Failing to hash into supersingular isogeny graphs (extended abstract)

Jeremy Booher[1], Ross Bowden[2], Javad Doliskani[3], Tako Boris Fouotsa[4], Steven D. Galbraith[5], Sabrina Kunzweiler[6], Simon-Philipp Merz[7], Christophe Petit[8,13], Benjamin Smith[9], Katherine E. Stange[10], Yan Bo Ti[11], Christelle Vincent[12], José Felipe Voloch[1], Charlotte Weitkämper[13], and Lukas Zobernig[5]

[1] University of Canterbury, New Zealand, jeremy.booher,felipe.voloch@canterbury.ac.nz; [2] University of Bristol, UK, ross.bowden@bristol.ac.uk; [3] Ryerson University, Canada, javad.doliskani@ryerson.ca; [4]EPFL, Switzerland, tako.fouotsa@epfl.ch; [5]The University of Auckland, New Zealand, s.galbraith,lukas.zobernig@auckland.ac.nz; [6]Ruhr-Universität Bochum, Germany, sabrina.kunzweiler@ruhr-uni-bochum.de; [7]Royal Holloway, University of London, UK, simon-philipp.merz.2018@rhul.ac.uk; [8]Laboratoire d'Informatique, Université libre de Bruxelles, Belgium, christophe.f.petit@gmail.com; [9]Inria and Laboratoire d'Informatique (LIX), CNRS, École polytechnique, France, smith@lix.polytechnique.fr; [10]University of Colorado Boulder, USA, kstange@math.colorado.edu; [11]DSO, Singapore, yanbo.ti@gmail.com; [12]University of Vermont, USA, christelle.vincent@uvm.edu; [13]University of Birmingham, UK, c.weitkaemper@pgr.bham.ac.uk

1 Introduction

Supersingular isogeny graphs – that is, supersingular curves over $\overline{\mathbb{F}}_p$, together with the isogenies between them – have become the basis for one of the principal candidates for post-quantum cryptography. The central hard problems are to compute the endomorphism ring of such a curve, or to compute a path in the graph between two given supersingular curves, which are related [12,14,23]. For many applications, it is desirable to be able to hash into the vertices of the graph, and it is important in these applications that the hashing process does not reveal the endomorphism ring, or a path from the curve to another known curve.

There are several methods to generate supersingular curves, but they each reveal information that may make the curve easier with respect to the hard problems mentioned above. The Charles-Goren-Lauter hash function [9] reveals a path from a starting curve, while the CM method [6] reveals information about the endomorphism ring, which can be exploited [4,8]. Currently the only known method to hash without revealing such information is to employ a trusted party to 'forget' the information generated by these methods. There are a number of papers that have already mentioned this problem [1,4,8].

Among other applications, using a starting curve that is generated uniformly at random in the SIDH key exchange [15] would avoid torsion point attacks [17,20,21]. Further, it would circumvent the trusted setup in an isogeny-based verifiable delay function [11], in delay encryption [7] and in an SIDH-based oblivious pseudorandom function [3]. For the latter, the necessity of the trusted setup was pointed out by [2].

There are (at least) three general problems that are of interest for isogeny-based cryptography:

1. Given a prime p, to compute a supersingular curve E over $\mathbb{F}_{p^2}$ without revealing anything about the endomorphism ring or providing any information to help solve the isogeny problem (for isogenies from E to some other supersingular curve over $\mathbb{F}_{p^2}$). This is the problem of **demonstrating a hard curve** [4].
2. Given a prime p, to generate **uniformly random** supersingular curves E over $\mathbb{F}_{p^2}$ without revealing anything about the endomorphism ring or providing any information to help solve the isogeny problem to other supersingular curves over $\mathbb{F}_{p^2}$.
3. **Defining a hash function to the entire supersingular graph**. To produce a hash function taking arbitrary strings as input, and giving supersingular j-invariants as output. The hard problems in this context include both pre-image finding and collision-finding for the hash function, and path finding and endomorphism ring computation for the output curve (which should remain hard).

It would also be of interest to define a hash function to the $\mathbb{F}_p$ subgraph. In all cases we are interested in, an efficient public algorithm that takes input p, can be executed without any secret information, and that outputs (the j-invariant of) a supersingular elliptic curve over $\mathbb{F}_{p^2}$. We do not want the algorithm to provide any additional information that would be useful to the person who executes it. For the problem of

generating a single hard curve (e.g., to bypass the requirement for trusted set up), the meaning of "efficient" might be relaxed, as long as it is feasible in applications. The goal of the paper is to explain some possible approaches and to discuss the obstructions to getting a practical solution. As the third problem above, a full hash function, would seem to require a solution to the first two, we focus on those first two problems in our approaches.

The underlying difficulty with the CM method is that the polynomial must have small degree if we hope to find its roots, and the small degree leads to small endomorphisms. The first three approaches suggested here attempt to find roots of high degree polynomials without computing the polynomials directly: by an iterative root finding method; by taking certain gcd's; and by considering a system of lower degree polynomials. Despite a wide range of approaches and polynomials (the supersingular polynomial, modular polynomials, and division polynomials), a full, efficient algorithm remains elusive.

The fourth approach considers walking on a related graph of genus 2 curves, to hide the path information in a path-based hash function. This approach fails in a number of interesting ways, raising questions about the placement of supersingular elliptic products in the isogeny graph of abelian surfaces.

The final approach asks what new possibilities a quantum computer may provide, but relies on quantum randomness, which cannot be turned into a reproducible hash function, and would require some type of 'quantumness certificate.'

The full paper for which this is the extended abstract can be found at [5]. Between submission and revisions for this work, the concurrent work [19], which also proposes some approaches to the hashing problem, was made public.

2 Existing insecure methods

The Charles-Goren-Lauter hash function [9] does, in fact, provide an (insecure) hash function into the supersingular curves over $\mathbb{F}_{p^2}$. At each vertex of the supersingular isogeny graph, the out-directed edges are labelled in some fixed deterministic manner. Starting from a known curve such as $j = 1728$, the bitstring to be hashed is interpreted as directions for a walk through the graph, via the labelling just mentioned. If the walk is sufficiently long, it is known from the properties of the graph (it is Ramanujan) that the endpoint will be uniformly randomly chosen from amongst all the vertices of the graph. However, the walk itself gives a path to $j = 1728$ and therefore the pathfinding problem from the endpoint is trivial, unless this information is discarded.

The CM method [6] finds supersingular roots to a Hilbert class polynomial. The Hilbert class polynomial $H_{\mathcal{O},p}$ for a quadratic order $\mathcal{O}$ modulo p is a polynomial in $\mathbb{F}_p[x]$ whose roots in $\overline{\mathbb{F}}_p$ are the j-invariants whose endomorphism rings contain a copy of $\mathcal{O}$. In order to apply known root-finding algorithms, or indeed, to obtain the polynomial at all, the degree of $H_{\mathcal{O},p}$ must be small. But the degree is approximately the square root of the discriminant of $\mathcal{O}$, so this implies that $\mathcal{O}$ itself has non-integral elements of small norm. The images of such elements in the endomorphism ring are termed *small endomorphisms*, and so all the curves obtained have small endomorphisms. Unfortunately, having a small endomorphism is known to be a vulnerability [4,8]. So the curves obtained are far from uniformly random curves, and in fact none of them is a secure curve.

Nevertheless, these two paradigms form the basis of the methods proposed in this work, which fall broadly into methods based on random walks, and methods based on finding roots to high degree polynomials (or systems of such).

3 Iterating to supersingular j-invariants

This is an approach to finding uniformly random supersingular curves by iterating to the fixed points of a dynamical system, if the starting point is restricted somehow. Optimistically, it could be an approach to construct a hash function, if the starting point can be chosen more or less arbitrarily. For a prime number $p > 2$, define the polynomial $H_p(t) := \sum_{j=0}^{(p-1)/2} \binom{\frac{p-1}{2}}{j}^2 t^j$. Letting E_λ be the elliptic curve whose Legendre

form is $y^2 = x(x-1)(x-\lambda)$, for $\lambda \in \mathbb{F}_p$ or $\lambda \in \mathbb{F}_{p^2}$, respectively, we have

$$\#E_\lambda(\mathbb{F}_p) \equiv p + 1 - H_p(\lambda) \pmod{p} \quad \text{respectively} \quad \#E_\lambda(\mathbb{F}_{p^2}) \equiv p^2 + 1 - H_p(\lambda)^{p+1} \pmod{p}, \tag{1}$$

so the roots of $H_p(t)$ correspond to supersingular elliptic curves. Taking inspiration from the Newton-Raphson method over the reals, a natural approach to finding a root of $H_p(t)$ is to iterate a carefully chosen polynomial function in search of a fixed point which would correspond to a root of $H_p(t)$ and hence a supersingular elliptic curve. This could also give a hash function if the starting point of the iteration were determined by the hash input. Explicitly, we fix an initial t_0 and iterate via

$$t_{n+1} = t_n - H_p(t_n) \text{ (over } \mathbb{F}_p) \quad \text{or} \quad t_{n+1} = t_n - H_p(t_n)^{p+1} \text{ (over } \mathbb{F}_{p^2}). \tag{2}$$

The key property is that fixed points correspond to zeroes of $H_p(t)$. (A denominator of $H_p'(t)$ would speed up convergence to a root in a field with a metric, but is irrelevant over a finite field.) There are three issues:

1. The algorithm may not halt at a fixed point (the iteration may become stuck in a cycle).
2. The algorithm may reach a fixed point, but require too many iterations to efficiently compute.
3. The polynomial $H_p(t)$ has degree $(p-1)/2$ and so it is difficult to directly carry out the iteration.

The third issue can be addressed using Schoof's point counting algorithm to efficiently compute $\#E_\lambda(\mathbb{F}_p)$ or $\#E_\lambda(\mathbb{F}_{p^2})$ and hence $H_p(\lambda)$ or $H_p(\lambda)^{p+1}$ using (1). The first and second issues are real obstructions. The key question is how many elements reach a fixed point within a certain number of iterations of (2).

We model the iteration as applying a random function over $\mathbb{F}_p$ or $\mathbb{F}_{p^2}$ that has many fixed points; $H_p(t)$ has $(p-1)/2$ roots over $\mathbb{F}_{p^2}$, and if $p = 4k+3$ then $p^{1/2+o(1)}$ of them are defined over $\mathbb{F}_p$. By adapting the asymptotic analysis introduced in [13], we prove:

Proposition 1. *For fixed m and k, the number of elements which reach a fixed point after k iterations for a random function on n elements with m fixed points is asymptotically $(k+1)m$ as $n \to \infty$.*

Experimentally, iterating (2) looks like iterating a random function with many fixed points. In our situation, n would be p or p^2 and m would be on the order of $\sqrt{n}$ (which is the number of supersingular Legendre curves over $\mathbb{F}_p$ when $n = p$ or $\mathbb{F}_{p^2}$ when $n = p^2$). To efficiently iterate and find a fixed point, the number of iterations would need to be polynomial in $\log(p)$, so heuristically and experimentally it is unlikely that iterating from a randomly chosen initial point will efficiently lead to a fixed point.

The analysis suggests that iterating k times is no better than randomly checking if k elements of $\mathbb{F}_{p^2}$ are roots of $H_p(t)$. To do better, one would need a way to perform a "giant step" and efficiently iterate many times at once. Experimentally, many additional elements iterate to fixed points but with too long a path.

4 Modular polynomials and curves isogenous to their conjugates

Next we consider using modular polynomials to demonstrate a hard curve. Whereas using the roots of Hilbert class polynomials produces curves with small endomorphism rings (a vulnerability), one might consider using other polynomials without this property. Recall that if n is a positive integer coprime to p, then the classical modular polynomial $\Phi_n(x,y) \in \mathbb{Z}[x,y]$ has the property that $\Phi_n(x,y) = 0$ in $\overline{\mathbb{F}}_p$ if and only if x and y in $\overline{\mathbb{F}}_p$ are j-invariants related by a cyclic n-isogeny (see [18, Chapter 5] for background). Taking inspiration from [10], consider the roots of the univariate polynomial $\Phi_n(x,x^p)$. These roots are the j-invariants of curves with cyclic n-isogenies to their conjugates, and hence with an inseparable np-endomorphism. There is no particular reason why these curves should also have small-degree non-integer endomorphisms.

While this polynomial is quite sparse, it has degree exponential with respect to $\log p$, and we cannot compute its roots efficiently. The idea is to reduce that degree, and make computations manageable, by instead computing roots in $\mathbb{F}_{p^2}$ of the factor(s)

$$f_{n,m,p}(x) := \gcd(\Phi_n(x,x^p), \Phi_m(x,x^p))$$

for some auxiliary m, without explicitly computing $\Phi_n(x, x^p)$ or $\Phi_m(x, x^p)$. The proposed approach for constructing supersingular curves is then: (i) Choose n and m; (ii) Compute one or more roots of $f_{n,m,p}(x)$ in $\mathbb{F}_{p^2}$; (iii) Test each root to see if it is a supersingular j-invariant, using known algorithms. Although any n and m will lead to supersingular curves, the choice must be made in such a way that (i) the resulting computation is feasible; and (ii) the resulting curves are secure.

Regarding feasibility, note that simply computing $\Phi_m(x, x^p)$ and $\Phi_n(x, x^p)$ in $\mathbb{F}_p[x]$, computing their gcd and finding its roots is exponential in $\log p$, because $\deg \Phi_m(x, x^p) > mp$ and $\deg \Phi_n(x, x^p) > np$. We provide an algorithm that computes all of the $\mathbb{F}_{p^2}$-roots of $f_{n,m,p}(x)$ in polynomial time with respect to m, n, and $\log p$. The key idea is to compute $\Phi_m(x, y)$ and $\Phi_n(x, y)$ and then avoid the degree of p obtained when substituting $y = x^p$ by using a Weil descent (i.e., restriction of scalars from $\mathbb{F}_{p^2}$ to $\mathbb{F}_p$) with a resultant in place of the gcd.

Regarding security, this method produces curves known to have endomorphisms of degree nm, np and mp. Since we wish to avoid endomorphisms of small degree, we should take at least one of n and m to be exponentially large. It is plausible that then the information about the endomorphism leaked from the process of construction is not enough to allow us to compute $\mathrm{End}(E)$ efficiently (i.e., in polynomial time). But if n (or m) is super-polynomially large with respect to $\log p$, then the algorithm described above requires super-polynomial time and space, since it must work explicitly with the polynomials Φ_n. Hence a natural open question is whether we can do better when one (or both) of n and m is large.

Even if a better algorithm is found, for this method to work, we need convincing evidence that supersingular roots of $f_{n,m,p}$ are common, and that the maximal factor of $f_{n,m,p}$ with roots in $\mathbb{F}_{p^2}$ has manageable degree. We give heuristics in favour of the relative frequency of supersingularity based on expansion properties of ordinary and supersingular graphs, and the dimension of the endomorphism rings, as well as some numerical experiments that confirm this. Some heuristic counting arguments predict that the $\mathbb{F}_{p^2}$-factor of $f_{n,m,p}$ should have degree $O(\sqrt{nm})$. This is no longer exponential with respect to $\log p$, but it is still too large when n or m is taken large (though one may hope for special families, or consider three-way gcds). In practice we observe degrees that are slightly smaller than those predicted above.

5 Reverse Schoof

In this section, we write down a polynomial system whose roots are supersingular j-invariants; solving such a system might demonstrate a hard curve. A supersingular curve is characterized by the number of points over any extension. We can therefore attempt to derive a system of equations whose solutions represent curves with a specified number of points. (Schoof's algorithm [22] takes in a curve and gives out its trace; the set of equations here can be thought of as a sort of Schoof's algorithm "backwards," taking in a desired trace and returning curves.) The set of equations will specify curves having a specified number of points only (no further restrictions), hence we do not expect this method to leak any further information about the curve. In other words, if successful, the obtained curves would, with overwhelming probability, be hard.

To this end, we fix a set of small primes or prime powers ℓ_i such that their product is above the Hasse bound. Then, we use the ℓ_i-th division polynomials parametrized by a curve parameter such as the j-invariant or the Montgomery coefficient to write down a polynomial system that forces all of its solutions to correspond to supersingular curves. Unlike other approaches using Hasse polynomials or modular polynomials, the polynomial systems of this approach can be easily written down explicitly. In particular, we write down the polynomial systems in the cases where $p + 1$ is the product of small distinct odd primes and where p is an SIDH-type prime.

We also propose several variants of this approach. For example, one could restrict some variables to a randomly chosen coset of a multiplicative subgroup to lower the degree of the equations at the cost of reducing the number of possible solutions. Finally, we discuss another approach which proceeds by sampling the x-coordinate of a point first and then tries to find a curve that contains a point with this x-coordinate of a given order. Unfortunately, computing the division polynomials arising in this case becomes too expensive. We leave to future work the study of the complexity of solving the polynomial systems in the general case, taking into account their full monomial structure as well as the impact on the complexity of all the variants.

6 Genus-2 walks

We explore approaches to generate supersingular elliptic curves by lifting an initial curve to genus 2 and then performing a random walk on an isogeny graph of genus two curves. This may lead to a method to generate random supersingular curves. We begin at a known supersingular elliptic curve $E_0/\mathbb{F}_q$, where $2 \nmid q$, glue it to itself along its 2-torsion to construct a genus-2 Jacobian $A \cong \mathrm{Jac}(C)$ explicitly isogenous to E_0^2, and then connect A with a new random elliptic product via an isogeny, hoping that these genus-2 operations will "hide" obvious isogenies between the elliptic curves involved.

In the procedure outlined above, the first step is eminently doable. The step to find a random elliptic product on the other hand is a little more complicated. We hope to be able to obtain randomisation by performing random walks on the superspecial graph. Again, this can be performed efficiently since Richelot isogenies are well-understood and are extremely efficient. Having completed the random walk, we will need to find an elliptic product. It is this final step of finding an elliptic product that we do not have an efficient algorithm for. We explore three methods to find these elliptic products. The first method is by finding elliptic products in the superspecial graph using Richelot isogenies. The other two methods are geometric inspections of the Jacobian via two representations of the Kummer surface.

Our first method proposes to set $q = p^2$, where $p > 2$, and to continue the random walk until an elliptic product is encountered in the graph. However, given that elliptic products have an occurrence of $1/p$ in the superspecial graph, this procedure will be no better than randomly sampling j-invariants to obtain a supersingular elliptic curve. Furthermore, there is a concern that the random walk in genus-2 can reveal information that can be used to compute the endomorphism ring of the final curve.

Our second method is based on the observation that every superspecial abelian surface is isomorphic to an elliptic product *as an unpolarised abelian variety*; so we propose to go looking for a new supersingular elliptic curve directly from the superspecial abelian surface. We work with the Kummer surface, which is the quotient of A by the action of the involution $[-1]$, because it is easier to manage (computationally) than the abelian surface. We consider two different models for the Kummer surface: the singular quartic model in $\mathbb{P}^3$, and a desingularised model in $\mathbb{P}^5$. When trying to find elliptic curves in the singular model, our first attempt is to find elliptic curves that do not pass through any of the nodes of the model. The second attempt is to find genus-0 curves on the singular model that pass through exactly 4 nodes. We also describe two approaches to find elliptic curves on the desingularised Kummer. In the first approach, we construct elliptic curves as quotients of non-hyperelliptic genus-5 curves on the desingularised Kummer. In the second approach, we construct pairs of elliptic curves that appear as the intersection of the Kummer surface with a hyperplane.

In all our attempts we were unsuccessful in finding supersingular elliptic curves. We attempted to probe the reason for these failures. In particular, we show that the elliptic curves that we constructed do not correspond to elliptic curves in the original PPSSAS.

7 Quantum algorithm for sampling a hard curve

Our final approach is an approach for generating random supersingular curves by means of a quantum random walk.

The Charles-Goren-Lauter hash function takes as input a bitstring and interprets it as directions to take a random walk through the ℓ-isogeny graph [9]. On a classical computer, the CGL hash function returns a random curve in the supersingular ℓ-isogeny graph, but the path information is patent, and can be used to compute the endomorphism ring of the curve which was produced. Therefore, using this as a hash to produce a "hard curve" requires a trusted party to throw away the path information.

We propose analogous quantum algorithms that avoid producing the path information at all (in any measurable way). Of course, then one needs to vouch for the algorithm being run properly on a quantum computer. In the hopes that the future may provide a "certificate of quantumness" that may apply to such a situation, we present a quantum algorithm inspired by work on quantum money by Kane, Sharif and Silverberg [16]. The algorithm depends upon the randomness of quantum measurement, and cannot be turned into a reproducible hash function: that is, it produces a random curve each time. However, random

supersingular curves are expected to be hard curves. Since the path information is quantumly inaccessible, the hope is that the method of random sampling will be secure.

Naïvely, one might attempt to use the CGL algorithm by performing the walk on a superposition of all possible hash inputs. However, this has the drawback that the path is stored in a quantum register and may be measured. Instead, we consider a continuous-time quantum random walk performed using the unitary operator $U_\ell = \exp(iT_\ell)$ where T_ℓ is the adjacency matrix of the ℓ-isogeny graph. Standard methods describe the probability distribution for the measurement of such a walk, at least in the abstract. However, it is not easy to analyse. Instead, we propose an efficient way to measure from the more accessible *limiting distribution*, whose description can be given in terms of graph theoretic properties of the ℓ-isogeny graph. The hope is that this will provide a provably uniform distribution on supersingular curves. The method depends upon the simultaneity of the eigenstates of Hecke operators, and the use of phase estimation.

Acknowledgements

This project was first undertaken as part of the Banff International Research Station (BIRS) Workshop 21w5229, *Supersingular Isogeny Graphs in Cryptography*. The project owes a debt of gratitude to BIRS and to the organizers of that workshop: Victoria de Quehen, Kristin Lauter, Chloe Martindale, and Christophe Petit. The project was led by Steven Galbraith, Christophe Petit, Yan Bo Ti, and Katherine E. Stange. We would also like to thank Chloe Martindale for useful discussions, Annamaria Iezzi for her involvement in Section 4, as well as Wouter Castryck and Eyal Goren for contributing ideas to Section 6.

References

1. Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 411–439. Springer, 2020.
2. Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 160–184. Springer, 2021.
3. Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 520–550. Springer, 2020.
4. Dan Boneh and Jonathan Love. Supersingular curves with small noninteger endomorphisms. In *ANTS XIV—Proceedings of the Fourteenth Algorithmic Number Theory Symposium*, volume 4 of *Open Book Series*, pages 7–22. Mathematical Sciences Publishers, 2020.
5. Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D. Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E. Stange, Yan Bo Ti, Christelle Vincent, José Felipe Voloch, Charlotte Weitkämper, and Lukas Zobernig. Failing to hash into supersingular isogeny graphs. https://eprint.iacr.org/2022/518, 2022.
6. Reinier Bröker. Constructing supersingular elliptic curves. *J. Comb. Number Theory*, 1(3):269–273, 2009.
7. Jeffrey Burdges and Luca De Feo. Delay encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 302–326. Springer, 2021.
8. Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. Rational isogenies from irrational endomorphisms. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020 Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 523–548. Springer, 2020.
9. Denis X. Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *J. Cryptology*, 22(1):93–113, 2009.
10. Mathilde Chenu and Benjamin Smith. Higher-degree supersingular group actions. *Mathematical Cryptology*, 1(2):85–101, 2022.
11. Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 248–277. Springer, 2019.

12. Kirsten Eisenträger, Sean Hallgren, Kristin E. Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 329–368. Springer, 2018.
13. Philippe Flajolet and Andrew M. Odlyzko. Random mapping statistics. In *Advances in cryptology—EUROCRYPT '89 (Houthalen, 1989)*, volume 434 of *Lecture Notes in Comput. Sci.*, pages 329–354. Springer, Berlin, 1990.
14. Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 63–91, 2016.
15. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *International Workshop on Post-Quantum Cryptography*, pages 19–34. Springer, 2011.
16. Daniel M. Kane, Shahed Sharif, and Alice Silverberg. Quantum money from quaternion algebras. arXiv:2109.12643, 2021.
17. Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 242–271. Springer, 2021.
18. Serge Lang. *Elliptic functions*, volume 112 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, second edition, 1987. With an appendix by J. Tate.
19. Marzio Mula, Nadir Murru, and Federico Pintore. On random sampling of supersingular elliptic curves. Cryptology ePrint Archive, Paper 2022/528, 2022. https://eprint.iacr.org/2022/528.
20. Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, pages 330–353, 2017.
21. Victoria de Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E. Stange. Improved torsion-point attacks on SIDH variants. In *Annual International Cryptology Conference*, pages 432–470. Springer, 2021.
22. René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Mathematics of Computation*, 44(170):483–494, 1985.
23. Benjamin Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1100–1111. IEEE, 2022.