Quantum Money

Shagun Khare

June 2021-August 2021

Affiliation: Summer Quantum Camp at New York University

Advisers: Professor Javad Shabani, Dr. Mohammad Farzaneh, Dr. Neda Lotfizadeh, Eva Gurra

ABSTRACT

Today, we utilize two basic forms of money: the type we carry around and the type we entrust to someone else. The type we carry around includes coins, bank notes, poker chips, and precious metals. These are objects that are made by a mint and can be counterfeited easily by using the same production process as the one used to make the original. The other kind of money is the kind that you entrust to someone else. This is where bank accounts, credit card lines, and checks come into play where you instruct the bank to move money on your behalf. However, this form of money has its own disadvantages: every time you pay someone, you need to tell your bank whom to send money to. This process leaves a paper trail and has high risk if your connection to the bank is not secure. This is where quantum money comes into play. A quantum money scheme is a quantum cryptographic protocol to create and validate banknotes which are impossible to forge. The key idea is that arbitrary quantum states cannot be perfectly copied, otherwise known as the no-cloning theorem. Utilizing this idea, quantum money schemes produce banknotes which are impossible to forge by including quantum systems in their design.

BACKGROUND

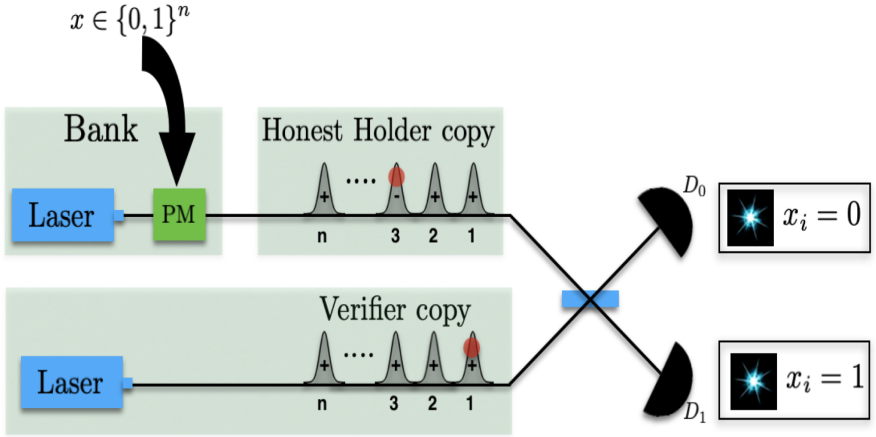
To start out my research process, I examined Stephen Wiesner's money scheme to start understanding how to implement quantum money. In the Wiesner money scheme, a central bank prints "quantum bills," each of which contains a classical serial number as well as a long string of qubits. Then, each qubit is prepared in one of four possible quantum states .

The bank, which acts as a central database, stores the serial number of every bill in circulation and the preparation instructions for each of the bill's qubits. If the sender wants to verify a bill as genuine, they have to bring it back to the bank. The bank, using its secret knowledge of how each qubit was prepared, measures each qubit in the appropriate basis and checks that it gets the expected outcomes. If even one qubit yields the wrong outcome, the bill is rejected as counterfeit.

Now consider the situation of a counterfeiter, who holds a quantum bill but does not have access to the bank's secret database. When the counterfeiter tries to copy the bill, they won't know the right basis in which to measure each qubit. However, they make the wrong mistake, not only do they fail to make an accurate copy, but the measurement goes as far to destroy the original copy as well. Therefore, every time the counterfeiter tries to make an attempt to forge a bill, his chances of correctly forging the bill decrease, which is represented by Wiesner's model as ¾^n, where n is the number of qubits. As the number of qubits increases, the likelihood for the forger succeeding decreases on an exponential level.

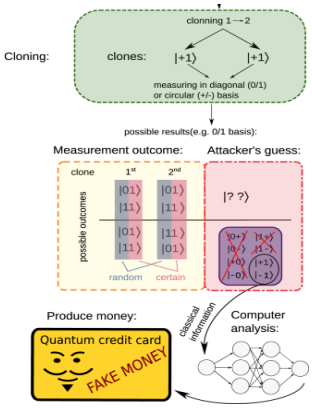
METHODOLOGY

Classical variation was introduced by Wiesner and enhanced by multiple scientists as an impriviation of the private quantum money scheme. In this method, classical communication with a bank is used in order to verify a quantum coin.



Essentially, as represented by the picture above, this demonstrates an example of a verified process. Here, the bank has an Honest Holder copy, which is the secret database it has which contains the verified bill states. The banklooks up the serial number and retrieves the description of the corresponding quantum state. Then the mint verifies the given state is the state that goes with the attached serial number. If the results match, then the bill is verified.

However, in the second example, if the bank detects a forged bill by finding that the bank's classical string does not align with the forged bill's, it detects the forgery and erases the bill's serial number, which doesn't allow for it to be passed along or distributed.



Qiskit Implementation

```
def measure_message(message, bases):
    backend = Aer.get_backend('aer_simulator')
    measurements = []
    for q in range(3):
        if bases[q] == 0: # measuring in Z-basis
            message[q].measure(0,0)
        if bases[q] == 1: # measuring in X-basis
            message[q].h(0)
            message[q].measure(0,0)
        aer_sim = Aer.get_backend('aer_simulator')
        qobj = assemble(message[q], shots=1, memory=True)
        result = aer_sim.run(qobj).result()
        measured_bit = int(result.get_memory()[0])
        measurements.append(measured_bit)
    return measurements
```

```
prob = 0
shots = 100

for i in range(100):
    forge_bits = randint(2, size=3)
    forge_bases = randint(2, size=3)
    forge_note = create_money(forge_bits, forge_bases)
    bank_measure = measure_message(forge_note, bank_bases)
    bank_fixed = measure_message(banknote, bank_fixedbase)
    if(bank_fixed==bank_measure):
        print("Bank Serial Number: " + str(bank_fixed))
        print("Verfied Bill: " + str(bank_measure))
    else:
        print("Bank Serial Number: " + str(bank_fixed))
        print("Forged Bill: " + str(bank_measure))
        prob = prob+1
    print("")

print("Probability of bills being forged:" + str(prob/shots))
```

Therefore, I decided to implement the classical variation quantum money scheme in Qiskit. In this method, the bank has a set serial number which is designed by 3 qubit states. The counterfeiter is randomly generating qubits which are all equal in size, thus representing 3 qubits as well in this example. Then, these quantum coins are measured by the bank in the correct basis through Quantum Key Cyrootpgprahy, which essentially encodes the forger's state and allows the bank to measure either in the x or x basis. Then, the bank compares this classically measured state with its own serial state and determines whether the money is forfeited or not.

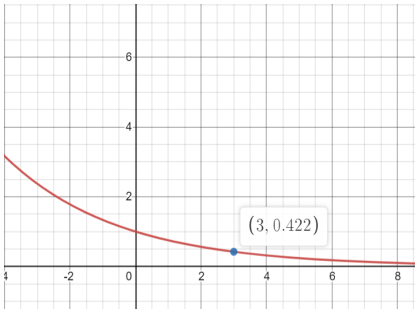
Thus, as represented by this example, the forger has 100 attempts to create counterfeit money. However, as supported by the classical variation example, every time a bill is recognized as counterfeit, the bank erases this measurement from memory and randomly generates a new serial number. As a result, this process results in the likelihood for the froger to accurately counterfeit a bill decreasing on an exponential basis as the number of qubits increases. As represented by this example, the probability represented by Wiesner's scheme is (¾)^n, with n number of qubits equating to 3. As the program detected from experimental probability over theoretical probability, exactly 42 attempts out of the 100 attempts resulted in forgery. This is equivalent to the exponential equation represented by the graph below.

```
Bank Serial Number: [1, 1, 1]
Verfied Bill: [1, 1, 1]
```

```
Bank Serial Number: [1, 1, 1]
Forged Bill: [0, 1, 0]
```

```
Bank Serial Number: [1, 1, 0]
Forged Bill: [0, 1, 1]
```

```
Probability of bills being forged:0.42
```



Enhancing the Qiskit Implementation with the No-Cloning Theorem

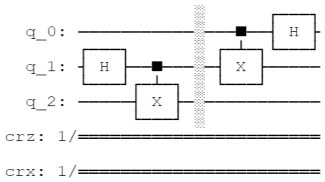
Now that we covered the specific elements that go into the classical variation scheme, how can we protect our bills to ensure that a counterfeiter can't actually make a copy of our bill? This is where the no-cloning theorem is introduced. The no-cloning algorithm states that you cannot create a copy of an arbitrary, or unknown quantum state. Therefore, backup copies of a state in the middle of a quantum computation cannot be created. The no-cloning algorithm is a fundamental part in quantum cryptography as it forbids additional parties from creating copies of a transmitted quantum key, such as banknotes. Lastly, the theorem protects the uncertainty principle in quantum mechanics as it prevents users from measuring each clone of the original message.

Therefore, in this Qiskit implementation, I incorporated the no-cloning theorem into our quantum money scheme. For example, when a coin holder Alice wants to pass her coin to a new coin holder Bob, they run the following protocol:

1. Alice sends to the bank the string s and tells the bank that she wants to pass the coin to Bob.
2. The bank checks that s is a valid secret string, then erases s from the list of valid strings and adds to the list a newly generated secret string s. If a forgery attempt is detected, the bank erases the string and generates a copy.
3. If the bill is verified, the bank then sends the string to Bob and Bob is in sole possession of the coin.

```
def alice_bill(qc, psi, a):
    qc.cx(psi, a)
    qc.h(psi)
```

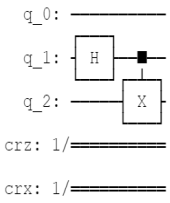
```
bank_circuit.barrier()
alice_bill(bank_circuit, 0, 1)
bank_circuit.draw()
```




```
bill = QuantumRegister(3, name="q")
crz = ClassicalRegister(1, name="crz")
crx = ClassicalRegister(1, name="crx")
bank_circuit = QuantumCircuit(bill, crz, crx)
```

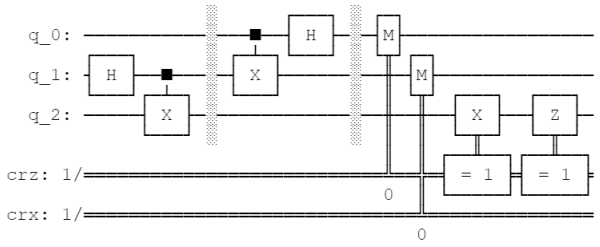
```
def create_bell_pair(qc, a, b):
    qc.h(a)
    qc.cx(a,b)
```

```
create_bell_pair(bank_circuit, 1, 2)
bank_circuit.draw()
```




```
def bank_apply(qc, qubit, crz, crx):
    qc.x(qubit).c_if(crx, 1)
    qc.z(qubit).c_if(crz, 1)
```

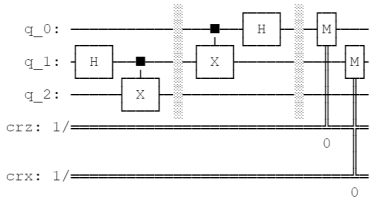
```
bank_apply(bank_circuit, 2, crz, crz)
bank_circuit.draw()
```



10

```
def measure_and_send(qc, a, b):
    """Measures qubits a & b and 'sends' the results to Bob"""
    qc.barrier()
    qc.measure(a,0)
    qc.measure(b,1)
```

```
measure_and_send(bank_circuit, 0 ,1)
bank_circuit.draw()
```



This demonstrates that even with a local verification process, the banknote obtains exponential security as it doesn't allow the forger to create a copy either in the private or public quantum scheme. As we've seen in this transaction, the process is irreversible. So once Alice sends the banknote to Bob after being verified, Alice doesn't have a copy of the money herself. This theorem has been extremely helpful in helping uncover the public-quantum scheme because

it represents the possibility that banks produce notes that are public variables which can't be cloned even with an access to a verifier.

CONCLUSION

Overall, the theory of quantum money is still in its experimental process, with the majority of its theorem only mathematically enhanced right now. According to Wiesner's scheme, many are looking for ways to continue to incorporate Quantum Key Distribution and its advanced algorithms in order to determine any possible attacks and what measures we can take to improve the security of a quantum money scheme.

With that said, the entire process is still in its initial states, where many theorems have been heavily researched, only a few being experimentally proven, and none have been put into any practical implementation of the concept.

Thus, if we were to go back to our ideal money scheme in which you can carry around a large amount of money which is protected, which refers back to our public quantum scheme, this theory has many beneficial prospects that appeal to both government and commercial sectors. For example, a few software companies have already stated their interest in sharing programs that anyone in the public can use, but no one can copy. Hopefully, in the foreseeable future, we can gain a larger understanding of how quantum money can truly be applied in real life transactions.

REFERENCES

1. Aaronson, S. Quantum copy-protection and quantum money. In Annual IEEE Conference on Computational Complexity (2009), 229242.

2. S. Aaronson and P. Christiano. Quantum money from hidden subspaces. In preparation, 2012.

3. A. Lutomirski. An Online Attack Against Wiesner's Quantum Money. http://arxiv.org/abs/1010.0256, 2010.

4. Bennett, C.H., Brassard, G., Breidbart, S. and Wiesner, S. Quantum cryptography, or unforgeable subway tokens. In Advances in CryptologyProceedings of Crypto (1983), volume 82, 267275.

5. Jiráková, K., Bartkiewicz, K. Černoch, A. and Lemr,K. Experimentally attacking quantum money schemes based on quantum retrieval games. (2019)

6. Mosca, M. and Stebila, D. A framework for quantum money. Poster at Quantum Information Processing (QIP) (Brisbane, Australia, 2007).