High-speed Implementation of Privacy Amplification in Continuous-variable Quantum Key Distribution

Yujie Luo , Yang Li , Jie Yang , Li Ma , Wei Huang , Bingjie Xu*
Science and Technology on Security Communication Laboratory, Institute of Southwestern Communication, Chengdu 610093, China

ABSTRACT

Privacy amplification (PA) is an essential process for high-speed and real-time implementation of a continuous-variable quantum key distribution (CV-QKD) system. This work focuses on the improvement of the performance of PA, and we realize PA with Toeplitz matrix and accelerate it using fast Fourier transform (FFT) on graphic processing unit (GPU). Based on the architectural feature of FFT, we adjust its form of input length and obtained an average speed of PA about 2Gbps with input length ranges from 1Mbits to 128Mbits, which is length-adaptable to satisfy various requirements of CV-QKD systems at different transmission distances. Furthermore, we test this work with different compress ratios of PA, which can also achieve a high implementation speed around 2Gbps. With the method used in this paper, the requirements of PA for the high-speed and real-time CV-QKD system can be entirely satisfied.

Keywords: continuous-variable quantum key distribution; privacy amplification; Toeplitz matrix; FFT.

1. INTRODUCTION

Quantum key distribution (QKD) is one of the most practical applications of quantum information technology, which accomplishes the generation and distribution of a set of physically unconditional secure key between Alice and Bob. The key is then used to encrypt the information to ensure secure communication. There are mainly two kinds of QKD systems: Discrete Variable-Quantum Key Distribution (DV-QKD) system and Continuous Variable-Quantum Key Distribution (CV-QKD) system. Due to the advantages in generation, transmission and detection of the continuous variable quantum states, CV-QKD has attracted significant interest recently. A practical CV-QKD system contains two phases: the quantum processing and the post-processing. The former procedure mainly includes the generation, transmission and detection of the quantum information, and is implemented via a quantum channel which can be eavesdropped by an eavesdropper (Eve). After this procedure, Alice and Bob share a set of correlated but insecure raw data. It is essential to perform post-processing for the security and consistency of the final key through a classical authenticated channel, and this procedure usually includes basis sifting, parameter estimation, information reconciliation and privacy amplification (PA). In this paper, we focus on the PA of CV-QKD system.

After the basis sifting, parameter estimation and information reconciliation, Alice and Bob will obtain a set of binary data which are theoretically identical but not secure because Eve may have partial information on it. As a result, it is necessary to remove the information known by Eve via the PA operations so that Alice and Bob can obtain a set of consistent and secure final key, which increases the security of final key at the expense of shortening its length. Bennett and Brassard first proposed PA in 1988[1], and then they pointed out that the selection of hash functions is the crux for the security of PA in 1995[2]. The universal hash function families are wildly used to eliminate Eve's information about the final key, which are first proposed by Carter in 1979[3]. As one of the universal hash function families, Toeplitz matrix is frequently used in PA for its simple structure and high-speed implementation[4].

*xbjpku@ 163.com

Quantum and Nonlinear Optics VII, edited by Kebin Shi, Dai-Sik Kim, Chuan-Feng Li, Proc. of SPIE
Vol. 11558, 115580K · © 2020 SPIE · CCC code: 0277-786X/20/$21 · doi: 10.1117/12.2574969

To improve the security of CV-QKD system, large input and output length of PA is required which can increase the randomness and security of the final key. Furthermore, it is essential to consider the finite size effect for a practical CV-QKD system, which influences the security of the final key after PA[5]. To reduce the finite size effect, the input length of PA should be on the order of $10^8, 10^9, 10^{10}$ when the transmission distance is about 50km, 80km, 100km, respectively[6]. This leads to high computation complexity and large storage demand of the data, and efficient implementation of PA is very difficult. As a result, the comprehensive performance of the CV-QKD system is limited. For these reasons, PA with long input length becomes one of the performance bottlenecks of the CV-QKD system.

To solve this problem, various works have been done in many aspects, such as the hash functions, the implementation algorithms and platforms. Instead of Toeplitz matrix, four basic multiplication algorithms[7] are chosen to construct an optimal algorithm to adapt to different input length of PA. But this is an iterative algorithm meaning it consumes large resources with large input length. When the input length is 1Mbits and 10Mbits, the execution speed of PA is 14.86Mbps and 10.88Mbps, respectively. In [8], the authors speeded up PA by Toeplitz matrix using Field Programmable Gate Array (FPGA), and first divided the matrix into many smaller blocks which can be processed in parallel. In 2016, Ref. [9]first speeded up the Toeplitz matrix on CPU with number theoretical transform (NTT), which can reduce the computational complexity from $O(n^2)$ to $O(n\log_2 n)$. The authors in [10] proposed using fast Fourier transform (FFT) to accelerate Toeplitz matrix, which achieved 60.443Mbps when the input length is 12.8Mbits. The computational complexity of FFT-based algorithm is also $O(n\log_2 n)$. Based on the methods listed above, many works have been done to further improve the performance of PA. On the hardware implementation, Ref. [11] and [12] both proposed improved block algorithm of PA with Toeplitz matrix on FPGA, and achieved 41Mbps and 65.443Mbps respectively. In [13], traditional FFT-based algorithm is improved via modified 2D-FFT to reduce the number of computations and read/write operations. In addition, PA can be also implemented on GPU. The authors in [14] proposed a parallel FFT-based algorithm which is length-compatible on GPU, and obtained a speed over 1Gbps at arbitrary input length with a fixed compress ratio of 0.1. An FFT enhanced high-speed and large-scale PA scheme is proposed on commercial CPU platform in [15]. When input scale is 128Mbits, the speed can reach 71.16Mbps, 54.08Mbps and 39.15Mbps with the compress ratio of 0.125, 0.25 and 0.375 respectively.

Until now, various schemes of PA have been proposed and demonstrated by using different methods and many progresses have been made. However, it is necessary to optimize the implementation of PA for the CV-QKD system, which may have a high repetition frequency more than GHz, e.g. the local local oscillator (LLO) CV-QKD system. As a result, the execution speed of PA should be also fast enough to guarantee the high-speed CV-QKD system. Though [14] achieved PA process around 1.35Gbps, the result is only verified under a low compress ratio of 0.1 and the speed may decrease with a higher compress ratio for more storage demand and computations are involved in.

In this paper, we implement PA based on Toeplitz matrix and accelerate it with FFT which decreases the computation complexity from $O(n^2)$ to $O(n\log_2 n)$ on graphic processing unit (GPU). We analyze and demonstrate the influence of the input and output length on the performance of PA. A high average speed of PA around 2Gbps under the input length ranging from 1Mbits to 128Mbits with a compress ratio of 0.1, 0.2 and 0.3 respectively. As far as we know, this work achieves the fastest execution speed compared to the previous implementations of PA in the literature, and this speed is high enough to be adopted with GHz or even higher repetition frequency CV-QKD systems. Furthermore, the implementation scheme of PA in this work can be applied to the post-processing of quantum random number generator (QRNG).

The rest of this paper is organized as follows. PA is briefly introduced in Section 2. In Section 3, the improved implementation of PA is described in detail. In Section 4, the experiment results are described and analyzed. In Section 5, one of applications of PA is introduced. In Section 6, a brief conclusion is given.

2. RELATED WORK

2.1 Privacy amplification

Privacy amplification is an essential step in the post-processing of QKD systems, and is implemented through an authenticated classical public channel, so that Eve can only obtain the information exchanged by Alice and Bod but cannot modify it without been discovered. The purpose of PA is to remove partial information known by Eve, and extract shorter but more secure key from weak kay. Hash functions are used to complete this compression, and the security of final key depends on its collision probability.

Supposing Alice and Bob share an N-bit identical string after the information reconciliation, called weak key W. Eve learns a correlated string E with $t(t \le N)$ bits mutual information about W. Then Alice and Bob publicly choose a

hash function G: $\{0,1\}^N \to \{0,1\}^M$ randomly to perform PA, and then obtain an *M*-bit string *K* called final key. After PA, Eve almost know nothing about *K* with its partial information about *W*.

2.2 Universal hash function families

A proper hash function means low collision probability and low computation complexity, which are significant to the security and performance of PA. The collision probability should be very small to guarantee security of the final key after PA. Computation complexity is related to the choice of hash function, where lower complexity leads to higher speed for PA.

Among the universal hash function families, Toeplitz matrix is an excellent choice, where the whole matrix elements can be determined by cyclic shift with the first row and first column. It only requires $N+M-1$ elements to construct a $N \times M$ Toeplitz matrix, whose structure is shown in (1).

$$T = \begin{bmatrix} t_{N-1} & t_N & \cdots & t_{N+M-2} \\ \vdots & t_{N-1} & \ddots & \vdots \\ t_{M-1} & \ddots & \ddots & t_N \\ \vdots & \ddots & \ddots & t_{N-1} \\ t_1 & \cdots & \ddots & \vdots \\ t_0 & t_1 & \cdots & t_{M-1} \end{bmatrix}, \tag{1}$$

Due to the structural characteristics of Toeplitz matrix, it is easy to know all elements as long as $t_0, \cdots, t_{N+M-2}$ are known. It also brings a low collision probability of $N * 2^{-M+1}$, where *N*(*M*) is the input (output) bits string length of PA.

2.3 Challenge brought by the finite size effect in PA

In a pratical CV-QKD system, the finite size effect has a great influence on the security of the final key after PA. Due to the finite size effect, the result obtained directly in parameter estimation is inaccurate. It is essential to reduce the finite size effect as much as possible. Supposing that *x* and *y* are the raw data of Alice and Bob after their prepration and measurement of the quantum states, *E* is the corresponding correlated quantum state of Eve. When considering the finite size effect, the final secret key rate of a CV-QKD system can be expressed as [4]:

$$k = \frac{n}{N}[\beta I(x:y) - S_{\varepsilon_{PE}}(y:E) - \Delta(n)], \tag{2}$$

where *N* is the length of the signal exchanged by Alice and Bob, and *n* is the length of bit string used for the generation of the final key, the rest $m = N - n$ are used for parameter estimation. β is the reconciliation efficiency, $I(x:y)$ represents the mutual information of Alice and Bob, $S_{\varepsilon_{PE}}(y:E)$ represents the Helovo bound of Bob and Eve. $\Delta(n)$ represents the effect of finite size that can be expressed by:

$$\Delta(n) \equiv (2\dim \mathrm{H}_x + 3)\sqrt{\frac{\log_2(2/\bar{\varepsilon})}{n}} + \frac{2}{n}\log_2(1/\varepsilon_{PA}), \tag{3}$$

where $\dim \mathrm{H}_x$ is the dimension of the Hilbert space corresponding to the variable *x* used in the raw key, $\bar{\varepsilon}$ is a smoothing parameter and $\varepsilon_{\mathrm{PA}}$ is the failure probability of PA, and *n* is the input length of PA. Both $\bar{\varepsilon}$ and $\varepsilon_{\mathrm{PA}}$ can be optimized as small as possible during the computation under the condition of satisfing some equalities.

As described in (2), $\Delta(n)$ has a direct impcat on the final key rate. From (3), one can see that the value of $\Delta(n)$ is mainly decided by the input length of PA, which should be as large as possible when other parameters are fixed to ensure the security of the CV-QKD system. Fig.1 shows the relationship between the final key rate and transmission distance when considering the finite size effect. Tab.1 gives specific values of the final secret key rate of different tranmission distances with various input length of PA. From Fig.1 and Tab.1, to get non-negative secure key rates in CV-QKD system, the input length of PA should be at least the order of $10^8, 10^9, 10^{10}$ when the transmission distance are about 50km, 80km, 100km, respectively.

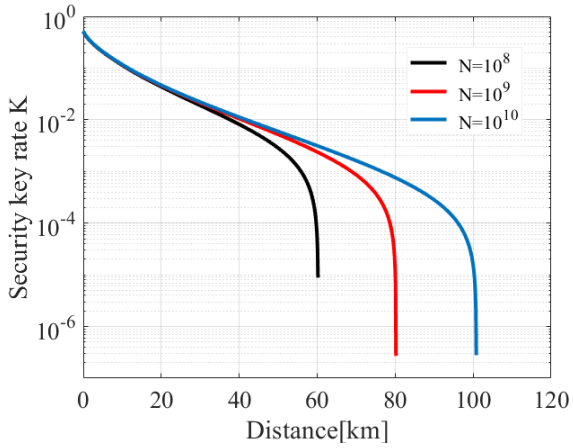

Figure 1. Relationship between the transmission distance and the key rate when considering the finite size with the electrical noise, detection efficiency, excess noise and reconciliation efficiency of 0.041, 0.606, 0.04 and 0.94 respectively.

Table 1. The final secret key rate of different transmission distance with various input length.

Distance / N	25km	50km	80km	100km
10^8	0.0287	0.0029	-0.0022	-0.0027
10^9	0.0311	0.0052	1.88e-05	-6.83e-04
10^{10}	0.0319	0.0060	7.68e-04	1.73e-05

Generally speaking, the final key rate decreases as the increase of the transmission distance of the CV-QKD system with a fixed security parameter, which corresponds to the failure probaility of the system (usually fixed to 10^{-10}). As a result, the effect of the finite size brings more obvious influence on the practical system with long transmission distance . In a word, the input length of PA should be large enough to reduce the effect of the finite size and guarantee the security of the CV-QKD system.

3. HIGH SPEED IMPLEMENTATION OF PA

As mentioned above, PA is implemented via the employment of hash functions, e.g. Toeplitz matrix. The most widely used method to perform PA is based on matrix multiplication, which can be expressed by $W \times T = K$, where the weak key W multiplied by the Toeplitz matrix T to produce the final secure key K. For the matrix multiplication between a $1\times N$ vector and a $N \times M$ Toeplitz matrix, as described in (4), where N(M) is the input (output) length of PA.

$$\begin{bmatrix} a_0 & \cdots & a_i & \cdots & a_{N-1} \end{bmatrix} \times \begin{bmatrix} t_{N-1} & t_N & \cdots & t_{N+M-2} \\ \vdots & t_{N-1} & \ddots & \vdots \\ t_{M-1} & \ddots & \ddots & t_N \\ \vdots & \ddots & \ddots & t_{N-1} \\ t_1 & \cdots & \ddots & \vdots \\ t_0 & t_1 & \cdots & t_{M-1} \end{bmatrix} = \begin{bmatrix} k_0 & \cdots & k_i & \cdots & k_{M-1} \end{bmatrix} . \quad (4)$$

However, direct matrix multiplication needs large resource when N and M are required to be as large as possible. For the specific structure, Toeplitz matrix is chosen to reduce the data needed and can be accelerated by FFT.

Because of its structure, $N+M-1$ rather than $N*M$ elements are enough for the Toeplitz matrix to do this operation. What's more, FFT can be used to speed up the process, which enables the computation complexity to decrease from $O(n^2)$ to $O(n\log_2 n)$. There is based on an important property of the cyclic matrix during the operation using FFT, i.e.

$$C = F \cdot diag(f_c) \cdot F^{-1} , \quad (5)$$

where C is a cyclic matrix, F and F^{-1} represent the transformation matrix of FFT and IFFT separately, f_c is the FFT result of the vector c, which is the first row of C. To use this property, we can transform the $N \times M$ Toeplitz matrix into a $(N+M-1)\times(N+M-1)$ cyclic matrix first. As for the weak key, should be also extended from N to the same length of $N+M-1$ by filling 0 at the end. Then the initial matrix multiplication can be converted to:

$$K = W^{'} \times C = W^{'} \cdot F \cdot diag(f_c) \cdot F^{-1} = ifft[\, fft(c) .* fft(W^{'})], \tag{6}$$

where K is the output of PA with the length of $N+M-1$, c refers to the first row elements of the transformed cyclic matrix C and $W^{'}$ is the extended weak key. Finally, it is necessary to distract the 0^{th} to $(M-1)^{th}$ of K as the result of PA.

As shown in (6), to obtain the final key, three FFT operations and one multiplication operation are needed. It is obvious that the FFT operation affects the performance of PA significantly. Among the factors influencing the performance of FFT in different platforms, the input size is important to FFT. When the input length of FFT can be described as 2^n, it performs with excellent efficiency. What is more, the data to be processed will be extended to 2^n by filling 0 if it is lower than 2^n. So it is important to set the input size correctly in practical application to obtain high performance of FFT.

In this work, we optimize FFT with its input length, which significantly influences the performance of PA. For the parallel computing capability of GPU, we first divide the initial large Toeplitz matrix into many smaller Toeplitz matrixes with some regularity by columns, which can be accelerated by FFT independently and effectively. The weak key is also divided into blocks with the same number. This process can be expressed in the following form:

$$\begin{aligned} T &= \begin{bmatrix} T_1 & \cdots & T_i & \cdots & T_P \end{bmatrix}^T \\ W &= \begin{bmatrix} w_1 & \cdots & w_i & \cdots & w_P \end{bmatrix} \end{aligned} \tag{7}$$

where P is the number of blocks and usually set as 2^r. The smaller Toeplitz matrix T_i can be constructed by $n+m-1$ elements, the length of w_i is n, which should be extended to $n+m-1$ by filling 0 at the end. Then T_i and w_i perform PA independently in parallel as described in (6), and obtain the intermediate key K_i, which should be distracted first and then add them together. Finally we can get the final result of PA in the form of

$$K = \begin{bmatrix} K_1 & \oplus & K_i & \oplus & K_p \end{bmatrix}. \tag{8}$$

As mentioned above, blocks of the Toeplitz matrix and weak key perform PA independently, which means the input length of FFT in a block equal to $n+m-1$. We define two kinds of input length: one is the input length of PA denoted by N, and the other is the input length of FFT denoted $N+M-1$. M is the output length of PA, which is related to the compress ratio. This work focus on the latter, and we transform it into a close value which can be written as 2^s, equaling to $2^r * (n+m-1)$ in our scheme, which may bring excellent performance of FFT. To prove this, we compare the result of this work with others related work under the same condition, i.e. the range of input length of PA, implementation platform and compress ratio. What is more, different compress ratios lead to limited change of the execution speed when using this method. This means we can obtain a high speed even with a higher compress ratio, which is suitable for different requirements of the practical CV-QKD systems

4. RESULTS AND ANALYSIS

This work realizes PA with the Toeplitz matrix and uses FFT to speed up the process on GPU. There are many factors affecting the execution speed of PA, such as the memory and performance indicators of GPU, the length of input data, the parameter configuration of FFT et al. So the performance of PA can be optimized via different methods. We speed up PA mainly by optimizing the FFT operation. The results are given as follows.

Table 2. The execution speed of PA under different input length of FFT.

Input length of FFT(bit)	2^{20}	2^{21}	2^{22}	2^{23}	2^{24}	2^{25}	2^{26}	2^{27}
Speed (Gbps)	2.18	2.14	2.38	2.10	2.16	2.19	2.17	2.02

The execution speed of PA implemented with different input length of FFT under the compress ratio of 0.1 is given in Tab. 2. The GPU platform is NVIDIA TITIAN XP. As mentioned above, the input length of FFT is 2^s, which is

suitable for this operation. Furtherly, GPU is capable to calculate many blocks in parallel to make full use of the computation recourse. We achieve an average execution speed of PA more than 2Gbps under the input length of FFT ranging from 2^{20} to 2^{27} with compress ratio of 0.1, so that the input length of PA approximately equals to $2^s \times 10^6$ ($s = 0,1,2,3,4,5,6,7$). Fig.2 shows the performance of PA with different input length, which are both in the range of 1Mbits to 128Mbits. The difference between this work and [14] is that whether input length of FFT can be described in 2^n. Performance will be highly optimized when input sizes can be written in the form $2^a \times 3^b \times 5^c \times 7^d$ for the cuFFT library. In general, smaller prime factor brings better performance, i.e. powers of two are fastest. Tab. 3 shows two kinds of time taken to execute FFT under different forms of input length.

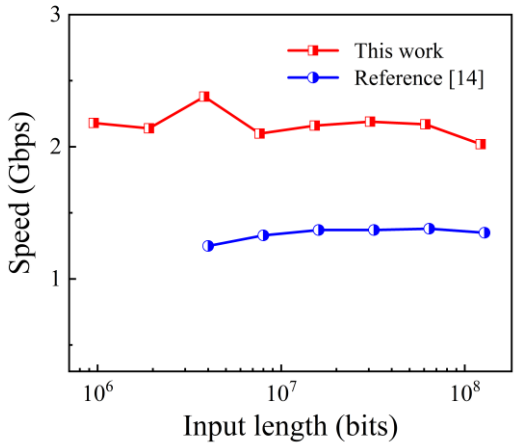

Figure 2. The execution speed comparison between this work and reference [14] with compress ratio of 0.1.

Table 3. Two kinds of time taken to execute FFT under different forms of input length with compress ratio of 0.1.

Input length of FFT	2^{27}=134217728	2^{26}=67108864	2^{25}=33554432
Time consumed (ms)	13	6	3
Input length of FFT	128E6+128E5-1 =140799999	64E6+64E5-1 =70399999	32E6+32E5-1 =35199999
Time consumed (ms)	40	20	10

Compared with the result obtained by [14], the speed is increased more than 50% on the same GPU with the same compress ratio of 0.1 in this paper, which can satisfy the requirement of high-speed CV-QKD system under variable input length considering finite size effect. This result obviously proves the significant influence on the performance of PA with different input length of FFT.

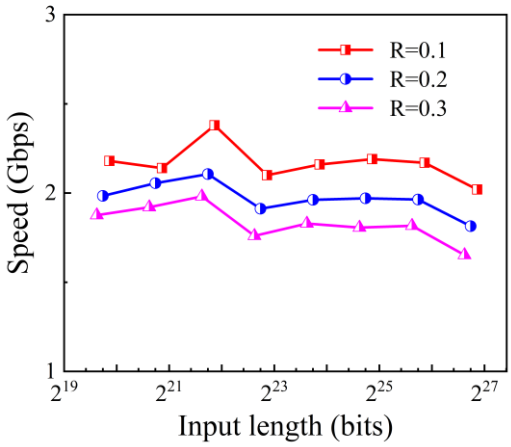

Figure 3. The different execution speed of PA when input length of FFT is powers of two under different compress ratio of 0.1, 0.2 and 0.3.

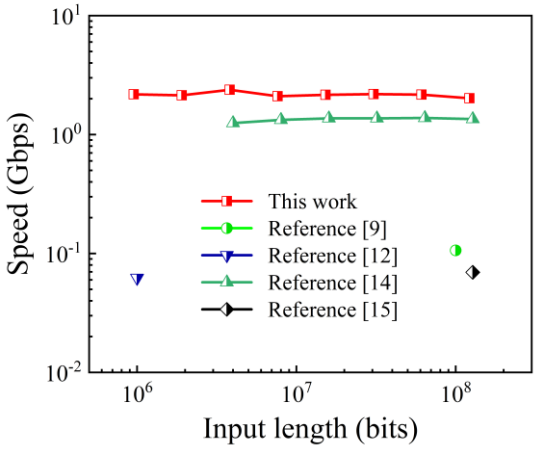

Figure 4. Comparison of execution speed of PA between this work and others.

Fig. 3 describes the various execution speed of PA under different compress ratios, where R means the compress ratio. When the input length of PA remains the same, different compress ratios mean different output length. Generally, the larger compress ratio leads to more random data of the output, which means more data to be processed and requires larger computation recourse. As a result, this will degrade the performance. From Fig.3, we can also achieve a high speed of about 2Gpbs even with a compress ratio of 0.3, which is high enough to be applied to the practical application of CV-QKD systems demanding high speed and security.

In Fig.4, it shows the comparison of the execution speed of PA of this work and others. We obtain an average execution speed about 2Gbps, which is faster than other known results with the same input length, and it is fast enough to ensure a high-speed and real-time CV-QKD system in practical applications. Furthermore, when the input length is 2^{27} bits, it is large enough to reduce the finite size effect with the transmission distance within 50km. Finally, the result is stable with variable input length. In other words, we may achieve this speed with larger input length as long as recourse of GPU is enough.

5. APPLICATION

The implementation of PA using Toeplitz matrix can be applied to the post-processing of QRNG. In this work, we test the execution speed of the post-processing of QRNG with the implementation scheme of PA using Toeplitz matrix. The result in shown in Tab. 4.

Table 4. The speed of post-processing of QRNG via PA with different-size Toeplitz matrix with compress ratio 0.1.

Size of Toeplitz matrix (bit)	Execution speed (Gbps)
122016256*12201984	2.02
61008384*6100992	2.17
30504448*3050496	2.19
15252480*1525248	2.16

From Tab. 4, we can achieve 2 Gbps for the post-processing of QRNG. Furthermore, we test the randomness of compressed data after PA. Fig. 5 gives ENT tests results of the data before and after PA respectively. The result comparison shows that data obtained after PA is random when with nonrandom input data.

Table 5. The ENT test results of data before PA and after PA.

ENT test before PA	ENT test after PA
Entropy = 7.637831 bits per byte (Optimum compression would reduce the size of the byte file by 4 percent) Chi square distribution is 194038167.65 (randomly would exceed this value less than 0.01 percent of the times) Arithmetic mean value of data bytes is 127.5829 (127.5 = random) Monte Carlo value for π is 3.443157007 (error 9.60 percent) Serial correlation coefficient is -0.003552 (totally uncorrelated = 0.0)	Entropy = 7.999963 bits per byte (Optimum compression would reduce the size of the byte file by 0 percent) Chi square distribution is 282.83 (randomly would exceed this value 11.13 percent of the times) Arithmetic mean value of data bytes is 127.4351 (127.5 = random) Monte Carlo value for π is 3.143505773 (error 0.06 percent) Serial correlation coefficient is -0.000077 (totally uncorrelated = 0.0)

6. CONCLUSIONS

This paper mainly implements PA with Toeplitz matrix which is accelerated by FFT on GPU. Due to the finite size effect, the input length of PA should be as large as possible, which leads to difficulty to implement PA with high efficiency for the limited resource and computational ability. Even worse, it may degrade the performance of the whole CV-QKD system. To solve this problem, we focus on the influence of the input length of FFT, which brings excellent performance when it is set as 2^s. Based on this feature of FFT, we transform the input size into a close value which is powers of two. An average speed of about 2Gbps with different input length is obtained using this method, which is enough for the high-speed and real-time CV-QKD system, even with a high repetition frequency more than GHz.

ACKNOWLEDGMENTS

This work was supported in part by China NSF under Grants 61901425, in part by Sichuan Youth Science and Technology Foundation under Grants 2017JQ0045 and 2019JDJ0060, and in part by CETC Fund under Grant 6141B08231115.

REFERENCES

[1] C. H. Bennett, G. Brassardm, and J.-M. Robert, “Privacy amplification by public discussion,” SIAM journal on Computing, 17(2), 210-229 (1988).
[2] C. H. Bennett, G. Brassard, C. Crkpeau et al., “Generalized privacy amplification,” IEEE Transactions on Information Theory, 41(6), 1915-1923 (1995).
[3] J. L. Carter, and M. N. Wegman, “Universal classes of hash functions,” Journal of computer and system sciences (18), 143-154 (1979).
[4] H. Krawczyk, "LFSR-based Hashing and Authentication." 129-139.

[5] A. Leverrier, F. Grosshans, and P. Grangier, “Finite-size analysis of a continuous-variable quantum key distribution,” Physical Review A, 81(6), 36-43 (2010).
[6] P. Jouguet, S. e. Kunz-Jacques, A. Leverrier et al., “Experimental demonstration of long-distance continuous-variable quantum key distribution”.
[7] C.-M. Zhang, M. Li, J.-Z. Huang et al., “Fast implementation of length-adaptive privacy amplification in quantum key distribution,” Chinese Physics B, 23(9), (2014).
[8] H.-f. Zhang, J. Wang, K. Cui et al., “A real-time QKD system based on FPGA,” Journal of Lightwave Technology, 30(20), 3226-3234 (2012).
[9] R. Takahashi, Y. Tanizawa, and A. R. Dixon, "High-Speed Implementation of Privacy Amplification in Quantum Key Distribution," 6th Int. Conf. Quantum Cryptography (2016).
[10] B. Liu, K.-B. Zhao, W.-R. Yu et al., “FiT-PA: Fixed Scale FFT Based Privacy Amplification Algorithm for Quantum Key Distribution,” Journal of Internet Technology, 17(2), 309-320 (2016).
[11] J. Constantin, R. Houlmann, N. Preyss et al., “An FPGA-Based 4 Mbps Secret Key Distillation Engine for Quantum Key Distribution Systems,” Journal of Signal Processing Systems for Signal Image and Video Technology, 86(1), 1-15 (2017).
[12] S.-S. Yang, Z.-L. Bai, X.-Y. Wang et al., “FPGA-Based Implementation of Size-Adaptive Privacy Amplification in Quantum Key Distribution,” IEEE Photonics Journal, 9(6), 1-8 (2017).
[13] Q. Li, B. Z. Yan, H. K. Mao et al., “High-Speed and Adaptive FPGA-Based Privacy Amplification in Quantum Key Distribution,” Ieee Access, 7, 21482-21490 (2019).
[14] X. Y. Wang, Y. C. Zhang, S. Yu et al., “High-Speed Implementation of Length-Compatible Privacy Amplification in Continuous-Variable Quantum Key Distribution,” IEEE Photonics Journal, 10(3), (2018).
[15] B. Y. Tang, B. Liu, Y. P. Zhai et al., “High-speed and Large-scale Privacy Amplification Scheme for Quantum Key Distribution,” Scientific Reports, 9, 1-8 (2019).