Quantum Money for Knot Theorists

Ben Edelman

May 6, 2017

1 Introduction

This paper is intended for an audience of people who know something about knot theory but next to nothing about quantum information. Public key quantum money based on knots is a completely unexpected application of knot theory. The first portion of this paper will provide a crash course on the relevant topics in quantum information and quantum computation. Then, I'll introduce the concept of public-key quantum money, a proposed form of money that anyone (with a quantum computer) can verify is genuine, but nobody can forge. We'll then explore the only potentially viable proposal for public-key quantum money [1], which uses knot theory (specifically, the Alexander polynomial).

2 Quantum preliminaries

2.1 Quantum states

Electrons have a property called spin. When we measure the spin of an electron, we can obtain two possible answers: 'up' or 'down'. But when we are not measuring the spin of the electron, the electron can be in a 'superposition' of up and down. This means that, in some sense, the state of the electron is a linear combination of the basis states 'up' and 'down'. We can write the state of the electron's spin as $\alpha_0|\text{up}\rangle + \alpha_1|\text{down}\rangle$ or, using 0 to represent up and 1 to represent down, $\alpha_0|0\rangle + \alpha_1|1\rangle$. We say α_0 is the amplitude of $|0\rangle$ and β_1 is the amplitude of $|1\rangle$. Note that amplitudes are complex numbers (this is how the universe apparently works). If we have two electrons, then there are four basis states: $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. $|01\rangle$, for example, corresponds to the spin of the first electron being up and the spin of the second electron being down. Thus, the state of the two-spin system can be written in the form $\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$. This notation is just another way of writing the 4×1 vector $\begin{bmatrix}\alpha_{00} & \alpha_{01} & \alpha_{10} & \alpha_{11}\end{bmatrix}^T$. For our purposes, it doesn't matter that we are looking at spins of electrons; we could also be looking at another two-level system such as polarities of photons. So from now on, I'll refer to the two-level systems as quantum bits, or *qubits*. A state consisting of n qubits is described with 2^n amplitudes (i.e., a 2^n-dimensional column vector).

2.2 Measurement of quantum states

When we measure any state, we always observe one of the basis states. The Born Rule, for which there is no known proof, states that the probability of measuring a given basis state is equal to the square of the magnitude of the corresponding amplitude. Since probabilities must always add up to 1, this means that the vector of amplitudes always has a Euclidean norm of 1. As an example, suppose we have the two-qubit state $|\psi\rangle = \frac{1}{2}|00\rangle + \frac{i}{2}|01\rangle - \frac{1}{2}|10\rangle - \frac{i}{2}|11\rangle$. Then, let's say we measure the second qubit of $|\psi\rangle$, but not the first qubit. What is the probability we will observe that the second qubit is $|0\rangle$? There are two basis states corresponding to this outcome: $|00\rangle$ and $|10\rangle$. Thus, the probability of measuring $|1\rangle$ is $|\alpha_{00}|^2 + |\alpha_{10}|^2 = |\frac{1}{2}|^2 + |-\frac{1}{2}|^2 = \frac{1}{2}$. Suppose this outcome does occur. Then, to determine the new state after the measurement, we take the portion of the original state that corresponded to the measurement result (in this case $\frac{1}{2}|00\rangle - \frac{1}{2}|10\rangle$), and we normalize the amplitudes so the squares of their magnitudes sum to 1 again, giving us $\frac{1}{\sqrt{2}}|00\rangle - \frac{1}{\sqrt{2}}|10\rangle$.

In fact, our original state $|\psi\rangle$ can be *separated* into two one-qubit states, like this:

$$|\psi\rangle = \tfrac{1}{2}|00\rangle + \tfrac{i}{2}|01\rangle - \tfrac{1}{2}|10\rangle - \tfrac{i}{2}|11\rangle = (\tfrac{1}{\sqrt{2}}|0\rangle - \tfrac{1}{\sqrt{2}}|1\rangle) \otimes (\tfrac{1}{\sqrt{2}}|0\rangle + \tfrac{i}{\sqrt{2}}|1\rangle)$$

The $\otimes$ symbol indicates that we are taking the tensor product of the vector on the left and the vector on the right. With this notation, we can expand tensor products easily using the distributive property. For the rest of this paper, we will omit the $\otimes$ sign in tensor products, so the above will become simply $(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle)(\frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle)$. Because the two qubits in our example are separable, performing a measurement on the second qubit does not affect the first qubit, so without knowing the result of the measurement we can already know that the state of the first qubit will remain $(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle)$.

2.3 Operations on quantum states

Measurement is one very special way to change a quantum state. We can also change a state without measuring it, by applying a *quantum operation*. All quantum operations are linear transformations acting on the state vector. Say we have a quantum operation that acts on an n-qubit state represented by the 2^n-dimensional column vector v. Then the operation can be represented by a $2^n \times 2^n$ matrix; call this U. We need to maintain the condition that $|Uv|_2 = 1$ for every possible v. A matrix that obeys this condition is called a *unitary matrix*. Another equivalent condition for U to be unitary is that its conjugate transpose is its inverse: $U^*U = UU^* = I$. In summary, all quantum operations are unitary linear transformations.

2.4 Quantum gates and computation

In the real world, there isn't an easy way to apply an arbitrary unitary operation to an n-qubit state for large n. But we do know how to implement arbitrary single-qubit unitary operations. Operations that act on only a few qubits are called *quantum gates*, because they are analogous to logical gates like NOT and AND. We also know how to implement a certain two-qubit gate called the controlled NOT gate (or CNOT for short). Since CNOT is a two-qubit operation, it is represented by a 4×4 matrix:

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

In order to understand a quantum gate, we often look at how it acts on basis states. When CNOT is applied to a basis state $|x_0x_1\rangle$, we find that $\text{CNOT}|x_0x_1\rangle = |x_0\rangle|x_0 + x_1\rangle$[1], where addition is mod 2. For example,

$$\text{CNOT}|11\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = |10\rangle$$

It can be proven that, in fact, any unitary operation on n qubits can be approximated to an arbitrary degree of accuracy with just single-qubit gates and CNOTs [2] (where each gate is applied to some subset of the qubits). A quantum computer works by applying these gates (or some other set of universal gates) and measurements to qubits in its memory. The 'running time' of a quantum algorithm is just the number of gates and measurements that occur during the course of the algorithm. Another important result in quantum computing is that such unitary operations and measurements can simulate any classical computation with only a constant factor slowdown [3]. There are some problems, such as factoring, for which there are quantum algorithms that are far faster than the most efficient classical algorithms.

2.5 No-cloning theorem

First, let's extend our notation. Just as $|\eta\rangle$ represents a column vector, we let $\langle\eta|$ represent the row vector that is its transpose. We write $\langle\eta|\chi\rangle$ for the product of the two vectors, which is the inner product of $|\eta\rangle$ with $|\chi\rangle$. Similarly, $|\chi\rangle\langle\eta|$ is the outer product of $|\eta\rangle$ with $|\chi\rangle$. We call $|\rangle$ 'bra' and $\langle|$ 'ket', and the notation is called bra-ket notation. Now we have the tools to prove our first theorem!

Theorem 1. *An arbitrary quantum state cannot be copied. More formally, there does not exist a unitary operation U on $2n$ qubits and an n-qubit state $|e\rangle$ such that for any n-qubit state $|\psi\rangle$, $U(|\psi\rangle|e\rangle) = |\psi\rangle|\psi\rangle$.*[2]

Proof. Suppose for the sake of contradiction that there exists such a U and $|e\rangle$. Let $|\psi\rangle$ and $|\phi\rangle$ be two arbitrary n-qubit states. Then $U(|\psi\rangle|e\rangle) = |\psi\rangle|\psi\rangle$ and $U(|\phi\rangle|e\rangle) = |\phi\rangle|\phi\rangle$. Thus,

$$(\langle\phi|\langle e|)(|\psi\rangle|e\rangle) = \langle\phi|\langle e|U^*U|\psi\rangle|e\rangle = \langle\phi|\langle\phi||\psi\rangle|\psi\rangle = \langle\phi|\psi\rangle^2$$

[1] Remember that $|x_0\rangle|x_0 + x_1\rangle$ means $|x_0\rangle \otimes |x_0 + x_1\rangle$. In fact, the notation $|x_0x_1\rangle$ can be understood as meaning $|x_0\rangle \otimes |x_1\rangle$.

[2] This is a slight simplification of the actual statement of the No-Cloning Theorem, because quantum states are actually invariant when multiplied by a scalar with magnitude 1. Taking into account this phase factor hardly affects the proof.

since U is unitary. But a property of the tensor product yields

$$(\langle\phi|\langle e|)(|\psi\rangle|e\rangle) = \langle\phi|\psi\rangle\langle e|e\rangle = \langle\phi|\psi\rangle$$

Hence, $\langle\phi|\psi\rangle = \langle\phi|\psi\rangle^2$, so $\langle\phi|\psi\rangle$ must be either 0 or 1, meaning $|\psi\rangle$ and $|\phi\rangle$ are either equal or orthogonal. But $|\psi\rangle$ and $|\phi\rangle$ are arbitrary unit vectors, and we could have chosen them so they weren't equal or orthogonal. □

The No-Cloning theorem is an immensely important result in quantum information. It can often be an annoyance. For example, one of the most common operations for a classical computer to perform is to copy data. But quantum computers are incapable of copying arbitrary quantum data because of the No-Cloning Theorem. However, as we are about to see, the No-Cloning theorem also opens up new possibilities.

3 Quantum money

3.1 The idea

Because of the No-Cloning theorem, we know that arbitrary quantum states cannot be copied. In 1969, Stephen Wiesner realized that we also want money to be un-copyable (i.e., unforgeable) [4]. He developed a scheme for *quantum money* that is provably secure, but has the undesirable feature that anyone who wants to check the authenticity of their money must hand it over to the mint that produced it.[3] In 1983, Wiesner and others introduced the concept of *public-key* quantum money in [5].

Each unit of quantum money would be a quantum state. Hypothetically, this could be stored in some physical medium such as a bill, or it could be stored on quantum computers and transmitted over a quantum internet. We want our public-key quantum money scheme to satisfy three properties (taken from [1]):

1. *The mint can produce money efficiently.* There is a polynomial-time quantum algorithm that randomly produces a quantum money state $|\$_p\rangle$ and an associated serial number p.[4]

2. *Anyone with a quantum computer can efficiently verify that a quantum money state was produced by the mint, without destroying the money.* There is a polynomial-time quantum algorithm that outputs "genuine" when given a valid quantum money state $|\$_p\rangle$ and the associated p as input; this algorithm should also output the original money state $|\$\rangle$ if it is valid. The mint would publish a list of all the valid serial numbers.

3. *Given a valid quantum money state, nobody can copy it.* There does not exist a polynomial-time quantum algorithm that takes as input $(|\$_p\rangle, p)$ and outputs two quantum money states such that the verification algorithm accepts both quantum money states (when paired with p) as genuine with more than exponentially small probability.

[3] Wiesner's paper was not accepted into a journal until 1983.

[4] The notation "$|\$\rangle$" is perhaps the best aspect of quantum money.

Several candidate public-key quantum money schemes have been proposed, but only Farhi et al.'s knot-theoretic scheme in [1] has not yet been broken (see Chapter 9 of [6] for a summary of recent efforts towards public-key quantum money).

4 Computing with links

4.1 Alexander polynomial in polynomial time

Given any reasonable finite method for representing a link diagram, the Alexander polynomial can be computed in polynomial time. In Alexander's paper introducing his eponymous polynomial [7], the polynomial was defined through an efficient algorithm:

1. Given a diagram for a link L we first apply some Reidemeister II moves if necessary in order to make sure the diagram is connected. If this diagram has v crossings, then by Euler's formula for polyhedra it will have $v+2$ regions, (including the exterior, if we are working in $\mathbb{R}^2$). We associate a variable r_i with each region.

2. For each crossing in the diagram, make note of the four regions r_j, r_k, r_l, r_m incident to that crossing (in the order determined by the Figure 1), and write down the equation $xr_j - xr_k + r_l + r_m = 0$.

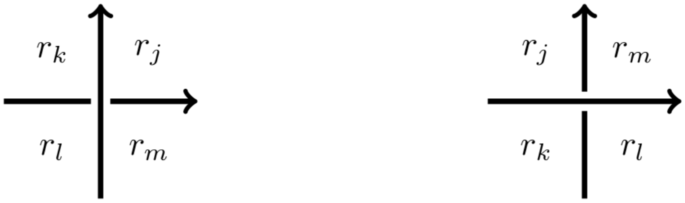


Figure 1: (from [1]) The order of the region variables in the equation associated with a crossing is shown here.

3. Think of the equations as linear equations in the variables $\{r_i\}_{i=1}^{v+2}$, and write out the matrix M corresponding to the system of equations. Delete two columns from M corresponding to any two adjacent regions.

4. Take the determinant of M (this can be calculated in polynomial time in terms of the number of crossings). This is the Alexander polynomial $\Delta(x)$. (Later on, when I mention the Alexander polynomial, I really mean the list of coefficients, ordered from coefficient of highest power of x to lowest power).

Of course, the poly-time classical algorithm can be converted into a poly-time quantum algorithm, by [3].

4.2 Planar grid diagrams

We need a method for finitely representing links. Farhi et al. use *planar grid diagrams*, introduced by Cromwell [8] as "loops and lines" diagrams, to represent links.[5]

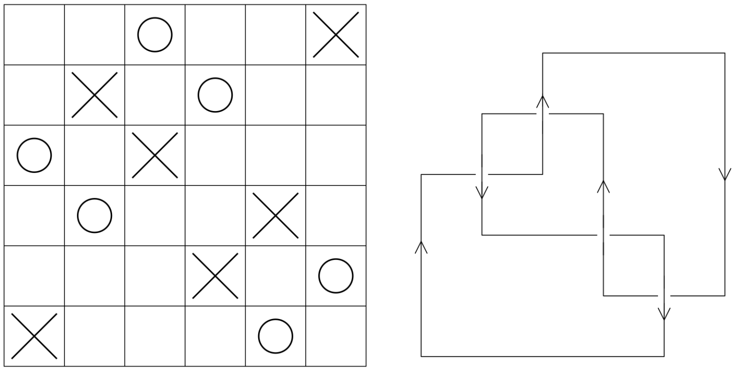
A planar grid diagram is a link diagram in the plane with the following properties:

1. The diagram is made up of only horizontal and vertical line segments.
2. At each crossing, the vertical segment passes over the horizontal segment.
3. No two segments are collinear.

Theorem 2. *Any link can be represented as a grid diagram.*

Proof. (rough sketch) Given any link diagram for a link L, we can assume without loss of generality that the diagram is made up of a finite number of (possibly not horizontal or vertical) segments. We can satisfy property (1) by decomposing each segment into a finite number of horizontal and vertical segments by 'zig-zagging', making the segments small enough so the segments only intersect at the original crossings. We can satisfy property (2) by adding a 90° twist at each crossing for which the horizontal segment passes under the vertical segment. We can satisfy property (3) by slightly shifting all the collinear horizontal segments away from each other, and then doing the same for collinear vertical segments. □

Since we will be dealing with oriented links, we can label each vertex with an X or an O such that vertical segments go from X to O and horizontal segments go from O to X. Each vertex is incident to one horizontal segment and one vertical segment, and each segment is incident on two vertices, so if d is the number of X vertices, then there are also d vertices labeled O, d horizontal segments, and d vertical segments.[6] Thus, without loss of generality, the vertices can be located in an $d \times d$ grid. See Figure 2 for an example.



		O			X
	X		O		
O		X			
	O			X	
			X		O
X				O	

Figure 2: (from [9]) A grid diagram for the figure eight knot.

[5] Grid diagrams (which are based on and equivalent to the earlier-developed *arc presentations*) have seen recent use in the theory of knot and link Floer homology, as in [9].

[6] The minimal grid dimension d of any grid diagram representing a link is an invariant called the *arc index*.

There will be one X and one O and each row and column. Thus, we can write down the grid diagram as (π_X, π_O), where π_X is the permutation on d elements describing the positions of the X's and π_O is the permutation describing the positions of the O's, and the two permutations are disjoint.

Cromwell, in [8], described three moves on grid diagrams, that, like the Reidemeister moves on link diagrams, can be used to relate any two equivalent links. The action of these three moves on any grid diagram are easy and efficient to compute.[7]

4.3 Hardness of determining whether links are equivalent

The quantum money scheme of Farhi et al. relies upon the assumption that it is hard to tell whether two links are equivalent, even in the average case for a quantum computer. This is called the *recognition problem*. There is no known polynomial-time algorithm to solve even the *unknot recognition* problem (is a given knot equivalent to the unknot), which is a special case. The best-known algorithms for both problems have exponential running-time [11].

5 Farhi et al.'s quantum money scheme

5.1 How the mint produces money

The mint begins by producing a superposition over all pairs of permutations on d elements, for $d \leq D$ (where D is very large).

$$\sum_{d=2}^{D} \left(\frac{1}{d!} \sum_{\pi_X \in S_d} |\pi_X\rangle \sum_{\pi_O \in S_d} |\pi_O\rangle \right)$$

Then, the mint measures whether the two permutations are disjoint. If they are not disjoint, the mint restarts the process. If they are disjoint (which occurs with probability $\approx 1/e$), then (as long as the measurement process doesn't cause the mint to glean extra information about the permutations), the mint will have a superposition over valid grid diagrams of dimension at most D:

$$\frac{1}{\sqrt{N}} \sum_{\text{grid diagrams } G} |G\rangle$$

where N is the number of such diagrams.[8] Then, the mint (quantum) computes the Alexander polynomial of the register containing the superposition of diagrams, and measures the result. Since the mint is only measuring the Alexander polynomial, it will obtain a superposition over all diagrams with a specific (random) Alexander polynomial:

$$|\$_p\rangle = \frac{1}{\sqrt{N'}} \sum_{G | \Delta_G(x) = p} |G\rangle$$

[7] See, for example, Gridlink, a software program that can calculate grid diagram moves.

[8] In Farhi et al.'s actual minting algorithm, each state in the sum has an amplitude depending on d, with the amplitudes approximating a normal distribution in terms of d. This is in order to prevent a certain attack on the scheme which I will gloss over; for exposition purposes, a uniform superposition will suffice.

where p is a randomly chosen Alexander polynomial, and N' is the number of diagrams with dimension $\leq D$ and Alexander polynomial p. The banknote is then $(|\$_p\rangle, p)$.

5.2 How to verify the money

Given a supposed banknote $(|\psi\rangle, p)$, the verifier needs to check whether $|\psi\rangle$ is in fact the proper superposition over all diagrams with Alexander polynomial p:

1. Let A be a quantum algorithm that checks whether its input encodes a valid grid diagram. The verifier measures whether A accepts the input $|\psi\rangle$. If this test fails, then the money is invalid. Otherwise, move on to Step 2. (If the test succeeds, then the original state $|\psi\rangle$ may have included some invalid diagrams in the superposition, but the new state $|\psi'\rangle$ will only include valid diagrams.)

2. The verifier computes the Alexander polynomial on input $|\psi'\rangle$, and measures the result. If the result is p, move on to Step 2 (the new state is some $|\psi''\rangle$ that is a superposition over diagrams with Alexander polynomial p). Otherwise, the money is invalid.

3. Now, the verifier wants to check that the superposition over diagrams with Alexander polynomial p is the right superposition. For the purposes of this exposition, we are supposing the superposition needs to be uniform.

   I mentioned earlier that there are three types of moves on grid diagrams under which links are invariant, and that any diagram for a link can map to any other other diagram for the same link using these moves.[9] Moreover, like the Reidemeister moves, each one of these moves has an inverse move. We can consider the Markov chain defined on grid diagrams with the update rule "apply a random valid move". For each link L, the set of all diagrams representing L is a stationary distribution for this Markov chain (this is because of the invariance and inverse properties of the moves). Consequently, for any set of links $\mathcal{L}$, the set of all diagrams representing some link $L \in \mathcal{L}$ is also a stationary distribution. Notably, if we take $\mathcal{L}$ to be the set of links with Alexander polynomial p, then we see that the diagrams in a valid quantum money superposition $|\$_p\rangle$ form a stationary distribution. We use this fact in this third verification step:

   (a) Let $\mathcal{S}$ be the set of possible moves of the three types on grid diagrams of dimension $\leq D$. Then, for each move $s \in \mathcal{S}$, we can write down a permutation matrix P_s that encodes the action of s on all grid diagrams of dimension $\leq D$ (when s is an invalid move for a specific grid diagram, we say it doesn't change the diagram). Let

   $$V = \sum_{s \in \mathcal{S}} P_s \otimes |s\rangle\langle s|$$

   where the second register has basis states $|s\rangle$ for $s \in \mathcal{S}$. V is unitary (this follows from the fact that any permutation matrix is unitary).

[9] See pages 8-11 of Farhi et. al. [1] for a description of the moves.

(b) Given $|\psi''\rangle$, let

$$|\phi\rangle = |\psi''\rangle \frac{1}{\sqrt{|\mathcal{S}|}} \sum_{s\in\mathcal{S}} |s\rangle$$

(c) Apply V to $|\phi\rangle$. The action of V on $|\phi\rangle$ can be computed by a quantum algorithm that applies the random grid move s found in the second register to the diagram in the first register.

$$\begin{aligned} V|\phi\rangle &= \left(\sum_{s\in\mathcal{S}} P_s \otimes |s\rangle\langle s|\right)\left(|\psi''\rangle \frac{1}{\sqrt{|\mathcal{S}|}} \sum_{s\in\mathcal{S}} |s\rangle\right) \\ &= \sum_{s\in\mathcal{S}} \frac{1}{\sqrt{\mathcal{S}}} \left(P_s|\psi''\rangle\right) \otimes |s\rangle \end{aligned}$$

If $|\psi''\rangle$ is a valid money state, then $P_s|\psi''\rangle = |\psi''\rangle$, so $V|\phi\rangle = |\phi\rangle$, and the second register will be separable from the first, with the value $\frac{1}{\sqrt{|\mathcal{S}|}} \sum_{s\in\mathcal{S}} |s\rangle$.

Measure whether the third register of $V|\phi\rangle$ contains a $+1$ eigenvector of

$$\sum_{s,s'\in\mathcal{S}} \frac{1}{|\mathcal{S}|} |s\rangle\langle s'|$$

We are allowed to do this because, in general, quantum measurements ask the question: "Is the input state a $+1$ or -1 eigenvector of a given Hermitian matrix?" (if it is a superposition of such vectors in the eigenspace, then the answer will be probabilistic). If the result is no, then the money state is invalid. If the result is yes, then repeat step (c) with the current state of the two registers instead of $|\phi\rangle$ (once this step has been passed some $r = \text{poly}(D)$ times, the verification algorithm is complete).

6 Conclusion

6.1 Is the scheme secure?

Suppose we want to forge a banknote. How succesful can we be?

We can pass verification step (1) if more than a negligible portion of our quantum money state superposition consists of valid grid diagrams. In fact, we might as well perform this verification step ourselves, so that the superposition is entirely over valid grid diagrams.

We can pass verification step (2) if we can construct a superposition over grid diagrams with Alexander polynomial p. Since the mint publishes a list of valid serial numbers p, we can only choose one of the p's in the list.

We can pass verification step (3) if the diagrams in the superposition of our forged state form a stationary distribution for the Markov chain described above (or a distribution that is very close to stationary if the Markov chain doesn't mix well, because step (3) isn't performed an infinite number of times). If we start from any single grid diagram for a link L, the Markov chain will mix into a superposition over all grid diagrams representing L, so it would seem that if the Markov chain mixes quickly, the forged money state must be a superposition over all grid diagrams representing the links in some set of links $\mathcal{L}$, all of which have Alexander polynomial p. It is not know whether this attack can be performed. Suppose, for example, we wanted to measure a valid money state $|\$_p\rangle$ in order to find one diagram of a link L with Alexander polynomial p, and we wanted to be able to generate the superposition of all diagrams representing L. If we could do this, then we could also solve the recognition problem: given two diagrams, generate the superposition of all diagrams equivalent to the first diagram, and the superposition of all diagrams equivalent to the second diagram; then check whether the superpositions are the same.

In short, the scheme may or may not be actually secure.

6.2 Why use links?

Why is it that links, and not some other mathematical object such as graphs, are used in this scheme?

The answer is that links have some desirable properties:

- There are many distinct links, and many link diagrams encoding each link.
- There is a link invariant (the Alexander polynomial) that is both fine-grained and easy to compute.

  This is why we do not use, for example the Jones polynomial, for which there is evidence that it is hard to compute, even by quantum computers and even on average [12].
- It might be very difficult to tell whether two random link diagrams represent the same link.

  This is why we do not use graphs (with equivalence being under isomorphism and moves being permutations): while there is no known worst-case polynomial time algorithm that determines whether two graphs are isomorphic, the graph isomorphism problem is solveable efficiently on most random graphs in practice.

References

[1] Edward Farhi, David Gosset, Avinatan Hassidim, Andrew Lutomirski, and Peter Shor, "Quantum money from knots" *Proceedings of the 3rd Innovations in Theoretical Computer Science*, pp. 276-289. ACM, 2012.

[2] Tycho Sleator and Harald Weinfurter, "Realizable universal quantum gates", *Physical Review Letters* 74, no. 20 (1995): 4087-4090.

[3] Ethan Bernstein and Umesh Vazirani, "Quantum complexity theory" *SIAM Journal on Computing* 26, no. 5 (1997): 1411-1473

[4] Stephen Weisner, "Conjugate coding" *ACM Sigact News* 15, no. 1 (1983): 78-88

[5] Charles H. Bennett, Gilles Brassard, Seth Breidbart, and Stephen Wiesner "Quantum cryptography, or unforgeable subway tokens" *Advances in Cryptology*, pp. 267-275. Springer US, 1983.

[6] Scott Aaronson, "The Complexity of Quantum States and Transformations: From Quantum Money to Black Holes." arXiv:1607.05256 (2016).

[7] James W. Alexander, "Topological invariants of knots and links." *Transactions of the American Mathematical Society* 30, no. 2 (1928): 275-306.

[8] Peter R. Cromwell, "Embedding knots and links in an open book I: Basic properties." *Topology and its Applications* 64, no. 1 (1995): 37-58.

[9] Ciprian Manolescu, Peter Ozsváth, Zoltán Szabó, and Dylan P. Thurston, "On combinatorial link Floer homology." *Geometry & Topology*, no. 4 (2007): 2339-2412.

[10] Marc Culler, Gridlink, www.math.uic.edu/~culler/gridlink/.

[11] Hoste, Jim, "The enumeration and classification of knots and links." *Handbook of knot theory* (2005): 209.

[12] Dorit Aharonav, Vaughan Jones, and Zeph Landau, "A polynomial quantum algorithm for approximating the Jones polynomial." *Algorithmica* 55, no. 3 (2009): 395-421.