ATLANTIS PRESS
Advances in Engineering Research (AER), volume 61
International Conference on Mechanical, Electronic, Control and Automation Engineering (MECAE 2017)

Fast Implementation of Privacy Amplification in Continuous-Variable Quantum Key Distribution

Jinpeng He, Xiangyu Wang, Song Yu [a]

State Key Laboratory of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China

[a]yusong@bupt.edu.cn

Keywords: Continuous-Variable Quantum Key Distribution, Privacy Amplification, Partition Matrix Method.

Abstract. By partition matrix method, we improve the speed of privacy amplification, the key phase of the post-processing in continuous-variable quantum key distribution system. The computational complexity of privacy amplification could be decreased by the partition matrix method. The optimal speed of the privacy amplification has been improved from 1.000 Mbit/s to 6.939 Mbit/s in our experimental tests.

1. Introduction

As a typical application in the field of quantum information, quantum key distribution (QKD) [1] shares common secret key between two legitimate users conventionally called Alice and Bob. Over the past few years, continuous-variable quantum key distribution (CV-QKD) [2-3] has received a lot of attention because of its standard telecom components. The post-processing plays a very important role in the CV-QKD system because its speed can influence the repetition rate of the CV-QKD system significantly. Privacy amplification [4-5] is a necessary post-processing step in the CV-QKD system. Owing to its computational complexity, Privacy amplification has a very slow speed. Slow speed of privacy amplification will slow down the speed of post-processing, which will cause the secure key rate to be very small. To solve this problem, in this paper, we focus on the method of partition matrix to improve the speed of privacy amplification. In privacy amplification, Faster Fourier Transform in the West (FFTW) [6] is used to accelerate the matrix multiplication and its computational complexity is $O(n\log_2 n)$ [6]. When the matrix is divided into blocks, the total computational complexity of FFTW decreases and the speed of privacy amplification increases. Therefore, the optimized speed could be found in our experiment.

2. Fast implementation of privacy amplification by using partition matrix method

2.1 Privacy amplification

The post-processing of the CV-QKD includes four processes: sifting, parameter estimation, information reconciliation and privacy amplification. In parameter estimation, we can get the secure key rate which is represented by k. And then we can get the key after error correction in information reconciliation. Finally, a further step called privacy amplification is required whose purpose is to get secret key from the key after error correction.

We introduce privacy amplification through a formula of $ST = Z$ in our experiment, where S is the key after error correction showing a length of n, T is Toeplitz matrix showing numbers of both rows and columns of n, and Z is the result of matrix multiplication which is called primary key. We get secret key from Z and the length of secret key is k which is estimated in parameter estimation. Here, FFTW is used to accelerate the matrix multiplication. As a result, the computational complexity of privacy amplification can be reduced from $O(n^2)$ to $O(n\log_2 n)$.

2.2 Partition matrix method

We know that the computational complexity of FFTW is $O(nlog_2n)$, which will get increasingly large if n increases. Bigger the length of key after error correction leads to slower speed of privacy amplification. Accordingly, in order to improve the speed, the method of partition matrix is applied and the detailed steps are given as follows.

Step 1: Firstly, we divide the key after error correction into p parts, of which part has a length of m (block size). Next, we divide Toeplitz matrix into p parts, with number of rows each part being m. According to $n = p*m$, each part of the key after error correction is represented by S_i and each part of hash matrix is represented by T_i, ($i=1,2,3\cdots p$).

Step 2: We can get Z_i by performing the matrix multiplication of S_i and T_i , ($i=1,2,3\cdots p$) and FFTW is used to accelerate the multiplication.

Step 3: By calculating, we get the primary key, $Z = Z_1 \oplus Z_2 \oplus Z_3 \oplus \cdots \oplus Z_p$.

We give an example to introduce the partition matrix method, n=4, p=2.

$$S = \begin{bmatrix} u & v & w & t \end{bmatrix}, T = \begin{bmatrix} a & e & f & g \\ b & a & e & f \\ c & b & a & e \\ d & c & b & a \end{bmatrix} \tag{1}$$

Firstly, S is divided into S_1 and S_2. $S = \begin{bmatrix} S_1 & S_2 \end{bmatrix}, S_1 = \begin{bmatrix} u & v \end{bmatrix}, S_2 = \begin{bmatrix} w & t \end{bmatrix}$. T is divided into T_1 and T_2.

$$T = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix}, T_1 = \begin{bmatrix} a & e & f & g \\ b & a & e & f \end{bmatrix}, T_2 = \begin{bmatrix} c & b & a & e \\ d & c & b & a \end{bmatrix}. \tag{2}$$

Secondly, we can get Z_i by performing the matrix multiplication of S_i and T_i, ($i=1,2$) $Z_1 = S_1T_1, Z_2 = S_2T_2$. and FFTW is used to accelerate the multiplication.

Finally, we get primary key with the corresponding bit of Z_1 and Z_2 carrying out modular two addition operation, $Z = Z_1 \oplus Z_2$.

3. Analysis of computational complexity

The computational complexity of FFTW gets gradually small as n decreases. Therefore, the computing speed of FFTW will be faster as the complexity reduces.

We are going to analyze the relationship between the total computational complexity of FFTW and p. The total computational complexity is represented by y.

$$p = x, y = (n/x)\log_2(n/x) + (n/x)\log_2(n/x) + \cdots + (n/x)\log_2(n/x) = n\log_2(n/x) \tag{3}$$

From the formula (3) listed above, we can know that larger the value of p leads to smaller the total computational complexity of FFTW.

From the Step 3 listed above, we can know that larger the value of p leads to bigger the computational complexity of primary key.

From Step 1, 2, 3, we can know that both the total computational complexity of FFTW and computational complexity of primary key contribute to the computational complexity of privacy amplification. With p increasing, the total computational complexity of FFTW decreases while the computational complexity of primary key increases. Thus a value of p or m can be found to minimize the computational complexity of privacy amplification, and in this case, the speed of privacy amplification is the fastest.

In this experiment, the value of n is $2.52*10^8$. For different m values, we test ten sets of speed and calculate the average speed of the ten sets of data. The results are shown in Table 1 and Fig. 1 We can find that when the value of m is $6.00*10^5$ on the computer in our experiment, the speed reaches to 6.939Mbit/s. Besides, detailed parameters of computer are listed. The parameter of CPU is "Intel (R) Core (TM) i7-4790K CPU @ 4.00GHz" and the value of RAM is "32.00GB" and operating system is 64bit. For the 10 groups of different m values, the p is 2520, 1260, 630, 420, 315, 252, 210.

Table 1. The speed of privacy amplification under different block size

Speed (Mbit/s) Value of m	The first group	The second group	The third group	The fourth group	The fifth group	The sixth group	The seventh group	The eighth group	The ninth group	The tenth group	average speed
100000	3.424	3.426	3.442	3.442	3.453	3.440	3.447	3.445	3.421	3.424	3.443
200000	4.507	4.503	4.510	4.529	4.549	4.551	4.551	4.539	4.548	4.545	4.533
400000	5.513	5.513	5.523	5.507	5.520	5.532	5.525	5.552	5.531	5.551	5.526
600000	6.982	6.916	6.915	6.911	6.930	6.911	6.856	6.990	6.996	6.991	6.939
800000	5.527	5.515	5.496	5.482	5.506	5.476	5.536	5.555	5.581	5.553	5.522
1000000	5.146	5.109	5.098	5.099	5.092	5.063	5.074	5.122	5.084	5.078	5.096
1200000	3.716	3.692	3.719	3.720	3.721	3.715	3.715	3.717	3.703	3.705	3.712

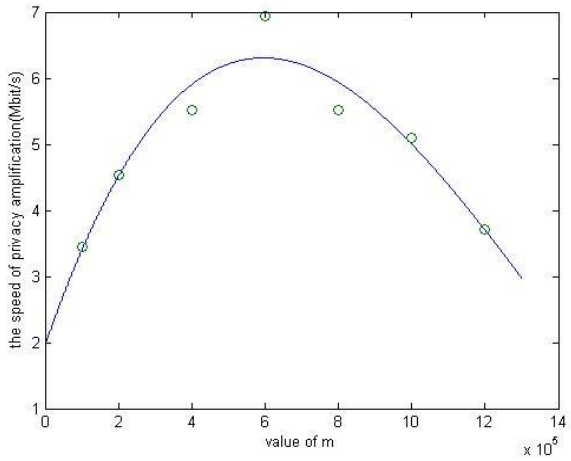

Fig. 1 Average speed of privacy amplification under different block size

4. Conclusion

In this paper, we improve the speed of privacy amplification by using the method of partition matrix and we present a detailed computational complexity analysis of the method. There is an optimal block size to achieve the optimal speed of privacy amplification in theory. In our experiment, the speed of privacy amplification is improved from 1.000 Mbit/s to 6.939 Mbit/s, and the optimal block size is $6.00*10^5$. The detailed parameters of computer are that the CPU is "Intel (R) Core (TM) i7-4790K CPU @ 4.00GHz", the value of RAM is "32.00GB" and operating system is 64bit.

References

[1]. V. Scarani, H. Bechmann-Pasquinucci, N. J. Cerf, M. Duek, N. Lütkenhaus, and M. Peev, "The security of practical quantum key distribution", Rev. Mod. Phys. 81 (2009), 1301-1350.
[2]. C. Weedbrook, S. Pirandola, N. J. Cerf, T. C. Ralph, J. H. Shapiro, S. Lloyd, "Gaussian quantum information", Rev. Mod. Phys. 84 (2012), 621-669.
[3]. P. Jouguet, S. Kunz-Jacques, A. Leverrier, P. Grangier, E. Diamanti, "Experimental demonstration of long-distance continuous-variable quantum key distribution", Nat. Photon. 7 (2013), 378-381.
[4]. C. H. Bennett, G. Brassard, "Generalized privacy amplification", IEEE T. Inform Theory. 41 (1995), 1915-1923.
[5]. C. H. Bennett, G. Brassard, "Privacy amplification by public discussion", SIAM J. Comput. 17 (1988), 210-229.
[6]. http://www.fftw.org/.