Isogenies in genus 2 for cryptographic applications

Benjamin Smith
with Wouter Castryck, Craig Costello, Thomas Decru, Enric Florit

October 4, 2022

Inria + Laboratoire d'Informatique de l'École polytechnique (LIX)

Isogeny-based cryptosystems

Group-action cryptosystems: based on the action of $\mathsf{Cl}(\mathcal{O})$ on an isogeny (sub)class $\mathsf{Ell}_{\mathbb{F}_q}(\mathcal{O})$ of curves $\mathcal{E}/\mathbb{F}_q$ with $\mathsf{End}(\mathcal{E}) \cong \mathcal{O}$ *(Steven's talk)*.

- **Couveignes**, **Rostovtsev–Stolbunov** (2006): key exchange on **ordinary** isogeny graphs (CRS)
- **CSIDH**: like CRS but with $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$ (supersingular over $\mathbb{F}_p$)
- **CSI-FiSh** signatures, ...

Everything else: crypto based on supersingular isogeny graphs over $\mathbb{F}_{p^2}$.

- **Charles–Goren–Lauter** hash function *(Kristin's talk)*.
- **SIDH** and **SIKE**, now broken! *(Wouter's talk, Chloe's talk)*
- **SQISign** signatures *(Luca's talk)*, ...

Why genus 2?

> *"It is a truth universally acknowledged that a number theorist in possession of a good elliptic curve cryptosystem, must be in want of a generalization to genus 2."*

In this context, "genus 2" means **principally polarized abelian surfaces (PPASes)**.

More generally, principally polarized abelian varieties (PPAVs) in dimension $g > 1$ (but let's start with $g = 2$...)

This is **not completely crazy**: genus-2 analogues of elliptic discrete-log-based cryptosystems give nice practical results (see e.g. Renes–S. (Asiacrypt 2017)).

However: DLP-based crypto turns out to be *less efficient* in $g > 2$ thanks to index calculus and isogeny attacks (see e.g. S. (Eurocrypt 2008)).

Today's focus: CGL in genus 2

This talk: we will **ignore genus-2 group-action crypto**

- Very little has been done!
- The endomorphism rings and their class groups are more complicated (especially because of real subrings $\neq \mathbb{Z}$, real units, …)
- The group action is much more complicated to compute

We will focus exclusively on the simplest example of the **everything else** class: the **Charles–Goren–Lauter (CGL) hash function**.

The supersingular elliptic 2-isogeny graph

Supersingular elliptic isogeny graphs

Fix a prime p, and let

$$S_1(p) := \{\text{supersingular elliptic curves } /\overline{\mathbb{F}}_p\}/\cong .$$

For primes $\ell \neq p$, we let $\Gamma_1(\ell; p)$ be the **ℓ-isogeny graph** on $S_1(p)$.

The graph $\Gamma_1(\ell; p)$ is

- **connected**,
- $(\ell + 1)$**-regular**, and
- **Ramanujan** (excellent expansion properties)

Random walks in $\Gamma_1(\ell; p)$ of length $O(\log p)$ give a uniform distribution on $S_1(p)$.

The supersingular graph is a directd weighted graph

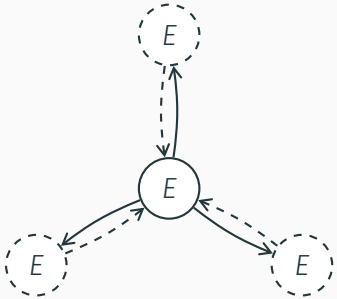
In general, $\Gamma_1(\ell; p)$ is a **directed weighted graph**.

- An edge $[\mathcal{E}_1] \to [\mathcal{E}_2]$ has **weight n**
  if there are **n distinct kernels** in $\mathcal{E}_1[\ell]$ with codomain $\cong \mathcal{E}_2$.
- These distinct kernels form an orbit under the **reduced automorphism group**

$$\mathrm{RA}(\mathcal{E}) := \mathrm{Aut}(\mathcal{E})/\langle \pm 1 \rangle \,,$$

  so we only have multiple edges at $j = 0$ and 1728 (if they are in $S_1(p)$).
- **Dual isogenies**: $\exists\, [\mathcal{E}_1] \to [\mathcal{E}_2] \implies \exists\, [\mathcal{E}_2] \to [\mathcal{E}_1]$.
- Multiple edges in one direction share a **single dual** in the other.

Neighbourhoods of vertices in $\Gamma_1(2; p)$



General j

$j = 0$

$j = 1728$

Supersingular isogeny problems

The general supersingular elliptic **isogeny problem** for fixed ℓ:

Given $\mathcal{E}$ and $\mathcal{E}'$ in $S_1(p)$, find a path from $\mathcal{E}$ to $\mathcal{E}'$ in $\Gamma_1(\ell; p)$

classical solution in $O(\sqrt{\#S_1(p)}) = O(\sqrt{p})$ *(random walks)*

quantum solution in $O(\sqrt[4]{\#S_1(p)}) = O(\sqrt[4]{p})$

This problem is related to the security of the Charles–Goren–Lauter hash function.

The Charles–Goren–Lauter function

Cryptographic hash functions

A **cryptgraphic hash function** $H : \{0,1\}^* \to \{0,1\}^n$ should have

- **preimage resistance**: given $t \in \{0,1\}^n$, hard to find m s.t. $H(m) = t$
  ideal: $\approx 2^n$ *operations*
- **collision resistance**: hard to find $m \neq m'$ s.t. $H(m) = H(m')$
  ideal: $\approx 2^{n/2}$ *operations*
- **2nd preimage resistance**: given m, hard to find $m' \neq m$ s.t. $H(m') = H(m)$
  ideal: $\approx 2^n$ *operations*

In addition to these security properties, we typically want

- **efficiency**: should be able to hash long inputs very quickly
- **pseudo-randomness**: H should act like a random oracle, i.e. indistinguishable from a random function into $\{0,1\}^n$ on distinct inputs

The Charles–Goren–Lauter hash function (2009)

Charles–Goren–Lauter: a hash function with provable preimage-resistance.

Parameters:

- a large prime p,
- an ordering on $\mathbb{F}_{p^2}$ (hence on $S_1(p)$),
- an edge $j_{-1} \to j_0$ in $\Gamma_1(2; p)$,
- a linear map $\pi : \mathbb{F}_{p^2} \to \mathbb{F}_p$ *(often ignored)*.

The hash function $H : \{0,1\}^* \to \mathbb{F}_p$ is

$$H(m) := \pi(\mathsf{CGL}(m)),$$

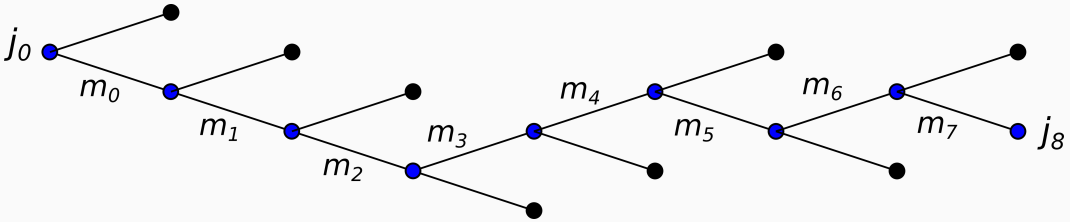
where $\mathsf{CGL} : \{0,1\}^* \to S_1(p) \subset \mathbb{F}_{p^2}$ is defined as follows...

The CGL function: data drives walks

To compute the image under CGL of an n-bit string $m = (m_0, \ldots, m_{n-1})$, we compute a non-backtracking walk $j_0 \to \cdots \to j_n$ in $\Gamma_1(2; p)$: for each $0 \le i < n$,

1. the 3 edges out of j_i are $j_i \to j_{i-1}$ and $j_i \to \alpha$ and $j_i \to \beta$, with $\alpha > \beta$
2. if $m_i = 0$, then set $j_{i+1} = \alpha$; otherwise, set $j_{i+1} = \beta$.

The **output** is $\mathrm{CGL}(m) = j_n$.



CGL hashing: security

Finding **preimages** for the CGL function

- $\implies$ solving the **isogeny problem** in $\Gamma_1(2; p)$
- Is **hard**. Best algorithm: $O(p^{1/2})$ (classical), $O(p^{1/4})$ (quantum)

Finding **collisions** for the CGL function

- $\implies$ computing **cycles** in $\Gamma_1(2; p)$ through j_0
- $\implies$ computing 2^*-endomorphisms of $\mathcal{E}_0$
- Supposed to be hard (in 2006)... But Kohel–Lauter–Petit–Tignol (KLPT) solves this in **polynomial time** if $\mathsf{End}(\mathcal{E}_0)$ is known (i.e., for reasonable choices of j_0)!

Open problem: efficiently constructing supersingular curves with *unknown* endomorphism ring.

Computing the CGL function

Traditional approach:

Given a bit m_i and an edge $j_{i-1} \to j_i$, we need to compute j_{i+1}.

1. Compute $f(X) = \Phi_2(j_1, X)/(X - j_{i-1}) \in \mathbb{F}_{p^2}[X]$
2. Find the two roots $\alpha > \beta$ of $f(X)$ in $\mathbb{F}_{p^2}$
3. if $m_i = 0$, set $j_{i+1} = \alpha$; otherwise, set $j_{i+1} = \beta$.

Computing the CGL function

Alternative approach: work (up to isomorphism) with curves

$$\mathcal{E}_i : y^2 = x(x^2 + a_2^{(i)}x + a_4^{(i)})\,.$$

To find $\mathcal{E}_i \to \mathcal{E}_{i+1}$: compute $\delta := \sqrt{(a_2^{(i)})^2 - 4a_4^{(i)}}$ in $\mathbb{F}_{p^2}$, using m_i to choose $\mathsf{sign}(\delta)$; then $\langle(-(a_2^{(i)} - \delta)/2, 0)\rangle \subset \mathcal{E}_i[2]$ is the kernel of the edge $\mathcal{E}_i \to \mathcal{E}_{i+1}$, with

$$a_2^{(i+1)} = a_2^{(i)} - 3\delta \qquad \text{and} \qquad a_4^{(i+1)} = a_2^{(i)}(a_2^{(i)} + \delta)/2 - a_4^{(i)}\,,$$

and the kernel of the dual edge $\mathcal{E}_{i+1} \to \mathcal{E}_i$ is $\langle(0, 0)\rangle \subset \mathcal{E}_{i+1}[2]$.

Either way: CGL requires **one square root in $\mathbb{F}_{p^2}$ per bit of input**. (Sloooow)

Limited speedup for suitable p: see Doliskani–Pereira–Barreto (2022)

A word on finalization

The CGL hash value is $H(m) = \pi(j_n)$, where π is a linear map $\mathbb{F}_{p^2} \to \mathbb{F}_p$.

Why do we use the finalization map π?

- because there are only $\approx p/12$ elements of $S_1(p)$,
- so the uniform distribution on $S_1(p)$ only has $\approx \log_2 p$ bits of entropy,
- so we should squash hash values down to $\log_2 p$ bits.
- A sufficiently general linear function $\mathbb{F}_{p^2} \to \mathbb{F}_p$ will do the job.

Even if you can solve the isogeny problem, to invert the true CGL hash function we must find preimages under $\pi|_{S_1(p)}$, and this already seems hard!

Open problem: Given a linear map $\pi : \mathbb{F}_{p^2} \to \mathbb{F}_p$ and a random $\alpha \in \mathbb{F}_p$,
find (if it exists) a supersingular $j \in \mathbb{F}_{p^2}$ such that $\pi(j) = \alpha$.

$g > 1$

Higher dimensions: superspecial and supersingular

A g-dimensional PPAV $\mathcal{A}$ is

Super*singular* if all slopes of the Newton polygon of its Frobenius are 1/2.
Any supersingular $\mathcal{A}$ is **isogenous** to a product of supersingular ECs.

Super*special* if Frobenius acts as 0 on $H^1(\mathcal{A}, \mathcal{O}_{\mathcal{A}})$.
Any superspecial $\mathcal{A}$ is **isomorphic** to a product of supersingular ECs, though generally only as unpolarized AVs.

- **Superspecial** $\implies$ **supersingular**.
- Superspeciality is preserved by $(\ell, \dots, \ell)$-isogeny.

Superspecial PPAVs are connected to non-superspecial supersingular PPAVs by p-isogenies, but these are much more complicated than in genus 1: see Brock–Howe for a guided tour with $(g, p) = (2, 2)$.

The superspecial set

For each $g > 0$ and prime p, we define

$$S_g(p) := \left\{\text{superspecial PPAVs over } \overline{\mathbb{F}}_p\right\} / \cong .$$

We have

$$\#S_g(p) = O(p^{g(g+1)/2})$$

(with much more precise statements for $g \leq 3$).

For primes $\ell \neq p$, we let $\Gamma_g(\ell; p)$ be the $(\ell, \ldots, \ell)$-isogeny graph on $S_g(p)$.

Recall: $(\ell, \ldots, \ell)$-isogeny kernels are maximal ℓ-Weil-isotropic subgroups of the ℓ-torsion; these isogenies respect the principal polarizations.
Such kernels are necessary isomorphic to $(\mathbb{Z}/\ell\mathbb{Z})^g$.

The superspecial graph

The graph $\Gamma_g(\ell; p)$ is

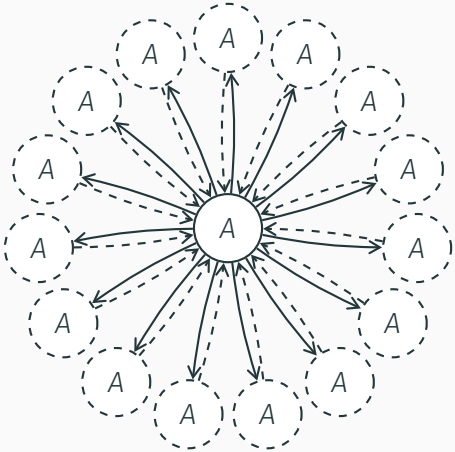
1. **connected** (implicit in Oort; explicit in Jordan–Zaytman 2020)
2. $N_g(\ell)$-**regular**, where

$$N_g(\ell) := \sum_{d=0}^{g} \begin{bmatrix} g \\ d \end{bmatrix}_\ell \cdot \ell^{\binom{g-d+1}{2}}$$

where $\begin{bmatrix} n \\ k \end{bmatrix}_\ell :=$ the number of k-dimensional subspaces of $\mathbb{F}_\ell^n$.

- $N_1(\ell) = \ell + 1$
- $N_2(\ell) = \ell^3 + (\ell + 1)\ell + 1$
- $N_g(\ell)$ is a polynomial in ℓ of degree $g(g+1)/2$

The neighbourhood of a general vertex in $\Gamma_2(2; p)$



Expansion hypothesis

When generalizing cryptosystems like the CGL hash to $g > 1$, we have an obvious

Question: Is $\Gamma_g(\ell; p)$ Ramanujan?

Jordan–Zaytman (2020): in general, **no**.

*This is no problem. For cryptographic applications, we just need $\Gamma_g(\ell; p)$ to have "**good expansion properties**": that is, random walks of length $O(\log p)$ in $\Gamma_g(\ell; p)$ should converge to the uniform distribution on $S_g(p)$.*

Florit–S. 2021: empirical support and approximate constants for $(g, \ell) = (2, 2)$.

Generalizing CGL to genus 2: Takashima

Takashima was the first to generalize CGL to PPAVs of dimension $g = 2$.

- $S_1(p)$ becomes $S_2(p)$
- $\Gamma_1(2; p)$ becomes $\Gamma_2(2; p)$: i.e. 2-isogenies become $(2, 2)$-isogenies,

To compute non-backtracking walks in $\Gamma_2(2; p)$, Takashima uses

- supersingular **genus-2 curves** to represent vertices,
- **Richelot's formulæ** to compute the isogeny steps, and
- Igusa–Clebsch invariants to replace the j-invariant.

Since $\Gamma_1(2; p)$ is 15-regular, the data to be hashed is coded in base 14 (!).

Richelot isogenies

Consider a genus-2 curve

$$\mathcal{C} : y^2 = G_1(x)G_2(x)G_3(x)$$

where the G_i are pairwise coprime quadratics in x (one may be linear).

Each of the G_i specifies a point of order 2 in $\mathrm{Jac}(\mathcal{C})$, and the subgroup whose nonzero elements correspond to $\{G_1, G_2, G_3\}$ is the kernel of a $(2,2)$-isogeny.

The codomain of the isogeny is the Jacobian of the genus-2 curve

$$\mathcal{C}' : \Delta y^2 = H_1(x)H_2(x)H_3(x)\,,$$

where $H_i = G_j'G_k - G_jG_k'$ and $\Delta = \det(G_1, G_2, G_3)$.

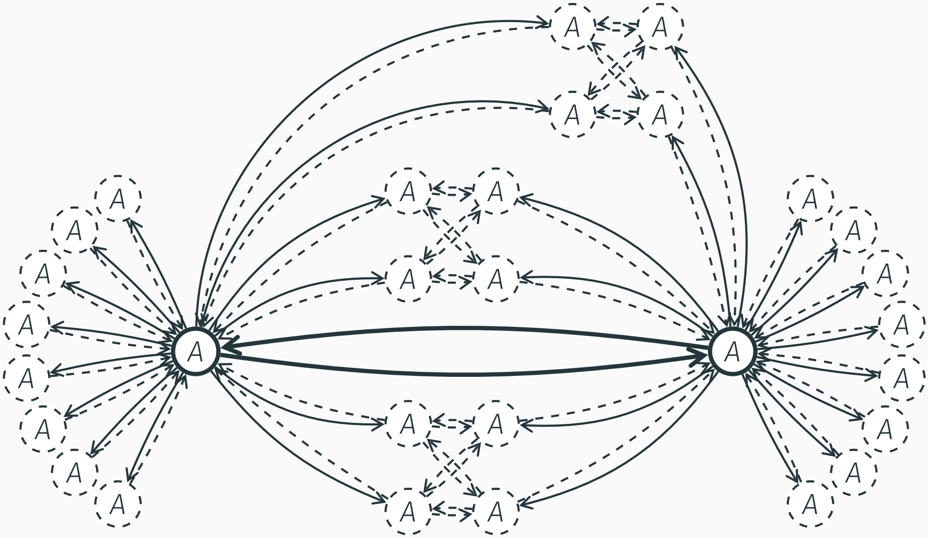
Trivial 4-cycles in the genus-2 graph

Flynn and Ti observe a serious issue with Takashima's hash function:
It is easy to construct **cycles of length 4** starting at any vertex of $\Gamma_2(\ell; p)$.

If we consider the neighbourhood of a general edge in $\Gamma_2(2; p)$, then for every $(2,2)$-isogeny $\mathcal{A}_1 \to \mathcal{A}_2$, there are always **twelve** ways of composing three more different $(2,2)$-isogenies to get a length-4 cycle (splitting multiplication by 4 on $\mathcal{A}_1$).

Non-backtracking is not a strong enough condition to avoid hash collisions.

The neighbourhood of a general edge (and its dual) in $\Gamma_2(2; p)$



The good, the bad, and the dual

Suppose $\phi : \mathcal{A} \to \mathcal{A}'$ and $\phi' : \mathcal{A}' \to \mathcal{A}''$ are (ℓ, ℓ)-isogenies.

Definition: We say that ϕ' is an **extension** of ϕ, and that the extension is

good if $\phi' \circ \phi$ is an (ℓ^2, ℓ^2)-isogeny;

bad if $\phi' \circ \phi$ is an (ℓ^2, ℓ, ℓ)-isogeny;

dual if $\phi' \circ \phi$ is a (ℓ, ℓ, ℓ, ℓ)-isogeny (i.e. $\cong [\ell]_{\mathcal{A}}$).

Of the $\ell^3 + (\ell + 1)\ell + 1$ extensions of ϕ,

- ℓ^3 are good;
- $\ell^2 + \ell$ are bad;
- 1 is dual.

Generalizing CGL to genus 2

Castryck–Decru–S. (Nutmic 2019): an attempt to repair Takashima's hash.

We use a **new rule for isogeny walks** to replace non-backtracking:

After each $(2,2)$-isogeny $\phi_i : \mathcal{A}_i \to \mathcal{A}_{i+1}$ in the walk,
we must take $\phi_{i+1} : \mathcal{A}_{i+1} \to \mathcal{A}_{i+2}$ to be one of the **eight good extensions** of ϕ_i.

The hash walks are thus (ℓ^n, ℓ^n)*-isogenies; no short cycles are possible.*

Implementation: following Takashima, we represent vertices with (Jacobians of) genus-2 curves, and compute edges using Richelot isogenies.

Good extensions of Richelot isogenies

To realise a $(2,2)$-isogeny $\phi_i : \mathsf{Jac}(\mathcal{C}_i) \to \mathsf{Jac}(\mathcal{C}_{i+1})$, Richelot's formulæ map

$$\mathcal{C}_i : y^2 = G_1^{(i)}(x)G_2^{(i)}(x)G_3^{(i)}(x)$$

to

$$\mathcal{C}_{i+1} : \Delta^{(i)}y^2 = H_1^{(i)}(x)H_2^{(i)}(x)H_3^{(i)}(x)\,.$$

where $H_1^{(i)} = (G_2^{(i)})'G_3^{(i)} - G_2^{(i)}(G_3^{(i)})'$, etc.

The next isogeny corresponds to a quadratic splitting $\{G_1^{(i+1)}, G_2^{(i+1)}, G_3^{(i+1)}\}$:

1x dual extension: $\{G_1^{(i+1)}, G_2^{(i+1)}, G_3^{(i+1)}\} = \{H_1^{(i)}, H_1^{(i)}, H_1^{(i)}\}$
6x bad extensions: $\#\big(\{G_1^{(i+1)}, G_2^{(i+1)}, G_3^{(i+1)}\} \cap \{H_1^{(i)}, H_1^{(i)}, H_1^{(i)}\}\big) = 1$
8x good extensions: $\#\big(\{G_1^{(i+1)}, G_2^{(i+1)}, G_3^{(i+1)}\} \cap \{H_1^{(i)}, H_1^{(i)}, H_1^{(i)}\}\big) = 0$

Computing the good extensions requires **three square roots in $\mathbb{F}_{p^2}$** to split the $H_j^{(i)}$.

An algorithmic inconvenience

Minor inconvenience: there are *two types* of PPAVs in dimension $g = 2$: **Jacobians** of genus-2 curves, and **elliptic products**.

- Isomorphism invariants are incompatible
- Richelot's formulæ break down ($\Delta = 0$) when the codomain is a product

Partition $S_2(p)$ into corresponding subsets, $S_2(p)^J$ and $S_2(p)^E$; then

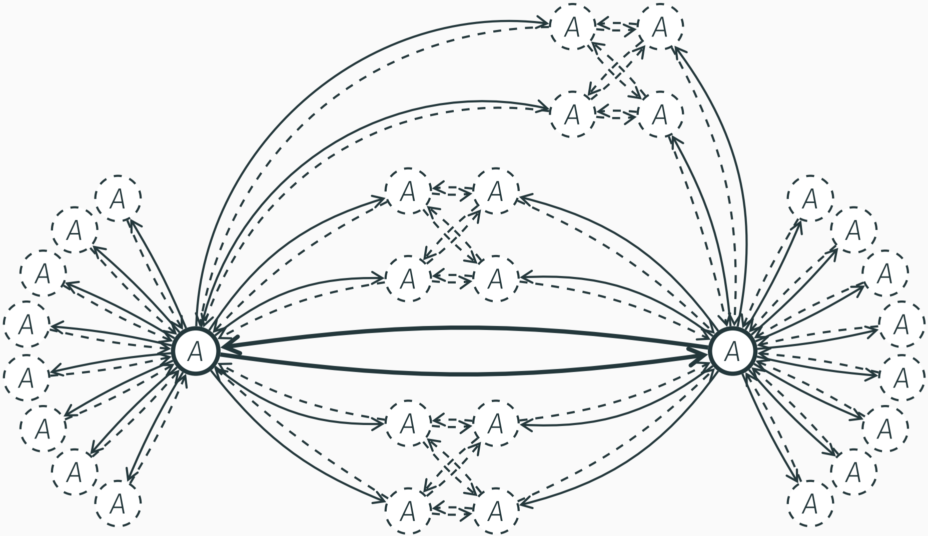
$$\#S_2(p)^J = \frac{1}{2880}p^3 + \frac{1}{120}p^2 \qquad \text{and} \qquad \#S_2(p)^E = \frac{1}{288}p^2 + O(p)\,.$$

Being a proof of concept, CDS takes a simple solution: *fail on elliptic products*. Justification: a random $\mathcal{A} \in S_2(p)$ has only a $O(1/p)$ chance of being in $S_2(p)^E$.

Bad news: from a cryptanalytic point of view, **this is not rare enough**.

A closer look at $\Gamma_2(2; p)$

Naive view of the graph: tessellate the generic edge neighbourhood



Special vertices

Katsura–Takashima (ANTS 2020) studies the interaction of Richelot isogenies and reduced automorphism groups.

Florit–S. (2020) goes further and builds an "atlas" of $\Gamma_2(2;p)$.

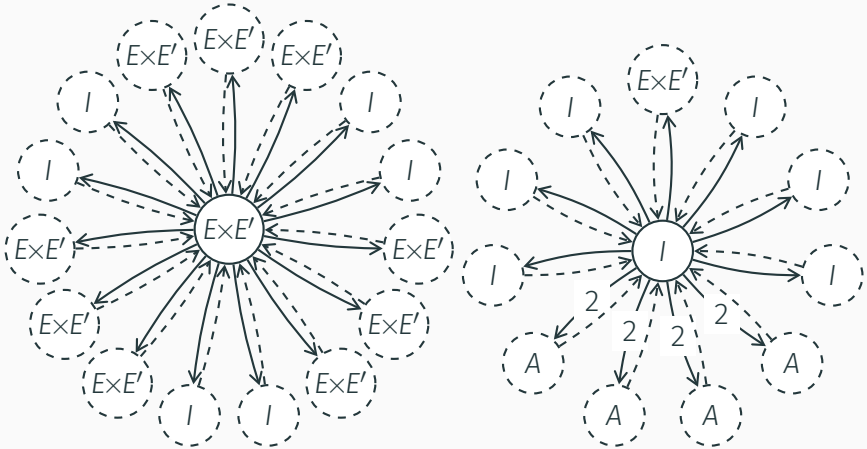
The typical 15-regular structure degenerates near vertices of $\Gamma_2(2;p)$ with nontrivial reduced automorphism group, as edges pick up nontrivial weights.

Much intuition comes from the **ratio principle**:

$$\mathsf{weight}([\phi : \mathcal{A} \to \mathcal{B}]) \cdot \#\,\mathsf{RA}(\mathcal{B}) = \mathsf{weight}([\widehat{\phi} : \mathcal{B} \to \mathcal{A}]) \cdot \#\,\mathsf{RA}(\mathcal{A})\,,$$

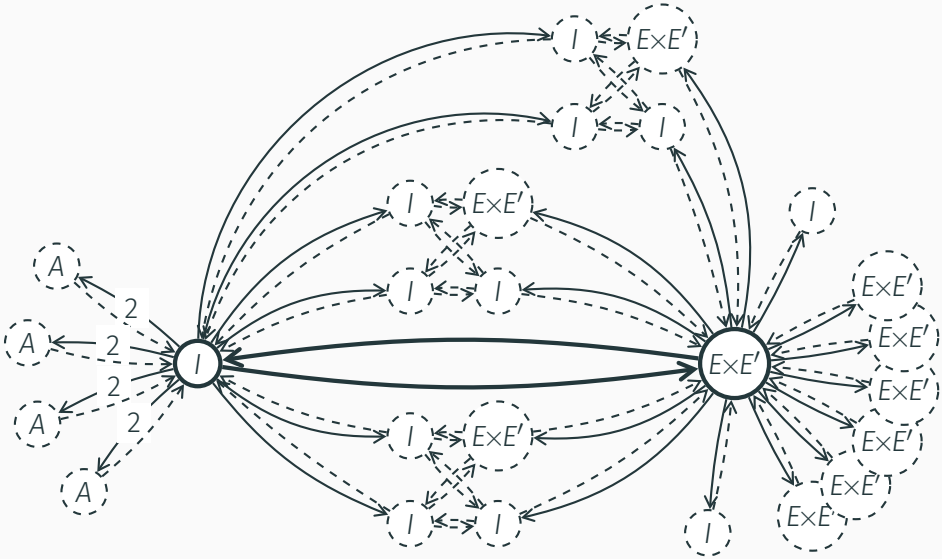
where $\mathsf{RA}(\mathcal{X}) := \mathsf{Aut}(\mathcal{X})/\langle \pm 1 \rangle$.

Type-I (RA $\cong C_2$) and elliptic product neighbourhoods

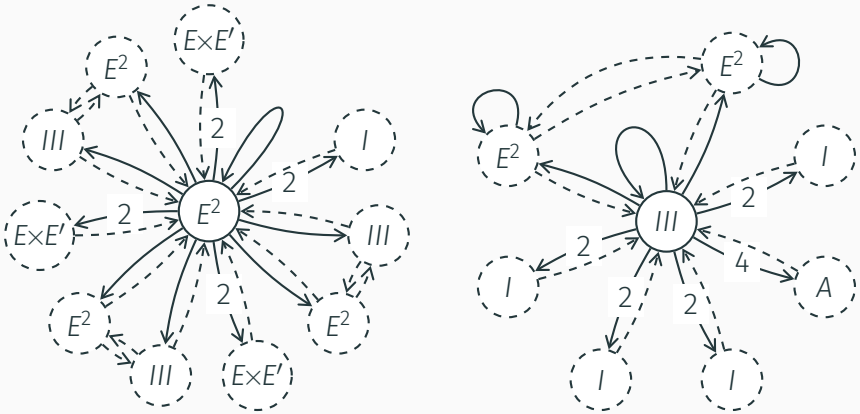


These vertices are far from isolated: there are $O(p^2)$ of them in the graph.

Neighbourhoods of edges from Type-I vertices to product neighbours

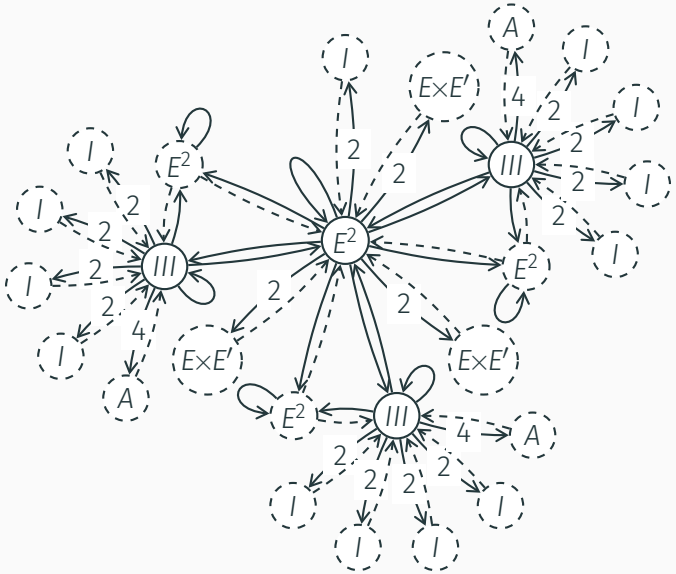


Type-III (RA $\cong C_2^2$) and elliptic square neighbourhoods

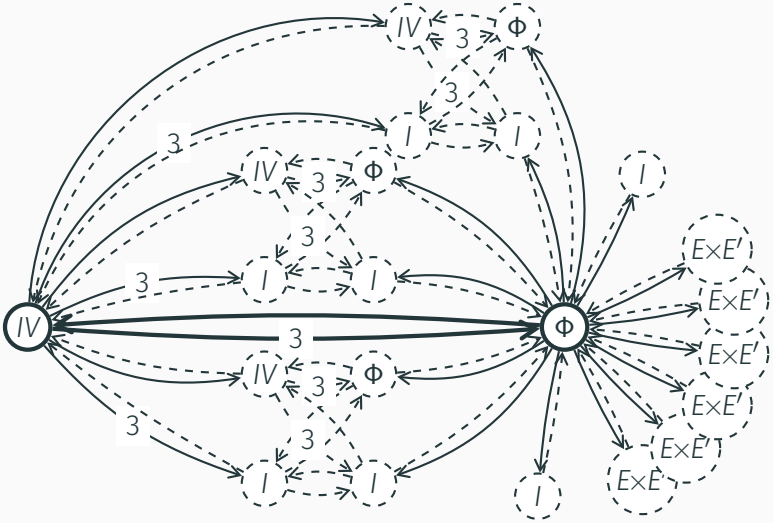


These vertices are far from isolated: there are $O(p)$ of them in the graph.

Connecting Type-III and elliptic-square vertices



Type-IV (RA $\cong S_3$) and products Φ of 3-isogenous elliptic curves



Solving the isogeny problem in $g > 1$

Results

Theorem (Costello–S., PQCrypto 2020):

1. There exists a **classical algorithm** which solves isogeny problems in $\Gamma_g(\ell; p)$ with probability $\geq 1/2^{g-1}$ in expected time $\widetilde{O}(p^{g-1}/P)$ on P processors as $p \to \infty$ (with ℓ fixed).
2. There exists a **quantum algorithm** which solves isogeny problems in $\Gamma_g(\ell; p)$ in expected time $\widetilde{O}(\sqrt{p^{g-1}})$ as $p \to \infty$ (with ℓ fixed).

This talk: the classical algorithm.

Attacking the isogeny problem

Recall: if we just view $\Gamma_g(\ell; p)$ as a generic $N_g(\ell)$-regular Ramanujan graph, then solving the path-finding problem would cost $O(p^{g(g+1)/4})$ (classical) isogeny steps.

Key observation: in $g = 2$, we have $\#S_2(p)^E > \sqrt{\#S_2(p)^J}$. This pattern continues in $g > 2$. We beat square-root algorithms by exploiting this special subset.

Let's look at the algorithm for $g = 2$ first. Recursive application will give us $g > 2$.

The algorithm in $g = 2$: Step 1

Step 1: Compute paths from our target PPASes into elliptic product vertices:

$$\phi : \mathcal{A} \to \cdots \to \mathcal{E}_1 \times \mathcal{E}_2 \in S_2(p)^E$$
$$\phi' : \mathcal{A}' \to \cdots \to \mathcal{E}_1' \times \mathcal{E}_2' \in S_2(p)^E$$

Expander hypothesis $\implies$ we find ϕ (and ϕ') after $O(p)$ random walks of length in $O(\log p)$: total cost is $\widetilde{O}(p/P)$ isogeny steps on P classical processors.

It remains to compute a path $\mathcal{E}_1 \times \mathcal{E}_2 \to \cdots \to \mathcal{E}_1' \times \mathcal{E}_2'$ in $\Gamma_2(\ell; p)$ in $\widetilde{O}(p)$ steps.

The algorithm in $g = 2$: Step 2

Step 2: to compute a path $\mathcal{E}_1 \times \mathcal{E}_2 \to \cdots \to \mathcal{E}'_1 \times \mathcal{E}'_2$ in $\Gamma_2(\ell; p)$,

1. Compute paths $\psi_1 : \mathcal{E}_1 \to \cdots \to \mathcal{E}'_1$ and $\psi_2 : \mathcal{E}_2 \to \cdots \to \mathcal{E}'_2$ in $\Gamma_1(\ell; p)$.
2. If $\text{length}(\psi_1) \not\equiv \text{length}(\psi_2) \pmod 2$, then go back to Step 1 (or swap $\mathcal{E}_1 \leftrightarrow \mathcal{E}_2$).
3. Trivially **stretch** the shorter of the ψ_i to the same length as the other, by stepping back and forth on the last component isogeny.
4. Compose the products of the i-th components of ψ_1 and ψ_2 to get a path

$$\psi^\times : \mathcal{E}_1 \times \mathcal{E}_2 \to \cdots \to \mathcal{E}'_1 \times \mathcal{E}'_2 \qquad \text{in } \Gamma_2(\ell; p)\,.$$

Cost: same as solving the isogeny problem in $\Gamma_1(\ell; p)$, i.e. $O(\sqrt{p}/P)$.

The composition $(\phi')^\dagger \circ \psi^\times \circ \phi$ is a path from $\mathcal{A}$ to $\mathcal{A}'$ in $\Gamma_2(\ell; p)$.

We can thus **solve the isogeny problem** in $\Gamma_2(\ell; p)$ in $\widetilde{O}(p)$ isogeny steps.

Attacking higher genus

The same idea works **in higher dimension** as follows.

Recall: $\#S_g(p) = O(p^{g(g+1)/2})$, so classical square-root algorithms solve the isogeny problem in $\Gamma_g(\ell; p)$ in $O(p^{g(g+1)/4})$ isogeny steps.

Let $T_g(p)$ be the image of $S_1(p) \times S_{g-1}(p)$ in $S_g(p)$ (product polarization).

We have $\#S_1(p) = O(p)$ and $\#S_{g-1}(p) = O(p^{g(g-1)/2})$, so

$$\#T_g(p) = O(p^{(g^2-g+2)/2})\,;$$

so the probability that a random $\mathcal{A}$ in $S_g(p)$ is in $T_g(p)$ is in $O(1/p^{(g-1)})$.

Key observation: $g - 1 < g(g+1)/4$ (and much smaller for large g).

We should be able to efficiently recognise steps into $T_g(p)$ by something analogous to the breakdown in Richelot's formulæ in $g = 2$ (theta relations?).

Solving the general isogeny problem

To find a path from $\mathcal{A}$ to $\mathcal{A}'$ in $\Gamma_g(\ell; p)$:

1. Compute paths $\phi : \mathcal{A} \to \mathcal{E} \times \mathcal{B} \in T_g(p)$ and $\phi' : \mathcal{A}' \to \mathcal{E}' \times \mathcal{B}' \in T_g(p)$ in $\Gamma_g(\ell; p)$
   Expander hypothesis $\implies$ $\widetilde{O}(p^{g-1}/P)$ *isogeny steps. Dominant step*
2. Compute a path $\psi_E : \mathcal{E} \to \cdots \to \mathcal{E}'$ in $\Gamma_1(\ell; p)$
   Usual elliptic algorithm $\implies$ $O(\sqrt{p}/P)$ *isogeny steps*
3. **Recurse** to compute a path $\psi_B : \mathcal{B} \to \cdots \to \mathcal{B}'$ in $\Gamma_{g-1}(\ell; p)$
   Expander hypothesis $\implies$ $\widetilde{O}(p^{g-2}/P)$ *isogeny steps*
4. Apply the elliptic isogeny-glueing technique to get the final path.
   Probability of compatible lengths: $1/2^{g-1}$.

Total cost: $\widetilde{O}(p^{g-1}/P)$, dominated by the cost of walking into $T_g(p)$ in Step 1.
Much faster than $O(p^{g(g+1)/4})$.

Cryptographic implications

Genus-2 isogeny-based hashing is **less efficient** than the elliptic equivalent.

Say we want to force $\approx 2^\lambda$ classical effort to compute preimages:

genus 1 $\approx p/12$ vertices, square-root preimage finding $\implies$ need $p \approx 2^{2\lambda}$
- 4λ-bit outputs in $\mathbb{F}_{p^2} \to 2\lambda$-bit outputs in $\mathbb{F}_p$ (2x the ideal)
- Slow! Each input bit $\implies$ square root in $\mathbb{F}_{p^2}$ $\implies$ $2 \times 2\lambda$-bit modular exponentiations

genus 2 $\approx p^3/2880$ vertices, **cube-root** preimage finding $\implies$ need $p \approx 2^\lambda$
- $\implies$ 6λ-bit outputs in $\mathbb{F}_{p^2}^3$ (moduli point) $\to 3\lambda$ bits (finalization to shorter hash values is unclear.
- A little faster! Each 3-bit input digit $\implies$ $3\times$ square roots in $\mathbb{F}_{p^2}$ $\implies$ $3 \times 2 \times \lambda$-bit modular exponentiations

No easy fix for the collision-resistance issue (starting vertex) in either case.

(Some) references

Selected references 1

- Brock + Howe: *Purely inseparable Richelot isogenies* https://arxiv.org/abs/2002.02122
- Castryck, Decru, Smith: *Hash functions from superspecial genus-2 curves using Richelot isogenies* https://ia.cr/2019/296
- Charles, Goren, Lauter: *Cryptographic hash functions from expander graphs* https://ia.cr/2006/021
- Costello + Smith: *The supersingular isogeny problem in genus 2 and beyond* https://ia.cr/2019/1387
- Doliskani, Pereira, Barreto: *Faster Cryptographic Hash Function From Supersingular Isogeny Graphs* https://ia.cr/2017/1202

Selected references 2

- Florit + Smith: *Automorphisms and isogeny graphs of abelian varieties, with applications to the superspecial Richelot isogeny graph* https://ia.cr/2021/012
- Florit + Smith: *An atlas of the Richelot isogeny graph* https://ia.cr/2021/013
- Jordan + Zaytman: *Isogeny graphs of superspecial abelian varieties and Brandt matrices* https://arxiv.org/abs/2005.09031
- Renes + Smith: *qDSA: Small and Secure Digital Signatures with Curve-based Diffie-Hellman Key Pairs* https://ia.cr/2017/518
- Smith: *Isogenies and the discrete logarithm problem on Jacobians of genus 3 hyperelliptic curves* https://ia.cr/2007/428