, $s > \eta_\epsilon(\mathbb{Z})$ and $t \cdot s = 2^{r-1}$.

KL-Based Analysis Given the one-to-one mapping between $\mathcal{A}$ and $X_{1:r}$ and the tailcut setting as above, D_γ over the finite constellation $\mathcal{A}$ can be written in the form

$$\begin{aligned} P(X_{1:r} = x) &= D_{\mathbb{Z},c,s}(a) / \sum_{a\in\mathcal{A}} D_{\mathbb{Z},c,s}(a) \\ &= D_{\mathbb{Z},c,s}(a|a \in \mathcal{A}). \end{aligned}$$

The KL divergence between D_γ and $D_{\mathbb{Z},c,s}$ is

$$\begin{aligned} D_{KL}(D_\gamma \| D_{\mathbb{Z},c,s}) &= \sum_{a\in\mathcal{A}} D_{\mathbb{Z},c,s}(a|a\in\mathcal{A}) \ln \frac{D_{\mathbb{Z},c,s}(a|a\in\mathcal{A})}{D_{\mathbb{Z},c,s}(a)} \\ &= \sum_{a\in\mathcal{A}} D_{\mathbb{Z},c,s}(a|a\in\mathcal{A}) \ln \frac{D_{\mathbb{Z},c,s}(a|a\in\mathcal{A})}{D_{\mathbb{Z},c,s}(a|a\in\mathcal{A}) D_{\mathbb{Z},c,s}(x\in\mathcal{A})} \\ &= \sum_{a\in\mathcal{A}} D_{\mathbb{Z},c,s}(a|a\in\mathcal{A}) \ln \frac{1}{D_{\mathbb{Z},c,s}(a\in\mathcal{A})} \\ &= \ln \frac{1}{D_{\mathbb{Z},c,s}(a\in\mathcal{A})}. \end{aligned}$$

According to the second-order Taylor bound, if $D_{\mathbb{Z},c,s}(x \in \mathcal{A}) = 1-\gamma$ for any $0 < \gamma < 1$, $D_{KL}(D_\gamma \| D_{\mathbb{Z},c,s})$ is bounded as

$$\begin{aligned} D_{KL}(D_\gamma \| D_{\mathbb{Z},c,s}) &= \gamma + O(\gamma^2) \\ &\approx \mathbb{V}(P_{X_{1:r}}, D_{\mathbb{Z},c,s}). \end{aligned} \tag{32}$$

and so is $D_{KL}(P_{X_{1:r}} \| D_{\mathbb{Z},c,s})$.

4.4 Approximation Error from Polar Sampling

KL-Based Error Analysis Let $P_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ denote the distribution of N i.i.d. $X_{1:r}$ defined as above. For any $0 < \beta < 0.5$, $N = 2^n, n \geq 1$ and the corresponding high and low-entropy sets defined in (25) and (26), one can generate a distribution $Q_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ using the rules (27) and (28). To give the KL divergence between $P_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ and $Q_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$, we first modify the implementation rules (27) and (28) to

$$U_k^{(i)} = \begin{cases} 0 \ w.p. \ \frac{1}{2} \\ 1 \ w.p. \ \frac{1}{2} \end{cases} \quad \text{if } i \in \mathcal{H}_{X_k|X_{1:k-1}} \tag{33}$$

and

$$U_k^{(i)} = \begin{cases} 0 \ w.p. \ P_{U_k^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}(0|x_{1:k-1}^{1:N}, u_k^{1:i-1}) \\ 1 \ w.p. \ P_{U_k^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}(1|x_{1:k-1}^{1:N}, u_k^{1:i-1}) \end{cases} \quad \text{if } i \in \mathcal{H}^c{}_{X_k|X_{1:k-1}}, \tag{34}$$

where only the deterministic decisions for $U_k^{(i)}$ in $\mathcal{L}_{X_k|X_{1:k-1}}$ are replaced by random decisions. Let $Q'_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ denote the distribution obtained from the implementation rule described in (33) and (34). The KL divergence between $P_{X_{1:r}^{1:N}}$ and $Q'_{X_{1:r}^{1:N}}$ is given as follows,

$$\begin{aligned}
&D_{KL}(P_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}}) \\
&\overset{(a)}{=} D_{KL}(P_{U_{1:r}^{1:N}} \| Q'_{U_{1:r}^{1:N}}) \\
&\overset{(b)}{=} D_{KL}(P_{U_1^{1:N}} P_{U_2^{1:N}|U_1^{1:N}} \cdots P_{U_r^{1:N}|U_{1:r-1}^{1:N}} \| Q'_{U_1^{1:N}} Q'_{U_2^{1:N}|U_1^{1:N}} \cdots Q'_{U_r^{1:N}|U_{1:r-1}^{1:N}}) \\
&\overset{(c)}{=} D_{KL}(P_{U_1^{1:N}} \| Q'_{U_1^{1:N}}) + D_{KL}(P_{U_1^{1:N}|U_1^{1:N}} \| Q'_{U_2^{1:N}|U_1^{1:N}}) + \cdots \\
&\qquad\qquad + D_{KL}(P_{U_r^{1:N}|U_{1:r-1}^{1:N}} \| Q'_{U_r^{1:N}|U_{1:r-1}^{1:N}})
\end{aligned} \tag{35}$$

where the equalities and inequalities are due to

(a) One-to-one mapping from $X_{1:r}^{1:N}$ to $U_{1:r}^{1:N}$;

(b) The Chain rule of joint distribution;

(c) The chain rule of KL divergence.

For any level $k \in \{1, \cdots r\}$, $D_{KL}(P_{X_k^{1:N}|X_{1:k-1}^{1:N}} \| Q'_{X_k^{1:N}|X_{1:k-1}^{1:N}})$ is bounded as follows,

$$\begin{aligned}
&D_{KL}(P_{X_k^{1:N}|X_{1:k-1}^{1:N}} \| Q'_{X_k^{1:N}|X_{1:k-1}^{1:N}}) \\
&\overset{(d)}{=} \sum_{i=1}^{N} D_{KL}(P_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}} \| Q'_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}}) \\
&\overset{(e)}{=} \sum_{i\in\mathcal{H}_k} D_{KL}(P_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}} \| Q'_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}}) \\
&\overset{(f)}{=} \sum_{i\in\mathcal{H}_k} \ln 2 \left[1 - H_P(U_k^{(i)}|U_k^{1:i-1}, U_{1:k}^{1:N})\right] \\
&\overset{(g)}{\leq} \sum_{i\in\mathcal{H}_k} \ln 2 \left[1 - Z_P(U_k^{(i)}|U_k^{1:i-1}, U_{1:k}^{1:N})^2\right] \\
&\overset{(h)}{\leq} 2\ln 2 \cdot N 2^{-N^\beta},
\end{aligned} \tag{36}$$

where the equalities and inequalities are due to

(d) The chain rule of KL divergence;

(e) For $i \in \mathcal{H}^c_{X_k|X_{1:k-1}}$, $Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) = P(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})$;

(f) The definition of $D_{KL}(\cdot\|\cdot)$ and $Q'(U_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) = \frac{1}{2}$ for $i \in \mathcal{H}_{X_k|X_{1:k-1}}$;

(g) $Z(X|Y)^2 < H(X|Y) < Z(X|Y)$ [2];

(h) (25).

According to (35) and (36), the KL divergence between $P_{X_{1:r}^{1:N}}$ and $Q'_{X_{1:r}^{1:N}}$ is bounded as

$$D_{KL}(P_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}}) \leq 2 \ln 2 \cdot rN2^{-N^\beta}.$$

In a similar fashion, the KL divergence between $Q_{X_{1:r}^{1:N}}$ and $Q'_{X_{1:r}^{1:N}}$ is given as follows,

$$\begin{aligned}
&D_{KL}(Q_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}}) \\
&= D_{KL}(Q_{U_{1:r}^{1:N}} \| Q'_{U_{1:r}^{1:N}}) \\
&= \sum_{k=1}^{r} \sum_{i=1}^{N} D_{KL}(Q_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}} \| Q'_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}}) \\
&\overset{(i)}{=} \sum_{k=1}^{r} \sum_{i \in \mathcal{L}_{X_k|X_{1:k-1}}} D_{KL}(Q_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}} \| Q'_{U_k^{(i)}|U_k^{1:i-1},U_{1:k}^{1:N}}) \\
&\overset{(j)}{=} \sum_{k=1}^{r} \sum_{i \in \mathcal{L}_{X_k|X_{1:k-1}}} \ln 2 \sum_{u_k^{1:i-1},u_{1:k-1}^{1:N}} -Q'(u_k^{1:i-1}, u_{1:k-1}^{1:N}) \log Q'(\bar{u}_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N}) \\
&\overset{(k)}{\leq} \sum_{k=1}^{r} \sum_{i \in \mathcal{L}_{X_k|X_{1:k-1}}} \ln 2 \cdot H_P(U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N}) \\
&\overset{(l)}{\leq} \sum_{k=1}^{r} \sum_{i \in \mathcal{L}_{X_k|X_{1:k-1}}} \ln 2 \cdot Z(U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N}) \\
&\overset{(m)}{\leq} \ln 2 \cdot rN2^{-N^\beta},
\end{aligned}$$

where the equalities and inequalities come from

(i) For $i \in \mathcal{L}^c_{X_k|X_{1:k-1}}$, $Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) = Q(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})$;
(j) The definition of $D_{KL}(\cdot\|\cdot)$ (see Appendix B);
(k) See Appendix B;
(l) $Z(X|Y)^2 < H(X|Y) < Z(X|Y)$ [2];
(m) (26).

If m samples are picked randomly from all the N samples produced by our sampler, we can bound the KL divergence of the m-dimensional distributions by sub-additivity as follows,

$$D_{KL}(P_{X_{1:r}^{1:m}} \| Q'_{X_{1:r}^{1:m}}) \leq \ln 2 \cdot r2N2^{-N^\beta} \text{and } \; D_{KL}(Q_{X_{1:r}^{1:m}} \| Q'_{X_{1:r}^{1:m}}) \leq \ln 2 \cdot rN2^{-N^\beta},$$

for any $m \leq N$. Although we cannot give the KL divergence between $P_{X_{1:r}^{1:N}}$ and $Q_{X_{1:r}^{1:N}}$ due to the lack of triangle inequality, the absence of $D_{KL}(P_{X_{1:r}^{1:N}} \| Q_{X_{1:r}^{1:N}})$ will not prevent us from the security analysis which will be explained in the sequel.

SD-Based Error Analysis

Theorem 4 (Polar Sampling Theorem). *Let $P_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ denote the distribution of N i.i.d. $X_{1:r}$ defined as above. For any $0 < \beta < 0.5$, $N = 2^n, n \geq 1$ and the corresponding high and low-entropy sets defined in (25) and (26), one can generate a distribution $Q_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ using the rules (27) and (28), and the SD between $P_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ and $Q_{X_{1:r}^{1:N}}(x_{1:r}^{1:N})$ is upper bounded as*

$$\mathbb{V}(Q_{X_{1:r}^{1:N}}, P_{X_{1:r}^{1:N}}) \leq (\sqrt{2}+1)\sqrt{\frac{1}{2}\ln 2 \cdot rN2^{-N^\beta}}.$$

Proof. From the definition of Q' given in KL-based analysis, it is straightforward to obtain the SD between $\mathbb{V}(P_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}}) \leq r2N2^{-N^\beta}$ by Pinsker's inequality as follows,

$$\begin{aligned}\mathbb{V}(P_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}}) &\leq \sqrt{\left(\frac{1}{2}\ln 2\right) D_{KL}(P_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}})} \\ &\leq \sqrt{\ln 2 \cdot rN2^{-N^\beta}}.\end{aligned}$$

In a similar fashion, we obtain

$$\begin{aligned}\mathbb{V}(Q_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}}) &\leq \sqrt{\left(\frac{1}{2}\ln 2\right) D_{KL}(Q_{X_{1:r}^{1:N}} \| Q'_{X_{1:r}^{1:N}})} \\ &\leq (\sqrt{2}+1)\sqrt{\frac{1}{2}\ln 2 \cdot rN2^{-N^\beta}}.\end{aligned}$$

Since SD satisfies the triangle inequality and symmetry property, the SD between $P_{X_{1:r}^{1:N}}$ and $Q_{X_{1:r}^{1:N}}$ is bounded as

$$\mathbb{V}(Q_{X_{1:r}^{1:N}}, P_{X_{1:r}^{1:N}}) \leq (\sqrt{2}+1)\sqrt{\frac{1}{2}\ln 2 \cdot rN2^{-N^\beta}} \tag{37}$$

It can be observed from (30), (31), (37) that for sufficiently large N and properly chosen β and r, the distribution $Q_{X_{1:r}^{1:N}}$ generated by our sampler can well approximate $D_{\mathbb{Z}^{1:N},c,s}$. Usually, a cryptographic scheme needs many more than one sample at each run. For example, in the context of Ring LWE-based encryption, each query to the oracle gives m discrete Gaussian samples where m is the dimension of the ring. If m samples are picked arbitrarily from all the N samples generated by our sampler, we can bound the SD of the two m-dimensional distributions by sub-additivity as:

$$\mathbb{V}(Q_{X_{1:r}^{1:m}}, P_{X_{1:r}^{1:m}}) \leq (\sqrt{2}+1)\sqrt{\frac{1}{2}\ln 2 \cdot rN2^{-N^\beta}}.$$

5 Security Analysis and Parameter Selection

Definition 7 (Standard cryptographic scheme [19]). *We consider an arbitrary cryptographic scheme S, consisting of one or more algorithms with oracle access to a probability distribution ensemble $\mathcal{P}_\theta$, and whose security against an adversary A (also consisting of one or more algorithms) is defined in terms of a game $G^{\mathcal{P}}_{S,A}$ defining the event that A succeed in breaking the scheme S. The success probability of A against S (when using samples from $\mathcal{P}_\theta$) is defined as $\epsilon^{\mathcal{P}}_A = Pr\{G^{\mathcal{P}}_{S,A}\}$. The cost of an attack A against S is defined as $t_A/\epsilon^{\mathcal{P}}_A$, and that the bit-security of S is the minimum (over all adversaries A) of $\log(t_A/\epsilon^{\mathcal{P}}_A)$.*

Lemma 2 ([19]). *Let $S^{\mathcal{P}}$ be a standard cryptographic scheme as in Definition 7 with a black-box access to a probability distribution ensemble $\mathcal{P}_\theta$, and δ any cryptographically useful measure. If $S^{\mathcal{P}}$ is λ-bit secure and $\delta(\mathcal{P}_\theta, \mathcal{Q}_\theta) \leq 2^{-\lambda}$, then $S^{\mathcal{Q}}$ is $(\lambda-1)$-bit secure.*

Lemma 2 illustrates how the security with respect to an ideal distribution is related to the security with respect to an approximated distribution. To approximate a λ-bit secure scheme $S^{\mathcal{P}}$ with a $(\lambda-1)$-bit secure scheme $S^{\mathcal{Q}}$, one need to guarantee that $\delta(\mathcal{P}_\theta, \mathcal{Q}_\theta) \leq 2^{-\lambda}$ for some useful metrics e.g. SD. Suppose $S^{\mathcal{D}_{\mathbb{Z}^m,c,s}}$ guarantees a security level of $\lambda+1$, and we expect to achieve λ-bit security for $S^{\mathcal{Q}_{X^{1:m}_{1:r}}}$ where $Q_{X^{1:m}_{1:r}}$ is an approximation of $D_{\mathbb{Z}^m,c,s}$.

5.1 Security Analysis with SD

As analyzed above, the SD between the ideal DGS and the one we generate vanishes exponentially with block-length N for any $\beta \in (0, 0.5)$ and appropriate choice of r. In implementation, we should choose N, β and r properly to achieve desired closeness due to the security requirement and the limitation on resources. Moreover, the closeness determines the security level of a cryptographic primitive in which an approximated distribution is applied instead of the ideal one. Before giving the relation between the SD and security, we first introduce the standard cryptographic scheme to which the SD can apply.

According to equation (30), the overall SD between $D_{\mathbb{Z}^m,c,s}$ and $Q_{X^{1:m}_{1:r}}$ breaks down into $\mathbb{V}(D_{\mathbb{Z}^m,c,s}, P_{X^{1:m}_{1:r}})$ caused by tailcut and $\mathbb{V}(P_{X^{1:m}_{1:r}}, Q_{X_{1:r}1:m})$ caused by polar coding. Lemma 2 suggests that $\mathbb{V}(D_{\mathbb{Z}^m,c,s}, Q_{X^{1:m}_{1:r}})$ should be no larger than $2^{-(\lambda+1)}$, therefore both $\mathbb{V}(D_{\mathbb{Z}^m,c,s}, P_{X^{1:m}_{1:r}})$ and $\mathbb{V}(P_{X^{1:m}_{1:r}}, Q_{X^{1:m}_{1:r}})$ should be no larger than $2^{-(\lambda+2)}$. Moreover, Lemma 1 implies that a tailcut of $D_{\mathbb{Z}^m,c,s}$ with support $\mathbb{Z} \cap [c - ts,\ c + ts]$ can approximate $D_{\mathbb{Z}^m,c,s}$ with $\mathbb{V}(D_{\mathbb{Z}^{1:m},c,s}, P_{X^{1:m}_{1:r}}) \leq 2^{-(\lambda+2)}$ if $t \geq \sqrt{(\lambda + \log m + 3)\ln 2/\pi}$ for $s > \eta_\epsilon(\mathbb{Z})$ with only a small loss in tightness; accordingly the number of levels r should be no less than $\left\lceil \log(2s\sqrt{(\lambda + \log m + 3)\ln 2/\pi}) \right\rceil$. By Theorem 4, the SD between $P^{1:m}_{X_1:r}$ and $Q^{1:m}_{X_1:r}$ is bounded as

$$\mathbb{V}(P_{X^{1:m}_{1:r}}, Q_{X^{1:m}_{1:r}}) \leq 2^{-2^{n\beta-1}+\frac{1}{2}n+\frac{1}{2}\log r+1}.$$

for $m \leq N$ and $N = 2^n$.

At the cost of some tightness, we can guarantee that $\mathbb{V}(P_{X_{1:r}^{1:m}}, Q_{X_{1:r}^{1:m}}) \leq 2^{-(\lambda+1)}$ if $2^{-2^{n\beta-1}+\frac{1}{2}n+\frac{1}{2}\log r+1} \leq 2^{-(\lambda+2)}$ for $\beta \in (0, 0.5)$. The number of levels r is determined once s and t is fixed, while n and β should be chosen properly to minimize the requirement for memory and running time subject to the security requirement.

As mentioned earlier, our algorithm consists of two stages: construction and implementation. The former is done offline and the latter runs online. At the implementation stage, the running time to produce one bit $U_k^{(i)}$ varies with respect to different sets due to different implementation rules according to which $U_k^{(i)}$ is generated either randomly or deterministically. The parameter β affects the running time to produce $U_k^{1:N}$ by defining the high and low entropy sets. n is obviously in a more dominant position because it greatly influences the two stages with respect to the efficiency of both computational and memory costs. Given multiple parameter options to achieve a target security level, we put the one with a smaller n (or N) as a first priority for the sake of efficiency. Fig. 5 illustrates how security level λ of a scheme is related to β and n given that the scheme with access to a perfect distribution $D_{\mathbb{Z}^m,c,s}$ is $(\lambda+1)$-bit secure. There is no surprise that the two graphs only have minor differences because a relatively small r can meet the requirement for closeness of a large s.

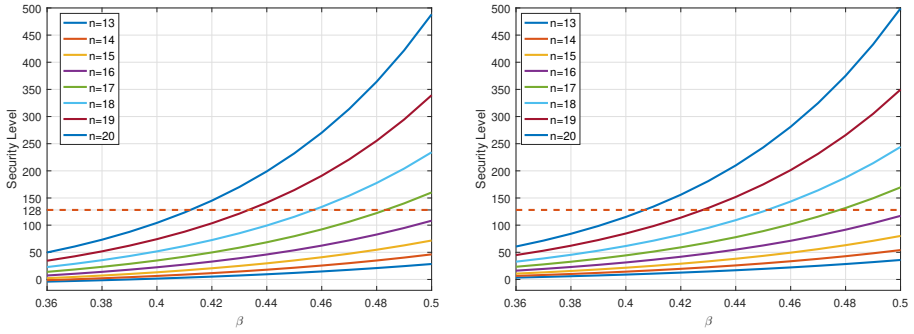


Fig. 5. The relation between security level of a scheme and β, n based on SD analysis: (a) s=8. (b) s=2^{10}.

5.2 Security Analysis with KL Divergence

Lemma 3 (Bounding Success Probability Variations, [22]). *Let $\mathcal{E}^{\mathcal{P}}$ be an algorithm making at most q queries to an oracle sampling from a distribution $\mathcal{P}$ and returning a bit. Let $\epsilon \leq 0$, and $\mathcal{Q}$ be a distribution such that $D_{KL}(\mathcal{P}\|\mathcal{Q}) < \epsilon$. Let x (resp. y) denote the probability that $\mathcal{E}^{\mathcal{P}}$ (resp. $\mathcal{E}^{\mathcal{Q}}$) outputs 1. Then, $|x - y| \leq \sqrt{q\epsilon/2}$.*

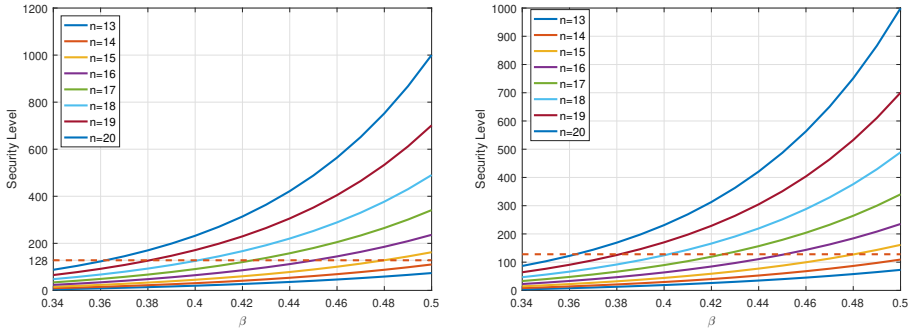
Security argument [24] It can be concluded from Lemma 3 that if a scheme is λ-bit secure with oracle access to a perfect distribution $\mathcal{P}$ and the KL divergence between $\mathcal{P}$ and another distribution $\mathcal{Q}$ satisfies $D_{KL}(\mathcal{P}\|\mathcal{Q}) \leq 2^{-\lambda}$, then this scheme is also about λ-bit secure with oracle access to $\mathcal{Q}$. Moreover, it turns a bound of KL divergence into a bound of SD. The security argument based on KL divergence verifies symmetry and triangle inequality though KL divergence itself does not (see Section 3.2 in [24] for detail).

Consider that a scheme with access to a perfect distribution $D_{\mathbb{Z}^m,c,s}$ is λ-bit secure. An adversary calls the sampler m times at each query. In our sampling algorithm, $m \leq$ the block-length N of polar codes. According to (32) and sub-additivity of KL divergence, $D_{KL}(D_{\mathbb{Z}^{1:m},c,s}\|P_{X_{1:r}^{1:m}}) \lesssim m(\varepsilon+\varepsilon^2)$. In order to achieve λ-bit secure after the tailcut, we have to guarantee $m(\varepsilon+\varepsilon^2) \leq 2^{-\lambda}$ by selecting $t \approx \sqrt{(\lambda+\log m)\ln 2/\pi}$. The number of levels needed is therefore $r = \lceil \log(2t \cdot s) \rceil$. It is observed that KL divergence is no better than SD with respect to efficiency when used to analyze the tailcut error.

As given in Section 4.4, the approximation error from polar coding is determined by $D_{KL}(P_{X_{1:r}^{1:m}}\|Q'_{X_{1:r}^{1:m}})$, $D_{KL}(Q'_{X_{1:r}^{1:m}}\|P_{X_{1:r}^{1:m}})$ which are upper bounded as

$$D_{KL}(P_{X_{1:r}^{1:m}}\|Q'_{X_{1:r}^{1:m}}) \leq r2N2^{-N^\beta}, \quad D_{KL}(Q_{X_{1:r}^{1:m}}\|Q'_{X_{1:r}^{1:m}}) \leq rN2^{-N^\beta}.$$

In order to achieve λ-bit security after $P_{X_{1:r}^{1:m}}$ is replaced by $Q_{X_{1:r}^{1:m}}$, we need to select $n = \log N$ and β properly such that $2^{-2^{n\beta}+n+\log(r)+1} \leq 2^{-\lambda}$. Fig. 6 shows how the security level is related to n, β in terms of different s. KL divergence shows its advantage over SD when used to analyse the approximation error from polar coding. To achieve the same security level, we can choose much smaller β and n than those obtained according to SD-based analysis, which will lead to a great reduction in both memory and running time.



(a) (b)

Fig. 6. The relation between security level of a scheme and β, n based on KL analysis: (a) s=8. (b) s=2^{10}.

5.3 Security Analysis of Tailcut With Rényi Divergence

Definition 8 (Rényi divergence). *Let P, Q be two distributions with supports $\mathcal{S}_P$ and $\mathcal{S}_Q$, respectively. Let $\mathcal{S}_P \subset \mathcal{S}_Q$. For $a \in (1, +\infty)$, we define the Rényi divergence of order a by*

$$R_a(P\|Q) = \left(\sum_{x \in \mathcal{S}_P} \frac{P(x)^a}{Q(x)^{a-1}} \right)^{\frac{1}{a-1}}.$$

In [25], a sharper bound on security based on Rényi divergence is given under the assumption that the number of adversary queries q to an λ-bit-secure scheme is far less than 2^λ. In NIST's proposals for post-quantum cryptography, it is assumed that $q \leq 2^{64}$. Denote by $\mathcal{Q}$ and $\mathcal{Q}_\gamma$ a distribution and its tailcut such that $\frac{\mathcal{Q}_\gamma}{\mathcal{Q}} \leq 1 + \gamma$ over the support $\mathcal{T}$ of $\mathcal{Q}_\gamma$ for some $\gamma \approx 1 - \mathcal{Q}(\mathcal{T})$. It is proved in [25] that $R_a(\mathcal{Q}_\gamma\|\mathcal{Q}) \leq 1 + \gamma$. Consider a cryptographic scheme with λ bits of security making $q \leq 2^{64}$ queries to $\mathcal{Q}$. By the security argument in [25], this scheme is believed to lose at most one bit of security if $\mathcal{Q}$ is replaced by $\mathcal{Q}_\gamma$ provided that

$$R_a(\mathcal{Q}_\gamma\|\mathcal{Q}) \leq 1 + \frac{1}{4q} \text{ for } a = 2\lambda.$$

In implementation, we only need to guarantee that

$$\frac{\mathcal{Q}_\gamma}{Q} \leq 1 + \gamma \text{ for } \gamma = \frac{1}{4q}.$$

In this paper, it is sufficient to claim at most 1 bit of security loss if the ideal distribution $D_{\mathbb{Z},c,s}$ is replaced by its tailcut $P_{X_{1:r}}$ for $\lambda = 128$ and $q = 2^{64}$ by taking

$$r = \left\lceil \log \left(2s \sqrt{\frac{(64 + \log m) \ln 2}{\pi}} \right) \right\rceil.$$

Compared with SD and KL-based analysis of tailcut, Rényi divergence is observed to yield a reduction on r by at most 1 level.

6 Complexity and Comparison

6.1 A Constant-Time Algorithm

At the implementation stage, the running time cost comes from two sources: calculating the LRs by SC decoding level by level and producing binary samples $U_{1:r}^{1:N}$ following the implementation rules given by (27) and (28). The polar sampler yields N samples at each run. For a fixed block-length N, no matter what the distribution is, the SC decoding at each level requires exactly $N(1 + \log N)$ steps of LR calculations where two LRs are assembled to give a new one (Appendices A and C). On the one hand, drawing a bit in constant time uniformly at random or deterministically according to (27) is easy and cheap. On the other

hand, sampling from a distribution statistically $2^{-\lambda}$-close to a Bernoulli variable $\mathcal{B}_p$ in (28) for a given bias p is also easy: take an approximation of p up to λ correct bits, then sample a uniform real $q \in [0, 1)$ up to λ bits of precision and answer 1 if and only if $q < p$ [7]. Moreover the fraction of unpolarized indexes vanishes as N tends to ∞. Therefore, the implementation stage shown in Fig. 4 is block-wise constant in time if the conditions below are satisfied:

- The number of levels r and the number of DGS samples N to produce at each run of the algorithm are fixed. Seeing that r is only related to the width s of $D_{\mathbb{Z},c,s}$ and the security level λ, but it has nothing to do with c. We claim polar sampling to be constant-time for fixed s but independent of c.
- The parameter β for each level is chosen properly such that $\mathcal{H}$, $\mathcal{L}$ and $\mathcal{H}^c \backslash \mathcal{L}$ for each level are determined.

6.2 Time Complexity

The Knuth-Yao and CDT sampler can work as a base sampler [19, 23] if combined with an expander. The polar sampler also allows such extension. Karmakar et al. [12] compared the time complexity of Knuth-Yao and CDT showing that the former can be made more time-saving. It is fair to compare the polar sampler only with Knuth-Yao. We used a non constant-time Knuth-Yao implemented in C++[3] which is faster than its constant-time version. We demonstrated the polar sampler in MATLAB with key functions (Appendix C) accelerated by C-based mex files. Seeing that this a prototype for functional test, there is a lot of room to improve the efficiency in the future.

The experiment was conducted on a PC with intel core-i7-6700 processor running at 3.40 GHz using one core. For the benchmarks, we select $s \in \sqrt{2\pi} \cdot \{3, 8, 32, 256\}$ and the target security level $\lambda = 64$. According to the KL-based security analysis, we specified β to achieve 64 bits of security with respect to $N \in \{2^{13}, 2^{14}, 2^{15}\}$, and we selected $r = \lceil \log(2st) \rceil$ where $t \approx \sqrt{(\lambda + \log m) \ln 2/\pi}$. We assume that one query to the sampling algorithms can obtain $m = 1024$ integer samples. The simulation results are shown in Table 1. Firstly, the polar sampler always outperforms Knuth-Yao in speed with respect to the above setting. Secondly, Knuth-Yao slows down almost linearly as 2^r grows while the polar sampler still provides a competitive speed. Thirdly, the polar sampler shows a modest speed reduction as N increases from 2^{14} to 2^{15}. This doesn't contradict the asymptotic information optimality claim. The online computation complexity of the polar sampler is $O(N \log N)$. The polarization phenomenon helps save time by reducing the fraction of unpolarized set. If N is not large enough such that the i.i.d. source pairs are not deeply polarized, the speedup benefited from polarization may not be able to compensate for the drop of speed induced by the increase of N.

[3] https://github.com/AaronHall4/BKW-Algorithm

Table 1. Comparison of time efficiency between the polar sampler and Knuth-Yao.

2^r	s	Knuth-Yao (samples/second)	polar sampler (samples/second)		
			$N = 2^{13}$	$N = 2^{14}$	$N = 2^{15}$
2^6	$3\sqrt{2\pi}$	1.934E4/s	$\beta = 0.487$ 3.639E4/s	$\beta = 0.4535$ 3.782E4/s	$\beta = 0.4244$ 3.737E4/s
2^7	$8\sqrt{2\pi}$	1.003E4/s	$\beta = 0.4876$ 3.171E4/s	$\beta = 0.454$ 3.225E4/s	$\beta = 0.425$ 3.217E4/s
2^9	$32\sqrt{2\pi}$	2.833E3/s	$\beta = 0.488$ 2.529E4/s	$\beta = 0.4544$ 2.591E4/s	$\beta = 0.4252$ 2.496E4/s
2^{12}	$256\sqrt{2\pi}$	3.555E2/s	$\beta = 0.4885$ 1.616E4/s	$\beta = 0.455$ 1.630E4/s	$\beta = 0.4257$ 1.610E4/s

6.3 Memory Cost

At the construction stage, one calculates the Bhatacharyya parameters using the degraded merging technique [29] to find the indexes of the high, low-entropy and unpolarized sets. We employ a 2-bit flag to specify which set an index i is in. Given the number of levels r and block-length N, to store all the flags requires $\frac{rN}{4}$ bytes in total.

We also need to store a likelihood ratio table of the symmetrized channel, e.g. $LR(x_{1:k-1}, x_k) = \frac{W(x_k=0|x_{1:k-1})}{W(x_k=1|x_{1:k-1})}$ for $k \leq r$. The table consists of 2^r data for all the r levels. The likelihood ratio is stored in natural order of $X_{1:k-1}$ such that once the samples for the first $k-1$ levels $X_{1:k-1}^{1:N}$ are ready we can find the corresponding likelihood ratio by the index $x_{1:k}$ without scanning the table.

In addition, as shown in Appendix C, the polar sampler will create a floating point array of size $N \times (n+1)$ and a bit array of the same size to store instant data. Fortunately, the space-efficient SC decoding proposed in [28] greatly reduces the array size to $N \times 1$.

7 Conclusions and Future Work

Our polar sampler is efficient, application-independent and constant-time. Our algorithm is effective in the case that when a large number of discrete Gaussian samples are required. The reduction in resources of random bits stems from the polarization process in which the randomness moves to the high-entropy set. For fixed parameters, the construction stage is prepared offline and the implementation stage is carried out online and constant in time. KL divergence is a more efficient metric than SD when used for security analysis of polar sampling. The Rényi divergence-based analysis of polar coding is still an open problem by now. It deserves more efforts to give a complete Rényi divergence-based analysis of our sampler and exploit the potential efficiency of Rényi divergence.

In this paper, we only use the basic 2×2 kernel, whose finite-length performance is not the best. Optimizing finite-length performance using other kernels of polar codes as well as various other issues are left to future work.

References

1. Arikan, E.: Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. IEEE Transactions on Information Theory **55**(7), 3051–3073 (2009)
2. Arikan, E.: Source polarization. 2010 IEEE International Symposium on Information Theory pp. 899–903 (2010)
3. Bai, S., Lepoint, T., Roux-Langlois, A., Sakzad, A., Stehlé, D., Steinfeld, R.: Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance. Journal of Cryptology **31**(2), 610–640 (2018)
4. Box, G.E.P., Muller, M.E.: A note on the generation of random normal deviates. Ann. Math. Statist. **29**(2), 610–611 (06 1958)
5. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley & Sons (2012)
6. Devroye, L.: Sample-based non-uniform random variate generation. In: Proceedings of the 18th conference on Winter simulation. pp. 260–265. ACM (1986)
7. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013. pp. 40–56. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
8. Dwarakanath, N.C., Galbraith, S.D.: Sampling from discrete Gaussians for lattice-based cryptography on a constrained device. Applicable Algebra in Engineering, Communication and Computing **25**(3), 159–180 (2014)
9. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. pp. 197–206. ACM (2008)
10. Hülsing, A., Lange, T., Smeets, K.: Rounded Gaussians – fast and secure constant-time sampling for lattice-based crypto. Cryptology ePrint Archive, Report 2017/1025 (2017)
11. Honda, J., Yamamoto, H.: Polar coding without alphabet extension for asymmetric models. IEEE Transactions on Information Theory **59**(12), 7829–7838 (2013)
12. Karmakar, A., Roy, S.S., Reparaz, O., Vercauteren, F., Verbauwhede, I.: Constant-time discrete Gaussian sampling. IEEE Transactions on Computers **67**(11), 1561–1571 (2018)
13. Knuth, D.: The complexity of nonuniform random number generation. Algorithm and Complexity, New Directions and Results pp. 357–428 (1976)
14. Liu, L., Yan, Y., Ling, C., Wu, X.: Construction of capacity-achieving lattice codes: Polar lattices. IEEE Transactions on Communications **67**(2), 915–928 (2019)
15. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 1–23. Springer (2010)
16. Marsaglia, G., Tsang, W.W., et al.: The Ziggurat method for generating random variables. Journal of Statistical Software **5**(8), 1–7 (2000)
17. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Advances in Cryptology–CRYPTO 2013, pp. 21–39. Springer (2013)
18. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. SIAM Journal on Computing **37**(1), 267–302 (2007)
19. Micciancio, D., Walter, M.: Gaussian sampling over the integers: Efficient, generic, constant-time. In: Annual International Cryptology Conference. pp. 455–485. Springer (2017)

20. Mori, R., Tanaka, T.: Performance of polar codes with the construction using density evolution. IEEE Communications Letters **13**(7) (2009)
21. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Annual Cryptology Conference. pp. 80–97. Springer (2010)
22. Pöppelmann, T., Ducas, L., Güneysu, T.: Enhanced lattice-based signatures on reconfigurable hardware. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 353–370. Springer (2014)
23. Pöppelmann, T., Ducas, L., Güneysu, T.: Enhanced lattice-based signatures on reconfigurable hardware. In: Batina, L., Robshaw, M. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2014. pp. 353–370. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
24. Prest, T.: Gaussian sampling in lattice-based cryptography. Ph.D. thesis, École Normale Supérieure (2015)
25. Prest, T.: Sharper bounds in lattice-based cryptography using the Rényi divergence. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 347–374. Springer (2017)
26. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing. pp. 84–93. STOC '05, ACM, New York, NY, USA (2005)
27. Saarinen, M.J.O.: Arithmetic coding and blinding countermeasures for lattice signatures. Journal of Cryptographic Engineering **8**(1), 71–84 (Apr 2018)
28. Tal, I., Vardy, A.: List decoding of polar codes. IEEE Transactions on Information Theory **61**(5), 2213–2226 (2015)
29. Tal, I., Vardy, A.: How to construct polar codes. IEEE Transactions on Information Theory **59**(10), 6562–6582 (2013)
30. Zhao, R.K., Steinfeld, R., Sakzad, A.: FACCT: Fast, Compact, and Constant-Time Discrete Gaussian Sampler over Integers. Cryptology ePrint Archive, Report 2018/1234 (2018)

A Computational Complexity of SC Decoding

Consider the SC decoding for an arbitrary polar code of length N. To estimate $u^{1:N}$ according to the rules given in (8), one needs to calculate the full set of LRs. Let $W : X \to Y$ denote a stochastic transition from the source X to side information Y with transition probability $W(Y|X) = P(Y|X)$. As shown in Fig. 1, the source polarization transform combines N i.i.d. copies of W in a recursive manner such that for any $0 \le m \le n$, $M = 2^m$, $N = 2^n$, $1 \le \kappa \le M/2$, the decoder calculates the LRs at the m-th layer of recursion as [1, Section VIII]

$$L_M^{(2\kappa-1)}(y_1^M, \hat{u}_1^{2\kappa-2}) = \frac{L_{M/2}^{(\kappa)}\left(y_1^{M/2}, \hat{u}_{1,o}^{2\kappa-2} \oplus \hat{u}_{1,e}^{2\kappa-2}\right) L_{M/2}^{(\kappa)}\left(y_{M/2+1}^{M}, \hat{u}_{1,e}^{2\kappa-2}\right) + 1}{L_{M/2}^{(\kappa)}\left(y_1^{M/2}, \hat{u}_{1,o}^{2\kappa-2} \oplus \hat{u}_{1,e}^{2\kappa-2}\right) + L_{M/2}^{(\kappa)}\left(y_{M/2+1}^{M}, \hat{u}_{1,e}^{2\kappa-2}\right)}, \tag{38}$$

and

$$L_M^{(2\kappa)}(y_1^M, \hat{u}_1^{2\kappa-1}) \left[L_{M/2}^{(\kappa)}\left(y_1^{M/2}, \hat{u}_{1,o}^{2\kappa-2} \oplus \hat{u}_{1,e}^{2\kappa-2}\right)\right]^{1-2\hat{u}_{2\kappa-1}} \cdot L_{M/2}^{(\kappa)}\left(y_{M/2+1}^{M}, \hat{u}_{1,e}^{2\kappa-2}\right). \tag{39}$$

where the notation $\hat{u}_{1,o}^{2\kappa-2}$ (resp. $\hat{u}_{1,e}^{2\kappa-2}$) represents a subvector of $\{\hat{u}^{(1)}, \cdots, \hat{u}^{(2\kappa-2)}\}$ with odd (resp. even) indexes. The stopping condition of the recursion is $L_1^{(1)}(y) = \frac{W(y|0)}{W(y|1)}$.

Observe that to calculate any LR pair $\left(L_M^{(2\kappa-1)}(y_1^M, \hat{u}_1^{2\kappa-2}), L_M^{(2\kappa)}(y_1^M, \hat{u}_1^{2\kappa-1})\right)$ at the m-th layer of the recursion, the decoder needs to know another LR pair $\left(L_{M/2}^{(\kappa)}(y_1^{M/2}, \hat{u}_{1,o}^{2\kappa-2} \oplus \hat{u}_{1,e}^{2\kappa-2}), L_{M/2}^{(\kappa)}(y_{M/2+1}^M, \hat{u}_{1,e}^{2\kappa-2})\right)$ at the (m-1)-th layer. The calculation of N LRs at layer m requires exactly N LR assembling at layer $m-1$. One can reversely compute the LRs layer by layer until he reaches the 0-th layer which is exactly the raw stochastic transition W. Suppose that assembling an LR pair of the $(m-1)$-th layer into one LR of the m-th layer takes one complexity unit, then computing all the N LRs of the n-th layer requires $N(1+\log N)$ units. Fig. 7 gives an example of $N = 8$.

B KL Divergence for the Low-Entropy Set

For $i \in \mathcal{L}_{X_k|X_{1:k-1}}$, Q' and Q follow the distribution respectively as

$$Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) = P(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})$$

and

$$\begin{aligned}&Q(\bar{u}_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) = 1\\ &\text{for } \ \bar{u}_k^{(i)} = \arg\max_{u\in\{0,1\}} P_{U_k^{(i)}|X_{1:k-1}^{1:N}, U_k^{1:i-1}}(u|x_{1:k-1}^{1:N}, u_k^{1:i-1}).\end{aligned}$$

By definition of KL divergence, we have

$$\begin{aligned}&D_{KL}(Q_{U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N}} \| Q'_{U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{N}})\\ &= \sum_{u_k^{1:i-1}, u_{1:k-1}^{1:N}} Q'(u_k^{1:i-1}, u_{1:k-1}^{1:N})[-1 \cdot \log Q'(\bar{u}_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})\\ &- 0 \cdot \log(1 - Q'(\bar{u}_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})) + (0\log 0 + 1\log 1)].\end{aligned}$$

By definition Shannon entropy,

$$\begin{aligned}&H_P(U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N})\\ &= -\sum_{u_k^{1:i-1}, u_{1:k-1}^{1:N}} Q'(u_k^{1:i-1}, u_{1:k-1}^{1:N}) \sum_{u_k^{(i)}} Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) \log Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})\end{aligned}$$

for $i \in \mathcal{L}_{X_k|X_{1:k-1}}$. It can be proved for $0.5 < Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) < 1$ that

$$-Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) \log Q'(u_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N}) \geq -\log Q'(\bar{u}_k^{(i)}|u_k^{1:i-1}, u_{1:k-1}^{1:N})$$

and hence

$$D_{KL}(Q_{U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N}} \| Q'_{U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{N}}) \leq H_P(U_k^{(i)}|U_k^{1:i-1}, U_{1:k-1}^{1:N}).$$

C Functions

At the construction stage, we employed [29, Algorithm A and C] to efficiently calculate the Bhatacharyya parameters. The following Algorithms are key functions at the implementation stage. Algorithm 1 yields $X_k^{1:N}$ given $x_{k-1}^{1:N}$ produced by upper levels of the partition tree as shown in Fig. 4. Due to space limitations, we briefly describe two sub-functions Algorithm 2 and 3 which evolve from [28, recursivelyCalcP(),recursivelyUpdateB()] to efficiently calculate the LRs in equation (18). We keep the notations consistent with Appendix A.

Fig. 7 shows an example of the butterfly circuit to calculate LRs when $N = 8$. We define two properties of each layer, i.e., phase and branch denoted by integers ϕ and ψ, respectively. At layer m, $1 \leq \phi \leq 2^m$ and $0 \leq \psi < 2^{n-m}$. In Fig. 7, we distinguish different phases at each layer by different colors. We have a global array LRReg$[N][n+1]$ indexed by integers $1 \leq i \leq N$ and $0 \leq m \leq n$. Note that for any layer m each integer $1 \leq i \leq 2^n$ has a unique representation as

$$i = \langle \phi, \psi \rangle_m = \phi + 2^m \cdot \psi.$$

Therefore, each element LRReg$[i][m]$ will be uniquely loaded by $L_{2^m}^{(\phi)}$ of phase ϕ and branch ψ in the routine. The other global array of the same size is denoted by UReg$[N][n+1]$ whose elements are single bits. For a generic array A we abbreviate $A[\langle \phi, \psi \rangle_m][m]$ as $A[\langle \phi, \psi \rangle][m]$.

Remark 1. The polar sampler yields samples in a block-by-block fashion rather than one-by-one. What the routines do is to perform exactly $N(\log N + 1)$ steps of LR assembling and to yield $X_k^{1:N}$. Despite the *if* and *else* statements in Algorithm 1, which conditional branch to go at each iteration of the *for loop* will be fixed given the three index sets. Algorithm 2 and 3 traverse the butterfly circuit and their overall time consumption is only determined by N.

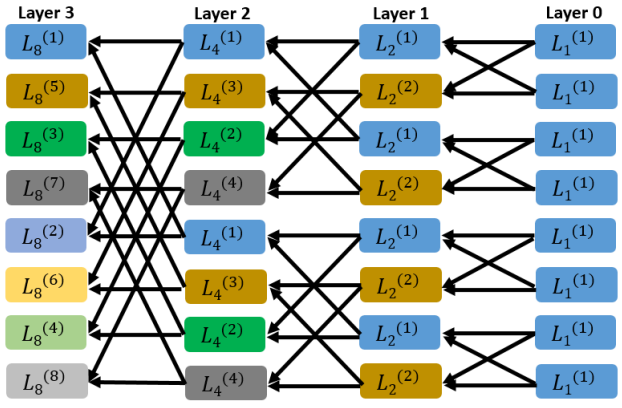


Fig. 7. The butterfly circuit to calculate LR.

input : $L_1^{1:N}$, HighEntropySet, LowEntropySet, (global:LRReg, URegister)
output: $X_k^{1:N}$

LRReg [:][0]=$L_1^{1:N}$;
for $i \leftarrow 1$ **to** N **do**
  LRReg $\leftarrow$ `CalcLR(n,i)`;
  if *index* $i \in$ HighEntropySet **then**
    URegister$[i][n] \leftarrow$`randomBin(); // equation (27).`
  end
  else if *index* $i \in$ LowEntropySet **then**
    URegister$[i][n] =$ LRReg$[i][n] < 1$; `// equation (27).`
  end
  else index $i \in$ Non-polarizedSet
    URegister$[i][n] =$`Uniform()` $< 1/(1 +$ LRReg$[i][n])$; `// Uniform() produces real` $y \in (0, 1]$ `uniformly at random; equation (28).`
  end
  URegister $\leftarrow$ `CalcBit(n,i)`;
end
return $X_k^{1:N} =$ URegister[:][0]

Algorithm 1: Polar sampler for any one level of the partition tree.

input : m, ϕ
output: updated LRReg

if $m = 0$ **then return**;
set $\kappa \leftarrow \lceil \phi/2 \rceil$;
if $\phi \mod 2 = 1$ **then** `CalcLR` $(m-1, \kappa)$;
for $\psi = 0, \cdots, 2^{n-m} - 1$ **do**
  if $\phi \mod 2 = 1$ **then** LRReg $[\langle \phi, \psi \rangle][m] \xleftarrow{\text{equation(38)}}$ $(\text{LRReg}[\langle \kappa, 2\psi \rangle][m-1], \text{LRReg}[\langle \kappa, 2\psi+1 \rangle][m-1])$;
  else temp=UReg $[\langle \phi - 1, \psi \rangle][m]$; LRReg $[\langle \phi, \psi \rangle][m] \xleftarrow{\text{equation(39)}}$ $(\text{LRReg}[\langle \kappa, 2\psi \rangle][m-1], \text{LRReg}[\langle \kappa, 2\psi+1 \rangle][m-1])$;
end

Algorithm 2: The CalLR( ) function.

input : m, ϕ
output: updated UReg

if $\phi \mod 2 = 1$ **then return**;
set $\kappa \leftarrow \lceil \phi/2 \rceil$;
for $\psi = 0, \cdots, 2^{n-m} - 1$ **do**
  UReg $[\langle \kappa, 2\psi \rangle][m-1] \leftarrow \text{UReg}[\langle \phi - 1, \psi \rangle][m] \oplus \text{UReg}[\langle \phi, \psi \rangle][m]$;
  UReg $[\langle \kappa, 2\psi+1 \rangle][m-1] \leftarrow \text{UReg}[\langle \phi, \psi \rangle][m]$;
end
if $\kappa \mod 2 = 0$ **then** `CalcBit` $(m-1, \kappa)$;

Algorithm 3: The CalBit( ) function.