Public-Key Quantum Money From Standard Assumptions

(In The Generic Model)

Jake Doliskani*

Abstract

Our main result is a quantum polynomial-time reduction from the group action discrete logarithm (DLP) problem to a specific cloning problem. A consequence of this result is that the public-key quantum money scheme proposed by Zhandry (2024), based on abelian group actions, is secure in the generic group action model. Specifically, our result shows that breaking the quantum money scheme is equivalent, under quantum polynomial-time reductions, to solving the group action DLP. Two immediate implications of our results are: i) A separation between quantum money and quantum lightning. This separation arises because our reduction is non-uniform, and quantum lightning is not secure against non-uniform adversaries. ii) Cloning vs. preparing Fourier states. Our main theorem shows that the problem of cloning group action Fourier states is equivalent to the problem of preparing these states.

*Department of Computing and Software, jake.doliskani@mcmaster.ca.

1 Introduction

Quantum money is the quantum-mechanical counterpart to classical money, where bills are represented by quantum states. Due to the fundamental no-cloning theorem, such bills cannot be counterfeited. The invention of quantum money was arguably the start of quantum information science. In a seminal paper, Wiesner [34] proposed the first quantum money scheme. In Wiesner's scheme, the bank is required to verify every bill, which means that the bank is involved in every transaction. This not only makes the scheme rather impractical but also leads to potential security problems. In modern terminology, Wiesner's scheme is referred to as a private-key money scheme. A desired scheme would allow anyone to verify a bill, while only the bank could create it. Such a scheme is known as *public-key* quantum money.

The first concrete construction of public-key quantum money was proposed by Aaronson [1], though it was later broken by Lutomirski et al. [23]. In recent years, several other proposals have been made [2, 16, 35, 19, 20, 21, 37]; however, each of these schemes has either been broken [12, 29, 5, 21], or relies on non-standard cryptographic assumptions. Despite these ongoing efforts, the construction of public-key quantum money based on standard cryptographic assumptions remains elusive.

Cryptographic group actions. The group action Discrete Logarithm Problem (DLP), a generalization of the classical group DLP, involves inverting the action: given two elements from the set being acted upon, find an element of the acting group that maps one to the other. The DLP is said to hold for a group action if no Quantum Polynomial Time (QPT) algorithm can solve the DLP for that action. Such group actions are referred to as cryptographic group actions. The concept of cryptographic group actions dates back to Brassard and Yung [7], though it was more formally defined by Couveignes [13]. The primary standard assumption in group action cryptography is the hardness of the group action DLP.

Quantum money from group actions. A promising candidate for public-key quantum money, based on abelian group actions, was proposed by Zhandry [37]. In Zhandry's scheme, the money states are group-action Fourier states, and the serial numbers are group elements. The verification is done through a group-action phase kickback unitary that recovers the serial number from the money state; See Section 3 for more details.

The security of Zhandry's scheme was proved in in both the generic model and the standard model, although based on new assumptions. In the generic model, security is proved under an assumption on the group action called the *Decisional 2X Assumption*. In the standard model, security is proved under two assumptions on the group action: the *Knowledge of Group Element Assumption* and the *Discrete Log with a Single Minimal CDH Query Assumption*. As mentioned in [37], these assumptions are new, and there are no known reductions from any standard assumptions to them.

This work. We prove that Zhandry's scheme is secure in the generic model. Specifically, we give a quantum polynomial time reduction from the group action DLP to the problem of cloning money states. To the best of our knowledge, this establishes the proposed group action quantum money scheme as the first public-key quantum money scheme based on standard assumptions in the generic model of group actions.

Our proof also yields additional noteworthy results. First, our proof demonstrates a computational separation between public-key quantum money and the quantum lightning scheme of [35] in

the following sense: *There exists a standard assumption under which quantum money is secure, but quantum lightning is not.* Quantum lightning is generally considered a stronger notion than quantum money. Informally, to break quantum money, an adversary must present a purported copy of a specific banknote, whereas, for quantum lightning, the adversary must present two copies of any banknote. Quantum lightning is clearly insecure against non-uniform adversaries with quantum advice, as such adversaries could have copies of certain banknotes hard-coded in their advice.

Another interesting result concerns the uniform superposition over an orbit of a given group action. Specifically, for a given group action, the problem is to efficiently prepare a uniform superposition over an orbit, which often consists of the entire set being acted upon. This problem is an important open question in cryptography, as highlighted by [21, 15, 6] in the context of isogeny graphs. A corollary of our main theorem is a quantum polynomial time reduction from the uniform superposition problem to the problem of cloning money states. This result is of independent interest beyond the quantum money literature, as both uniform superpositions and money states are examples of group-action Fourier states. Our result states that *given the ability to clone Fourier states, one can efficiently prepare any desired Fourier state.*

1.1 Technical Overview

In the following, we describe the main new ideas used in our proof. Let $(G, X, *)$ denote a group action, where the abelian group G acts on the set X via the operation $* : G \times X \to X$. Throughout this paper, we assume that G is of smooth order. Specifically, for any prime ℓ divisor of $|G|$, we require that $\ell \leq \text{poly}(\log|G|)$. The money states produced by the Gen algorithm of the quantum money scheme are of the form

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

where χ is a character of the group G.

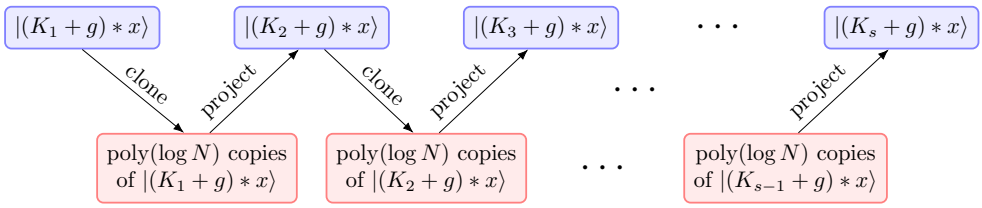
Projecting onto subgroups. Given a subgroup $K \leq G$ and a money state $|\psi\rangle$ over G, the projection algorithm is a QPT algorithm that outputs a money state over K. This process can be illustrated more clearly when $G = \mathbb{Z}_N$: for a subgroup $K \leq \mathbb{Z}_N$, the algorithm performs a certain measurement that produces the state

$$|(K + g)^{(r)} * x\rangle = \frac{1}{\sqrt{|K|}} \sum_{u \in K} \omega_N^{ur}|(u + g) * x\rangle, \tag{1}$$

where $0 \leq r < M$ is a random integer that is the outcome of the measurement.

Cloning subgroup states. Given a money state over a subgroup $K \leq G$, we use the adversary's algorithm to construct a cloning algorithm that can clone the money state with high probability and small error. Two key challenges in devising this cloning algorithm are: i) efficiently identifying elements of X that lie in the orbit of K (or one of its cosets), and ii) amplifying the cloning probability of the adversary o address the first challenge, we employ the random injection technique introduced by Zhandry in [37]. To tackle the second challenge, we use the amplification algorithm developed by Marriott and Watrous [24, 32].

Fixing the phase. The subgroup money states (1) obtained through the projection algorithm are not immediately useful for extracting information about g due to the presence of the phase ω_N^{ur}. To eliminate this phase, we utilize a composition series $0 < K_1 < \cdots < K_{s-1} < K_s = K$ of the subgroup K. The procedure begins by repeatedly projecting onto the subgroup K_1 until he outcome $r = 0$ is achieved.[1] In this case, the resulting state is $|(K_1 + g) * x\rangle = |K_1|^{-1/2} \sum_{u \in K_1} |(u + g) * x\rangle$ which is phase-free. From here, we use an algorithm similar to the projection algorithm to obtain the state $|(K_2 + g) * x\rangle = |K_2|^{-1/2} \sum_{u \in K_2} |(u + g) * x\rangle$. Continuing this process, we ultimately obtain the state $|(K + g) * x\rangle = |K|^{-1/2} \sum_{u \in K} |(u + g) * x\rangle$ which is the same as (1), but without the phase. Since the algorithm used to move from the subgroup K_j to the subgroup K_{j+1} is probabilistic, we require multiple copies of the state in each stage. This is where our cloning algorithm is applied. The sequence of projection and cloning steps is summarized in the following diagram.



Putting it all together. Given a pair $(x, g * x)$, where $g \in G$ is unknown, we can use the tools we have developed to recover g. We first consider the case where $G = \mathbb{Z}_N$ and later explain how the algorithms can be extended to handle a general abelian group G. Let $G = \mathbb{Z}_N$, and let $\ell \mid N$ be prime. We can efficiently prepare the state $|(\ell\mathbb{Z}_N + g) * x\rangle$ by starting from the state $|g * x\rangle$, and the state $|\ell\mathbb{Z}_N * x\rangle$ by starting from the state $|x\rangle$. From these states, we can compute g mod ℓ by shifting the latter state to obtain the states $|(\ell\mathbb{Z}_N + t) * x\rangle$ for different values of t, and then using the SWAP-TEST against the former state. It is important to note that multiple copies of the two states are required to perform the SWAP-TEST for various shifts of t. These copies are generated using our cloning algorithm. Once we have the ability to compute g mod ℓ for arbitrary prime divisors $\ell \mid N$, we can employ a group action variant of the Pohlig-Hellman algorithm to fully recover g.

2 Preliminaries

2.1 Group actions

We follow the presentation in [3] for the definition of cryptographic group actions. Abstractly, the action of a group G on a set X is a mapping $* : G \times X \to X$ that satisfies the following properties:

1. Compatibility: for every $a, b \in G$ and every $x \in X$, $g * (h * x) = (gh) * x$,
2. Identity: for the identity $1 \in G$ and every $x \in X$, $1 * x = x$.

We use the notation $(G, X, *)$ for a group G acting on a set X through $*$. A group action $(G, X, *)$ is said to be transitive if for every $x, y \in X$ there exists a $g \in G$ such that $g * x = y$. A group action $(G, X, *)$ is said to be free if for every $g \in G$, $g = 1$ if and only if there is an $x \in X$ such that $g * x = x$. A group action that is both transitive and free is called *regular*. Equivalently, a group action is regular if for every $x, y \in X$ there exists a unique $g \in G$ such that $g * x = y$. For a group action $(G, X, *)$ to be computationally useful, the group G, the set X, and the action $*$ must satisfy certain properties. This leads to the definition of *effective group actions*.

[1] We have assumed $r = 0$ in the present discussion for clarity. However, the algorithm works for any value of r.

Definition 2.1 (Effective Group Action)**.** A group action $(G, X, *)$ is said to be effective if it satisfies the following properties:

1. Both G and X are finite.
2. There are efficient algorithms for the following operations in G: membership testing, equality testing, sampling (close to) uniform elements, group operation and inversion.
3. There are efficient algorithms for the following in X: membership testing and unique representation.
4. There exists a distinguished element $x \in X$ with known bit-string representation.
5. There is exists an efficient algorithm for the mapping $*$: given any $g \in G$ and any $x \in X$, the algorithm outputs $g * x$.

In this paper, we assume that all group actions are effective. We will also (implicitly) work with regular groups actions. If a group action $(G, X, *)$ is free and $x \in X$ is the distinguished element, then the group action $(G, G * x, *)$, where $G * x$ is the orbit of x, is regular. The group action discrete logarithm problem is defined as follows:

Definition 2.2 (Group Action DLP)**.** Let $(G, X, *)$ be an effective group action. Given a pair $(x, g * x)$, where $g \in G$, the discrete logarithm problem is to compute g.

We say that the DLP assumption holds for $(G, X, *)$ if no QPT algorithm can solve the DLP with respect to $(G, X, *)$. A group action for which the DLP assumption holds is called a *cryptographic group action*. In the literature, cryptographic group actions often rely on other assumptions beyond the DLP assumption to ensure security and functionality.

2.2 Generic Group Actions

The generic group action model (GGAM) is a an adaptation of the generic group model to group actions. In this paper, we adopt the definitions from [37, 36], which are themselves adaptations of the definitions in [30] given for the generic group model. Let $(G, X, *)$ be an effective group action, and let $m \geq \log|G|$ be an integer. Let $L : G \to \{0,1\}^m$ be a random injection, called the labeling function. For a fixed $x \in X$, we typically set $L(0) = x$ so that $L(g) = g * x$ for all $g \in G$. This implies that $L(G) = X$ when the action is regular. In the GGAM, access to the group action is provided in a black-box manner. Specifically:

1. The element $x = L(0)$ is public, and
2. There is an oracle $\mathsf{GGAM}_{G,m}$ that on input $(y, g) \in \{0,1\}^m \times G$ returns $L(g + L^{-1}(y))$ if $y \in \mathrm{Im}(L)$; otherwise it returns $\perp$.

We will refer to the queries $(y, g) \in \{0,1\}^m \times G$ made to $\mathsf{GGAM}_{G,m}$ as group action queries.

The random injection technique. The following technique for simulating a "strict" group action oracle using another "less strict" group action oracle was introduced in [37]. We will use this technique for cloning subgroup money states.

Let $\mathcal{F}$ be a family of functions $f : Y \to Y$ for some set Y, and let $y_0 \in Y$ be fixed. Let $L : Y \to \{0,1\}^{m'}$ be a random injection, and let $\mathcal{O} : \{0,1\}^{m'} \times \mathcal{F} \to \{0,1\}^{m'}$ be an oracle defined as follows: if $x \in \mathrm{Im}(L)$ then $\mathcal{O}(x, f) = L(f(L^{-1}(x)))$, otherwise $\mathcal{O}(x, f) = \perp$.

Now let $\Gamma : Y \to \{0,1\}^m$ be another injection, and let $P : \{0,1\}^m \times \mathcal{F} \to \{0,1\}^m$ be an oracle defined as follows: for $x \in \mathrm{Im}(\Gamma)$, $P(x,f) = \Gamma(f(\Gamma^{-1}(x)))$, but for $x \notin \mathrm{Im}(\Gamma)$, P may output other values than $\perp$. The goal is to simulate $\mathcal{O}$ using P. To do this, we choose a random injection $\Pi : \{0,1\}^m \to \{0,1\}^{m'}$ and construct an oracle $\mathcal{O}' : \{0,1\}^{m'} \times \mathcal{F} \to \{0,1\}^{m'}$ as follows: if $x \in \mathrm{Im}(\Pi)$ then $\mathcal{O}'(x,f) = \Pi(P(\Pi^{-1}(x), f))$, otherwise $\mathcal{O}'(x,f) = \perp$. Consider any algorithm $\mathcal{A}^{\mathcal{O}}$ with input $L(y_0)$ and superposition access to $\mathcal{O}$, and let $\mathcal{A}^{\mathcal{O}'}$ be the same algorithm but with input $\Pi(\Gamma(y_0))$, and superposition access to $\mathcal{O}'$ instead. The following lemma proves that the output distributions of $\mathcal{A}^{\mathcal{O}}$ and $\mathcal{A}^{\mathcal{O}'}$ are indistinguishable.

Lemma 2.3 ([37]). *Let $Y, \mathcal{F}, \Gamma$ and $y_0 \in Y$ be as above. Assume $m' \geq m + t$ for some integer $t > 0$. Then for any algorithm $\mathcal{A}$ that makes q queries to its oracle*

$$\left| \Pr\left[\mathcal{A}^{\mathcal{O}}(L(y_0)) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}'}(\Pi(\Gamma(y_0))) = 1\right] \right| \leq O(2^{-t/2}q)$$

Here, L and Π are random injections using which $\mathcal{O}$ and $\mathcal{O}'$ are derived as above. The probabilities are over the random choices of L and Π and the randomness of $\mathcal{A}$.

2.3 Quantum Computation

A finite Hilbert space $\mathcal{H}$ of dimension N is a complex Euclidean space, isomorphic as a $\mathbb{C}$-vector space to $\mathbb{C}^N$. An N-dimensional quantum state is a unit vector $|\psi\rangle \in \mathcal{H}$, where $|\cdot\rangle$ denotes the Dirac notation. In practice, we usually use specific bases for a Hilbert space. For instance, when considering a finite group G, we work with the Hilbert space $\mathcal{H} = \mathbb{C}^G$, which is a $|G|$-dimensional space spanned by the basis $\{|g\rangle : g \in G\}$.

Quantum Fourier transform. Let G be an abelian group. The set of characters of G, denoted by $\hat{G}$, are the set of homomorphisms $\chi(a,\cdot) : G \to \mathbb{C}$ where $a \in G$. If $G \cong \mathbb{Z}_{N_1} \oplus \cdots \oplus \mathbb{Z}_{N_k}$ then the character $\chi(a,\cdot)$ can be explicitly written as

$$\chi(a,x) = \omega_{N_1}^{a_1 x_1} \cdots \omega_{N_k}^{a_k x_k}$$

where $\omega_M = \exp(2\pi i/M)$ is a primitive M-th root of unity. The Fourier transform of a function $f : G \to \mathbb{C}$ is given by

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

The quantum Fourier transform of a (normalized) state $\sum_{g \in G} f(g)|g\rangle$ is given by $\sum_{x \in G} \hat{f}(x)|x\rangle$. For a regular group action $(G, X, *)$, any subset $S \subseteq G$, any $y \in X$, and any $h \in G$, we define

$$|S^{(h)} * y\rangle = \frac{1}{\sqrt{|S|}} \sum_{g \in S} \chi(g,h)|g * y\rangle. \tag{2}$$

There are two orthonormal bases of the space $\mathbb{C}^X$. One basis is $\{|x\rangle : x \in X\}$. For a fixed element $x \in X$, this basis is the same as $\{|g * x\rangle : g \in G\}$, which follows from the fact that the action is regular and thus $|X| = |G|$. The other basis is given by the states

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

These state are simultaneous eigenstates of the group action operation. Specifically, for the unitary $U_k : |y\rangle \mapsto |k * y\rangle$, where $k \in G$, we have $U_k|G^{(h)} * x\rangle = \chi(-k,h)|G^{(h)} * x\rangle$. These states resemble the set of Fourier states over the abelian group G. We will also sometimes refer to them as *Fourier states*.

Non-uniform algorithms. A non-uniform quantum algorithm is modeled as a family of quantum circuits, where each circuit is specific to inputs of a given size. Similar to the classical model of non-uniform algorithms, the quantum model allows each algorithm to take advantage of pre-supplied advice, which may be either classical or quantum. In this work, a non-uniform quantum algorithm with quantum advice is modeled as a family $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ of quantum algorithms such that:

- $\mathcal{A}_\lambda$ takes inputs of size λ and a fixed quantum state $|\psi_\lambda\rangle$, referred to as quantum advice.
- There exists a polynomial function $f : \mathbb{N} \to \mathbb{N}$ such that, for every $\lambda \in \mathbb{N}$, the description length of $\mathcal{A}_\lambda$ is bounded by $f(\lambda)$.
- There exists a polynomial function $g : \mathbb{N} \to \mathbb{N}$ such that, for every $\lambda \in \mathbb{N}$, the running time of $\mathcal{A}_\lambda$ is bounded by $g(\lambda)$.

A key assumption in this paper is that the adversary is modeled as a non-uniform quantum algorithm.

The $\mathsf{cmpIndex}$ algorithm. Given a state $|G^{(h)} * x\rangle$, there is an efficient algorithm for computing h. Specifically, there is a unitary operator that performs the transformation $|G^{(h)} * x\rangle|0\rangle \mapsto |G^{(h)} * x\rangle|h\rangle$ using the *phase kickback* technique. To see this, start with the state $|G^{(h)} * x\rangle|0\rangle$, apply the quantum Fourier transform to the second register, and then apply the unitary $\sum_{k\in G} U_k \otimes |k\rangle\langle k|$ to both registers. This results in the state

$$\frac{1}{\sqrt{|G|}} \sum_{k\in G} |G^{(h)} * x\rangle \chi(-k, h)|k\rangle.$$

Finally, applying the inverse quantum Fourier transform to the second register yields $|G^{(h)} * x\rangle|h\rangle$.

3 The Quantum Money Scheme

In this section, we briefly review the quantum money scheme introduced in [37]. A public-key quantum money scheme consists of two QPT algorithms:

- $\mathsf{Gen}(1^\lambda)$ takes a security parameter as input. It outputs a pair (s, ρ_s), where s is a binary string called the serial number, and ρ_s is a quantum state called a banknote. The pair (s, ρ_s), or just ρ_s, is sometimes denoted by \$.
- $\mathsf{Ver}(s, \rho_s)$ takes a serial number and an alleged banknote as input. It outputs either 1 (accept) or 0 (reject).

A quantum money scheme is said to be *correct* if the genuine banknotes generated by Gen are accepted by Ver. More precisely,

$$\Pr[\mathsf{Ver}(s, \rho_s) = 1 : (s, \rho_s) \leftarrow \mathsf{Gen}(1^\lambda)] \geq 1 - \mathrm{negl}(\lambda).$$

where the probability is taken over the randomness of Gen and Ver. The money scheme $(\mathsf{Gen}, \mathsf{Ver})$ is said to be secure if, given a genuine bill (s, ρ_s), there is no QPT algorithm $\mathcal{A}$ that can prepare two (possibly entangled) bills (s, ρ_1) and (s, ρ_2) that are both accepted by Ver with non-negligible probability. More precisely,

$$\Pr\left[\mathsf{Ver}(s, \rho_1) = \mathsf{Ver}(s, \rho_2) = 1 : \begin{smallmatrix}(s,\rho_s)\leftarrow\mathsf{Gen}(1^\lambda)\\(\rho_1,\rho_2)\leftarrow\mathcal{A}(s,\rho_s)\end{smallmatrix}\right] \leq \mathrm{negl}(\lambda).$$

Construction 3.1. Let $\{(G_\lambda, X_\lambda, *)\}_{\lambda \in J}$, where $J \subset \mathbb{N}$, be a collection of effective regular group actions for abelian groups G_λ, and let $x_\lambda \in X_\lambda$ be fixed. The quantum money scheme of [37] is as follows:

- $\mathsf{Gen}(1^\lambda)$. Start with the state $|0\rangle|x_\lambda\rangle$ and apply the quantum Fourier transform over G_λ to the first register to obtain the superposition

$$\frac{1}{\sqrt{|G_\lambda|}} \sum_{g \in G_\lambda} |g\rangle|x_\lambda\rangle.$$

  Next, apply the unitary $|h\rangle|y\rangle \mapsto |h\rangle|h * y\rangle$ to the above state, and then apply the quantum Fourier transform to the first register. This yields

$$\frac{1}{|G_\lambda|} \sum_{h \in G_\lambda} \sum_{g \in G_\lambda} \chi(g,h)|h\rangle|g * x_\lambda\rangle = \frac{1}{\sqrt{|G_\lambda|}} \sum_{h \in G_\lambda} |h\rangle|G^{(h)} * x_\lambda\rangle$$

  where $|G^{(h)} * x_\lambda\rangle$ is defined as in (2). Measure the first register to obtain a random $h \in G_\lambda$ collapsing the state to $|G^{(h)} * x_\lambda\rangle$. Return the pair $(h, |G^{(h)} * x_\lambda\rangle)$.

- $\mathsf{Ver}(h, |\psi\rangle)$. First, check if $|\psi\rangle$ has support in X_λ. If not, return 0. Next, apply $\mathsf{cmpIndex}$ to the state $|\psi\rangle|0\rangle$, then measure the second register to obtain some $h' \in G_\lambda$. If $h' = h$, return 1; otherwise return 0.

For the rest of the paper, unless otherwise stated, we make the security parameter λ implicit in the notation and use G for G_λ, X for X_λ, and so on. The above scheme is clearly correct, since a genuine pair $(h, |G^{(h)} * x\rangle)$ generated by Gen passes verification with probability 1. The following theorem asserts that any state $|\psi\rangle$ accepted by the verifier must have non-negligible "overlap" with the given banknote.

Theorem 3.2 ([37])**.** *Let $|\psi\rangle$ be a state with support in X. Then*

$$\Pr[\mathsf{Ver}(h, |\psi\rangle) = 1] = \|\langle\psi|G^{(h)} * x\rangle\|^2.$$

Furthermore, if the verifier accepts, the post-verification state is exactly $|G^{(h)} * x\rangle$.

It is crucial for the Ver algorithm to be able to recognize the elements of X. In the above scheme, this is guaranteed by the assumption that the group action $(G, X, *)$ is effective. If Ver cannot efficiently recognize X, for example, if X is a subset of a larger set Y and there is no known algorithm that can efficiently distinguish X from the rest of Y, the adversary can create counterfeit banknotes that are accepted by Ver with non-negligible probability. An example of such a scenario is discussed in Section 4.2.

4 The New Proof of Security

In this section, we provide the details of our security proof for the quantum money scheme. Let $(G, X, *)$ be a regular group action, and let $x \in X$ be fixed. Given a pair $(x, g * x)$ with $g \in G$ unknown, the goal is to show that given an efficient algorithm for cloning money states in Construction 3.1, there is an efficient algorithm for computing g. We first consider the case where $G = \mathbb{Z}_N$, and then extend the proof to general abelian groups.

4.1 Projecting Onto Subgroup Money States

In this section, we show how to project the classical states $|g * x\rangle$, where $g \in \mathbb{Z}_N$, onto money states over any given subgroup $K \leq G$. This is done using the following general algorithm.

Algorithm 1 (Subgroup Projection)**.**

Input: Regular group action $(\mathbb{Z}_N, X, *)$. A state $|\psi\rangle$ supported on X. A nontrivial subgroup $K \leq \mathbb{Z}_N$.

Output: A specific projection of $|\psi\rangle$ with support in X

1. Run cmpIndex on the state $|\psi\rangle|0\rangle$. If we write $|\psi\rangle = \sum_{h\in\mathbb{Z}_N} \alpha_h |\mathbb{Z}_N^{(h)} * x\rangle$, the resulting state is $|\psi\rangle = \sum_{h\in\mathbb{Z}_N} \alpha_h |\mathbb{Z}_N^{(h)} * x\rangle|h\rangle$.
2. Let $M = |K|$. Then K is the unique subgroup $K = (N/M)\mathbb{Z}_N$.
3. Compute $(N/M)h$ into an extra register to obtain the state
$$\frac{1}{\sqrt{N}} \sum_{h\in\mathbb{Z}_N} \alpha_h |\mathbb{Z}_N^{(h)} * x\rangle|h\rangle|(N/M)h\rangle.$$
4. Measure the last register to obtain the state
$$\frac{\sqrt{M}}{\sqrt{N}} \sum_{h\in M\mathbb{Z}_N+r} \alpha_h |\mathbb{Z}_N^{(h)} * x\rangle|h\rangle.$$
for some $0 \leq r < M$. Here, r is known from the measurement outcome. Specifically, the measurement outcome is some $r' = (N/M)h'$, from which we compute a unique $r = h' < M$.
5. Uncompute the last register using cmpIndex, and return the resulting state.

The effect of Algorithm 1 on a general state $|\psi\rangle$ is not immediately clear. Expanding the state returned by the algorithm, we have

$$\begin{aligned}\frac{\sqrt{M}}{\sqrt{N}} \sum_{h\in M\mathbb{Z}_N+r} \alpha_h |\mathbb{Z}_N^{(h)} * x\rangle &= \frac{\sqrt{M}}{N} \sum_{u\in\mathbb{Z}_N} \sum_{h\in M\mathbb{Z}_N+r} \alpha_h \omega_N^{uh} |u * x\rangle \\ &= \frac{M}{N} \sum_{u\in\mathbb{Z}_N} \omega_N^{ur} \hat{\beta}(u) |u * x\rangle\end{aligned}$$

where $\beta : M\mathbb{Z}_N \to \mathbb{C}$ is the function defined by $\beta(a) = \alpha_{a+r}$, and $\hat{\beta}$ is its Fourier transform over $M\mathbb{Z}_N$. In particular, β is the shifted restriction of the function $\alpha : \mathbb{Z}_N \to \mathbb{C}$, $\alpha(h) = \alpha_h$. The duality of the Fourier transform on abelian groups suggests that for "nice" functions α, we will obtain with a state supported on a coset orbit $(K + t) * x$ for some $t \in \mathbb{Z}_N$. This is indeed the case when $|\psi\rangle$ is a basis state $|g * x\rangle$ where $g \in \mathbb{Z}_N$. Since these states are of particular interest in this paper, we carry out the calculation of their projections in the following.

Lemma 4.1. *Let $(\mathbb{Z}_N, X, *)$ be a regular group action and let $K \leq \mathbb{Z}_N$ be a subgroup of order M. On input $(|g * x\rangle, K)$, Algorithm 1 outputs the state $|(K + t)^{(r)} * x\rangle$, where $r \in \mathbb{Z}_M$ is uniformly random and $t = g \bmod K$.*

Proof. We start by writing

$$|g * x\rangle = \frac{1}{\sqrt{N}} \sum_{h\in\mathbb{Z}_N} \omega_N^{-gh} |\mathbb{Z}_N^{(h)} * x\rangle.$$

Applying cmpIndex to $|g * x\rangle|0\rangle$ and computing $(N/M)h$ into an extra register gives

$$\frac{1}{\sqrt{N}} \sum_{h \in \mathbb{Z}_N} \omega_N^{-hg} |\mathbb{Z}_N^{(h)} * x\rangle |h\rangle |(N/M)h\rangle.$$

Measuring the last register gives

$$\frac{\sqrt{M}}{\sqrt{N}} \sum_{h \in M\mathbb{Z}_N + r} \omega_N^{-hg} |\mathbb{Z}_N^{(h)} * x\rangle |h\rangle$$

for a uniformly random $0 \leq r < M$. Now, uncompute the second register using cmpIndex to obtain

$$\begin{aligned}
\frac{\sqrt{M}}{\sqrt{N}} \sum_{h \in M\mathbb{Z}_N + r} \omega_N^{-hg} |\mathbb{Z}_N^{(h)} * x\rangle &= \frac{\sqrt{M}}{N} \sum_{u \in \mathbb{Z}_N} \sum_{h \in M\mathbb{Z}_N + r} \omega_N^{-hg} \omega_N^{uh} |u * x\rangle \\
&= \frac{\sqrt{M}}{N} \sum_{u \in \mathbb{Z}_N} \omega_N^{(u-g)r} \sum_{h \in M\mathbb{Z}_N} \omega_N^{(u-g)h} |u * x\rangle \\
&= \frac{1}{\sqrt{M}} \sum_{\substack{u \in \mathbb{Z}_N \\ u \equiv g \bmod (N/M)}} \omega_N^{(u-g)r} |u * x\rangle. \\
&= \frac{1}{\sqrt{|K|}} \sum_{u \in K} \omega_N^{ur} |(u+g) * x\rangle \\
&= |(K+g)^{(r)} * x\rangle.
\end{aligned}$$

Therefore, Algorithm 1 projects the classical state $|g*x\rangle$ onto a coset state of the form $|(K+g)^{(r)}*x\rangle$ for a random $r \in \mathbb{Z}_M$. Note that for $t = g \bmod K$, this state is the same as the state $|(K+t)^{(r)} * x\rangle$ up to global phase. □

Let $y = t * x$ and $Y = K * y$. If we consider $y \in Y$ as the fixed element, then $|(K+t)^{(r)} * x\rangle = |K^{(r)} * y\rangle$ is a money state with respect to the group action $(K, Y, *)$. Finally, according to Lemma 4.1, running Algorithm 1 on the input $(|x\rangle, K)$ results in the state $|K^{(r)} * x\rangle$ for a uniformly random $r \in \mathbb{Z}_M$.

4.2 Cloning Subgroup Money States

As mentioned in Section 2.3, the adversary is modeled as a family of quantum algorithms $\{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ where $\mathcal{A}_\lambda$ handles inputs of length λ. Let K be a subgroup of $\mathbb{Z}_N$, and let $\mathcal{A} \in \{\mathcal{A}_\lambda\}$ be the algorithm that clones money states with respect to the group action $(K, K * y, *)$, where $y \in X$ is fixed. The (possibly non-unitary) quantum operation carried out by $\mathcal{A}$ can be described as follows: on input $|K^{(r)} * y\rangle|0\rangle$, where

$$|K^{(r)} * y\rangle = \frac{1}{\sqrt{|K|}} \sum_{u \in K} \omega_N^{ur} |u * y\rangle \tag{3}$$

is a money state over K, the output of $\mathcal{A}$ is a state $|\psi\rangle$, which purportedly consists of two (possibly entangled) copies of $|K^{(r)} * y\rangle$.

Ideally, we would like to obtain a state very close to $|K^{(r)} * y\rangle|K^{(r)} * y\rangle$ from the output of $\mathcal{A}$, enabling us to generate arbitrarily many copies of $|K^{(r)} * y\rangle$ without concerns about handling approximations or errors. To achieve this, we must address two key challenges: i) How do we ensure

that both registers of the state $|\psi\rangle$ have supports in $K * y$? and ii) Assuming both registers of $|\psi\rangle$ have support in $K * y$, how do we then obtain $|K^{(r)} * y\rangle|K^{(r)} * y\rangle$? We address these two problems separately in this section and the next.

In the first problem, the challenge arises from the fact that, although the group action $(K, K * y, *)$ is regular, it is not effective. This is because we do not know how to recognize the elements of $K * y$, i.e., how to distinguish $K * y$ from the rest of X. As a result, some states that are not close to $|K^{(r)} * y\rangle$ can be accepted by Ver. For instance, consider the state $|(K+s)^{(r)} * y\rangle = |K|^{-1/2} \sum_{u \in K} \omega_N^{ur} |(u+s) * y\rangle$ for some $s \in \mathbb{Z}_N$. Although this state is supported on $(K+s) * y$, the Ver algorithm is completely oblivious to this fact and accepts the state with probability 1. This means that, on input $|K^{(r)} * y\rangle$, the adversary $\mathcal{A}$ could return two copies of $|(K+s)^{(r)} * y\rangle$, or even $(|K^{(r)} * y\rangle + |(K+s)^{(r)} * y\rangle)/\sqrt{2}$, as legitimate clones of the input. Therefore, we cannot expect to obtain high-fidelity copies of $|K^{(r)} * y\rangle$ by relying solely on Ver. To address this issue, we employ the random injection technique from [37]. The idea is to construct an encoding of the elements of X such that, given initial access to elements of $K * y$, the adversary has a negligible chance of computing elements in $X \setminus K * y$.

Before stating the result of this section, we first clarify the notion of the success probability of the adversary $\mathcal{A}$. Consider $\mathcal{A}$ with respect to the generic group action $\mathsf{GGAM}_{K,m'}$. Let $(K, Y, *)$ be an effective group action, where $Y = L(K)$ for some injection $L : K \to 0,1^{m'}$, and let $y = L(0)$ be fixed. Given a banknote $(r, |K^{(r)} * y\rangle)$, the output of $\mathcal{A}$ is a pair of registers $\mathsf{X}_1\mathsf{X}_2$ containing two alleged copies of the money state $|K^{(r)} * y\rangle$. This output can be wriiten as

$$\alpha |K^{(r)} * y\rangle|K^{(r)} * y\rangle + |\Phi_1\rangle$$

for some $\alpha \in \mathbb{C}$ and some state $|\Phi_1\rangle$ that is orthogonal to $|K^{(r)} * y\rangle|K^{(r)} * y\rangle$, with $|\Phi_1\rangle$ supported on Y. We say that the cloning success probability of $\mathcal{A}$ is p if $|\alpha|^2 = p$. In this case, the output can be written as

$$\sqrt{p}\,|K^{(r)} * y\rangle|K^{(r)} * y\rangle + \sqrt{1-p}\,|\Phi_1\rangle,$$

where $|\Phi_1\rangle$ is a normalized state orthogonal to $|K^{(r)} * y\rangle|K^{(r)} * y\rangle$. Algorithmically, this is equivalent to saying that if we run Ver on both X_1 and X_2 from the output of $\mathcal{A}$, the probability of returning two "accept" outcomes is p.

Lemma 4.2. *Assume $X \subset \{0,1\}^m$, and let $y \in X$ be fixed. Let $m' \geq m + \omega(\log \lambda)$. Let $\mathcal{A}$ be an adversary for the quantum money security with respect to the generic group action $\mathsf{GGAM}_{K,m'}$. Suppose the success probability of $\mathcal{A}$ for cloning a banknote with serial number r is p. Then, there exists a quantum algorithm $\mathcal{B}$ that can clone the money state $|K^{(r)} * y\rangle$ with a success probability negligibly close to p. The algorithm B uses one call to $\mathcal{A}$ and $\mathrm{poly}(\log N)$ additional operations.*

Proof. Define the injection $\Gamma : K \to X$ by $\Gamma(a) = a * y$, and for any $h \in K$, define the oracle $P(\Gamma(a), h) = h * \Gamma(a)$. Note that P also acts on elements that are not in the image of Γ. Choose a random injection $\Pi : \{0,1\}^m \to \{0,1\}^{m'}$, and define an oracle $\mathcal{O}'$ as follow: if z is in the image of Π then $\mathcal{O}'(z,h) = \Pi(P(\Pi^{-1}(z), h))$; otherwise $\mathcal{O}'(z,h) = \perp$. The adversary $\mathcal{B}$ is given $\Pi(\Gamma(0)) = \Pi(y)$ and superposition access to $\mathcal{O}'$.

To clone the state in (3), $\mathcal{B}$ runs the adversary $\mathcal{A}$ on the input state $|K^{(r)} * y\rangle$, where $\mathcal{A}$'s queries $(z,h) \in \{0,1\}^{m'} \times K$ to the group action are simulated using $\mathcal{O}'$. Let X_1 and X_2 be the two registers containing the output of $\mathcal{A}$. By Lemma 2.3, the combined state of the registers $\mathsf{X}_1\mathsf{X}_2$ is negligibly close to the state

$$\sqrt{p}\,|K^{(r)} * y\rangle|K^{(r)} * y\rangle + \sqrt{1-p}\,|\Phi_1\rangle,$$

where $|\Phi_1\rangle$ is a normalized state supported on $K * y$ and orthogonal to $|K^{(r)} * y\rangle|K^{(r)} * y\rangle$. □

4.3 Amplifying The Cloning Probability

Let $(r, |\psi\rangle)$ be a given banknote, where $|\psi\rangle = |K^{(r)} * y\rangle$, and $y \in X$ is known. Let $Y = K * y$, so, $|\psi\rangle$ is supported on Y. Let $\mathcal{A}$ be an adversary that on input $(r, |\psi\rangle, y)$, outputs a state of the form

$$\sqrt{p}\,|\psi\rangle|\psi\rangle + \sqrt{1-p}\,|\Phi_1\rangle,$$

where $p > 0$ is a real number, and $|\Phi_1\rangle$ is a normalized state that is supported on Y and is orthogonal to $|\psi\rangle|\psi\rangle$. The second problem, mentioned in the beginning of Section 4.2, can be stated more precisely as follows: Given access to $\mathcal{A}$, compute a state negligibly close to $|\psi\rangle|\psi\rangle$.

To solve this problem, we use the amplification technique first proposed by Marriott and Watrous [24, 32]. Abstractly, this technique works by repeatedly applying two carefully designed measurements in an alternating fashion until a desired output is obtained. These measurements are constructed based on the states obtained by applying (or rewinding) the adversary $\mathcal{A}$.

Suppose that $\mathcal{A}$ performs all measurements at the end. In this case, we can model the action of $\mathcal{A}$ as a unitary operator Q, which does not perform the final measurement. We extend this action to a larger Hilbert space $\mathcal{X}$ by introducing an indicator qubit and k workspace qubits. The action of Q on the state $|\psi\rangle|0^m\rangle|0\rangle|0^k\rangle \in \mathcal{X}$ is given by

$$Q|\psi\rangle|0^m\rangle|0\rangle|0^k\rangle = \alpha(\sqrt{p}\,|\psi\rangle|\psi\rangle + \sqrt{1-p}\,|\Phi_1\rangle)|\phi\rangle|0^{\ell+1}\rangle + \beta|\Phi_2\rangle|0^{\ell+1}\rangle,$$

where

- $\alpha, \beta \in \mathbb{C}$ satisfy $|\alpha|^2 + |\beta|^2 = 1$,
- $|\phi\rangle$ is a normalized state represented using $k - \ell$ qubits,
- $|\Phi_2\rangle$ is a normalized such that $(\mathbb{1} \otimes |\phi\rangle\langle\phi| \otimes \mathbb{1})|\Phi_2\rangle = 0$.

The final measurement done by $\mathcal{A}$ is described by $\{\Pi_{|\phi\rangle}, \mathbb{1} - \Pi_{|\phi\rangle}\}$, where $\Pi_{|\phi\rangle} = \mathbb{1} \otimes |\phi\rangle\langle\phi| \otimes \mathbb{1}$. For simplicity, and without loss of generality, we write the action of Q as

$$Q|\psi\rangle|0^{m+k+1}\rangle = \sqrt{p}\,|\psi\rangle|\psi\rangle|\phi\rangle|0^{\ell+1}\rangle + \sqrt{1-p}\,|\Phi_3\rangle|0^{\ell+1}\rangle, \tag{4}$$

for some $p \geq 1/\operatorname{poly}(\log N)$ and some normalized state $|\Phi_3\rangle$ that is orthogonal to the state $|\Psi\rangle = |\psi\rangle|\psi\rangle|\phi\rangle$. This is because the phase of α can be absorbed into $|\phi\rangle$, and the new value p is defined as $|\alpha|^2 p$.

We need to implement a measurement described by $\{|\Psi\rangle\langle\Psi| \otimes \mathbb{1}, \mathbb{1} - |\Psi\rangle\langle\Psi| \otimes \mathbb{1}\}$. To do this, we use the indicator qubit as follows. Let T be a unitary that transform the state $|\Psi\rangle|0\rangle$ to $|\Psi\rangle|1\rangle$ when th first two registers are in the states $|\psi\rangle|\psi\rangle$, and leaves $|\Psi\rangle|0\rangle$ unchanged otherwise. In other words, T set the indicator qubit to 1 precisely when the first two registers contain two copies of $|\psi\rangle$. This unitary can be efficiently implemented using the cmpIndex unitary as follows: compute the serial numbers of first and second registers, set the indicator qubit to 1 if the serials numbers are both equal to r, and then uncompute the serial numbers.

Define the projection $\Pi_1 = \mathbb{1} \otimes |1\rangle\langle 1| \otimes \mathbb{1}$. The measurement described by $\{\Pi_1, \mathbb{1} - \Pi_1\}$ measures the indicator qubit in the computational basis. Next, we assume that we have access to an efficient implementation of the measurement $\{\Pi_{|\phi\rangle}, \mathbb{1} - \Pi_{|\phi\rangle}\}$, as it is implemented by $\mathcal{A}$ [2]. The projection $\Pi_1\Pi_{|\phi\rangle}$ projects onto states where the third register is $|\phi\rangle$ and the indicator qubit has value 1. In other words, the projection $\Pi_1\Pi_{|\phi\rangle}$ efficiently implements the projection $|\Psi\rangle\langle\Psi| \otimes \mathbb{1}$, and therefore the measurement $\{|\Psi\rangle\langle\Psi| \otimes \mathbb{1}, \mathbb{1} - |\Psi\rangle\langle\Psi| \otimes \mathbb{1}\}$.

[2] This assumes a white-box view of $\mathcal{A}$.

Define the unitary operator $U = TQ$, where Q and T are as above. The action of U on the state $|\psi\rangle|0^{m+k+1}\rangle$ can be written as

$$U|\psi\rangle|0^{m+k+1}\rangle = \sqrt{p}\,|\psi\rangle|\psi\rangle|\phi\rangle|1\rangle|0^{\ell}\rangle + \sqrt{1-p}\,|\Phi\rangle|0^{\ell}\rangle. \tag{5}$$

for some normalized state $|\Phi\rangle$ that is orthogonal to the state $|\Psi\rangle|0\rangle$. Specifically, $|\Phi\rangle|0^{\ell}\rangle = T|\Phi_3\rangle|0^{\ell+1}\rangle$, where $|\Phi_3\rangle$ is defined by Equation (4).

Next, we define the following states to facilitate the analysis of our amplification algorithm:

$$\begin{aligned}
|\phi_1\rangle &= \frac{1}{\sqrt{p}}(|\psi\rangle|0^{m+k+1}\rangle - \sqrt{1-p}\,U^*|\Phi\rangle|0^{\ell}\rangle),\\
|\phi_1^{\perp}\rangle &= U^*|\Phi\rangle|0^{\ell}\rangle\\
|\phi_2\rangle &= |\psi\rangle|0^{m+k+1}\rangle,\\
|\phi_2^{\perp}\rangle &= \frac{1}{\sqrt{p}}(\sqrt{1-p}\,|\psi\rangle|0^{m+k+1}\rangle - U^*|\Phi\rangle|0^{\ell}\rangle),
\end{aligned}$$

where $|\Phi\rangle$ is defined by Equation (5). Let $\mathcal{H}$ be the 2-dimensional Hilbert space spanned by $|\phi_1\rangle$ and $|\phi_1^{\perp}\rangle$. Then both $\{|\phi_1\rangle, |\phi_1^{\perp}\rangle\}$ and $\{|\phi_2\rangle, |\phi_2^{\perp}\rangle\}$ form orthonormal bases of $\mathcal{H}$. From the definitions of these states, we obtain the relations

$$\begin{aligned}
|\phi_1\rangle &= \sqrt{p}\,|\phi_2\rangle + \sqrt{1-p}\,|\phi_2^{\perp}\rangle,\\
|\phi_1^{\perp}\rangle &= \sqrt{1-p}\,|\phi_2\rangle - \sqrt{p}\,|\phi_2^{\perp}\rangle,\\
|\phi_2\rangle &= \sqrt{p}\,|\phi_1\rangle + \sqrt{1-p}\,|\phi_1^{\perp}\rangle,\\
|\phi_2^{\perp}\rangle &= \sqrt{1-p}\,|\phi_1\rangle - \sqrt{p}\,|\phi_1^{\perp}\rangle.
\end{aligned} \tag{6}$$

Define the projection operator

$$\Pi = U^*\Pi_1\Pi_{|\phi\rangle}U. \tag{7}$$

Then, we have $\Pi|\phi_1\rangle = |\phi_1\rangle$ and $\Pi|\phi_1^{\perp}\rangle = 0$. Our algorithm will use the measurement described by $\Pi, \mathbb{1} - \Pi$. Additionally, we will utilize the reflection unitary $2|\phi_2\rangle\langle\phi_2| - \mathbb{1}$, which can be efficiently implemented using cmpIndex as follows:

1. Compute the serial number of the first register and store it in an ancilla.
2. Compute an additional qubit b, where $b = 1$ if the serial number is not equal to r or if the last $m + k + 1$ qubits is not in an all-zero state; otherwise, set $b = 0$
3. Apply the phase $(-1)^b$
4. Uncompute the b and the serial number.

With all these in hand, we can now outline our amplification algorithm.

Algorithm 2 (cloning amplification)**.**
Input: Banknote $(r, |\psi\rangle = |K^{(r)} * y\rangle)$, real number $\varepsilon > 0$
Output: The state $|\psi\rangle|\psi\rangle$ or "Fail"

1. Prepare the state $|\psi\rangle|0^{m+k+1}\rangle$
2. Repeat the following $\lceil (1/p)\log(1/\varepsilon)\rceil$ times
   (a) Apply the measurement described by $\{\Pi, \mathbb{1} - \Pi\}$, where Π is the projection defined in (7). If the outcome is 0, associated to Π, return the first two registers.

(b) Apply the reflection unitary $2|\phi_2\rangle\langle\phi_2| - \mathbb{1}$.

3. Return "Fail"

Theorem 4.3. *Given a banknote $(r, |\psi\rangle)$, let $\mathcal{A}$ be a cloning adversary with unitary circuit Q defined in (4). For any $\varepsilon \in (0,1)$, Algorithm 2 outputs $|\psi\rangle|\psi\rangle$ with probability at least $1-\varepsilon$. The algorithm makes $O((1/p)\log(1/\varepsilon))$ calls to Q and Q^*, and uses $\mathrm{poly}(\log N)$ additional operations.*

Proof. The algorithm starts with the state $|\phi_2\rangle = |\psi\rangle|0^{m+k+1}\rangle$. According to (6),

$$|\phi_2\rangle = \sqrt{p}\,|\phi_1\rangle + \sqrt{1-p}\,|\phi_1^\perp\rangle.$$

Applying the measurement $\{\Pi, \mathbb{1}-\Pi\}$, the outcome 0 occurs with probability p, resulting in the post-measurement state $|\phi_1\rangle$. The outcome 1 occurs with probability $1-p$, resulting in the post-measurement state$|\phi_1^\perp\rangle$. In case of outcome 1, the algorithm continuous with the state

$$|\phi_1^\perp\rangle = \sqrt{1-p}\,|\phi_2\rangle - \sqrt{p}\,|\phi_2^\perp\rangle.$$

Applying the reflection $2|\phi_2\rangle\langle\phi_2| - \mathbb{1}$ transforms this state to $\sqrt{1-p}\,|\phi_2\rangle + \sqrt{p}\,|\phi_2^\perp\rangle$, which, using (6), can be written as

$$2\sqrt{(p-p^2}\,|\phi_1\rangle + (1-2p)|\phi_1^\perp\rangle.$$

Applying the measurement $\{\Pi, \mathbb{1}-\Pi\}$ to this state, the outcome 1 occurs with probability $(1-2p)^2$, resulting in the post-measurement state $|\phi_1^\perp\rangle$. Therefore, the probability of outcome 1 in the first iteration is $1-p$, and for all subsequent iterations, it is $(1-2p)^2$. The probability that the algorithm does not stop after k iterations is then $(1-p)(1-2p)^{2k}$. For $k = \lceil (1/p)\log(1/\varepsilon)\rceil$, we have $(1-p)(1-2p)^{2k} \le \varepsilon$. □

Remark 1. In Theorem 4.3, it is assumed that the success probability p of the adversary $\mathcal{A}$ is known. However, this assumption can be eliminated by modifying Algorithm 2 to allow the loop to run indefinitely until the measurement in Step 2(a) outputs 0. The resulting algorithm still runs in expected polynomial time and guarantees the correct result.

Remark 2. In the following sections, we will need to clone states of the form $|(K+t)^{(r)} * x\rangle$, where K is a subgroup of $\mathbb{Z}_N$, $t = g \bmod K$ with an unknown $g \in \mathbb{Z}_N$, and $g * x$ is known, with $x \in X$ as a known fixed element. However, the cloning algorithm described earlier accepts inputs of the form $|K^{(r)} * y\rangle$ for a known $y \in X$. Note that $|(K+t)^{(r)} * x\rangle = |K^{(r)} * (t * x)\rangle = |K^{(r)} * y\rangle$ for $y = t * x$, but t is not known a priori. Fortunately, $|K^{(r)} * (t * x)\rangle \propto |K^{(r)} * (u * x)\rangle$ for any $u \in K + t$. Therefore, we only need to sample from the set $(K+t) * x$ to obtain y. This is a relatively trivial task, as we can simply use $y = g * x$. If a random sample is needed, Algorithm 1 can be run on the input $(|g * x\rangle, K)$, followed by a measurement of the resulting state.

4.4 Fixing The Phase Using Clones

Given a subgroup group $K \le \mathbb{Z}_N$ of order M, Algorithm 1 projects the classical state $|g * x\rangle$ onto a state $|(K+t)^{(r)} * x\rangle$, where $r \in \mathbb{Z}_M$ is uniformly random and $t = g \bmod (N/M)$. However, for our security reduction, we will need such states where r is given, i.e., states of the form

$$|(K+t)^{(r)} * x\rangle = \frac{1}{\sqrt{|K|}} \sum_{u \in K} |(u+t) * x\rangle.$$

for a given $r \in \mathbb{Z}_M$. A naive approach wold be to run Algorithm 1 until we obtain the given r. But since the r outputted by Algorithm 1 is uniformly random, for each run of the algorithm, the

probability of obtaining the above state is $1/M$, which becomes negligible for large M. In following, we give an algorithm that efficiently prepares $|(K+t)^{(r)} * x\rangle$, for a given r, using the adversary's cloning algorithm.

Lemma 4.4. *Let $\mathcal{A}$ be a cloning algorithm for the money states in Construction 3.1 for the regular group action $(\mathbb{Z}_N, X, *)$. Given any subgroup $K \leq \mathbb{Z}_N$, any $r \in \mathbb{Z}_M$, where $M = |K|$, and any element $g * x \in X$, there exists a quantum algorithm that prepares the state $|(K+t)^{(r)} * x\rangle$, where $t = g \bmod K$. The Algorithm makes $\mathrm{poly}(\log N)$ calls to $\mathcal{A}$ and performs $\mathrm{poly}(\log N)$ additional operations.*

Proof. By Lemma 4.2 and Theorem 4.3, we assume that $\mathcal{A}$ is a nearly perfect cloning algorithm; that is, given a banknote $(r, |\psi\rangle)$ as input, $\mathcal{A}$ outputs $(r, |\psi\rangle|\psi\rangle)$ with probability exponentially close to 1.

Let $0 < K_1 < \cdots < K_{s-1} < K_s = K$ be a composition series of K, where $q_i = |K_i/K_{i-1}|$ is prime for all $i = 1, \ldots, s$. This means $M = |K_s| = q_1 q_2 \cdots g_s$. Let $(r_0, r_2, \ldots, r_{s-1})$ be the mixed radix representation of r with respect to the radix vector $(q_1, q_2, \ldots, q_{s-1})$. In this representation, we have $0 \leq r_i < q_i$ and

$$r = r_0 + r_1 q_1 + r_2 q_1 q_2 + \cdots + r_{s-1} q_1 \cdots q_{s-1}.$$

Define the partial sums $v_1 = r_0$ and $v_i = r_0 + r_1 q_1 + \cdots + r_{i-1} q_1 \cdots q_{i-1}$ for $i = 2, \ldots, s$. Our strategy is to begin with the projected state $|(K_1+t)^{(v_1)} * x\rangle$ over K_1. This state can be prepared by repeatedly running Algorithm 1 until we obtain the phase $r = r_0$ as in Lemma 4.1. Next, we apply Lemma 4.2 to prepare multiple copies of $|(K_1+t)^{(v_1)} * x\rangle$, which are then used to project onto the state $|(K_2+t)^{(v_2)} * x\rangle$. We continue this process, using copies of $|(K_2+t)^{(v_2)} * x\rangle$ to project onto $|(K_3+t)^{(v_3)} * x\rangle$, and so on, until we obtain the desired state over K.

We first show how to prepare the initial state. Since $|K_1| = q_1$, by running Algorithm 1 on the input $(|g * x\rangle, K_1)$, we can project $|g * x\rangle$ onto the state

$$|(K_1+t)^{(v)} * x\rangle = \frac{1}{\sqrt{q_1}} \sum_{u \in K_1} \omega_N^{uv} |(u+t) * x\rangle$$

where $v \in \mathbb{Z}_{q_1}$ is random, and $t = g \bmod (N/q_1)$. Since $q_1 \leq \mathrm{poly}(\log N)$ and v is uniformly random, the probability of obtaining the state $|(K_1+t)^{(v_0)} * x\rangle$ is $1/q_1$. Therefore, on average, we only need to run Algorithm 1 $O(q_1)$ times to obtain the outcome $v = v_0$.

Now, assume we have prepared the state $|(K_i+t)^{(v_i)} * x\rangle$ for some $2 \leq i \leq s$. We show how to prepare the state $|(K_{i+1}+t)^{(v_{i+1})} * x\rangle$ over K_{i+1}. Let $q = |K_i|$, i.e., $q = q_1 q_2 \cdots q_i$. Writing $|(K_i+t)^{(v_i)} * x\rangle$ in the basis $\{|\mathbb{Z}_N^{(h)} * x\rangle\}_{h \in \mathbb{Z}_N}$, we have

$$|(K_i+t)^{(v_i)} * x\rangle = \omega_N^{-v_i t} \frac{\sqrt{q}}{\sqrt{N}} \sum_{j \in \mathbb{Z}_{N/q}} \omega_{N/q}^{-jt} |\mathbb{Z}_N^{(jq+v_i)} * x\rangle.$$

We then use $\mathsf{cmpIndex}$ to compute $jq + v_i$ into an extra register, subtract v_i, and divide by q to obtain the state

$$\frac{\sqrt{q}}{\sqrt{N}} \sum_{j \in \mathbb{Z}_{N/q}} \omega_{N/q}^{-jt} |\mathbb{Z}_N^{(jq+v_i)} * x\rangle |j\rangle.$$

Next, we compute $j \pmod{q_{i+1}}$ into another register and measure the register. If the measurement outcome is not equal to r_i, we start over using another copy of $|(K_i+t)^{(v_i)} * x\rangle$. Note that outcome r_i occurs with probability $1/q_{i+1}$. Therefore, on average, we require at most $O(q_{i+1})$ copies of

$|(K_i + t)^{(v_i)} * x\rangle$ to obtain this outcome. When the outcome is r_i, we can write $j = kq_{i+1} + r_i$, where $k \in \mathbb{Z}_{N/q_{i+1}q}$. Therefore,

$$jq + v_i = kq_{i+1}q + r_iq + v_i = kq_{i+1}q + v_{i+1}.$$

The post-measurement state is

$$\frac{\sqrt{q_{i+1}q}}{\sqrt{N}} \sum_{k \in \mathbb{Z}_{N/q_{i+1}q}} \omega_{N/q_{i+1}q}^{-kt} |\mathbb{Z}_N^{(kq_{i+1}q+v_{i+1})} * x\rangle |kq_{i+1} + r_i\rangle.$$

Finally, using cmpIndex again, we uncompute the second register to obtain the state

$$\frac{\sqrt{q_{i+1}q}}{\sqrt{N}} \sum_{k \in \mathbb{Z}_{N/q_{i+1}q}} \omega_{N/q_{i+1}q}^{-kt} |\mathbb{Z}_N^{(kq_{i+1}q+v_{i+1})} * x\rangle = \omega_N^{v_{i+1}t} |(K_{i+1} + t)^{(v_{i+1})} * x\rangle,$$

which is the same as $|(K_{i+1} + t)^{(v_{i+1})} * x\rangle$ up to global phase. By repeating this process, we can prepare $|(K_i + t)^{(v_i)} * x\rangle$ for all the subgroups K_i in the composition series. □

4.5 Computing DL Modulo a Prime

Given a group action $(\mathbb{Z}_N, X, *)$, a pair $(x, g * x)$ for some $g \in \mathbb{Z}_N$, and a prime $\ell \mid N$, we show how to efficiently compute $g \bmod \ell$. The idea is to first project $|x\rangle$ onto a state $|K^{(r)} * x\rangle$ for a random r, and then use Lemma 4.4 to prepare the states $|(K + t)^{(r)} * x\rangle$, where $t = g \bmod K$. From there, we use the SWAP-TEST to determine the value of j for which the shifted state $|(K + j)^{(r)} * x\rangle$ is the same as the state $|(K + t)^{(r)} * x\rangle$.

Lemma 4.5. *Let $\mathcal{A}$ be a cloning algorithm for the money states in Construction 3.1 for the regular group action $(\mathbb{Z}_N, X, *)$. Given a pair $(x, g * x)$ for some $g \in \mathbb{Z}_N$, and a prime $\ell \mid N$, there exists a quantum algorithm that computes $g \bmod \ell$ using $\mathrm{poly}(\log N)$ calls to $\mathcal{A}$ and $\mathrm{poly}(\log N)$ additional operations.*

Proof. Let $K = \ell\mathbb{Z}_N$ be the unique subgroup of index ℓ in $\mathbb{Z}_N$. Using Lemma 4.1, we project $|x\rangle$ onto a state $|K^{(r)} * x\rangle$ for some uniformly random $r \in \mathbb{Z}_{N/\ell}$. Next, we use Lemma 4.4 to prepare the state $|(K + t)^{(r)} * x\rangle$, where $t = g \bmod \ell$. For each $0 \le j < \ell$, define the unitary U_j acting on $\mathbb{C}^X$ by $|y\rangle \mapsto |j * y\rangle$. Then, we have

$$U_j |K^{(r)} * x\rangle = |(K + j)^{(r)} * x\rangle.$$

Given j, suppose we perform the following procedure λ times: use Lemma 4.2 and Theorem 4.3 to prepare fresh copies of both $|(K + t)^{(r)} * x\rangle$ and $|K^{(r)} * x\rangle$. Then, run the SWAP-TEST on the states $|(K+t)^{(r)} * x\rangle$ and $U_j|K^{(r)} * x\rangle$. If the outputs of all λ runs are 0, then $t = j$ with probability at least $1 - 2^{-\lambda}$; otherwise $j \neq t$. Therefore, we can efficiently find t by testing all values of $0 \le j < \ell$. □

4.6 A Quantum Pohlig-Hellman Algorithm

Let $(\mathbb{Z}_N, X, *)$ be a regular group action, and let $N = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r}$ be the prime factorization of N. Given a pair $(x, g * x)$, where $g \in \mathbb{Z}_N$, we showed in the previous section how to compute $g \bmod \ell$ for any prime $\ell \mid N$. This motivates a Pohlig-Hellman-like algorithm for computing g. The approach is to compute each $\alpha_i = g \bmod p_i^{e_i}$ separately for $1 \le i \le r$, and then use the Chinese remainder theorem to reconstruct g from the collection of α_i's.

We proceed as follows. First, use Lemma 4.5 to compute $t = g \bmod p_1$. With t in hand, compute $y = (-t)*(g*x) = (g-t)*x$ to form a new pair (x, y). Now, consider the subgroup $K = p_1\mathbb{Z}_N \leq \mathbb{Z}_N$. The pair (x, y) is a new DLP instance with respect to the regular group action $(K, K * x, *)$. To apply Lemma 4.5 again, construct a new group action $(H, H \star x, \star)$, where $H = \mathbb{Z}_{N/p_1}$, as follows: for $u \in H$, define $u \star x = (p_1 u) * x$. Let $g' = (g - t)/p_1$. Then, the instance (x, y) with respect to $(K, K*x, *)$ corresponds to the instance $(x, y') = (x, g'\star x)$ with respect to $(H, H\star x, \star)$. We can now compute g' mod p_1 again and repeat this process until we obtain $\alpha_1 = g \bmod p_1^{e_1}$. This procedure can be used to obtain $\alpha_i = g \bmod p_i^{e_i}$ for all $i = 1, \ldots, r$. Finally, using the Chinese remainder theorem, we combine the results to recover g. The steps of this algorithm are summarized bellow.

Algorithm 3 (Pohlig-Hellman)**.**

Input: Regular group action $(\mathbb{Z}_N, X, *)$. A pair $(x, g * x)$ where $g \in G$.
Output: g

1. Let $N = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r}$ be the prime factorization of N.
2. For $i = 1$ to r do the following:
   (a) Let $y = g * x$, $p = p_i$, $\alpha_i = 0$.
   (b) For $e = 0$ to $e_i - 1$ do the following:
      i. Use Lemma 4.5 to compute $t := h \bmod p$ where h is defined in $y = h * x$.
      ii. Compute $y := (g - t) * x$, $N := N/p$, $\alpha_i = \alpha_i + p_i^e t$.
      iii. Let $K = \mathbb{Z}_N$. Construct a new group action $(K, K \star x, \star)$ as follows: for $u \in K$, $u \star x = (pu) * x$. Replace the old group action with this new group action.
3. Use the Chinese remainder theorem to recover g from $\alpha_1, \alpha_2, \ldots, \alpha_r$.
4. Return g.

Theorem 4.6. *Let $\mathcal{A}$ be a cloning algorithm for the money states in Construction 3.1 for the regular group action $(\mathbb{Z}_N, X, *)$. Given a pair $(x, g * x)$ for some $g \in \mathbb{Z}_N$, there exists a quantum algorithm that computes g using* poly$(\log N)$ *calls to $\mathcal{A}$ and* poly$(\log N)$ *additional operations.*

Proof. The number of iterations in the Pohlig-Hellman algorithm is bounded by poly$(\log N)$. Since N is smooth, the theorem follow from Lemma 4.5. □

An immediate corollary of Theorem 4.6 is that, given the ability to clone the Fourier states (3), we can efficiently prepare any desired Fourier state.

Corollary 4.7. *Given a quantum polynomial time algorithm for cloning the Fourier states (3), there exists a quantum polynomial time algorithm for preparing any desired Fourier state (3).*

A special case of the above corollary is that there is a quantum polynomial time reduction from the problem of preparing uniform superpositions over any given subgroup $K \leq \mathbb{Z}_N$ to cloning The Fourier states (3). Constructing such superpositions is a major open problem in the cryptography literature (see, for example, [21, 15, 6]). Note that we can also use Lemma 4.4 to generate the uniform superposition

$$|K^{(0)} * x\rangle = \frac{1}{\sqrt{|K|}} \sum_{u \in K} |u * x\rangle.$$

by setting $g = 0$ and $r = 0$.

We are now ready to state the main result for cyclic groups.

Theorem 4.8. *Let $(\mathbb{Z}_N, X, *)$ be an effective group action for which the DLP assumption holds. Assume $X \subset \{0,1\}^m$ and $m' \geq m + \omega(\log \lambda)$, where $\lambda = \mathrm{poly}(\log N)$ is the security parameter. Let $(\mathsf{Gen}, \mathsf{Ver})$ be the quantum money scheme from Construction 3.1, using the generic group action $\mathsf{GGAM}_{\mathbb{Z}_N, m'}$. Then this quantum money construction is secure.*

Proof. Let $\mathcal{A}$ be an adversary for the security of the quantum money scheme. By Theorem 3.2, we know that the only states accepted by Ver are those sufficiently close to valid money states. Therefore, we can consider $\mathcal{A}$ as a cloning adversary that clones money states with non-negligible probability. The theorem now follows directly from Theorem 4.6. □

5 General Abelian Groups

In this section, we outline how our security proof can be extended to general abelian groups. Let $(G, X, *)$ be a regular group action, where G an abelian group. We can decompose G as a direct sum of cyclic groups: $G \cong \mathbb{Z}_{N_1} \oplus \mathbb{Z}_{N_2} \oplus \cdots \oplus \mathbb{Z}_{N_k}$. Given an adversary for the security of the quantum money with respect to $(G, X, *)$, we show how to efficiently solve the group action DLP in this setting. The idea is to run an extension of our algorithm for the cyclic group $\mathbb{Z}_N$ on each of the components $\mathbb{Z}_{k_i}$ of G. Specifically, given a pair $(x, g * x)$ where $g = (g_1, g_2, \ldots, g_k) \in G$, recovering g reduces to the following problem: given a prime $p \mid N_1$, compute $t = g_1 \bmod p$. To achieve this, we extend the algorithms from Sections 4.1–4.4 with slight modifications. To simplify the notation, let $G = H \oplus \mathbb{Z}_N$, where $H = \mathbb{Z}_{N_1} \oplus \cdots \oplus \mathbb{Z}_{N_{k-1}}$ and $N = N_k$, and let $g = (\tilde{g}, g_k)$, where $\tilde{g} \in H$ and $g_k \in \mathbb{Z}_N$.

Projecting onto subgroup states. The projection algorithm (Algorithm 1) remains unchanged, except that we need to apply it to the second component of g. To demonstrate the extended algorithm, we carry out the steps on the input state $|g * x\rangle$. Let K be a subgroup of $\mathbb{Z}_{N_k}$ of order $|K| = M$. Write

$$|g * x\rangle = \frac{1}{\sqrt{|G|}} \sum_{v \in H} \sum_{z \in \mathbb{Z}_N} \chi_H(-\tilde{g}, v) \omega_N^{-z g_k} |G^{(v,z)} * x\rangle,$$

where $\chi_H(-\tilde{g}, \cdot)$ is a character of H. Running $\mathsf{cmpIndex}$ on the above state gives

$$\frac{1}{\sqrt{|G|}} \sum_{v \in H} \sum_{z \in \mathbb{Z}_N} \chi_H(-\tilde{g}, v) \omega_N^{-z g_k} |G^{(v,z)} * x\rangle |v, z\rangle.$$

Next, compute $(N/M)z$ into an extra register, and measure, to obtain the state

$$\frac{\sqrt{M}}{\sqrt{|G|}} \sum_{v \in H} \sum_{z \in M\mathbb{Z}_N + r_k} \chi_H(-\tilde{g}, v) \omega_N^{-z g_k} |G^{(v,z)} * x\rangle |v, z\rangle$$

for some $r_k \in \mathbb{Z}_M$. Let $|\psi\rangle$ be the state resulting from uncomputing the last register using $\mathsf{cmpIndex}$. Then

$$\begin{aligned} |\psi\rangle &= \frac{\sqrt{M}}{\sqrt{|G|}} \sum_{v \in H} \sum_{z \in M\mathbb{Z}_N + r_k} \chi_H(-\tilde{g}, v) \omega_N^{-z g_k} |G^{(v,z)} * x\rangle \\ &= \frac{\sqrt{M}}{|G|} \sum_{u \in G} \sum_{v \in H} \sum_{z \in M\mathbb{Z}_N + r_k} \chi_H(\tilde{u} - \tilde{g}, v) \omega_N^{(u_k - g_k) z} |u * x\rangle \qquad \text{(where } u = (\tilde{u}, u_k)\text{)} \end{aligned}$$

$$= \frac{1}{\sqrt{M}} \sum_{\substack{u_k \in \mathbb{Z}_N \\ u_k \equiv g_k \bmod (N/M)}} \omega_N^{(u_k - g_k) r_k} |(\tilde{g}, u_k) * x\rangle$$
$$= \frac{1}{\sqrt{|K|}} \sum_{u \in K} \omega_N^{u r_k} |(\tilde{g}, u + g_k) * x\rangle$$
$$= |(0 \oplus K + (\tilde{g}, g_k))^{(0, r_k)} * x\rangle.$$

Here, $0 \oplus K$ is the subgroup of G, where the first $k-1$ components are zero. Similarly, $(0, r_k)$ denotes an element of G in which the first $k-1$ components are zero. As before, for $t = g_k \bmod K$, the above state is the same as $|(0 \oplus K + (\tilde{g}, t))^{(0, r_k)} * x\rangle$ up to global phase. Let $y = (\tilde{g}, t) * x$ and $Y = (0 \oplus K) * y$. Then, $|(0 \oplus K + (\tilde{g}, t))^{(0, r_k)} * x\rangle = |(0 \oplus K)^{(0, r_k)} * y\rangle$ is a money state with respect to the group action $(0 \oplus K, (0 \oplus K) * x, *)$.

Cloning subgroup states. The cloning algorithm of Sections 4.2 and 4.3 remains unchanged. In fact, the algorithm does not rely on the assumption that the underlying group is $\mathbb{Z}_N$. Therefore, we can replace $\mathbb{Z}_N$ with a general abelian group G, and the proofs of Lemma 4.2 and Theorem 4.3 remains valid without modification.

The reduction. Let $\ell \mid N$ be prime and, let $K = \ell \mathbb{Z}_N$ be the subgroup of $\mathbb{Z}_N$ of size N/ℓ. Using the projection algorithm described above, we can prepare the state $|(0 \oplus K + (\tilde{g}, g_k))^{(0, r_k)} * x\rangle$ for a random $r \in \mathbb{Z}_{N/\ell}$. The idea is to zero out the components of g one by one, starting with g_k, in this state until we obtain a state that depends only on g_1. From there, we can compute $g_1 \bmod p$. We begin by writing

$$|(0 \oplus K + (\tilde{g}, g_k))^{(0, r_k)} * x\rangle = \frac{\sqrt{N}}{\sqrt{\ell |G|}} \sum_{v \in H} \sum_{z \in (N/\ell)\mathbb{Z}_N + r_k} \chi_H(-\tilde{g}, v) \omega_N^{-z g_k} |G^{(v,z)} * x\rangle$$
$$= \frac{\sqrt{N}}{\sqrt{\ell |G|}} \sum_{v \in H} \sum_{j \in \mathbb{Z}_\ell} \chi_H(-\tilde{g}, v) \omega_N^{-r_k g_k} \omega_\ell^{-j g_k} |G^{(v, jN/\ell + r_k)} * x\rangle.$$

Applying cmpIndex to the above state stores $(v, jN/\ell + r_k)$ in an ancilla register. By subtracting r_k from the last register and dividing the result by N/ℓ, we obtain the state

$$\frac{\sqrt{N}}{\sqrt{\ell |G|}} \sum_{v \in H} \sum_{j \in \mathbb{Z}_\ell} \chi_H(-\tilde{g}, v) \omega_\ell^{-j g_k} |G^{(v, jN/\ell + r_k)} * x\rangle |v, j\rangle.$$

Measuring the last register, we obtain the measurement outcome $j = 0$ with probability $1/\ell$. Therefore, using enough copies of the above state, with at most $O(\ell)$ trials, we obtain the state

$$\frac{\sqrt{N}}{\sqrt{|G|}} \sum_{v \in H} \chi_H(-\tilde{g}, v) |G^{(v, r_k)} * x\rangle = |(0 \oplus \mathbb{Z}_{N_k} + (\tilde{g}, 0))^{(0, r_k)} * x\rangle.$$

Continuing a similar process with the subgroup H and its components, we can zero out g_{k-1} to obtain the state $|(0 \oplus \mathbb{Z}_{N_{k-1}} \oplus \mathbb{Z}_{N_k} + (g_1, \ldots, g_{k-2}, 0, 0))^{(0, r_{k-1}, r_k)} * x\rangle$, and then the state

$$|(0 \oplus \mathbb{Z}_{N_{k-2}} \oplus \mathbb{Z}_{N_{k-1}} \oplus \mathbb{Z}_{N_k} + (g_1, \ldots, g_{k-3}, 0, 0, 0))^{(0, r_{k-2}, r_{k-1}, r_k)} * x\rangle,$$

and so on, until we obtain

$$|(0 \oplus \mathbb{Z}_{N_2} \oplus \cdots \oplus \mathbb{Z}_{N_k} + (g_1, 0, \ldots, 0))^{(0,r_2,\ldots,r_k)} * x\rangle = \frac{1}{\sqrt{|B|}} \sum_{u \in B} \chi_B(u, r)|(g_1, u) * x\rangle$$
$$= \frac{1}{\sqrt{N_1}} \sum_{h \in \mathbb{Z}_{N_1}} \omega_{N_1}^{-hg_1} |G^{(h,r)} * x\rangle,$$

where $B = \mathbb{Z}_{N_2} \oplus \cdots \oplus \mathbb{Z}_{N_k}$ and $r = (r_2, \ldots, r_k)$. At this point, we can apply the extension of the algorithm from Section 4.5 to the above state to compute $g_1 \bmod p$. Note that after running $\mathsf{cmpIndex}$ on the above state, the content of the last register will be (h, r). Therefore, the algorithms from Sections 4.4 and 4.5 only act on the first component h.

The following theorem is a direct generalization of Theorem 4.8 and can be proved using the same reasoning.

Theorem 5.1. *Let $(G, X, *)$ be an effective abelian group action for which the DLP assumption holds. Assume $X \subset \{0,1\}^m$ and $m' \geq m + \omega(\log \lambda)$, where $\lambda = \mathrm{poly}(\log|G|)$ is the security parameter. Let $(\mathsf{Gen}, \mathsf{Ver})$ be the quantum money scheme from Construction 3.1, using the generic group action $\mathsf{GGAM}_{G,m'}$. Then this quantum money construction is secure.*

6 On Smoothness

Since the assumption regarding the smoothness of the order of the abelian group G in the group action $(G, X, *)$ is crucial to our security proof, we need to address two main concerns related to this assumption:

1. Is the group action DLP easier when $|G|$ is smooth?
2. Is it always possible to have potential secure instantiations for different security parameters under this smoothness condition?

Regarding the first question, as long as the group G contains a large cyclic subgroup, there is no known evidence suggesting that the hardness of DLP with respect to a group action $(G, X, *)$ depends on the structure of G. It is commonly believed in the literature that the one-wayness of the action $*$ depends solely on the action itself, not on the structure of G or the set X.

For the second question, we will discuss in the following a potential secure instantiation using isogenies of elliptic curves. In isogeny-based group actions, the group G is the *ideal class group* of an imaginary quadratic order $\mathcal{O}$, and the set X consists of elliptic curves with endomorphism ring $\mathcal{O}$. To achieve smooth group orders, we employ a generalization of well-known group actions, such as CSIDH [8], based on oriented elliptic curves. *This generalization allows us to tweak parameters in order to find ideal class groups of smooth order.*

Before we discuss oriented isogenies, let us briefly review some basics of isogeny-based group actions. We refer the reader to [31, 18, 22] for background on elliptic curves and isogenies. Let $\mathcal{O}$ be a quadratic order in a number field K. The set of invertible ideals in $\mathcal{O}$, modulo principle ideals, form a multiplicative abelian group called the ideal class group of $\mathcal{O}$, denoted by $\mathrm{Cl}(\mathcal{O})$. Let $\mathbb{F}_q$ be the finite field with $q = p^k$ elements. For an elliptic curve E over $\mathbb{F}_q$, written as $E/\mathbb{F}_q$, denote by $\mathrm{End}(E)$ the endomorphism ring of E. Given a quadratic order $\mathcal{O}$, let $\mathrm{Ell}_q(\mathcal{O})$ be the set of elliptic $E/\mathbb{F}_q$ such that $\mathrm{End}(E) \cong \mathcal{O}$. Given an ideal $\mathfrak{a} \in \mathcal{O}$ and an elliptic curve $E \in \mathrm{Ell}_q(\mathcal{O})$, there is a unique isogeny $\phi_\mathfrak{a} : E \to E_\mathfrak{a}$ with kernel $E[\mathfrak{a}] = \cap_{\alpha \in \mathfrak{a}} \ker(\alpha)$. This defines a group action

$$\mathrm{Cl}(\mathcal{O}) \times \mathrm{Ell}_q(\mathcal{O}) \longrightarrow \mathrm{Ell}_q(\mathcal{O})$$

$$(\mathfrak{a}, E) \longmapsto \mathfrak{a} * E$$

where $\mathfrak{a} * E = E_{\mathfrak{a}}$. This is a regular group action.

The above group action is defined only for *ordinary* elliptic curves, which are curves whose endomorphism rings are isomorphic to a quadratic order in a number field. In contrast, the endomorphism rings of *supersingular* elliptic curves are orders in quaternion algebras, where the ideal classes do not even form a group. Despite this, the group action in CSIDH involves a quadratic order in a number field acting on a set of supersingular elliptic curves. As we will see, this action is explained by the concept of orientations, first introduced by Colo and Kohel [11].

6.1 Oriented isogenies

Let K be a quadratic number field, and let $\mathcal{O}$ be an order in K. A K-orientation on an elliptic curve E is a homomorphism $\iota : K \hookrightarrow \mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}$. Note that when $E/\mathbb{F}_q$ is ordinary, the $\mathbb{Q}$-algebra $\mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}$ is a number field, whereas if E is supersingular, it is the quaternion algebra $B_{p,\infty}$, ramified at p and ∞. An $\mathcal{O}$-orientation on E is a K-orientation for which $\iota(\mathcal{O}) \subseteq \mathrm{End}(E)$. An $\mathcal{O}$-orientation is called primitive if $\iota(\mathcal{O}) = \iota(K) \cap \mathrm{End}(E)$. We write (E, ι) to denote a K-oriented elliptic curve E with orientation ι.

Any isogeny $\phi : E \to F$ between elliptic curves E and F induces a map between K-orientations on E and F: Given a K-orientation $\iota : K \hookrightarrow \mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}$ there is a K-orientation $\phi_*(\iota) : K \hookrightarrow \mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}$ defined by

$$\phi_*(\iota)(\alpha) = (\phi \circ \iota(\alpha) \circ \hat{\phi}) \otimes \frac{1}{\deg(\phi)}.$$

Here, $\iota(\alpha)$ is an endomorphism of E, and $\hat{\phi} : F \to E$ is the dual isogeny of ϕ, defined by the condition $\phi \circ \hat{\phi} = \hat{\phi} \circ \phi = [\deg \phi]$.

Definition 6.1 (Oriented isogeny)**.** Let (E_1, ι_1) and (E_2, ι_2) be two K-oriented elliptic curves. An isogeny $\phi : E_1 \to E_2$ is said to be K-oriented if $\phi_*(\iota_1) = \iota_2$, i.e., the orientation ι_2 on E_2 is induced by the orientation ι_1 on E_1, in which case we write the isogeny as $\phi : (E_1, \iota_1) \to (E_2, \iota_2)$. We write $(E_1, \iota_1) \cong (E_2, \iota_2)$ if there is a K-oriented isomorphism between E_1 and E_2.

Given a quadratic order $\mathcal{O}$, let $\mathrm{SS}_{\mathcal{O}}(p)$ be the set of K-oriented isomorphism classes of supersingular elliptic curves, over the algebraic closure $\overline{\mathbb{F}}_p$, with a primitive $\mathcal{O}$-orientation. It was shone by Onuki [26] that $\mathrm{SS}_{\mathcal{O}}(p)$ is not empty if and only if p does not divide the conductor of $\mathcal{O}$ and does not split in K. We assume that p always satisfies these conditions so that $\mathrm{SS}_{\mathcal{O}}(p)$ is always nonempty.

Similar to the ordinary case, ideals in $\mathcal{O}$ define unique isogenies from the elements of $\mathrm{SS}_{\mathcal{O}}(p)$ as follows. Consider an element $(E, \iota) \in \mathrm{SS}_{\mathcal{O}}(p)$. Any ideal $\mathfrak{a} \in \mathcal{O}$ defines a subgroup

$$E[\mathfrak{a}] = \bigcap_{\alpha \in \mathfrak{a}} \ker(\iota(\alpha)).$$

This subgroup defines a unique isogeny $\phi_{\mathfrak{a}} : E \to E_{\mathfrak{a}}$, which induces an isogeny of K-oriented elliptic curves $(E, \iota) \to (E_{\mathfrak{a}}, (\phi_{\mathfrak{a}})_*(\iota))$. While it may not be immediately clear that this defines an action of $\mathrm{Cl}(\mathcal{O})$ on $\mathrm{SS}\mathcal{O}(p)$, Onuki [26] showed that it indeed does, yielding the following action:

$$\begin{aligned} \mathrm{Cl}(\mathcal{O}) \times \mathrm{SS}_{\mathcal{O}}(p) &\longrightarrow \mathrm{SS}_{\mathcal{O}}(p) \\ (\mathfrak{a}, (E, \iota)) &\longmapsto \mathfrak{a} * (E, \iota), \end{aligned} \tag{8}$$

where $\mathfrak{a} * (E, \iota) = (E_{\mathfrak{a}}, (\phi_{\mathfrak{a}})_*(\iota))$. This action is automatically free but not transitive, as it may have two orbit, leading to $|\mathrm{SS}_{\mathcal{O}}(p)| = 2|\mathrm{Cl}(\mathcal{O})|$. However, as proved in [4, Theorem 4.4], when p is ramified in K, the above action becomes transitive. Therefore, under the assumption that p is ramified in K, this action is regular.

With the above background, the particular case of CSIDH can be easily explained as follows. Let $K = \mathbb{Q}(\sqrt{-p})$, and let $\iota : K \hookrightarrow B_{p,\infty}$ be a K-orientation that sends $\sqrt{-p}$ to the p-th power Frobenius endomorphism π. Then, the set of (isomorphism classes) of elliptic curves used in CSIDH is $\mathrm{SS}_{\mathbb{Z}[\sqrt{-p}]}(p)$, i.e., the set of supersingular elliptic curves with a primitive $\mathbb{Z}[\sqrt{-p}]$-orientation.

Computing the action. We always have $\mathcal{O} = \mathbb{Z}[\alpha]$, where α is called a generator of $\mathcal{O}$. Therefore, an orientation $\iota : \mathcal{O} \hookrightarrow B_{p,\infty}$ is completely determined by the endomorphism $\iota(\alpha)$. An efficient representation of ι would then mean an efficient representation of both α and $\iota(\alpha)$. Given such an efficient representation of $(E, \iota) \in \mathrm{SS}_{\mathcal{O}}(p)$, and any ideal $\mathfrak{a} \in \mathrm{Cl}(\mathcal{O})$, the action $\mathfrak{a} * (E, \iota)$ can be computed efficiently [27, 28, 25].

6.2 Class groups of smooth order

Now that we can choose arbitrary quadratic orders for the action (8) on oriented elliptic curves, we have significant flexibility in selecting class groups of a specific order. A naive approach is to randomly choose an order $\mathcal{O} \subset K = \mathbb{Q}(\sqrt{D})$ and compute the size $|\mathcal{O}|$ to check if it is smooth. However, computing $|\mathcal{O}|$ requires a quantum computer. This approach may not yield an efficient algorithm, as the probability of $|\mathcal{O}|$ being smooth is comparable to the probability of a random integer being smooth, which is negligible [10].

A more efficient algorithm is based on the reverse of the this approach: instead of randomly choosing $\mathcal{O}$, we first choose a random smooth integer t and then find a quadratic order $\mathcal{O}$ such that $t = |\mathcal{O}|$. This method is inspired by the following formula, which relates the class numbers of two quadratic orders $\mathcal{O}' \subseteq \mathcal{O}$.

Lemma 6.2 ([14, Corollary 7.28])**.** *Let $D \equiv 0, 1 \bmod 4$ be a negative integer. Let $\mathcal{O}' \subseteq \mathcal{O}$ be orders in the number field $K = \mathbb{Q}(\sqrt{D})$ such that $[\mathcal{O} : \mathcal{O}'] = f$. Then*

$$h(\mathcal{O}') = \frac{h(\mathcal{O}) f}{[\mathcal{O}^* : \mathcal{O}'^*]} \prod_{p \mid f} \left(1 - \left(\frac{D}{p}\right) \frac{1}{p}\right). \tag{9}$$

The idea is to equate the right-hand side of (9) to random smooth numbers and solve for f until an appropriate value is found. Note that the order $\mathcal{O}'$ can be written as $\mathcal{O}' = \mathbb{Z} + f\mathcal{O}$. To simplify computations, we can choose $\mathcal{O}$ to be the maximal order $\mathcal{O}_K$, in which case f is the conductor of $\mathcal{O}'$. Additionally, we can choose D such that $\mathcal{O}_K$ has class number 1. Define

$$\Phi_D(f) = f \prod_{p \mid f} \left(1 - \left(\frac{D}{p}\right) \frac{1}{p}\right).$$

Given a random smooth integer $t > 0$, the goal is to find f such that $\Phi_D(f) = t$. This reduces to finding a prime number f such that $f - \left(\frac{D}{f}\right)$ is smooth. Again, this can be done efficiently by choosing random smooth integers and checking whether they are of the form $f - \left(\frac{D}{f}\right)$ for some prime f. This approach is used in [17, 9].

To summarize, we aim to find a set of primes $f_1, f_2, \ldots, f_r$ such that each $f_i - \left(\frac{D}{f_i}\right)$ is smooth, and then set $f = f_1 f_2 \cdots f_r$ as the desired conductor. To ensure that the DLP with respect to the action (8) is not vulnerable to known attacks [33, 11], f must not be smooth, i.e., at least one of the primes f_i must be large.

References

[1] Scott Aaronson. Quantum copy-protection and quantum money. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 229–242. IEEE, 2009.

[2] Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 41–60, 2012.

[3] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*, pages 411–439. Springer, 2020.

[4] Sarah Arpin, Mingjie Chen, Kristin E Lauter, Renate Scheidler, Katherine E Stange, and Ha TN Tran. Orientations and cycles in supersingular isogeny graphs. In *Research Directions in Number Theory: Women in Numbers V*, pages 25–86. Springer, 2024.

[5] Andriyan Bilyk, Javad Doliskani, and Zhiyong Gong. Cryptanalysis of three quantum money schemes. *Quantum Information Processing*, 22(4):177, 2023.

[6] Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E Stange, et al. Failing to hash into supersingular isogeny graphs. *The Computer Journal*, page bxae038, 2024.

[7] Gilles Brassard and Moti Yung. One-way group actions. In *Advances in Cryptology-CRYPTO'90: Proceedings 10*, pages 94–107. Springer, 1991.

[8] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III 24*, pages 395–427. Springer, 2018.

[9] Mingjie Chen, Antonin Leroux, and Lorenz Panny. Scallop-hd: group action from 2-dimensional isogenies. In *IACR International Conference on Public-Key Cryptography*, pages 190–216. Springer, 2024.

[10] Henri Cohen and Hendrik W Lenstra Jr. Heuristics on class groups of number fields. In *Number Theory Noordwijkerhout 1983: Proceedings of the Journées Arithmétiques held at Noordwijkerhout, The Netherlands July 11–15, 1983*, pages 33–62. Springer, 2006.

[11] Leonardo Colo and David Kohel. Orienting supersingular isogeny graphs. *Journal of Mathematical Cryptology*, 14(1):414–437, 2020.

[12] Marta Conde Pena, Raul Durán Díaz, Jean-Charles Faugère, Luis Hernández Encinas, and Ludovic Perret. Non-quantum cryptanalysis of the noisy version of aaronson–christiano's quantum money scheme. *IET Information Security*, 13(4):362–366, 2019.

[13] Jean-Marc Couveignes. Hard homogeneous spaces. *Cryptology ePrint Archive*, 2006.

[14] David A Cox. *Primes of the Form x2+ ny2: Fermat, Class Field Theory, and Complex Multiplication. with Solutions*, volume 387. American Mathematical Soc., 2022.

[15] Javad Doliskani. How to sample from the limiting distribution of a continuous-time quantum walk. *IEEE Transactions on Information Theory*, 69(11):7149–7159, 2023.

[16] Edward Farhi, David Gosset, Avinatan Hassidim, Andrew Lutomirski, and Peter Shor. Quantum money from knots. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 276–289, 2012.

[17] Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. Scallop: scaling the csi-fish. In *IACR international conference on public-key cryptography*, pages 345–375. Springer, 2023.

[18] D. Husemoller. *Elliptic Curves*. Graduate Texts in Mathematics. Springer New York, 2013.

[19] Daniel M Kane, Shahed Sharif, and Alice Silverberg. Quantum money from quaternion algebras. *arXiv preprint arXiv:2109.12643*, 2021.

[20] Andrey Boris Khesin, Jonathan Z Lu, and Peter W Shor. Publicly verifiable quantum money from random lattices. *arXiv preprint arXiv:2207.13135*, 2022.

[21] Jiahui Liu, Hart Montgomery, and Mark Zhandry. Another round of breaking and making quantum money: How to not build it from lattices, and more. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 611–638. Springer, 2023.

[22] Luca De Feo. Mathematics of Isogeny Based Cryptography. arXiv:1711.04062, 2017.

[23] Andrew Lutomirski, Scott Aaronson, Edward Farhi, David Gosset, Avinatan Hassidim, Jonathan Kelner, and Peter Shor. Breaking and making quantum money: toward a new quantum cryptographic protocol. *arXiv preprint arXiv:0912.3825*, 2009.

[24] Chris Marriott and John Watrous. Quantum arthur–merlin games. *computational complexity*, 14(2):122–152, 2005.

[25] Arthur Herlédan Le Merdy and Benjamin Wesolowski. The supersingular endomorphism ring problem given one endomorphism. *arXiv preprint arXiv:2309.11912*, 2023.

[26] Hiroshi Onuki. On oriented supersingular elliptic curves. *Finite Fields and Their Applications*, 69:101777, 2021.

[27] Aurel Page and Damien Robert. Introducing clapoti (s): Evaluating the isogeny class group action in polynomial time. *Cryptology ePrint Archive*, 2023.

[28] Damien Robert. Some applications of higher dimensional isogenies to elliptic curves (overview of results). 2023.

[29] Bhaskar Roberts. Security analysis of quantum lightning. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 562–567. Springer, 2021.

[30] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology—EUROCRYPT'97: International Conference on the Theory and Application of Cryptographic Techniques Konstanz, Germany, May 11–15, 1997 Proceedings 16*, pages 256–266. Springer, 1997.

[31] Joseph H Silverman. *The arithmetic of elliptic curves*, volume 106. Springer, 2009.

[32] John Watrous. Zero-knowledge against quantum attacks. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of Computing*, pages 296–305, 2006.

[33] Benjamin Wesolowski. Orientations and the supersingular endomorphism ring problem. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 345–371. Springer, 2022.

[34] Stephen Wiesner. Conjugate coding. *ACM Sigact News*, 15(1):78–88, 1983.

[35] Mark Zhandry. Quantum lightning never strikes the same state twice. or: quantum money from cryptographic assumptions. *Journal of Cryptology*, 34:1–56, 2021.

[36] Mark Zhandry. To label, or not to label (in generic groups). In *Annual International Cryptology Conference*, pages 66–96. Springer, 2022.

[37] Mark Zhandry. Quantum money from abelian group actions. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2024.