New constructions for Quantum Money

Marios Georgiou¹ and Iordanis Kerenidis²

1 CAISS-CCNY-The City University of New York
Shepard Hall, 160 Convent Avenue, New York, NY 10031, USA
mgeorgiou@gc.cuny.edu
2 CNRS-LIAFA-University Paris 7 Diderot
Batiment Sophie Germain, 8 Place FM/13, Paris 75013, France
jkeren@liafa.univ-paris-diderot.fr

Abstract

We create an information theoretically secure secret-key quantum money scheme in which the verification of a coin is classical and consists of only one round; namely, a query from the user to the bank and an accept/reject answer from the bank to the user. A coin of \( n \) qubits can be verified \( \text{poly}(n) \) number of times before it expires. Our scheme is an improvement on Gavinsky’s scheme [6], where multiple rounds of interaction are needed. The construction is based on the notion of quantum retrieval games.

Moreover, we propose a public-key quantum money scheme which is computationally secure in the one-time memory–random oracle model. This construction is derived naturally from our secret-key scheme using the fact that one-time memories are a special case of quantum retrieval games.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases Quantum Money, Quantum Cryptography, Quantum Retrieval Games

1 Introduction

Wiesner [18] in the early ’80s proposed the idea of creating money whose unforgeability is guaranteed by the laws of quantum mechanics. Quantum states seemed an ideal way to encode money, since the no-cloning theorem of quantum states could possibly lead to a no-cloning theorem of money. Informally, a quantum money scheme consists of two main processes; a process Bank that creates valid coins and a process Ver that verifies whether a coin is valid. The use of such a scheme is straightforward; the authorized bank will produce valid money by running the process Bank and the users will be able to pay each other and verify that a coin $ is valid by running the process Ver($).

In Wiesner’s construction a coin consists of several BB84 states (that form a big state $\rho$) together with a classical identification string $s$. The verification of the coin $(\rho, s)$ is a simple one round protocol in which the user of a coin sends the full coin to the bank and the bank replies with a yes/no answer. The answer of the bank depends on its secret key (which corresponds to $s$) as well as the outcomes it gets by applying a measurement on the computational or Hadamard basis to $\rho$. These kinds of schemes are known as (secret-key) quantum money with quantum verification since the user has to communicate quantumly with the bank.

Until recently, the question of whether there exist quantum money schemes where the verification protocol consists only of classical communication was open. Gavinsky [6] in 2012 answered this question in the affirmative by creating the first secret-key quantum money.
scheme with classical verification. His scheme makes use of a new quantum cryptographic idea, that of quantum retrieval games (QRGs) and its security is information theoretic. However, a clear drawback in contrast to Wiesner’s scheme was that the verification of a coin consists of multiple rounds between the user and the bank, thus forcing the bank to maintain a temporary memory for each verification session.

In 2012, Aaronson [1] proposed the idea of public-key quantum money where no communication with the bank is needed in order to verify the coin. In such a scheme, although information theoretic security is impossible, computationally secure schemes may still exist. Classically, it is impossible to create public key money schemes since, in that case, a coin would consist only of a bitstring and, therefore, the copy of a coin would be trivial. Public-key quantum money are essentially the optimal kind of money we could hope for since they can be used as ordinary cash. Although several schemes have been proposed as candidates for public-key quantum money [1, 5], all of them are based on non-standard computational hardness conjectures.

Our contributions are twofold. First, we give the first information theoretically secure quantum money scheme that requires only classical communication with the bank, tolerates errors and the verification consists of a single round, a query to the bank and an answer. The important contribution of this scheme compared to that of Gavinsky [6] is that in the latter, the verification requires a six-message interaction with the bank and, therefore, the bank has to maintain a temporary session memory. Moreover, we have made the proof more modular and conceptually simpler by introducing a new cryptographic primitive as tool for the security analysis.

Second, we create a public-key quantum money scheme from one-time memories in the random oracle model. Considering hash functions as random oracles is a common tool for the security proofs of cryptographic schemes which is invoked when standard properties of hash functions (such as collision resistance) are not enough. Briefly, a hash function behaves as a random oracle if on each query it returns a uniformly random element in its range, being in the same time consistent with the previous queries; e.i. on the same query it returns the same answer. One-time memories are a very natural special case of quantum retrieval games and, thus, our construction is a simple modification of our secret-key scheme. In our construction we also make use of the notion of a quantum money mini-scheme proposed by Aaronson and Cristiano [1] (see subsection 2.2). A clear advantage of this scheme compared to other works in the literature [2, 8, 16] is the direct application of one-time memories to quantum money without going through one-time programs which makes our scheme more efficient. In both our schemes the number of allowed verifications is polynomial on the size of the coins. Our contributions, compared to previous work are summarized in Table 1.

The paper is structured as follows; in section 2 we give the definitions of secret-key and public-key quantum money as well as the corresponding secret-key and public-key mini-schemes. In section 3 we give the necessary tools for the security analysis of our schemes. Last, in sections 4 and 5 we present our secret-key and public-key constructions respectively.

2 Quantum Money Definitions

In this section we give the definitions for quantum money. We first define secret-key quantum money schemes where there is a verification protocol run between a user and the bank in order to verify a coin. We give a definition of secret-key quantum money mini-schemes, and claim that there is a direct way to go from a mini-scheme to a full scheme [6], similar to the public-key case [1]. Then, we give the definition proposed by Aaronson and Christiano [1] of
a public-key quantum money scheme as well as the mini-scheme and we state their standard construction theorem that makes a full public-money scheme out of a mini-scheme using signatures.

2.1 Secret-key Quantum Money

Informally, a secret-key quantum money scheme consists of an algorithm that is used by the bank in order to create valid coins, and a protocol that is run between a holder of a coin and the bank in order for the holder to verify that the coin is valid. The security requirement states that it is impossible for an algorithm to create more coins than what it had in the beginning.

Definition 1 (Secret-key Quantum Money). A quantum money scheme with classical verification consists of an algorithm Bank and a verification protocol Ver such that

1. Bank$(1^n) = \$ = (\rho, s_n)$ is the algorithm that creates a quantum coin $\$\$ where $\rho$ is a quantum state and $s_n$ is a classical serial number.
2. Ver is a protocol with classical communication, run for a coin $\$, between a holder $H$ of a number of coins and the bank $B$. The final message of this protocol is a bit $b$ sent by the bank, that corresponds to whether the coin is valid or not. Denote by $Ver_B^H(\)$ this final bit.

- Correctness: The scheme is correct if for every honest holder $H$, $Pr[Ver_B^H(Bank(1^n)) = 1] = 1 - negl(n)$.
- Security: The scheme is secure if for any quantum adversary $F$ who possesses $q$ coins, interacts at most $t$ times with the bank and finally produces $q'$ coins $\$, it holds that

$$Pr \left( \bigwedge_{i \in [q']} Ver_B^H(\_i) = 1 \right) \leq poly(t) \cdot negl(n)$$

where $H$ is any honest holder.

In general, the security parameter $n$ corresponds to the number of qubits a valid coin consists of. Note that, although the adversary $F$ may deviate from the verification protocol in an attempt to create more coins, these coins will be checked for validity by an honest holder who will correctly follow the protocol. Note that the previous definition gives information theoretic security; the adversary $F$ is not restricted to be computationally efficient.
New constructions for Quantum Money

As studied by Aaronson and by Gavinsky, it is enough to prove the security of a smaller scheme (mini-scheme) in order to guarantee security of the full scheme. In the mini-scheme, the adversary \( F \) possesses only one coin \( \$ \) and interacts \( t \) times with the bank in order to create two coins. Therefore, the security game of the mini-scheme is as before, but the adversary is allowed to run Ver only for its unique coin \( \$ \). In this case where the verification includes interaction with the bank, note that the coin does not need to have a classical serial number.

▶ **Definition 2** (Secret-key Quantum Money Mini-Scheme). A quantum money mini-scheme with classical verification consists of an algorithm \( \text{Bank} \) and a verification protocol \( \text{Ver} \) such that

1. \( \text{Bank}(1^n) = \$ = \rho \) is the algorithm that creates a quantum coin \( \$ \) where \( \rho \) is a quantum state.
2. \( \text{Ver} \) is a classical protocol, run between a holder \( H \) of \( \$ \) and the bank \( B \). The final message of this protocol is a bit \( b \in \{0,1\} \) sent by the bank, that corresponds to whether the coin is valid or not. Denote by \( \text{Ver}_H^B(\$) \) this final bit.

- Correctness: The scheme is correct if for every honest holder \( H \), \( \Pr[\text{Ver}_H^B(\text{Bank}(1^n)) = 1] = 1 - \text{negl}(n) \).
- Security: The scheme is secure if for any quantum adversary \( F \) who interacts at most \( t \) times with the bank and finally produces two coins \( \$, \$_1, \$_2 \) it holds that

\[
\Pr \left[ \left( \text{Ver}_H^B(\$_1) = 1 \wedge \text{Ver}_H^B(\$_2) = 1 \right) \right] \leq \text{poly}(t) \cdot \text{negl}(n)
\]

where \( H \) is any honest holder.

In order to go from a secret-key quantum money mini-scheme to a full scheme, it is enough for the bank to add a serial number to a coin of the mini-scheme. Then, consulting that serial number the bank can run the verification protocol of the mini-scheme for that coin.

▶ **Lemma 3** (Mini-scheme to full scheme [6]). There exists a secure secret-key quantum money scheme \( M \) with classical verification if and only if there exists a secure secret-key quantum money mini-scheme \( m \) with classical verification.

### 2.2 Public-key Quantum Money

We now give the definition of a public-key quantum money scheme [1]. In this case we have three algorithms; one that creates a public key and a secret key, one that uses the secret key to create coins, and one that uses the public key to verify that a coin is valid.

▶ **Definition 4** (Quantum Money [1]). A public-key quantum money scheme \( M \) consists of three algorithms:

1. \( \text{KeyGen}(1^n) = (\text{sk}, \text{pk}) \) that returns a secret key \( \text{sk} \) and a public key \( \text{pk} \).
2. \( \text{Bank}(\text{sk}) = \$ \) a randomized algorithm that takes as input the secret key and returns a coin \( \$ \).
3. \( \text{Ver}(\text{pk}, \$) = 0/1 \) that takes as input the public key \( \text{pk} \), and a potential coin \( \$ \) and either accepts or rejects.

- Correctness: \( M \) is correct if for a pair \( (\text{sk}, \text{pk}) \) that is an output of KeyGen it holds that \( \text{Ver}(\text{pk}, \text{Bank}(\text{sk})) = 1 - \text{negl}(n) \)
Security: $M$ is secure if for any polynomial time quantum adversary $F$ that takes as input the public key $pk$ and $q$ valid coins $\$_1, \cdots, \$_q$ and outputs $q'$ potential coins $\$'_1, \cdots, \$'_q'$, it holds that
\[
\Pr \left( \bigwedge_{i \in [q']} \text{Ver}(pk, \$'_i) = 1 \right) \wedge (q' > q) = \text{negl}(n)
\]
Here, $n$ is the security parameter of the scheme and corresponds to the number of bits of $sk$ as well as the number of qubits of each coin.

Now, as before, we give the notion of public key mini-schemes. A mini-scheme consists only of an algorithm that creates a coin and an algorithm that verifies a coin. Here the coin is of the form $(s, \rho)$ where $s$ is a classical string and $\rho$ is a quantum state. Although anyone can create a coin that passes the verification test (the creation algorithm is public), the security property states that no algorithm that takes a coin with serial number $s$ can create an extra valid coin with the same serial $s$.

\textbf{Definition 5} (Quantum Money mini-scheme [1]). A public-key quantum money mini-scheme $M$ consists of two algorithms:
1. $\text{Bank}(1^n) = \$ = (s, \rho)$ a randomized algorithm that returns a coin $\$$, where $s$ is a classical serial number and $\rho$ is a quantum state.
2. $\text{Ver}(\$) = 0/1$ that takes as input a potential coin $\$'$ and either accepts or rejects.

Correctness: $M$ is correct if it holds that $\text{Ver}(\text{Bank}(1^n)) = 1$

Security: $M$ is secure if for any polynomial time quantum adversary $F$ that takes as input a coin $(s, \rho)$ and outputs two quantum states $\rho_1, \rho_2$ it holds that
\[
\Pr [(\text{Ver}(s, \rho_1) = 1 \wedge \text{Ver}(s, \rho_2) = 1)] = \text{negl}(n)
\]
Here, $n$ corresponds to the number of qubits of $\rho$.

The tool that Aaronson and Christiano use in order to go from a public money mini-scheme to a full scheme is digital signatures that are secure against quantum adversaries.

\textbf{Definition 6}. A signature scheme $S$ consists of three algorithms:
1. $\text{KeyGen}(1^n) = (sk, pk)$ that returns a secret key $sk$ and a public key $pk$.
2. $\text{Sign}(sk, m) = s$ that takes as input a secret key and a message $m$ and returns its signature $s$.
3. $\text{Ver}(pk, m, s) = 0/1$ that takes as input the public key $pk$, a message $m$ and a potential signature $s$ and either accepts or rejects.

Correctness: $S$ is correct if for a pair $(sk, pk)$ that is an output of KeyGen it holds that $\text{Ver}(pk, m, \text{Sign}(sk, m)) = 1$

Security: The security of $S$ is defined by the following game between a Challenger $\mathcal{C}$ and an adversary $\mathcal{F}$. $\mathcal{C}$ runs $\text{KeyGen}(1^n)$ and creates a pair $(sk, pk)$ and gives $pk$ to $\mathcal{F}$. $\mathcal{F}$ picks messages $m_1, \cdots, m_q$ of its choice and gives them to $\mathcal{C}$. $\mathcal{C}$ using $sk$ signs these messages and replies with their signatures $s_1, \cdots, s_q$. Finally, $\mathcal{F}$ outputs a message-signature pair $(m^*, s^*)$ and wins if this pair is different from all other pairs $(m_i, s_i)$ for all $i \in [q]$ and if $\text{Ver}(pk, m^*, s^*) = 1$. $S$ is existentially unforgeable under non-adaptive chosen message attacks if for every polynomial time quantum adversary $\mathcal{F}$ it holds that $\Pr[\text{Ver}(pk, m^*, s^*) = 1] = \text{negl}(n)$.

Here, $n$ is the security parameter of the scheme and corresponds to the number of bits of $sk$. 
New constructions for Quantum Money

Theorem 7 (Standard Construction [1]). If there exists a secure public-key quantum money mini-scheme and if there exists an existentially unforgeable under non-adaptive chosen message attacks signatures scheme, then there exists a secure public-key quantum money scheme.

Briefly, in this standard construction, a full coin consists of a coin from the mini-scheme combined with a signature of its serial number.

In the following, therefore, we focus on constructing a secret-key and a public-key mini-scheme and these can be extended to full schemes using the previous constructions.

Tools for security analysis

In this section we define an important tool towards the construction of quantum money, that of quantum retrieval games (QRG). From a QRG we go through some intermediate notions of QRG that are more convenient for our money schemes and prove the equivalence between them.

3.1 Quantum Retrieval Games

Suppose that we have an encoding function that takes as input a classical string \( x \) and gives as output an encoding \( \tilde{\rho}_x \), which in the quantum case is a mixed quantum state. Suppose, furthermore, that \( x \) is chosen from some distribution and is described by a random variable \( X \). How easy is it for an algorithm that takes as input only \( \tilde{\rho}_x \) to answer a question about \( x \)? A good way to formalize this question is via a relation \( \sigma \). Then, we would like to know how well an optimal algorithm can find an answer \( a \) such that \( (x, a) \in \sigma \). For example, \( \sigma \) could be the identity \( (x, a) \in \sigma \) if and only if \( x = a \) or a function \( g \) \( (x, a) \in \sigma \) if and only if \( a = g(x) \). In the most general setting \( \sigma \) is a relation and therefore there are several valid answers. Informally, in a quantum retrieval game, an algorithm takes as input \( \tilde{\rho}_x \) and wants to find an answer for \( x \). In order to succeed in this, it has to find the best decoding procedure that, when applied to \( \tilde{\rho}_x \), will give a valid answer. In the quantum case, the best decoding procedure corresponds to the best measurement of the state \( \tilde{\rho}_x \) and the probability that this best measurement will give a valid answer is called the physical value of the game.

Note that if \( \tilde{\rho}_x \) is a mixed quantum state it holds that \( \text{Tr}[\tilde{\rho}_x] = 1 \). By defining \( \rho_x = \text{Pr}[X = x] \cdot \tilde{\rho}_x \) we can integrate the randomness of \( x \) into the state \( \rho_x \). Note that \( \rho_x \geq 0 \), \( \text{Tr}[\rho_x] \leq 1 \), \( \text{Tr}[X = x] = \text{Pr}[\rho_x] \) and \( \text{Tr}[\sum_x \rho_x] = \text{Tr}[\sum_x \text{Pr}[X = x] \cdot \tilde{\rho}_x] = \sum_x \text{Pr}[X = x] \cdot \text{Tr}[\tilde{\rho}_x] = 1 \).

It is common to call the string \( x \) a secret that takes values from a set of secrets \( S \), \( a \) a potential answer that takes values from a set of answers \( A \) and \( \rho_x \) the quantum state that is the encoding of \( x \). A decoding procedure is a general measurement on the state \( \rho_x \) with operators \( \{m_a\}_{a \in A} \), each one corresponding to a possible answer.

Definition 8 (Quantum Retrieval Games [6]). Let \( S, A \subseteq \mathbb{N}, \sigma \subseteq S \times A \) and \( \forall x \in S \) let \( \rho_x \geq 0 \) such that \( \text{Tr}[\sum_{x \in S} \rho_x] = 1 \). Then the tuple \( G = (S, A, \{\rho_x\}_{x \in S}, \sigma) \) is called a quantum retrieval game (QRG). The physical value of \( G \) is denoted by \( \text{PVal}(G) \) and is the maximum probability of correctly decoding a state; i.e. producing an answer \( a \in A \) such that \( (x, a) \in \sigma \) (where the probability is taken over the randomness of \( x \) and the randomness of the decoding procedure).

The physical value of a game can be expressed as the solution of the semidefinite program of figure 1. In several cases we are interested in an upper bound of the physical value of a game. Towards this, it is convenient to define the selective value of the game \( \text{SVal}(G) \).
which describes the best decoding probability when the measurements \( \{ m_a \}_{a \in A} \) satisfy the property: \( \sum_{a \in A} m_a \leq I \). In other words, the selective value of the game corresponds to the solution of the relaxation of the SDP of the physical value (figure 2) and in general it is not achievable, yet easier to manipulate. It is clear that the selective value of a game is always greater or equal to its physical value and, thus, an upper bound of the selective value gives also an upper bound of the physical value. The following theorem gives an easy way to compute the selective value of a game.

**Theorem 9 (Selective Value [19]).** Let \( G = (S, A, \{ \rho_x \}_{x \in S}, \sigma) \) be a QRG. Let also \( \rho = \sum_{x \in S} \rho_x \) and \( O_a = \sum_{x: (x, a) \in \sigma} \rho^{-1/2} \rho_x \rho^{-1/2} \). Then \( \text{SVal}(G) = \max_a \| O_a \| \).

This equality is useful since it is possible to find the selective value of a game without going through any specific measurement.

In the case we want to play a big QRG that consists of playing in parallel many small QRGs, it is useful to know what happens to the physical value of that big game. The following lemma states that the selective value of such a game is multiplicative and therefore the probability of winning all the QRGs drops exponentially fast on the number of small games.

**Lemma 10 (Parallel Repetition [19]).** Let \( G_1 = (S_1, A_1, \{ \rho_{x_1} \}_{x_1 \in S_1}, \sigma_1) \) and \( G_2 = (S_2, A_2, \{ \rho_{x_2} \}_{x_2 \in S_2}, \sigma_2) \) be two QRGs. Let also \( \rho = \sum_{x \in S} \rho_x \) and \( \rho_x = \sum_{x: (x, a) \in \sigma} \rho^{-1/2} \rho_x \rho^{-1/2} \). Then \( \text{SVal}(G) = \text{SVal}(G_1) \cdot \text{SVal}(G_2) \).

Let \( M_1, M_2 \) be the solutions that optimize the selective value for the games \( G_1, G_2 \) respectively. Then the previous equality states that the optimal solution for the product game \( G \) is just the product of the two solutions. This provides an upper bound on the physical value of the product game \( G \), which is the product of the selective values of the games \( G_1, G_2 \). It is clear that by taking the product of \( n \) games with constant selective value \( \epsilon \), we can create a game whose physical value is at most \( \epsilon^n \).

For the construction of our money scheme, it is useful to define another notion of a QRG, that of 1-out-of-2 QRG. Here, an algorithm is given as before a state \( \rho_x \), but now two relations \( \sigma_a, \sigma_b \). The basic property that we expect from such a game is that it should be impossible for any quantum algorithm (quantum measurement), to answer with high probability both relations correctly, but it is still possible to answer correctly one of them.

**Definition 11 (1-out-of-2 QRG).** For a set of secrets \( S \), set of answers \( A \), and two relations \( \sigma_a, \sigma_b \) we define: \( G_a = (S, A, \{ \rho_x \}_{x \in S}, \sigma_a) \), \( G_b = (S, A, \{ \rho_x \}_{x \in S}, \sigma_b) \), \( G_c = (S, A \times A, \{ \rho_x \}_{x \in S}, \sigma) \) where \( (x, (a, b)) \in \sigma \) if and only if \( (x, a) \in \sigma_a \) and \( (x, b) \in \sigma_b \). We say that \( G = (S, A, \{ \rho_x \}_{x \in S}, \sigma_a, \sigma_b, \sigma) \) is an \( \varepsilon-(\frac{\ell}{2}) \) QRG if it satisfies the following properties:
New constructions for Quantum Money

1. Correctness: There exist measurements $M^{(a)}$, $M^{(b)}$ such that $(M^{(a)}(\rho_x), x) \in \sigma_a$ and $(M^{(b)}(\rho_x), x) \in \sigma_b$. Equivalently: $\text{PVal}(G_a) = \text{PVal}(G_b) = 1$

2. Security: $\text{PVal}(G_c) \leq \varepsilon$

3. Independence: The probability of winning $G_a$ ($G_b$) is independent of knowing $\sigma_b$ ($\sigma_a$).

The third property states that even knowing all the answers of one of the relations, this does not increase the probability of winning the other relation at all. We will call a $(\frac{1}{2})$QRG secure if $c = 1 - \text{negl}(n)$ and $\varepsilon \leq \text{negl}(n)$ where $n$ is the size of the secret $x$.

Theoretically, it is possible to create games with perfect correctness. However, in practice it is reasonable to assume that errors may occur and therefore, the correctness may not be guaranteed. In this case, we can assume that the games $G_a$ and $G_b$ cannot be answered correctly with probability 1 but only with a constant probability $c < 1$. Then, we can define an 1-out-of-2 game as $(c, \varepsilon) - (\frac{1}{2})$QRG where $\varepsilon$ is again the security of the scheme. We can show that if we repeat such a game $n$ times, we can create a $(c', \varepsilon') - (\frac{1}{2})$QRG where $c'$ is now exponentially close to 1 and $\varepsilon'$ is exponentially close to 0.

**Lemma 12.** Let $c, \varepsilon, \delta$ be positive constants such that $\delta = \frac{2c-\varepsilon-1}{3}$. If there exists a $(c, \varepsilon) - (\frac{1}{2})$QRG $G$, then there exists a $\left(1 - e^{-\frac{a^2}{2}\delta^2}, e^{-\frac{a^2}{2}\delta^2}\right) - (\frac{1}{2})$QRG $G'$.

Note that even though the original “small” game may have a considerably large error probability, we can achieve a quantum retrieval game that tolerates the errors with probability exponentially close to 1.

### 3.2 QRGs with Verification

We now define a new version of QRG, that of QRG with verification $(\frac{1}{3})$QRGv that is useful for the construction of our money schemes. Informally, in a $(\frac{1}{3})$QRGv, an adversary has some extra help for finding an answer to $\sigma_a$ and $\sigma_b$; he is allowed to ask multiple queries of whether an answer is correct for a relation. What we require from such a game, is that the winning probability of any such adversary does not increase more than polynomially on the number of queries it asks.

**Definition 13** $(\frac{1}{2})$QRGv. Let $G = (S, A, \{\rho_x\}_{x \in S}, \sigma_a, \sigma_b, \sigma)$ be a $(c, \varepsilon) - (\frac{1}{2})$QRG. We define the following game $G$ between an adversary $\mathcal{F}$ and an algorithm $\mathcal{C}$. $\mathcal{C}$ prepares a normalized state $\rho_x/\text{Tr}[\rho_x]$ of the game $G$ and gives it to $\mathcal{F}$. Then $\mathcal{F}$ is allowed to interact with $\mathcal{C}$ at most $t$ times in the following way:

1. $\mathcal{F}$ picks a $\sigma' \in \{\sigma_a, \sigma_b\}$, generates an answer $d$ and sends $(\sigma', d)$ to $\mathcal{C}$.
2. $\mathcal{C}$ returns 1 if and only if $(x, d) \in \sigma'$.

After $t$ interactions $\mathcal{F}$ outputs $(a_1^*, a_2^*)$ and wins if and only if $(x, (a_1^*, a_2^*)) \in \sigma$. We say that $G$ is a $(c, \varepsilon) - (\frac{1}{2})$ quantum retrieval game with verification $(\frac{1}{2})$QRGv if it satisfies the following properties:

1. **Correctness**: Given any state $\rho_x/\text{Tr}[\rho_x]$ the probability of answering $\sigma_a$ (or $\sigma_b$) is at least $c$.
2. **Soundness**: For any $t$ and for any adversary $\mathcal{F}$ interacting the way defined above, it holds that $\text{Pr}[\mathcal{F} \text{ wins}] \leq \text{poly}(t) \cdot \varepsilon$

We would like to argue the following: allowing such an adversary $\mathcal{F}$ to check whether a query $(\sigma', d)$ is correct, does not increase considerably his probability of winning. Therefore, for an exponentially small $\varepsilon$ an adversary would require a superpolynomial number of such interactions in order to have a non-negligible probability of winning. Towards this, we define below a more restricted version of the game and we show that this definition is equivalent to that of a $(\frac{1}{2})$QRGv.
We now restrict the adversary $F$ in the following manner. Suppose that $F$ is allowed to interact with $C$ as previously and the $i$-th interaction is the first interaction when he sends to $C$ some $(\sigma', a)$ such that $(x, a) \in \sigma'$ (without loss of generality we can assume that $\sigma' = \sigma_a$).

Then, for the remaining $t - i$ interactions $F$ is allowed to play only with $\sigma_b$. We call this game a restricted 1-out-of-2 quantum retrieval game with verification $(\binom{t}{1})rQRGv$ and the adversary $F$ a restricted adversary. It can be proven that since finding an answer for $\sigma_a$ is independent from any answer of $\sigma_b$, these two games are equivalent. In other words, allowing the adversary to succeed in more than one interaction with the same relation $\sigma'$ does not help him win the game more than succeeding only once for $\sigma'$.

► **Lemma 14.** Let $G = (S, A, \{\rho_x\}_{x \in S}, \sigma_a, \sigma_b, \sigma)$ be a $(c, \varepsilon) = \binom{t}{1}QRGv$. Then $G$ is also a $(c, \varepsilon) - (\binom{t}{1})rQRGv$.

Using the previous lemma we can show that an adversary that is allowed to ask at most $t$ queries regarding the relations $\sigma_a, \sigma_b$ does not increase significantly his probability of winning. More specifically, it can be shown that if the original $\binom{t}{1}QRG$ has security $\varepsilon$ then allowing interaction, increases the winning probability at most quadratically in $t$.

► **Theorem 15.** If there exists a $(c, \varepsilon) - \binom{t}{1}QRG$ $G'$ then there exists a $(c, \varepsilon) - \binom{t}{1}QRGv$ $G$. In particular, any adversary against $G$ has winning probability of at most $4t^2 \cdot \varepsilon$, where $t$ is the number of queries.

### 3.3 One-time Memories

For the creation of our public-key scheme we will use the notion of one-time memories (OTM) defined by Goldwasser et al in [8]. OTM are essentially devices which contain two secrets $x_a, x_b$, however, we are able to extract only one of these secrets. There is a very natural connection between $\binom{t}{1}QRG$ and OTMs as we will see below.

► **Definition 16.** A $(c, \varepsilon)$—one-time memory (OTM) is a device that has the following behavior. Suppose that the device is programmed with two $n$-bit messages $x_a, x_b$ chosen from some distribution $D$. Then:

1. Correctness: There exists an honest strategy $M^{(a)}$ that interacts with the device and recovers the message $x_a$ with probability $c$. Likewise, there is an honest strategy $M^{(b)}$ that interacts with the device and recovers the message $x_b$ with probability $c$.

2. Security: For any strategy $M$, if $X$ is the random variable corresponding to the classical output of $M$, then $\Pr[X = (x_a, x_b)] \leq \varepsilon$.

We will call the OTM secure if $c = 1 - \text{negl}(n)$ and $\varepsilon = \text{negl}(n)$.

Note that in this paper, we deal with quantum OTM, namely the “device” is a quantum state $\rho_{x_a, x_b}$. Although secure OTM are impossible in the plain quantum model even with computational assumptions, Liu [10, 11, 12] has shown that OTM are possible in the isolated qubits model, where an adversary can use only local operations and classical communication. His OTM construction is a quantum state that consists of qubits that do not need to be entangled and thus it is easier and more efficiently implementable.

It is not hard to see that OTM are equivalent to $\binom{t}{1}QRG$ restricted so that the relations $\sigma_a, \sigma_b$ are, in fact, functions.

► **Lemma 17.** There exists a secure $\binom{t}{1}QRG$ $G = (S, A, \{\rho_x\}_{x \in S}, \sigma_a, \sigma_b, \sigma)$ such that the relations $\sigma_a, \sigma_b$ are functions if and only if there exists a secure OTM.

**Proof.** Using $G$ we can create an OTM with secrets $x_a = \sigma_a(x)$ and $x_b = \sigma_b(x)$. The OTM device is simply $\rho_x$. Clearly, if there exists an algorithm that can retrieve both secrets from
the OTM then this algorithm can also break $G$. For the opposite direction, the role of the encoding $\rho_x$ is played by the OTM device, which is a quantum state. The secret $x$ of $G$ is defined as the concatenation of $x_a$ and $x_b$ and the functions $\sigma_a, \sigma_b$ are defined such that $\sigma_a(x_a|x_b) = x_a$ and $\sigma_b(x_a|x_b) = x_b$. Clearly, if there exists an algorithm that can retrieve answers for both $\sigma_a$ and $\sigma_b$ from the encoding $\rho_x$ then this algorithm can also break the OTM.

Similarly to the QRG with verification, we can define \( (c, \varepsilon) - \text{one-time memories with verification} \) (OTMv); where the adversary is allowed to choose $d \in \{a, b\}$ and $y \in \{0, 1\}^n$ and ask whether $x_d = y$. Again, a secure OTMv means that $c \geq 1 - \text{negl}(n)$ and $\varepsilon \leq \text{negl}(n)$. However, as we have shown, such a power does not really help the adversary.

Finally, a hash based OTMv (hOTM) is an OTMv where the adversary instead of being allowed to interact in order to find an answer, it is given as input the hashes of the two answers $H(x_a), H(x_b)$. This way, if an answer is correct, the adversary can verify that on its own. It can be shown that if the original OTMv is secure, then the hash based OTMv is still secure in the random oracle model.

A random oracle is essentially an oracle that behaves as follows. First, it keeps a list $L$ of pairs of the form $(x, y)$ where $x$ is an element of its domain and $y$ is an element of its range. In the beginning $L$ is empty. On input $x_0$, first it searches $L$ for a pair of the form $(x_0, y_0)$, and if such a pair exists in $L$ then it returns $y_0$. Otherwise, it picks a uniformly random element $y_0$ from its range, inserts $(x_0, y_0)$ in the list $L$, and returns $y_0$. Hash functions are usually assumed to have this ideal property when other properties such as one-wayness or collision resistance are not enough for a security proof. When hash functions are used as random oracles in a proof that a scheme is secure, we say that the scheme is secure in the random oracle model.

\begin{definition}
A hash based one-time memory (hOTM) is a device that has the following behavior. Suppose that the device is programmed with two $n$-bit messages $x_a, x_b$ chosen from some distribution $D$. Then:

1. Correctness: There exists an honest strategy $M^{(a)}$ that interacts with the device and recovers the message $x_a$ with probability $c = 1 - \text{negl}(n)$. Likewise, there is an honest strategy $M^{(b)}$ that interacts with the device and recovers the message $x_b$ with probability $c = 1 - \text{negl}(n)$.

2. Security: For any polynomial time strategy $M$ that takes as input the hash values $H(x_a), H(x_b)$, if $X$ is the random variable corresponding to the classical output of $M$, then $\Pr[X = (x_a, x_b)] \leq \text{negl}(n)$.

Note that in contrast to the previous definitions, the security of a hOTM is computational.
\end{definition}

\begin{lemma}
A secure OTMv is also a secure hOTM in the random oracle model.
\end{lemma}

\begin{proof}
Suppose that there exists a polynomial algorithm $F$ that is able to break the hOTM property. We can create an algorithm $A$ against the OTMv property. $A$ takes as input a state $\rho_{x_a,x_b}$ and is allowed to ask verification queries of the form $(d, y)$, where $d \in \{a, b\}$ and receive an answer 1 if and only if $x_d = y$. $A$ initiates $F$ by choosing two random values $(\alpha, \beta)$ as the hashes of the answers and giving to $F$ the tuple $(\rho_{x_a,x_b}, \alpha, \beta)$. When $F$ asks for the hash of a value $y$, $A$ makes two queries of the form $(a, y)$, $(b, y)$ and if one of them accepts, $A$ returns to $F$ the value $\alpha$ or $\beta$ depending on which of the two queries accepted. If none of the two accepted, then $A$ returns a random (but consistent with the previous queries) value to $F$ as a hash of $y$. When $F$ outputs its two final answers $(x_a^*, x_b^*)$, $A$ also outputs $(x_a^*, x_b^*)$. We can see that $F$ always takes proper answers to its queries ($F$ is allowed to ask only for hash
values) and therefore works as if it attacks the hOTM. Since \( F \) is a polynomial algorithm, it cannot ask more than a polynomial number of hash values and therefore \( A \) cannot have asked more than a polynomial number of queries. Thus, if the winning probability of \( F \) is non-negligible, \( A \) has also a non-negligible winning probability. ▶

4 Secret-key Quantum Money Construction

In this section we create a secret-key mini-scheme and we analyze its security. Our scheme, in contrast to that proposed by Gavinsky [6], allows a one-round protocol between the bank and the user to accomplish the verification of a coin: a query to the bank and an answer by the bank that states whether the coin is valid or not. Therefore, in our scheme the bank does not need to maintain memory during the verification procedure; it just consults its secret database and returns the result. In the scheme of Gavinsky, however, the verification protocol consists of six messages during which, the bank has to maintain a temporary memory associated with a specific coin. Furthermore, unlike the scheme of Gavinsky, our proof of security is simpler, more modular and it includes noise and losses.

Gavinsky has shown that a \( \binom{2}{1} \) QRG with the following parameters exists:

▶ Theorem 20 (Hidden Matching QRG [6, 7]). There exists a \( \binom{1}{0} (1 - 2^{-n}, 2^{-n}) - \binom{1}{1} \) QRG \( G \).

Starting from this and using theorem 15 we can create a \( \binom{1}{1} \) QRG with the same parameters. Our construction is essentially a way of going from a \( \binom{1}{1} \) QRGv to a mini-scheme. Then, using the reduction from a mini-scheme to a full-scheme, the existence of a \( \binom{1}{1} \) QRG leads to the existence of a full quantum money scheme. The sequence of reductions appears in figure 3.

![Figure 3 From a \( \binom{1}{1} \) QRG to a secret-key quantum money scheme](image)

We now propose our construction that uses a \( \binom{1}{1} \) QRGv to create a mini-scheme. The algorithm Bank and the protocol Ver are defined as follows:

<table>
<thead>
<tr>
<th>Bank(1(^n^2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. For ( i \in [n] ) create ( G_i = (S, A, { \rho_{x_i} }_{x_i}, \sigma_a, \sigma_b, \sigma) ), ( (1 - 2^{-n}, 2^{-n}) - \binom{1}{1} ) QRGv.</td>
</tr>
<tr>
<td>2. Create a classical binary register ( r ) of size ( n ) and initialize it to ( 0^n ).</td>
</tr>
<tr>
<td>3. Return the state ( s = (\otimes_i \rho_{x_i}, r) ) as a coin for the mini-scheme.</td>
</tr>
</tbody>
</table>

Verification protocol for a coin \( s = (\otimes_i \rho_{x_i}, r) \):

1. The holder creates an empty set \( L \). Then, for each \( i \in [n] \) such that \( r_i = 0 \), the holder puts \( i \) in the set \( L \) with probability \( 1/n^{1/3} \). For each \( i \in L \) the holder picks at random a relation \( \sigma'_i \leftarrow \{ \sigma_a, \sigma_b \} \) and applies to \( \rho_{x_i} \) the measurement \( M^{(a)} \) if \( \sigma'_i = \sigma_a \) or \( M^{(b)} \) if \( \sigma'_i = \sigma_b \), in order to retrieve an answer \( d_i \). Furthermore, for all \( i \in L \) the holder sets \( r_i = 1 \). Finally, the holder sends to the bank the \( i \)'s he has picked, the relation he has picked for each \( i \), as well as the answers \( d_i \). |
2. The bank compares the answers it has received with its secret \(x_1 \cdots x_n\) and accepts if all answers are correct; namely if for all \(i \in L\) it holds that \((x_i, d_i) \in \sigma'_i\).

**Remark.** The coin is returned to the bank for replacement when the hamming weight of \(r\) is greater than \(n/4\) (more than \(n/4\) of the \(\rho_{x_i}\) are marked as used). Note that the scheme consists of \(O(n^2)\) qubits in total (there are \(n\) states \(\rho_{x_i}\) and each state consists of \(O(n)\) qubits) and that the verification protocol consists of only one round.

**Theorem 21.** The scheme is secure; namely any (even computationally unbounded) adversary who interacts with the bank at most \(t\) times has winning probability of at most \(e^{-n^2/3}/8 + 4t^2 \cdot n \cdot 2^{-n} \).

**Proof.** Suppose there is an adversary \(A\) who knows the answers to the relations, whereas for the \(i\)-th state he has as input. For clarity we will denote the secret \(F\) for the mini-scheme, namely when \(F\) receives as input a valid coin \(S\) and after running \(t\) verification protocols with the bank, he can produce two coins \(S' = (\rho'_1, \cdots, \rho'_n, r'), S'' = (\rho''_1, \cdots, \rho''_n, r'')\) that can pass the verification protocol with non-negligible probability \(\varepsilon\) greater than \(p(t) \cdot 2^{-n}\), for all polynomials \(p\). Then, one can create an adversary \(A\) for the \((1 - 2^{-n}, 2^{-n}) - (\frac{t}{2})\) QRG, namely when \(A\) receives from the algorithm \(C\) as input a state \(\rho^*\) that is the encoding of a secret \(x^*\), and after interacting \(t\) times with the algorithm \(C\), he can win the game with probability greater than \(p(t) \cdot 2^{-n}\) for all polynomials \(p\). By theorem 15 this also implies breaking the security of the \((\frac{t}{2})\) QRG.

Let \(A\) receive as input the state \(\rho^*\), that is the encoding of a secret \(x^*\). He creates an input for \(F\) in the following way:

**Bank** \((1^{n^2})\):
1. Pick at random \(i^* \leftarrow [n]\).
2. For \(i \in [n] - \{i^*\}\) create \(G_i = (S, A, \{\rho_{x_i}\}_{x_i}, \sigma_a, \sigma_b, \sigma)\) where \(G_i\) is a \((1 - 2^{-n}, 2^{-n}) - (\frac{t}{2})\) QRG.
3. Create a classical binary register \(r\) of size \(n\) and initialize it to \(0^n\).
4. Return to \(F\) the coin \(S = (\rho, r)\), where \(\rho = \rho_{x_1} \otimes \cdots \otimes \rho_{x_{i^*}} \otimes \rho^* \otimes \rho_{x_{i^*+1}} \otimes \cdots \otimes \rho_{x_n}\).

In other words, \(A\) creates a totally valid coin, but in the \(i^*\)-th position he puts the state he has as input. For clarity we will denote the secret \(x^*\) as \(x_{i^*}\). Note that \(A\) is able to pretend to be the honest bank during the verification protocol with \(F\); since for all \(i \in [n] - \{i^*\}\) he knows the answers to the relations, whereas for the \(i^*\)-th state, he can use his own interaction with the algorithm \(C\) in order to decide whether the query asked by \(F\) is correct. Therefore, \(A\) simulates the verification protocol between the bank and \(F\) in the following way:

1. \(A\) receives from \(F\) a set \(L\) of \(i\)'s, a set of challenges \(\sigma'_i \in \{\sigma_a, \sigma_b\}\) and a set of answers \(d_i\) for each \(i \in L\).
2. \(A\) returns 1 if all answers are correct; namely, if \((x_i, d_i) \in \sigma'_i\) for all \(i \in L\). Note that for those \(i\)'s that are different from \(i^*\), \(A\) can easily consult his own secret \(x_i\) in order to find if the answer is correct. However, for \(i = i^*\), \(A\) can make a query \((\sigma'_{i^*}, d_{i^*})\) to the algorithm \(C\) in order to find if the answer \(d_{i^*}\) is correct.
Hence, $\mathcal{A}$ can provide $\mathcal{F}$ with a valid initial coin and simulate the bank in the $t$ verification protocols with $\mathcal{F}$, and in the end, he receives from $\mathcal{F}$ two coins $\$' = (r'_1, \ldots, r'_n, r''_1), \$'' = (r''_1, \ldots, r''_n, r''_t')$ that can pass a verification protocol with an honest verifier with non-negligible probability $\epsilon$. For the two coins $\$'$, $\$''$ to be considered as valid, there must be at least $3/4n$ of the $r'_i$'s denoted as valid and at least $3/4n$ of the $r''_i$'s denoted as valid (a state $r'_i$ is denoted as valid if $r'_i = 0$). Therefore, there are at least $n/2$ indices $i$ such that $r'_i = r''_i = 0$. We want to argue that there must be an index $i$ among them for which $\mathcal{A}$ can win the $\frac{1}{t}$QRGv game with probability greater than $p(t) \cdot 2^{-n}$ for all polynomials $p$, otherwise the probability that the adversary $\mathcal{F}$ could create two valid coins is negligible.

Let $I = \{i : r'_i = r''_i = 0\}$. Since the coins $\$'$, $\$''$ are denoted as valid, it holds that $|I| \geq n/2$. Assume now that two honest verifiers Ver', Ver'' run the verification protocol for the two coins respectively. Let $L^a, L^b$ be the sets chosen by the honest verifiers and $L' = \{i \in L^a \cap L^b : \sigma'_i = \sigma''_i = \sigma_0\}$, where $\sigma'_i, \sigma''_i$ are the relations chosen for the index $i$ by the verification protocols for the two coins. In other words, $L'$ contains the indices where chosen by both Ver' and Ver'' in such a way that Ver' chose $\sigma_0$ for this $i$ and Ver'' chose $\sigma_0$ for this $i$. It holds that $\Pr[i \in L^a \cap L^b] = 1/2^{t/3}$ and $\Pr[i \in L'] = 1/4n^{2/3}$. Therefore, $\Pr[\exists i \in I : i \notin L'] = \Pr[L'' = \emptyset] = (1 - 1/4n^{2/3})^{n/2} = e^{-n^{1/3}/8}$, since $|I| = n/2$.

In other words, the probability that there exists an $i$ with $r'_i = r''_i = 0$ such that Ver' chose it during the verification protocol and picked the relation $\sigma_0$ for it and Ver'' also chose it and picked $\sigma_0$ for it, is exponentially close to 1. Now it holds that

$$\Pr[\mathcal{F} \text{ wins}] = \Pr[\text{Ver'}(\$') = 1 \land \text{Ver''}(\$'') = 1]$$

$$= \Pr[L' = \emptyset] \cdot \Pr[\text{Ver'}(\$') = 1 \land \text{Ver''}(\$'') = 1 | L' = \emptyset] + \Pr[\text{Ver'}(\$') = 1 \land \text{Ver''}(\$'') = 1 | L' \neq \emptyset]$$

$$\leq e^{-n^{1/3}/8} + \Pr[\text{Ver'}(\$') = 1 \land \text{Ver''}(\$'') = 1 | L' \neq \emptyset]$$

$$\leq e^{-n^{1/3}/8} + \Pr[\exists i \in L' : (M^{(a)}(\rho'_i), x_i) \in \sigma_a \land (M^{(b)}(\rho''_i), x_i) \in \sigma_b]$$

where $M^{(a)}, M^{(b)}$ are the measurements applied to the states of the QRG’s in order to retrieve an answer to $\sigma_a, \sigma_b$ respectively. The last line comes from the fact that if $L'$ is not empty and both verifications succeed, then both verifications must succeed for all $i \in L'$. Therefore, if $\Pr[\mathcal{F} \text{ wins}] = \epsilon$ is non-negligible then $\Pr[\exists i \in L' : (M^{(a)}(\rho'_i), x_i) \in \sigma_a \land (M^{(b)}(\rho''_i), x_i) \in \sigma_b] \geq \epsilon - e^{-n^{1/3}/8}$ is non-negligible as well. This trivially implies that $\Pr[\exists i \in [n] : (M^{(a)}(\rho'_i), x_i) \in \sigma_a \land (M^{(b)}(\rho''_i), x_i) \in \sigma_b] \geq \epsilon - e^{-n^{1/3}/8}$. In other words, with non-negligible probability there exists an index $i$ for which both verifications succeed. At this point it is clear that the goal of $\mathcal{A}$ is just to guess that index $i$ and put $\rho^*$ in that position.

Overall, the adversary $\mathcal{A}$ works as follows: Upon receiving as input the state $\rho^*$, he picks a random position $i^*$, creates the valid coin for $\mathcal{F}$ as we described above, receives the states $\rho'_{i^*}$ and $\rho''_{i^*}$ from $\mathcal{F}$ and returns the answers $(M^{(a)}(\rho'_{i^*}), M^{(b)}(\rho''_{i^*}))$. Now it holds that

$$\Pr[\mathcal{A} \text{ wins}] = \Pr[\exists i \in [n] : (M^{(a)}(\rho'_{i^*}), x_{i^*}) \in \sigma_a \land (M^{(b)}(\rho''_{i^*}), x_{i^*}) \in \sigma_b]$$

$$\geq \frac{1}{n} \cdot \Pr[\exists i \in L' : (M^{(a)}(\rho'_i), x_i) \in \sigma_a \land (M^{(b)}(\rho''_i), x_i) \in \sigma_b]$$

which contradicts the fact that the security of the $\frac{1}{t}$QRGv is $2^{-n}$. 

M. Georgiou and I. Kerenidis 13
Therefore, since by theorem 15 the maximum winning probability of \( A \) is \( 4t^2 \cdot 2^{-n} \), the maximum winning probability of \( F \) is \( e^{-n^{1/3}}/n + 4t^2 \cdot 2^{-n} \cdot n. \)

\[ \triangle \]

### 5 Public-key Quantum Money Construction

In the construction of a public key scheme, it suffices to create a secure public-key mini-scheme, and this, combined with signatures, can give a full scheme [1]. The advantage of our construction is that it is a simple modification of the previous secret key one: the answers of the bank are encoded in their hash values. Therefore, instead of requiring from the user to communicate with the bank in order to find out if an answer is valid, the bank announces the hash values of the answers. It is clear that for a regular QRG there may exist too many answers and hence giving all these hashes as part of the coin would violate the correctness of the scheme. Hence, for our construction, we need to use QRG with functions or equivalently one-time memories. Despite the fact that quantum one-time memories do not exist unconditionally, they exist in the isolated qubits model.

\[ \triangleright \text{Theorem 22 ([10, 11, 12])}. \text{There exists a secure OTM in the isolated qubits model.} \]

Using this, together with lemma 19, we get the following corollary.

\[ \triangleright \text{Corollary 23}. \text{There exists a secure hOTM in the isolated qubits-random oracle model.} \]

Our purpose, now, is to go from hOTM to a public-key mini-scheme. The sequence of reductions appears in figure 4.

\[ \text{Figure 4} \text{ From an OTM to a full public-key quantum money scheme. OTMs in the random oracle model give hOTM. hOTM imply public-key mini-schemes which, together with signatures, imply public-key quantum money.} \]

Since for each OTM there are only two secrets, a hashing of each answer can be given as part of the coin. Then the verification algorithm works similarly to the secret key scheme. It chooses each state-game with probability \( 1/n^{1/3} \), it chooses at random whether to retrieve the first (\( x_a \)) or the second (\( x_b \)) secret for each game, it measures the OTM (using \( M^{(a)} \) or \( M^{(b)} \)) in order to retrieve an answer and finally it verifies that the hash value of that answer is the same as the one given. So the two algorithms Bank and Ver of the mini-scheme are the following:

\[ \text{Bank(1}^{n^2}) : \]
1. For \( i \in [n] \) create the OTM \( \rho_i \) with secrets \( x_i^a, x_i^b \in \{0,1\}^n \).
2. Create a classical binary register \( r \) of size \( n \) and initialize it to \( 0^n \).
3. Return \( S = ((h_1, g_1), \cdots, (h_n, g_n), \rho_1, \cdots, \rho_n, r) \) as a coin for the mini-scheme, where \( h_i = H(x_i^a), g_i = H(x_i^b) \) and \( H \) is the hash function. The string \( (h_1, g_1), \cdots, (h_n, g_n) \) corresponds to the classical serial number of the coin (that has to be signed in order to give a full coin), and \( (\rho_1, \cdots, \rho_n, r) \) is the quantum state.
We created a secret-key quantum money scheme that is unconditionally secure and has optimal communication: a single round of classical communication. We also provided a conceptually simpler and more modular proof. Moreover, if we instantiate the hidden Matching QRG with the Hidden Matching (1)QRG, we can tolerate an error rate of up to 12.5%; see lemma 12 in Appendix A. Note that in every verification of the coin we invalidate on average $n/4$ quantum states (each consisting of $4$ qubits) and thus the number of allowed verifications before the coin is returned to the bank is $n/(4 \cdot n^{1/3}) = n^{2/3}/4$. Therefore, for a coin of say $10^{12}$ qubits, we succeed 2,500 verifications on average. A polynomial number of verifications is optimal for unconditionally secure schemes, nevertheless, a natural question that still remains open is whether we can have computationally secure secret-key schemes that allow exponentially many classical verifications.

In addition, we showed how a simple extension of our secret key construction can give rise to a public-key quantum money scheme that is computationally secure against quantum adversaries in the random oracle model given one-time memories. We note that previous schemes were also based on non-standard computational assumptions. The main open question is to construct public-key quantum money that are provably secure based on some standard cryptographic assumptions such as one-way functions.

Acknowledgements We thank Jamie Sikora for bringing to our attention the semidefinite approach of QRG. This work was partially supported by the ERC project QCC, the ANR project RDAM and the EU project QAlgo.
References


4. Aharon Brodutch, Daniel Nagaj, Or Sattath, and Dominique Unruh. Bombs do not have to explode, forgers can get away scot-free (an adaptive attack on wiesner’s quantum money). 2014.


A Instantiating a QRG

As shown above, the existence of a \((c, \varepsilon) - \left(\frac{c}{2}\right)\) QRG implies the existence of a secret-key quantum money scheme as long as \(c\) is reasonably large and \(\varepsilon\) is any constant smaller than 1. To instantiate such a quantum money scheme one has to give specific quantum retrieval games with this property.

**Hidden Matching QRG[6, 7]**

**Definition 25.** The Hidden Matching \(\left(\frac{c}{4}\right)\) QRG \(G = (S, A, \{\rho_x\}_{x \in S}, \sigma_a, \sigma_b, \sigma)\) is defined as follows: \(S = \{0, 1\}^4, A = \{0, 1\} \times \{0, 1\}\), \(|\psi_x\rangle = \frac{1}{2} \sum_{i \in [4]} (-1)^{x_i}|i\rangle\), \(\rho_x = \frac{1}{16}|\psi_x\rangle\langle\psi_x|\).

The relation \(\sigma_\alpha\) is defined as \((x, (a, b)) \in \sigma_\alpha\) if and only if the following holds: if \(a = 0\) then \(x_1 \oplus x_2 = b\); if \(a = 1\) then \(x_3 \oplus x_4 = b\). Similarly, the relation \(\sigma_\beta\) is defined as \((x, (a, b)) \in \sigma_\beta\) if and only if the following holds: if \(a = 0\) then \(x_1 \oplus x_3 = b\); if \(a = 1\) then \(x_2 \oplus x_4 = b\).

**Lemma 26.** The Hidden Matching is a \((1, \frac{2}{3}) - \left(\frac{2}{3}\right)\) QRG.

**Proof.** The correctness in a noise-free environment we can be succeeded with zero error probability. Indeed, if we want to find an answer for the relation \(\sigma_a\) we measure in the basis \(\{\frac{|1\rangle + |2\rangle}{\sqrt{2}}, \frac{|1\rangle - |2\rangle}{\sqrt{2}}, \frac{|3\rangle + |4\rangle}{\sqrt{2}}, \frac{|3\rangle - |4\rangle}{\sqrt{2}}\}\) and we return the values \((a, b) = (0, 0), (0, 1), (1, 0), (1, 1)\) respectively. If we want to find an answer for the relation \(\sigma_b\) we measure in the basis \(\{\frac{|1\rangle + |3\rangle}{\sqrt{2}}, \frac{|1\rangle - |3\rangle}{\sqrt{2}}, \frac{|2\rangle + |4\rangle}{\sqrt{2}}, \frac{|2\rangle - |4\rangle}{\sqrt{2}}\}\) and we return \((a, b) = (0, 0), (0, 1), (1, 0), (1, 1)\) respectively.

For the security of the game, we use Theorem 9. By definition, we have \((x, (a_1, b_1), (a_2, b_2)) \in \sigma\) if and only if \((x, (a_1, b_1)) \in \sigma_a\) and \((x, (a_2, b_2)) \in \sigma_b\). It holds that \(\rho = \sum_{x \in \{0, 1\}^4} \rho_x = \frac{1}{16}I\) and therefore \(\rho^2 = 2I\). In order to find the selective value of the game \((S, A \times A, \{\rho_x\}_{x \in S}, \sigma)\) it is enough to consider one value of \(O_a\) for some possible answer \(a \in A \times A\). For example, by taking \(a = ((0, 0), (0, 0))\) the values of \(x\) that satisfy \((x, a) \in \sigma\) are \(0000, 0001, 1110, 1111\) and the corresponding density matrices are \(\rho = \frac{1}{4} \left[ \begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array} \right] \) \(= \frac{1}{16} I\). Therefore, \(O_{((0,0),(0,0))} = \frac{1}{4} \left[ \begin{array}{cccc} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right] \) and thus it holds that \(\|O_{((0,0),(0,0))}\| = \frac{3}{4} = \text{Val}(G)\).

By Lemma 12, it is enough to guarantee that \(c \geq \frac{1}{2} + \delta\) for some constant \(\delta\) in order succeed an exponentially good error tolerance. Thus, the hidden matching \(\left(\frac{1}{4}\right)\) QRG can tolerate up to 12.5% of errors.

B Technical proofs

**Proof of lemma 12.** Let \(G = (S, A, \{\rho_x\}_{x \in S}, \sigma_a, \sigma_b, \sigma)\). We create the following game \(G' = (S', A', \{\rho'_x\}_{x \in S'}, \sigma'_a, \sigma'_b, \sigma')\) by taking the product of \(n\) games \(G\). Then we require that \((x_1 \cdots x_n, a_1 \cdots a_n) \in \sigma'_a\) if at least \(c - \delta\) of the \((x_i, a_i)\) are in \(\sigma_a\) and \((x_1 \cdots x_n, b_1 \cdots b_n) \in \sigma'_b\) if at least \(c - \delta\) of the \((x_i, b_i)\) are in \(\sigma_b\). Furthermore, by definition, it holds that \((x_1 \cdots x_n, a_1 \cdots a_n, b_1 \cdots b_n) \in \sigma'\) if \((x_1 \cdots x_n, a_1 \cdots a_n) \in \sigma'_a\) and \((x_1 \cdots x_n, b_1 \cdots b_n) \in \sigma'_b\).

Since \(\delta > 0\), we have \(c > (1 + \varepsilon)/2\) and hence \(c - \delta > 1/2\). This implies there exist at least \(2c - 2\delta - 1 = \varepsilon + \delta\) common values \((i's\) such that \((x_i, a_i) \in \sigma_a\) and \((x_i, b_i) \in \sigma_b\). Therefore,
Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Lemma 14. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.

Proof of Theorem 15. The correctness of $(\tilde{\gamma})$QRGv and $(\tilde{\gamma})$rQRGv is exactly the same. Furthermore, it is clear that if a non-restricted adversary has probability $\varepsilon$ to win, this probability cannot increase by restricting this adversary. In order to show the equivalence between the two definitions, it remains to show that a non-restricted adversary has no more power than a restricted adversary. Without loss of generality, we assume that the first successful query was for $\sigma_a$. Then, one more success for $\sigma_a$ does not help the adversary towards finding an answer for $\sigma_a$ (since, it already knows one). Furthermore, by the independence property, even knowing all the answers for $\sigma_a$ does not give the adversary extra power to find an answer to $\sigma_b$. Therefore, restricting the adversary to one successful query per relation, does not decrease his winning probability.
b. If \( i = 2 \) then \( F' \) chooses at random one of the \( \sigma_a \) queries and answers it with 1. To all the others, \( F' \) answers with 0. When \( F \) returns its final answers \((a'_1, a'_2)\), \( F' \) returns \((a'_1, a'_2)\).

c. If \( i = 3 \) then \( F' \) chooses at random one of the \( \sigma_b \) queries and answers it with 1. To all the others, \( F' \) answers with 0. When \( F \) returns its final answers \((a'\ast_1, a'\ast_2)\), \( F' \) returns \((a'\ast_1, a'\ast_2)\).

d. If \( i = 4 \) then \( F' \) chooses at random one of the \( \sigma_a \) and one of \( \sigma_b \) questions and answers them with 1. After the second positive answer, \( F' \) stops simulating \( F \) and returns as \((a'\ast_1, a'\ast_2)\) the queries that he answered with 1.

Then the winning probability of \( F' \) can be computed as follows

\[
q = \Pr[ F' \text{ wins} ]
\]

\[
= p_1 \Pr[ F' \text{ wins} | E_1 ] + p_2 \Pr[ F' \text{ wins} | E_2 ] + p_3 \Pr[ F' \text{ wins} | E_3 ] + p_4 \Pr[ F' \text{ wins} | E_4 ]
\]

\[
\leq \Pr[ F' \text{ wins} | E_1 ] + \Pr[ F' \text{ wins} | E_2 ] + \Pr[ F' \text{ wins} | E_3 ] + \Pr[ F' \text{ wins} | E_4 ]
\]

\[
\leq t^2 \cdot \left( \Pr[ F' \text{ wins} | E_1 ] + \Pr[ F' \text{ wins} | E_2 ] + \Pr[ F' \text{ wins} | E_3 ] + \Pr[ F' \text{ wins} | E_4 ] \right)
\]

\[
= 4t^2 \cdot \Pr[ F' \text{ wins} ]
\]

where the third line comes from the fact that \( F' \) has probability \( 1, \frac{1}{t}, \frac{1}{t}, \) and \( \frac{1}{t^2} \) respectively to correctly respond to \( F \)'s queries. Therefore for any polynomial \( p \) it holds that

\[
\frac{p(t)}{4t^2} \cdot \epsilon < \Pr[ F' \text{ wins} ]
\]

and in particular for \( p(t) = 4t^2 \) it holds that \( \epsilon < \Pr[ F' \text{ wins} ] \) which contradicts the fact that \( G' \) is a \((c, \epsilon) - \frac{1}{t}\) QRG. Hence we see that a \((\frac{1}{t})\) QRG implies a \((\frac{1}{t})\) QRGv. ◀