Classically Driven Delegated Blind Quantum Computing

Léo COLISSON, supervisé par Elham KASHEFI,
Laboratory for Foundations of Computer Science (LFCS)
School of Informatics, Edinburgh

1er Février - 8 Juillet

Le contexte général

Mon stage s'est déroulé en deux parties. Durant les 2-3 premiers mois, j'ai travaillé sur un moyen d'améliorer l'efficacité des protocoles liés à bitcoin en utilisant des algorithmes quantiques. Cependant, je n'étais pas très inspiré par cette question, et j'ai fini par m'intéresser à une question assez différente qui m'a beaucoup plus motivé sur la deuxième moitié de mon stage, liée à la cryptographie quantique. Dans la suite de ce rapport, je vais donc me consacrer exclusivement à cette deuxième question.

D'ici quelques décennies, il est possible qu'un nouveau type d'ordinateur arrive à être fabriqué : un ordinateur quantique. Ces ordinateurs, qui exploitent des propriétés de la physique quantique pourraient résoudre de manière efficace des problèmes actuellement considérés comme très difficiles, et qu'un ne peut pas résoudre efficacement avec un ordinateur classique (par exemple RSA ou le logarithme discret peuvent être résolus simplement avec un ordinateur quantique). Cependant ces ordinateurs sont extrêmement difficiles et coûteux à produire, il est donc fort probable que pendant une longue période, seul quelques sociétés en possèdent. Il serait alors possible d'exécuter des calculs sur ces ordinateurs quantiques à distance. Ceci est d'ailleurs presque déjà réalité, puisqu'il est déjà possible de lancer des algorithmes quantiques sur un petit ordinateur quantique d'IBM.

Le problème étudié

Cependant, il est possible que l'utilisateur veuille garder ses calculs secrets : que ce soit pour cacher ses données, ses résultats, ou même l'algorithme qu'il utilise. Si on s'autorise à avoir un canal quantique entre le client et le serveur (il faut que le client puisse envoyer des bits quantiques au serveur), on sait qu'il est possible de tout cacher au serveur. Cependant cette condition est très forte : de nos jours on arrive difficilement à envoyer un bit quantique à plus de 100km de distance, on est donc loin d'avoir un réseau Internet entièrement quantique. Je me suis donc posé la question suivante : serait-il possible de faire la même chose, mais en se limitant à des communications purement classiques ? Au moment où j'ai fait mon stage, un seul article essayait de répondre à cette question (écrit en partie par un étudiant Atul Mantri), mais la solution proposée n'était pas constructive, et ne pouvait fonctionner que si on arrivait à montrer l'universalité d'une construction, chose que l'on n'a pas encore réussi à montrer. Cependant, début août (mon stage était déjà terminé, mais notre solution n'avait pas encore été publiée), un autre article à été publié, affirmant résoudre également cette question...

La contribution proposée

Durant toute ma deuxième partie de stage, j'ai travaillé avec Alexandru COJOCARU, étudiant en thèse dans le même laboratoire que moi.

Pour résoudre ce problème, nous avons commencé par travailler avec Atul Mantri, étudiant à Singapour afin d'essayer de rendre sa solution constructive. Je pense avoir un début de preuve pour montrer l'universalité de la construction, mais il reste un problème de taille à résoudre : il n'est pas vraiment possible d'utiliser des propriétés de composition dans cette construction, ce qui rend la traduction d'un algorithme vers cette construction difficile... J'ai donc essayé de chercher d'autres pistes.

La construction d'Atul se basait principalement sur une espèce d'obfuscation : on essayait de noyer le serveur dans un tas d'informations, mais on n'utilisait aucun problème « difficile » pour le serveur. Un autre article nous indiquant que nous avions gagné si nous arrivions à générer côté serveur un qubit aléatoire, et que le client connaissait l'état de ce qubit. J'ai donc cherché un moyen pour générer un qubit aléatoire, mais toutes mes tentatives se vouaient à un échec à cause d'une propriété simple : le serveur pouvait tout le temps relancer les mêmes instructions (quitte à faire quelques corrections) pour produire de nombreux échantillons du même qubit, ce qui lui permet d'obtenir une description de ce qubit, cassant la sécurité du protocole. J'ai donc cherché des problèmes impossibles à reproduire, et je suis tombé sur les problèmes de type PostBQP. En attendant de trouver explicitement la fonction que j'avais besoin, j'ai également postulé l'existence d'une fonction de hash contenant une backdoor un peu spéciale. Enfin, en mélangeant le tout et en utilisant une méthode inventée pour l'occasion, nous avons réussi à générer notre qubit aléatoire. Il nous reste cependant à trouver une implémentation de notre fonction de hash, mais nous avons déjà quelques pistes pour ça.

Les arguments en faveur de sa validité

D'un point de vue intuitif, notre solution semble être la bonne car elle arrive à résoudre les principaux problèmes fondamentaux qui se posent en sécurité quantique : la reproductibilité est résolue par l'utilisation de PostBQP, la sécurité « computationnelle » est obtenue en passant par notre fonction de hash à backdoor. De plus, nous avons une preuve de correction qui montre que l'algorithme fait bien ce qu'on espère qu'il fasse.

Ensuite, d'un point de vue plus pragmatique, la solution publiée à posteriori début août a également besoin d'une fonction de hash très semblable à la notre, les solutions semblent donc converger vers cette fonction...

Au niveau de la stabilité, notre solution est intéressante car elle ouvre de nouvelles portes, et ce indépendamment de la fonction de hash utilisée. Cependant, pour être utilisable en pratique, il reste à expliciter cette fameuse fonction. Nous avons cependant plusieurs pistes pour la trouver. . .

Le bilan et les perspectives

Comme expliqué plus haut, il nous reste à expliciter cette fonction de hash. Certains travaux ont publiés des fonctions qui s'y rapprochent, mais il nous reste à vérifier qu'elles respectent les bonnes propriétés. Cependant, comme notre approche suppose l'existence d'une telle fonction comme hypothèse, notre méthode est assez générale pour fonctionner avec plusieurs fonctions de hash, et ouvre de nombreuses portes pour la suite.

Internship Report: Classically Driven Delegated Blind Quantum Computing

Léo Colisson

Master 1 Computer Science
École Normale Supérieure Paris-Saclay

Supervisor: Elham Kashefi

Laboratory for Foundations of Computer Science
School of Informatics, Edinburgh

Contents

1 Quick overview and related work

At the beginning of my internship, I tried to find a way to improve the bitcoin-like protocols using quantum algorithms. However, I was not really inspired by the question, and after 2-3 months I got interest on a quite different question, related to quantum cryptography. Let us understand the problem in details.

In a few decades, it's possible that a new kind of computers will be created: a quantum computer. Indeed, the quantum laws of physics would allow us to efficiently solve some problems for which we don't know any algorithms running on a Turing machine that can efficiently solve them (a famous example is the Shor algorithm that can break RSA). The rough idea to understand partially this speedup is that it's possible to run in parallel a computation on all the possible inputs. Now, let us suppose that a big company Bob has a quantum computer [1], and that Alice (a classical computer) wants to use this computer. Moreover, Alice wants to hide to Bob what she is computing (she wants to hide the (classical) input, the (classical) output, and the algorithm). We already know[5] that it is possible to have such hiding if there is a quantum channel between Alice and Bob (Alice needs to send to Bob some quantum data). However in practice, it's pretty hard to build a quantum channel on a long distance, and it's definitely not usable on the actually Internet. *So it's pretty natural to wonder if it's possible to keep a classical channel between Alice and Bob.* You can note that a negative result[2] already exists : it very unlikely that such a protocol can exist with unconditional security, so our solution is instead computationally secure (which means that, if Bob has access to unlimited computational power, he can get information on the computation we want to perform).

Before our attempt to solve this open question, one paper[9] already tried to solve the problem, but it came up with a solution which is non constructive, so it's not usable in practice.

During my internship, I worked with Alexandru COJOCARU, a PhD student, and at the end we came up with the first protocol that can perform classically driven blind delegated quantum computation though a classical channel. However, we rely on the existence of a specific cryptographic function that we didn't explicit yet, so we still need to build this function to have a fully usable protocol (we already have some ideas to build this function).

You can note that we didn't publish our work yet, but in the meantime Urmila Mahadev, a PhD student from Berkeley, published a paper[8] supposed to solve our problem.

2 Introduction to Quantum Algorithms

2.1 Dirac Notation

A quantum algorithm is a set of instruction working on a quantum state. A quantum state is a vector with norm 1 belonging to a given Hilbert space $\mathcal{H}$. It's very common to use the Dirac notation for vectors in quantum computing. The idea is that each vector $\overrightarrow{v}$ is represented by a "ket" : $|v\rangle := \overrightarrow{v}$. Then, the normal form of this vector is represented by a "bra" : $\langle v| := v^\dagger = (v^*)^T$. You can note that this notation let us write the scalar product between two vectors $\overrightarrow{u}$ and $\overrightarrow{v}$ like this : $\langle u||v\rangle$, or in a more condensed form $\langle u|v\rangle$, and that the projector on the vector $\overrightarrow{v}$ is written $|v\rangle\langle v|$. Most of the time, we use $|\phi\rangle$ or $|\psi\rangle$ instead of $\overrightarrow{v}$ to represent a quantum state. You can note that, because the quantum states have norm 1, $\langle\phi|\phi\rangle = 1$.

In quantum computer, we often work with a canonical Hilbert space $\mathcal{H}_2$ of dimension 2 described

[1]Actually, IBM already provides on the internet an access to a small quantum computer

by an orthonormal basis called *computational basis* : $\{|0\rangle, |1\rangle\}$. We call it a *qubit.* So basically, all the single qubits can be described as $|\phi\rangle = a|0\rangle + b|1\rangle$, with $a, b \in \mathbb{C}$ and $a^2 + b^2 = 1$. Here are some usual notation used to represent some special qubit states:

$$|0\rangle = |0\rangle \tag{1}$$
$$|1\rangle = |1\rangle \tag{2}$$
$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{3}$$
$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{4}$$
$$|+_\alpha\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}|1\rangle) \tag{5}$$
$$\tag{6}$$

You can note that $\{|+\rangle, |-\rangle\}$ is also an orthonormal basis of $\mathcal{H}_2$.

Then, to create Hilbert states of bigger dimension (with more qubits), we use the tensor product : $\mathcal{H}_2 \otimes \mathcal{H}_2$. Here, the basis of this bigger Hilbert space is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. We can also interpret these sequences of 0 and 1 as numbers written in binary, letting us rewrite the above basis as $\{|0\rangle, |1\rangle, |2\rangle, |3\rangle\}$. Then, for any n qubits, the Hilbert space $\mathcal{H}_2^{\otimes n}$ can be described using the basis $\{|n\rangle \mid n \in [\![0, \dots, 2^n[\![\}$.

NB : for simplicity, the quantum states here does not have always a norm equals to 1, but it's easy to normalize them back.

2.2 Operations on qubits

Given a state $|\phi\rangle$, (usually we start with a given number of n qubits in the state $|0\dots0\rangle$), it is then possible to apply on it several kinds of operations:

- we can perform a unitary U on the state ($U^\dagger U = 1$, where $U^\dagger$ is the conjugate transpose matrix of U), and we get at the end the state $U|\phi\rangle$. When we apply on the i-th qubit of bit quantum state a unitary U defined on one qubit, the unitary which is actually applied is U tensored with the identity on the others qubits:

$$\underbrace{Id \otimes \cdots \otimes Id}_{i-1 \text{ times}} \otimes U \otimes Id \otimes \cdots \otimes Id$$

- we can "measure" the i-th qubit according to an orthonormal basis $\mathcal{B} = \{|\psi_0\rangle, |\psi_1\rangle\}$. To do so, we decompose $|\phi\rangle$ according to this basis:

$$|\phi\rangle = c_0|\phi_{00}\rangle \otimes |\psi_0\rangle \otimes |\phi_{01}\rangle + c_1|\phi_{10}\rangle \otimes |\psi_1\rangle \otimes |\phi_{11}\rangle \text{ with } \phi_{00}, \phi_{10} \in \mathcal{H}_2^{\otimes(j-1)}$$

  Then the measurement will output an integer $b \in \{0, 1\}$, such that the output is j with probability $|c_j|^2$. Then, the state that you will get after the measurement is

$$|\phi_{b0}\rangle \otimes |\psi_b\rangle \otimes |\phi_{b1}\rangle$$

Sometimes when we don't need the i-th qubit anymore, we consider that the measurement destruct the qubit, so the final state is

$$|\phi_{b0}\rangle \otimes |\phi_{b1}\rangle$$

We denote by M_i the measurement on the i-th qubit along the computational basis ($|0\rangle$ and $|1\rangle$), and we say that we perform a measurement using an angle α when the basis used is $\{|+_\alpha\rangle, |-_\alpha\rangle\}$.

Here are some well known unitaries, denoted as "gate" when we use them in a quantum circuit:

- The Hadamard gate:

  $$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

  This gate is useful to go from the computational basis $\{|0\rangle, |1\rangle\}$ to the basis $\{|+\rangle, |-\rangle\}$. It's also very useful to create a superposition of states. For example, if we apply the Hadamard gate on the two qubits in the state $|00\rangle$, you will get the state $|00\rangle+|01\rangle+|10\rangle+|11\rangle = |0\rangle+|1\rangle+|2\rangle+|3\rangle$, which is a superposition of all the possible 2-bits inputs.

- The phase-shift gate:

  $$R(\alpha) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$$

- The X, Y, and Z gates:

  $$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

  These matrix are also known as the Pauli matrices.

- For any one-gate matrix U, we denote by CU or Control-U (replace the gate U by the name of the gate) the controlled version of the gate. If U is defined like this:

  $$U = \begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}$$

  then CU is defined like this:

  $$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix}$$

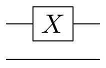
  For example, we will use the Control-Z gate:

  $$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

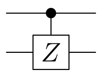
In the following, we will write when needed in subscript the number of the qubit(s) on which the gate is applied. For example H_i is the Hadamard gate applied on the i-th qubit, and $CZ_{i,i+1}$ is a control-Z applied between the qubits i and $i+1$.

2.3 Circuit notation

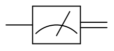
We can also represent a quantum algorithm using a circuit. Each wire represent a qubit, and each box represent a unitary (gate) applied on this qubit. For example, an X gate applied on the first qubit is represented like this:



A control-Z would be denoted like this:

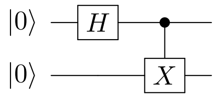


And a measurement on the computational basis is denoted by the following picture (a double wire is a wire containing classical information, here the output of the measurement):



2.4 Example

First, let us define a very famous example to illustrate what is the entanglement, and to let us be more familiar with quantum gates. We will consider this circuit:



We can write it using the usual notations like this: $CX_{1,2}H_1|00\rangle$. Now let's expand this equation to get the final state.

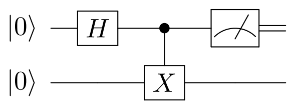
$$CX_{1,2}H_1|00\rangle = \frac{1}{\sqrt{2}}CX_{1,2}(|00\rangle + |10\rangle) \tag{7}$$

$$= \frac{1}{2}(|00\rangle + |11\rangle) \tag{8}$$

And because we don't mind normalisation:

$$= |00\rangle + |11\rangle \tag{9}$$

This state is called a Bell pair (or EPR pair). It is very interested because this state cannot be written as a tensor product of two separated qubits $|\phi_1\rangle \otimes |\phi_2\rangle$: we say that the two states are entangled. Now, let us imagine that we add at the end a measurement on the first qubit:



Then we have two options: if the output of the measurement is 0, then the state becomes:

$$|0\rangle\langle 0|(|00\rangle + |11\rangle) = |0\rangle\langle 0|(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle) \tag{10}$$

$$= |0\rangle \underbrace{\langle 0|0\rangle}_{=1} \otimes |0\rangle + |0\rangle \underbrace{\langle 0|1\rangle}_{=0} \otimes |1\rangle) \tag{11}$$
$$= |0\rangle \otimes |0\rangle \tag{12}$$
$$= |00\rangle \tag{13}$$

and if the output of the measurement is 1, then the state becomes:

$$|1\rangle\langle 1|(|00\rangle + |11\rangle) = |1\rangle\langle 1|(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle) \tag{14}$$
$$= |1\rangle \underbrace{\langle 1|0\rangle}_{=0} \otimes |0\rangle + |1\rangle \underbrace{\langle 1|1\rangle}_{=1} \otimes |1\rangle) \tag{15}$$
$$= |1\rangle \otimes |1\rangle \tag{16}$$
$$= |11\rangle \tag{17}$$

So, just by doing a measurement on the first qubit, we changed the value of the second qubit : the first qubit and the second qubit now have the same value.

2.5 Universality

Like in the classical boolean circuits, we can define a notion of universality. We say that a set of gate is universal if any unitary can be approximate up to an arbitrary small ε using a circuit made of gate took in this set of gate, and measurements in the computational basis. The set of gate $\{CX, H, R(\frac{\pi}{4})\}$ is known to be universal [3].

2.6 The no-cloning theorem

In quantum physics, the no-cloning theorem is very important. This theorem states that it is impossible to create an identical copy of an arbitrary unknown quantum state. This theorem is very simple, but it has lot's of implication in quantum computing. For example, if Alice secretly choose a $\alpha \in \{0, \frac{1}{4}\pi, \ldots, \frac{7}{4}\pi\}$, prepare a state $|\phi\rangle = |+_\alpha\rangle$, and send this state to Bob, there is no way for Bob to know for sure which angle was chosen by Alice.

However, if the server knows that the state he received belongs to a set of orthonormal states, then he can determine which state it is. For example, if Alice choose randomly one state $|\phi\rangle$ in the set $S = \{|0\rangle, |1\rangle\}$, sends the state to Bob, and says to Bob "my state is a state belonging to the set S", then Bob can check whether $\phi = |0\rangle$ or $\phi = |1\rangle$. To do so, Bob needs to perform a measurement on the qubit : $M_1|\phi\rangle$. If the output is 0, then Bob knows that the state was $\phi = |0\rangle$, and if the output is 1, then Bob knows that the state was $\phi = |1\rangle$.

The idea is the same for any orthonormal vector, the idea is to choose the good basis to use to perform the measurement, or to apply some operators on the state before measuring in the computational basis.

2.7 PostBQP

As defined in [1], PostBQP is a complexity class which consists of all problems solvable by a quantum computer in polynomial time, given the ability to postselect on a measurement yielding a specific outcome. This class, equivalent to PP, is believed to be far bigger than BQP, the set of problems solvable by a quantum computer in polynomial time.

Intuitively, this means that it is impossible for a quantum computer to force the outcome of a measurement. For example, let us imagine that we have a state

$$|\phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |1\psi\rangle)$$

Then, if we measure the first qubit in the computational basis, we get $|0\rangle$ with probability $\frac{1}{2}$, and we get $|\psi\rangle$ with probability $\frac{1}{2}$. And because BQP is believed to be different from PostBQP, it is impossible to force the outcome of the measurement to be 1. So it means that with probability $\frac{1}{2}$, we lost the state ψ, and that we cannot do anything to increase this probability. Using the same idea, if we get a bigger state

$$|\phi'\rangle = \frac{1}{\sqrt{2^n}}(|(2^n-1)\psi\rangle + \sum_{i=0}^{2^n-2} |i0\rangle)$$

the best thing we can do to get get the state ψ is to measure the n first qubits, and hope that all of them will output 1... Which will occur with probability $\frac{1}{2^n}$! This will be very useful later to avoid that the server reproduce the computation to get several copies of the same qubit.

3 Universal Blind Quantum Computing and Q Factory

The idea of this section is *not* that you fully understand how MBQC and UBQC work, it would take too much time. This section is here just to give an intuition why giving to Alice the opportunity to send any random qubit to Bob (through what I called *Q Factory*) is enough to provide Blind Quantum Computing.

3.1 Q Factory

We want to define an interactive protocol between a classical client and a quantum computer to perform Blind Quantum Computation. To do that, we will define a protocol that provides a "(classically driven) Secret Q[2] Factory", a box that can produce on the server side a qubit that the client "knows", but such that the server "cannot get information about it". Here is a more formal definition:

Definition 1 (Q Factory). *A* Q Factory *is an ideal functionality, or any protocol indiguishable from this idea functionality, that provides the following thing:*

- *Secretly choose an angle* $\alpha \in \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$
- *Send* $|+_\alpha\rangle$ *to the server Bob*
- *Send* α *to the client Bob*

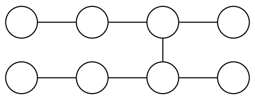
[2]Despite what someone might think, Q does not stand for qubit, but for the character in the James Bond series (https://en.wikipedia.org/wiki/Q_%28James_Bond%29)

Before defining a protocol that provides a Q Factory, we will understand how having a Q Factory as a blackbox is enough to get Blind Quantum Computation. And to understand that, let's see how UBQC has been defined, when we have a quantum channel between Alice and Bob.

3.2 Measurement Based Quantum Computing (MBQC)

As explained in [6], it is possible to convert any quantum circuit into another kind of circuit. To do this conversion we need to follow the following steps:

- Convert the quantum circuit in another quantum circuit using only gates in the set $\{CX, H, R(\frac{\pi}{4})\}$ (it's possible because this set is known to be universal)
- Then, replace each gate by 8 qubits in the following shape (each line represent a CZ applied to the two qubits):

  and associate to each of these qubit a given angle of measurement, depending on the gate implemented (see [6, p. 27] for more details). When no gate are present, just put 0. You will obtain at the end what we call a "brickwork" state.

To run the computation, the idea is to prepare the big brickwork state using $|+\rangle$ as initial value, take each qubit one by one, from left to right, and measure it using the specified angle. When you get a non zero output as the output of the measurement, you need to correct the angle on the next qubits to keep a good outcome (more precisely, you multiply by -1 the angle used on one of the qubits which are linked to the qubit you just measured, and you need to add π to the angles of the qubits which are still after this qubit, but it's not really relevant for the next part).

3.3 UBQC

As stated before, the paper written by Elham Kashefi, Joseph Fitzsimons and Anne Broadbent [5] explains how it's possible for a client Alice to perform Universal Blind Quantum Computing (UBQC) on a remote server when there is a quantum channel between Alice and Bob. To do so, we proceed in several steps:

1. First, the client Alice converts the quantum circuit into the Measurement Based Quantum Computing (see above) on her side.
2. For each qubit i, she choose a random "one time pad" angle θ_i in the set $\{0, \frac{1}{4}\pi, \ldots, \frac{7}{4}\pi\}$, and sends to the server the state $|+_{\theta_i}\rangle$
3. Bob create on his side the brickwork state, using the qubits sent by Alice
4. For each qubit i:
   - Alice chooses a random $r_i \in \{0, 1\}$ (it will be used to re-interpret 0 as 1 and 1 as 0 if $r_i = 1$)

- She computes the angle depending on the qubit already measured, she add to this angle the random one time pad angle θ_i (it will cancel the angle of the qubit that she send before to the server) as well as $r_i\pi$, and she sends this angle δ_i to Bob.
- Bob measures the qubit i in the basis $\{|+_{\delta_i}\rangle, |-_{\delta_i}\rangle\}$, and send the outcome s_i of the measurement to Alice
- If $r_i = 1$, Alice flips s_i.

3.4 Q Factory and Blind Quantum Computing

This scheme can look pretty complicated (if you want more details, you can read [5]), and the goal of this document is not to explain it. I put it here just to give a sketch of proof for the following lemma, that will be useful later:

Lemma 1 (Q Factory is enough for Blind Quantum Computation)**.** *If we have an access to a Q Factory, then it is possible to perform Classically Driven Delegated Blind Quantum Computing.*

Proof. The proof is pretty easy if we use the work of [5]. The only part where we need quantum communication is during the second step. And if we replace the quantum communication in step 2 by the functionality provided by the Q Factory, it's strictly equivalent on server side (it doesn't matter if the qubit comes from Alice, or from an ideal functionality), and also strictly equivalent on client side (it doesn't matter if the client randomly choose θ, or if an idea functionality randomly choose θ). The remaining part of the proof is exactly the same as the one in [5]. □

4 My contribution

Now that we understood why having a classically driven Q Factory is useful, let us see the protocol I created to provide a Q Factory. But first, let us study our main assumption: the existence of a special cryptographic function, a kind of 2-regular backdoor hash function:

4.1 Assumptions

Here, we suppose that we know a set of classical functions $f_s : \mathcal{X} \to \mathcal{Y}$ such that

1. $\mathcal{X}$ is the set of all binary strings of size n
2. $\mathcal{Y}$ is the set of all binary strings of size m
3. for all s, f_s can be described using a polynomial-sized circuit $\mathcal{C}$
4. for all s, f_s is 2-regular (or at least with high probability), ie, for all $y \in \mathcal{Y}$, $|f^{-1}(y)| = 2$. (A generalization of this assumption would be that f is not injective and has a polynomial number of pre-image, but this would also require a generalized version of the client mentalism part.)
5. Knowing the secret parameter s, a classical client can efficiently compute for all $y \in \mathcal{Y}$ the set $f^{-1}(y)$.
6. For this last assumption, we can define two versions (the second one is more general, we just need to use H equals to identity to get the simple version):

(a) Simple version : For all $y \in \mathcal{Y}$, a quantum server that knows the circuit $\mathcal{C}$, and has a single copy of the state $|\phi\rangle = \sum_{x \in f^{-1}(y)} |x\rangle$ cannot find “enough” information about x_1 and x_2. More formally, we would like that having $\mathcal{C}$, $|\phi\rangle$, a list of $\alpha_1 \dots \alpha_{n-1}$ randomly chosen in $\{\frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$ and a list of measurement results $b_1 \dots b_{n-1}$, we have no information on the resulted angle θ defined in subsection 5.4, except that this angle belongs to $\theta \in \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$.

(b) Generalized version : For all f_s, there exists a classical hash function H with a poly-sized circuit $\mathcal{H}$ such that, for all $y \in \mathcal{Y}$, a quantum server that knows the circuit $\mathcal{C}$ and $\mathcal{H}$, has a single copy of the state $|\phi\rangle = \sum_{x \in f^{-1}(y)} |x\rangle$ and has a list of $\alpha_1 \dots \alpha_{n-1}$ randomly chosen in $\{\frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$ and a list of measurement results $b_1 \dots b_{n-1}$, cannot get any information on the angle θ defined in subsection 5.4, except that this angle belongs to $\theta \in \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$.

For simplicity and as a first step, we consider that the server is honest but curious, which means that the server follows the protocol, but tries to get as much information as possible.

5 Protocol for the simple version

5.1 Unitary

First Alice sends to Bob a classical circuit of f_s. Bob uses it to implement the unitary U_{f_s} that maps $|x\rangle|0\rangle \dots |0\rangle$ into two registers (two sets of qubits) $|x\rangle|f_s(x)\rangle$. Bob then run this unitary on a superposition of all possible x using a Hadamard gate in front of all single $|0\rangle$ input (see the comment in the definition of the Hadamard function in the subsection 2.2).

After this step, the outputs qubits will contains $\sum_{x \in [\![0, 2^n-1]\!]} |x\rangle|f(x)\rangle$. And because the function is supposed to be 2-regular, we can rewrite it as $\sum_{y \in f([\![0,2^n-1]\!])} (|f_1^{-1}(y)\rangle + |f_2^{-1}(y)\rangle) \otimes |y\rangle$, where f_1^{-1} and f_2^{-1} are the two pre-images of f.

5.2 Measurement

Then, we measure the second register (it will give us a value y_0), the first register will then be $|\phi\rangle = \sum_{x \in f^{-1}(y_0)} |x\rangle = |x_1\rangle + |x_2\rangle$. We denote by $x_1 = c_1 \dots c_n$ and $x_2 = d_1 \dots d_n$ their binary representation.

Intuitively, here the huge superposition state collapses: only one output value amount the exponential number of output value of the function is selected. At the same time, we get the two preimages of this function. You may think that it’s strange, because the function is supposed to be collision-resistant. However, it’s not a problem, because in the definition of collision-resistant, we first fix a random output y, and we ask “can you find the pre-images for this precise y ?”. And because PostBQP is bigger than BQP, we have no way to force the output to be y when we perform the measurement, so we have no way to recover the preimages of y. That’s why the server cannot run the same algorithm again and again to get several copies of the qubit produced by the Q Factory: every time he will run the algorithm, he will get a different y_0.

5.3 Squeezing

Intuitively, the idea of this step is to compress the n qubits into only one qubit. Indeed, before this state, we have n qubits in an unknown state, but in the UBQC protocol (and in the definition of Q Factory), we want to provide only one qubit, that's why this state is useful.

Then the client chooses randomly $n-1$ random angles $\alpha_1, \dots, \alpha_{n-1} \in \{\frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$ (we don't take 0 because, as we will see, we "lose" one bit of security for each 0 angle), each α_i corresponding to the i-th qubit. The server on his side applies Ctrl-Z on all the remaining consecutive qubits of $|\phi\rangle$, and finish by measuring the $n-1$ first qubits using their corresponding angle $\alpha'_i = (-1)^{b_{i-1}}\alpha_i + (-1)^{b_{i-2}}\pi$, where b_i are the outcome of the measurement of qubit i using the convention that if $i < 1$, $b_i = 0$ (actually, it's also possible to use directly α_i instead of α'_i, but the computation are a little bit easier to perform if we do this correction). That way, the server has at the end a single qubit in a pure state $|\phi'\rangle$.

5.4 Client mentalism and UBQC plug-in

The idea is that the client can get a full (classical) description of the state $|\phi'\rangle$ after the squeezing "efficiently":

$$|\phi'\rangle = X^{b_{n-1}} Z^{b_{n-2}} \left(\exp\left(-i \left(\sum_{i=1}^{n-1} c_i \alpha_i \right) \right) (-1)^{\sum_{i=1}^{n-1} c_i c_{i+1}} |c_n\rangle + \exp\left(-i \left(\sum_{i=1}^{n-1} d_i \alpha_i \right) \right) (-1)^{\sum_{i=1}^{n-1} d_i d_{i+1}} |d_n\rangle \right) \tag{18}$$

Intuitively, the X and Z correction comes from the fact that on the last qubit we cannot correct a "bad" (i.e. non zero) previous outcome by modifying the measurement angle (because there is no measurement angle...), and the right part of the expression comes from a direct induction proof (see proof in the correctness section).
Because the client knows everything (the b_i, the c_i, and the d_i), he can compute this state very easily. However, the server does not know c_i and d_i "so" intuitively, he cannot compute $|\phi\rangle$.

We notice that this state is pure, and that we can write it as, up to a global phase : $|0\rangle$, $|1\rangle$, or $|0\rangle + e^{i\theta}|1\rangle = |+_\theta\rangle$. Then, because UBQC relies only on preparing states with the form $|+_\theta\rangle$, we can do one of the following things:

- reject the computation when the state does not provides a $|+_\theta\rangle$, repeat from scratch until you have a $|+_\theta\rangle$ (really inefficient)
- just before the squeezing, reorder the qubits so that the last measured qubit corresponds to different bit values on x_1 and x_2

In the following, we choose the second option : we reorder the qubits using a permutation σ. To simplify the notation we define for all i, $c'_i = c_{\sigma^{-1}(i)}$ and $d'_i = d_{\sigma^{-1}(i)}$. Without loss of generality (it's just exchanging x_1 and x_2), we can consider that the last re-ordered bit of x_1 is 0, and that the last re-ordered bit of x_2 is 1, i.e.:

$$c'_n = 0 \tag{19}$$

$$d'_n = 1 \tag{20}$$

We can show (see subsubsection 6.2.3) that, in this setting, if we pose

$$\boxed{\theta = (-1)^{b_{n-1}} \left(\sum_{i=1}^{n-1} (c'_i - d'_i)\alpha_i \right) + \pi \left(b_{n-2} + \sum_{i=1}^{n-1} (c'_i c'_{i+1} - d'_i d'_{i+1}) \right)} \tag{21}$$

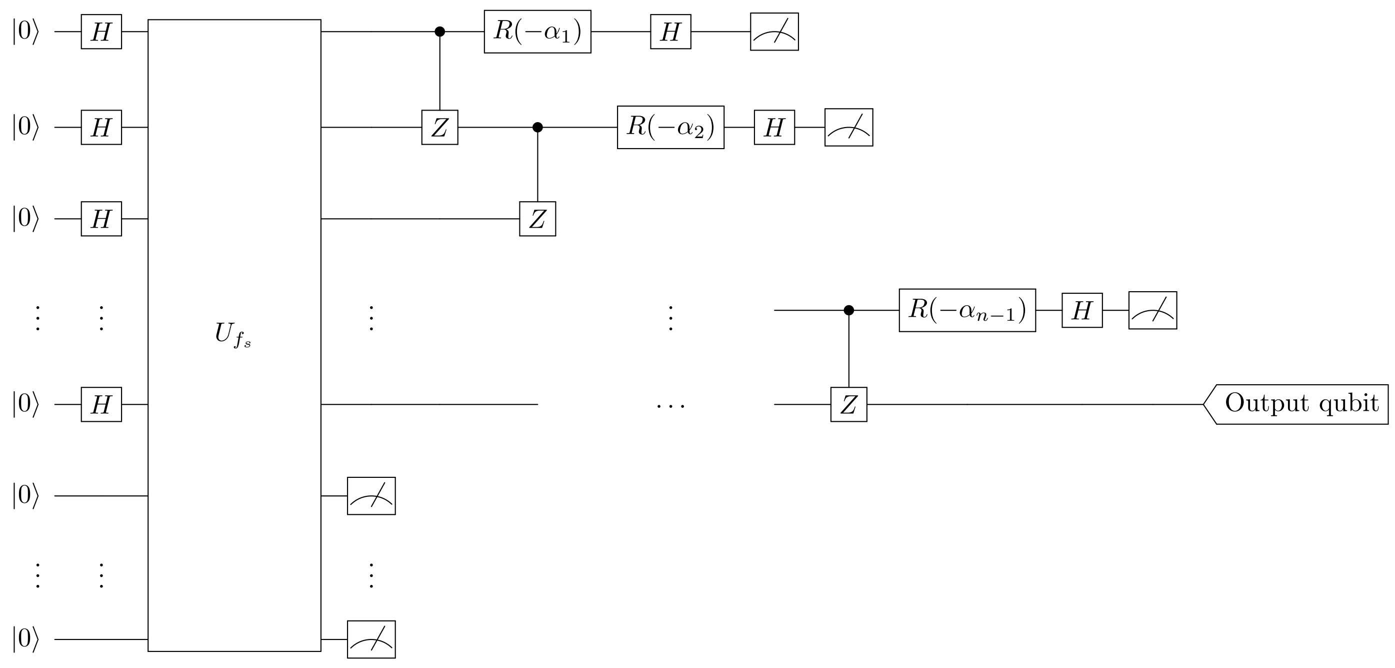
then

$$\boxed{|\phi'\rangle = |0\rangle + e^{i\theta}|1\rangle = |+_\theta\rangle} \tag{22}$$

You can notice that here the user can easily choose the angle θ (up to a global X/Z correction, that he knows). And, intuitively, because the server does not have information about x_1 and x_2, he cannot get θ... Which is more or less the requirement to have a Q Factory ! Let us now prove the correctness of this protocol (i.e. that the protocol do what it is supposed to do).

5.5 Sum-up

Here is the whole circuit used to produce the Q Factory:



6 Proof of correctness

6.1 Correctness until Squeezing

Basically in this part the only part to check is that the unitary presented exists (not a problem), and that the state at the end is indeed $|c_1 \dots c_n\rangle + |d_1 \dots d_n\rangle$. It's trivial, and the protocol already explains pretty clearly why.

6.2 Correctness of the client mentalism

6.2.1 Case 1: All measurement outcomes are null

Let us first assume that all the measurement outcomes are zero. Before the squeezing, the quantum state is $|c_1 \dots c_n\rangle + |d_1 \dots d_n\rangle$. After the squeezing, the state becomes, by definition :

$$|\phi'\rangle = \left(\prod_{i=1}^{n-1} M_i H_i R_i(-\alpha_i) CZ_{i,i+1}\right)(|c_1 \dots c_n\rangle + |d_1 \dots d_n\rangle) \tag{23}$$

with all measurement outcomes equal to zero.
Now, let us prove the following more general lemma (which let us conclude without any difficulty) by induction on n (using the convention that for all $i < 1$ or $i > n$, $\alpha_i = b_i = 0$) :

Lemma 2. *For all $n, m \in \mathbb{N}$ such that $m > n$,*

$$\left(\prod_{i=1}^{n} M_i H_i R_i(-\alpha_i) CZ_{i,i+1}\right)(|c_1 \dots c_m\rangle + |d_1 \dots d_m\rangle) \tag{24}$$

$$= \exp\left(-i\left(\sum_{i=1}^{n} c_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} c_ic_{i+1}}|c_{n+1} \dots c_m\rangle + \exp\left(-i\left(\sum_{i=1}^{n} d_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} d_id_{i+1}}|d_{n+1} \dots d_m\rangle \tag{25}$$

Proof. Proof by induction on n:
Base case $n = 0$ is trivial.

Induction step: we assume the formula is true for $n - 1$ and we prove it holds for n. Using the induction hypothesis we have:

$$\left(\prod_{i=1}^{n} M_i H_i R_i(-\alpha_i) CZ_{i,i+1}\right)(|c_1 \dots c_m\rangle + |d_1 \dots d_m\rangle) \tag{26}$$

$$= (M_n H_n R_n(-\alpha_n) CZ_{n,n+1})\left(\prod_{i=1}^{n-1} M_i H_i R_i(-\alpha_i) CZ_{i,i+1}\right)(|c_1 \dots c_m\rangle + |d_1 \dots d_m\rangle) \tag{27}$$

$$= M_n H_n R_n(-\alpha_n) CZ_{n,n+1}\left(\exp\left(-i\left(\sum_{i=1}^{n-1} c_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n-1} c_ic_{i+1}}|c_n \dots c_m\rangle\right. \tag{28}$$

$$\left. + \exp\left(-i\left(\sum_{i=1}^{n-1} d_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n-1} d_id_{i+1}}|d_n \dots d_m\rangle\right) \tag{29}$$

$$= M_n H_n R_n(-\alpha_n)\left(\exp\left(-i\left(\sum_{i=1}^{n-1} c_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} c_i c_{i+1}}|c_n \dots c_m\rangle \right. \tag{30}$$

$$\left. + \exp\left(-i\left(\sum_{i=1}^{n-1} d_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} d_i d_{i+1}}|d_n \dots d_m\rangle\right) \tag{31}$$

$$= M_n H_n \left(\exp\left(-i\left(\sum_{i=1}^{n} c_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} c_i c_{i+1}}|c_n \dots c_m\rangle \right. \tag{32}$$

$$\left. + \exp\left(-i\left(\sum_{i=1}^{n} d_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} d_i d_{i+1}}|d_n \dots d_m\rangle\right) \tag{33}$$

$$= M_n \left(\exp\left(-i\left(\sum_{i=1}^{n} c_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} c_i c_{i+1}}\left(|0c_{n+1} \dots c_m\rangle + (-1)^{c_n}|1c_{n+1}\dots c_m\rangle\right) \right. \tag{34}$$

$$\left. + \exp\left(-i\left(\sum_{i=1}^{n} d_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} d_i d_{i+1}}\left(|0d_{n+1} \dots d_m\rangle + (-1)^{d_n}|1d_{n+1}\dots d_m\rangle\right)\right) \tag{35}$$

$$= \exp\left(-i\left(\sum_{i=1}^{n} c_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} c_i c_{i+1}}|c_{n+1} \dots c_m\rangle \tag{36}$$

$$+ \exp\left(-i\left(\sum_{i=1}^{n} d_i\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n} d_i d_{i+1}}|d_{n+1} \dots d_m\rangle \tag{37}$$

$$\tag{38}$$

Which ends the proof.

□

6.2.2 Case 2: Non-null measurement outcomes: α_i', X and Z correction

If we write the squeezing circuit, use the non-physical anachronic correction Z when the measurement outcome is not 0, and then apply some basics rules using how Z and X commutes with CZ, and the fact that applying an X and measuring using an angle α is equivalent to measure using the angle $-\alpha$, while applying a Z and then measure using an angle α is equivalent to measure using an angle $\alpha+\pi$ (it's basically what MBQC does), you will see that the scheme where you do these angle and X/Z correction is equivalent to the scheme where you don't do any angle or X/Z correction but all your measurement outcomes are equal to 0. So now, it's easy to conclude using the result in the case of zero measurement outcomes and the equivalence between zero measurement outcomes and $\alpha/X/Z$ corrections schemes.

6.2.3 Proof of the nice value of θ

Using the notation define in subsection 5.4, we have

$$|\phi'\rangle = X^{b_{n-1}}Z^{b_{n-2}}\left(\exp\left(-i\left(\sum_{i=1}^{n-1} c_i'\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n-1} c_i'c_{i+1}'}|0\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} d_i'\alpha_i\right)\right)(-1)^{\sum_{i=1}^{n-1} d_i'd_{i+1}'}|1\rangle\right) \tag{39}$$

$$= X^{b_{n-1}} \left(\exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}} |0\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1} d'_i d'_{i+1}} |1\rangle \right) \tag{40}$$

$$= \exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}} |b_{n-1}\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1} d'_i d'_{i+1}} |1-b_{n-1}\rangle \tag{41}$$

Now, we have two cases: First case: $b_{n-1} = 0$. Then

$$|\phi'\rangle = \exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}} |0\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1} d'_i d'_{i+1}} |1\rangle \tag{42}$$

$$= \exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}} \left(|0\rangle + \frac{\exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1} d'_i d'_{i+1}}}{\exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}}} |1\rangle \right) \tag{43}$$

so, up to a global phase,

$$|\phi'\rangle = |0\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} (d'_i - c'_i)\alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1}(d'_i d'_{i+1} - c'_i c'_{i+1})} |1\rangle \tag{44}$$

Second case : $b_{n-1} = 0$

$$|\phi'\rangle = \exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}} |1\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}\sum_{i=1}^{n-1} d'_i d'_{i+1}} |0\rangle \tag{45}$$

$$= \exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}\sum_{i=1}^{n-1} d'_i d'_{i+1}} \left(|0\rangle + \frac{\exp\left(-i\left(\sum_{i=1}^{n-1} c'_i \alpha_i\right)\right) (-1)^{\sum_{i=1}^{n-1} c'_i c'_{i+1}}}{\exp\left(-i\left(\sum_{i=1}^{n-1} d'_i \alpha_i\right)\right) (-1)^{b_{n-2}\sum_{i=1}^{n-1} d'_i d'_{i+1}}} |1\rangle \right) \tag{46}$$

so up to a global phase

$$|\phi'\rangle = |0\rangle + \exp\left(-i\left(\sum_{i=1}^{n-1} (c'_i - d'_i)\alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1}(c'_i c'_{i+1} - d'_i d'_{i+1})} |1\rangle \tag{47}$$

so, if we put together the two cases, and using the fact that $(-1)^n = (-1)^{-n}$, we obtain

$$|\phi'\rangle = |0\rangle + \exp\left((-1)^{b_{n-1}} i \left(\sum_{i=1}^{n-1} (c'_i - d'_i)\alpha_i\right)\right) (-1)^{b_{n-2}+\sum_{i=1}^{n-1}(c'_i c'_{i+1} - d'_i d'_{i+1})} |1\rangle \tag{48}$$

$$= |0\rangle + \exp\left(i\left((-1)^{b_{n-1}}\left(\sum_{i=1}^{n-1} (c'_i - d'_i)\alpha_i\right) + \pi\left(b_{n-2} + \sum_{i=1}^{n-1}(c'_i c'_{i+1} - d'_i d'_{i+1})\right)\right)\right) |1\rangle \tag{49}$$

so if we pose

$$\boxed{\theta = (-1)^{b_{n-1}} \left(\sum_{i=1}^{n-1} (c'_i - d'_i)\alpha_i \right) + \pi \left(b_{n-2} + \sum_{i=1}^{n-1} (c'_i c'_{i+1} - d'_i d'_{i+1}) \right)} \tag{50}$$

then

$$\boxed{|\phi'\rangle = |0\rangle + e^{i\theta}|1\rangle = |+_\theta\rangle} \tag{51}$$

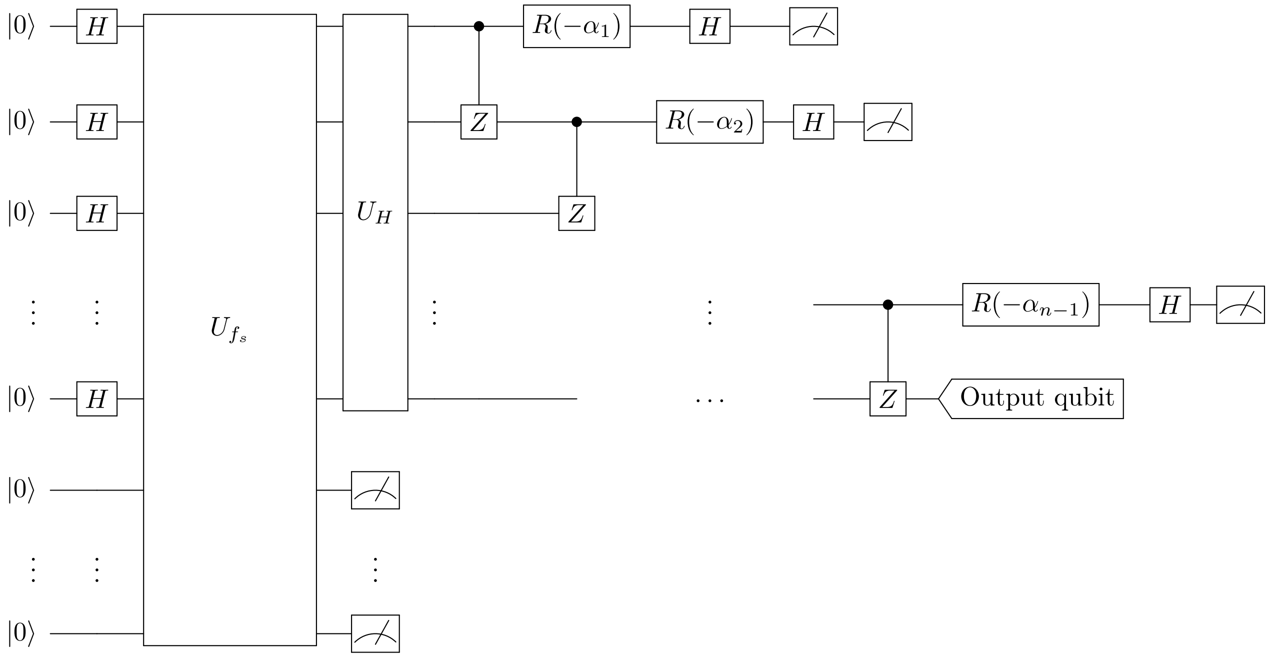
7 Proof of blindness

We didn't wrote yet a very formal proof for the blindness part (i.e. to proof that the server cannot get any information on the state), but the intuition already give a good idea, and most of the "difficulties" are in the assumption of the hash function. The first thing to understand why the server cannot get more information on θ than in the Q Factory is that he cannot reproduce the algorithm several times, because to do so he needs to postselect on the outcome of the first measurement... Which is impossible according to PostBQP (see subsection 2.7). And moreover, the assumption 6.a on the hash function with backdoor make sure that we cannot recover any information once we did the postselection.

8 Protocol for the generalized version

If we want to use the generalized assumption on f, we need to apply one more step between the step "Measurement" and the step "Squeezing" : a hashing step. To do so, we apply the unitary representing the $\mathcal{H}$ function on the qubits $|\phi\rangle$ and some ancilla qubits. Then, nothing changes, we apply the squeezing on this new state, except that during the client mentalism, the values of c_i and d_i are the bits of respectively $H(x_1)$ and $H(x_2)$ instead of x_1 and x_2. The interest of this function is that it can create more bits where $c_i \neq d_i$ (if you look at the value of θ, we want to have basically a probability of around $\frac{1}{2}$ to have $c_i \neq d_i$ so that we never know if we actually took the α_i or not).

The correctness and blindness proof should be pretty identical. Here is the quantum circuit:



9 Conclusion

In conclusion, we can say that this work is one of the first that let us think that it is possible to perform Classically Driven Delegated Blind Quantum Computing. The security comes from two different points:

- the impossibility to postselect on the outcome of a measurement is useful to avoid that the server run the same algorithm again and again
- the use of a hash function with a backdoor that the server cannot invert is useful to hide the value of the qubit to the server

However, to make this protocol fully usable in practice, we still need to explicit the hash function with backdoor. We already have a few idea to do that, and one of them is to use the construction defined in [7], or to use a modified version of Fully Homomorphic Encryption (FHE) with Learning With Error (LWE) [4].

References

[1] S. Aaronson. Quantum Computing, Postselection, and Probabilistic Polynomial-Time. *eprint arXiv:quant-ph/0412187*, December 2004.

[2] S. Aaronson, A. Cojocaru, A. Gheorghiu, and E. Kashefi. On the implausibility of classical client blind quantum computing. *ArXiv e-prints*, April 2017.

[3] A. Barenco, C. H. Bennett, R. Cleve, D. P. Divincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. 52:3457–3467, November 1995.

[4] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. In *Proceedings of the 2011 IEEE 52Nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 97–106, Washington, DC, USA, 2011. IEEE Computer Society.

[5] A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computation. *ArXiv e-prints*, July 2008.

[6] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. *Measurement-Based and Universal Blind Quantum Computation*, pages 43–86. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[7] Craig Gentry and Chris Peikert. How to Use a Short Basis: Trapdoors for Hard Lattices and New Cryptographic Construction. August 2008.

[8] U. Mahadev. Classical Homomorphic Encryption for Quantum Circuits. *ArXiv e-prints*, August 2017.

[9] A. Mantri, T. F. Demarie, N. C. Menicucci, and J. F. Fitzsimons. Flow Ambiguity: A Path Towards Classically Driven Blind Quantum Computation. *Physical Review X*, 7(3):031004, July 2017.

[10] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. http://eprint.iacr.org/2015/939.

[11] P. S. Turner and D. Markham. Derandomizing Quantum Circuits with Measurement-Based Unitary Designs. *Physical Review Letters*, 116(20):200501, May 2016.