Publicly Verifiable Blind Quantum Computation (Extended Abstract)

Kentaro Honda*
Department of Computer Science, School of Information Science and Technology, the University of Tokyo, Japan

Blind quantum computation protocols allow a user only having so limited quantum devices to delegate an intractable computation to a quantum server, keeping the computation perfectly secret. While some of them are verifiable, where the user can verify that given outcomes are correct, a third party cannot do and hence a cheating party will be able to get benefits illegally. We propose a new blind quantum computation protocol with a new property called public verifiability, which enables any third party to assure that a party does not benefit from its cheating.

a. Introduction If only she has a classical computer and so limited quantum devices such as a single qubit generator [3] or a measurement device [15], blind quantum computation (BQC) protocols [1–4, 6, 7, 11–16] enable Alice to delegate her computation to Bob's quantum computer with wonderful properties: correctness (she can receive the correct outcomes from honest Bob), universality (she can delegate any quantum computation), and blindness (even if Bob is evil, he learns nothing about her computation other than an upper-bound of its size). Some proposed BQC protocols have an additional property called unconditional verifiability [1, 3, 6, 7, 12, 13]. Even though she does not know whether Bob is honest or not and she cannot check directly the received outcomes, the property allows Alice to verify unconditionally that the outcomes are correct. The main idea for giving unconditional verifiability to a BQC protocol is as follows [6]. Alice secretly inserts independent and trivial parts, which are called traps, into her desired computation. Since the traps are trivial and independent, she knows their expected outcomes and she can detect cheating by checking whether the outcomes of traps match the expected ones. As the insertion was done secretly, Bob cannot know where the traps are, and therefore he cannot tamper with her computation avoiding being detected except for a small probability.

Unconditional verifiability gives Alice an ability to detect cheating. However, since the verification method essentially uses her private information, the property does not allow a third party to see through Bob's lie. Hence, when Alice claims to find cheating, a third party cannot verify her claim. The fact not only means theoretical incompleteness but also causes a practical problem. In a realistic setting, Alice has to pay a fee to delegate her computation to Bob. It is reasonable that Alice pays if and only if Bob returns the correct outcomes. Otherwise, evil Alice can obtain the correct outcomes without the payment or evil Bob can earn just by returning random values, in order words, without running his quantum computer. However, since any third party has no way to judge whether Alice receives the correct outcomes, the reasonable scheme cannot be realised. Of course, if there exists a trustworthy party, an unconditionally verifiable BQC protocol can be improved so that the dispute between Alice and Bob can be resolved: the trustworthy party chooses traps instead of Alice; the party secretly tells them to Alice before starting the protocol, and checks the outcomes of the traps after finishing the protocol. However, it is too difficult to find such a party and moreover there is no way to check whether a party is really reliable. Therefore, a challenge is to resolve the dispute without relying on the existence of a special party.

In this talk, we introduce a new property into a BQC protocol, which we call public verifiability. Public verifiability guarantees that a third party (say Justin) who has a classical computer can verify that a party does not benefit from cheating. We allow Justin to communicate with neither Alice nor Bob. He just observes classical communication between Alice and Bob, and finally judges which party cheats. Therefore, another party can recheck his judgement as long as the party has a classical computer and the communication log is kept.

Based on an existing unconditionally verifiable BQC protocol, we propose a new unconditionally verifiable BQC protocol. Our protocol still has correctness, universality, blindness, and unconditional verifiability, without any additional assumption. In this sense, our protocol can be said to be at least as good as the original protocol. Moreover, with the help of classical public-key cryptography, our protocol achieves computational public verifiability, provided that Alice has only the minimum quantum device, where computational public verifiability means that Justin can detect Bob's lie unconditionally and Alice's lie computationally. Our protocol is a more practical BQC protocol than others, and also shows how classical public-key cryptography enhances a quantum secure protocol.

b. Main idea Our idea for adding public verifiability to an unconditionally verifiable BQC protocol can be understood as a use of a computationally hiding and perfectly binding bit commitment scheme, which can be implemented using classical public-key cryptography. Bob commits to computation results by generating a public-key and a secret-key, encrypting the computation results, and announcing them and the public-key. Alice and Bob follow the original protocol, and then Alice announces the locations of traps and their expected outcomes. Bob checks whether his results of the traps are equal to the expected outcomes, and if so, he reveals the computation results by announcing the secret-key. Because of perfect binding, evil Bob cannot change the committed results after getting information about the traps. More-

over, since the commitment scheme is computationally hiding, it is computationally guaranteed that when evil Alice forges the expected outcomes of traps, she cannot learn the computation results. The method requires no extra party and, since the computational ability of Alice is so limited, classical public-key cryptography needs not to be very strong.

However, the idea faces a problem. On one hand, as any one-round BQC protocol needs exponentially growing numbers of qubits [18], Alice and Bob exchange messages multiple times during execution of a BQC protocol. It means that Alice has to send a message that depends on the previous messages from Bob. On the other hand, the messages from Bob contain partial information about computation results, and hence they should be hidden from Alice. We solve this dilemma by making Alice also encrypt her messages and assuming that classical public-key cryptography to be homomorphic. This solution allows malicious Bob to attack against Alice's secret: he may send an ill-formed public-key or message so that he can extract more information than Alice's desired message from an ill-formed message that Alice unawares sends. We also require cryptography to be secure against this kind of attacks. In summary, we require the following.

1. It is homomorphic sufficient to run our protocol.
2. Even if Bob sends ill-formed public-key and/or messages and Alice is unaware of the illegality, her messages encrypted by the public-key do not reveal any information other than the desired messages.
3. Given a pair of candidates of a public-key and a secret-key, Justin can confirm that the pair is the genuine one.
4. Whatever Alice knows about its plaintext in prior, a ciphertext gives her only negligible additional information about the plaintext under acceptable computational assumptions.

The requirement 3 is necessary for perfect binding. Note that although Alice has no quantum computer, the requirement 4 is stronger than usual semantic security claims because a plaintext is possibly related to the secret-key, which she compute using the power of Bob's quantum computer.

c. Results We apply the above idea to an unconditionally verifiable BQC protocol, Fitzsimons-Kashefi (FK) protocol [6]. Using the requirements 1–3, we prove that our protocol preserves the properties of original FK protocol: correctness, universality, blindness, and unconditionally verifiability. Especially, the probability of Alice detecting Bob's cheating is preserved. The requirements 1–3 need no assumption and so the preservation holds with no extra assumption.

As shown in the requirement 4, public verifiability needs some additional assumptions. In order to show the property, we put three assumptions: (i) Alice does not have any quantum device other than a single qubit generator, (ii) the number of qubits is polynomial in the security parameter, and (iii) the computational assumptions in the requirement 4. The second and the third assumptions are standard. The first one is used to derive the fact that Alice cannot set secretly a quantum channel between her and Bob, and cannot obtain the computation results directly. Using the requirements and the assumptions, we prove that our protocol achieves computational public verifiability.

Finally, we show the existence of classical public-key cryptography that can be used to give public verifiability to FK protocol. Specifically, we choose ElGamal cryptography [5], and use inattentive evaluations [17] to make it homomorphic. We prove this construction satisfies all of the above requirements.

d. Discussion In order to add public verifiability, we required Bob to commit to his computation results. Another choice is a commitment by Alice: before starting an original protocol, she commits to traps that she chooses, and she reveals them after the protocol. However, since she receives computation results before revealing the committed value, this method allows Alice to borrow the power of Bob's quantum computer without any restriction and she can use it to break binding. Hence, the method requires binding to be secure against such attacks. An unconditionally secure bit commitment scheme [8–10] is a possible solution and it seems that it achieves unconditional public verifiability. However, it requires that Alice and Bob each prepare agents in a distant place. The agents should be trusted, and if the agent of Alice betrays her, verifiability no longer holds. It should be emphasised that our protocol needs no such assumption for verifiability.

[1] Barz, S., Fitzsimons, J. F., Kashefi, E., and Walther, P., Nat. Phys. **9**, 727 (2013).
[2] Barz, S., Kashefi, E., Broadbent, A., Fitzsimons, J. F., Zeilinger, A., and Walther, P., Science **335**, 303 (2012).
[3] Broadbent, A., Fitzsimons, J., and Kashefi, E., in *Foundations of Computer Science, 2009. FOCS '09. 50th Annual IEEE Symposium on* (2009) pp. 517–526.
[4] Dunjko, V., Kashefi, E., and Leverrier, A., Phys. Rev. Lett. **108**, 200502 (2012).
[5] Elgamal, T., Information Theory, IEEE Transactions on **31**, 469 (1985).
[6] Fitzsimons, J. F. and Kashefi, E., "Unconditionally verifiable blind computation," (2013), arXiv:1203.5217.
[7] Giovannetti, V., Maccone, L., Morimae, T., and Rudolph, T. G., Phys. Rev. Lett. **111**, 230501 (2013).
[8] Kent, A., Phys. Rev. Lett. **83**, 1447 (1999).
[9] Kent, A., Journal of Cryptology **18**, 313 (2005).
[10] Kent, A., Phys. Rev. Lett. **109**, 130501 (2012).

[11] Mantri, A., Pérez-Delgado, C. A., and Fitzsimons, J. F., Phys. Rev. Lett. **111**, 230502 (2013).
[12] Morimae, T., Nat. Phys. **9**, 693 (2013).
[13] Morimae, T., Phys. Rev. A **89**, 060302 (2014).
[14] Morimae, T. and Fujii, K., Nat. Commun. **3**, 1036 (2012).
[15] Morimae, T. and Fujii, K., Phys. Rev. A **87**, 050301 (2013).
[16] Morimae, T. and Fujii, K., Phys. Rev. Lett. **111**, 020502 (2013).
[17] Sander, T., Young, A., and Yung, M., in *Foundations of Computer Science, 1999. 40th Annual Symposium on* (1999) pp. 554–566.
[18] Yu, L., Pérez-Delgado, C. A., and Fitzsimons, J. F., Phys. Rev. A **90**, 050303 (2014).

ACKNOWLEDGMENTS

We thank Takahiro Kubota, Tomoyuki Morimae, and Joseph Fitzsimons for insightful discussions. This work was supported by JSPS Grant-in-Aid for JSPS Fellows Grant No. 26 · 9148.

Publicly Verifiable Blind Quantum Computation (Full Version)

Kentaro Honda*
Department of Computer Science, School of Information Science and Technology, the University of Tokyo, Japan

Blind quantum computation protocols allow a user only having so limited quantum devices to delegate an intractable computation to a quantum server, keeping the computation perfectly secret. While some of them are verifiable, where the user can verify that given outcomes are correct, a third party cannot do and hence a cheating party will be able to get benefits unreasonably. We propose a new blind quantum computation protocol with a new property called public verifiability, which enables any third party to assure that a party does not benefit from its cheating.

I. INTRODUCTION

If only she has a classical computer and so limited quantum devices such as a single qubit generator [4] or a measurement device [19], blind quantum computation (BQC) protocols [1, 2, 4, 5, 7, 9, 15–20] enable Alice to delegate her computation to Bob's quantum computer with wonderful properties: correctness (she can receive the correct outcomes from honest Bob), universality (she can delegate any quantum computation), and blindness (even if Bob is evil, he learns nothing about her computation other than an upper-bound of its size). Some proposed BQC protocols have an additional property called unconditional verifiability [1, 4, 7, 9, 16, 17]. Even though she does not know whether Bob is honest or not and she cannot check directly the received outcomes, the property allows Alice to verify unconditionally that the outcomes are correct. The main idea for giving unconditional verifiability to a BQC protocol is as follows [7]. Alice secretly inserts independent and trivial parts, which are called traps, into her desired computation. Since the traps are trivial and independent, she knows their expected outcomes and she can detect cheating by checking whether the outcomes of traps match the expected ones. As the insertion was done secretly, Bob cannot know where the traps are, and therefore he cannot tamper with her computation avoiding being detected except for a small probability.

Unconditional verifiability gives Alice an ability to detect cheating. However, since the verification method essentially uses her private information, the property does not allow a third party to see through Bob's lie. Hence, when Alice claims to find cheating, a third party cannot verify her claim. The fact not only means theoretical incompleteness but also causes a practical problem. In a realistic setting, Alice has to pay a fee to delegate her computation to Bob. It is reasonable that Alice pays if and only if Bob returns the correct outcomes. Otherwise, evil Alice can obtain the correct outcomes without the payment or evil Bob can earn just by returning random values, in order words, without running his quantum computer. However, since any third party has no way to judge whether Alice receives the correct outcomes, the reasonable scheme cannot be realised. Of course, if there exists a trustworthy party, an unconditionally verifiable BQC protocol can be improved so that the dispute between Alice and Bob can be resolved: the trustworthy party chooses traps instead of Alice; the party secretly tells them to Alice before starting the protocol, and checks the outcomes of the traps after finishing the protocol. However, it is too difficult to find such a party and moreover there is no way to check whether a party is really reliable. Therefore, a challenge is to resolve the dispute without relying on the existence of a special party.

In this talk, we introduce a new property into a BQC protocol, which we call public verifiability. Public verifiability guarantees that a third party (say Justin) who has a classical computer can verify that a party does not benefit from cheating. We allow Justin to communicate with neither Alice nor Bob. He just observes classical communication between Alice and Bob, and finally judges which party cheats. Therefore, another party can recheck his judgement as long as the party has a classical computer and the communication log is kept.

Based on an existing unconditionally verifiable BQC protocol, we propose a new unconditionally verifiable BQC protocol. Our protocol still has correctness, universality, blindness, and unconditional verifiability, without any additional assumption. In this sense, our protocol can be said to be at least as good as the original protocol. Moreover, with the help of classical public-key cryptography, our protocol achieves computational public verifiability, provided that Alice has only the minimum quantum device, where computational public verifiability means that Justin can detect Bob's lie unconditionally and Alice's lie computationally. Our protocol is a more practical BQC protocol than others, and also shows how classical public-key cryptography enhances a quantum secure protocol.

II. FK PROTOCOL

Let us briefly review a verifiable BQC protocol Fitzsimons-Kashefi (FK) protocol [7], since our protocol is based on it. The protocol uses measurement-based quantum computation (MBQC) [22] and proceeds as follows.

(I) Alice selects a graph G and randomly the locations T of traps from the vertexes V. She also chooses uniformly randomly $\{d_i\}_{i\in N_G(T)}$ and $\{\theta_i\}_{i\in V}$ from $\{0,1\}$ and $\{\frac{k\pi}{4} \mid k=0,\dots,7\}$ respectively, where $N_G(T)$ is the neighbourhoods of T. Let $\{\phi_i\}_{i\in V}$ be her computational angle on G where $\phi_i=0$ for all $i\in T\cup N_G(T)$. She announces the graph G and sends Bob single qubit states $\{|q_i\rangle\}_{i=1}^{|V|}$ where

$$|q_i\rangle = \begin{cases} |d_i\rangle & (i\in N_G(T)) \\ \prod_{j\in N_G(i)\cap N_G(T)} Z^{d_j}|+_{\theta_i}\rangle & (i\notin N_G(T)) \end{cases} \tag{1}$$

and $|+_{\theta_i}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\theta_i}|1\rangle)$. Bob receives the qubits and applies controlled-Z gates to the pairs of qubits adjoining in the graph G.

(II) They repeat the following for all qubits.

(a) Alice chooses a random bit r_i and computes a measurement angle δ_i by

$$\delta_i = (-1)^{\sum_{j\in X_i}(b_j+r_j)}\phi_i + \theta_i + \pi r_i + \pi\sum_{j\in Z_i}(b_j+r_j) \pmod{2\pi} \tag{2}$$

where subscripts X_i, Z_i are sets determined by the graph.

(b) Alice sends δ_i to Bob.

(c) Bob measures the ith qubit with the angle δ_i and obtains the measurement result b_i.

(d) Bob sends b_i to Alice.

(III) Alice accepts if $b_t = r_t$ for all $t\in T$.

Note that the measurement angle δ_i depends on the previous measurement results $\{b_j\}_{j<i}$. In FK protocol, the random angles $\{\theta_i\}$ and the random bits $\{r_i\}$ make the measurement angles and the measurement results completely random respectively, but the angle adjustment (2) does FK protocol work correctly. Moreover, since a state of any neighbourhood of a trap qubit is $|0\rangle$ or $|1\rangle$, any trap qubit is separated from the other qubits. Hence, Alice knows that the measurement result b_t should be r_t. With a carefully chosen graph state and computation encoded in a fault-tolerant manner, FK protocol achieves universality and verifiability with an exponentially high probability.

III. RESULTS

a. Publicly verifiable BQC protocol Now, we show our protocol. Our protocol uses classical public-key cryptography. We do not specify cryptography here, but assume that before starting the protocol, Alice and Bob agree on which cryptography they use. Moreover, whenever receiving messages, Alice or Bob checks validity of the messages and aborts if they are found to be invalid. The protocol runs as follows.

(I) Alice selects a graph G and randomly the locations T of traps from the vertexes V. She also chooses uniformly randomly $\{d_i\}_{i\in N_G(T)}$ and $\{\theta_i\}_{i\in V}$ from $\{0,1\}$ and $\{\frac{k\pi}{4} \mid k=0,\dots,7\}$ respectively, where $N_G(T)$ is the neighbourhoods of T. Let $\{\phi_i\}_{i\in V}$ be her computational angle on G where $\phi_i=0$ for all $i\in T\cup N_G(T)$. She announces the graph G and sends Bob single qubit states $\{|q_i\rangle\}_{i=1}^{|V|}$ where

$$|q_i\rangle = \begin{cases} |d_i\rangle & (i\in N_G(T)) \\ \prod_{j\in N_G(i)\cap N_G(T)} Z^{d_j}|+_{\theta_i}\rangle & (i\notin N_G(T)) \end{cases} \tag{3}$$

and $|+_{\theta_i}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\theta_i}|1\rangle)$. Bob receives the qubits and applies controlled-Z gates to the pairs of qubits adjoining in the graph G. He generates a pair of a public-key pk and a secret-key sk, and sends pk.

(II) They repeat the following for all qubits.

(a) Alice chooses a random bit r_i and computes a measurement angle δ_i and its ciphertext δ_i^*.

(b) Alice sends the ciphertext δ_i^* to Bob.

(c) Bob decrypts it, measures the ith qubit with the angle δ_i, and obtains the measurement result b_i.

(d) Bob computes the ciphertext b_i^* of b_i and sends b_i^* to Alice.

(III) Alice announces T and the expected results $\{r_t\}_{t\in T}$. Then, Bob checks the associated results $\{b_t\}_{t\in T}$ and announces the secret-key sk if $b_t = r_t$ for all $t \in T$. She decrypts all received messages using the secret-key, and accepts if the secret-key is valid and $b_t = r_t$ for all $t \in T$.

(IV) As a public verification procedure, Justin accepts if the secret-key is valid and $b_t = r_t$ for all $t \in T$.

The big difference between our protocol and FK protocol is that Alice and Bob encrypt their messages and Alice reveals the traps. Intuitively, the changes give public verifiability to FK protocol for the following reasons. As she has a limited computational ability, the encryption of Bob's messages makes Alice unable to obtain the measurement results until Bob verifies that the traps are untouched and then she receives the secret-key. In order to obtain the secret-key, evil Alice has to announce the true traps and hence she cannot cheat, otherwise Bob aborts the protocol. Moreover, even if evil Bob accepts the measurement results dishonestly and reveals the secret-key, since he cannot change already public information, Alice and Justin recheck them with the revealed secret-key and can find the lie. Another explanation for the reasons is that we implement a computationally hiding and perfectly binding bit commitment scheme using classical public-key cryptography. Bob commits to the measurement results by sending ciphertexts of them, and reveals the committed results by announcing the secret-key. Perfect binding makes impossible for Bob to change the committed measurement results and computational hiding disables Alice to read out the measurement results until the reveal phase.

The encryption of messages raises questions about our protocol. How does Alice compute the ciphertext of the measurement angle (2) without the previous measurement results? Does not the encryption damage the security? We solve the questions by choosing cryptography carefully. We require cryptography to be homomorphic sufficient to compute (2) so that Alice can compute the ciphertext of the measurement angle, although she cannot compute the angle itself. Moreover, we have to protect blindness and verifiability from an attack of malicious Bob using ill-formed messages: malicious Bob sends ill-formed ciphertexts of measurement results to receive an ill-formed ciphertext that may contain more information than the measurement angle; he sends a false secret-key in order that Alice accepts incorrect outcomes wrongly. In summary, we require that classical public-key cryptography should satisfy the following.

1. Given the encrypted measurement results $\{b_j\}_{j<i}$, Alice can compute a ciphertext of the measurement angle δ_i, which is defined by the equation (2).
2. Even if Bob sends ill-formed public-key and/or messages and Alice is unaware of the illegality, her messages encrypted by the public-key do not reveal any information other than the desired angles.
3. Given a pair of candidates of a public-key and a secret-key, Justin can confirm that the pair is the genuine one.
4. Whatever Alice knows about its plaintext in prior, a ciphertext gives her only negligible additional information about the plaintext under acceptable computational assumptions.

We will see there exists suitable cryptography later.

b. Properties Thanks to the properties of FK protocol, our protocol has correctness, universality, blindness, and verifiability.

Theorem 1. *If classical public-key cryptography satisfies the requirements 1–3, then our protocol is correct, universal, blind, and verifiable.*

Proof. The requirement 1 enables Alice to compute the correct measurement angle δ_i, and preserves correctness. Since we do not restrict possible graphs or measurement angles, Alice can compute what she can in FK protocol. Next, we show blindness. In our protocol, the classical information that Bob obtains is G, $\{\delta_i^*\}$, $\{b_i\}$, T, and $\{r_t\}$. The requirement 2 ensures that δ_i^* contains no information than δ_i. In addition, since T and $\{r_t\}$ are chosen uniformly randomly, they are independent from Alice's computation. Therefore, Bob obtains no extra classical information, and blindness is preserved. Finally, we prove verifiability by showing the probability that honest Alice wrongly accepts in our protocol is equal to one in FK protocol. Fix the parameters of Alice $P \equiv (G, T, \{d_i\}, \{\theta_i\}, \{\phi_i\}, \{r_i\})$. We ignore parameters used in encryption for a moment. Assume that evil Bob succeeds in making Alice accept by sending $\{m_i\}$ where m_i is expected to be b_i^*. Then, he has to announce pk and sk, and the requirement 3 guarantees that they are a genuine key pair. Using the secret-key, Alice can decrypt all received messages and can verify the messages are valid. In short, Bob cannot send any ill-formed message to deceive Alice. Hence, m_i is a valid ciphertext b_i^* for any i and b_t is equal to r_t for any $t \in T$. Now, suppose that Alice starts FK protocol with parameters P. Then, if Bob sends $\{b_i\}$ to Alice during the protocol, he succeeds in cheating her. Conversely, if messages $\{b_i\}$ can be used in FK protocol to deceive Alice who chose parameters P, then $\{b_i^*\}$ also can be used in our protocol. The arguments does not depend on the choice of parameters in encryption and so they holds in any case. In summary, Bob's strategy to cheat Alice in our protocol can be used also in FK protocol just by making messages plaintexts, and vice versa. It leads that the probabilities of Bob successfully deceiving Alice in our protocol and FK protocol are the same. □

Now, we show that our protocol has public verifiability. Before that, we define public verifiability. Let Justin be a third party who has a classical computer and records all public classical information, but cannot send any message to another party. Intuitively, public verifiability claims evil party cannot influence his judgement in her/his flavour beyond a bound. Formally, a BQC protocol is (ϵ, δ)-publicly verifiable if

- the probability that Justin accepts but Alice does not obtain the correct outcomes is less than ϵ when she follows the protocol and
- whatever she knows and tries to obtain about the outcomes, with and without messages she obtains in the protocol, the difference in probabilities that Alice succeeds in obtaining it is less than δ when Justin rejects and Bob follows the protocol

when Justin follows the protocol and the computational ability of Alice does not exceed one of a probabilistic classical computer. Note that while computational ability of Alice is limited, she can borrow partially the computational power of Bob by delegating carefully chosen computation. We say unconditionally (ϵ, δ)-publicly verifiable if the condition about her computational ability is omitted.

Precisely speaking, public verifiability does not guarantee that evil party does not affect Justin. If evil Alice abandons her attempt to obtain computation results, she is able to make Justin reject wrongly. Moreover, if evil Alice delegates some computation to another quantum server, she can use the outcome to cheat Justin. However, in either case, she cannot benefit from the cheating. In the former case, she cannot get the computation results, and, in the latter case, she has to pay the server for the outcome, otherwise she cannot obtain it. Therefore, we ignore this kind of attacks. In other words, we assume that evil Alice always tries to both obtain computation results and make Justin reject, and that evil Bob always attempts to make Justin accept incorrect outcomes.

Now, we prove our protocol is publicly verifiable. For detection of Alice's lie, we put three additional assumptions:

(i) Alice does not have any quantum device other than a single qubit generator,

(ii) the number of qubits is polynomial in the security parameter, and

(iii) the computational assumptions in the requirement 4.

Note that (ii) and (iii) are standard assumptions.

Theorem 2. *Let ϵ be the probability that Bob succeeds in deceiving Alice in FK protocol, and δ be a negligible function. Under the above assumptions, if classical public-key cryptography satisfies the requirements 1–4, then our protocol is computationally (ϵ, δ)-publicly verifiable.*

Proof. Since what Justin does in the public verification procedure is the same as what Alice does, Justin can detect Bob's cheat if and only if Alice can detect the cheat, provided that Alice follows the protocol. The proof of the previous theorem shows the first condition of public verifiability is satisfied.

Suppose that Bob follows the protocol and that Alice obtains the outcomes but succeeds in making Justin reject. Since Bob is honest, when Bob announces the secret-key sk, Justin always accepts. Hence, Alice has to obtain the measurement results without sk. The assumption (i) rejects that Alice entangles Bob's system and her own system, and read the measurement results out, and therefore her strategy should be classical. The requirement 4 and the assumptions (ii), (iii) guarantee that information that Alice can extract is negligible. □

c. Cryptography We proved that computationally publicly verifiable BQC protocol exists if there exists cryptography having good properties. We show that such cryptography really exists.

We use bitwise encryption using ElGamal cryptography [6] with inattentive evaluations [23]. While ElGamal cryptography is known to be multiplicatively homomorphic, it is ill-defined in our setting. As we use bitwise encoding, it is enough for the computation (2) to evaluate a formula $(\alpha_1 \vee \alpha_2) \oplus (\alpha_3 \vee \alpha_4)$ where $\alpha_j \in \{0, 1, b_{X_i}, b_{Z_i}, \neg b_{X_i}, \neg b_{Z_i}\}$. In order to evaluate it, we use inattentive evaluations, which make secret evaluations of log-depth circuits possible using inductive construction. Furthermore, although ElGamal cryptography is proved to be semantic secure [25], it is not proved to be secure when a plaintext is related to a secret-key as far as we know. As inattentive evaluations enable Bob to encrypt every measurement results by different public-keys, we employ the update so that cryptography is secure against such an attack. That is, Bob encrypts the ith measurement result by the ith public-key and he sends the public-key together with the ciphertext. Precisely speaking, it deviates from the description of our protocol, but this modification obviously does not affect the proofs of the properties. Inattentive evaluations require Bob to send $(0, 1)$ or $(1, 0)$ instead of 0 or 1, respectively, so evil Bob sends ill-formed messages such as $(0, 0)$ and possibly obtains partial information about dependency of the above logical formula on b_{X_i} and b_{Z_i}. Although a non-interactive zero-knowledge proof [3] was employed in the original paper [23], we do not use it, because it needs an additional assumption that all parties share a common reference string. We modify inattentive evaluations such that Alice encodes a bit 0 or 1 using

the received message and uses it instead of the plain bit. It can make her message completely meaningless when once Bob sends an ill-formed message.

Now, we describe details.

- To generate the ith key-pair, Bob chooses a prime p_i where $2p_i+1$ is also a prime. Let G_i be the cyclic subgroup of $\mathbb{Z}_{2p_i+1}$ whose order is p_i. The ith public-key is a trio of p_i, a randomly chosen generator $g_i \in G_i$, and a randomly chosen element $g_i^{x_i} \in G$. The associated secret-key is $x_i \in \mathbb{Z}_{p_i}$.

- To encrypt a bit 0, Alice or Bob chooses a random value $r \in \mathbb{Z}_{p_i}$ and computes $0_i^* \equiv (g_i^{x_i r}, g_i^r)$. A ciphertext of a bit 1 is $1_i^* \equiv (g_i^m g_i^{x_i r}, g_i^r)$ where r and m are randomly chosen from $\mathbb{Z}_{p_i}$ and $\mathbb{Z}_{p_i}^*$ respectively.

- Alice can randomise a ciphertext $b_i^* = (g_i^l, g_i^r)$ even though she does not know the plaintext. She chooses random values y and z from $\mathbb{Z}_{p_i}^*$ and $\mathbb{Z}_{p_i}$ respectively, and computes a new ciphertext $((g_i^l)^y g_i^{x_i z}, (g_i^r)^y g_i^z)$. The ciphertext has the plaintext b but it is completely random.

Now, suppose Bob sends (b_j^*, c_j^*) as the jth measurement result where c_j is expected to be $\neg b_j$, and Alice tries to compute the ith measurement angle.

1. In the zeroth level, Alice encodes a bit into four pairs of bits: 0 or 1 into exactly three pairs of the four have the same or different bits, respectively. Explicitly, she creates $\mathbf{0}_j, \mathbf{1}_j, \mathbf{b}_j, \neg\mathbf{b}_j$ where

$$\mathbf{0}_j \equiv \{(b_j^*, 0_j^*), (c_j^*, 0_j^*), (b_j^*, b_j^*), (b_j^*, b_j^*)\} \tag{4}$$
$$\mathbf{1}_j \equiv \{(b_j^*, 0_j^*), (c_j^*, 0_j^*), (b_j^*, c_j^*), (b_j^*, c_j^*)\} \tag{5}$$
$$\mathbf{b}_j \equiv \{(b_j^*, 0_j^*), (b_j^*, 0_j^*), (b_j^*, b_j^*), (b_j^*, c_j^*)\} \tag{6}$$
$$\neg\mathbf{b}_j \equiv \{(c_j^*, 0_j^*), (c_j^*, 0_j^*), (b_j^*, b_j^*), (b_j^*, c_j^*)\}. \tag{7}$$

   To flip the bit, Alice exchanges the first elements in every pairs. Note that flipping $\mathbf{0}_j$, Alice obtains $\mathbf{1}_j$ in a different order. Hence, with a permutation, a bit flip correctly works.

2. In the first level, Alice uses an i-length bit sequence whose bit summation denotes its value. For b_{X_i}, she creates an i-length sequence $\{a_j\}_{j<i}$ where a_j is $\mathbf{b}_j$ if $j \in X_i$, and otherwise $\mathbf{0}_j$. To denote the negation of it, she flips just the value of a_0.

3. The encoding of the second level is the same as the zeroth level. She encodes $\alpha \vee \beta$ into $\{(\alpha, 0), (\beta, 0), (\alpha, \beta), (1, 0)\}$.

4. In the third level, she forms a pair of them as does in the first level.

5. Finally, she obtains the evaluation result of her desired formula. She randomises the result so that it does not reveal any information except its bit value. At each level, appropriate permutations and bit flips make an encoded bit completely random, preserving its bit value. Note that Alice cannot permute the elements in the first level. If she does, Bob cannot decrypt them.

Theorem 3. *The above construction satisfies all requirements*

Proof. Satisfying the requirement 1 is a straightforward consequence of a property of inattentive evaluations that all log-depth circuits can be evaluated. Since being a prime and being an element of a prime order cyclic group is computable in deterministic polynomial time, Alice can check validity of given public-keys and ciphertexts. Moreover, if Bob sends $(0_j^*, 0_j^*)$ or $(1_j^*, 1_j^*)$ instead of $(0_j^*, 1_j^*)$ or $(1_j^*, 0_j^*)$, then a message of Alice will be completely random. Indeed, $\mathbf{0}_j = \mathbf{1}_j = \mathbf{b}_j = \neg\mathbf{b}_j$ when $b_j = c_j$, and the bit flip of it does not change the form. Thus, our construction meets the requirement 2. The requirement 3 is obviously satisfied. Finally, we show the construction satisfies the requirement 4. The security of ElGamal cryptography [25] was proved only when the plaintext does not depend on the secret-key. Since a quantum computer computes discrete-logarithm efficiently [24], given a public-key of ElGamal cryptography, a quantum computer can compute its secret-key, and thus the measurement results possibly depend on the secret-key. However, since Bob encrypts the ith measurement result using the ith public-key, he can generate the public-key after obtaining the result, which is obviously independent from the public-key. Hence, from the security of ElGamal cryptography and inattentive evaluations [23], we conclude that the requirement 4 is satisfied. □

IV. DISCUSSION

We proposed a new verifiable BQC protocol based on FK protocol. Compared with FK protocol, our protocol preserves all properties of FK protocol without any additional assumption and furthermore has public verifiability. However, in order to achieve public verifiability, our protocol puts several assumptions and needs messages longer. In particular, to allow Justin to detect Alice's cheat, we assume that her quantum devices are only a single qubit generator. Indeed, we used the assumption to derive two facts: her computational ability is the same as a classical computer and she cannot extract non-negligible information about the measurement results from Bob's resource directly. Thus, the assumption can be weakened as far as the above two facts are derived, e.g. she could measure a qubit, have any single qubit device, and generate constant-size entangled qubits. Moreover, the second disadvantage also is not serious. Although the messages are long in our protocol, they are polynomial in the size of the messages in FK protocol. Furthermore, we can use homomorphic cryptography [8, 10, 21] and can reduce the size if we make blindness hold under some assumptions [11] or give up perfect blindness [5].

In order to add public verifiability, we required Bob to commit to his computation results. Another choice is a commitment by Alice: before starting an original protocol, she commits to traps that she chooses, and she reveals them after the protocol. However, since she receives computation results before revealing the committed value, this method allows Alice to borrow the power of Bob's quantum computer without any restriction and she can use it to break binding. Hence, the method requires binding to be secure against such attacks. An unconditionally secure bit commitment scheme [12–14] is a possible solution and it seems that it achieves unconditional public verifiability. However, it requires that Alice and Bob each prepare agents in a distant place. The agents should be trusted, and if the agent of Alice betrays her, verifiability no longer holds. It should be emphasised that our protocol needs no such assumption for verifiability.

[1] Barz, S., Fitzsimons, J. F., Kashefi, E., and Walther, P., Nat. Phys. **9**, 727 (2013).
[2] Barz, S., Kashefi, E., Broadbent, A., Fitzsimons, J. F., Zeilinger, A., and Walther, P., Science **335**, 303 (2012).
[3] Blum, M., De Santis, A., Micali, S., and Persiano, G., SIAM Journal on Computing **20**, 1084 (1991).
[4] Broadbent, A., Fitzsimons, J., and Kashefi, E., in *Foundations of Computer Science, 2009. FOCS '09. 50th Annual IEEE Symposium on* (2009) pp. 517–526.
[5] Dunjko, V., Kashefi, E., and Leverrier, A., Phys. Rev. Lett. **108**, 200502 (2012).
[6] Elgamal, T., Information Theory, IEEE Transactions on **31**, 469 (1985).
[7] Fitzsimons, J. F. and Kashefi, E., "Unconditionally verifiable blind computation," (2013), arXiv:1203.5217.
[8] Gentry, C., in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09 (ACM, New York, NY, USA, 2009) pp. 169–178.
[9] Giovannetti, V., Maccone, L., Morimae, T., and Rudolph, T. G., Phys. Rev. Lett. **111**, 230501 (2013).
[10] Goldwasser, S. and Micali, S., in *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82 (ACM, New York, NY, USA, 1982) pp. 365–377.
[11] Groth, J., Ostrovsky, R., and Sahai, A., in *Advances in Cryptology - EUROCRYPT 2006*, Lecture Notes in Computer Science, Vol. 4004, edited by S. Vaudenay (Springer Berlin Heidelberg, 2006) pp. 339–358.
[12] Kent, A., Phys. Rev. Lett. **83**, 1447 (1999).
[13] Kent, A., Journal of Cryptology **18**, 313 (2005).
[14] Kent, A., Phys. Rev. Lett. **109**, 130501 (2012).
[15] Mantri, A., Pérez-Delgado, C. A., and Fitzsimons, J. F., Phys. Rev. Lett. **111**, 230502 (2013).
[16] Morimae, T., Nat. Phys. **9**, 693 (2013).
[17] Morimae, T., Phys. Rev. A **89**, 060302 (2014).
[18] Morimae, T. and Fujii, K., Nat. Commun. **3**, 1036 (2012).
[19] Morimae, T. and Fujii, K., Phys. Rev. A **87**, 050301 (2013).
[20] Morimae, T. and Fujii, K., Phys. Rev. Lett. **111**, 020502 (2013).
[21] Ostrovsky, R., Paskin-Cherniavsky, A., and Paskin-Cherniavsky, B., in *Advances in Cryptology CRYPTO 2014*, Lecture Notes in Computer Science, Vol. 8616, edited by J. Garay and R. Gennaro (Springer Berlin Heidelberg, 2014) pp. 536–553.
[22] Raussendorf, R. and Briegel, H. J., Phys. Rev. Lett. **86**, 5188 (2001).
[23] Sander, T., Young, A., and Yung, M., in *Foundations of Computer Science, 1999. 40th Annual Symposium on* (1999) pp. 554–566.
[24] Shor, P. W., SIAM Journal on Computing **26**, 1484 (1997).
[25] Tsiounis, Y. and Yung, M., in *Public Key Cryptography*, Lecture Notes in Computer Science, Vol. 1431, edited by H. Imai and Y. Zheng (Springer Berlin Heidelberg, 1998) pp. 117–134.

ACKNOWLEDGMENTS

We thank Takahiro Kubota, Tomoyuki Morimae, and Joseph Fitzsimons for insightful discussions. This work was supported by JSPS Grant-in-Aid for JSPS Fellows Grant No. 26 · 9148.