Quantum Walks and Application to Quantum Money

Ali Mousavi

April 2025

Contents

Abstract

This thesis explores the foundations of quantum computation, focusing on quantum walks and their application to quantum money. Quantum walks, particularly continuous-time quantum walks based on group actions, serve as a powerful computational tool with applications in search algorithms and cryptographic protocols. We examine their mathematical structure and their advantages over classical random walks, emphasizing their efficiency in state evolution and probability distribution spreading. As a part of this work, we examine efficient implementations of transforms such as the Quantum Fourier Transform (QFT) and the Quantum Hartley Transform (QHT), analyzing their role in encoding quantum states for secure cryptographic applications. In particular, we discuss a novel instantiation of a quantum money scheme based on QHT, leveraging its unique properties for improved security and efficiency. To ensure the robustness of this quantum money scheme, we develop a verification mechanism utilizing quantum walks. Unlike previous approaches, which rely on standard quantum state measurements, our method employs continuous-time quantum walks to authenticate quantum money, preventing counterfeiting while maintaining computational feasibility. Additionally, we present a detailed discussion on the efficient implementation of this scheme, including optimized circuit designs and error mitigation strategies.

Chapter 1

Introduction

Quantum money, first introduced in a seminal paper by Wiesner [29], is a form of currency represented by quantum states. Unlike classical money, an ideal quantum bill cannot be counterfeited due to the no-cloning theorem of quantum mechanics, which prohibits making an identical copy of an unknown quantum state. Wiesner's original scheme (now termed a *private-key quantum money* scheme) had significant practical drawbacks: it required the issuing bank's secret key to verify each banknote, meaning the bank had to be involved in every transaction. This reliance on the bank for verification severely limits the usability of private-key quantum money schemes.

In 2009, Aaronson [1] proposed the first *public-key quantum money* scheme, in which anyone can verify a quantum banknote while only the bank can mint (issue) new valid banknotes. Public-key quantum money removes the transactional bottleneck of involving the bank every time. Aaronson's specific scheme was later broken by Lutomirski et al. [24]. In the years since, several alternative public-key quantum money constructions have been explored [2, 17, 30, 19, 20, 23, 31]. However, each of these proposals has either been broken by further cryptanalysis [14, 26, 7, 23] or relies on unconventional cryptographic assumptions, making their security or practicality questionable.

Quantum Money from Group Actions and the Fourier Transform. A promising candidate for secure public-key quantum money based on more standard assumptions was recently proposed by Zhandry [31]. In Zhandry's scheme, the core idea is to use an abelian group action as the foundation for the quantum state. Each money state (quantum banknote) is prepared as a *group-action Fourier state*, and the corresponding serial number is an element of the underlying group (we will elaborate on these terms in later chapters). The verification algorithm in this scheme uses a group-action phase kickback routine to extract the serial number from the quantum banknote and check its validity. Doliskani [15] later proved that Zhandry's scheme is secure in the generic group action model, providing evidence for its security under well-defined assumptions.

This Work – Motivation and Contributions. In this thesis, we adapt Zhandry's group-action quantum money scheme by replacing its use of the Quantum Fourier Transform with the *Quantum Hartley Transform* (QHT) over finite abelian groups. The motivation behind this substitution is multifold. First, using the Hartley transform causes the banknotes to have *real* amplitudes instead of complex amplitudes. We believe this shift from complex to real quantum states may offer both computational and theoretical advantages. For example, certain mathematical identities hold for any two real orthonormal bases that do not generally hold for complex bases; such differences can affect the properties of quantum states and were a barrier in prior analyses (indeed, an attempted "quantum lightning" construction in [31] failed due to properties of complex phases that do not arise with real amplitudes). Beyond this specific context, our work is a first step toward employing real-valued quantum transforms in quantum cryptography and quantum algorithms. To the best of our knowledge, this is the first significant quantum cryptographic construction that makes essential use of the quantum Hartley transform. We hope that exploring real-amplitude quantum states will inspire further research into new quantum algorithms and optimizations for real-valued transforms.

We now summarize our main contributions:

- We propose a new **public-key quantum money scheme** based on abelian group actions and the quantum Hartley transform. This scheme is an adaptation of Zhandry's group-action money scheme, modified to output real-amplitude quantum states.
- When using the Hartley transform instead of the Fourier transform, the original verification method from Zhandry's scheme is no longer straightforward and requires the use of twists to function correctly. Instead, we offer a more direct verification approach based on quantum walks.
- We show that the serial number associated with a money state (which is a hidden group element) can be efficiently computed from the quantum state. In particular, we develop a **continuous-time quantum walk algorithm** to extract the serial number of a given banknote state. This algorithm leverages the structure of the group action and the spectral properties of an associated Cayley graph. Our method demonstrates that one can recover the hidden group element (serial) with high success probability, which not only underpins the new verification technique but is also of independent interest.
- In the course of our construction, we introduce a new algorithm for efficiently implementing the **quantum Hartley transform** and related real transforms. We present a recursive technique that exploits the structure of the Hartley transform, yielding a lower gate complexity than prior approaches in the literature.

In summary, our work combines ideas from quantum cryptography (public-key quantum money), quantum walks, and quantum signal processing (Hartley and related transforms) to create a novel and potentially practical quantum money scheme.

Thesis Organization. The remainder of this thesis is organized as follows. In **Chapter 2**, we provide background on the basics of quantum computation and the theory of group actions, including the concept of cryptographic group actions which underlie our money scheme. **Chapter 3** covers quantum walks, both continuous-time and discrete-time, outlining their differences and importance in quantum algorithms. In **Chapter 4**, we apply the quantum walk framework to group actions: we describe how a continuous-time quantum walk on a group action can be simulated efficiently, a crucial step for our serial number extraction algorithm. **Chapter 5** details the quantum money scheme itself and the role of the Hartley transform in its construction. We explain how the scheme is formulated and discuss the advantages and challenges introduced by using the Hartley transform. Finally, **Chapter 6** presents the verification algorithm for the Hartley-based quantum money. This chapter shows how we use quantum walks (as developed in Chapter 4) to reliably verify banknotes and compute their serial numbers, thereby addressing the verification issues that arose from the use of real amplitudes.

Chapter 2

Background of Quantum Computation and Group Actions

2.1 Quantum Computation Basics

Quantum computation operates on information stored in quantum states. The fundamental unit of quantum information is the *qubit*, which, unlike a classical bit, can exist in a superposition of basis states. A qubit is described by a state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ (in Dirac notation), where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$. A system of n qubits resides in a 2^n-dimensional Hilbert space and its state can be a superposition of all 2^n basis states $|b_1 b_2 \cdots b_n\rangle$.

Computation is carried out by unitary transformations on these states. A quantum algorithm is typically a sequence of quantum gates (unitary operators) applied to an initial state (usually a simple basis state like $|00\cdots 0\rangle$), followed by a measurement of some qubits to obtain a classical result. A crucial aspect of quantum computation is that certain transformations—such as the Fourier transform over an N-element group—can be implemented very efficiently on a quantum computer (in $\mathrm{poly}(\log N)$ time), whereas their classical counterparts might require $\mathrm{poly}(N)$ time.

2.1.1 The Quantum Fourier Transform

One example important to this work is the **quantum Fourier transform** (QFT). For an abelian group G of order N, the QFT is the unitary map

$$|g\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{h \in G} \chi_h(g)\, |h\rangle,$$

where $\{\chi_h\}$ are the characters (complex exponential homomorphisms) of G. When $G = \mathbb{Z}_N$ (the integers mod N), the QFT takes $|j\rangle$ to $\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i jk/N} |k\rangle$.

Let $G \cong \mathbb{Z}_{N_1} \oplus \cdots \oplus \mathbb{Z}_{N_k}$ be a decomposition of G. The character $\chi(a, x)$ is then given by

$$\chi(a, x) = \omega_{N_1}^{a_1 x_1} \cdots \omega_{N_k}^{a_k x_k}, \quad \text{where } \omega_M = \exp(2\pi i/M).$$

The Fourier transform of a function $f : G \to \mathbb{C}$ is given by

$$\hat{f}(a) = \frac{1}{\sqrt{|G|}} \sum_{x \in G} \chi(a, x) f(x),$$

and the QFT of a quantum state $\sum_{g\in G} f(g)\,|g\rangle$ is $\sum_{x\in G} \hat{f}(x)\,|x\rangle$.

Fourier Basis and Group Actions. For a regular group action $(G, X, *)$, define

$$\left|S^{(h)} * y\right\rangle = \frac{1}{\sqrt{|S|}} \sum_{g\in S} \chi(g,h)\,|g * y\rangle\,.$$

For fixed $x \in X$, we obtain two orthonormal bases of $\mathbb{C}^X$: the standard basis $\{|x\rangle : x \in X\}$, and the Fourier basis

$$\left|G^{(h)} * x\right\rangle = \frac{1}{\sqrt{|G|}} \sum_{g\in G} \chi(g,h)\,|g * x\rangle\,.$$

These states are eigenstates of the action unitary $U_k : |y\rangle \mapsto |k * y\rangle$:

$$U_k \left|G^{(h)} * x\right\rangle = \chi(-k,h) \left|G^{(h)} * x\right\rangle.$$

The cmpIndex Algorithm. Given a state $\left|G^{(h)} * x\right\rangle$, one can recover h using phase kickback. Starting from $\left|G^{(h)} * x\right\rangle|0\rangle$, apply:

- QFT to second register,
- Controlled unitary $\sum_{k\in G} U_k \otimes |k\rangle\langle k|$,
- Inverse QFT to second register.

This yields $\left|G^{(h)} * x\right\rangle|h\rangle$, extracting h efficiently.

2.1.2 The Quantum Hartley Transform

Another concept we use is the **quantum Hartley transform** (QHT), which replaces complex exponentials with real sinusoidal kernels.

Definition. For $G = \mathbb{Z}_N$, define

$$\mathrm{QHT}_N : |a\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} cas\left(\frac{2\pi a y}{N}\right)|y\rangle\,, \quad \text{where } cas(x) = \cos(x) + \sin(x).$$

For general abelian $G = \mathbb{Z}_{N_1} \oplus \cdots \oplus \mathbb{Z}_{N_k}$, the Hartley transform of $f : G \to \mathbb{R}$ is

$$\mathrm{QHT}_G(f)(a) = \frac{1}{\sqrt{|G|}} \sum_{y\in G} cas(2\pi\langle a, \alpha(y)\rangle) f(y), \tag{2.1}$$

where $\alpha(y) = y_1/N_1 + \cdots + y_k/N_k$.

Unitary Equivalence. Using the identity

$$cas(2\pi\langle a, \alpha(y)\rangle) = \frac{1-i}{2}\chi(a,y) + \frac{1+i}{2}\chi(-a,y),$$

we obtain:

$$\mathrm{QHT}_G = \frac{1-i}{2}\,\mathrm{QFT}_G + \frac{1+i}{2}\,\mathrm{QFT}_G^*, \tag{2.2}$$

showing that $\mathrm{QHT}_G^2 = 1$ and hence the QHT is unitary.

Quantum Formulation. The quantum Hartley transform over G maps:

$$\sum_{x\in G} f(x)\,|x\rangle \mapsto \sum_{a\in G} \mathrm{QHT}_G(f)(a)\,|a\rangle\,.$$

Because the cas function is real-valued, the resulting amplitudes remain real—a fact that plays a crucial role in real-amplitude quantum protocols.

2.2 Group Actions in Cryptography

At the heart of our quantum money scheme is the notion of a group action. A **group action** of a group G on a set X is a function $*\colon G \times X \to X$ that satisfies two properties: (1) the identity element $e \in G$ acts trivially on every $x \in X$ ($e * x = x$), and (2) the action is compatible with the group operation, meaning $g * (h * x) = (gh) * x$ for all $g, h \in G$ and $x \in X$. We will often denote the result of g acting on x by $g * x$ or simply gx when the action is clear from context.

The orbit of an element $x \in X$ under the action is the set $G * x = \{g * x : g \in G\} \subseteq X$. If there is exactly one orbit (i.e., $G * x = X$ for some x), the action is called *transitive*. If, in addition, no non-identity group element fixes any x (i.e., $g * x = x$ implies $g = e$), then the action is *free*. A group action that is both transitive and free is sometimes called a **regular action**. In a regular action, for any two elements $y, z \in X$, there is a unique group element g such that $g * y = z$. In other words, G acts like a permutation group that moves elements of X around, and knowing the starting and ending point uniquely determines the "move" g. In this case $|X| = |G|$, and we can think of X as essentially a copy of G with the group structure "hidden" by the action.

Group actions are ubiquitous in modern cryptography. A **cryptographic group action** is one where the action is easy to compute in the forward direction, but inverting the action (recovering g from x and $g * x$) is computationally hard. This essentially functions as a one-way function: given x and g, it is easy to compute $y = g * x$, but given x and y it is infeasible to find g (except with negligible probability). One classical example is the discrete logarithm problem: consider the cyclic group $G = \mathbb{Z}_p^*$ (the multiplicative group of a prime field) acting on itself by exponentiation. Let $X = G$ and define $g * x = x^g \pmod{p}$ (treating group elements as integers). Here g is an exponent. This is a group action (in fact, just the group's own operation written differently), and computing $x^g \bmod p$ is efficient. However, given x and $y = x^g$, finding g requires solving a discrete log problem, which is believed to be hard for suitable choices of p. Thus, exponentiation can be viewed as a one-way group action.

Another prominent example comes from elliptic-curve cryptography: the action of an ideal class group on a set of elliptic curves. In schemes like CSIDH (Commutative Supersingular Isogeny Diffie–Hellman), an abelian group of ideal classes G acts on the set X of supersingular elliptic curves by isogeny (an isogeny is a structure-preserving map between curves). Starting from a curve x_0, one can efficiently compute $g * x_0$ (which is another elliptic curve) for any group element g, but given two curves x_0 and $x = g * x_0$, it is believed to be computationally hard to recover g. This hard problem is the foundation of isogeny-based cryptography. We refer to such G and X as a *cryptographic group action* pair.

Cryptographic group actions provide a natural platform for public-key quantum money schemes. In such schemes, the secret "serial number" of a quantum banknote can be an element $g \in G$, and the quantum state of the banknote can be some function of $g * x_0$ for a public base point $x_0 \in X$. The hardness of inverting the action (x_0 and $x = g * x_0$ do not easily reveal g) contributes to the security against counterfeiting, while the structure of the group action can be used to efficiently verify authenticity (as we will see with the Fourier or Hartley transform techniques).

We will assume throughout that we have a family of abelian group actions (G, X) that are cryptographically suitable: $|G|$ is large (growing with the security parameter), the action is efficiently computable, transitive and free (so $|X| = |G|$), and inverting the action is infeasible without the secret key. Such families are conjectured to exist (for example, the isogeny-based actions, and others discussed in cryptographic literature).

Finally, we note that working in the **generic group model** (or generic group *action* model) is a common theoretical approach to analyze security. In a generic model, the group elements are treated as black-box labels without exploiting any special algebraic structure beyond group axioms. Doliskani's proof of security for Zhandry's scheme [15] was in such a generic model for group actions, lending confidence that no "generic" attack can break the scheme. In this thesis, we focus on designing the scheme and algorithms at a high level and assume the underlying group action is secure; a detailed security proof is beyond our scope, though we rely on prior results for reassurance.

Chapter 3

Quantum Walks – Continuous and Discrete

Quantum walks are the quantum-mechanical counterparts of classical random walks on graphs. Instead of a probability distribution over vertices, a quantum walk evolves a superposition of vertex states. Interference between multiple paths leads to markedly different behavior from classical walks, often yielding faster spreading and algorithmic speedups [?].

In particular, both major models of quantum walks — continuous-time and discrete-time — achieve *ballistic spreading* (standard deviation growing linearly in time) in contrast to the diffusive $O(\sqrt{t})$ spread of classical random walks [?]. Quantum walks have become a central framework in quantum computing, underpinning various algorithms and even universal quantum computation [?].

In this chapter, we introduce continuous- and discrete-time quantum walks, compare their properties, and discuss their implementation on highly symmetric graphs known as Cayley graphs. We then explore how group actions and symmetry can be leveraged to analyze and construct quantum walks on these graphs.

3.1 Continuous-Time Quantum Walk

A **continuous-time quantum walk (CTQW)** is defined by a time-dependent Schrödinger equation on the vertices of a graph. Given a graph $G = (V, E)$ with $|V| = N$ vertices, one assigns a basis state $|j\rangle$ to each vertex $j \in V$. The walk is governed by a Hermitian Hamiltonian H that respects the graph's structure. A common choice is to take H proportional to either the adjacency matrix A of G or the graph Laplacian L. For example, one may set

$$H = \gamma A \quad \text{or} \quad H = \gamma L,$$

where γ is a constant coupling rate with dimensions of $(\text{time})^{-1}$.

The Hamiltonian connects adjacent vertices: $\langle j| H |k\rangle \neq 0$ if and only if $(j, k) \in E$. The state of the walk $|\psi(t)\rangle = \sum_{j\in V} \psi_j(t) |j\rangle$ evolves according to the Schrödinger equation:

$$i\frac{d}{dt} |\psi(t)\rangle = H |\psi(t)\rangle ,$$

with formal solution

$$|\psi(t)\rangle = e^{-iHt} |\psi(0)\rangle .$$

In coordinates, if we use the Laplacian L defined by $L_{jk} = -\deg(j)\delta_{jk} + A_{jk}$, the equation becomes

$$i\frac{d}{dt}\psi_j(t) = \sum_{k\in V} L_{jk}\psi_k(t),$$

which mirrors the equation of a classical continuous-time random walk except for the factor i. Since L (and A) are Hermitian, the quantum dynamics are unitary and conserve total probability (i.e., the norm of the state vector), in contrast to the irreversible nature of classical diffusion.

The continuous-time quantum walk model was first formalized by Farhi and Gutmann in 1998 as a quantum analogue of continuous Markov processes.

3.1.1 Hamiltonian Choices

One can equivalently define a CTQW using $H = A$ (the adjacency matrix) instead of the Laplacian. For an undirected d-regular graph, $L = dI - A$, and using $H = A$ introduces only an overall phase factor e^{-idt} in the evolution, since dI contributes a global phase. Thus, on regular graphs, the choice of Laplacian versus adjacency matrix is unimportant up to a global phase. In general, the adjacency matrix is often used because it directly encodes graph connectivity. Farhi and Gutmann's original construction corresponds to $H = A$.

Throughout this thesis, we primarily consider $H = A$ for continuous-time walks on unweighted graphs.

3.1.2 Example – CTQW on a Hypercube

The n-dimensional Boolean hypercube Q_n is the graph on 2^n vertices $\{0,1\}^n$, where edges connect vertices differing in one bit. This is a $d = n$ regular graph, and can be viewed as the Cayley graph of $(\mathbb{Z}_2)^n$ with the generating set consisting of flipping each bit.

The adjacency matrix of Q_n can be written as a sum of n commuting terms, each acting on one bit:

$$A_{Q_n} = \sum_{j=1}^{n} \sigma_x^{(j)},$$

where $\sigma_x^{(j)}$ is the Pauli X operator acting on the j-th bit (and identity on the others). Because all $\sigma_x^{(j)}$ commute, the time-evolution operator factorizes:

$$e^{-iA_{Q_n}t} = \prod_{j=1}^{n} e^{-i\sigma_x^{(j)}t} = \bigotimes_{j=1}^{n}\left(\cos t\, I - i\sin t\,\sigma_x^{(j)}\right),$$

yielding an analytical solution for the walk. In particular, starting from the state $|0^n\rangle$, the walker's state remains an even superposition over all vertices at Hamming distance k at time t, with amplitudes given by the k-th term in the expansion of $(\cos t - i\sin t)^n$.

This shows that the quantum walk spreads across the hypercube exponentially faster than a classical random walk, which requires $O(n\log n)$ steps to approach a uniform distribution. The hypercube example illustrates how symmetry (in this case, the direct product structure) enables explicit solutions.

3.1.3 Algorithmic Applications

Continuous-time quantum walks have been employed to achieve algorithmic speedups. A famous example is the *glued trees traversal problem* by Childs et al. (2003), where a CTQW on a graph formed by two randomly "glued" binary trees can find the unique exit vertex exponentially faster than any classical algorithm. This result gave the first exponential separation between quantum and classical query complexity in a black-box setting.

Another application is *spatial search* on graphs. Childs and Goldstone (2004) showed that a CTQW can find a marked vertex on a d-dimensional lattice faster than classical diffusion. In the limit $d \to \infty$, the CTQW dynamics converge to the continuous Dirac equation.

More generally, CTQWs provide a natural framework for quantum search algorithms on graphs, often achieving Grover-like speedups or better. It is also known that continuous-time quantum walks are **universal for quantum computation**, meaning that sufficiently complex graphs can encode arbitrary quantum logic operations.

3.2 Discrete-Time Quantum Walks

In a **discrete-time quantum walk (DTQW)**, evolution proceeds in discrete time steps. Formally, one defines a unitary step operator U that is repeatedly applied to an initial state. Designing such a U for an arbitrary graph is nontrivial, since a naive attempt to "jump to a random neighbor" quantumly is not unitary.

The resolution is to enlarge the Hilbert space by introducing an auxiliary *coin register* that provides additional degrees of freedom for unitary evolution. This model, first proposed by Watrous in the context of quantum graph algorithms, leads to the *coined discrete-time quantum walk*.

3.2.1 Coined Quantum Walk

For a graph $G = (V, E)$, the state space is $\mathcal{H} = \mathbb{C}^{|V|} \otimes \mathbb{C}^d$, where d is the maximum vertex degree (or a fixed value in regular graphs). A basis state $|j, k\rangle$ represents the walker at vertex j with coin state k, where the coin index can be thought of as a label for one of j's outgoing edges.

Each step of the walk is a composition of two operations:

1. A *coin flip* C acting on the coin register (possibly conditioned on the current vertex),
2. A *shift* S that moves the walker to a neighboring vertex determined by the coin state.

In the case of a d-regular graph, a common choice is the Grover diffusion operator as the coin:

$$C = \sum_{j \in V} |j\rangle \langle j| \otimes C_j, \quad C_j := 2 |\chi_j\rangle \langle \chi_j| - I_d,$$

where $|\chi_j\rangle := \frac{1}{\sqrt{\deg(j)}} \sum_{k:(j,k)\in E} |k\rangle$ is the uniform superposition over the coin labels of neighbors of j. This is a Householder reflection about $|\chi_j\rangle$ and acts as a Grover diffusion operator on the coin space.

Next, the shift operator S acts as

$$S |j, k\rangle = |k, j\rangle , \quad \text{for each } (j, k) \in E,$$

i.e., if the coin state k indicates a neighbor of j, the shift S moves the walker to vertex k and updates the coin to j. The overall unitary step of the walk is then:

$$U = SC.$$

By construction, U is unitary and effects a "quantum hop" on the graph at each time step. On a regular graph, C performs a uniform mixing over possible directions (like tossing a d-sided quantum coin), and S implements a conditional shift. On irregular graphs, care must be taken to define appropriate coin spaces, or one may instead use Szegedy's alternate framework.

3.2.2 Szegedy's Walk

In 2004, Szegedy proposed a different formulation of DTQW based on classical Markov chains. Here, no coin register is needed. Instead, the walk is defined on $\mathbb{C}^N \otimes \mathbb{C}^N$, with basis states $|j, k\rangle$ corresponding to directed edges.

Given a classical transition matrix P (an $N \times N$ stochastic matrix), one defines:

$$|\psi_j\rangle := |j\rangle \otimes \sum_k \sqrt{P_{kj}}\,|k\rangle\,, \quad \Pi := \sum_j |\psi_j\rangle\langle\psi_j|\,.$$

Then the walk step is:

$$U = S(2\Pi - I),$$

where S again swaps the two registers: $S|j, k\rangle = |k, j\rangle$.

This construction yields a unitary U, and when $P_{kj} = \frac{1}{\deg(j)} A_{kj}$ (i.e., a symmetric random walk), Szegedy's walk is equivalent to the coined model.

Szegedy's approach is especially useful for quantizing non-regular Markov chains and is widely used in algorithm design—for example, in the element distinctness problem.

3.2.3 Properties and Applications

Discrete-time quantum walks share many features with their continuous-time counterparts. On the infinite line, the standard coined walk (with Hadamard coin) spreads ballistically, creating a bimodal distribution, unlike the diffusive spread of a classical random walk.

Both DTQW and CTQW have been used to achieve quadratic speedups in search problems. In many cases, the two models can simulate each other. DTQWs can even achieve **universal quantum computation**: Childs et al. (2009) constructed a coined DTQW that simulates an arbitrary quantum circuit.

In practice, the coined model is convenient for quantum circuit implementations, as the step $U = SC$ can be decomposed into standard quantum gates. Experimental DTQW implementations have been realized on photonic and ion trap systems.

An important feature of DTQWs is that their behavior depends sensitively on the choice of coin operator. Different coins can induce localization, ballistic spread, or symmetry breaking. This tunability can be exploited in algorithm design. By contrast, CTQWs have less flexibility since the Hamiltonian is fully determined by the graph.

3.3 Comparing Continuous and Discrete-Time Quantum Walks

Continuous- and discrete-time quantum walks are closely related, yet not strictly equivalent models. Both involve the interference of quantum amplitudes on a graph and can produce similar high-level behaviors, such as linear spreading and faster hitting times compared to classical random walks. In many algorithmic contexts, either model can be used to achieve a quantum speedup.

However, several important distinctions exist between the two models:

3.3.1 State Space

- **CTQW:** Evolves on the vertex Hilbert space $\mathbb{C}^{|V|}$.
- **DTQW (Coined):** Uses an enlarged space $\mathbb{C}^{|V|} \otimes \mathbb{C}^d$, introducing a coin register to enable unitary evolution.

The coin adds extra degrees of freedom, allowing more design flexibility. However, it also increases the complexity of analysis and implementation since one must choose both the coin and shift operators carefully to ensure desired evolution.

3.3.2 Time Evolution

- **CTQW:** Evolution is governed by the continuous-time Schrödinger equation: $|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle$, allowing probing at arbitrary real-valued times t.
- **DTQW:** Evolves in discrete time-steps via a unitary operator U applied iteratively: $|\psi(t)\rangle = U^t|\psi(0)\rangle$, with $t \in \mathbb{Z}$.

DTQWs can exhibit periodic revivals and step-dependent interference patterns, while CTQWs exhibit behavior determined by the spectral decomposition of H. A DTQW can approximate a CTQW through Trotterization: setting $U \approx e^{-iH\Delta t}$ for small Δt and iterating.

3.3.3 Generality

- **CTQW:** Any undirected graph naturally defines a Hermitian Hamiltonian and thus a valid walk.
- **DTQW:** For general (especially irregular) graphs, defining a local coinless unitary operator may be challenging. Enlarged Hilbert spaces or alternate frameworks like Szegedy's construction are often required.

While coined DTQWs require more setup, they are ultimately general, and Szegedy's method can handle any reversible Markov chain.

3.3.4 Analytical Tools

- **CTQW:** Analysis reduces to diagonalizing the Hermitian Hamiltonian H.

- **DTQW:** One must diagonalize the larger unitary operator U, which may have a more intricate spectrum due to the coin space.

In both cases, analysis is often facilitated by symmetry. For example, when the underlying graph is a Cayley graph or has high symmetry, representation theory can be used to obtain explicit eigenstates and eigenvalues.

3.3.5 Equivalences

There exist formal correspondences between the two models. For instance:

- A coined DTQW can approximate a CTQW in the limit of small time-steps (Childs et al.).
- CTQWs can be embedded into DTQWs by using additional coin degrees of freedom to emulate continuous evolution.

Thus, the two models are equivalent in computational power. However, in practice, one may be more natural or efficient depending on the problem. For example:

- **CTQW:** Favored for spatial search on low-degree graphs due to the natural definition of a marked Hamiltonian.
- **DTQW:** Preferred for quantizing arbitrary Markov chains (e.g., element distinctness on the Johnson graph), particularly using Szegedy's framework.

3.3.6 Summary

Continuous-time and discrete-time quantum walks are powerful and related models for quantum computation and algorithm design. CTQWs rely on continuous evolution under a Hermitian Hamiltonian and are tightly linked to physical systems. DTQWs require coin and shift operators but offer additional control and ease of implementation in gate-based architectures.

Each model has advantages:

- CTQWs often more closely mirror physical intuition.
- DTQWs offer greater flexibility and modularity for algorithm design.

Results in one model can often be translated to the other with suitable modifications. Consequently, both are indispensable tools in the study of quantum algorithms.

3.4 Cayley Graphs

Having introduced the two types of quantum walks, we now focus on walks on highly symmetric graphs that arise from group structures. A **Cayley graph** encodes the algebraic structure of a group in graph-theoretic form and provides a natural setting for analyzing quantum walks due to its symmetry properties.

Let G be a group with identity element e, and let $S \subset G$ be a generating set for G. The *Cayley graph* $\Gamma(G, S)$ is defined as follows:

- Each group element $g \in G$ corresponds to a vertex.
- For each $g \in G$ and each generator $s \in S$, there is a directed edge from g to gs. That is, (g, gs) is an edge labeled by s.

If S is symmetric (i.e., $s \in S \Rightarrow s^{-1} \in S$) and does not contain the identity, the graph can be treated as **undirected**, since for every edge from g to gs, there is a corresponding edge back from gs to g.

When S is symmetric, $\Gamma(G, S)$ is an undirected $|S|$-regular graph. Note that the Cayley graph depends on the choice of generating set S: different generating sets for the same group yield different graphs, though all share key symmetry features.

3.4.1 Examples of Cayley Graphs

- **Cycle Graph C_n:** The cycle graph is the Cayley graph of the cyclic group $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ with generating set $S = \{1, -1\}$. Each vertex $j \in \mathbb{Z}_n$ is connected to $j \pm 1 \mod n$.
- **Hypercube Q_n:** The n-dimensional Boolean hypercube is the Cayley graph of the group $(\mathbb{Z}_2)^n$ with generating set $S = \{e_1, \ldots, e_n\}$, where e_i flips the i-th bit. This graph is n-regular and bipartite.
- **Complete Graph K_N:** The complete graph on N vertices can be viewed as the Cayley graph of $\mathbb{Z}_N$ with generating set $S = \{1, 2, \ldots, N-1\}$ (all nonzero elements). Each vertex g has an edge to every other vertex $g + s$.
- **Infinite Tree (Free Group):** The free group on two generators a and b has a Cayley graph that is an infinite 4-regular tree. Each vertex corresponds to a word in $\{a^{\pm 1}, b^{\pm 1}\}$, and edges represent multiplication by a or b. A finite portion of this graph is shown in Figure **??**.
- **Dihedral Groups:** Cayley graphs of dihedral groups often form cyclic or prism-like structures. These graphs are used in modeling symmetric structures in both classical and quantum settings.

3.4.2 Symmetry and Group Action

Cayley graphs are highly symmetric. The group G acts regularly on the vertices of $\Gamma(G, S)$ via left multiplication:

$$g : h \mapsto gh.$$

This action is:

- **Vertex-transitive:** All vertices are structurally identical; there is no distinguished origin unless one is chosen explicitly.
- **Regular:** For any two vertices $u, v \in G$, there exists exactly one $g \in G$ such that $g \cdot u = v$.

This symmetry implies that quantum (or classical) walks can often be analyzed by considering representative initial conditions—commonly, the walk is assumed to start at the identity element e.

3.4.3 Remarks

Cayley graphs provide a powerful framework for analyzing quantum walks. Their regularity and symmetry often lead to analytical tractability, particularly using representation theory and spectral techniques. Throughout this thesis, we will explore quantum walks on Cayley graphs, leveraging their symmetry to obtain insights into walk dynamics and algorithmic applications.

3.5 Quantum Walks on Cayley Graphs

Cayley graphs provide an ideal setting for quantum walks due to their rich symmetry properties, which can be exploited to simplify analysis and implementation. Both continuous-time and discrete-time quantum walks on Cayley graphs have been studied extensively. We now highlight several key features and advantages of quantum walks in this setting.

3.5.1 Spectral Structure

The adjacency matrix A of a Cayley graph is closely related to the group algebra of G. It can be expressed as:

$$A = \sum_{a \in G} \lambda_a |\hat{a}\rangle\langle\hat{a}|,$$

where $|\hat{a}\rangle$ is the quantum Fourier transform of $|a\rangle$. The eigenvalues λ are given by

$$\lambda_a = \sum_{q \in Q} \chi(a, q).$$

Note that the eigenvectors $|\hat{a}\rangle$ of A depend only on G and not on the set Q.

To see this, we have

$$A\,|\hat{a}\rangle = A.\frac{1}{\sqrt{|G|}} \sum_{y \in G} \chi(a, y)\,|y\rangle = \frac{1}{\sqrt{|G|}} \sum_{y \in G} \chi(a, y).A\,|y\rangle$$

$$= \frac{1}{\sqrt{|G|}} \sum_{y \in G} \chi(a, y). \sum_{q \in Q} |qy\rangle$$

Consider $\beta = qy$. Then:

$$= \frac{1}{\sqrt{|G|}} \sum_{q \in Q} \chi(a, q) \sum_{\beta} \chi(a, \beta)\,|\beta\rangle = \sum_{q \in Q} \chi(a, q).\,|\hat{a}\rangle$$

$$= \lambda_a\,|\hat{a}\rangle$$

3.5.2 Invariant Subspaces and Symmetry Reduction

Cayley graphs possess vertex-transitive symmetry under the group action $g : h \mapsto gh$. This symmetry induces invariant subspaces for the quantum walk evolution. If the initial state is uniform over a symmetry orbit, it remains so under the walk.

In particular, for a continuous-time quantum walk starting at the identity element:

$$|\psi(t)\rangle = e^{-iAt} |e\rangle ,$$

the resulting state remains a *class function*, i.e., the amplitude $\langle g|\psi(t)|g|\psi(t)\rangle$ depends only on structural properties of g (such as Hamming weight or conjugacy class).

This symmetry drastically reduces the effective state space. For instance, on distance-regular graphs (including many Cayley graphs), amplitudes are equal for all vertices at the same distance from the starting point. This idea is also used in the analysis of the glued trees graph, where the walk remains uniform on each layer.

3.5.3 Hitting and Mixing Properties

Quantum walks on Cayley graphs have been applied to study hitting and mixing times. In the classical case, random walks on many Cayley graphs are rapidly mixing (e.g., those based on nonabelian simple groups, which often form expanders).

In the quantum case, mixing is more subtle due to unitary evolution. One typically considers *instantaneous* or *time-averaged* mixing. Some Cayley graphs support perfect state transfer or periodicity in CTQWs.

Examples:

- On the cycle graph C_n, a CTQW exhibits perfect revival at time $t = \pi$ if n is even.
- On the hypercube, the CTQW exhibits approximate uniform mixing at specific times.

These properties are derived from the spectral structure, which is accessible via group characters.

3.5.4 Discrete-Time Walks and Symmetry

Coined DTQWs also benefit from Cayley graph symmetry. If a symmetric coin (e.g., the Grover coin) is used at each vertex of a vertex-transitive graph, the walk respects the full symmetry. As a result:

- Amplitudes often remain uniform on symmetry-equivalent vertices.
- Walk dynamics can often be analyzed by reducing to a lower-dimensional invariant subspace.
- In some cases, symmetry can lead to *localization*, where amplitude becomes trapped due to spectral properties of the coin.

These effects can be mitigated or exploited through careful coin design or by using Szegedy's method, which naturally respects symmetry.

3.5.5 Summary

Cayley graphs offer a rich and powerful structure for quantum walks. Their group-theoretic properties allow:

- Spectral analysis via irreducible representations.
- Dimensionality reduction using symmetry and invariant subspaces.
- Explicit calculation of probabilities, mixing behavior, and hitting times.

Many efficient quantum walk algorithms exploit these properties either directly or implicitly. In subsequent sections, we will explore quantum walks on more general structures where group actions go beyond simple Cayley graphs.

3.6 Cayley Graphs with Group Action

Cayley graphs can also be generalized through group actions. Let $(G, X, *)$ be a regular group action with a fixed element $x \in X$, and let $Q = \{q_1, q_2, \ldots, q_k\} \subset G$ be a subset as before. Define the Cayley graph $\Gamma = (X, E)$, where the vertex set is X, and the edge set consists of all pairs $(x, y) \in X \times X$ such that $y = q * x$ for some $q \in Q$.

The adjacency matrix of this graph can be written as:

$$A = \sum_{h \in G} \lambda_h \left|G^{(h)} * x\right\rangle \left\langle G^{(h)} * x\right|,$$

where

$$\lambda_h = \sum_{q \in Q} \chi(h, q), \quad \text{and} \quad \left|G^{(h)} * x\right\rangle = \frac{1}{\sqrt{|G|}} \sum_{g \in G} \chi(g, h) \left|g * x\right\rangle.$$

As in the previous case, the eigenvectors $\left|G^{(h)} * x\right\rangle$ are determined solely by the structure of the group G. This construction generalizes the standard Cayley graph: setting $X = G$ and the group action $*$ as group multiplication recovers the original definition.

To see this, we have:

$$A \left|G^{(h)} * x\right\rangle = \frac{1}{\sqrt{|G|}} . \sum_{g \in G} \chi(g, h) A . \left|g * x\right\rangle$$

$$= A \left|G^{(h)} * x\right\rangle = \frac{1}{\sqrt{|G|}} . \sum_{g \in G} \chi(g, h) \sum_{q \in Q} \left|q * (g * x)\right\rangle$$

Consider $\beta = qg$. Then:

$$== \frac{1}{\sqrt{|G|}} \sum_{q \in Q} \sum_{\beta \in G} \chi(\beta, h) \chi(q, h) \left|\beta * x\right\rangle$$

$$= \sum_{q \in Q} \chi(q, h) . \left|G^{(h)} * x\right\rangle = \lambda_h \left|G^{(h)} * x\right\rangle$$

Since the group action $(G, X, *)$ is regular, the two Cayley graph constructions are isomorphic. In the first construction, the vertex set is G, and the adjacency matrix is indexed by elements of G. In the second construction, the vertex set is X, with the adjacency matrix indexed by elements of X. The isomorphism between these graphs is given by the bijection

$$\begin{aligned} \phi : G &\longrightarrow X \\ g &\longmapsto g * x, \end{aligned}$$

which maps each group element $g \in G$ to the corresponding point in its orbit under the action.

When the group action $(G, X, *)$ is assumed to be cryptographic, the isomorphism ϕ becomes a one-way function: given $g \in G$, it is efficient to compute $g * x$, but given the pair $(x, g * x)$, it is computationally hard to recover g. As a result, while the two Cayley graphs are mathematically isomorphic, computational tasks associated with them may require fundamentally different approaches. A particularly illustrative example is the implementation of quantum walks on these graphs.[1]

In the next section, we develop techniques for simulating continuous-time quantum walks on such symmetric graphs using quantum circuits. These simulations exploit the group action structure to simplify circuit construction and enhance algorithmic efficiency.

[1]Continuous-time quantum walks have been explored in the context of cryptographic group actions in [**?**, **?**], specifically in supersingular isogeny graphs. These walks can be viewed as group-action quantum walks.

Chapter 4

Simulating Group Action Quantum Walks

From this point forward, we assume the generating set Q is small, i.e., $|Q| = \text{poly}(\log |G|)$. In the first construction (where $X = G$), the walk e^{-iAt} can be efficiently implemented even for values of t that are exponentially large in $\log |G|$. This is a consequence of the identity:

$$e^{-iAt} = \text{QFT}_G \left(\sum_{a \in G} e^{-i\lambda_a t} |a\rangle \langle a| \right) \text{QFT}_G^*, \tag{4.1}$$

where QFT_G and its inverse can be implemented in $\text{poly}(\log |G|)$ time. Moreover, the diagonal phase unitary $|a\rangle \mapsto e^{-i\lambda_a t} |a\rangle$ is also efficient to implement, since the eigenvalues λ_a can be computed classically to arbitrary precision in $\text{poly}(\log |G|)$ time.

In contrast, the group-action setting presents greater computational difficulty. Although the states $\left|G^{(h)} * x\right\rangle$ play a role analogous to the Fourier basis states $\left|\hat{h}\right\rangle$, the presence of the action $*$ makes the situation more subtle. In particular, implementing general transformations over these states is difficult; we are typically limited to operations of the form $|y\rangle \mapsto |a * y\rangle$ for $a \in G$. As a result, no efficient decomposition similar to (4.1) is known in the group-action setting. Instead, the best we can express is the spectral decomposition:

$$e^{-iAt} = \sum_{h \in G} e^{-i\lambda_h t} \left|G^{(h)} * x\right\rangle \left\langle G^{(h)} * x\right| .$$

Despite this limitation, the adjacency matrix A retains a high degree of sparsity and algebraic structure. In the next section, we will show that for $t = \text{poly}(\log |G|)$, the walk e^{-iAt} can still be efficiently approximated to polynomial accuracy.

4.0.1 Simulating continuous-time walks

Childs and Berry, Childs, and Kothari showed that continuous-time quantum walks can be simulated using discrete-time quantum walks. Since our approach relies on their methods, we begin by outlining the relevant notation and framework.

Let H be a Hamiltonian of dimension $N = 2^n$, and define $\|H\|_{\max} = \max_{i,j} |H_{ij}|$. To facilitate the simulation, we expand the Hilbert space from $\mathbb{C}^N$ to $\mathbb{C}^{2N} \otimes \mathbb{C}^{2N}$ by appending an ancilla qubit initialized to $|0\rangle$ and duplicating the resulting space.

To define the discrete-time walk operator, we first construct an orthonormal set of states:

$$\begin{aligned}
|\phi_{j0}\rangle &:= \frac{1}{\sqrt{d}} \sum_{\ell \in F_j} |\ell\rangle \left(\sqrt{\frac{H^*_{j\ell}}{K}} |0\rangle + \sqrt{1 - \frac{|H^*_{j\ell}|}{K}} |1\rangle \right), \\
|\phi_{j1}\rangle &:= |0\rangle |1\rangle,
\end{aligned} \tag{4.2}$$

where F_j denotes the set of nonzero entries in column j of H, and $K \geq \|H\|_{\max}$ is a fixed constant. Based on these states, we define the isometry $T : \mathbb{C}^{2N} \to \mathbb{C}^{2N} \otimes \mathbb{C}^{2N}$ as follows:

$$T := \sum_{j=0}^{N-1} \sum_{b \in \{0,1\}} (|j\rangle \langle j| \otimes |b\rangle \langle b|) \otimes |\phi_{jb}\rangle . \tag{4.3}$$

The discrete-time quantum walk operator is then defined as

$$W = iS(2TT^* - \mathbf{1}),$$

where S is the swap operator acting as

$$S |j_1\rangle |b_1\rangle |j_2\rangle |b_2\rangle = |j_2\rangle |b_2\rangle |j_1\rangle |b_1\rangle ,$$

for all $0 \leq j_1, j_2 < N$ and $b_1, b_2 \in \{0, 1\}$.

To efficiently simulate the continuous-time evolution e^{-iHt}, it suffices to implement the isometry T, its adjoint T^*, and the walk operator W efficiently.

4.0.2 Simulating group action quantum walks.

Assume $t = \mathrm{poly}(\log |G|)$. We show that the continuous-time walk $W = e^{-iAt}$ can be efficiently simulated using the discrete-time quantum walk framework developed in [?, ?]. Recall the isometry T defined in Equation (4.3), where the states $|\phi_b\rangle$ are as given in Equation (4.2). To carry out the simulation, it suffices to show that the isometry T, its adjoint T^*, and the discrete-time walk operator

$$W = iS(2TT^* - \mathbf{1})$$

can all be efficiently implemented in the group-action setting.

Let $\Gamma = (X, E)$ be the Cayley graph associated with the group action $(G, X, *)$, and let A denote its adjacency matrix. The Hamiltonian H of interest is simply A, indexed by elements of X. In this context, the isometry T takes the form:

$$\begin{aligned}
T &:= \sum_{y \in X} \sum_{b \in \{0,1\}} (|y\rangle \langle y| \otimes |b\rangle \langle b|) \otimes |\phi_{yb}\rangle \\
&= \sum_{y \in X} \sum_{b \in \{0,1\}} |y\rangle |b\rangle |\phi_{yb}\rangle \langle y| \langle b| .
\end{aligned}$$

Because of the structure of Γ, the states .... simplify as follows. First, since the set Q is symmetric, the graph Γ is undirected and the adjacency matrix A is symmetric with non-negative entries; the nonzero entries of A are all equal to 1. Therefore, we can set $K = A_{\max} = 1$. As a result, for $y \in X$, the state $|\phi_{y0}\rangle$ becomes

$$|\phi_{y0}\rangle = \frac{1}{\sqrt{|Q|}} \sum_{q \in Q} |q * y\rangle |0\rangle .$$

We now describe how to implement the isometry T by constructing efficient unitaries U_0 and U_1, where each U_b acts as

$$U_b : |0\rangle |y, b\rangle |0, 0\rangle \mapsto |0\rangle |y, b\rangle |\phi_{yb}\rangle , \quad b \in \{0, 1\},$$

with the first register serving as an ancilla.

The unitary U_1 is straightforward to implement, since $|\phi_{y1}\rangle = |0\rangle |1\rangle$ is independent of y. We now focus on the construction of U_0.

Because $Q \subset G$ is a small set (i.e., $|Q| = \text{poly}(\log |G|)$), we can construct an efficient unitary $V_Q : \mathbb{C}^Q \to \mathbb{C}^Q$ such that

$$V_Q |0\rangle = \frac{1}{\sqrt{|Q|}} \sum_{q \in Q} |q\rangle .$$

Applying $V_Q \otimes \mathbf{1}$ to the state $|0\rangle |y, 0\rangle |0, 0\rangle$ produces:

$$\frac{1}{\sqrt{|Q|}} \sum_{q \in Q} |q\rangle |y, 0\rangle |0, 0\rangle .$$

Next, we apply the unitary V_1, defined by:

$$V_1 : |q\rangle |y, 0\rangle |0, 0\rangle \mapsto |q\rangle |y, 0\rangle |q * y, 0\rangle .$$

Following this, we uncompute the first register to recover $|0\rangle |y, 0\rangle |\phi_{y0}\rangle$. This is achieved by applying the unitary V_2, defined as:

$$V_2 : |q\rangle |y, 0\rangle |q * y, 0\rangle \mapsto |0\rangle |y, 0\rangle |q * y, 0\rangle .$$

Since Q is small, we can efficiently determine q from the pair $(y, q * y)$ by exhaustively checking each $q \in Q$. Thus, V_2 (and therefore U_0) can be implemented efficiently.

To implement $U_0^* : |0\rangle |y, 0\rangle |\phi_{y0}\rangle \mapsto |0\rangle |y, 0\rangle |0, 0\rangle$, we sequentially apply V_2^*, followed by V_1^*, and finally $V_Q^* \otimes \mathbf{1}$. Since each of these unitaries can be implemented efficiently, it follows that U_0^* is also efficient.

We now construct the isometry T using a conditional unitary U_T, which applies either U_0 or U_1 depending on the value of the qubit b. That is,

$$U_T = \text{Controlled-}U_b \quad \text{with control on } b \in \{0, 1\}.$$

Next, we demonstrate that the walk unitary

$$W = iS(2TT^* - \mathbf{1})$$

can be applied efficiently. This reduces to implementing the reflection $2TT^* - \mathbf{1}$, which we analyze as follows:

$$\begin{aligned}
2\left|0\right\rangle\left\langle 0\right|\otimes TT^{*}-\mathbf{1} &= 2\left|0\right\rangle\left\langle 0\right|\otimes\sum_{y\in X}\sum_{b\in\{0,1\}}\left|y,b\right\rangle\left|\phi_{yb}\right\rangle\left\langle y,b\right|\left\langle\phi_{yb}\right|-\mathbf{1}\\
&= 2\sum_{y\in X}\sum_{b\in\{0,1\}}\left|0\right\rangle\left|y,b\right\rangle\left|\phi_{yb}\right\rangle\left\langle 0\right|\left\langle y,b\right|\left\langle\phi_{yb}\right|-\mathbf{1}\\
&= U_T\left(2\sum_{y\in X}\sum_{b\in\{0,1\}}\left|0\right\rangle\left|y,b\right\rangle\left|0,0\right\rangle\left\langle 0\right|\left\langle y,b\right|\left\langle 0,0\right|-\mathbf{1}\right)U_T^{*}\\
&= U_T\left(2\left|0\right\rangle\left\langle 0\right|\otimes\mathbf{1}_{X,b}\otimes\left|0,0\right\rangle\left\langle 0,0\right|-\mathbf{1}\right)U_T^{*}.
\end{aligned}$$

Since U_T, U_T^{*}, and the operator $2\left|0\right\rangle\left\langle 0\right|\otimes\mathbf{1}_{X,b}\otimes\left|0,0\right\rangle\left\langle 0,0\right|-\mathbf{1}$ can all be implemented efficiently, it follows that the reflection $2\left|0\right\rangle\left\langle 0\right|\otimes TT^{*}-\mathbf{1}$ is also efficient.

Finally, for any input state $\left|\psi\right\rangle$, the reflection acts as:

$$\left(2\left|0\right\rangle\left\langle 0\right|\otimes TT^{*}-\mathbf{1}\right)\left|0\right\rangle\left|\psi\right\rangle=\left|0\right\rangle\left(2TT^{*}-\mathbf{1}\right)\left|\psi\right\rangle,$$

confirming that the full walk operator W can be implemented efficiently in the group-action setting.

Chapter 5

Quantum Money and Hartley Transform

We now present our public-key quantum money scheme based on group actions and the Hartley transform. We begin by reviewing Zhandry's quantum money scheme from abelian group actions (which uses the Fourier transform) as a baseline. Then we describe our Hartley-transform-based variant and discuss its properties. The quantum Hartley transform will play a central role in both the minting and verification of banknotes.

5.1 Public-Key Quantum Money: Definitions

A public-key quantum money scheme consists of two algorithms, traditionally called Mint and Verify:

- $\mathsf{Gen}(1^n)$ is a probabilistic quantum polynomial-time (QPT) algorithm that, given a security parameter n, outputs a pair $(s, |\$\rangle)$. Here s is a classical string called the *serial number* (or public key) and $|\$\rangle$ is a quantum state representing the banknote. We often call the pair $(s, |\$\rangle)$ a banknote, and s by itself the serial number of that banknote.
- $\mathsf{Verify}(s, |\$\rangle)$ is a QPT algorithm (which may be implemented as a quantum circuit acting on the state $|\$\rangle$ and some ancillary workspace). It takes as input a serial number s and an alleged quantum banknote state $|\$\rangle$. It outputs `accept` (valid) or `reject` (invalid).

We require two properties from such a scheme:

Correctness: For any valid output $(s, |\$\rangle)$ of $\mathsf{Mint}(1^n)$, the verification algorithm accepts with high probability. Formally, $\Pr[\mathsf{Verify}(s, |\$\rangle) = \texttt{accept}]$ should be very close to 1 (typically we require it to be $1 - negl(n)$, where $negl(n)$ denotes a negligible function in n).

Security (Unforgeability): No efficient adversary, given one or more genuine banknotes, can produce a new valid banknote. In the strongest formulation, if an adversary is given a single valid $(s, |\$\rangle)$, it should be infeasible for them to produce two quantum states $|\$'\rangle$ and $|\$''\rangle$ such that $\mathsf{Verify}(s, |\$'\rangle) = \mathsf{Verify}(s, |\$''\rangle) = \texttt{accept}$ (i.e., they cannot create a second copy that passes verification while retaining the original). More generally, even given multiple banknotes, the adversary should not be able to create new ones with fresh serial numbers that pass verification. This captures the idea that quantum money cannot be counterfeited.

In a *public-key* scheme, the verifier uses only the public information (here, the serial number and any scheme-wide public parameters) to check validity, and does not need any secret key. The bank (mint) may have a secret key to produce the states, but after minting, the bank's involvement is not needed. In our scheme, the serial number will effectively contain the information needed for public verification (similar to how a classical bill might have a number anyone can look up, except here it's used in a quantum procedure).

We next outline Zhandry's scheme, which our scheme builds upon.

5.2 Fourier-Based Quantum Money from Group Actions

Zhandry's quantum money scheme [31] is built on an abelian group action $G \curvearrowright X$. We have a family of group actions parameterized by the security parameter n (so the size of G grows with n). For simplicity, assume $|G| = N$. The scheme also fixes a particular element $x_0 \in X$ that everyone knows (a public base point in X).

The minting algorithm Mint in Zhandry's scheme works as follows (informally):

1. On input 1^n, sample a random group element $g \leftarrow G$. This g will serve as the serial number s for the banknote.
2. Prepare two registers: the first is a group element register, initialized to $|0_G\rangle$ (an encoding of the identity element of G), and the second is an X-register initialized to $|x_0\rangle$.
3. Apply the quantum Fourier transform over G to the first register. This creates a uniform superposition over G:
$$\frac{1}{\sqrt{N}} \sum_{h \in G} |h\rangle \otimes |x_0\rangle .$$
4. Now apply the group action in a controlled manner: for each basis state $|h\rangle$ in the first register, apply h's action on the second register. This transforms the joint state into
$$\frac{1}{\sqrt{N}} \sum_{h \in G} |h\rangle \otimes |h * x_0\rangle ,$$
which is an entangled state between a superposition of group elements and their corresponding X-elements.
5. Apply the inverse QFT (which is the adjoint of the QFT) on the first register. This recombines the amplitudes in the first register. The result (by properties of the QFT) can be shown to be
$$\frac{1}{\sqrt{N}} \sum_{h \in G} \omega^{\langle g, h\rangle} |h\rangle \otimes |h * x_0\rangle ,$$
where $\omega^{\langle g, h\rangle}$ denotes the phase picked up that depends on g and h. In fact, this state can be written as $|g\rangle \otimes \frac{1}{\sqrt{N}} \sum_{h \in G} |h * x_0\rangle$; essentially, the first register ends up in $|g\rangle$ (the serial number) and the second register is an equal superposition of all elements of X in the orbit of x_0 (which is X itself, since the action is transitive) but weighted by a phase related to g. This is the so-called group-action Fourier state.
6. The first register (which contains $|g\rangle$) is measured in the computational basis, yielding the random serial number g and collapsing the second register to a state that we will denote $|\$_g\rangle$. This $|\$_g\rangle$ is the actual quantum money state corresponding to serial g.
7. Output the pair $(s = g,\ |\$\rangle = |\$_g\rangle)$.

In the above, a bit of algebra reveals that $|\$_g\rangle = \frac{1}{\sqrt{N}} \sum_{h \in G} \chi_g(h) \, |h * x_0\rangle$, where $\chi_g(h) = \omega^{\langle g, h \rangle}$ is essentially the character of G indexed by g. In other words, $|\$_g\rangle$ is (up to global phase) the state one gets by applying the representation (character) χ_g to weight the superposition over the orbit X. These states for different g are almost orthogonal and can be distinguished with the right measurement.

The verification algorithm $\mathsf{Verify}(g, |\$_g\rangle)$ for Zhandry's scheme goes as follows:

1. Given the purported serial number $g \in G$ and an input state $|\psi\rangle$ (which should equal $|\$_g\rangle$ if genuine), attach an auxiliary register prepared in $|x_0\rangle$.

2. Perform a controlled group action "kickback" using g on the auxiliary register. Specifically, perform the unitary that maps $|y\rangle \otimes |x_0\rangle \mapsto |y\rangle \otimes |g * x_0\rangle$ for each basis element $|y\rangle$ of the first register. (This operation uses the classical g to act on the second register conditioned on the first—essentially, it multiplies the second register's state by g if the first register is in an appropriate basis state. There are various ways to implement this, but conceptually it's a controlled operation using the known g.)

3. Now measure the second register (the auxiliary one) in the computational basis of X to see if it is still $|x_0\rangle$. If the state $|\psi\rangle$ was the correct Fourier state $|\$_g\rangle$, one can show that this measurement will yield x_0 with high probability (constructive interference causes the x_0 component to be strong). If $|\psi\rangle$ was not the correct state, the measurement is unlikely to give x_0.

4. Accept if and only if the measurement outcome is x_0 (i.e. the second register returns to the base state).

The intuition is that the genuine state $|\$_g\rangle$ contains a phase that exactly cancels out the action of g in the verification step, causing the auxiliary register to end up back at x_0. A counterfeit state, by contrast, would not generally have the right phase relationship, and the auxiliary register's state would be some superposition over X that is not concentrated on x_0, hence likely giving a different outcome upon measurement.

Zhandry's scheme is shown to be correct (valid notes pass) and is believed to be secure under the assumption that the group action is one-way. The hardness essentially boils down to: given x_0 and $|\$_g\rangle$, an adversary cannot figure out a different g' that would produce a second valid state $|\$_{g'}\rangle$ or replicate $|\$_g\rangle$ (because that would solve the hidden subgroup or discrete log type problem in the group action).

5.3 Hartley-Transform Money Scheme Construction

Our scheme modifies the above Fourier-based construction by utilizing the **quantum Hartley transform (QHT)** in place of the QFT. Since the Hartley transform is closely related to the Fourier transform (essentially it produces the real and imaginary parts in a single real-valued transform), one might expect this substitution to be straightforward. And indeed, for the minting procedure it largely is: we will use the QHT to create real-amplitude superposition states.

Concretely, assume again a group action $G \curvearrowright X$ with $|G| = N$. We fix the same generating set S and basepoint x_0. The Mint_H (Hartley mint) algorithm:

1. Sample a random $g \leftarrow G$ to be the serial number.

2. Prepare $|0_G, x_0\rangle$ as before.

3. Apply the *quantum Hartley transform* over G to the first register. The Hartley transform (on an abelian group of order N) is defined as:

$$\mathrm{QHT} : |h\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k \in G} \mathrm{cas}\Big(\frac{2\pi\langle k, h\rangle}{N}\Big)|k\rangle,$$

where $\mathrm{cas}(\theta) = \cos\theta + \sin\theta$ and $\langle k, h\rangle$ denotes some pairing like an exponent or dot product (for cyclic groups this is just kh). In simpler terms, $\mathrm{QHT}(|0_G\rangle) = \frac{1}{\sqrt{N}} \sum_{k \in G} |k\rangle$ (because $cas(0) = 1$), *the same uniform superposition as the QFT produces for* $|0\rangle$. So after this step, we again have $\frac{1}{\sqrt{N}} \sum_{h \in G} |h\rangle|x_0\rangle$, but crucially the coefficients are all *real* and positive (in fact all $1/\sqrt{N}$).

4. Perform the controlled group action by the first register on the second: $\frac{1}{\sqrt{N}} \sum_h |h\rangle|h * x_0\rangle$.

5. Apply the inverse QHT on the first register (which being its own inverse up to a constant factor, is the same operation again since Hartley is symmetric like Fourier). Now here is where things differ: after this operation, the joint state will be

$$\frac{1}{\sqrt{N}} \sum_{h \in G} \mathrm{cas}\Big(\frac{2\pi\langle g, h\rangle}{N}\Big)\, |h\rangle \otimes |h * x_0\rangle.$$

Because the Hartley transform combines what would be cosine and sine terms, this state is not as nicely factorable as in the Fourier case. In the Fourier case, we got $|g\rangle \otimes$ (orbit state). In the Hartley case, we do not get a single basis state in the first register; instead we get a superposition in the first register with real coefficients $c_h = cas(2\pi\langle g, h\rangle/N)$. However, note that c_h is just some real number in $[-1, 1]$ depending on g and h. Another way to express the above state is:

$$|\$\rangle = \frac{1}{\sqrt{N}} \sum_{x \in X} \Big(\frac{1}{\sqrt{N}} \sum_{h : h * x_0 = x} \mathrm{cas}\big(\frac{2\pi\langle g, h\rangle}{N}\big)\Big)\, |x\rangle.$$

Since for a given $x \in X$, all h such that $h * x_0 = x$ have the form $h = ug$ for a unique $u \in G$ (because the action is free and transitive), one can simplify the internal sum using properties of the cas function. But we need not dive into that here.

6. No measurement of the first register is performed. In fact, we do not have a nice product state separation, so we cannot simply measure one register to get g. Instead, the output of minting is just the quantum state $|\$\rangle$ (which is entangled between what used to be two registers) and the known classical serial number g (we know what g we picked).

The resulting money state $|\$\rangle$ has only real amplitudes (because the initial superposition was real and the controlled operation doesn't introduce complex numbers, and the Hartley transform yields real coefficients). In particular, in the computational basis of X, the state $|\$\rangle$ is some real-vector state (with both positive and negative values possible). We can think of it as a "Hartley orbit state" corresponding to g.

The verification algorithm in our Hartley-based scheme attempts to mirror the Fourier-based verification: it will use the given serial g and try to check the state. A first idea would be: do exactly the same as Zhandry's verify (the group-action kickback and measure). However, this fails in certain cases. Specifically, the Hartley transform being real causes an ambiguity: it turns out that there exist distinct $g \neq g'$ for which the Hartley-based states $|\$\rangle_g$ and $|\$\rangle_{g'}$ are not orthogonal, and the original verification might accept a forged state that is a certain superposition. In fact, the verification algorithm might accept an illegitimate state with some probability bounded away from zero. This is the "breakdown" of the straightforward Hartley substitution that we alluded to earlier.

To address this, our verification algorithm takes a more sophisticated approach:

1. Given $(g, |\psi\rangle)$, it first uses the **quantum walk-based phase estimation** method (to be detailed in Chapter 6) to *compute the eigenphase corresponding to g*. In short, we will treat $|\psi\rangle$ as an eigenvector of the group action adjacency matrix with a certain eigenvalue λ related to g. We run a procedure to estimate λ to high precision.
2. From the estimated eigenvalue(s), we reconstruct a candidate group element g^* that we believe generated the state. (This uses the fact that by running phase estimation for multiple carefully chosen walk durations, one can solve for g as shown in [31].)
3. We check if $g^* = g$ (the serial number provided). If not, we reject, since the state does not match the serial. If yes, we accept.

This procedure essentially extracts the hidden group element from the state and compares it to the claimed serial. If an adversary tries to forge a state $|\psi\rangle$ for a known serial g, the only way to consistently pass verification is if $|\psi\rangle$ truly encodes the same g in its eigen-spectrum. A fake note that was causing trouble for the naive Hartley verification (e.g., a state that is a mixture of two different "Hartley orbit" states) would yield an eigenphase that does not correspond to a single valid g, and the algorithm would detect the mismatch.

The key quantum tool enabling this verification is the continuous-time quantum walk on the group action graph and the ability to simulate it, which we prepared in Chapter 4. The walk Hamiltonian $A(X, S)$ has the genuine money state $|\$\rangle_g$ as an eigenstate with a known eigenvalue $\lambda(g)$. By running phase estimation on e^{-iAt} with the state $|\psi\rangle$, we attempt to measure that eigenvalue. We repeat for a few different values of t (the "twists" or different phases) to get enough information to pinpoint g. This approach can be viewed as adding extra "twists" to the verification: effectively, each choice of t in phase estimation is like looking at the state in a different interference fringe pattern, analogous to applying different group actions as twists in the verification procedure. This is why we described the new verification as using group action twists—by varying a continuous parameter related to the group action's eigenvalues, we break the symmetry that allowed a counterfeit to slip through.

We defer the full analysis of the success probabilities and how many repetitions are needed to Chapter 6, where we describe the verification algorithm in detail. For now, the takeaway is that by using the quantum walk approach, we can reliably verify Hartley-based money states, restoring security while still enjoying the properties of real amplitudes.

5.4 Efficient Quantum Hartley Transform Implementation

Before moving on to the verification algorithm, we briefly discuss how we implement the quantum Hartley transform (QHT) efficiently, as this is an important practical aspect of our scheme. The Fourier transform is well-known to have efficient circuits; for the Hartley transform, fewer results are available, so we contributed a new circuit construction.

The QHT we need is over an abelian group, which for concreteness one can think of as $\mathbb{Z}_N$ (the general finite abelian group case can usually be reduced to a direct sum of cyclic groups, applying QHT on each cyclic component). Classical Hartley transforms have a recursive structure similar to fast Fourier transforms. In fact, there are known formulas to express a Hartley transform of size N in terms of two Hartley transforms of size $N/2$ plus some additional linear operations [25]. We leveraged such structures to design a divide-and-conquer quantum circuit.

Our algorithm for QHT works roughly as follows: - If N is even, we can express the length-N Hartley transform in terms of two length-$N/2$ Hartley transforms plus a combination of cheap operations (additions, permutations of data, etc.). We implement this decomposition recursively as a quantum circuit. The base of the recursion is when the size is small (like $N = 2$ or 4, where the transform can be done with a constant number of gates). - We found that by carefully optimizing this recursion and using some known quantum subroutines for certain linear combinations, the overall gate count is improved compared to naive methods or previous proposals [21, 22]. In particular, our circuit avoids introducing any quantum Fourier transform internally (unlike [22] which computed Hartley via Fourier plus some adjustments), and this yields a purely real rotation-based circuit.

Our algorithm for the quantum Hartley transform, QHT_N, is modeled after the recursive structure of the quantum Fourier transform algorithm QFT_N. To motivate this, we begin by reviewing the standard approach for computing QFT_N. For clarity, we assume $N = 2^n$, allowing each element of $\mathbb{Z}_N$ to be represented using exactly n qubits. The same recursive framework can be extended to handle general values of N. For any $a \in \mathbb{Z}_N$, the quantum Fourier transform is defined as:

$$\begin{aligned}\mathsf{QFT}_N |a\rangle &= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{ay} |y\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{y=0}^{N/2-1} \omega_N^{ay} |y\rangle + (-1)^a \frac{1}{\sqrt{N}} \sum_{y=0}^{N/2-1} \omega_N^{ay} |y + N/2\rangle \\ &= \frac{1}{\sqrt{N/2}} \sum_{y=0}^{N/2-1} \omega_N^{ay} \frac{1}{\sqrt{2}} (|0\rangle + (-1)^a |1\rangle) |y\rangle ,\end{aligned}$$

In the final expression, we have isolated the first qubit for clarity. Let $|a\rangle = |t\rangle |b\rangle$, where b denotes the least significant bit of a, implying that $a = 2t + b$ for some $t \in \mathbb{Z}_{N/2}$. Applying $\mathsf{QFT}_{N/2}$ to the first register yields the state:

$$\frac{1}{\sqrt{N/2}} \sum_{y=0}^{N/2-1} \omega_N^{2ty} |y\rangle |b\rangle .$$

Next, we apply the phase unitary $P(y, b)$, defined by $|y\rangle |b\rangle \mapsto \omega_N^{by} |y\rangle |b\rangle$, followed by a Hadamard transform on the final qubit. This results in the state given in Equation (**??**).

We now describe our algorithm for efficiently computing the quantum Hartley transform QHT_N. The key idea is to leverage the recursive structure of QHT_N, analogous to the strategy used for QFT_N. To facilitate this, we begin by rewriting the summation in Equation (**??**) to expose its recursive form. Our approach is as follows:

$$\frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big) |y\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big) |y\rangle + \frac{1}{\sqrt{N}} \sum_{y=N/2}^{N-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big) |y\rangle . \qquad (5.1)$$

The second summation on the right-hand side can be expressed as:

$$\begin{aligned}\sum_{y=N/2}^{N-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big) |y\rangle &= \sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{2\pi ay}{N} + \pi a\Big) |y + N/2\rangle \\ &= (-1)^a \sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big) |y + N/2\rangle ,\end{aligned}$$

The second equality follows from the angle-sum identity for the cas function,

$$\operatorname{cas}(\alpha + \beta) = \cos(\alpha)\operatorname{cas}(\beta) + \sin(\alpha)\operatorname{cas}(-\beta),$$

along with the fact that $\cos(\pi a) = (-1)^a$ for all integers a. Substituting this into Equation (5.1) yields:

$$\begin{aligned}\frac{1}{\sqrt{N}}\sum_{y=0}^{N-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big) |y\rangle &= \frac{1}{\sqrt{N}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big)(|y\rangle + (-1)^a |y + N/2\rangle) \\ &= \frac{1}{\sqrt{N/2}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{2\pi ay}{N}\Big)\frac{1}{\sqrt{2}}(|0\rangle + (-1)^a |1\rangle) |y\rangle , \qquad (5.2)\end{aligned}$$

In the final equality, the most significant qubit has been isolated to emphasize its contribution to the overall transformation.

We now demonstrate how to compute QHT_N recursively. For a given $a \in \mathbb{Z}_N$, we decompose the basis state as $|a\rangle = |t\rangle |b\rangle$, where b denotes the least significant bit, such that $a = 2t + b$ for some $t \in \mathbb{Z}_{N/2}$. Assuming the existence of an efficient quantum circuit for computing $\mathsf{QHT}_{N/2}$, we proceed to construct QHT_N in a recursive fashion. To begin the transformation on $|a\rangle$, we introduce an ancilla qubit initialized to $|0\rangle$, yielding the joint state $|0\rangle |t\rangle |b\rangle$. We then proceed as follows:

$$\begin{aligned}|0\rangle |t\rangle |b\rangle &\mapsto \frac{1}{\sqrt{N/2}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{2\pi ty}{N/2}\Big) |0\rangle |y\rangle |b\rangle && (\mathbf{1} \otimes \mathsf{QHT}_{N/2} \otimes \mathbf{1}) \\ &= \frac{1}{\sqrt{N/2}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{4\pi ty}{N}\Big) |0\rangle |y\rangle |b\rangle \\ &\mapsto \frac{1}{\sqrt{N}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{4\pi ty}{N}\Big)(|0\rangle + |1\rangle) |y\rangle |b\rangle . && (H \otimes \mathbf{1})\end{aligned}$$

Next, we apply a controlled negation operation, where the ancilla qubit is flipped if and only if the least significant bit b is equal to 1.

$$V : |0\rangle |y\rangle \mapsto |0\rangle |y\rangle , \quad |1\rangle |y\rangle \mapsto |1\rangle |N/2 - y\rangle ,$$

$$\frac{1}{\sqrt{N}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{4\pi ty}{N}\Big) |0\rangle |y\rangle |b\rangle + \frac{1}{\sqrt{N}}\sum_{y=0}^{N/2-1} \operatorname{cas}\Big(\frac{4\pi ty}{N}\Big) |1\rangle |-y\rangle |b\rangle .$$

By performing a change of variables in the second summation and using the identity $\operatorname{cas}\left(\frac{4\pi t(N/2-y)}{N}\right) =$

$\mathrm{cas}\left(-\frac{4\pi ty}{N}\right)$, we obtain the state:

$$\frac{1}{\sqrt{N}} \sum_{y=0}^{N/2-1} \Big(\mathrm{cas}\Big(\frac{4\pi ty}{N}\Big) |0\rangle + \mathrm{cas}\Big(-\frac{4\pi ty}{N}\Big) |1\rangle \Big) |y\rangle |b\rangle .$$

We define the single-qubit rotation $R(\theta)$ as the unitary operator acting on a single qubit given by:

$$R(y,b) = \begin{bmatrix} \cos(2\pi by/N) & \sin(2\pi by/N) \\ -\sin(2\pi by/N) & \cos(2\pi by/N) \end{bmatrix}, \tag{5.3}$$

Now, consider the unitary U_R defined by

$$U_R : |c\rangle |y\rangle |b\rangle \mapsto (R(y,b)|c\rangle) |y\rangle |b\rangle ,$$

where $R(y,b)$ is a single-qubit rotation depending on y and b. Applying U_R followed by the controlled negation V, we obtain the state:

$$\begin{aligned} |\phi_1\rangle &= \frac{1}{\sqrt{N}} \sum_{y=0}^{N/2-1} \Big(\mathrm{cas}\Big(\frac{2\pi ay}{N}\Big) |0\rangle + \mathrm{cas}\Big(-\frac{2\pi a(N/2-y)}{N}\Big) |1\rangle \Big) |y\rangle |b\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{y=0}^{N/2-1} \mathrm{cas}\Big(\frac{2\pi ay}{N}\Big)\Big(|0\rangle + (-1)^b |1\rangle \Big) |y\rangle |b\rangle , \end{aligned}$$

Here, we have used the identity $\mathrm{cas}\left(-\pi a + \frac{2\pi ay}{N}\right) = (-1)^a \mathrm{cas}\left(\frac{2\pi ay}{N}\right)$, along with the fact that $(-1)^a = (-1)^b$. Applying the Hadamard transform to the first qubit yields:

$$|\psi\rangle = \frac{1}{\sqrt{N/2}} \sum_{y=0}^{N/2-1} \mathrm{cas}\Big(\frac{2\pi ay}{N}\Big) |b\rangle |y\rangle |b\rangle$$

Next, we uncompute the first qubit by applying a CNOT gate with the last qubit as control. We then apply a Hadamard transform followed by a swap operation, resulting in the state:

$$\begin{aligned} |\psi\rangle &\mapsto \frac{1}{\sqrt{N/2}} \sum_{y=0}^{N/2-1} \mathrm{cas}\Big(\frac{2\pi ay}{N}\Big) |0\rangle |y\rangle |b\rangle && (\textsc{cnot}) \\ &\mapsto \frac{1}{\sqrt{N/2}} \sum_{y=0}^{N/2-1} \mathrm{cas}\Big(\frac{2\pi ay}{N}\Big) |0\rangle |y\rangle \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b |1\rangle) && (\mathbf{1} \otimes H) \\ &= \frac{1}{\sqrt{N/2}} \sum_{y=0}^{N/2-1} \mathrm{cas}\Big(\frac{2\pi ay}{N}\Big) |0\rangle |y\rangle \frac{1}{\sqrt{2}}(|0\rangle + (-1)^a |1\rangle). && ((-1)^b = (-1)^a) \end{aligned}$$

The final summation matches precisely with Equation (5.2), which defines the quantum Hartley transform of $|a\rangle$. Thus, we have successfully implemented the transformation:

$$|a\rangle \mapsto \mathsf{QHT}_N |a\rangle .$$

$$|0\rangle |a\rangle \mapsto |0\rangle \mathsf{QHT}_N |a\rangle .$$

The complete procedure is summarized in the following algorithm.

algorithm QHT_N

```
1. Initialize an ancilla qubit to 0 to obtain the state $|0\rangle |\psi\rangle$
2. Compute $\mathbf{1} \otimes \mathsf{QHT}_{N/2} \otimes \mathbf{1}$ recursively.
3. Apply $H \otimes \mathbf{1}$.
4. Apply the controlled negation $|0\rangle |y\rangle \mapsto |0\rangle |y\rangle, |1\rangle |y\rangle \mapsto |1\rangle |N/2 - y\rangle$ to the first two registers.
5. Apply the unitary $U_R$.
6. Apply $H \otimes \mathbf{1}$
7. Apply CNOT to the first and last qubits.
8. Apply $\mathbf{1} \otimes H$.
9. Trace out the first qubit
```

Theorem. Algorithm **??** is correct and can be implemented using $\approx \log^2 N + O(\log N)$ elementary gates.

proof. The correctness of the algorithm follows from the preceding discussion. Except for the unitary U_R and the negation unitary in Step 4, all steps of the algorithm can be implemented using $O(1)$ elementary gates. The negation operation in Step 4 can be realized using approximately $\lceil \log N \rceil$ elementary gates.

To implement the unitary U_R, which requires constructing the conditional rotation operator $R(y, b)$ for arbitrary y and b, we make use of the two-qubit controlled rotations:

$$R_j = |0\rangle \langle 0| \otimes \mathbf{1} + |1\rangle \langle 1| \otimes \begin{bmatrix} \cos\left(\frac{2\pi 2^j}{N}\right) & \sin\left(\frac{2\pi 2^j}{N}\right) \\ -\sin\left(\frac{2\pi 2^j}{N}\right) & \cos\left(\frac{2\pi 2^j}{N}\right) \end{bmatrix},$$

for $j = 0, 1, \ldots, n-1$. When $b = 0$, we have $R(y, 0) = \mathbf{1}$, and when $b = 1$, the operator $R(y, 1)$ is the product of those R_j for which the jth bit of y is 1. Therefore, for a binary string y of length $k = \lceil \log y \rceil$, we can implement U_R using at most k gates from the set $\{R_j\}$.

Let $T(N)$ denote the gate complexity of Algorithm **??** for an input of dimension N. Assuming access to the R_j gates as elementary operations, the recurrence relation becomes

$$T(N) \approx T(N/2) + 2 \log N + O(1).$$

Solving this recurrence yields $T(N) \approx \log^2 N + O(\log N)$, as claimed.

- The resulting complexity for an N-point QHT is $O(N \log N)$ basic quantum gates (up to polylog factors for precision), which is on par (up to constant factors) with the complexity of a QFT circuit. We also derived explicit constants for small cases to show improvements over prior work.

For the purposes of this thesis, the main point is: *we can implement the Hartley transform needed for our money scheme with polynomial efficiency.* Thus, switching to the Hartley transform does not introduce any prohibitive cost. All steps of minting and verification remain efficient. Having covered the construction of the money scheme and the necessary tools, we now proceed to the final piece:

the detailed verification algorithm using quantum walks, and an analysis of how it validates genuine banknotes and foils counterfeit attempts.

Chapter 6

Verification Algorithm Using Quantum Walks

The Hartley-based quantum money scheme introduced in the previous chapter requires a new verification procedure to address the issues that arise from using real amplitude states. In this chapter, we describe and analyze the verification algorithm, which leverages continuous-time quantum walks on the group action graph (and their efficient simulation from Chapter 4) to extract the information needed to authenticate a banknote.

6.1 Challenges with Naïve Verification

Before detailing the new algorithm, let us briefly recap why the straightforward approach fails. In the Fourier-based scheme, verification was done by a single "kickback" operation using the claimed serial h, and measuring an auxiliary register. In the Hartley-based scheme, if we attempted the analogous one-step verification, we would perform the controlled-h action and measure the auxiliary system. A genuine state $|\$_h\rangle$ (the Hartley money state for serial h) would cause some interference pattern in the auxiliary register, but unlike the Fourier case, it does not return the auxiliary to $|x_0\rangle$ deterministically. In fact, there is an ambiguity: certain superpositions of eigenstates corresponding to h and $-h$ (or other group-related variants) can produce the same measurement statistics in that one-step test. This means an adversary might prepare a counterfeit state that is not a legitimate $|\$_h\rangle$ but still passes the one-step verification with non-zero probability. Essentially, the Hartley transform being real means we lost some phase information, and a single measurement cannot distinguish some mirrored states.

To overcome this, our strategy is to perform a more complete measurement of the state's "phase spectrum." Instead of just one operation and measurement, we will use the quantum walk (with Hamiltonian $A = A(X, Q)$ as defined earlier) to perform a form of phase estimation.

6.2 Using Quantum Walks to Extract the Serial

Given a state of the form $\left|\mathbb{Z}_N^{(h)} * x\right\rangle_H$, we demonstrate how the value of h can be extracted using continuous-time quantum walks. For any $u \in \mathbb{Z}_N$, consider the Cayley graph $\Gamma = (\mathbb{Z}_N, E)$ generated by $Q = \{-u, u\}$. Let A denote the adjacency matrix of Γ. The eigenvectors of A are given by $\left|\mathbb{Z}_N^{(h)} * x\right\rangle$, with corresponding eigenvalues $\lambda_h = 2\cos(2\pi uh/N)$ for each $h \in \mathbb{Z}_N$. According to

Theorem **??** and the discussion that follows, the unitary evolution operator $W = e^{iAt}$ can be simulated efficiently to exponential precision. To proceed, we first state the following lemma.

lemma. The money state $\left|\mathbb{Z}_N^{(h)} * x\right\rangle_H$ is an eigenstate of W with eigenvalue $e^{i\lambda_h t}$.

proof. We have

$$\begin{aligned}
e^{iAt}\left|\mathbb{Z}_N^{(h)} * x\right\rangle_H &= \sum_{g\in\mathbb{Z}_N} e^{i\lambda_g t}\left|\mathbb{Z}_N^{(g)} * x\right\rangle\left\langle\mathbb{Z}_N^{(g)} * x\middle|\mathbb{Z}_N^{(h)} * x\right\rangle_H \\
&= \sum_{g\in\mathbb{Z}_N} e^{i\lambda_g t}\left|\mathbb{Z}_N^{(g)} * x\right\rangle\left\langle\mathbb{Z}_N^{(g)} * x\right|\left(\frac{1-i}{2}\left|\mathbb{Z}_N^{(h)} * x\right\rangle + \frac{1+i}{2}\left|\mathbb{Z}_N^{(-h)} * x\right\rangle\right) \\
&= e^{i\lambda_h t}\frac{1-i}{2}\left|\mathbb{Z}_N^{(h)} * x\right\rangle + \frac{1+i}{2}e^{i\lambda_{-h} t}\left|\mathbb{Z}_N^{(-h)} * x\right\rangle \\
&= e^{i\lambda_h t}\left|\mathbb{Z}_N^{(h)} * x\right\rangle_H,
\end{aligned}$$

where the second equality follows from the identity in (**??**), and the last equality follows from the fact that $\lambda_h = \lambda_{-h}$.

By setting $t = \text{poly}(\log N)$, Lemma 6.2 ensures that phase estimation can be performed using the unitary W and the eigenstate $\left|\mathbb{Z}_N^{(h)} * x\right\rangle_H$, yielding an approximation $\tilde{\lambda}_h$ of λ_h such that

$$\left|\tilde{\lambda}_h - \lambda_h\right| \leq \frac{1}{\text{poly}(\log N)}.$$

From this approximation, we can extract a real number $0 \leq \theta \leq 1$ satisfying

$$|\theta - \frac{uh}{N}| \leq \frac{1}{\text{poly}(\log N)}.$$

Since phase estimation can be applied with various choices of u, we can obtain multiple approximations of $\frac{uh}{N}$. As shown in [**?**], selecting u appropriately allows us to reconstruct h exactly from these estimates.

Bibliography

[1] S. Aaronson, “Quantum Copy-Protection and Quantum Money,” In Proc. of CCC, 2009.

[2] M. Zhandry, “Quantum Money from Modular Forms,” In Proc. of CRYPTO, 2022.

[3] A. M. Childs, “Quantum Information Processing in Continuous Time,” PhD Thesis, MIT, 2004.