Computing endomorphism rings of supersingular elliptic curves and connections to pathfinding in isogeny graphs

Kirsten Eisenträger, Sean Hallgren, Chris Leonardi, Travis Morrison, and Jennifer Park

Abstract. Computing endomorphism rings of supersingular elliptic curves is an important problem in computational number theory, and it is also closely connected to the security of some of the recently proposed isogeny-based cryptosystems. In this paper we give a new algorithm for computing the endomorphism ring of a supersingular elliptic curve $E$ that runs, under certain heuristics, in time $O((\log p)^2 p^{1/2})$. The algorithm works by first finding two cycles of a certain form in the supersingular $\ell$-isogeny graph $G(p, \ell)$, generating an order $\Lambda \subseteq \text{End}(E)$. Then all maximal orders containing $\Lambda$ are computed, extending work of Voight [28]. The final step is to determine which of these maximal orders is the endomorphism ring. As part of the cycle finding algorithm, we give a lower bound on the set of all $j$-invariants $j$ that are adjacent to $j^p$ in $G(p, \ell)$, answering a question in [1].

We also give a new polynomial-time reduction from computing $\text{End}(E)$ to pathfinding in the $\ell$-isogeny graph which is simpler in several ways than previous ones. We show that this reduction leads to another algorithm for computing endomorphism rings which runs in time $\tilde{O}(p^{1/2})$. This allows us to break the second preimage resistance of a hash function in the family constructed in [5].

1. Introduction

Computing the endomorphism ring of an elliptic curve defined over a finite field is a fundamental problem in computational arithmetic geometry. For ordinary elliptic curves the fastest algorithm is given by Bisson and Sutherland [5] who gave a subexponential time algorithm to solve this problem. In the supersingular case, no subexponential time algorithm is known.

Computing endomorphism rings of supersingular elliptic curves is also closely connected to the security of some of the recently proposed isogeny-based cryptosystems [14, 11], and one proposal that advanced to Round 2 of the NIST postquantum competition [9] is based on isogenies [19, 2]. Using the supersingular $\ell$-isogeny graph for cryptographic purposes was first proposed in [8].

Computing the endomorphism ring of a supersingular elliptic curve $E$ was first studied by Kohel [21, Theorem 75], who gave an approach for generating a subring
of finite index of the endomorphism ring \( \text{End}(E) \). The algorithm was based on finding cycles in the \( \ell \)-isogeny graph of supersingular elliptic curves, and the running time of the probabilistic algorithm was \( O(p^{1+\varepsilon}) \). In [14] it is argued that heuristically one expects \( O(\log p) \) many calls to a cycle finding algorithm until the cycles generate \( \text{End}(E) \). An algorithm for computing cycles with complexity \( \tilde{O}(p^{1/2}) \) and polynomial storage is given by Delfs and Galbraith [10].

One can also compute \( \text{End}(E) \) using an isogeny \( \phi : E_0 \rightarrow E \), where \( E_0 \) is an elliptic curve with known endomorphism ring. McMurdy was the first to compute \( \text{End}(E) \) via such an approach [25], but did not determine its complexity. In [15] a polynomial-time reduction from computing \( \text{End}(E) \) to finding an isogeny \( \phi \) of powersmooth degree was given assuming some heuristics, while [13] used an isogeny \( \phi \) of \( \ell \)-power degree.

In this paper we give a new algorithm for computing the endomorphism ring of a supersingular elliptic curve \( E \): first we compute two cycles through \( E \) in the supersingular \( \ell \)-isogeny graph that generate an order \( \Lambda \) in \( \text{End}(\mathcal{O}) \). Then we compute all maximal orders \( \mathcal{O} \) that contain \( \Lambda \). The final step is to determine which of those superorders of \( \Lambda \) is the endomorphism ring of \( E \). Our algorithm for computing maximal superorders of \( \Lambda \) is efficient when \( \Lambda \) is local and Bass. This extends work of Voight who showed how to compute one maximal order containing \( \Lambda \) in polynomial time if the factorization of the discriminant of \( \Lambda \) is given [28, Theorem 7.14]. As part of the analysis of the cycle finding algorithm, we give a lower bound on the size of the set of all \( j \)-invariants \( j \) that are adjacent to \( j^p \) in \( G(p, \ell) \), answering the lower-bound part of Question 3 in [1].

Both the algorithm for generating the suborder of \( \text{End}(E) \) and the algorithm for computing the maximal orders containing a given order are new. However, our overall algorithm is still exponential: the two cycles are found in time \( O((\log p)^2 p^{1/2}) \), and the overall algorithm has the same running time, assuming several heuristics. This saves at least a factor of \( \log p \) versus the previous approach by [15] that uses cycles in \( G(p, \ell) \) to compute \( \text{End}(E) \). This is because with that approach one expects to compute \( O(\log p) \) many cycles, while the expected number of calls to an algorithm for computing cycles in \( G(p, \ell) \) in our new algorithm for computing \( \text{End}(E) \) is bounded by a constant, assuming that certain heuristics hold. Also, our cycle finding algorithm requires only polynomial storage, while achieving the same run time as a generic collision-finding algorithm, which require exponential storage.

In the last section of the paper we give a new polynomial-time reduction from computing \( \text{End}(E) \) to pathfinding in the \( \ell \)-isogeny graph which is simpler in several ways than previous ones. For this we need to assume GRH and the heuristics of [15]. We observe that this reduction, together with the algorithms in [22] [10] [15] [11] gives an algorithm for pathfinding in \( G(p, \ell) \) with \( O((\log p)^2 p^{1/2}) \) time and polynomial storage, assuming the various required heuristics for these algorithms [22] [15].

The paper is organized as follows. In Section 3 we give an algorithm for computing cycles in the \( \ell \)-isogeny graph \( G(p, \ell) \) so that the corresponding endomorphisms generate an order in the endomorphism ring of the associated elliptic curve. In Section 4 we show that this order can be enlarged locally to match the local data of a maximal order, and in Section 5 we construct all global maximal orders that satisfy these local conditions. In Section 6 we show how to determine which of these maximal orders is isomorphic to \( \text{End}(E) \). In Section 7 we give a reduction from the endomorphism ring problem to the problem of computing \( \ell \)-power isogenies.
in $G(p, \ell)$ that is then used to attack the second preimage resistance of the hash function in [8].

2. Background on Elliptic Curves and Quaternion Algebras

For the definition of an elliptic curve, its $j$-invariant, isogenies of elliptic curves, their degrees, and the dual isogeny see [26].

2.1. Endomorphism rings, supersingular curves, $\ell$-power isogenies. Let $E$ be an elliptic curve defined over a finite field $\mathbb{F}_q$. An isogeny of $E$ to itself is called an *endomorphism* of $E$. The set of endomorphisms of $E$ defined over $\mathbb{F}_q$ together with the zero map is called the endomorphism ring of $E$, and is denoted by $\text{End}(E)$.

If the endomorphism ring of $E$ is non-commutative, $E$ is called a *supersingular elliptic curve*. Otherwise we call $E$ *ordinary*. Every supersingular elliptic curve over a field of characteristic $p$ has a model that is defined over $\mathbb{F}_p^{\times}$ because the $j$-invariant of such a curve is in $\mathbb{F}_p^{\times}$.

Let $E, E'$ be two supersingular elliptic curves defined over $\mathbb{F}_p^{\times}$. For each prime $\ell \neq p$, $E$ and $E'$ are connected by a chain of isogenies of degree $\ell$. By [21, Theorem 79], $E$ and $E'$ can be connected by $m$ isogenies of degree $\ell$, where $m = O(\log p)$. For $\ell$ a prime different from $p$, the *supersingular $\ell$-isogeny graph* in characteristic $p$ is the multi-graph $G(p, \ell)$ whose vertex set is

$$V = V(G(p, \ell)) = \{j \in \mathbb{F}_p^{\times} : j = j(E) \text{ for } E \text{ supersingular}\},$$

and the number of directed edges from $j$ to $j'$ is equal to the multiplicity of $j'$ as a root of $\Phi_\ell(j, Y)$. Here, given a prime $\ell$, $\Phi_\ell(X, Y) \in \mathbb{Z}[X, Y]$ is the *modular polynomial*. This polynomial has the property that $\Phi_\ell(j, j') = 0$ for $j, j' \in \mathbb{F}_q$ and $q = p^r$ if and only if there exist elliptic curves $E(j), E(j')$ defined over $\mathbb{F}_q$ with $j$-invariants $j, j'$ such that there is a separable $\ell$-isogeny from $E(j)$ to $E(j')$.

2.2. Quaternion Algebras, orders and sizes of orders. For $a, b \in \mathbb{Q}^\times$, let $H(a, b)$ denote the quaternion algebra over $\mathbb{Q}$ with basis $1, i, j, ij$ such that $i^2 = a, j^2 = b$ and $ij = -ji$. That is, $H(a, b) = \mathbb{Q} + \mathbb{Q}i + \mathbb{Q}j + \mathbb{Q}ij$. Any quaternion algebra over $\mathbb{Q}$ can be written in this form. There is a *canonical involution* on $H(a, b)$ which sends an element $\alpha = a_1 + a_2i + a_3j + a_4ij$ to $\overline{\alpha} := a_1 - a_2i - a_3j - a_4ij$. Define the *reduced trace* of an element $\alpha$ as above to be $\text{Trd}(\alpha) = \alpha + \overline{\alpha} = 2a_1$, and the *reduced norm* to be $\text{Nrd}(\alpha) = \alpha\overline{\alpha} = a_1^2 - a_2^2 - a_3^2 + a_4^2$.

A subset $I \subseteq H(a, b)$ is a *lattice* if $I$ is finitely generated as a $\mathbb{Z}$-module and $I \otimes \mathbb{Q} \simeq H(a, b)$. If $I \subseteq H(a, b)$ is a lattice, the *reduced norm of $I$, denoted $\text{Nrd}(I)$, is the positive generator of the fractional $\mathbb{Z}$-ideal generated by $\{\text{Nrd}(\alpha) : \alpha \in I\}$. An *order* $\mathcal{O}$ of $H(a, b)$ is a subring of $H(a, b)$ which is also a lattice, and if $\mathcal{O}$ is not properly contained in any other order, we call it a *maximal order*.

We define $\mathcal{O}_R(I) := \{x \in H(a, b) : 1x \subseteq I\}$ to be the *right order of the lattice $I$*, and we similarly define its *left order $\mathcal{O}_L(I)$*. If $\mathcal{O}$ is a maximal order in $H(a, b)$ and $I \subseteq \mathcal{O}$ is a left ideal of $\mathcal{O}$, then $\mathcal{O}_R(I)$ is also a maximal order. The *size of a lattice* or an order $\Lambda$ in a quaternion algebra $B$ is the size of the coefficients of the basis specifying $\Lambda$ plus the size of $B$, which is specified by a basis and a multiplication table. We denote by $B_{p, \infty}$, the unique quaternion algebra over $\mathbb{Q}$ that is ramified exactly at $p$ and $\infty$. The endomorphism ring of a supersingular elliptic curve is isomorphic to a maximal order in $B_{p, \infty}$. 
2.3. Bass, Eichler, and Gorenstein orders in quaternion algebras; discriminants and reduced discriminants. Let $B$ be a quaternion algebra over $\mathbb{Q}$. We define the discriminant of $B$, denoted $\text{disc} B$, to be the product of primes that ramify in $B$; then $\text{disc} B$ is a squarefree positive integer. If $O \subseteq B$ is an order, we define the discriminant of $O$ to be $\text{disc}(O) := |\det(\text{Trd}(\alpha_1\alpha_j))_{i,j}| \in \mathbb{Z} > 0$, where $\alpha_1, \ldots, \alpha_4$ is a $\mathbb{Z}$-basis for $O$.

The discriminant of an order is always a square, and the reduced discriminant $\text{discr}(O)$ is the positive integer square root so that $\text{discr}(O)^2 = \text{disc}(O)$. The discriminant of an order measures how far the order is from being a maximal order. The order $O$ is maximal if and only if $\text{discr}(O) = \text{disc} B$. Associated to a quaternion algebra $B$ over $\mathbb{Q}$ there is also a discriminant form $\Delta : B \to \mathbb{Q}$, defined by $\Delta(\alpha) = \text{Trd}(\alpha)^2 - 4 \text{Nrd}(\alpha)$. Now let $O \subseteq B$ be a $\mathbb{Z}$-order. We say that $O$ is an Eichler order if $O \subseteq B$ is the intersection of two (not necessarily distinct) maximal orders. The codifferent of an order is the lattice $\text{codiff}(O)$ obtained as the dual of the trace pairing, i.e., $\text{codiff}(O) = \{\alpha \in B : \text{Trd}(\alpha O) \subseteq \mathbb{Z}\}$. We say that $O$ is Gorenstein if $\text{codiff}(O)$ is invertible, and that an order $O$ is Bass if every superorder $O' \supseteq O$ is Gorenstein. An order is basic if it contains a commutative, quadratic subalgebra $R$ such that $R$ is integrally closed in $\mathbb{Q}R$. When $B$ is a quaternion algebra over $\mathbb{Q}_p$ and $O$ is a $\mathbb{Z}_p$-order in $B$, we similarly define lattices, ideals, and the various types of orders in $B$.

3. Computing an order in the endomorphism ring of a supersingular elliptic curve

3.1. Computing cycles in $G(p, \ell)$. Fix a supersingular elliptic curve $E_0$ defined over $\mathbb{F}_{p^2}$ with $j$-invariant $j_0$. In this section we describe and analyze an algorithm for computing two cycles through $j_0$ in $G(p, \ell)$ that generate an order in $\text{End}(E_0)$.

We will first show how to construct two distinct paths from $j_0$ to $j_0'$. Given two such paths $P$ and $P'$, then first traversing through $P$ and then traversing through $P'$ in reverse gives a cycle through $j_0$. This uses the fact that if $j$ is adjacent to $j'$, then $jP$ is adjacent to $(j')P$.

Now let $P_1$ be a path of length $k$ from $j_0$ to some other vertex $j_k$ in $G(p, \ell)$. Denote the not necessarily distinct vertices on the path by $j_0, j_1, \ldots, j_k$ and assume that $j_k$ is adjacent to $j_0'' \in G(p, \ell)$. Let $P_{1}'''' = [j_k, j_{k-1}''', j_k', j_{k-1}''$, $\ldots, j_k', j_0''']$. The concatenation $P := P_1 P_{1}''''$ is a path from $j_0$ to $j_0''''$.

If $j_0 = j_0''''$, then $P$ is already a cycle starting at $j_0$, so this gives a cycle through $j_0$. Otherwise, one can repeat this process to find another path $P'' := P_2 P_{2}''''$ (with possibly a different $j_k$) that passes through at least one vertex not in $P$. Concatenating $P$ and $P''$ (in reverse order) gives a cycle starting and ending at $j_0$; this corresponds to an endomorphism of $E$. We will need the notion of a path/cycle with no backtracking and trimming a path/cycle to remove backtracking.

Definition 3.1. Suppose $e_j, e_j'$ are edges in $G(p, \ell)$ that correspond to $\ell$-isogenies $\phi_j : E(j) \to E(j')$ and $\phi_j' : E(j') \to E(j)$ between curves $E(j)$ and $E(j')$ with $j$-invariants $j, j'$. We say that $e_j$ is dual to $e_j'$ if up to isomorphism $\phi_{j'}$ equals the dual isogeny $\phi_j$ of $\phi$. That is $\phi_{j'} = \alpha \phi_j$, where $\alpha \in \text{Aut}(E(j))$. We say that a path or cycle with a specified start vertex $j_0$, following edges $(e_1, \ldots, e_k)$ and ending at vertex $j_k$ has no backtracking if $e_{i+1}$ is not dual to $e_i$ for $i = 1, \ldots, k-1$.
In our definition, a cycle has a specified start vertex \( j_0 \), and we consider a cycle a special case of a path starting and ending at \( j_0 \). So according to our definition, even if the first edge \( e_1 \) and the last edge \( e_k \) are dual to each other in a cycle, it is not considered backtracking.

**Definition 3.2.** Given a path \((e_1, \ldots, e_k)\) from \( j_0 \) to \( j_k \) (with \( j_0 \neq j_k \)) or a cycle with specified start vertex \( j_0 = j_k \), define *trimming* as the process of iteratively removing pairs of adjacent dual edges until none are left.

One can show that given a path \( P \) from \( j_0 \) to \( j_k \) with \( j_0 \neq j_k \), or a cycle \( C \) with start vertex \( j_0 = j_k \), the trimmed versions \( \tilde{P} \) or \( \tilde{C} \) may result in a smaller set of vertices. The vertices \( j_0 \) and \( j_k \) will still be there in \( \tilde{P} \), and the only way that \( j_0 \) and \( j_k \) may disappear from \( \tilde{C} \) is if the whole cycle gets removed.

**Definition 3.3.** Given a path \( P \) in \( G_{p,\ell} \) from \( j_0 \) to \( j_k \), we define \( P^R \) to be the path \( P \) traversed in reverse order, from \( j_k \) to \( j_0 \), using the dual isogenies.

Let \( S^p := \{ j \in \mathbb{F}_{p^2} : j \text{ is supersingular and } j \text{ is adjacent to } j^p \in G(p,\ell) \} \).

We can now give the algorithm to find cycle pairs:

**Algorithm 3.4.** Finding cycle pairs for prime \( \ell \)

Input: prime \( p \neq \ell \) and a supersingular \( j \)-invariant \( j_0 \in \mathbb{F}_{p^2} \).

Output: two cycles in \( G(p,\ell) \) through \( j_0 \).

1. Perform \( N = \Theta(\sqrt{\log p \log \log p}) \) random walks of length \( k = \Theta(\log(p^{1/4}(\log \log p)^{1/2})) \) starting at \( j_0 \) and select a walk that hits a vertex \( j_k \in S^p \), i.e. such that \( j_k \) is \( \ell \)-isogenous to \( j_k^p \); let \( P_1 \) denote the path from \( j_0 \) to \( j_k \).
2. Let \( P_1^p \) be the path given by \( j_k, j_k^p, j_k^p, \ldots, j_0^p \).
3. Let \( P \) denote the path from \( j_0 \) to \( j_0^p \) given as the concatenation of \( P_1 \) and \( P_1^p \).

4. If \( j_0 \in \mathbb{F}_p \) then \( P_1 P_1^p \) is a cycle through \( j_0 \). Remove any self-dual self-loops and backtracking from \( P_1 P_1^p \).
5. If \( j_0 \in \mathbb{F}_{p^2} \neq \mathbb{F}_p \) repeat steps (1)-(3) again to find another path \( P' = P_2 P_2^p \) from \( j_0 \) to \( j_0^p \), then \( P(P')^R \) is a cycle. Remove any self-dual self-loops and backtracking from the cycle.
6. Repeat Steps (1)-(5) a second time to get a second cycle.

**Remark 3.5.** Instead of searching for a vertex \( j \) in Step (1) such that \( j \) is adjacent to \( j^p \), one could also search for a vertex \( j \in \mathbb{F}_p \), i.e. \( j = j^p \) or a vertex \( j \) whose distance from \( j^p \) in the graph is bounded by some fixed integer \( B \). Our algorithm that searches for a vertex such that \( j \) is adjacent to \( j^p \) was easier to analyze because there were fewer cases to consider in which the trimmed cycles would not generate an order.

To analyze the running time of Algorithm 3.4 we will use the mixing properties in the Ramanujan graph \( G(p,\ell) \). This is captured in the following proposition, which is an extension of [20] Lemma 2.1 in the case that \( G(p,\ell) \) is not regular or undirected (that is, when \( p \neq 1 \mod 12 \)).

**Proposition 3.6.** Let \( p > 3 \) be prime, and let \( \ell \neq p \) also be a prime. Let \( S \) be any subset of the vertices of \( G(p,\ell) \) not containing 0 or 1728. Then a random walk of
length at least

\[ t = \frac{\log \left( \frac{p}{6|S|^2} \right)}{\log \left( \frac{c+1}{\sqrt{2}} \right)} \]

will land in \( S \) with probability at least \( \frac{6|S|}{p} \).

One can prove this since the eigenvalues for the adjacency matrix of \( G(p, \ell) \) satisfy the Ramanujan bound. This allows us to prove the following theorem.

**Theorem 3.7.** Under GRH, Algorithm 3.4 computes two cycles in \( G(p, \ell) \) through \( j_0 \) that generate an order in the endomorphism ring of \( E_0 \) in time \( O(\sqrt{p} \log p)^2 \), as long as the two cycles do not pass through the vertices 0 or 1728, with probability \( 1 - O(1/p) \). The algorithm requires \( \text{polylog} p \) space.

**Remark 3.8.** In Section 6 we use this proposition to compute endomorphism rings, and from this point there is no problem with excluding paths through 0 or 1728. This is because the endomorphism rings of the curves with \( j \)-invariants 0 and 1728 are known, and a path of length \( \log P \), starting at \( j_0 \) going through 0 or 1728 lets us compute \( \text{End}(E_0) \) via the reduction in Section 7.

**Proof.** Assuming GRH, by Theorem 3.9 below, \( |S^p| = \Omega(\sqrt{p} \log p) \) (treating \( \ell \) as a constant). Proposition 3.6 implies that the endpoint \( j_k \) of a random path found in Step (1) is in \( S^p \) with probability \( \Omega(1/(\sqrt{p} \log p)) \). The probability that none of the \( N + 1 \) paths lands in \( S^p \) is at most \((1 - C/(\sqrt{p} \log p))^{N+1} \leq (1 + C/(\sqrt{p} \log p))^{-N+1} \leq e^{-c \log p/C} = O(1/p) \) if \( c = C \), where \( C \) is from Theorem 3.9 and \( c \) is the constant used in the choice of \( N \).

If \( j_0 \in \mathbb{F}_p \), then \( P_1P_1^p \) is a cycle through \( j_0 \) since \( j_0 = j_0^p \). If \( j_0 \in \mathbb{F}_{p^2} - \mathbb{F}_p \), then Step (5) computes a second path from \( j_0 \) to \( j_0^p \), which is concatenated to the first path, but in reverse, forming a cycle \( C_0 \). In Step 6, a second cycle \( C_1 \) is found.

Now we must show that with high probability the two cycles \( C_0, C_1 \) returned by the algorithm are linearly independent. We will use Corollary 4.12 of [3]. This corollary states that two cycles \( \tilde{C}_0, \tilde{C}_1 \) with no self-loops generate an order inside \( \text{End}(E_0) \) if (1) they do not go through 0 or 1728, (2) have no backtracking, and (3) have the property that one cycle contains a vertex that the other does not contain. Let \( \tilde{C}_0, \tilde{C}_1 \) be the trimmed versions of the cycles \( C_0, C_1 \) returned by Algorithm 3.4. It is easy to see that if the trimmed cycles \( \tilde{C}_0, \tilde{C}_1 \) are linearly independent, then so are the original cycles \( C_0, C_1 \). Hence it is enough to show that the trimmed cycles \( \tilde{C}_0, \tilde{C}_1 \) satisfy the conditions of Corollary 4.12 of [3].

To prove this, we first claim that with high probability, the end vertex \( j_k \in S^p \) in the path \( P_1 \) from \( j_0 \) to \( j_k \) will not get removed if the path \( P_1P_1^p \) is trimmed. Then we show it’s also still there in the trimmed cycle. Observe that if the path \( P_1 \) were to be trimmed to obtain a path \( \tilde{P}_1 \) with no backtracking, then \( \tilde{P}_1 \) is still a nontrivial path that starts at \( j_0 \) and ends at \( j_k \) as long as \( j_0 \) and \( j_k \) are different which occurs with probability \( 1 - O(1/p) \). After concatenating \( \tilde{P}_1 \) with its corresponding path \( \tilde{P}_1^p \), the path \( \tilde{P}_1\tilde{P}_1^p \) has backtracking only if the last edge of \( \tilde{P}_1 \) is dual to the first edge in \( \tilde{P}_1^p \), i.e. if \( j_{k-1} = j_k^p \). If that is the case, remove the last edge from \( \tilde{P}_1 \) and the first edge from \( \tilde{P}_1^p \), and call the remaining path \( \tilde{P}_1 \). The new path \( \tilde{P}_1 \) still has the property that it ends in a vertex \( j = j_k^p \) that is \( \ell \)-isogenous to its conjugate \( (j_k^p)^p = j_k \). After concatenating \( \tilde{P}_1 \) with its corresponding \( \tilde{P}_1^p \), this still gives a path
from $j_0$ to $j_0^p$. Again, the concatenation of these two paths has no backtracking unless the last edge in $\tilde{P}_1$ is the first edge in $\tilde{P}_1^p$, i.e. if the last edge in $\tilde{P}_1$ is an edge from $j_k$ to $j_k^p$. But this cannot happen, because otherwise the trimmed path $\tilde{P}_1$ would have backtracking because it would go from $j_k$ to $j_k^p$ and back to $j_k$, contradicting the definition of a trimmed cycle. (With negligible probability, the vertex $j_k$ has multiple edges, so we exclude this case here.) Hence the trimmed version of $P_1P_1^p$ is $\tilde{P}_1\tilde{P}_1^p$, and this path still contains the vertex $j_k$, since $\tilde{P}_1^p$ contains the vertex $j_k$. Now we can finish the argument by considering two cases:

Case 1: $j_0 \in \mathbb{F}_p$. The above argument about trimming shows that if the vertex $j_k$ appearing in the second cycle $C_1$ is different from all the vertices appearing in $C_0$ and their conjugates, which happens with probability $1 - O(\log p/p)$, then that vertex $j_k$ will appear in the trimmed cycle $C_1$, but not in $\tilde{C}_0$. (This is because in this case the trimmed path $\tilde{P}_1P_1^p$ is already a cycle.) Hence by [3, Corollary 4.12], $\tilde{C}_0$ and $\tilde{C}_1$ are linearly independent.

Case 2: If $j_0 \in \mathbb{F}_p^2 - \mathbb{F}_p$, then with probability $1 - O(\log(p)/p)$, the endpoint $j_k$ of $P_2$ is a vertex such that it or its conjugate do not appear as a vertex in $P_1$. The concatenation of the two paths $P = P_1P_1^p$ and $P' = P_2P_2^p$ in reverse is a cycle $C_0$ through $j_0$. When we trim it, it is still a cycle through $j_0$ in which the endpoint $j_k$ from $P_2$ appears because that $j_k$ or its conjugate did not appear in $P_1$. Similarly, Algorithm 3.4 finds a second cycle $C_1$ with probability $1 - \log(p)/p$ that contains a random vertex that was not on the first cycle $C_0$. This means that by Corollary 4.12 of [3], $\tilde{C}_0$ and $\tilde{C}_1$ and hence $C_0$ and $C_1$ are linearly independent.

The running time is $O(\sqrt{p}(\log p)^2)$ because we are considering $O(\sqrt{p})$ paths of length $O(\log p)$, and going from one vertex to the next takes time polynomial in $\ell \log p$, and we are assuming that $\ell$ is fixed here. The storage is polynomial in $\log p$ because we only have to store the paths $P_1, P_2$ that lands in $S^p$ and can delete all other ones.

\[ \square \]

### 3.2. Determining the size of $S^p$.

Will now determine a lower bound for the size of the set $S^p := \{ j \in \mathbb{F}_p^2 : j \text{ is supersingular and } j \text{ is adjacent to } j^p \text{ in } G(p, \ell) \}$. In [8] Section 7, an upper bound is given for $S^p$, but in order to estimate the chance that a path lands in $S^p$ we need a lower bound for this set.

Let $\ell, p$ be primes such that $\ell < p/4$. Let $\mathcal{O}_K$ be the ring of integers of $K := \mathbb{Q}(\sqrt{-\ell p})$. We use the terminology and notation in in [12, 4]. Let $\text{Emb}_{\mathcal{O}_K}(\mathbb{F}_p^2)$ be the collection of pairs $(E, f)$ such that $E$ is an elliptic curve over $\mathbb{F}_p^2$ and $f : \mathcal{O}_K \to \text{End}(E)$ is a normalized embedding, taken up to isomorphism. We say $f : \mathcal{O}_K \to \text{End}(E)$ is normalized if each $\alpha \in \mathcal{O}_K$ induces multiplication by its image in $\mathbb{F}_p^2$ on the tangent space of $E$, and $(E, f)$ is isomorphic to $(E', f')$ if there exists an isomorphism $g : E \to E'$ such that $f(\alpha)' = gf(\alpha)g^{-1}$ for all $\alpha \in \mathcal{O}_K$.

**Theorem 3.9.** Assume that $\ell < p/4$. Let

$$S^p = \{ j \in \mathbb{F}_p^2 : j \text{ is supersingular and } \Phi_\ell(j, j^p) = 0 \}. $$

Under GRH there is a constant $C > 0$ (depending on $\ell$) such that $|S^p| > C \frac{\sqrt{p}}{\log \log(p)}$.

**Proof.** First, if $E$ is a supersingular elliptic curve defined over $\mathbb{F}_p^2$ with $j$-invariant $j$ and $E^{(p_i)}$ is a curve with $j$-invariant $j^p$ and $\ell < p/4$ is also a prime, then $E$ is $\ell$-isogenous to $E^{(p_i)}$ if and only if $\mathbb{Z}[\sqrt{-\ell p}]$ embeds into $\text{End}(E)$ (see [8], Lemma 6).
For any element \((E, f) \in \text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})\), \(E\) is supersingular, since \(p\) ramifies in \(\mathbb{Q}(\sqrt{-\ell p})\). Moreover \(j(E) \in S^p\) by the above fact. Thus the map \(\rho: \text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2}) \to S^p\) that sends \((E, f)\) to \(\rho(E, f) = j(E)\) is well-defined.

To get a lower bound for \(S^p\) we will show that for \(j \in S^p\), the size of \(\rho^{-1}(j)\) is bounded by \((\ell + 1) \cdot 6\) and that \(|\text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})| \gg \sqrt{\ell p} / \log \log (\ell p)\). These two facts imply that

\[
|S^p| \geq |\text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})|/((\ell + 1) \cdot 6) > \frac{1}{(\ell + 1) \cdot 6} \cdot \frac{\sqrt{\ell p}}{\log \log (\ell p)}.
\]

To get a lower bound for \(|\text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})|\) we can use Proposition 2.7 of \([16]\) to show that this order equals \(|\text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})| = |\text{Cl}(\mathcal{O}_K)|\). Class group estimates from \([24]\) give \(|\text{Cl}(\mathcal{O}_K)| = h(-\ell p) \gg \sqrt{\ell p}\).

It remains to bound the size of \(\rho^{-1}(j)\). We claim that an equivalence class of pairs \((E, f)\) determines an edge in \(G(p, \ell)\). Let \([([E, f]) \in \text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})\) and choose a representative curve \(E\). First assume that \(j(E) \neq 0, 1728\). Then \((E, f) \simeq (E, g)\) implies that \(f = g\), since \(\text{Aut}(E) = \pm 1\). Thus we may identify \([([E, f])\) with the edge in \(G(p, \ell)\) corresponding to the kernel of \(f(\sqrt{-\ell p})\). When \(j(E) = 0\) or 1728, we may assume that \(E\) is defined over \(\mathbb{F}_p\). Then let \([([E, f]) \in \text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})\) and suppose \((E, f)\) is equivalent to \((E, g)\). We can factor \(f(\sqrt{-\ell p}) = \pi \circ \phi\) and \(g(\sqrt{-\ell p}) = \pi \circ \phi'\), where \(\phi, \phi'\) are degree \(\ell\) endomorphisms of \(E\) and \(\pi\) is the Frobenius endomorphism of \(E\). Additionally, \(\pi \phi = u \pi \phi' u^{-1}\). We claim that \(u\) and \(\phi\) commute. If not, then they generate an order \(\Lambda\) such that the following formula holds (see \([23]\)):

\[
discr(\Lambda) = \frac{1}{4}(\text{disc}(u) \text{disc}(\phi) - (\text{Trd}(u) \text{Trd}(\phi) - 2 \text{Trd}(u\phi))^2) \leq \frac{1}{4}(\text{disc}(u) \text{disc}(\phi)).
\]

One can show that this contradicts our assumption that \(p/4 > \ell\). Thus \(u\) and \(\phi\) commute, and we see that \(f(\sqrt{-\ell p})\) and \(g(\sqrt{-\ell p})\) have the same kernel and thus determine the same edge in \(G(p, \ell)\).

We now count how many elements of \(\text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})\) determine the same edge in \(G(p, \ell)\). Suppose that \([([E, f], ([E, g]) \in \text{Emb}_{\mathcal{O}_K}(\mathbb{F}_{p^2})\) and that \(\ker(f(\sqrt{-\ell p})) = \ker(g(\sqrt{-\ell p}))\). Writing \(f(\sqrt{-\ell p}) = \phi \circ \text{Frob}\) and \(g(\sqrt{-\ell p}) = \phi' \circ \text{Frob}\) we see that \(\phi\) and \(\phi'\) must have the same kernel as well. Thus \(\phi' = u\phi\) for some \(u \in \text{Aut}(E)\).

Because \(p > 4\ell > 3\), \(\text{Aut}(E) \leq 6\) and we conclude that there are at most 6 classes \([([E, f])\) determining the same edge emanating from \(E\) in \(G(p, \ell)\). Thus \(|\rho^{-1}(j)| \leq (\ell + 1) \cdot 6\). \(\square\)

Assuming GRH, this result settles the lower-bound portion of Question 3 in \([1]\). See Lemma 6 of \([8]\) regarding the upper-bound.

4. Enumerating maximal superorders: the local case

Let \(q\) be a prime. In this section, we give an algorithm for the following problem:

**Problem 1.** Given a \(\mathbb{Z}_q\)-order \(\Lambda_q \subseteq M_2(\mathbb{Q}_q)\), find all maximal orders containing \(\Lambda_q\).

The following algorithm solves Problem 1. To compute the maximal superorders of a \(\mathbb{Z}_q\)-order, one uses the theory of the Bruhat-Tits tree (§23.5, \([27]\)). Given an Eichler order \(\Lambda_q = \mathcal{O} \cap \mathcal{O}'\), the maximal orders containing \(\Lambda_q\) are the maximal orders in the path in \(\mathcal{T}\) between \(\mathcal{O}\) and \(\mathcal{O}'\) in this setting (23.5.15 of \([27]\)). A maximal order \(\mathcal{O}\) corresponds to a homothety class of lattices \(L \subseteq \mathbb{Q}_q^2\). Two maximal orders \(\mathcal{O}\) and \(\mathcal{O}'\) are adjacent if there exist lattices \(L, L'\) for \(\mathcal{O}\) and \(\mathcal{O}'\) such that \(qL \subsetneq L' \subsetneq L\).
Thus the neighbors of $\mathcal{O}$ in $\mathcal{T}$ correspond to the one-dimensional subspaces of $L/qL$ and can thus be efficiently computed.

**Algorithm 4.1.** Enumerate all maximal orders containing a local order

Input: A $\mathbb{Z}_q$-order $\Lambda_q \subseteq M_2(\mathbb{Q}_q)$

Output: All maximal orders in $M_2(\mathbb{Q}_q)$ containing $\Lambda_q$

1. Compute a maximal order $\mathcal{O} \supseteq \Lambda_q$
2. Let $A = B = \{\mathcal{O}\}$
3. While $B \neq \emptyset$:
   a. Remove $\mathcal{O}$ from $B$
   b. Compute the neighbors $\mathcal{O}_0, \ldots, \mathcal{O}_q$ of $\mathcal{O}$ in $\mathcal{T}$
   c. for $i = 0$ to $q$:
      i. If $\mathcal{O}_i \supseteq \Lambda_q$, append $\mathcal{O}_i$ to $A$ and $B$
4. Return $A$.

**Proposition 4.2.** On input $\Lambda_q$, Algorithm 4.1 correctly computes $A := \{\mathcal{O} : \mathcal{O} \supseteq \Lambda_q\}$ and runs in time polynomial in $\log q \cdot \sum_{\mathcal{O} \in A} \text{size}(A)$. If $\Lambda_q$ is a Bass order, the runtime is polynomial in $\log q \cdot \log(\text{discr}(\Lambda_q))$.

**Proof.** For the definition of the size of a lattice see Section 2.2. Step 1 can be accomplished in time polynomial in $\log q \cdot \text{size}(\Lambda_q)$ with Algorithm 7.10 of [28].

We can accomplish Step 3(c)i with linear algebra over $\mathbb{Z}_q$. Now we show that the algorithm correctly computes every maximal order containing $\Lambda_q$ by showing that $A$ forms a connected subtree of $\mathcal{T}$. Suppose $\mathcal{O}, \mathcal{O}'$ are maximal orders containing $\Lambda_q$. Then $\mathcal{O} \cap \mathcal{O}'$ is an Eichler order containing $\Lambda_q$ by showing that $\mathcal{O}$ forms a connected subtree of $\mathcal{T}$. Suppose $\mathcal{O}, \mathcal{O}'$ are maximal orders containing $\Lambda_q$. Then $\mathcal{O} \cap \mathcal{O}'$ is an Eichler order containing $\Lambda_q$. The maximal orders in $M_2(\mathbb{Q}_q)$ containing $\mathcal{O} \cap \mathcal{O}'$ form a path in $\mathcal{T}$, and each such order also contains $\Lambda_q$. Thus $A$ forms a connected set, as claimed.

Finally, to see that the algorithm is efficient when $\Lambda_q$ is Bass, we use the results of [5]. First assume $\Lambda_q$ is Eichler. Then there are $e + 1$ maximal orders containing $\Lambda_q$, where $e = \log_q(\text{discr}(\Lambda_q))$ by Corollary 2.5 of [5]. If $\Lambda_q$ is Bass but not Eichler, then there are either 1 or 2 maximal orders containing $\Lambda_q$ by Proposition 3.1, Corollary 3.2, and Corollary 4.3 in [5].

**Algorithm 4.3.** Enumerate the $q$-maximal $\mathbb{Z}$-orders $\mathcal{O}$ containing $\Lambda$

Input: a $\mathbb{Z}$-order $\Lambda$ and prime $q$

Output: All $\mathbb{Z}$-orders $\mathcal{O} \supseteq \Lambda$ such that $\mathcal{O}$ is $q$-maximal and $\mathcal{O} \otimes \mathbb{Z}_q' = \Lambda \otimes \mathbb{Z}_q'$ for all primes $q \neq q'$

1. Let $e = v_q(\text{discr}(\Lambda))$
2. Compute an approximate embedding $f : \Lambda \hookrightarrow M_2(\mathbb{Z}_q)$ up to precision $e$
3. Let $A$ be the output of Algorithm 4.1 on input $f(\Lambda)$
4. Return $\{f^{-1}(\mathcal{O}) + \Lambda : \mathcal{O} \in A\}$

**Lemma 4.4.** Algorithm 4.3 is correct. If $\Lambda \otimes \mathbb{Z}_q$ is Bass, the run time is polynomial in $\log(\text{discr}(\Lambda \otimes \mathbb{Z}_q))$. 

Note that if $B$ is a division quaternion algebra over $\mathbb{Q}_q$, then the algorithm of [28] already solves Problem 1 in that case: there is only one maximal order in $B$. In later sections, we will be interested in enumerating the $q$-maximal orders containing a $\mathbb{Z}$-order $\Lambda$ in a quaternion algebra $B$ over $\mathbb{Q}$. The algorithm below uses Algorithm 4.1 to accomplish this.

**Algorithm 4.3.** Enumerate the $q$-maximal $\mathbb{Z}$-orders $\mathcal{O}$ containing $\Lambda$

Input: a $\mathbb{Z}$-order $\Lambda$ and prime $q$

Output: All $\mathbb{Z}$-orders $\mathcal{O} \supseteq \Lambda$ such that $\mathcal{O}$ is $q$-maximal and $\mathcal{O} \otimes \mathbb{Z}_q' = \Lambda \otimes \mathbb{Z}_q'$ for all primes $q \neq q'$

1. Let $e = v_q(\text{discr}(\Lambda))$
2. Compute an approximate embedding $f : \Lambda \hookrightarrow M_2(\mathbb{Z}_q)$ up to precision $e$
3. Let $A$ be the output of Algorithm 4.1 on input $f(\Lambda)$
4. Return $\{f^{-1}(\mathcal{O}) + \Lambda : \mathcal{O} \in A\}$

**Lemma 4.4.** Algorithm 4.3 is correct. If $\Lambda \otimes \mathbb{Z}_q$ is Bass, the run time is polynomial in $\log(\text{discr}(\Lambda \otimes \mathbb{Z}_q))$. 

Proof. First, in Step 2 we compute approximations of the coefficients in \( \mathbb{Z}_q \)-linear combinations of the basis elements of \( \Lambda \) which can be identified with a basis of an order in \( \mathbb{M}_2(\mathbb{Q}_q) \) isomorphic to \( \Lambda \). This can be efficiently executed using Algorithm 4.3 of [28]. We will compute these coefficients up to precision \( e = v_q(\text{discd}(\Lambda)) \). For each maximal \( \mathbb{Z}_q \)-order \( \mathcal{O} \supset \Lambda \), we then compute a corresponding \( \mathbb{Z} \)-order \( \mathcal{O}' \supset \Lambda \), whose generators are \( \mathbb{Z}[q^{-1}] \)-linear combinations of generators of \( \Lambda \), where the denominator of these coefficients is at most \( q^{-e} \). It is straightforward to check that the lattice \( \Lambda + \mathcal{O}' \) is actually a \( \mathbb{Z} \)-order and has the desired completions. \( \square \)

5. Enumerating maximal orders: the global case

In this section we enumerate all maximal orders \( \mathcal{O} \) of a quaternion algebra \( B \) over \( \mathbb{Q} \) that contain a given \( \mathbb{Z} \)-order \( \Lambda \). To do this, we use the results from the previous section; there we showed that for any prime \( q \) dividing \( \text{disc}(\Lambda) \) we can enumerate all \( \mathbb{Z} \)-orders \( \mathcal{O} \) such that \( \mathcal{O} \otimes \mathbb{Z}_q \) is a maximal order containing \( \Lambda \otimes \mathbb{Z}_q \) and such that \( \mathcal{O} \otimes \mathbb{Z}_{q'} = \Lambda \otimes \mathbb{Z}_{q'} \) for all \( q' \neq q \).

Algorithm 5.1. Enumerate all maximal orders containing an order
Input: An order \( \Lambda \) with factored reduced discriminant \( \prod_{i=1}^m q_i^{e_i} \).
Output: A complete list of maximal orders \( \mathcal{O} \) containing \( \Lambda \)

1. Let \( M = \{ \} \).
2. Run Algorithm 4.3 \( m \) times, namely on \((\Lambda, q_1), \ldots, (\Lambda, q_m)\).
   Let \( \{X_1, \ldots, X_m\} \) be the output, where \( X_i = \{O_{i1}, \ldots, O_{in_i}\} \).
3. For each \((O_{i1}, \ldots, O_{in_j}) \in \prod X_i:\)
   (a) Set \( \mathcal{O} = \Lambda \).
   (b) For each \( i = 1 \) to \( m \):
      (i) Update \( \mathcal{O} \) with \( \mathcal{O} + O_{ij} \).
   (c) Update \( M = M \cup \{O\} \).

Proposition 5.2. Algorithm 5.1 is correct and runs in time which is polynomial in the size of its output.

Proof. Step 3(b)i is done by computing a generating set for \( \mathcal{O} + O_{ij} \), using lattice reduction, and expressing the resulting \( \mathbb{Z} \)-basis as \( \mathbb{Q} \)-linear combinations of a basis of \( \Lambda \). We now show that the algorithm is correct. First, let \( O_i \) be an arbitrary order in \( X_i \), for \( i = 1 \) to \( m \), and define \( \mathcal{O}' = O_1 + \cdots + O_m \). Recall that Algorithm 4.3 ensures that \( O_i \) has the property that \( O_i \otimes \mathbb{Z}_q \) is a maximal order containing \( \Lambda \otimes \mathbb{Z}_q \) if \( q = q_i \), and \( O_i \otimes \mathbb{Z}_q = \Lambda \otimes \mathbb{Z}_q \) if \( q \neq q_i \). For each \( i = 1, \ldots, m \), we have
\[
\mathcal{O}' \otimes \mathbb{Z}_q = O_1 \otimes \mathbb{Z}_q + \cdots + O_n \otimes \mathbb{Z}_q = O_i \otimes \mathbb{Z}_q.
\]

The second equality follows because \( O_i \otimes \mathbb{Z}_q = \Lambda \otimes \mathbb{Z}_q \subset O_i \otimes \mathbb{Z}_q \) for \( i \neq j \). This implies \( \mathcal{O}' \otimes \mathbb{Z}_q \) is a maximal \( \mathbb{Z}_q \)-order, for each \( i \). If \( q \neq q_i \) for any \( i = 1, \ldots, m \), then \( (q, \text{discd}(\Lambda)) = 1 \), which implies that \( \mathcal{O}' \otimes \mathbb{Z}_q = \Lambda \otimes \mathbb{Z}_q \) is maximal. We note that \( \mathcal{O}' \) is a maximal order, because all of its localizations are maximal orders (Theorem 10.4.2 of [27]). Finally, we observe that every maximal order \( \mathcal{O} \supset \Lambda \) is enumerated, because in our construction we loop through every possible tuple of maximal orders in the completions (see Theorem 9.5.1 of [27]). \( \square \)

6. Recovering \( \text{End}(E) \) from a suborder

6.1. Computing \( \text{End}(E) \) by considering all possible local conditions. In this section, we describe how to recover \( \text{End}(E) \) from a Bass suborder \( \Lambda \), using the
algorithms from Sections 4 and 5. For this we use an isomorphism \( f : \Lambda \otimes \mathbb{Q} \rightarrow B_{p,\infty} \), which can be computed using the algorithms in \([15]\), which require factoring integers and polynomials in finite fields.

**Algorithm 6.1.** Determine \( \text{End}(E) \) via enumeration of maximal superorders

**Input:** A Bass order \( \Lambda \subseteq \text{End}(E) \) with factored reduced discriminant \( \prod q_i^{\epsilon_i} \), and \( f : \Lambda \otimes \mathbb{Q} \simeq B_{p,\infty} \), an isomorphism of quaternion algebras.

**Output:** A compact representation of \( \text{End}(E) \).

1. For each \( i = 1 \) to \( n \):
   (a) Compute all orders \( \{O_{i,1}, \ldots, O_{i,e_i+1}\} \) which are maximal at \( q_i \) and equal to \( \Lambda \) at primes \( q^i \neq q_i \) by running Algorithm 6.3 with input \( \Lambda \) and prime \( q_i \).
2. For each choice of indices \( (i_1, \ldots, i_n) \in [e_1 + 1] \times \cdots \times [e_n + 1] \):
   (a) Set \( \mathcal{O} := O_{i_1,i_1} + \cdots + O_{i_n,i_n} \).
   (b) Compute \( E'/\mathbb{F}_{p^2} \) such that \( \text{End}(E') \simeq f(\mathcal{O}) \).
   (c) If \( j(E') = j(E) \) or \( j(E') = j(E)^p \), return \( (f^{-1}, f(\mathcal{O})) \).

**Proposition 6.2.** For Bass orders \( \Lambda \) of size polynomial in \( \log p \) with factored, square-free reduced discriminant of size \( O(p^k) \), Algorithm 6.1 terminates in time \( O(p^\epsilon) \) for any \( \epsilon > 0 \), assuming that the heuristics in \([15, 11]\) hold.

**Proof.** For the definition of the size of an order see Section 2.2. Let \( \text{discr}(\Lambda) = p \cdot \prod_{i=1}^m q_i \) with \( q_1, \ldots, q_m \) distinct and different from \( p \). There is one maximal order corresponding to each collection of \( q_i \)-maximal orders \( \{O_i\} \) with \( O_i \supset \Lambda \otimes \mathbb{Z}_{q_i} \). Then there are at most \( 2^{\omega(\text{discr}(\Lambda)) - 1} \) many distinct maximal orders containing \( \Lambda \), where \( \omega(n) \) denotes the number of distinct prime factors of an integer \( n \). Since \( \omega(n) = O\left(\frac{\log n}{\log \log n}\right) \) \([17]\), Ch. 22, §10], there are \( O(p^\epsilon) \) many maximal \( \mathbb{Z} \)-orders \( \mathcal{O} \supset \Lambda \), at least one of which is isomorphic to \( \text{End}(E) \). We can test whether \( f(\mathcal{O}) \simeq \text{End}(E) \) using Algorithm 12 of \([11]\). Checking each order takes time polynomial in \( \log p \) (assuming the heuristics in \([15, 11]\)).

Our computational data shows that we get the same running time when the discriminant of \( \Lambda \) is not square-free. This leads us to the following

**Conjecture 6.3.** Fix an integer \( k \geq 0 \) and assume that \( \Lambda \subseteq \text{End}(E) \) is a Bass order of size polynomial in \( \log p \) and with \( \text{discr}(\Lambda) = O(p^k) \). Then for any \( \epsilon > 0 \), Algorithm 6.1 terminates in time \( O(p^\epsilon) \).

6.2. **Computing \( \text{End}(E) \).** Now we can describe our algorithm to compute the endomorphism ring of \( E \). By computing \( \text{End}(E) \) we mean computing a basis for an order \( \mathcal{O} \) in \( B_{p,\infty} \) that is isomorphic to \( \text{End}(E) \), and that we can evaluate the basis at all points of \( E \) via an isomorphism \( B_{p,\infty} \rightarrow \text{End}(E) \otimes \mathbb{Q} \).

**Algorithm 6.4.** Compute \( \text{End}(E) \)

**Input:** a supersingular elliptic curve \( E \)

**Output:** A compact representation of \( \text{End}(E) \).

1. Repeat
   (a) Compute two distinct cycles in \( G(p, \ell) \).
   (b) Let \( \alpha, \beta \) denote endomorphisms corresponding to the cycles from Step Ia. Compute \( \text{Trd}(\alpha), \text{Trd}(\beta), \text{Trd}(\alpha\beta) \) and the Gram matrix for \( \Lambda = \langle 1, \alpha, \beta, \alpha\beta \rangle \).
(c) Factor $\text{disc}(\Lambda) = \prod_{i=1}^n q_i^{e_i}$.

Until $\Lambda$ is Bass.

2. Using the Gram-Schmidt process, compute $a, b \in \mathbb{Z}$ so that $\Lambda \otimes \mathbb{Q} \simeq H(a, b)$.

3. Use Algorithm 6.1 to determine $\text{End}(E)$.

Based on our numerical experiments, in which more than 60% of our orders were Bass, we introduce a new heuristic assumption. This is needed to argue that the expected number of calls we must make to our cycle finding algorithm in order to compute a Bass order is bounded by a constant. When we generated random orders inside a maximal order we received a similar proportion of how many of those were Bass.

**Heuristic:** Our cycle finding algorithm produces cycles of length at most $L = \lceil 3 \log p \rceil$, and the corresponding endomorphisms’ discriminants have the distribution which is approximately the distribution of random discriminants in the range $[-4p^3, 0]$.

**Theorem 6.5.** Assume Conjecture 6.3 and the heuristics in [15] and the above heuristic. Let $E$ be a supersingular elliptic curve. Algorithm 6.4 computes $\text{End}(E)$ in time $O((\log p)^2 \sqrt{p})$.

**Proof.** The algorithm computes the endomorphism ring by putting together algorithms from previous sections. In Step 1b the Gram matrix for $\Lambda$ is computed using a generalization of Schoof’s algorithm (see Theorem A.6 of [3]), which can be done in time polynomial in $\log p$ and log of the norm of $\alpha, \beta$.

Next, we discuss the number of calls we must make to an algorithm for factoring integers. For each pair of cycles computed, the discriminant of the order they generate must be factored. Let $\Lambda \subseteq \text{End}(E)$ denote the Bass order generated by the cycles. Computing the isomorphism $f: \Lambda \otimes \mathbb{Q} \simeq B_{p, \infty}$ requires one call to an algorithm for factoring integers (and poly $\log p$ calls to algorithms for factoring polynomials over $\mathbb{F}_p$, see [18]). Hence the expected number of calls to an algorithm for factoring integers is bounded by a constant.

To check that $\Lambda$ is Bass, it is enough to check that $\Lambda$ is Bass at each $q$ dividing $\text{disc}(\Lambda)$ ([7, Theorem 1.2]). To check that $\Lambda$ is Bass at $q$ it is enough to check that $\Lambda \otimes \mathbb{Z}_q$ and $(\Lambda \otimes \mathbb{Z}_q)^\times$ are Gorenstein (see Corollary 1.3 of [7]). An order is Gorenstein if and only if its ternary quadratic form is primitive (see Corollary 24.2.10 of [27]), and this can be checked efficiently. Thus, given a factorization of $\text{disc}(\Lambda)$, we can efficiently decide if $\Lambda$ is Bass.

As an order is Bass if and only if it is basic, and being basic is a local property (see Theorem 1.2 in [7]), it follows that $\Lambda$ is Bass whenever the conductors of $\mathbb{Z}[\alpha]$ and $\mathbb{Z}[\beta]$ are coprime and the fundamental discriminants of $\mathbb{Z}[\alpha]$ and $\mathbb{Z}[\beta]$ are not equal. This will occur whenever the discriminants of $\alpha$ and $\beta$ are coprime which will happen with probability approximately $6/\pi^2 \approx .6$. See the data in Figure 6.2.

Since each cycle we compute has length $O(\log p)$, the corresponding endomorphism has norm $O(p)$. Using the bound that we obtained for $\text{disc}(\Lambda)$ in the proof of Theorem 3.9 shows that our orders $\Lambda \subseteq \text{End}(E)$ satisfy $\text{disc}(\Lambda) = O(p^k)$ for some $k$. If $\Lambda$ is Bass, then Step 3 requires polynomial storage, since we must store $O(\log \text{disc}(\Lambda))$ many $q$-maximal orders containing $\Lambda$, each of which has size which is polynomial in the size of $\Lambda$. By Conjecture 6.3 this step takes time $O(p^\epsilon)$ for any $\epsilon > 0$. □
We implemented a cycle finding algorithm in Sage along with an algorithm for computing traces of cycles in \( G(p, \ell) \). For each \( p \) in Figure 6.2 and for 100 iterations, we computed a pair of cycles in \( G(p, 2) \). We then tested whether they generate a Bass order by testing whether the two quadratic orders had coprime conductors and computed the discriminant of the order that they generate. We also computed an upper bound on the number of maximal orders containing \( \Lambda \) when \( \Lambda \) was Bass: suppose \( \text{disc}(\Lambda) = p \prod_i q_i^{e_i} \), then there are at most \( N(\Lambda) := \prod_i (e_i + 1) \) many maximal orders containing \( \Lambda \). We report how often the two cycles generated an order, how many of those orders were Bass, and the average value of \( N(\Lambda) \). We remark that the cycle-finding algorithm we implemented differs from the version described earlier – it searches for \( j \in \mathbb{F}_p^2 \setminus \mathbb{F}_p \) to build the cycles, and it does not take care to avoid finding a second cycle which may commute with the first. We also only computed cycles at \( j \in \mathbb{F}_p^2 \setminus \mathbb{F}_p \) because this is the case of interest: there are no obvious embeddings of imaginary quadratic orders when Frobenius is not an endomorphism.

Given an order \( O \supset \Lambda \) such that \( O \) is \( q \)-maximal but otherwise has localizations equal to \( \Lambda \), and given \( \alpha \in O \), we can test whether \( \alpha \in \text{End}(E) \) in time polynomial in \( \text{disc}(O \otimes \mathbb{Z}_q) \) and the size of \( \Lambda \). We expect this to speed up the computation of \( \text{End}(E) \) in practice.

### 7. Computing \( \text{End}(E) \) via pathfinding in the \( \ell \)-isogeny graph

In this section, we give a reduction from the endomorphism ring problem to the problem of computing \( \ell \)-power isogenies in \( G(p, \ell) \), using ideas from [22], [15], and [11]. This reduction is simpler than the one in [11], and uses only one call to a pathfinding oracle (rather than poly log \( p \) calls to an oracle for cycles in \( G(p, \ell) \), as in [11]). We apply this reduction in two ways, noting that it gives an algorithm for computing the endomorphism ring, and that it breaks second preimage resistance of the variable-length version of the hash function in [8].

#### 7.1. Reduction from computing \( \text{End}(E) \) to pathfinding in the \( \ell \)-isogeny graph.

We first define the pathfinding problem in the supersingular \( \ell \)-isogeny graph \( G(p, \ell) \):

**Problem 2 (\( \ell \)-PowerIsogeny).** Given a prime \( p \), along with two supersingular elliptic curves \( E \) and \( E' \) over \( \mathbb{F}_{p^2} \), output an isogeny from \( E \) to \( E' \) represented as a chain of \( \ell \)-isogenies of length \( k \) with \( k \) polynomial in \( \log p \).

Computing the endomorphism ring of a supersingular elliptic curve via an oracle for \( \ell \)-PowerIsogeny proceeds as follows. On input \( p \), Algorithm 3 of [11] returns a...
The algorithm is correct because at the 7.2. Using Algorithm 7.1 to compute endomorphism rings and break second preimage of the CGL hash. Algorithm 7.1 can be used to give an algorithm for computing the endomorphism ring of a supersingular elliptic curve $E$ by combining it with algorithms from [10, 15, 11]. This yields a $O((\log p)^2 p^{1/2})$ time algorithm with polynomial storage, assuming the relevant heuristics in [15, 11].

We now consider the hash function in [8] constructed from Pizer’s Ramanujan graphs $G(p, 2)$. For each supersingular elliptic curve $\tilde{E}$, there is an associated hash function. The input to the hash function is a binary number of $k$ digits, and from this one computes a sequence of $k$ 2-isogenies, starting at $\tilde{E}$, whose composition maps to some other supersingular curve $E$. The $j$-invariant of $E$ is the output of the hash function. The following is an improvement over [11], which gave a collision attack on the CGL hash for this specific hash function.

**Proposition 7.2.** Let $E_0$ be the elliptic curve computed in Step (1) of Algorithm 7.1. For the hash function associated to $E_0$, Algorithm 7.1 gives a second preimage attack (and hence, also a collision attack) on the CGL hash that runs in time polynomial in $\log p$. 

The following algorithm gives a polynomial time reduction from computing endomorphism rings to the path-finding problem, which uses only one call to the pathfinding oracle. It assumes the heuristics of [15] and GRH (to compute $\text{End}(E)$).

**Algorithm 7.1.** Reduction from computing $\text{End}(E)$ to $\ell$-PowerIsogeny

Input: prime $p$, $E/\mathbb{F}_{p^2}$ supersingular.

Output: A maximal order $\mathcal{O} \simeq \text{End}(E)$, whose elements can be evaluated at any point of $E$, and a powersmooth isogeny $\psi_E : E_0 \to E$, with $E_0$ as above.

1. Compute $E_0, \mathcal{O}$ with Algorithm 3 in [11].
2. Run the oracle for pathfinding on $E_0, E$ to obtain an $\ell$-power isogeny $\phi = \phi_{e \cdot \cdots \cdot \phi_1} : E_0 \to E$ of degree $\ell^e$.
3. Let $J_0 := \mathcal{O}_0, P_0 := \mathcal{O}_0$.
4. for $k := 1, \ldots, e$:
   a. Compute $I_k \subseteq \mathcal{O}_{k-1}$, the kernel ideal of $\phi_k$.
   b. Compute $J_k := J_{k-1} I_k$.
   c. Compute $P_k$, an ideal equivalent to $J_k$ of powersmooth norm.
   d. Compute an isogeny $\psi_k : E_0 \to E_k$ corresponding to $P_k$.
   e. Set $\mathcal{O}_k := \mathcal{O}_R(P_k)$.
5. Return $\mathcal{O}_R(P_e), \psi_E$.

Proof sketch for correctness of reduction and running time: The kernel ideal ideal $I_k$, which is the ideal of $\mathcal{O}_{k-1}$ of norm $\ell$ corresponding to $\phi_k$, can be computed in polynomial time. This uses the fact that we can evaluate endomorphisms efficiently using Proposition 3 of [11]. The ideal $J_k$ corresponds to $\psi_k : E_0 \to E_k$.

The algorithm is correct because at the $e$-th step we have $\mathcal{O}_R(P_e) = \mathcal{O}_R(J_e) = \text{End}(E_0) = \text{End}(E)$. 

supersingular elliptic curve $E_0$ defined over $\mathbb{F}_{p^2}$ and a maximal order $\mathcal{O}_0 \subseteq B_{p,\infty}$ with an explicit $\mathbb{Z}$-basis $\{x_1, \ldots, x_4\}$. Proposition 3 of [11] gives an explicit isomorphism $g : \mathcal{O}_0 \to \text{End}(E_0)$ with the property that we can efficiently evaluate $g(x_i)$ at points of $E_0$. From this, the endomorphism ring of any supersingular elliptic curve $E$ defined over $\mathbb{F}_{p^2}$ can be computed, given a path in $G(p, \ell)$ from $E_0$ to $E$, with $\ell \neq p$ a small prime.

The following algorithm gives a polynomial time reduction from computing endomorphism rings to the path-finding problem, which uses only one call to the pathfinding oracle. It assumes the heuristics of [15] and GRH (to compute $\text{End}(E)$).
Proof. The attack works as follows: Given a path from $E_0$ to $E$, use Algorithm 7.1 above to compute $\text{End}(E)$. Then use Algorithm 7 of [11] to compute new paths from $E_0$ to $E$. We note that Algorithm 7 uses the main algorithm of [22] to compute a connecting ideal of $\ell$-power norm, whose output can be randomized. Then for each such ideal, a corresponding path also hashes to $j(E)$. The running time of these algorithms is polynomial in $\log p$. □

Remark 7.3. When a start vertex $E' \neq E_0$ is chosen, the resulting hash function might still admit a second preimage attack if $E'$ was obtained by choosing a path of $\log p$ from $E_0$ to $E'$ so that the endomorphism ring of $E'$ is known.

8. Acknowledgements

We would like to thank John Voight for several helpful discussions and suggestions. We thank Ben Diamond for alerting us that an algorithm similar to Algorithm 7.1 in Section 7 already appeared in [13]. Finally we would like to thank an anonymous reviewer of a previous version of this paper whose suggestions greatly simplified Section 4.

References

[1] Sarah Arpin, Catalina Camacho-Navarro, Kristin Lauter, Joelle Lim, Kristina Nelson, Travis Scholl, and Jana Sotáková. Adventures in supersingular land. Preprint, 2019. arXiv:1909.07779.
[2] Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, David Jao, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, and David Urbanik. Supersingular isogeny key encapsulation. Submission to the NIST Post-Quantum Standardization project, 2017. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions.
[3] Efrat Bank, Catalina Camacho-Navarro, Kirsten Eisenträger, Travis Morrison, and Jennifer Park. Cycles in isogeny graphs and corresponding endomorphisms. Proceedings of the Women in Numbers 4 Conference. To appear in WIN 4 proceedings, 2019. arXiv:1804.04063.
[4] Juliana Belding, Reinier Bröker, Andreas Enge, and Kristin Lauter. Computing Hilbert class polynomials. In Algorithmic number theory, volume 5011 of Lecture Notes in Comput. Sci., pages 282–295. Springer, Berlin, 2008.
[5] Gaetan Bisson and Andrew V. Sutherland. Computing the endomorphism ring of an ordinary elliptic curve over a finite field. J. Number Theory, 131(5):815–831, 2011.
[6] Juliusz Brzeziński. On orders in quaternion algebras. Comm. Algebra, 11(5):501–522, 1983.
[7] Sara Chari, Daniel Smertnig, and John Voight. On basic and Bass quaternion orders. Preprint, 2019. arXiv:1903.00560.
[8] Denis X. Charles, Eyal Z. Goren, and Kristin Lauter. Cryptographic hash functions from expander graphs. J. Cryptology, 22(1):93–113, 2009.
[9] Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel Smith-Tone. Report on post-quantum cryptography. NIST IR 8105, February 2016.
[10] Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. Des. Codes Cryptography, 78(2):425–440, February 2016.
[11] Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: reductions and solutions. Eurocrypt 2018, LNCS 10822, pages 329–368, 2018.
[12] Noam D. Elkies. Supersingular primes for elliptic curves over real number fields. Compositio Math., 72(2):165–172, 1989.
[13] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. 2019. https://eprint.iacr.org/2019/168.
[14] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Advances in cryptography—ASIACRYPT 2016. Part I, volume 10031 of Lecture Notes in Comput. Sci., pages 63–91. Springer, Berlin, 2016.
[15] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In Advances in cryptology—ASIACRYPT 2017. Part I, volume 10624 of Lecture Notes in Comput. Sci., pages 3–33. Springer, 2017.

[16] Benedict Gross and Don Zagier. An introduction to the theory of numbers. Oxford University Press, Oxford, sixth edition, 2008.

[17] Godfrey H. Hardy and Edward M. Wright. An introduction to the theory of numbers. Oxford University Press, Oxford, sixth edition, 2008.

[18] Gábor Ivanyos, Lajos Rónyai, and Josef Schicho. Splitting full matrix algebras over algebraic number fields. J. Algebra, 354:211–223, 2012.

[19] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Post-quantum cryptography, volume 7071 of Lecture Notes in Comput. Sci., pages 19–34. Springer, Heidelberg, 2011.

[20] David Jao, Stephen D. Miller, and Ramarathnam Venkatesan. Expander graphs based on GRH with an application to elliptic curve cryptography. J. Number Theory, 129(6):1491–1504, 2009.

[21] David Kohel. Endomorphism rings of elliptic curves over finite fields. PhD thesis, University of California, Berkeley, 1996.

[22] David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion ℓ-isogeny path problem. LMS Journal of Computation and Mathematics, 17:418–432, 2014.

[23] Kristin Lauter and Bianca Viray. An arithmetic intersection formula for denominators of Igusa class polynomials. Amer. J. Math., 137(2):497–533, 2015.

[24] John E. Littlewood. On the Class-Number of the Corpus \( \sqrt{-k} \). Proc. London Math. Soc. (2), 27(5):358–372, 1928.

[25] Ken McMurdy. Explicit representations of the endomorphism rings of supersingular elliptic curves. 2014.

[26] Joseph H. Silverman. The arithmetic of elliptic curves. Springer, New York, 2009.

[27] John Voight. Quaternion Algebras. Version v.0.9.14, July 7, 2018.

[28] John Voight. Identifying the matrix ring: algorithms for quaternion algebras and quadratic forms. Developments in Mathematics, 31:255–298, 2013.

KIRSTEN EISENTRÄGER, DEPARTMENT OF MATHEMATICS, THE PENNSYLVANIA STATE UNIVERSITY, 109 MCALLISTER BUILDING, UNIVERSITY PARK, PA 16802, USA
E-mail address: eisentra@math.psu.edu

SEAN HALLGREN, DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, THE PENNSYLVANIA STATE UNIVERSITY, 350W WESTGATE BUILDING, UNIVERSITY PARK, PA 16802, USA
E-mail address: hallgren@cse.psu.edu

CHRISS LEONARDI, THE UNIVERSITY OF WATERLOO, 200 UNIVERSITY AVE W, WATERLOO, ON N2L 3G1, CANADA
E-mail address: cfoleona@uwaterloo.ca

TRAVIS MORRISON, INSTITUTE FOR QUANTUM COMPUTING, THE UNIVERSITY OF WATERLOO, 200 UNIVERSITY AVE W, WATERLOO, ON N2L 3G1, CANADA
E-mail address: travis.morrison@uwaterloo.ca

JENNIFER PARK, THE OHIO STATE UNIVERSITY, 231 W. 18TH AVE., MW 512 COLUMBUS, OH 43210, USA
E-mail address: park.270@osu.edu