Improvement on private database queries based on the quantum key distribution

D. S. SHEN, X. C. ZHU[a*], W. P. MA, X. R. YIN, M. L. WANG
State Key Laboratory of ISN, Xidian University, Xi'an, China
[a]*School of Information Management, Wuhan University, Wuhan, China*

This paper presents a scheme to improve the private database query protocols proposed by Jakobi and Gao respectively. By randomly choosing permutation functions to generate auxiliary strings, we reduce the length of the raw key from kN to N, which makes a significant decrease in communication complexity and saves quantum resources. Furthermore, the whole protocol can avoid being restarted when the communication fails due to the user obtaining no bit in the finial key. The communicating parties can obtain a new finial key and successfully achieve communication by randomly rechoosing permutation functions. This is quite different from the previous protocols. The detailed analysis shows that our protocol can provide better communication complexity without loss of security and privacy. Compared with the previous protocols, our protocol is more practical and flexible.

(Received March 11, 2012; accepted June 6, 2012)

Keywords: Private information retrieval, Quantum key distribution, Communication complexity, Permutation functions

1. Introduction

A Symmetrically Private Information Retrieval (SPIR) [1] protocol allows a database user to obtain information from a database in a manner that prevents the database from knowing which data is retrieved and the user from obtaining extra information from database besides what she or he wants to retrieve. It is a generalization of the private information retrieval problem (PIR) [2] which deals with user privacy alone. To make the whole problem more simple, the database which is in Bob's possession is modeled as an N-bit string and the user, Alice, wants to obtain the i^{th} bit in the string, such that i remains unknown to Bob. She knows in advance the address of that bit in Bob's database.

It has attracted much attention both in computer science and in quantum information [3,4]. Classically, there are several schemes for PIR [5-8] and SPIR [1]. Gertner [1] exhibited quite efficient share-randomness SPIR schemes. However, the necessity of shared randomness among multi-servers is a significant drawback, since information-theoretic security requires new shared randomness for each application of the scheme. User and data privacy are apparently in conflict: the most straightforward way to obtain user privacy is for Alice to have Bob send her the entire database, leading to no data privacy whatsoever. Conversely, techniques for guaranteeing the server's data privacy typically leave the user vulnerable. So far, no efficient solutions in terms of communication complexity [9] are known for SPIR. The communication complexity is measured by the number of bits transmitted between the user and the server per query. One might hope that quantum mechanics could solve this dilemma. Several quantum symmetrically private information retrieval (QSPIR) schemes [10-13] were proposed. The private information in Ref. [10] and [13] is directly encoded in qubits and in the phase respectively. Perfect though they are in theory, the two protocols are difficult to implement because when large database is concerned the dimension of the oracle operation will be very high.

Jakobi et al. [14] recently proposed a new protocol based on quantum key distribution (QKD), which is the first practical quantum private query protocol and quite different from the previous ones. The protocol can be divided into two phases: the raw key distribution phase and the post-processing phase. In the raw key distribution phase, The SARG04 QKD scheme [15] is utilized to generate asymmetric key served as a raw key between Alice and Bob. In the post-processing phase, the raw key string is diluted to generate the finial key which is used to encrypt the whole database after a relative shift. By introducing parameter θ which determines the sent photons in the first phase, Gao et al. [16] extended Jakobi et al's protocol [14] to a general form. For simplicity, suppose that Jakobi et al's protocol [14] and Gao et al's protocol [16] are shortened as JQP and GQP respectively. The main idea underlying GQP is completely identical to JQP with the only difference that the sent photons are adjustable.

In these two protocols, kN-bit raw key must be created in order to generate a diluted finial key in the post-processing phase. For example, when $N = 10^6, k = 6$ will be appropriate in JQP. That means that the length of the created raw key must be 6×10^6. That is to say, the raw key string must be k times longer than the database. The security parameter k increases with the increase of N. Obviously, the value of the parameter k affects the communication complexity. They can obtain a smaller k and achieve lower communication complexity by choosing a smaller $\theta(0 < \theta < \pi/4)$ in GQP. However, the user privacy decreases with decreasing the parameter $\theta(0 < \theta < \pi/2)$. Furthermore, if they pursue $k = 1$, θ will be very small for large N. That will not only reduce the user privacy but also make its realization technically difficult. From this we know that we can not reduce the communication complexity effectively only by adjusting the value of θ. In a word, both JQP and GQP can achieve very good levels of privacy and security which relies on fundamental physical principles, but neither can provide a good communication complexity. Especially for a large database, the drawback greatly affects communication efficiency and its practical application.

To achieve lower complexity and minimize the communication cost, we present an effective scheme in which pseudo-random strings generated by random permutation functions are cleverly used to dilute the raw key in the post-processing phase. We can achieve $k = 1$ no matter how large the database is. The detailed analysis is given in the following sections.

2. Review of GQP

The basic idea of JQP [14] and GQP [16] is that it uses QKD in combination with adequate post-processing to generate an N-bit string K^f that serves as an oblivious key [17] for a database of N bits. For this purpose, the key K^f must meet the following conditions: (1) Bob knows the key entirely; (2) Alice knows only a few bits of K^f-ideally exactly one (database security); (3) Bob does not know which bits are known to Alice (user privacy). Bob adds database and key bit-wise with a relative shift chosen by Alice and sends her the encrypted database. The relative shift is needed in order to ensure that Alice's bit of interest is encoded with an element of K^f she knows, so that she can decipher the bit and thus receive the answer to her private query.

Since GQP is a generalization of JQP, we will give a brief description of GQP and then present a technique to improve it.

Step 1, Bob sends Alice a long random sequence of qubits (e.g., photons) which are in states $|0\rangle$, $|1\rangle$, $|0'\rangle$ and $|1'\rangle$. States $|0\rangle$ and $|1\rangle$ code for 0, and states $|0'\rangle$ and $|1'\rangle$ correspond to bit value 1. Here

$$|0'\rangle = \cos\theta|0\rangle + \sin\theta|1\rangle,$$

$$|1'\rangle = \cos\theta|0\rangle - \sin\theta|1\rangle.$$

The parameter $\theta \in (0, \pi/2)$ can be selected continuously according to particular situations, which will be demonstrated below.

Step 2, Alice measures each state in the basis $B = \{|0\rangle, |1\rangle\}$ or $B' = \{|0'\rangle, |1'\rangle\}$ basis at random. Obviously this measurement does not allow her to infer the value of the bit sent by Bob.

Step 3, Alice announces what she has successfully detected; lost or not detected photons are disregarded. It does not allow Alice to cheat since she still has no information on the sent bit values and can't tell which measurement result she wants. As a consequence, the protocol is completely loss-independent.

Step 4, for each qubit that Alice has successfully measured, Bob announces one bit 0 or 1, where 0 represents the bit which is originally in the state $|0\rangle$ or $|0'\rangle$, while 1 implies the qubit is $|1\rangle$ or $|1'\rangle$.

Step 5, she interprets her measurement results of step 4. According to the basis she has chosen and the result she has obtained, Alice can obtain the sent bit with a certain probability. For example, if $|0'\rangle$ has been sent and 0 has been declared, Alice can rule out $|0\rangle$ only if she has measured in the basis B and obtained the result $|1\rangle$. Then she can conclude that the state is $|0'\rangle$ and the bit value is 1. As a result, direct measurement as under step 2 will yield $p = \sin^2\theta/2$ of conclusive results and $1-p$ of inconclusive ones. Both conclusive and inconclusive results are kept. So Alice and Bob now share a raw key K^r which is known entirely to Bob and partly to Alice (she knows $p = \sin^2\theta/2$ of the whole).

Step 6, they both execute postprocessing to the key. The raw key K^r must be of length $k\times N$ (with k being a security parameter). By cutting it into k substrings of length N and adding them bitwise, both parties generate an N-bit string of which Alice knows a few bits at most (ideally exactly one).

Step 7, if Alice is left with no known bit after step 6, the protocol has to be restarted. The probability for this to occur can be kept small.

Step 8, Alice will know at least one element of it after K^r has been established correctly. Suppose she knows the j^{th} bit K_j^f and wants to obtain the i^{th} bit of the database X_i. In order to decrypt correctly Alice announces the number $s = j - i$ in advance to allow Bob to encode the database by bitwise adding K^f, shifted by s. This will make sure that the bit in which Alice is interested is coded with a key element she knows so that the private query can be completed.

They show that, the average number of the key bits Alice obtains can be located on any fixed value the users wanted for any database size by adjusting the value of θ. And the parameter k is generally smaller (even $k = 1$ can be achieved) when θ < π/4, which implies lower complexity of both quantum and classical communications. The scheme is completely loss-resistant since the discarded bits contain no information about database or the finial key. It achieved good database security and high user privacy.

3. The improved scheme

The above protocol can be divided into two phases: the raw key distribution phase and the post-processing phase. Step 1 to 5 of the above protocol are similar with B92 QKD protocol [18]. This remains in this paper and only post-processing is modified.

According to the above protocol, the raw key string must be k times longer than the database. This means that Bob has to send at least $k\times N$ photons to Alice. For a given N, we can achieve a smaller k by choosing a smaller parameter θ. But, as discussed in Ref. [16], the smaller the parameter θ, the higher the probability with which Bob can correctly guess the address of Alice's query. Furthermore, if they pursue $k=1$, which means the optimal communication complexity in our protocol, θ will be very small for large N. This might make its realization technically difficult. Therefore, $k>1$ is needed when N is large in their protocol. From this we know that we can not effectively reduce the communication complexity only by adjusting the value of θ.

On the other hand, the whole protocol has to be restarted if the finial key string is generated with no bit known for Alice. It, to a certain extent, also affects the efficiency of communication.

To improve it, we present a simple technique in which permutations are cleverly used to generate new strings to dilute the raw key. The new protocol is described as follows. Step 1 to 5 are the same as those in GQD above.

Step $6'$, when the created string is of length N, Bob stops sending photos. Let us denote the raw key as X.

Step $7'$, Bob randomly chooses $t-1$ different permutation functions $Y_j = f_j(X)$ and announces them. Then they both perform these permutation functions on X and obtain $t-1$ auxiliary strings $Y_j, j = 1,\cdots,t-1$. They add bitwise the $t-1$ auxiliary strings to the raw string and obtain a diluted finial key. In order to dilute the raw key effectively, these permutation functions should not be identical. This step is very important for this paper and the details can be seen in the following security analysis.

Step $8'$, if Alice is left with no known bit after step $6'$, the whole protocol does not have to be restarted. They can return to step $7'$. By choosing randomly other permutation functions, they can generate new $t-1$ auxiliary strings Y_j' and obtain a new diluted finial string. Otherwise, continue.

Step $9'$, it is the same as step 8 in section 2 above.

4. Discussion

Apparently, compared to GQP, the stage of key distribution keeps unaltered and only post-processing phase is improved in our scheme. As discussed in JQP and GQP, Alice is always confronted with the problem of discriminating two non-orthogonal quantum states, and will hence always have incomplete knowledge on the raw key. This lack of information is subsequently further amplified by step $7'$.

In JQP and GQP, we may regard the first N-bit substring as an object string, and the remaining $k-1$ substrings as auxiliary strings which are used to dilute the object string. So the inconclusive bits in the auxiliary strings play a key role in the whole process. As designed in JQP and GQP, the k substrings are truly random. In our scheme, by performing different permutation functions f_j on the N-bit object string, we obtain $t-1$ permutated strings served as auxiliary strings. The permutated strings are pseudo-random. At first sight the permutated strings seem not good enough since they are not truly random. However, it should be noted that the core of the post-processing is how to dilute the key. Compared with the specific value of a conclusive bit in the finial key, we are more concern about how many bits have been diluted, which is very important for our protocol. In fact, the following detailed analysis can show that it does work well.

On one hand, the amount of information Bob can obtain about the finial key in our protocol is the same as that in GQP. Bob knows which bits in the auxiliary strings will contribute a bit in the finial key. This is the same as that in GQP, there is only difference that a raw bit will contribute t finial key bits. Although he knows that the relationship exists among the bits in the finial key. Since he can not conclude a raw bit that Alice knows without any error due to Alice's random measurement, he can not know precisely the corresponding bit in the finial key.

On the other hand, the amount of information Alice can obtain from the finial key in our protocol is almost the same as that in GQP. As the bits in the raw object string are random, no matter what the permutation functions f_j are, Alice can not obtain more information about each bit from the permutated strings. Every inconclusive bit in the permutated auxiliary strings will also make the corresponding bit in the finial key inconclusive. That is to say, for Alice, the degree of uncertainty about the inconclusive bits in the finial key in both protocols is almost the same. Owing to the permutation functions f_j which are randomly chosen by Bob, the object string can also be well diluted in our protocol.

Following the improved protocol, after adding bitwise the permutated strings to the object string, Alice will on average know $\bar{n} \approx N \times (\frac{\sin^2\theta}{2})^t$ bits, where the number n follows approximately a Poisson distribution. The probability that she does not know any bits at all and that the protocol must be restarted is $P_0 \approx (1-(\frac{\sin^2\theta}{2})^t)^N$. The result is very similar to that in GQP [16]. Obviously, t is equivalent to the previous k. For large N, we can also choosing an appropriate value of t to ensure both $\bar{n} << N$ and small P_0. The parameter t affects the computational complexity rather than the communication complexity. Obviously, the computational complexity increases with the parameter t. Compared to JQP and GQP in Ref [14] and [16], our improved protocol shows a little increase in computational complexity.

Only a N-bit raw key is long enough to generate an ideal final key in our improved protocol. That is to say, no matter how large the database is, we only need a raw key with the same length of database according to our protocol. It greatly saves the sent photons. For instance, for a database $N = 10^6$ and $\theta = \pi/4$, $k = 1$ and $t = 9$ is a choice providing Alice with $\bar{n} \approx 4$ elements of the final key on average whereas the probability of failure is only about 4%. According to JQP and GQP, $k = 9$ and 9×10^6 photons must be sent in a perfect quantum channel. While only 10^6 photons are enough in accordance with our protocol. It greatly saves photons and brings about a significant decrease in communication complexity. So our protocol provides us with the optimal communication complexity and is much more efficient.

There is an important advantage in our improved protocol. In GQP, Alice maybe be left with no known bit after step 6, the protocol has to be restarted, although the probability for this to occur can be kept small. If it happens, it will produce additional communication traffic. This situation can be avoided in our protocol. The communication participants can successfully achieve communication by repeating step $7'$. So our protocol is more flexible.

5. Error and security analysis

We now turn to error analysis and the question of which degree of privacy our protocol offers precisely. As an improved protocol, we will mainly discuss the security under the attacks the same as discussed in GQP.

5.1 Error analysis

Obviously in a practical QPQ protocol Alice's final key bits may be different from Bob's, which is caused by an outside eavesdropper's attack or channel noise. Since the value of a finial conclusive key bit affects whether Alice can accurately decode the bit she is interested in or not, the error bit rate should be paid attention to. At the same time it is necessary to make certain whether the error rate varies with parameter θ. So we will also explore the impact of the parameter on the error rate.

The value of a conclusive bit in the finial key is depended on the k raw contributing bits. So we will discuss the error rate of a raw bit at first.

Suppose that a state has been sent $|0'\rangle$ and $\{|0\rangle, |0'\rangle\}$ has been announced. Alice rules out the state $|0'\rangle$ and mistakes it for $|0\rangle$ if and only if she obtains the result $|1'\rangle$ which is orthogonal with the original state $|0'\rangle$ by measuring in the basis $\{|0'\rangle, |1'\rangle\}$. With noise present, the state will flip with a certain probability in the quantum channel. Let the probability of Alice obtaining the corresponding orthogonal state by measuring the sent state in the basis which it belongs to be δ. Obviously, the value of δ does not depend on the parameter θ, and is just determined by the noise in the quantum channel. So the value of θ will not affect the error in the communication. As Alice randomly chooses basis $B = \{|0\rangle, |1\rangle\}$ or $B' = \{|0'\rangle, |1'\rangle\}$ and the probabilities of the two events are equal, the error rate of a raw conclusive bit is $\frac{1}{2}\delta$. When the number of error bits is even, it does not affect the correctness of the finial result. So the error rate η of a finial conclusive bit satisfies:

$$(\frac{1}{2}\delta)^t < \eta < 1-(1-\frac{1}{2}\delta)^t$$

Since we still have no effective way to perform error correction or privacy amplification to achieve high correctness of the final key in such a special QKD protocol (that is, Alice only gets parts of the whole key and Bob does not know which bits are obtained by Alice). The error rate η to some extent reflects the correctness of the shared key bit.

5.2. Security analysis

We will consider its security from two aspects: database security and user privacy.

Let us first discuss database security. In general one must assume that Alice disposes of a quantum memory and is hence not forced to measure directly as in step 2. Instead she can keep the photons and doesn't measure them until step $7'$ ends. Having obtained the permutation functions $f_j, j = 1,\cdots,t-1$, she can conclude which bits contribute a finial bit. Then she performs measurement. As analyzed in GQP, Alice can also perform the optimal unambiguous state discrimination (USD) measurement [19], [20] and Helstrom's minimal error-probability measurement. For Alice, she is almost in the same situation in GQP and our protocol with the only difference that the t contributing raw bits are from the same string in our protocol. This does not make him better measure. So, under the same conditions, in our protocol we can achieve database security almost the same as that in GQP. We now consider user privacy. As we have discussed above, Bob randomly chooses permutation functions f_j . In fact, he can prepare them beforehand. This allows Bob to know which bits will act for a certain finial bit. This situation is just the same as that in GQP. As discussed in Ref.[16], Bob can not succeed in attacking without any error.

In addition, the smaller the parameter θ, the higher the probability with which Bob can correctly guess the address of Alice's query. So, we do not have to pursue a small θ since θ no longer affects the communication complexity in our protocol. We can choose an appropriate θ (e.g., $\theta = \pi / 4$), such that our protocol achieves both database security and user privacy. The security of it relies on fundamental physical principles (the impossibility to deterministically discriminate nonorthogonal states, and the impossibility of superluminal communication).

6. Conclusions

In this paper we have presented a scheme to improve JQP and GQP in terms of communication complexity. In order to retain the security and privacy in our protocol, the raw key distribution phase keeps unchanged. In the post-processing phase, by randomly choosing permutation functions on the object N-bit string to generate strings served as auxiliary strings, we reduce the length of the raw key from kN to N. This saves quantum sources and greatly reduces communication complexity of the protocol. Error analysis shows that the error rate is not affected by the parameter θ. In our protocol, we do not have to pursue a small θ since θ no longer affects the communication complexity. We can choose an appropriate θ such that our protocol achieves both good database security and high user privacy. Furthermore, if the communication fails due to that the user is left with no known bit in the finial key, we can obtain a new finial key only by repeating step $7'$. So, the protocol can avoid being restarted. In comparison with JQP and GQP, our protocol offers better communication complexity without loss of security and privacy. Therefore, our protocol is more practical and flexible.

Acknowledgments

We are grateful to the anonymous reviewer for helpful comments. This work was supported by National Science Foundation of China under grant No. 61072140 and 60773002, the 111 Project under grant No. B08038, and the Specialized Research Fund for the Doctoral Program of Higher Education under grant No. 20100203110003.

References

[1] Y. Gertner et al., J. Comput. Syst. Sci., **60**, 592 (2000).
[2] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan, Private information retrieval, J. ACM **45**(6), 965C (1998), Earlier version in Proc. FOCS95.
[3] I. Kerenidis and R. de Wolf, J. Comput. Syst. Sci., **69**, 395 (2004).
[4] I. Kerenidis and R. de Wolf, Info. Proc. Lett., **90**, 109 (2004).
[5] A. Beimel, Y. Ishai, E. Kushilevitz, J. Raymond, Breaking the O(n1/(2k.1)) barrier for information-theoretic private information retrieval, in: Proceedings of 43rd IEEE FOCS, 2002, pp. 261C.
[6] C.Gentry and Z. Ramzan, in Proc. 32nd ICALP (Springer- Verlag, Berlin,2005), p. 803;
[7] S.Yekhanin, Technical Report No. ECCC TR06-127, 2006.
[8] E.Kushilevitz and R. Ostrovsky, in Proc. 38th IEEE Symposium FOCS97 (1997), p. 364.
[9] A. Ambainis, in Proceedings of the 24th ICALP, Lect. Notes Comput. Sci. (Springer-Verlag, Berlin,), **1256**, 401 (1997).
[10] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett., **100**, 230502 (2008).

[11] V. Giovannetti, S. Lloyd, L. Maccone, IEEE Trans. Inf. Theo., **56**, 34465 (2010).

[12] F. De Martini, V. Giovannetti, S. Lloyd, L. Maccone, E. Nagali, L. Sansoni, and Fabio Sciarrino, Phys. Rev. A, **80**, 010302 (2009).

[13] L.Olejnik, Phys. Rev. A, **84**, 022313 (2011).

[14] M.Jakobi, C.Simon, N.Gisin,J-D.Bancal, Phys. Rev. A, **83**(2), 022301 (2011).

[15] V. Scarani, A. Ac´ in, G. Ribordy, N. Gisin, Phys. Rev. Lett., **92**, 057901 (2004).

[16] F.Gao, B.Liu and Q-Y.Wen, quant-ph/1111. 1511v1, (2011).

[17] D. Beaver, Lecture Notes in Computer Science, Vol. 963 (Springer, London, 1995), p. 97.

[18] C. H. Bennett, Phys. Rev. Lett. **68**, 3121 (1992).

[19] P. Raynal, e-print arXiv: quant-ph/0611133.

[20] U. Herzog and J. A. Bergou, Phys. Rev. A **71**, 050301 (2005).

*Corresponding author: xhongtan@yahoo.cn