SCIENTIFIC REPORTS

OPEN

High speed error correction for continuous-variable quantum key distribution with multi-edge type LDPC code

Received: 10 January 2018
Accepted: 28 June 2018
Published online: 12 July 2018

Xiangyu Wang[1], Yichen Zhang[1], Song Yu[1] & Hong Guo[2]

Error correction is a significant step in postprocessing of continuous-variable quantum key distribution system, which is used to make two distant legitimate parties share identical corrected keys. We propose an experiment demonstration of high speed error correction with multi-edge type low-density parity check (MET-LDPC) codes based on graphic processing unit (GPU). GPU supports to calculate the messages of MET-LDPC codes simultaneously and decode multiple codewords in parallel. We optimize the memory structure of parity check matrix and the belief propagation decoding algorithm to reduce computational complexity. Our results show that GPU-based decoding algorithm greatly improves the error correction speed. For the three typical code rate, i.e., 0.1, 0.05 and 0.02, when the block length is 10^6 and the iteration number are 100, 150 and 200, the average error correction speed can be respectively achieved to 30.39 Mbits/s (over three times faster than previous demonstrations), 21.23 Mbits/s and 16.41 Mbits/s with 64 codewords decoding in parallel, which supports high-speed real-time continuous-variable quantum key distribution system.

Quantum key distribution (QKD)[1] allows two legitimate parties Alice and Bob to share unconditional security keys through an untrusted quantum channel and a classical authenticated channel, even if in the presence of an eavesdropper Eve. Many QKD protocols have been proposed since the first QKD protocol was proposed in 1984, they encode the key information on discrete variables (DV)[2,3] (such as the polarization or phase of single photon pulses) or continuous variables (CV)[4,5] (such as the quadratures of coherent states). Compared to DV protocols, CV protocols use homodyne detector or heterodyne detector to measure the quantum states, which have the advantage of using standard telecommunication technologies[6,7]. Recently, a new CV protocol design framework (LZG framework) has been proposed to allow one to design the protocol using arbitrary non-orthogonal states with their application scenarios[8]. CV-QKD protocols eliminate the limitation of single photon detector and have more advantages in practical QKD protocols.

For a practical Gaussian-modulated coherent state CV-QKD system, the speeds of information reconciliation and privacy amplification have an important influence on the secret key rate, and the efficiency of information reconciliation affects the secret key rate and transmission distance[9–11]. High speed and high performance reconciliation has been studied in DV-QKD[12,13]. High speed privacy amplification has also been implemented[14]. However, the speed of information reconciliation still limits the performance of CV-QKD systems. Due to the raw keys of Alice and Bob are correlated Gaussian variables, some approaches[15,16] have been proposed to achieve excellent efficiency. Multidimensional reconciliation[16] obtains high efficiency at low signal-to-noise ratios (SNR) by rotating the Gaussian variables to construct an virtual binary input additive white Gaussian noise channel. The error correction performance of multi-edge type low-density parity check codes (MET-LDPC)[17,18] are close to Shannon limit. Multidimensional reconciliation and MET-LDPC codes can be combined to achieve excellent efficiency at low SNRs[9,11,19], which supports CV-QKD system. Thus, we mainly focus on accelerating the speed of information reconciliation.

[1]State Key Laboratory of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing, 100876, China. [2]State Key Laboratory of Advanced Optical Communication Systems and Networks, School of Electronics Engineering and Computer Science, Center for Quantum Information Technology, Center for Computational Science and Engineering, Peking University, Beijing, 100871, China. Correspondence and requests for materials should be addressed to Y.Z. (email: zhangyc@bupt.edu.cn) or S.Y. (email: yusong@bupt.edu.cn)

As previously described, information reconciliation contains two processes: multidimensional reconciliation and error correction with MET-LDPC codes. The computational complexity of the first process is low, which can achieve high speed on central processing unit (CPU). However, for decoding with CPU, the speed of the error correction process will be quite slow when MET-LDPC codes approach to the Shannon limit at low SNRs[20]. The main reasons are that: (1) the computational complexity of belief propagation decoding algorithm is high for long-block-length (on the order of 10^6) and low-code-rate (no higher than 0.1) MET-LDPC codes; (2) belief propagation decoding algorithm requires more iterations to converge at low SNRs. Several work have been proposed to speed up the error correction process. They achieve the decoding speed to 7.1 Mb/s[20] and 9.17 Mb/s[21] with LDPC codes based on graphic processing unit (GPU).

In this paper, we propose a high speed parallel multiple codewords MET-LDPC code error correction method based on GPU. We optimize the memory structure of parity check matrix, making the decoding process more efficient. We modify the belief propagation decoding algorithm, which reduces computational complexity. This work has been applied to the longest field test of a CV-QKD system and achieves secure key rates two orders-of-magnitude higher than previous field test demonstrations[10].

Results

Information reconciliation for CV-QKD system. Information reconciliation is an efficient way for Alice and Bob to distill common corrected keys from their related variables. In a Gaussian-modulated coherent state CV-QKD system, the raw keys of Alice and Bob are continuous variables which cannot directly use the channel coding technology to correct errors between them. To solve this problem, several work have been done to extract common string from Gaussian variables. Sign reconciliation[22] encodes information on the sign of Gaussian variables. However, since most Gaussian values are close to 0 at low SNRs, it is difficult to distinguish the sign of Gaussian variables. In ref.[22], they only use high-amplitude data by post-selection, but this method discards a large number of small-amplitude data, which reduces the data utilization rate. Another method called slice reconciliation[15,23] divides Gaussian variables to different slices and then encodes information on the quantized slices. Due to the limitation of efficiency, this method is applicable to short distance CV-QKD system. In ref.[16], they rotate the Gaussian variables to construct a virtual binary input additive white Gaussian noise channel, then Alice and Bob's Gaussian variables will be converted to a binary string and the noise form of this binary string respectively. This method is called multidimensional reconciliation which is suitable for CV-QKD system.

Information reconciliation has two modes: direct reconciliation and reverse reconciliation[24]. Due to the limitation of 3 dB loss, the maximum transmission distance of direct reconciliation algorithm is 15 km when the optical fiber loss is 0.2 dB/km. However, reverse reconciliation algorithm can break this limit. In order to achieve long distance and high secret key rate of CV-QKD system, efficient error correction codes are required to distill secret keys from Alice and Bob's correlated Gaussian variables at low SNRs. MET-LDPC codes[18] are one of the error correction codes, which have well error correction performance even if at low SNRs.

For CV-QKD system, we combine multidimensional reconciliation and MET-LDPC codes to obtain excellent reconciliation efficiency at low SNRs by using reverse reconciliation protocols. Assuming that the Gaussian variables of Alice follow a zero mean and σ_X^2 variance Gaussian distribution $X \sim (0, \sigma_X^2)$, Bob's Gaussian variables $Y \sim (0, \sigma_Y^2)$ and the quantum channel noise $Z \sim (0, \sigma_Z^2)$, where $\sigma_Y^2 = \sigma_X^2 + \sigma_Z^2$, and $Y = X + Z$. In order to achieve effective error correction at low SNRs, Bob and Alice first use multidimensional reconciliation to convert their Gaussian variables Y and X to binary string U and the noise form V of this binary string. Then Alice and Bob correct their errors with MET-LDPC codes based on belief propagation decoding algorithm. Finally, they share a common binary string U with a certain probability. The secret key rate of CV-QKD system can be calculated by $k = \beta I(x; y) - S(y; E)$, where β is the efficiency of information reconciliation, $I(x; y)$ is the Shannon entropy of Alice and Bob, $S(y; E)$ is the Von Neumann entropy of Bob and Eve.

High speed error correction with MET-LDPC codes. High speed error correction is required to support real-time CV-QKD system. The error correction speed of MET-LDPC codes is related to the decoding algorithm, code length, the number of iterations, implementation method and other factors. For CV-QKD system, the error correction is quite difficult due to the low SNRs. Thus, we have to choose belief propagation decoding algorithm which iteratively updates message between variable nodes and check nodes to converge on valid codewords. The code length of a codeword is on the order of 10^6. When the MET-LDPC codes near to the Shannon limit, the reconciliation efficiency approaches to 1, the decoder needs more iterations to converge. In order to achieve high speed error correction at low SNRs, we implement the MET-LDPC decoder on GPU platform which supports to update the messages of variable nodes and check nodes in parallel. To maximize the parallel performance of GPU, we propose a method for simultaneously decoding multiple codewords. We also modify the belief propagation decoding algorithm and optimize the memory structure of parity check matrix to further accelerate the error correction process.

The decoding speed is extremely slow for long code length at low SNRs when we perform the decoder on CPU. Thus, we implement the MET-LDPC multiple codewords decoder on GPU with compute unified device architecture application programming interface developed by NVIDIA corporation[25]. The GPU-based parallel decoding process is shown in Fig. 1. We first copy the messages of permutated raw keys from host (CPU) to device (GPU). Then we initialize the messages of variable nodes and check nodes with kernel function on GPU. Next, we build two kernel functions to iteratively update messages of check nodes and variable nodes. It is not necessary to update all the variable nodes, the iteration process only update probabilities messages of all check nodes and the variable nodes whose degree is greater than 1 without making hard decisions. In our GPU-based decoding process, we ignore the variable nodes whose degree is equal to 1, this will reduce computational complexity and save a large number of threads. Without hard decision, the decoder will be simplified. After the maximum number of iterations is reached, the LDPC decoder calculates the probability messages of the variable nodes whose degree is

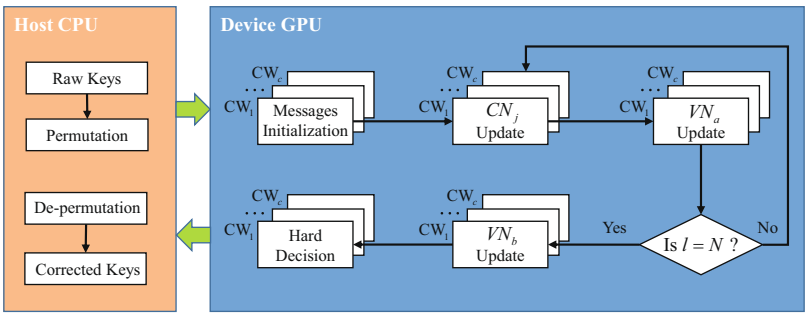

Figure 1. The process of GPU-based multiple codewords parallel decoding algorithm. CW_1, …, CW_c represent different codewords. c stands for the number of codewords decoded in parallel. CN_j represents the set of check nodes. VN_a and VN_b represent the sets of the variable nodes whose degrees are greater than 1 and equal to 1. l and N stand for the current number of iterations and maximum number of iterations, respectively.

equal to 1 and then performs hard decisions to get the decoded data and copy them from device to host. Finally, we de-permutate the decoded data to obtain the corrected keys.

For NVIDIA TITAN Xp GPU, the maximum number of thread blocks and threads per thread block on a grid are 65536 and 1024 respectively. Thus the maximum data that can be simultaneously decoded is 65536*1024 = 67108864. However, the block length of a codeword is 10^6 in our system. The parallel performance of the GPU can not be fully exploited when decoding with only one codeword. We can further accelerate the error correction speed by parallel decoding with multiple codewords. According to the parameters of GPU, we calculate that 64 codewords can be simultaneously decoded at most. Actually, since only the messages of the variable nodes whose degree is greater than 1 are updated, there are still a large number of threads that can be allocated when the messages of variable nodes are updated. For updating the check nodes messages, we reuse the threads that have been performed. Therefore, the number of simultaneous decoding codewords can be greater than 64. Theoretically, any number of codewords is possible as long as the GPU has enough memory.

The latency of global memory access has a significant impact on error correction speed. Coalesced global memory access can hide the latency. However, in order to obtain excellent reconciliation efficiency, the parity check matrix H is randomly constructed, where H is a two-dimensional matrix. And the block length of H is very long, we have to allocate the messages in global memory. When the decoder updates messages, the latency of non-consecutive global memory access of H limits the MET-LDPC error correction speed. No matter whether updating the variable nodes or the check nodes messages, the read and write access to global memory is non-consecutive because that both the variable nodes and check nodes of H are unorder. The latency can be hidden by optimizing the memory structure of H. We store H in two files, one of which stores variable nodes sequentially, and the other stores the mapping relations of variable nodes to check nodes. We can also swap variable nodes and check nodes. In this way, memory access for variable nodes will be consecutive. For simultaneous decoding of multiple codewords, the raw key permutation enables the memory access of check nodes to become consecutive. The kernels of GPU are performed by warps. A wrap contains multiple threads which perform the same program instruction in parallel, but with different data. Different type of GPU has different number of threads in a wrap. Typically, it is 32. If the threads inside a wrap access consecutive global memory, the latency will be hidden. Thus, when the number of simultaneously decoded codewords is an integer multiple of 32, both of the variable nodes and check nodes memory access are consecutive. Actually, when the latency of memory access equal to the latency of the messages update, the error correction speed is no longer improved by increasing the number of parallel codewords. By simultaneously decoding with multiple codewords based on GPU, the error correction speed is greatly improved, which supports high speed real-time CV-QKD system.

GPU-based error correction speed. We implement high speed error correction with multiple codewords based on GPU. For CV-QKD system, we choose low-code-rate MET-LDPC codes to correct error at low SNRs. Three typical code rates are designed in this work, *i.e.*, 0.1, 0.05 and 0.02, we all achieve high error correction speed on long block length and high iteration number. The block length of each code is 10^6. For different codes, the number of iteration are uncertain because that they apply to different distances (Actually, it is mainly affected by SNRs). We apply these three codes to correct errors when the SNR are 0.161, 0.075 and 0.029 and we set the iteration number to 100, 150 and 200, respectively. The degree distribution of these three codes are proposed in[19]. The parity check matrices are randomly constructed by progressive edge growth algorithm. In Fig. 2, we show the error correction speed of different number of codewords simultaneous decoding.

As shown in Fig. 2, when the number of codewords is less than 32, the error correction speeds increase rapidly. The main reason is that the latency is hidden by coalesced global memory access. When the threads in a wrap access non-consecutive global memory, the latency will be very long, even longer than updating the messages. Thus, the GPU-based decoder spends almost the same time when the number of codewords is less than 32. In other words, the total time is almost the same, either waiting for memory access or updating the messages. The error correction speed will be no longer improved by increasing the number of codewords when the access memory latency is the same as updating messages latency. Only by simplifying the decoding computational complexity can we further accelerate the error correction speed. As shown in Fig. 2, the error correction speed is almost no longer increased when the number of codewords is greater than 64. The requirement for CPU and GPU are too

Code Rate	0.1		0.05		0.02	
SNR	0.160		0.075		0.029	
β	93.40%		95.84%		96.99%	
Iterations	100		150		200	
FER	0.055		0.203		0.375	
Total Number of Edges	3,767,500		3,480,000		3,337,500	
Updated CNs	900,000		950,000		980,000	
Updated VNs	1,000,000	125,000	1,000,000	70,000	1,000,000	40,000
Ignored VNs[a]	0	875,000	0	930,000	0	960,000
Number of Edges to pass messages (CNs to VNs)[b]	3,767,500	2,892,500	3,480,000	2,550,000	3,337,500	2,377,500
Latency Per Iteration (ms)[c]	0.363	0.329	0.361	0.314	0.357	0.305
Error CorrectionSpeed (Mbits/s)[d]	27.54	30.39	18.49	21.23	14.00	16.41

Table 1. GPU-based error correction speed and error correction performance with 64 codewords parallel decoding. SNR: signal-to-noise ratio. β: reconciliation efficiency. [a]These VNs (variable nodes) are ignored only when the decoder performs the iterative process. Their messages will be computed before the hard decision process. [b]Because only the VNs have degree 1, the number of edges to pass messages would be reduced only when the messages pass from CNs (check nodes) to VNs. [c]The latency per iteration is an average for total decoding latency, including the latency of initialization, iterative message-passing, CNs and VNs updated, hard decision and memory copy between CPU and GPU. [d]The results are obtained on a NVIDIA TITAN Xp GPU.

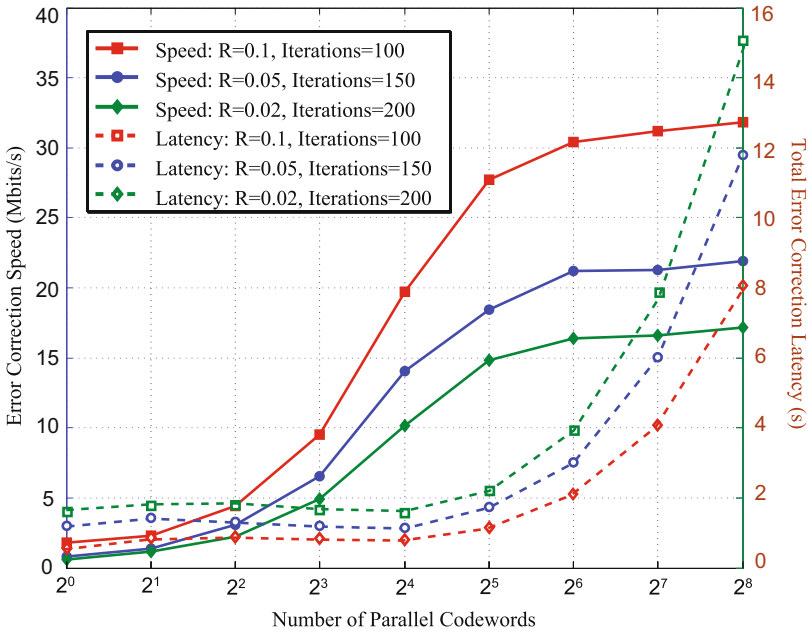

Figure 2. The error correction speeds and latency of different number of codewords to decode in parallel with rate 0.1, 0.05 and 0.02 code and the iterations are 100, 150, and 200 respectively. The solid lines refer to the error correction speed. The dotted lines refer to the total error correction latency. The block length is 10^6. The results are obtained on a NVIDIA TITAN Xp GPU.

much if the number of codewords is too large. After comprehensive consideration, we choose 64 codewords to decode in parallel.

Table 1 gives the GPU-based error correction speed and error correction performance of the three codes at low SNRs. For the rate 0.1, 0.05 and 0.02 codes with block length 10^6, we achieve the error correction speeds to 30.39 Mbits/s, 21.23 Mbits/s and 16.41 Mbits/s when the maximum number of iterations are 100, 150 and 200, respectively. The corresponding SNRs are 0.160, 0.075, and 0.029, the reconciliation efficiencies can be achieved to 93.4%, 95.84%, and 96.99% respectively. The frame error rate (FER) indicates the error correction performance of MET-LDPC codes, it refers to the failure probability of error correction. For the implementation of the three code rates, they are 0.055, 0.203, and 0.375. Moreover, the failure probabilities can be reduced by increasing the maximum number of iterations.

Refs	Code Type	Block Length	Average Iterations	Latency Per Iteration (ms)	Decoding Speed (Mbits/s)
Paul *et al.*[20]	Polar	2^{27}	1	18.386	7.3
Paul *et al.*[20]	MET-LDPC	2^{27}	100	1.477	7.1
Milicevic *et al.*[21]	QC-LDPC	2^{20}	78	1.466	9.17
This work	MET-LDPC	10^6	100	0.329	30.39

Table 2. Error correction speed comparison by different type of codes.

Discussion

We propose an experiment implement of GPU-based high speed error correction for CV-QKD system. A Multiple codewords parallel belief propagation decoder is presented to accelerate the iterative message-passing algorithm. For belief propagation decoding algorithm, the computational complexity of MET-LDPC codes originates from the number of connected edges between variable nodes and check nodes and the number of iterations. High error correction performance is required for CV-QKD system, we can not reduce the complexity by simplifying the decoding algorithm or shortening the block length. To reduce the computational complexity, we optimize the decoder by ignoring the variable nodes whose degree is equal to 1 when the decoder iteratively passes messages. These nodes do not affect message-passing. The messages of these variable nodes are computed after the iterative process. To hide the latency of the decoder, we modify the memory structure of parity check matrix so that the global memory access becomes consecutive.

As shown in Table 2, we compare the performance between the proposed GPU-based multiple codewords parallel decoding and the results obtained by other work with rate 0.1 code at SNR = 0.161. Paul *et al.* respectively obtain the speed to 7.1 Mbits/s with MET-LDPC code on GPU and 7.3 Mbits/s with Polar code on CPU[20]. The generator matrix of Polar codes have regular recursion structure. And the decoder is implemented by successive cancellation algorithm, which does not require iteration. However, the Polar decoder can not be implemented on GPU because that the nodes are associated when using successive cancellation algorithm. Milicevic *et al.* obtain the speed to 9.17 Mbits/s with quasi-cyclic (QC) LDPC codes and early termination of the iteration process[21]. QC-LDPC codes simplify the randomness connection of parity check matrix. However, the error correction performance will be decreased when the expansion factor is too large. The early termination scheme is an efficient way to reduce the complexity of LDPC decoder and avoids unnecessary iterations. On the contrary, the complexity of decoder will be increased if we use the early termination scheme to multiple codewords parallel decoding because that the early termination condition of each codeword is different. The error correction speed we achieved is over three times faster than previous demonstrations, which is supporting high speed real-time continuous-variable quantum key distribution system[10].

Methods

Belief propagation decoding algorithm of LDPC code. LDPC coeds[18,26] are block error correction codes with a sparse parity check matrix proposed by Gallager in 1962. Its error correction performance is close to Shannon limit. MET-LDPC codes[18] are generalization form of LDPC codes, which has better error correction performance even if at low SNRs. Typically, LDPC code is defined by parity check matrix H of size $m \times n$, $m < n$. The code rate is defined by $R = (n - m)/n$. LDPC codes also can be represented by bipartite factor graphs[27]. For a parity check matrix, m represents the number of check nodes and n represents the number of variables nodes. The variable nodes and check nodes are connected by edges. We use progressive edge-growth method[28] to construct parity check matrix based on the degree distribution proposed in[19]. The MET-LDPC code decoding algorithm we used is belief propagation which iteratively propagates message between variable nodes and check nodes to converge on valid codewords until the decoding termination condition is satisfied or reaching to the maximum number of iterations. The belief propagation decoding algorithm in the reverse reconciliation postprocessing of CV-QKD system is described as follows.

Let R_j be the set of variable nodes that are connected to the jth check node, C_i be the set of check nodes that are connected to the ith variable node, $R_j \backslash i$ be the set R_j excludes i, $C_i \backslash j$ be the set C_i excludes j, q_{ij} be the message passed from ith variable node to jth check node, r_{ji} be the message passed from jth check node to ith variable node.

Step 1: Bob calculates the syndromes S_B of his binary string that is achieved by multidimensional reconciliation and sends the syndromes to Alice.

Step 2: Alice calculates the initialization probabilities q_{ij}^0 ($i = 1, 2, \cdots, n, j = 1, 2, \cdots, m$) that binary input additive white Gaussian noise channel passes to variable nodes. The superscript represents the current number of iterations. Theoretically, since the information that we extract are binary strings, the initialization probabilities include the probability of 0 and 1. To simplify the computational complexity, we use the ratio of $q_{ij}^0(1)$ to $q_{ij}^0(0)$ to represents the initialization probability.

$$q_{ij}^0 = \frac{q_{ij}^0(1)}{q_{ij}^0(0)} \tag{1}$$

Step 3: Alice updates the messages of check nodes. For the jth check node and R_j, she calculates the messages that variable nodes pass to check nodes when the iteration number is l, $l = 1, 2, \ldots, N$, where N is the maximum number of iterations.

$$r_{ji}^{l} = \frac{r_{ji}^{l}(1)}{r_{ji}^{l}(0)} = \frac{1-t}{1+t} \tag{2}$$

$$t = \prod_{i' \in R_j \backslash i} \frac{1 - q_{i'j}^{l-1}}{1 + q_{i'j}^{l-1}} \tag{3}$$

Step 4: Alice updates the messages of variable nodes. For the ith variable node and C_i, she calculates the messages that check nodes pass to variable nodes when the iteration number is l.

$$q_{ij}^{l} = \frac{q_{ij}^{l}(1)}{q_{ij}^{l}(0)} = q_{ij}^{0} \prod_{j' \in C_i \backslash j} r_{j'i}^{l} \tag{4}$$

Step 5: Alice makes hard decisions. If $q_i^l > 1$, the codeword $c_i = 1$, otherwise $c_i = 0$. Alice calculates the syndrome S_A of codeword c, such that $S_A = Hc^T$. If S_A is equal to S_B or reaching to the maximum number of iterations, the decoding is ended, otherwise repeat step 3 to step 5.

$$q_i^{l} = \frac{q_i^{l}(1)}{q_i^{l}(0)} = q_{ij}^{0} \prod_{j \in C_i} r_{ji}^{l} \tag{5}$$

We can use log-likelihood ratios to represent the probabilities messages. This decoding algorithm converts a large number of multiplication into addition, which reduces the computational complexity of belief propagation algorithm. A lookup table can be built to accelerate the process of updating the messages of log-likelihood ratios.

References

1. Gisin, N., Ribordy, G., Tittel, W. & Zbinden, H. Quantum cryptography. *Rev. Mod. Phys.* **74**, 145 (2002).
2. Bennett, C. H. & Brassard, G. Quantum cryptography: Public key distribution and cointossing. *IEEE International Conference on Computers, Systems, and Signal Processing*. 175–179 (1984).
3. Ekert, A. K. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.* **67**, 661 (1991).
4. Grosshans, F. & Grangier, P. Continuous variable quantum cryptography using coherent states. *Phys. Rev. Lett.* **88**, 057902 (2002).
5. Weedbrook, C. *et al*. Quantum cryptography without switching. *Phys. Rev. Lett.* **93**, 170504 (2004).
6. Weedbrook, C. *et al*. Gaussian quantum information. *Rev. Mod. Phys.* **84**, 621 (2012).
7. Diamanti, E. & Leverrier, A. Distributing secret keys with quantum continuous variables: principle, security and implementations. *Entropy* **17**, 6072–6092 (2015).
8. Li, Z., Zhang, Y. C. & Guo, H. User-defined quantum key distribution. Preprint at https://arxiv.org/abs/1805.04249 (2018).
9. Jouguet, P., Kunz-Jacques, S., Leverrier, A., Grangier, P. & Diamanti, E. Experimental demonstration of long-distance continuous-variable quantum key distribution. *Nat. Photonics* **7**, 378–381 (2013).
10. Zhang, Y. C. *et al*. Continuous-variable QKD over 50 km commercial fiber. Preprint at https://arxiv.org/abs/1709.04618 (2017).
11. Jouguet, P., Kunz-Jacques, S. & Leverrier, A. Long-distance continuous-variable quantum key distribution with a Gaussian modulation. *Phys. Rev. A* **84**, 062317 (2011).
12. Martinez-Mateo, J., Elkouss, D. & Martin, V. Key Reconciliation for High Performance Quantum Key Distribution. *Sci. Rep.* **3**, 1576 (2013).
13. Dixon, A. R. & Sato, H. High speed and adaptable error correction for megabit/s rate quantum key distribution. *Sci. Rep.* **4**, 7275 (2014).
14. Wang, X., Zhang, Y., Yu, S. & Guo, H. High-Speed Implementation of Length-Compatible Privacy Amplification in Continuous-Variable Quantum Key Distribution. *IEEE Photonics Journal* **10**, 1–9 (2018).
15. Lodewyck, J. *et al*. Quantum key distribution over 25 km with an all-fiber continuous-variable system. *Phys. Rev. A* **76**, 042305 (2007).
16. Leverrier, A., Alléaume, R., Boutros, J., Zémor, G. & Grangier, P. Multidimensional reconciliation for a continuous-variable quantum key distribution. *Phys. Rev. A* **77**, 042325 (2008).
17. Richardson, T. & Urbanke, R. Multi-edge type LDPC codes. Presented at Workshop honoring Prof. Bob McEliece on his 60th birthday, California Institute of Technology, Pasadena, California, 24–25 (2002).
18. Richardson, T. & Urbanke, R. *Modern Coding Theory* (Cambridge University Press), Chap. 7 (2008).
19. Wang, X. *et al*. Efficient rate-adaptive reconciliation for continuous-variable quantum key distribution. *Quantum Inf. Comput.* **17**, 1123–1134 (2017).
20. Jouguet, P. & Kunz-Jacques, S. High performance error correction for quantum key distribution using polar codes. *Quantum Inf. Comput.* **14**, 329–338 (2014).
21. Milicevic, M., Chen, F., Zhang, L. M. & Gulak, P. G. Quasi-cyclic multi-edge LDPC codes for long-distance quantum cryptography. *NPJ Quantum Information* **4**, 1–9 (2018).
22. Silberhorn, C., Ralph, T. C., Lutkenhaus, N. & Leuchs, G. Continuous variable quantum cryptography: beating the 3 dB loss limit. *Phys. Rev. Lett.* **89**, 167901 (2002).
23. Jouguet, P., Elkouss, D. & Kunz-Jacques, S. High-bit-rate continuous-variable quantum key distribution. *Phys. Rev. A* **90**, 042329 (2014).
24. Grosshans, F. Quantum key distribution using gaussian-modulated coherent states. *Nature* **421**, 238–241 (2003).
25. Compute Unified Device Architecture Programming Guide, Nvidia Inc., Santa Clara, CA (2007).
26. Gallager, R. Low-density parity-check codes. *Inf. Theory, IRE Trans.* **8**, 21–28 (1962).
27. Tanner, R. A recursive approach to low complexity codes. *IEEE Trans. Inf. Theory* **27**, 533–547 (1981).
28. Hu, X. Y., Eleftheriou, E. & Arnold, D. M. Regular and irregular progressive edge-growth tanner graphs. *IEEE Trans. Inf. Theory* **51**, 386–398 (2005).

Acknowledgements
This work was supported by the Key Program of National Natural Science Foundation of China under Grant 61531003, the National Natural Science Foundation under Grant 61427813, the National Basic Research Program of China (973 Program) under Grant 2014CB340102, and the Fund of State Key Laboratory of Information Photonics and Optical Communications.

Author Contributions
H.G. and S.Y. proposed and guided the work. X.W. and Y.Z. designed and performed the experiment. All authors analysed the results and wrote the manuscript.

Additional Information
Competing Interests: The authors declare no competing interests.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.