MODULAR FORMS AND HECKE OPERATORS FOR POST-QUANTUM CRYPTOGRAPHY

TREY LI

University of Manchester, Manchester, UK
trey.li@manchester.ac.uk

Abstract. We introduce modular forms and Hecke operators to cryptography and propose the Hecke problem as a new foundation for post-quantum cryptography. Given two modular forms, the Hecke problem asks to recover the Hecke operator that maps one to the other. While there is a deep relation to isogeny problems through the modularity theorem, this problem is rooted in arithmetic geometry and differs fundamentally in structure and mechanism. We prove NP-hardness of this problem and use it to construct a non-interactive key exchange scheme that achieves higher efficiency than isogeny-based schemes and smaller key sizes than lattice-based and code-based schemes.

1. Introduction

1.1. **Background and Motivation.** From classical elliptic curve cryptography to post-quantum isogeny-based cryptography, we observed a shift in the mathematical foundations: the elliptic curve discrete logarithm problem (ECDLP) in number theory was replaced by the supersingular isogeny problem in algebraic geometry [22].

Based on the supersingular isogeny problem, Supersingular Isogeny Diffie-Hellman (SIDH) [22] was once a promising candidate for post-quantum key exchange, offering high efficiency and compact keys. However, SIDH was shown to be insecure [8, 25, 28] due to the leakage of structural information via the auxiliary torsion points required for shared secret derivation. This vulnerability reveals a fundamental challenge in isogeny-based cryptographic constructions: torsion data can leak geometric information about the underlying isogeny graph.

Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) [9] avoids this issue by framing isogeny walks in an algebraic setting using class group actions. These actions are commutative and thereby eliminates the need for auxiliary torsion points, enhancing resistance to torsion-based attacks. However, this improvement comes at the cost of reduced efficiency, and its quantum security is still debatable.

1.2. **Our Contributions.** In this paper, we shift the foundation from algebraic geometry to arithmetic geometry, providing a novel security source and achieving higher efficiency and very compact keys. Rather than relying on the geometry of elliptic curves and isogenies, we explore the arithmetic of modular forms and Hecke operators, focusing on the action of Hecke operators on the Fourier expansions of modular forms. Hecke operators exhibit a structure analogous to isogenies but do not have a geometric interpretation vulnerable to torsion-based attacks.

We propose the following computational problem: given a modular form f and its image $H(f)$ under a composite Hecke operator $H = T_{N_1} \circ \cdots \circ T_{N_s}$, recover the factorization of H. This problem lives entirely in the world of Hecke modules over number fields, involving intricate manipulations of Fourier coefficients, and is currently not known to be reducible to any known isogeny, group action, or lattice problem.

Based on the hardness of this problem, we propose a new candidate for post-quantum (non-interactive) key exchange. By avoiding geometric isogenies and explicit class groups,

our construction inherits the efficiency of SIDH while introduces a fundamentally new source of security.

Our work opens a pathway for modular forms, a long central object to arithmetic geometry, to play a cryptographic role beyond their classical analytical and number-theoretic applications. Our approach diversifies the mathematical foundations of post-quantum cryptography and invites mathematicians to explore the untapped cryptographic potential of modular forms.

1.3. **Technical Overview.** Let $M_k(\Gamma_0(N))$ be the space of modular forms of weight k and level N, where $\Gamma_0(N)$ is the congruence subgroup of $\mathrm{SL}_2(\mathbb{Z})$ consisting of matrices $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ with $c \equiv 0 \pmod{N}$. Consider a modular form $f \in M_k(\Gamma_0(N))$ and a composite Hecke operator $H := \prod_{i=1}^{s} T_{N_i} := T_{N_1} \circ \cdots \circ T_{N_s}$, where $T_{N_1}, \ldots, T_{N_s}$ are Hecke operators with N_i positive integers coprime to N. The action of H transforms f into $H(f)$ by sequentially applying each T_{N_i} to entangle the Fourier coefficients nonlinearly as

$$(T_{N_i} f)(z) = N_i^{k/2-1} \sum_{n=0}^{\infty} \left(\sum_{d \mid \gcd(n, N_i)} d^{k-1} a_{nN_i/d^2} \right) e^{2n\pi \mathsf{i} z},$$

where the inner sum is over divisors d of $\gcd(n, N_i)$, with $a_{nN_i/d^2} = 0$ whenever nN_i/d^2 is not an integer. Here, the i appearing in the exponent of $e^{2\pi \mathsf{i} n z}$ denotes the imaginary unit and should not be confused with the index i. Based on this setup, the Hecke Problem is: given (finitely many Fourier coefficients of) f and $H(f)$, recover the factorization of H, or equivalently, recover the integers $N_1, \ldots, N_s$.

Although the modularity theorem [33] links elliptic curves over $\mathbb{Q}$ with weight-2 modular forms, and Hecke operators associate with isogenies or ideal class actions, these correspondences do not extend to supersingular curves over finite fields $\mathbb{F}_{p^2}$ or $\mathbb{F}_p$, nor do they translate geometric isogeny actions into Hecke operator actions. This fact positions the Hecke problem as an independent computational challenge, distinct from both the supersingular isogeny problem in SIDH [22] and the group action inverse problem in CSIDH [9].

Our non-interactive key exchange is a Diffie-Hellman-like key exchange. Fix a public modular form $f \in M_k(\Gamma_0(N))$, and two public sets of primes $S_A = \{p_1, \ldots, p_r\}$ and $S_B = \{q_1, \ldots, q_r\}$, where $p_i, q_i \nmid N$ and $S_A \cap S_B = \varnothing$. To share a common secret, Alice samples s integers $N_1, \ldots, N_s$, each a square-free integer consisting of precisely three prime factors in S_A, she denotes $H_A = T_{N_1} \circ \cdots \circ T_{N_s}$, computes $H_A(f)$, and sends $H_A(f)$ to Bob. Simultaneously, Bob samples s integers $M_1, \ldots, M_s$, each a square-free integer consisting of precisely three prime factors in S_B, he denotes $H_B = T_{M_1} \circ \cdots \circ T_{M_s}$, computes $H_B(f)$, and sends $H_B(f)$ to Alice.[1] Alice and Bob then compute $H_A(H_B(f))$ and $H_B(H_A(f))$ respectively. By the commutativity of Hecke operators, we have $H_A H_B = H_B H_A$, yielding a shared secret $H_A(H_B(f)) = H_B(H_A(f)) = \prod_{i=1}^{s} T_{N_i} T_{M_i}(f)$. This scheme avoids the auxiliary points in SIDH and the heavier operations in CSIDH, offering promising security with small keys and high efficiency.

1.4. **Organization.** The rest of the paper is organized as follows: Section 2 discusses the connections between elliptic curves and modular forms, as well as between isogenies and Hecke operators; Section 3 introduces our Hecke problem; Section 4 presents our Hecke-based key exchange protocol; Section 5 provides experimental results on efficiency, key sizes, and correctness; Section 6 presents additional applications of the Hecke problem; Section 7 briefly concludes the paper.

[1] Here we set the number of prime factors in each N_i and M_i to be "three" for the reduction from the NP-hard 3-Set Cover problem (X3C).

2. Elliptic Curves, Modular Forms, and Hecke Operators

We discuss the relation between elliptic curves and modular forms, and the relation between isogenies and Hecke operators.

2.1. **Elliptic Curves.** An *elliptic curve* E over $\mathbb{Q}$, denoted $E/\mathbb{Q}$, is a smooth, projective curve defined by an equation of the form $y^2 = x^3 + ax + b$, where $a, b \in \mathbb{Q}$ and the discriminant $\Delta = -16(4a^3 + 27b^2) \neq 0$ ensures that the curve is non-singular. The set of $\mathbb{F}_p$-rational points on E is denoted $E(\mathbb{F}_p)$. The arithmetic properties of an elliptic curve are encoded in the number of points in $E(\mathbb{F}_p)$, denoted $\#E(\mathbb{F}_p)$, for primes p. Hasse's theorem [21] asserts that $|p + 1 - \#E(\mathbb{F}_p)| \leq 2\sqrt{p}$.

The *conductor* of an elliptic curve E is an integer N that encodes information about the primes where E has bad reduction, that is, where the curve becomes singular modulo a prime p. The conductor not only indicates which primes cause bad reduction, but also quantifies the severity of that bad reduction. More specifically, the conductor is a product over all primes p, namely $N = \prod_p p^{e_p}$, where $e_p \geq 0$ with $e_p = 0$ if p is a prime of good reduction, and $e_p \geq 1$ if p is a prime of bad reduction.

The *L-function* of an elliptic curve E is

$$L(E, s) = \prod_p (1 - a_p p^{-s} + \epsilon(p) p^{1-2s})^{-1},$$

a function from $\mathbb{C}$ to $\mathbb{C}$, where $a_p = p + 1 - \#E(\mathbb{F}_p)$ for a prime p of good reduction, and $a_p = \pm 1$ for multiplicative bad reduction (a node) or $a_p = 0$ for additive bad reduction (a cusp); while $\epsilon(p) = 1$ for good reduction, keeping the full term $1 - a_p p^{-s} + p^{1-2s}$, and $\epsilon(p) = 0$ for bad reduction, simplifying it to $1 - a_p p^{-s}$ or 1.

2.2. **Isogenies.** An *isogeny* between elliptic curves E and E' over a field K is a rational map $\phi : E \to E'$ that is a group homomorphism. I.e., $\phi(P + Q) = \phi(P) + \phi(Q)$ for all $P, Q \in E(K)$. The kernel of an isogeny ϕ, denoted $\ker(\phi)$, is the set of points in E that maps to the identity of E'. The *degree* of an isogeny ϕ, denoted $\deg(\phi)$, is the order of $\ker(\phi)$.

The *ℓ-isogeny graph* is a graph with vertices being isomorphism classes of elliptic curves over $K = \mathbb{F}_{p^2}$, and edges being isogenies of a prime degree ℓ. In the supersingular case, this graph is a Ramanujan graph, a highly regular $(\ell + 1)$-regular expander.

Supersingular elliptic curves over a finite field $\mathbb{F}_q$ of characteristic p (e.g., $\mathbb{F}_p$ or $\mathbb{F}_{p^2}$) are elliptic curves such that the trace of the Frobenius endomorphism $\pi : (x, y) \mapsto (x^p, y^p)$ satisfies $\mathrm{Tr}(\pi) = 0$. For $p = q$, this implies $\#E(\mathbb{F}_p) = p + 1$. This results in rich endomorphism structures and highly connected isogeny graphs, making supersingular curves ideal for cryptographic use.

The *supersingular isogeny problem* asks to compute an isogeny $\phi : E \to E'$ between two supersingular curves. The *group action inverse problem* asks to determine the ideal class $[\mathfrak{a}] \in \mathrm{Cl}(\mathcal{O})$ such that $[\mathfrak{a}] * E = E'$, where $\mathrm{Cl}(\mathcal{O})$ is the ideal class group of a quadratic imaginary order $\mathcal{O}$ acting on the set of supersingular curves over $\mathbb{F}_p$.

2.3. **Modular Forms.** Let $\Gamma_0(N) = \left\{ \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in \mathrm{SL}_2(\mathbb{Z}) \;\middle|\; c \equiv 0 \pmod{N} \right\}$ be the congruence subgroup of $\mathrm{SL}_2(\mathbb{Z})$ with $c \equiv 0 \pmod{N}$. The group $\Gamma_0(N)$ acts on the upper half-plane $\mathbb{H} = \{\tau \in \mathbb{C} \mid \mathrm{Im}(\tau) > 0\}$ by fractional linear transformations: $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \cdot \tau = \frac{a\tau + b}{c\tau + d}$. A *modular form* f of weight k and level N is a holomorphic function $f : \mathbb{H} \to \mathbb{C}$ satisfying the transformation property $f\left(\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \cdot z\right) = (cz + d)^k f(z)$ for all $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in \Gamma_0(N)$, and it is

holomorphic at the cusps. Such a function admits a Fourier expansion

$$f(z) = \sum_{n=0}^{\infty} a_n q^n, q = e^{2\pi i z},$$

where the coefficients a_n determine the arithmetic properties of the corresponding L-function. Denote $M_k(\Gamma_0(N))$ as the space of modular forms of weight k and level N. The subspace $S_k(\Gamma_0(N)) \subset M_k(\Gamma_0(N))$ of cusp forms consists of those $f \in M_k(\Gamma_0(N))$ vanishing at all cusps. For the cusp at infinity, this is equivalent to the constant term $a_0 = 0$ in the q-expansion at infinity; for other cusps, one must verify the q-expansion (after suitable transformation to infinity) has vanishing constant term as well [16].

The *L-function* of a cusp form $f \in S_2(\Gamma_0(N))$ is

$$L(f, s) = \sum_{n=1}^{\infty} \frac{a_n}{n^s} = \prod_p (1 - a_p p^{-s} + p^{1-2s})^{-1},$$

a function from $\mathbb{C}$ to $\mathbb{C}$, where a_n is the n-th Fourier coefficient of $f(z) = \sum_{n=1}^{\infty} a_n e^{2\pi i n z}$.

2.4. **Hecke Operators.** Hecke operators are linear endomorphisms on spaces of modular forms, defined via double coset actions. For a prime p not dividing the level N, i.e., $p \nmid N$, the Hecke operator T_p acts on a modular form $f \in M_k(\Gamma_0(N))$ of weight k as follows. Suppose the Fourier expansion of f at the cusp ∞ is given by

$$f(z) = \sum_{n=0}^{\infty} a_n q^n, \text{ where } q = e^{2\pi i z}.$$

Then the action of T_p is defined by

$$(T_p f)(z) = \sum_{n=0}^{\infty} (a_{pn} + p^{k-1} a_{n/p}) q^n,$$

where $a_{n/p} = 0$ if p does not divide n. This formula can be viewed as a combination of two effects on the coefficients a_n of f: the multiplicative term a_{pn} corresponds to scaling the argument of f by p, and the divisibility term $p^{k-1} a_{n/p}$ which has nonzero value only when n/p is an integer, giving a sparse contribution.

For any positive integers m and n, the composition of Hecke operators satisfies:

$$T_m T_n = \sum_{d \mid \gcd(m,n)} d^{k-1} T_{mn/d^2}.$$

When m and n are coprime, this simplifies to $T_m T_n = T_{mn}$. In particular, for distinct primes p and q, we have $T_p T_q = T_{pq}$.

Hecke operators commute. I.e., for any positive integers m and n, we have $T_m T_n = T_n T_m$. This commutativity is central to our key exchange protocol, enabling Alice and Bob to compute a common shared secret regardless of the order of Hecke applications.

2.5. **Relation between Elliptic Curves and Modular Forms.** The modularity theorem [33] shows a deep connection between elliptic curves and modular forms. It asserts that for every elliptic curve E over $\mathbb{Q}$ with conductor N, there exists a cusp form $f \in S_2(\Gamma_0(N))$ such that $L(E, s) = L(f, s)$. In particular, the Fourier coefficients a_p of f match the arithmetic data $a_p = p + 1 - \#E(\mathbb{F}_p)$ for each prime p of good reduction. This implies that arithmetic properties of E, such as its point counts over finite fields, are encoded in the Fourier coefficients a_p of f.

However, this correspondence is non-constructive, offering no efficient algorithm to derive f from E or vice versa. Also, not all forms in $S_2(\Gamma_0(N))$ arise from elliptic curves,

some are "non-rational" or lack a matching L-function [16], not to mention modular forms that are not in $S_2(\Gamma_0(N))$.

For instances, forms in $S_2(\Gamma_0(12))$, $S_2(\Gamma_0(24))$, and $S_2(\Gamma_0(48))$ may correspond to elliptic curves over $\mathbb{Q}$, as their weight and level align with possible conductors, though not all do (e.g., $S_2(\Gamma_0(12))$ has dimension 1, and its form may not match the L-function of an elliptic curve). In contrast, spaces like $S_3(\Gamma_0(2))$, $S_4(\Gamma_0(2))$, $S_2(\Gamma_0(3))$, and $S_3(\Gamma_0(3))$ never correspond to elliptic curves, as weights above 2 or levels too small mismatch L-functions of elliptic curves.

Apart from $M_k(\Gamma_0(N))$, other common spaces of modular forms include $M_k(\Gamma_1(N))$, where $\Gamma_1(N)$ imposes $a \equiv d \equiv 1$, $c \equiv 0 \pmod{N}$, offering finer control via characters [16]; $M_k(\mathrm{SL}_2(\mathbb{Z}))$, level-1 forms (e.g., $E_4 = 1 + 240\sum_{n=1}^{\infty} \sigma_3(n)q^n$) [29]; $M_k(\Gamma(N))$, with $\Gamma(N)$ requiring $a \equiv d \equiv 1$, $b \equiv c \equiv 0 \pmod{N}$, tied to Galois representations [29]; vector-valued forms, under $\mathrm{SL}_2(\mathbb{Z})$ representations, enabling multidimensional key spaces [6]; half-integral weight forms (e.g., $k = 1/2$), linked to theta functions [30]; Siegel forms, on $\mathrm{Sp}_{2g}(\mathbb{Z})$, for abelian varieties [20]; and Jacobi forms, of (z, τ), for lattice structures [19]. This highlights a fertile domain for cryptographic research, beyond those that are covered by elliptic curves.

2.6. **Relation between Isogenies and Hecke Operators.** The action of Hecke operators on modular forms connects to the theory of isogenies through two fundamental but distinct relationships.

On one hand, Hecke operators T_p have a well-known geometric interpretation on the modular curve $X_0(N)$, which parametrizes pairs (E, C) where E is an elliptic curve and C is a cyclic subgroup of order N. For a prime $p \nmid N$, the action of T_p on a point (E, C) is given by:

$$T_p : (E, C) \mapsto \sum_{D \subset E[p], |D| = p} (E/D, (C + D)/D),$$

where D ranges over all cyclic subgroups of order p. This geometric action shows how T_p encodes information about all degree p isogenies originating from E.

One the other hand, through the Eichler-Shimura correspondence [29], a weight-2 newform $f \in S_2(\Gamma_0(N))$ is associated to an elliptic curve $E/\mathbb{Q}$ of conductor N such that their L-functions coincide: $L(E, s) = L(f, s) = \sum_{n=1}^{\infty} a_n n^{-s}$. For primes $p \nmid N$, the coefficient a_p of f satisfies $a_p = p + 1 - \#E(\mathbb{F}_p)$, relating the Fourier coefficients of f to point counts modulo p. This arithmetic relation stems from the action of the Frobenius endomorphism on the Tate module rather than from isogenies in characteristic zero.

While both perspectives (the geometric interpretation on modular curves and the arithmetic interpretation via Eichler-Shimura) involve elliptic curves and the same Hecke operators, they represent different mathematical phenomena: one concerns geometric averaging over isogenies in characteristic zero, while the other relates to arithmetic properties in finite characteristic.

3. The Hecke problem

We introduce the Hecke problem and prove its NP-hardness. This problem acts a role similar to the discrete logarithm problem in Diffie-Hellman key exchange. The Hecke Diffie-Hellman problems will be discussed after presenting our key exchange protocol in the next section.

We define the Hecke problem for modular forms in $M_k(\Gamma_0(N))$, with the understanding that it extends to broader classes of modular forms.

Definition 1. *Let $N_1, \ldots, N_s \in \mathbb{N}$ and $f \in M_k(\Gamma_0(N))$. Denote $H = \prod_{i=1}^{s} T_{N_i} = T_{N_1} \circ \cdots \circ T_{N_s}$ as the composition of Hecke operators $T_{N_1}, \ldots, T_{N_s}$. The Hecke problem is: given f and $H(f)$, find H or $N_1, \ldots, N_s$.*

Note that a modular form has infinitely many terms. For the Hecke problem to make computational sense, when we say "given f and $H(f)$", we mean to give the first $\ell \geq B$ Fourier coefficients of f and $H(f)$, where $B \approx \frac{k}{12}[\mathrm{SL}_2(\mathbb{Z}) : \Gamma_0(N)]$ is the Sturm bound. It is standard that $B := \left\lfloor \frac{k}{12} \left[\mathrm{SL}_2(\mathbb{Z}) : \Gamma_0(N)\right] \right\rfloor$ and two modular forms having identical coefficients up to this bound must coincide [32, Corollary 9.20]. This bound grows polynomially in the security parameter s, assuming k, N are polynomially bounded in s.

An Easy-to-Solve Scenario. To see the mechanism of the problem, we discuss a special scenario where the problem can be easily solved. Suppose the problem not only gives the first ℓ coefficients of each modular form, but oracle access to any coefficient. Also, suppose $N_1, \ldots, N_s$ are distinct primes $p_1, \ldots, p_s$ in a small range (hence can be easily enumerated) and $a_1 \neq 0$ (which is true for normalized Hecke eigenforms, Eisenstein series, and most cusp forms and general forms). Then the corresponding Hecke problem can be efficiently solved by observing the relations between the coefficients b_n of $H(f)$ and the coefficients a_n of f. Specifically, when $N_1, \ldots, N_s$ are distinct primes $p_1, \ldots, p_s$, the formula for b_n is simplified to

$$b_n = \sum_{d \mid \gcd(P,n)} d^{k-1} a_{n/d}$$

where $P := \prod_{i=1}^{s} p_i$. For any prime p, to see if p is in P, we look at the term

$$b_p = \sum_{d \mid \gcd(P,p)} d^{k-1} a_{p/d}.$$

Note that d can only be 1 or p, and the term with respect to $d = p$ exists if and only if $p|P$. We therefore have $b_p = a_p$ if and only if $p \nmid P$, and $b_p - a_p = p^{k-1} a_1 \neq 0$ (recall that we assume $a_1 \neq 0$) if and only if $p|P$. Hence by testing all potential primes p one by one, we can factor P and recover $H = T_P = \prod_{i=1}^{s} T_{p_i}$ in polynomial time.

In fact, even if $a_1 = 0$, the problem can still be easily solved. For a prime p, instead of looking at b_p, we now consider b_{p^2}, whose formula is

$$b_{p^2} = \sum_{d \mid \gcd(P,p^2)} d^{k-1} a_{p^2/d}.$$

Since P is square-free, there can be at most one p in P, and therefore d can only be 1 or p, and the term with respect to $d = p$ exists if and only if p is in P. Thus $b_{p^2} = a_{p^2} + d^{k-1} a_p$. As long as $a_p \neq 0$, the coefficient $b_{p^2} - a_{p^2}$ can tell whether $p|P$ or $p \nmid P$.

Nevertheless, for the Hecke problem as we defined, no classic nor quantum algorithms are known to be able to efficiently solve it in general. In fact, the Hecke problem is NP-hard, as we prove next.

3.1. **NP-Hardness.** In order to prove NP-hardness of the Hecke problem, we first specify a situation where Hecke operators have unique factorizations.

Definition 2. *Let $\mathbb{H}$ be the Hecke algebra acting on $M_k(\Gamma_0(N))$, and let $f \in M_k(\Gamma_0(N))$ be a nonzero modular form. We call f a distinguishing form if it distinguishes all Hecke operators in the sense that the map*

$$\phi_f : \mathbb{H} \to M_k(\Gamma_0(N)), \quad H \mapsto H(f)$$

is injective.

The condition that ϕ_f being injective is equivalent to f satisfying $H(f) = H'(f) \Rightarrow H = H'$ for all $H, H' \in \mathbb{H}$, or $f \notin \ker(H)$ for all nonzero $H \in \mathbb{H}$.

The existence of this kind of modular forms can be proved from standard arguments on the action of the Hecke algebra on finite-dimensional spaces of modular forms [15, 31].

Lemma 3. *Let $\mathbb{H}$ be the Hecke algebra acting on the space $M_k(\Gamma_0(N))$ of modular forms of weight k and level N over $\mathbb{C}$. Then there exists a distinguishing modular form f in $M_k(\Gamma_0(N))$.*

Proof. Since $M_k(\Gamma_0(N))$ is a finite-dimensional complex vector space, the Hecke algebra $\mathbb{H} \subset \mathrm{End}(M_k(\Gamma_0(N)))$ is also a finite-dimensional commutative $\mathbb{C}$-algebra. Consider the evaluation map

$$\phi_f : \mathbb{H} \to M_k(\Gamma_0(N)), \quad H \mapsto H(f).$$

This is a linear map of $\mathbb{C}$-vector spaces. For each nonzero $H \in \mathbb{H}$, the set of $f \in M_k(\Gamma_0(N))$ such that $H(f) = 0$ is a proper subspace of $M_k(\Gamma_0(N))$. Since there are only countably many such $H \in \mathbb{H}(\mathbb{Q})$ (defined over $\mathbb{Q}$), the union of their kernels lies in a proper Zariski-closed subset of $M_k(\Gamma_0(N))$.

Thus, a generic modular form f avoids this union and satisfies $H(f) \neq 0$ for all nonzero $H \in \mathbb{H}$, i.e., $\ker(\phi_f) = \{0\}$. Hence, ϕ_f is injective. In particular, this implies that $H(f) = H'(f)$ if and only if $H = H'$ for such f. □

In fact, this kind of modular forms are very common. A modular form $f \in M_k(\Gamma_0(N))$ chosen generically (e.g., with algebraically independent Fourier coefficients) will satisfy $\ker(\phi_f) = \{0\}$, and hence distinguish all Hecke operators. This follows from the fact that for each nonzero $H \in \mathbb{H}$, the kernel $\ker(H) \subset M_k(\Gamma_0(N))$ is a proper subspace, and the union of all such kernels is a countable union of nowhere dense sets. Therefore, a "random" modular form f avoids this union with probability 1.

Lemma 4. *If f is a distinguishing modular form, then the Hecke problem has a unique solution.*

Proof. Immediate from the definition of distinguishing modular forms. □

Theorem 5. *The Hecke problem is NP-hard for distinguishing forms f.*

Proof. We reduce the X3C problem to the Hecke problem. Let (S, C) be an X3C instance. Without loss of generality, we define $S = \{p_1, \ldots, p_r\}$ to be a set of prime numbers, and $C = \{C_1, \ldots, C_t\}$ to be a collection of 3-subset of S. We also denote $N_1, \ldots, N_t$ to be the corresponding square-free integers, i.e., N_i is the product of the three primes in C_i, for $i = 1, \ldots, t$.

Now let $P = \prod_{i=1}^{r} p_i$ be the square-free integer precisely factored over all elements of S. By the unique factorization property of integers, if we can find $N'_1, \ldots, N'_s \in \mathbb{N}$ such that $\prod_{i=1}^{s} N'_i = P$, then we have found a subset C' of C such that the elements of C' are disjoint and they union to S. Define $H = \prod_{i=1}^{r} T_{p_i}$. By the Hecke operator property that $T_p T_q = T_{pq}$ for primes p and q, we have $H = T_P$. Again, by Lemma 4, if given f and $H(f)$ we can find $N'_1, \ldots, N'_s$ such that $\prod_{i=1}^{s} T_{N'_i}(f) = T_P(f)$, then we must have $\prod_{i=1}^{s} N'_i = P$. Then we have found a solution to the X3C problem (S, C). □

3.2. **Computational Aspects.** There are two natural ways to compute $H(f)$: direct coefficient computation and matrix-based computation. We first explain why the first method is often impractical, then show how to compute $H(f)$ using the second method.

3.2.1. *Direct Coefficient Computation.* Note that having the numbers $N_1, \ldots, N_s$ is enough for computing the Hecke operators $T_{N_1}, \ldots, T_{N_s}$ on a modular form f, and there is no need to find their prime factorizations. To see this, suppose the original modular form is

$$f(z) = \sum_{n=0}^{\infty} a_n q^n.$$

Then the resulting modular form after applying the Hecke operator $H = T_{N_1} \circ \cdots \circ T_{N_s}$ is

$$H(f)(z) = \sum_{n=0}^{\infty} b_n q^n,$$

where the coefficients b_n are determined recursively by the following formula:

$$\begin{cases} b_n^{(0)} = a_n, \\ b_n^{(i)} = \displaystyle\sum_{d \mid \gcd(N_i, n)} d^{k-1} b_{\frac{nN_i}{d^2}}^{(i-1)}, \\ b_n = b_n^{(s)}. \end{cases} \tag{1}$$

We can see that the main obstacle for computing $H(f)$ is the computation of $\gcd(N_i, n)$, which is efficient using the Euclidean algorithm.

However, while formula (1) allows efficient computation of $H(f)$ given access to arbitrary coefficients a_n of f, this method is still impractical as it requires access to coefficients a_n that are exponentially far way from a_0, due to the rapidly increasing indices $\frac{nN_i}{d^2}$. In fact, if we truncate the first $\ell = poly(s)$ coefficients of f and set the rest zeros, then there can be only one nonzero term in the first ℓ terms of $H(f)$, and the Hecke action H becomes trivial integer multiplication. In the following, we first prove the case where $N_1, \ldots, N_s$ are primes $p_1, \ldots, p_s$. The case of composite $N_1, \ldots, N_s$ follows immediately.

Lemma 6. *Let $s, \ell \in \mathbb{N}$ with ℓ polynomial in s. Let f be a modular form with all terms after the ℓ-th term being set zero, and let $g = T_{p_1} \circ \cdots \circ T_{p_s}(f)$ be the modular form after applying s Hecke operators $T_{p_1}, \ldots, T_{p_s}$ to f, where $p_1, \ldots, p_s$ are primes. For any ℓ polynomial in s, in the first ℓ coefficients of g, only the constant term can be nonzero.*

Proof. Denote by $f^{(1)}, \ldots, f^{(s)}$ the modular forms obtained by successively applying the Hecke operators $T_{p_1}, \ldots, T_{p_s}$ to an initial form f, so that $f^{(s)} = g$. Denote the first ℓ coefficients of $f^{(i)}$ as $b_0^{(i)}, b_1^{(i)}, \ldots, b_{\ell-1}^{(i)}$. Note that

$$f^{(i)}(z) = (T_{p_i} f)(z) = \sum_{n=0}^{\infty} (a_{p_i n} + p_i^{k-1} a_{n/p_i}) q^n.$$

Each Fourier coefficient of $T_{p_i} f$ consists of two parts: the multiplicative term $a_{p_i n}$ from scaling the index by p_i, and the divisibility term $p_i^{k-1} a_{n/p_i}$ from reducing the index by p_i and weighted by p_i^{k-1}. We show that except for the constant term $b_0^{(s)}$, the multiplicative terms in the first ℓ coefficients of $f^{(s)}$ only rely on the j-th coefficients of f for $j \geq 2^s > \ell$ and thus are zero. We also show that except for the constant term $b_0^{(s)}$, the divisibility terms in the first ℓ coefficients of $f^{(1)}$ can only affect the j-th coefficients of $f^{(s)}$ for $j \geq 2^s > \ell$ and thus do not contribute to the first ℓ coefficients of $f^{(s)}$.

For the first target, it suffices to look at the second coefficient $b_1^{(s)}$ of $f^{(s)}$. Its multiplicative term is from the $(p_s + 1)$-th coefficient $b_{p_s}^{(s-1)}$ of $f^{(s-1)}$, whose multiplicative term is in turn from the $(p_s p_{s-1} + 1)$-th coefficient $b_{p_s p_{s-1}}^{(s-2)}$ of $f^{(s-2)}$, so on and so forth. We conclude that the multiplicative term of $b_1^{(s)}$ is initially from the $(\prod_{i=1}^{n} p_i + 1)$-th coefficient $a_{\prod_{i=1}^{s} p_i}$

of f. But $\prod_{i=1}^{s} p_i \geq 2^s > \ell$ and that $a_{\prod_{i=1}^{s} p_i} = 0$. Therefore, the multiplicative term of $b_1^{(s)}$ is zero. The cases for the multiplicative terms of the other coefficients $b_2^{(s)}, \ldots, b_{\epsilon(s)}^{s}$ are similar, they are all zero.

Now we prove the second target in the opposite direction from $f^{(1)}$ to $f^{(s)}$. Let $p_1^{k-1}a_1$ be the divisibility term of the second coefficient $b_1^{(1)}$ of $f^{(1)}$. It can only affect the next modular form $f^{(2)}$ on the j-th coefficients of $f^{(2)}$ for $j \geq p_1p_2+1$; so on and so forth. When it reaches $f^{(s)}$, it can only affect the j-th coefficients of $f^{(s)}$ for $j \geq \prod_{i=1}^{s} p_i + 1 \geq 2^s > \ell$. Namely it contributes zero to the first ℓ coefficients of $f^{(s)}$. The cases for the other divisibility terms $p_1^{k-1}a_2, p_1^{k-1}a_3, \ldots, p_1^{k-1}a_\ell$ are similar. Therefore, the divisibility terms of $b_1^{(s)}, \ldots, b_\ell^{(s)}$ are all zero.

In sum, the coefficients $b_1^{(s)}, \ldots, b_\ell^{(s)}$, each of which consists of its multiplicative term and divisibility term, are all zero.

By the way, it is not hard to derive the formula for the constant term $b_0^{(s)}$ of $f^{(s)}$ as:

$$b_0^{(s)} = \left(\prod_{i=1}^{s} \left(1 + p_i^{k-1}\right) \right) a_0,$$

where a_0 is the constant term of f. □

The conclusion is more true if we replace the primes p_i by composite integers N_i.

Corollary 7. *Let $s, \ell \in \mathbb{N}$ with ℓ polynomial in s. Let f be a modular form with all terms after the ℓ-th term being set zero, and let $g = T_{N_1} \circ \cdots \circ T_{N_s}(f)$ be the modular form after applying s Hecke operators $T_{N_1}, \ldots, T_{N_s}$ to f, where $N_1, \ldots, N_s$ are integers not equal to $-1, 0, 1$. For any ℓ polynomial in s, in the first ℓ coefficients of g, only the constant term can be nonzero.*

Proof. This is immediate by the proof of Lemma 6, as each $T_{N_i} = T_{p_1}^{e_1} \circ \cdots \circ T_{p_t}^{e_t}$ is a composition of at least one Hecke operator T_{p_j} with respect to a prime p_j. □

Therefore, if we use direct coefficient computation to compute $H(f)$ from the first ℓ coefficients of f, in the end we have at most a single nonzero coefficient $b_0 = \left(\prod_{i=1}^{s} \left(1 + p_i^{k-1}\right)\right) a_0$ within the first ℓ coefficients of $H(f)$, and the factor $\prod_{i=1}^{s} \left(1 + p_i^{k-1}\right)$ can be easily recovered by dividing b_0 by a_0.

We might have noticed that for Eisenstein series we have an explicit formula to compute any coefficient a_n from n in polynomial time without generating the entire q-series expansion. However, Eisenstein series are simultaneous eigenforms for all Hecke operators, a Hecke operator acting on it is merely scalar multiplication of the entire form, i.e., every Fourier coefficient is multiplied by the same factor. This trivializes the Hecke problem into a factoring problem. We therefore avoid using Eisenstein series as secure forms, and can alternatively adopt a matrix-based approach to compute $H(f)$, which we discuss next.

3.2.2. *Matrix-Based Computation.* Let $\{f_1, \ldots, f_d\} \subset M_k(\Gamma_0(N))$ be a basis, where $d \approx \frac{k \cdot [\mathrm{SL}_2(\mathbb{Z}):\Gamma_0(N)]}{12}$. First, express $f = \sum_{j=1}^{d} c_j f_j$, where the coordinate vector $c = (c_1, \ldots, c_d)$ is found by solving the system

$$\sum_{j=1}^{d} c_j a_n(f_j) = a_n, \quad n = 0, \ldots, \ell - 1,$$

with complexity $O(d^\omega)$, where $\omega \approx 2.373$ [2]. Second, notice that each Hecke operator T_{N_i} has a matrix $[T_{N_i}]$ such that

$$T_{N_i}(f_j) = \sum_{k=1}^{d} t_{kj}^{(i)} f_k, \quad [T_{N_i}] = [t_{kj}^{(i)}]_{d\times d}.$$

Given N_i, the matrix $[T_{N_i}]$ can be computed in time $O(d^2 \cdot \ell \cdot \log N_i)$ by applying the coefficient formula

$$a_n(T_{N_i}(f_j)) = \sum_{d|\gcd(N_i,n)} d^{k-1} a_{n/d}(f_j)$$

and expressing $T_{N_i}(f_j)$ in the basis.

The action of $H = T_{N_1} \circ \cdots \circ T_{N_s}$ is:

$$H(f) = [T_{N_1}] \cdots [T_{N_s}]c,$$

and the coefficients of $H(f) = \sum_{n=0}^{\infty} b_n^{(s)} q^n$ are:

$$b_n^{(s)} = \sum_{j=1}^{d} c_j^{(s)} a_n(f_j), \quad n = 0, \ldots, \ell - 1,$$

where $c^{(s)} = [T_{N_1}] \cdots [T_{N_s}]c$. This is equivalent to the coefficient formula:

$$b_n^{(s)} = \sum_{d|\gcd(P,n)} d^{k-1} a_{n/d}, \quad P = \prod_{i=1}^{s} N_i.$$

The total complexity is $O(d^\omega + s \cdot d^2 \cdot \ell \cdot \log N_i)$. Since Hecke operator matrices are typically sparse, this cost can be reduced to nearly $O(s \cdot d \cdot \log d)$ with sparse matrix multiplication techniques [12, 34], making it efficient to compute $H(f)$.

Remark that the first $\ell \geq B$ Fourier coefficients of $H(f)$, denoted $b_0^{(s)}, \ldots, b_{\ell-1}^{(s)}$ (which are $b_1, \ldots, b_{\ell-1}$), uniquely determine the coordinate vector $c^{(s)} = [T_{N_1}] \cdots [T_{N_s}]c$. This follows from the Sturm bound, $B \approx \frac{k \cdot [\mathrm{SL}_2(\mathbb{Z}):\Gamma_0(N)]}{12} \approx d$, which ensures that $\ell \geq B$ coefficients uniquely identify a modular form in $M_k(\Gamma_0(N))$. Specifically, the linear system:

$$\sum_{j=1}^{d} c_j^{(s)} a_n(f_j) = b_n^{(s)}, \quad n = 0, \ldots, \ell - 1,$$

has a unique solution for $c^{(s)} = (c_1^{(s)}, \ldots, c_d^{(s)})$ when $\ell \geq B$, solvable in $O(d^\omega)$ time. Thus, the choice of $\ell \geq B$ ensures that the coefficients of $H(f)$ uniquely determine $c^{(s)}$.

3.3. **A Natural Attack.** A natural attack using the module structure is as follows. Let $\{f_1, \ldots, f_d\}$ be a Hecke eigenbasis of $M_k(\Gamma_0(N))$. Suppose that the target modular form f lies entirely within a common eigenspace of all relevant Hecke operators $\{T_{N_i}\}$, for example, f is itself an eigenform. Then one can write

$$f = \sum_{j=1}^{d} c_j f_j, \quad H(f) = \sum_{j=1}^{d} \lambda_j c_j f_j,$$

where $\lambda_j = \prod_{i=1}^{s} \lambda_j(N_i)$ and $\lambda_j(N_i)$ is the eigenvalue of f_j under T_{N_i}.

Under this assumption, the attack aims to recover H by finding $x = (x_1, \ldots, x_t) \in \{0,1\}^t$ such that

$$\lambda_j = \prod_{i=1}^{t} \lambda_j(M_i)^{x_i},$$

where $\{M_1, \ldots, M_t\}$ denotes the domain of possible N_i values (e.g., products of three distinct primes).

However, this attack is fundamentally limited. Even if all λ_j and $\lambda_j(M_i)$ are known, the resulting system is a multiple subset product problem with $d \ll t$ constraints, making it highly underdetermined. In particular, when $d = 1$ (i.e., f is a single eigenform), this reduces to the standard subset product problem, equivalent to the X3C problem by Theorem 5, confirming NP-hardness. For general forms that are not eigenforms or do not lie in a common Hecke eigenspace, the action of H cannot be expressed as a simple product of eigenvalues, this attack is inapplicable.

3.4. **Relation to Isogeny Problems on Modular Curves.** Although our Hecke operator decomposition problem is formulated in algebraic terms, it admits a geometric interpretation when viewed through the lens of modular curves. This perspective helps illuminate why the problem is computationally interesting, while also highlighting its fundamental differences from standard isogeny problems, which are classically defined over $\mathbb{Q}$ (arithmetic) or $\mathbb{C}$ (geometric), and cryptographically defined over finite fields $\mathbb{F}_q$.

Let $X_0(N)$ be the modular curve that parametrizes isomorphism classes of pairs (E, C), where E is an elliptic curve and C is a cyclic subgroup of E of order N (corresponding to a cyclic N-isogeny). The Hecke operator T_p (for $p \nmid N$) acts on the function theory of $X_0(N)$ (specifically, on its space of modular forms) and has a well-known geometric description. Its action on a point (E, C) is given by:

$$T_p : (E, C) \mapsto \sum_{D \subset E[p], |D|=p} (E/D, (C + D)/D),$$

where the sum is over all cyclic subgroups D of E of order p. Intuitively, T_p maps a point to a formal sum of points obtained by performing all possible p-isogenies originating from E and transporting the level structure.

In this geometric framework, our problem can be loosely interpreted as follows: given a modular form f (which can be viewed as a function on $X_0(N)$) and its image $H(f) = (T_{N_1} \circ \cdots \circ T_{N_s})(f)$, recover the sequence of degrees $(N_1, \ldots, N_s)$. Geometrically, this corresponds to identifying the sequence of "averaging" operations over isogenies of degrees N_i that were applied to the function f.

This interpretation reveals both a conceptual similarity and a fundamental difference from the isogeny path problem. For similarity, both problems involve reasoning about compositions of operations related to isogenies of prime power degrees. For difference, the isogeny path problem asks for a *specific path* between two *points* (curves) in the isogeny graph. In contrast, our problem asks for the sequence of degrees used in a function operator H that acts by *averaging over all paths* of a given degree [29, 26]. The output $H(f)$ is a blend of information from all isogenies of the specified degrees, making it fundamentally different from the data provided by a single isogeny ϕ.

For example, consider the composite operator $H = T_6 \circ T_{15}$, which implies a composition of averaging operations over isogenies of degrees 6 and 15. The coefficients of $H(f)$ (e.g., the coefficient b_6 in the T_6 expansion) represent sums over all possible isogeny paths and do not pinpoint any specific kernel defining an isogeny. Conversely, if one were to find an isogeny ϕ of degree 90 connecting two curves, this would identify a specific kernel but would not reveal whether the decomposition of the degree was 6×15, 9×10, or any other factorization, which is information essential to recovering the operator H. This illustrates the fundamental discrepancy between the two problems.

This example highlights another crucial distinction: even when E and E' are isogenous (say, by an isogeny of degree 90), the transformed function $H(f)$ may still correspond to

the original curve E in the sense that the averaging process can preserve certain invariants. This further demonstrates that solving the isogeny path problem does not directly solve the Hecke operator decomposition problem.

We remark that this geometric interpretation does not involve the elliptic curve $E_f/\mathbb{Q}$ attached to f via the Eichler-Shimura correspondence [18, 29]. That construction relates the Fourier coefficients of f to the point counts of E_f over finite fields, which is an arithmetic property distinct from the geometric averaging process described above.

In summary, while the modular curve perspective provides valuable intuition, the Hecke problem is formulated purely in the algebraic setting of linear operators on spaces of modular forms. Its hardness relies on the complexity of decomposing the net effect of this averaging operation, a problem that is not known to be equivalent to finding isogeny paths between concrete elliptic curves. Moreover, isogeny problems in cryptography are typically defined over finite fields $\mathbb{F}_q$, rather than over $\mathbb{Q}$ or $\mathbb{C}$. These distinctions underscore the novelty of our Hecke-based framework and its independence from traditional isogeny- or group action-based cryptography.

4. The Key Exchange Protocol

Now we present our Diffie-Hellman-like key exchange from the Hecke problem. In particular, we will use the same type of integers $N_1, \ldots, N_s$ as in the X3C reduction for proving Theorem 5, and call this kind of integers, i.e., square-free integers that have precisely three prime factors, *3-square-free integers*.

4.1. **The Protocol.** Let $f \in M_k(\Gamma_0(N))$ be a public modular form. Let $S_A = \{p_1, \ldots, p_r\}$ and $S_B = \{q_1, \ldots, q_r\}$ be two public sets of prime numbers such that $p_i, q_i \nmid N$ and $S_A \cap S_B = \varnothing$. Our key exchange protocol is given by Figure 1.

Alice		**Bob**
Input: S_A, s, f		**Input:** S_B, s, f
Sample distinct 3-square-free integers $N_1, \ldots, N_s$ over S_A		Sample distinct 3-square-free integers $M_1, \ldots, M_s$ with over S_B
Compute $H_A(f) := \prod_{i=1}^{s} T_{N_i}(f)$		Compute $H_B(f) := \prod_{i=1}^{s} T_{M_i}(f)$
	$\xrightarrow{H_A(f)}$	
	$\xleftarrow{H_B(f)}$	
Output: $H_A(H_B(f))$		**Output:** $H_B(H_A(f))$

Figure 1. Hecke-KE.

The Diffie-Hellman-like diagram is given by Figure 2, where $H_{AB}(f) := \prod_{i=1}^{s} T_{N_i} T_{M_i}(f)$.

$$\begin{array}{ccc} f & \xrightarrow{H_A} & H_A(f) \\ {\scriptstyle H_B}\downarrow & & \downarrow{\scriptstyle H_B} \\ H_B(f) & \xrightarrow[H_A]{} & H_{AB}(f) \end{array}$$

Figure 2. Hecke Diffie-Hellman Diagram.

In the protocol, Alice and Bob sample distinct 3-square-free integers $N_1, \ldots, N_s$ and $M_1, \ldots, M_s$ factored over S_A and S_B respectively, and compute $H_A(f) = \prod_{i=1}^{s} T_{N_i}(f)$ and $H_B(f) = \prod_{i=1}^{s} T_{M_i}(f)$ respectively. They exchange $H_A(f)$ and $H_B(f)$. They then compute $H_A(H_B(f)) = \prod_{i=1}^{s} N_i(H_B(f))$ and $H_B(H_A(f)) = \prod_{i=1}^{s} M_i(H_A(f))$ respectively.

We prove that Alice and Bob eventually share a common secret.

Theorem 8. *In the protocol given by Figure 1, we have $H_A(H_B(f)) = H_B(H_A)(f)$.*

Proof. From commutativity of Hecke operators, i.e., $T_N T_M = T_M T_N$ for all $N, M \in \mathbb{Z}_{>0}$, it is straightforward that $H_A(H_B(f)) = H_B(H_A)(f) = \prod_{i=1}^{s} T_{N_i} T_{M_i}(f)$. □

4.2. **Security from Hecke Diffie-Hellman Problems.** We first define the *Hecke subset product problem* and show that it is computationally equivalent to the Hecke problem when N_i are 3-square-free integers. We then define the Hecke computational Diffie-Hellman and Hecke decisional Diffie-Hellman problems, and prove the IND-CPA security of our key exchange based on the decisional problem.

Definition 9. *Let $t, \ell \in \mathbb{N}$ with ℓ polynomial in t. Let $N_1, \ldots, N_t \in \mathbb{N}$, $x_1, \ldots, x_t \in \{0, 1\}$, and $f \in M_k(\Gamma_0(N))$. Denote $H = \prod_{i=1}^{m} T_{N_t}^{x_i} = T_{N_1}^{x_1} \circ \cdots \circ T_{N_t}^{x_t}$ as the composition of Hecke operators $T_{N_1}^{x_1}, \ldots, T_{N_t}^{x_t}$, where $T_{N_i}^{0}$ is the identity. The Hecke subset product problem is: given $N_1, \ldots, N_t$, f, and $H(f)$, find $x_1, \ldots, x_t$.*

The main difference between the Hecke problem and the Hecke subset product problem is that, in the Hecke problem, the potential N_i are not given, while in the Hecke subset product problem, the possible N_i are given. However, if each N_i is a 3-square-free integer factored over the same set of polynomially many known primes $p_1, \ldots, p_r$, then the domain of N_i in the Hecke problem is implicitly given because $\binom{r}{3} = \frac{n(n-1)(n-2)}{6}$ is polynomial in r, namely there are only polynomially many possible N_i and that they can be efficiently enumerated, making the two problems essentially the same problem.

Theorem 10. *Let $r, s, t, \ell \in \mathbb{N}$ be polynomial in each other. Let $P = \{p_1, \ldots, p_r\}$ be a set of known distinct primes. Suppose each integer N_i in the Hecke problem and Hecke subset product problem is a 3-square-free integer, i.e., $N_i = p_a p_b p_c$ for distinct $p_a, p_b, p_c \in P$. Then, the Hecke problem parametrized by r, s, ℓ is computationally equivalent to the Hecke subset product problem parametrized by r, t, ℓ.*

Proof. First, we reduce the Hecke problem to the Hecke subset product problem. Given an instance of the Hecke problem, namely the first ℓ coefficients of f, the first coefficient of $H(f)$, and the goal to find $N_1, \ldots, N_s$, we compute the set $C = \{p_a p_b p_c \mid p_a, p_b, p_c \in P, a < b < c\}$, which contains all 3-square-free integers over P. Since $|P| = r$, we have $|C| = \binom{r}{3} = \frac{n(n-1)(n-2)}{6}$. Enumerate $C = \{A_1, \ldots, A_t\}$ with $t = |C|$, and define a Hecke subset product problem instance: given $A_1, \ldots, A_t$, the first ℓ coefficients of f, and the first coefficient of $H(f)$, find $x_1, \ldots, x_t \in \{0, 1\}$ such that $\prod_{i=1}^{t} T_{A_i}^{x_i}(f)$ has the same first coefficient as $H(f)$. Solving this yields $x_1, \ldots, x_t$, and we set $N_1, \ldots, N_s$ as the A_i where $x_i = 1$, padding with repeated A_i if $\sum x_i < s$. The reduction is polynomial-time, as $|C| = O(r^3)$, and r, s, t, ℓ are polynomial. Correctness follows because C includes all possible N_i, and Hecke operators commute, ensuring $\prod_{i=1}^{t} T_{A_i}^{x_i}(f) = \prod_{i=1}^{s} T_{N_i}(f)$ when the x_i select the correct $A_i = N_i$.

Next, we reduce the Hecke subset product problem to the Hecke problem. Given an instance with $N_1, \ldots, N_t$, the first ℓ coefficients of f, the first coefficient of $H(f) = \prod_{i=1}^{t} T_{N_i}^{x_i}(f)$, and the goal to find $x_1, \ldots, x_t \in \{0, 1\}$, we construct a Hecke problem instance by setting $s = t$ and $N_i' = N_i$ for $i = 1, \ldots, s$. Define the Hecke problem as finding $N_1', \ldots, N_s'$ given the first ℓ coefficients of f and the first coefficient of $H(f) = \prod_{i=1}^{s} T_{N_i'}(f)$.

Solving this yields $N'_1, \ldots, N'_s$, and we set $x_i = 1$ if N_i appears in $N'_1, \ldots, N'_s$, otherwise $x_i = 0$. This reduction is polynomial-time, as it involves only renaming and solving an equivalent instance. Correctness holds because the solution $N'_1, \ldots, N'_s$ corresponds to the N_i with $x_i = 1$, and commutativity of Hecke operators ensures the first coefficient of $\prod_{i=1}^{s} T_{N'_i}(f)$ matches that of $\prod_{i=1}^{t} T_{N_i}^{x_i}(f)$.

In summary of both directions, the Hecke problem and Hecke subset product problem are computationally equivalent. □

Having the Hecke subset product problem as a DLP analogy, we now define the analogies of the computational and decisional Diffie-Hellman problems.

Definition 11. *Let $A_1, \ldots, A_t, B_1, \ldots, B_t$ be the $2t$ positive integers and $f \in M_k(\Gamma_0(N))$ be a modular form. Let $H_A = \prod_{i=1}^{t} T_{A_i}^{x_i}$, $H_B = \prod_{i=1}^{t} T_{B_i}^{y_i}$, where $x_1, \ldots, x_t, y_1, \ldots, y_t \in \{0,1\}$ are chosen uniformly at random. The Hecke computational Diffie-Hellman (Hecke-CDH) problem is: given $A_1, \ldots, A_t, B_1, \ldots, B_t$, f, $H_A(f)$, and $H_B(f)$, to compute $H_{AB}(f) = \prod_{i=1}^{t} T_{A_i}^{x_i} T_{B_i}^{y_i}(f)$.*

Definition 12. *Let $A_1, \ldots, A_t, B_1, \ldots, B_t$ be the $2t$ positive integers and $f \in M_k(\Gamma_0(N))$ be a modular form. Let $H_A = \prod_{i=1}^{t} T_{A_i}^{x_i}$, $H_B = \prod_{i=1}^{t} T_{B_i}^{y_i}$, where $x_1, \ldots, x_t, y_1, \ldots, y_t \in \{0,1\}$ are chosen uniformly at random. The Hecke decisional Diffie-Hellman (Hecke-DDH) problem is to distinguish between the following two distributions:*

$$D_0 := (f, H_A(f), H_B(f), H_{AB}(f)) \quad \text{and} \quad D_1 := (f, H_A(f), H_B(f), H_C(f)),$$

where $H_{AB}(f) = \prod_{i=1}^{t} T_{A_i}^{x_i} T_{B_i}^{y_i}(f)$ and $H_C(f) = \prod_{i=1}^{t} A_i^{z_i} B_i^{w_i}(f)$ with $z_1, \ldots, z_t, w_1, \ldots, w_t \in \{0,1\}$ chosen uniformly at random.

Now we prove IND-CPA security of our key exchange in the random oracle model, provided the shared secret is hashed to ensure sufficient entropy.

Theorem 13. *If Hecke-DDH is hard, the Hecke-KE protocol is IND-CPA secure in the random oracle model.*

Proof. The proof idea is standard. We show that for any adversary $\mathcal{A}$ with IND-CPA advantage ϵ, there exists a distinguisher $\mathcal{D}$ for Hecke-DDH with advantage 2ϵ.

In the IND-CPA game, the challenger generates a modular form $f \in M_k(\Gamma_0(N))$, selects integers $A_1, \ldots, A_t$, $B_1, \ldots, B_t$, and chooses random exponents $x_1, \ldots, x_t, y_1, \ldots, y_t \in \{0,1\}$. The challenger computes $H_A(f) = \prod_{i=1}^{t} T_{A_i}^{x_i}(f)$ and $H_B(f) = \prod_{i=1}^{t} T_{B_i}^{y_i}(f)$, then selects a bit $b \in \{0,1\}$. If $b = 0$, set $h = H_{AB}(f) = \prod_{i=1}^{t} T_{A_i}^{x_i} T_{B_i}^{y_i}(f)$; if $b = 1$, set $h = H_C(f) = \prod_{i=1}^{t} T_{A_i}^{z_i} T_{B_i}^{w_i}(f)$, where $z_i, w_i \in \{0,1\}$ are chosen uniformly at random. The challenger computes $K = \mathrm{Hash}(h)$ using the random oracle and sends $(f, H_A(f), H_B(f), K)$ to the adversary $\mathcal{A}$, who outputs a guess $b' \in \{0,1\}$. The IND-CPA advantage of the adversary is $\mathrm{Adv}(\mathcal{A}) = |\Pr[b' = b] - \frac{1}{2}| = \epsilon$.

We construct a Hecke-DDH distinguisher $\mathcal{D}$ that calls $\mathcal{A}$ to solve Hecke-DDH. Algorithm $\mathcal{D}$ receives a challenge tuple $(f, H_A(f), H_B(f), h)$, where $h = H_{AB}(f)$ if $b = 0$ or $h = H_C(f)$ if $b = 1$. To simulate the IND-CPA game, $\mathcal{D}$ computes $K = \mathrm{Hash}(h)$ using the random oracle and sends $(f, H_A(f), H_B(f), K)$ to $\mathcal{A}$. The adversary returns its guess b', and $\mathcal{D}$ outputs b'. If $h = H_{AB}(f)$ $(b = 0)$, then $K = \mathrm{Hash}(H_{AB}(f))$, matching the real key of Hecke-KE, so $\Pr[\mathcal{D} = 0 \mid h = H_{AB}(f)] = \Pr[\mathcal{A} = 0 \mid b = 0]$. If $h = H_C(f)$ $(b = 1)$, then $K = H(H_C(f))$, a hash of a random modular form, so $\Pr[\mathcal{D} = 0 \mid h = H_C(f)] = \Pr[\mathcal{A} = 0 \mid b = 1]$. The Hecke-DDH advantage of $\mathcal{D}$ is thus $\mathrm{Adv}(\mathcal{D}) = |\Pr[\mathcal{D} = 0 \mid h = H_{AB}(f)] - \Pr[\mathcal{D} = 0 \mid h = H_C(f)]| = |\Pr[\mathcal{A} = 0 \mid b = 0] - \Pr[\mathcal{A} = 0 \mid b = 1]|$.

The IND-CPA success probability is $\Pr[b' = b] = \frac{1}{2}\Pr[b' = 0 \mid b = 0] + \frac{1}{2}\Pr[b' = 1 \mid b = 1]$. Since $\Pr[b' = 1 \mid b = 1] = 1 - \Pr[b' = 0 \mid b = 1]$, we have $\Pr[b' =$

$b] = \frac{1}{2} + \frac{1}{2}(\Pr[b' = 0 \mid b = 0] - \Pr[b' = 0 \mid b = 1])$. Thus, the IND-CPA advantage is $\epsilon = |\Pr[b' = b] - \frac{1}{2}| = \frac{1}{2}|\Pr[b' = 0 \mid b = 0] - \Pr[b' = 0 \mid b = 1]|$. Therefore, $\mathrm{Adv}(\mathcal{D}) = |\Pr[\mathcal{A} = 0 \mid b = 0] - \Pr[\mathcal{A} = 0 \mid b = 1]| = 2\epsilon$. By the Hecke-DDH assumption, $\mathrm{Adv}(\mathcal{D})$ is negligible, so 2ϵ is negligible, implying ϵ is negligible. Hence, if the Hecke-DDH problem is hard, the Hecke-KE protocol is IND-CPA secure in the random oracle model. □

5. Experimental Results

We test efficiency, key sizes, and correctness. We test two ground fields, $\mathbb{Q}$ and $\mathbb{F}_p$, and compare against existing post-quantum schemes (Kyber, SIKE, Saber, NTRU, FrodoKEM, CSIDH, Classic-McEliece) at NIST security Levels 1, 2, 3, and 5.

The experiments were conducted on a MacBook Pro (14-inch, 2021) equipped with an Apple M1 Pro chip, 16 GB of memory, and macOS Sonoma (version 14.4.1). Storage was provided by a Macintosh HD with 230.88 GB available out of 494.38 GB total capacity. The experiments are performed under normal use of the laptop, with other applications running. The implementation used SageMath 10.3, with `sage.modular` for Hecke operator and q-expansion computations and `sage.combinat` for subset selection.

Specifically, we evaluate cusp forms over $\mathbb{Q}$ and $\mathbb{F}_p$, with coefficients of $H_{AB}(f)$ in $\mathbb{Q}$ reduced modulo a large prime before hashing to derive the shared key. The prime pool $S = \{p_1, \ldots, p_{128}\}$ consists of the first 128 primes, yielding $t = \binom{128}{3} = 970432$ many 3-subsets, forming a set C of 3-square-free integers $N_i = p_a p_b p_c$. This balances security (large t for high $\binom{t}{s}$ complexity) and computational feasibility.

Hecke operators are constructed as $H_A = \prod_{i=1}^{s} T_{N_i}$ and $H_B = \prod_{i=1}^{s} T_{M_i}$, with $s \in \{7, 10, 13, 20\}$. The combinatorial complexity is $\binom{970432}{s}$, ranging from 2^{139} ($s = 7$) to 2^{397} ($s = 20$). Classical and quantum security levels are defined by $\min\{\binom{t}{s}, p^B\}$ and $\min\{\sqrt{\binom{t}{s}}, p^{B/2}\}$, respectively, with p as the smallest prime greater than 2^{m+10} for $m \in \{64, 128, 256, 384\}$, and B is the Sturm bound. Security levels correspond to Level 1 ($s = 7, m = 64$), Level 3 ($s = 10, m = 128$), Level 5 ($s = 13, m = 256$), and Level 5+ ($s = 20, m = 384$).

To enhance efficiency, we precompute divisors for all N_i, caching results for $t = 970432$ integers. We use sparse representation for modular form coefficients and batch Hecke operator applications via matrix multiplication. Tests involve cusp forms with $k = 2$, $N \in \{23, 37, 53, 61\}$, coefficient bounds based on the Sturm bound $B \in \{4, 6, 9, 10\}$, and 100 runs per configuration.

Table 1 compares performance for various key exchange or key encapsulation mechanisms (KEM), sorted by security level and runtime.

The "Sec Lvl" column are NIST security levels.

The "Scheme" column are the names of the schemes with parameters, where "Hecke-KE $\mathbb{Q}$ N23 k2 s7 m64 B4" means the ground fields is $\mathbb{Q}$, the parameters N is 23, k is 2, s is 7, m is 64, and Sturm bound B is 4. Note that the table does not include all post-quantum key exchange schemes at every security level due to scheme-specific parameter designs and experimental constraints. For instance, SIKE [23] is listed at Levels 1, 2, 3, and 5, while NTRU-HPS, SABER [11], Kyber [27], FrodoKEM [1], and CSIDH [9] appear at Levels 1, 3, and 5, with no Level 2 configurations. Classic-McEliece [4] is only at Level 5, as its large key sizes are impractical for lower levels.

The "Time" column records the time it takes to complete a key exchange protocol. It shows that the efficiency of Hecke-KE is highly competitive, often outperforming the most efficient existing schemes. Also notice that our experiments are conducted on a MacBook

TABLE 1. Performance of Hecke-KE vs. Existing Post-Quantum Key Exchange

Sec Lvl	Scheme	Time (ms)	SK (byte)	PK (byte)	Succ
Level 1	Hecke-KE $\mathbb{Q}$ N23 k2 s7 m64 B4	0.83	17	32	100%
Level 1	Hecke-KE $\mathbb{F}_p$ N23 k2 s7 m64 B4	0.84	17	9	100%
Level 1	NTRU-HPS-2048-509 [10]	1.00	699	931	–
Level 1	LightSaber [11]	1.20	1568	672	–
Level 1	Kyber512 [27]	1.50	1632	800	–
Level 1	FrodoKEM-640-AES [1]	5.00	15632	9616	–
Level 1	SIKEp434 [23]	20.00	374	330	–
Level 1	CSIDH-512 [9]	80.00	32	64	–
Level 2	SIKEp503 [23]	30.00	402	378	–
Level 3	Hecke-KE $\mathbb{Q}$ N37 k2 s10 m128 B6	1.02	23	96	100%
Level 3	Hecke-KE $\mathbb{F}_p$ N37 k2 s10 m128 B6	1.05	23	105	100%
Level 3	NTRU-HPS-2048-677 [10]	1.30	935	1234	–
Level 3	Saber [11]	2.00	2304	992	–
Level 3	Kyber768 [27]	2.50	2400	1184	–
Level 3	FrodoKEM-976-AES [1]	8.00	24016	15688	–
Level 3	SIKEp610 [23]	50.00	462	462	–
Level 3	CSIDH-1024 [9]	120.00	48	80	–
Level 5	NTRU-HPS-4096-821 [10]	1.60	1230	1590	–
Level 5	Hecke-KE $\mathbb{F}_p$ N53 k2 s13 m256 B9	2.36	30	301	100%
Level 5	Hecke-KE $\mathbb{Q}$ N53 k2 s13 m256 B9	2.51	30	288	100%
Level 5	FireSaber [11]	2.80	3040	1312	–
Level 5	Kyber1024 [27]	3.50	3168	1568	–
Level 5	FrodoKEM-1344-AES [1]	12.00	32432	21520	–
Level 5	Classic-McEliece-6960119 [4]	15.00	13908	1047319	–
Level 5	SIKEp751 [23]	80.00	524	564	–
Level 5	CSIDH-1792 [9]	160.00	64	96	–
Level 5+	Hecke-KE $\mathbb{Q}$ N61 k2 s20 m384 B10	2.45	46	480	100%
Level 5+	Hecke-KE $\mathbb{F}_p$ N61 k2 s20 m384 B10	2.58	46	494	100%

Remarks: (1) The performance data of existing schemes is taken to algin with established benchmarks. For lattice-based constructions, the execution time of 1.5 ms for Kyber512 [7] aligns with the 1.0-2.0 ms range reported in the NIST official benchmarks [27], while the NTRU-HPS-2048 result of 1.0 ms is consistent with independent implementations [10]. The isogeny-based schemes have greater variability: the SIKEp434 result of 20 ms falls within the expected 10-30 ms window for optimized x64 implementations [23]. The CSIDH-512 result of 80 ms has a significant optimization beyond the original 100-200 ms baseline [9], aligning potential advancements like the CTIDH framework [3]. (2) The quantum security of CSIDH-512 is in fact below NIST Level 1; the Level 5 claim of CSIDH-1792 is also debated. (3) Hecke-KE runtimes are conservative due to SageMath and multitasking on M1 Pro. Others usually use optimized C/C++ on Skylake/Haswell. (4) To enhance Hecke-KE security, consider increasing N (e.g., 97, 151, 211) to achieve larger Sturm bounds ($B = 16, 25, 35$), increasing public key sizes to resist number-theoretic attacks.

Pro with the previously specified parameters, and the experiments run under typical multi-tasking conditions, with the system handling concurrent applications and processes, rather than being exclusively dedicated to the experiments. In contrast, other post-quantum key exchange schemes in Table 1, such as NTRU-HPS [10], Kyber [27], FrodoKEM [1], SIKE [23], CSIDH [9], Saber [17], and Classic-McEliece [4], are typically evaluated on high-performance x86-64 platforms, such as Intel or AMD processors (e.g., Intel Core i7 or Xeon, often with 16–32 GB RAM) running Linux or Windows. Again, the implementation of Hecke-KE used SageMath, utilizing `sage.modular` and `sage.combinat`, emphasizing mathematical precision over low-level optimization. In contrast, the existing schemes commonly use optimized C/C++ implementations, with some incorporating assembly for critical operations (e.g., Kyber's AVX2 optimizations) or reference implementations from NIST submissions. For instance, the NIST post-quantum cryptography standardization process often benchmarks schemes on standardized platforms like Intel Skylake or Haswell CPUs [27]. Even in this situation, the efficiency of Hecke-KE (0.83–2.58 ms) is highly competitive, surpassing or matching the performance of lattice-based schemes (1.00–3.50 ms).

The "SK" and "PK" columns are about secret key and public key sizes respectively. The secret key size of Hecke-KE is $s \cdot \lceil \log_2 \binom{r}{3} \rceil / 8 = s \cdot 20/8$ bytes, where $r = 128$, ranging from 17 bytes ($s = 7$) to 46 bytes ($s = 20$). The public key size of Hecke-KE is $B \cdot \lceil \log_2 p \rceil / 8 \approx B \cdot \lceil m/8 \rceil$ bytes, ranging from 9 bytes ($m = 64, \mathbb{F}_p$) to 494 bytes ($m = 384, \mathbb{F}_p$). Hecke-KE's compact public keys (9–494 bytes) significantly outperform all the other post-quantum key exchange schemes, including Classic-McEliece's extremely large public keys (261120–1047319 bytes) [4] and FrodoKEM's large keys (9616–21520 bytes) [1]. Its secret keys (17–46 bytes) are smaller than Kyber (1632–3168 bytes) [27], SABER (1568–3040 bytes) [17], Classic-McEliece (6452–13908 bytes) [4], and SIKE (374–524 bytes) [23, 22], but comparable to or slightly larger than CSIDH (32–64 bytes) [9]. Different approaches can be tried to further reduce the Hecke key sizes. For example, consider using a subset of C with larger s, or consider non-3-square-free integers N_i with an exponentially large domain C where we necessarily represent the secret key by the numbers $N_1, \ldots, N_i$ rather than the characteristic vector of the subset $\{N_1, \ldots, N_i\}$ of C, etc.

The "Succ" column are the rates that Alice and Bob successfully share the same key $H_A(H_B(f)) = H_B(H_A(f))$ within all 100 runs. They are 100% successful.

In general, Hecke-KE shows competitive efficiency and key sizes.

6. More Applications

The Hecke problem can be used to construct a variety of cryptographic primitives using typical ideas from discrete logarithm-based [24] or from isogeny-based schemes such as [13, 5, 14]. We describe ideas for public-key encryption (PKE), key encapsulation, digital signatures, and zero-knowledge proofs, mirroring discrete logarithm-based schemes such as ElGamal, Diffie-Hellman KEM, Schnorr signatures, and Schnorr proofs.

For *Hecke-PKE*, Alice uses the Hecke problem to encrypt a message $m \in \mathbb{F}_p$ for Bob, analogous to ElGamal encryption. Bob's public key is (f, g), where $g = H(f) \in M_k(\Gamma_0(N))$, and his private key is the Hecke operator H. Alice selects a random Hecke operator R, computes $c_1 = R(f)$, and encrypts m as $c_2 = m + \text{Hash}(R \circ H(f))$, where $\text{Hash} : M_k(\Gamma_0(N)) \to \mathbb{F}_p$ maps modular forms to field elements. She sends the ciphertext (c_1, c_2) to Bob. Bob applies H to compute $H(c_1) = R \circ H(f)$ and recovers $m = c_2 - \text{Hash}(H(c_1))$.

For *Hecke-KEM*, we mimic Diffie-Hellman KEM. Bob's public key is (f, g), with $g = H(f) \in M_k(\Gamma_0(N))$, and his private key is H. Alice chooses a random Hecke operator R, computes a ciphertext $c = R(f)$, and derives a key $K = \text{Hash}(R \circ H(f))$, where $\text{Hash} : M_k(\Gamma_0(N)) \to \mathbb{F}_p$. She sends c to Bob, who computes $H(c) = R \circ H(f)$ and obtains $K = \text{Hash}(H(c))$.

For *Hecke-Signature*, we mimic Schnorr signatures. Bob's private key is the Hecke operator H, and his public key is (f, g), where $g = H(f) \in M_k(\Gamma_0(N))$. To sign a message m, Bob selects a random Hecke operator R, computes $r = R(f)$, and forms a signature $\sigma = (r, u)$, where $u = R \circ H(f) + \text{Hash}(m, r) \cdot f$ and $\text{Hash} : \{0,1\}^* \times M_k(\Gamma_0(N)) \to \mathbb{F}_p$. Alice verifies by checking $u = H(r) + \text{Hash}(m, r) \cdot f$. The hardness of recovering H from f and g resists forgery.

For *Hecke-NIZK*, we mimic Schorr proofs. Alice proves to Bob that she knows a Hecke operator H such that $g = H(f) \in M_k(\Gamma_0(N))$, using the Hecke problem in a zero-knowledge Schorr proof. The public input is (f, g), and Alice's secret is H. She commits to $u = S(f)$, where S is a random Hecke operator, and responds to Bob's challenge $e \in \mathbb{F}_p$ with $Z = S \circ H^e$. Bob verifies that $Z(f) = u \circ H^e(f)$, confirming Alice's knowledge of H. For non-interactive proofs, Alice sets $e = \text{Hash}(u, f, g)$, where $\text{Hash} : M_k(\Gamma_0(N)) \times M_k(\Gamma_0(N)) \times M_k(\Gamma_0(N)) \to \mathbb{F}_p$. The difficulty of extracting H from f and g ensures zero-knowledge.

These are just the tip of the iceberg. A wide range of applications are expected.

7. Final Remarks

This work proposes the Hecke problem as a promising cryptographic foundation, rooted in the algebraic properties of modular forms different from the geometric structures in isogeny-based cryptography. Our Hecke-KE exhibits higher efficiency than SIDH and CSIDH, and exceptionally compact key sizes surpassing lattice-based schemes like Kyber and code-based schemes like Classic-McEliece. Its security, based on the Hecke-CDH and Hecke-DDH assumptions, requires further cryptanalysis to match the maturity of existing primitives from lattice theory or coding theory.

References

[1] Erdem Alkim et al. *FrodoKEM: Learning With Errors Key Encapsulation*. Tech. rep. NIST PQC Round 3 Submission, 2016.

[2] Josh Alman and Virginia Vassilevska Williams. "A refined laser method and faster matrix multiplication". In: *TheoretiCS* 3 (2024).

[3] Gustavo Banegas, Daniel J Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. "CTIDH: faster constant-time CSIDH". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2021.4 (2021), pp. 351–387.

[4] Daniel J. Bernstein et al. *Classic McEliece: Conservative Code-Based Cryptography*. Tech. rep. NIST PQC Round 4 Submission, 2021.

[5] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. "CSI-FiSh: efficient isogeny based signatures through class group computations". In: *International conference on the theory and application of cryptology and information security*. Springer. 2019, pp. 227–247.

[6] Richard E Borcherds. "Automorphic forms with singularities on Grassmannians". In: *Inventiones Mathematicae* 132.3 (1995), pp. 491–562.

[7] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. "CRYSTALS - Kyber: a CCA-secure module-lattice-based KEM". In: *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2018, pp. 353–367.

[8] Wouter Castryck and Thomas Decru. *An efficient key recovery attack on SIDH (preliminary version)*. Cryptology ePrint Archive, Paper 2022/975. `https://eprint.iacr.org/2022/975`. 2022. URL: `https://eprint.iacr.org/2022/975`.

[9] 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*. Springer. 2018, pp. 395–427.

[10] Cong Chen et al. *NTRU: A Ring-Based Public Key Cryptosystem*. Tech. rep. NIST PQC Round 3 Submission, 2020.

[11] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. *Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM*. `https://eprint.iacr.org/2018/230`. IACR Cryptology ePrint Archive, Report 2018/230. 2018.

[12] Timothy A Davis. *Direct methods for sparse linear systems*. SIAM, 2006.

[13] Luca De Feo and Steven D Galbraith. "SeaSign: compact isogeny signatures from class group actions". In: *Annual international conference on the theory and applications of cryptographic techniques*. Springer. 2019, pp. 759–789.
[14] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. "SQISign: compact post-quantum signatures from quaternions and isogenies". In: *International conference on the theory and application of cryptology and information security*. Springer. 2020, pp. 64–93.
[15] Fred Diamond and Jerry Shurman. *A First Course in Modular Forms*. Vol. 228. Graduate Texts in Mathematics. Springer, 2005. ISBN: 978-0-387-23229-3.
[16] Fred Diamond and Jerry Michael Shurman. *A first course in modular forms*. Vol. 228. Springer, 2005.
[17] Leo Ducas et al. *SABER: Mod-LWR Based KEM*. Tech. rep. NIST PQC Round 3 Submission, 2020.
[18] Martin Eichler. "Quaternäre quadratische Formen und die Riemannsche Vermutung für die Kongruenzzetafunktion". In: *Archiv der Mathematik* 5.1 (1954), pp. 355–366.
[19] Martin Eichler and Don Zagier. *The theory of Jacobi forms*. Vol. 55. Springer, 1985.
[20] Eberhard Freitag. "Hilbert modular forms". In: *Hilbert Modular Forms*. Springer, 1990, pp. 5–71.
[21] Helmut Hasse. "Zur Theorie der abstrakten elliptischen Funktionenkörper III: Die Struktur des Meromorphismenrings". In: *Journal für die reine und angewandte Mathematik* 175 (1936), pp. 193–208.
[22] David Jao and Luca De Feo. "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies". In: *Journal of Mathematical Cryptology* 4.3 (2011), pp. 287–325. DOI: 10.1515/jmc-2011-0009.
[23] David Jao et al. *Supersingular Isogeny Key Encapsulation*. Tech. rep. NIST PQC Round 3 Submission, 2020.
[24] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. 3rd. Chapman & Hall/CRC, 2020.
[25] Luciano Maino and Chloe Martindale. *An attack on SIDH with arbitrary starting curve*. Cryptology ePrint Archive, Paper 2022/1026. https://eprint.iacr.org/2022/1026. 2022. URL: https://eprint.iacr.org/2022/1026.
[26] Toshitsune Miyake. *Modular forms*. Springer Science & Business Media, 2006.
[27] National Institute of Standards and Technology. *FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard (Kyber)*. Tech. rep. NIST, 2023.
[28] Damien Robert. "Breaking SIDH in Polynomial Time". In: *Advances in Cryptology – EUROCRYPT 2023*. Ed. by Carmit Hazay and Martijn Stam. Cham: Springer Nature Switzerland, 2023, pp. 472–503. ISBN: 978-3-031-30589-4.
[29] Goro Shimura. *Introduction to the Arithmetic Theory of Automorphic Functions*. Princeton University Press, 1971.
[30] Goro Shimura. "On modular forms of half integral weight". In: *Annals of Mathematics* 97.3 (1973), pp. 440–481.
[31] William Stein. *Lectures on Modular Forms and Hecke Operators*. https://wstein.org/edu/2011/581g/ribet-stein/main.pdf. Lecture notes, University of Washington. 2011.
[32] William A Stein. *Modular forms, a computational approach*. Vol. 79. American Mathematical Soc., 2007.
[33] Andrew Wiles. "Modular elliptic curves and Fermat's last theorem". In: *Annals of Mathematics* 141.3 (1995), pp. 443–551. DOI: 10.2307/2118559.
[34] Raphael Yuster and Uri Zwick. "Fast sparse matrix multiplication". In: *ACM Transactions On Algorithms (TALG)* 1.1 (2005), pp. 2–13.