Attacking the Supersingular Isogeny Problem:
From the Delfs–Galbraith algorithm to oriented graphs

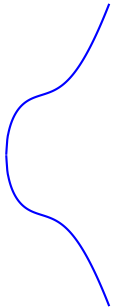
Arthur Herlédan Le Merdy

COSIC, KU Leuven

Cryptography seminar January 23, 2026, Rennes

Elliptic curves

Let p be a large prime.



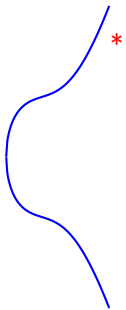
$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

Elliptic curves are:

- smooth projective curves given by affine models such as on the left.

Elliptic curves

Let p be a large prime.



$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

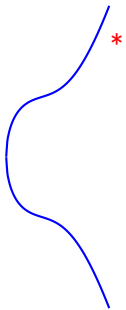
Elliptic curves are:

- smooth projective curves given by affine models such as on the left.

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

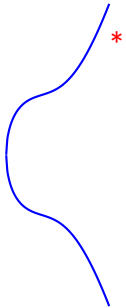
Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

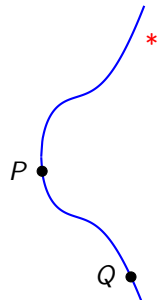
Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem
- everywhere in modern cryptography (70% of WWW traffic in 2018)

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

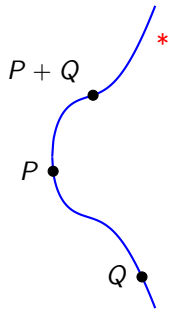
Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem
- everywhere in modern cryptography (70% of WWW traffic in 2018)

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

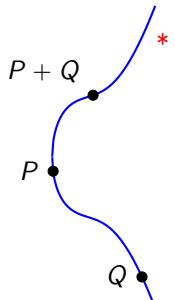
Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem
- everywhere in modern cryptography (70% of WWW traffic in 2018)

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem
- everywhere in modern cryptography (70% of WWW traffic in 2018)

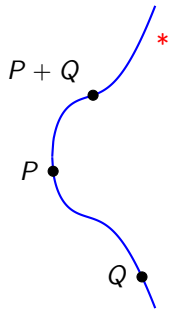
$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

From now on, elliptic curves are **supersingular** and considered **up to isomorphisms.**

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem
- everywhere in modern cryptography (70% of WWW traffic in 2018)

$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

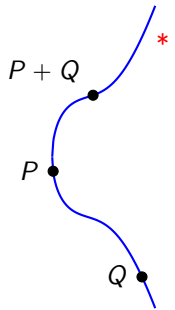
From now on, elliptic curves are **supersingular** and considered **up to isomorphisms.**
For the latter, we use the j-**invariant**,

$$j(E) := 1728\frac{4A^3}{4A^3 + 27B^2} \in \mathbb{F}_{p^2}$$

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Elliptic curves

Let p be a large prime.



Elliptic curves are:

- smooth projective curves given by affine models such as on the left.
- central objects in mathematics from ancient Greece to Fermat's last theorem
- everywhere in modern cryptography (70% of WWW traffic in 2018)

$E : y^2 = x^3 + Ax + B, \quad A, B \in \bar{\mathbb{F}}_p$

From now on, elliptic curves are **supersingular** and considered **up to isomorphisms.**
For the latter, we use the j-**invariant**,

$$j(E) := 1728\frac{4A^3}{4A^3 + 27B^2} \in \mathbb{F}_{p^2} \qquad E \simeq E' \text{ if and only if } j(E) = j(E').$$

*Non-contractual picture of a curve over $\bar{\mathbb{F}}_p$.

Hard problems in Cryptography

Let C be a cryptosystem.

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

Security of C

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If Q is *easy*, then C is *broken*.

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If Q is *easy*, then C is *broken*.

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If P is *hard*, then C is *secure*.
- If Q is *easy*, then C is *broken*.

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If P is *hard*, then C is *secure*.
- If Q is *easy*, then C is *broken*.
- If $P \rightleftarrows Q$, then the *security* of C is **equivalent** to the *hardness* of P.

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If P is *hard*, then C is *secure*.
- If Q is *easy*, then C is *broken*.
- If $P \rightleftarrows Q$, then the *security* of C is **equivalent** to the *hardness* of P.

Classical hard problems: Integer Factorization,
Discrete Logarithms

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If P is *hard*, then C is *secure*.
- If Q is *easy*, then C is *broken*.
- If $P \rightleftarrows Q$, then the *security* of C is **equivalent** to the *hardness* of P.

Classical hard problems: Integer Factorization,
Discrete Logarithms (Elliptic Curves Cryptography)

Hard problems in Cryptography

Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If P is *hard*, then C is *secure*.
- If Q is *easy*, then C is *broken*.
- If $P \rightleftarrows Q$, then the *security* of C is **equivalent** to the *hardness* of P.

Classical hard problems: ~~Integer Factorization~~,
~~Discrete Logarithms (Elliptic Curves Cryptography)~~
Broken by quantum computers! [Sho94]

Hard problems in Cryptography

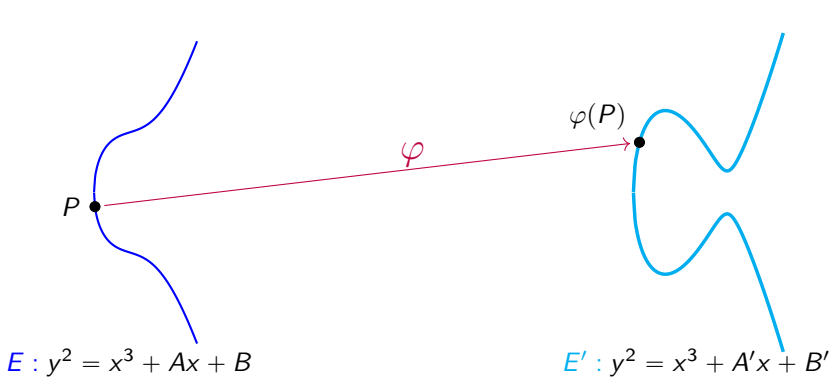
Let C be a cryptosystem. **Breaking the security of C** is a **computational problem**.

- If P is *hard*, then C is *secure*.
- If Q is *easy*, then C is *broken*.
- If $P \rightleftarrows Q$, then the *security* of C is **equivalent** to the *hardness* of P.

Classical hard problems: ~~Integer Factorization~~,
~~Discrete Logarithms (Elliptic Curves Cryptography)~~
Broken by quantum computers! [Sho94]

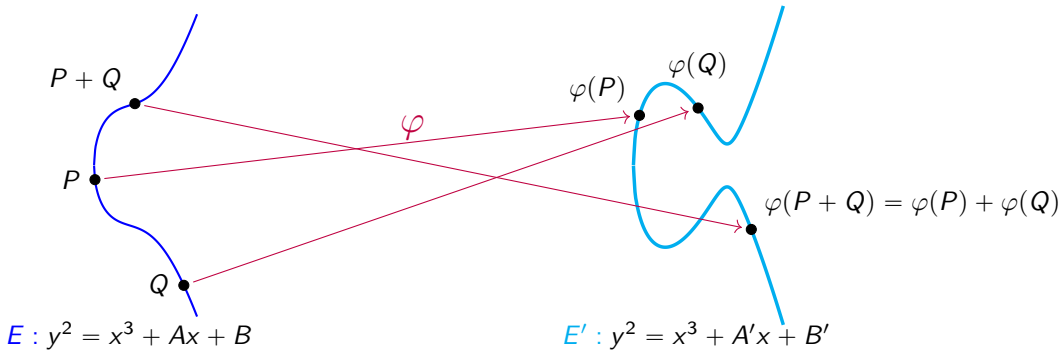
Post-quantum candidates: Lattice-based, Code-based,
Isogeny-based cryptography (Elliptic Curves come-back!)

Isogenies



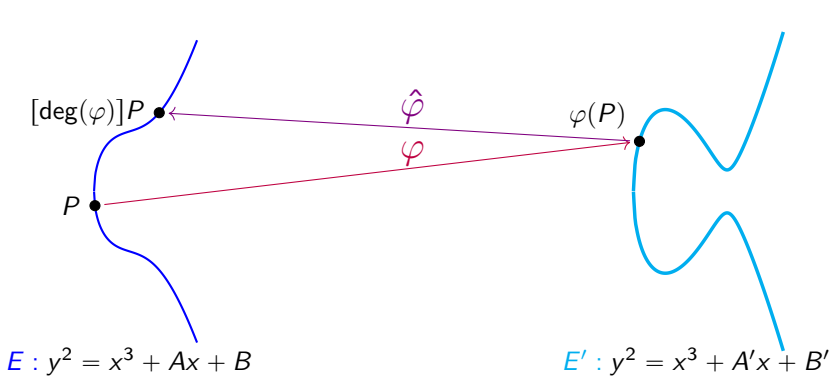
An **isogeny** φ is a "nice map" between two elliptic curves

Isogenies



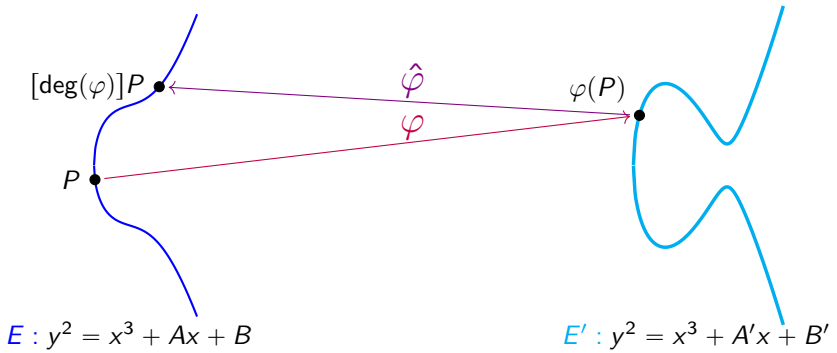
An **isogeny** φ is a "nice map" between two elliptic curves, i.e. a rational map inducing **a group morphism**

Isogenies



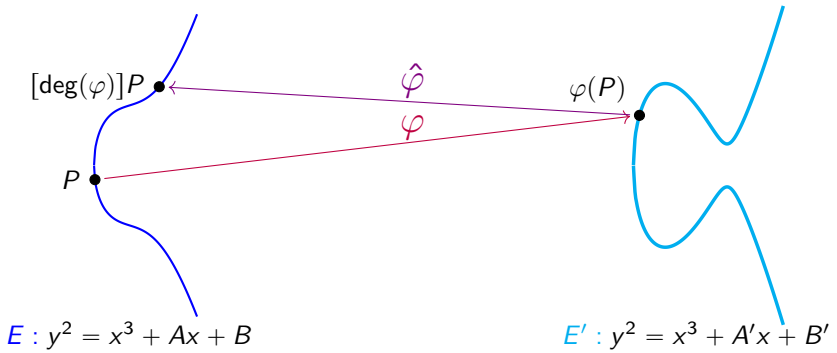
An **isogeny** φ is a "nice map" between two elliptic curves, i.e. a rational map inducing **a group morphism**, admitting a **dual map** $\hat{\varphi}$

Isogenies



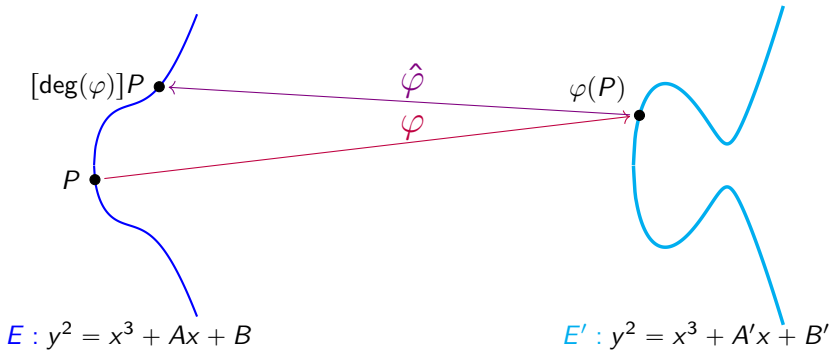
An **isogeny** φ is a "nice map" between two elliptic curves, i.e. a rational map inducing **a group morphism**, admitting a **dual map** $\hat{\varphi}$, having **a finite kernel**, being **surjective**, ...

Isogenies



An **isogeny** φ is a "nice map" between two elliptic curves, i.e. a rational map inducing **a group morphism**, admitting a **dual map** $\hat{\varphi}$, having **a finite kernel**, being **surjective**, ...

Isogenies



An **isogeny** φ is a "nice map" between two elliptic curves, i.e. a rational map inducing **a group morphism**, admitting a **dual map** $\hat{\varphi}$, having **a finite kernel**, being **surjective**, ...

Isogeny-based cryptography

CGL hash function

Isogeny-based cryptography

CGL hash function

CSIDH key exchange

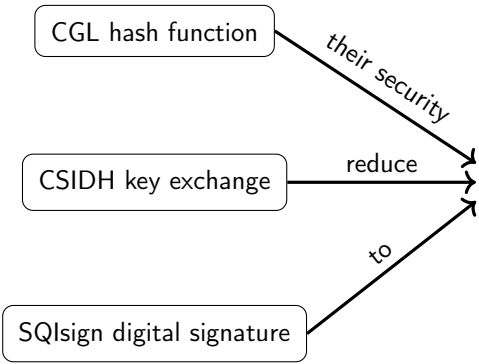
Isogeny-based cryptography

CGL hash function

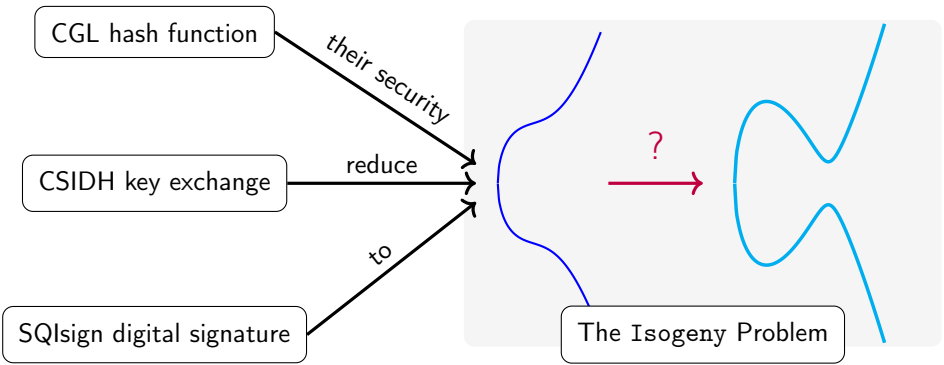
CSIDH key exchange

SQIsign digital signature

Isogeny-based cryptography



Isogeny-based cryptography



The ℓ-IsogenyPath Problem

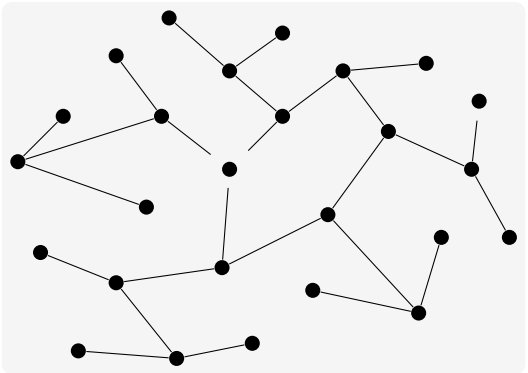
Let $\ell \neq p$ be a prime,

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



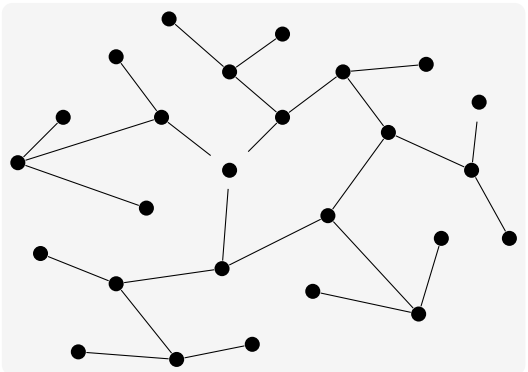
The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

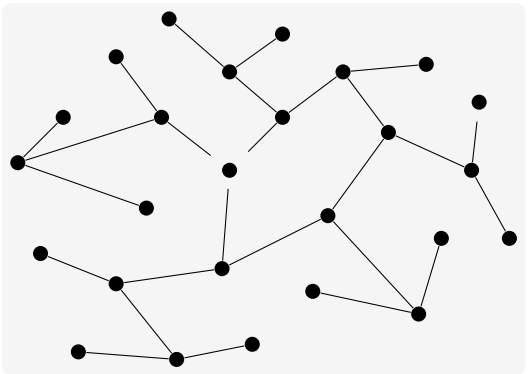
Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

$$E_1 \underset{\ell\text{-isogenous}}{\text{——}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

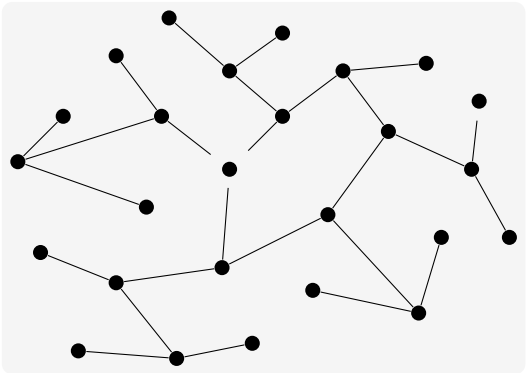
$$E_1 \underset{\ell\text{-isogenous}}{\text{——}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

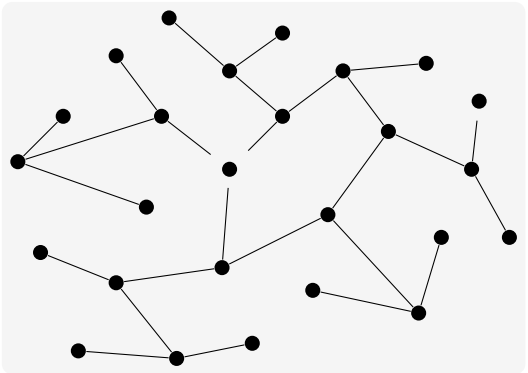
$$E_1 \underset{\ell\text{-isogenous}}{\text{---}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

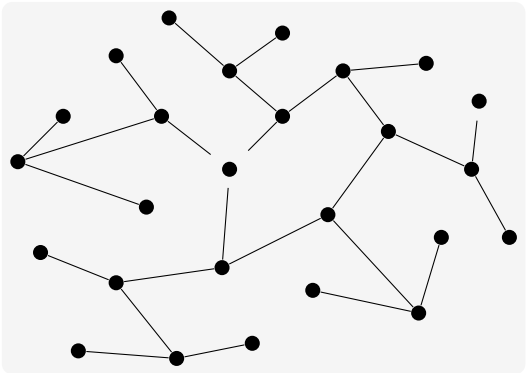
$$E_1 \underset{\ell\text{-isogenous}}{\text{---}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **rapidly mixing**,
- **huge** ($\approx p/12$ vertices),

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

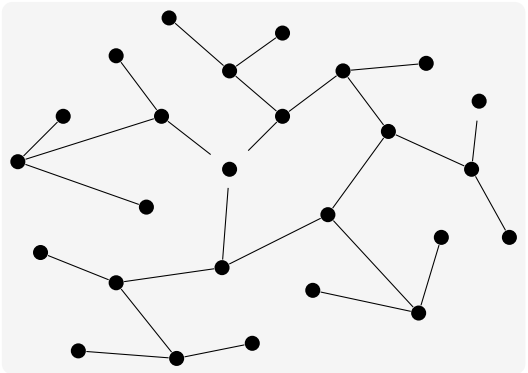
$$E_1 \underset{\ell\text{-isogenous}}{\text{---}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),
- **rapidly mixing**,
- **connected**.

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

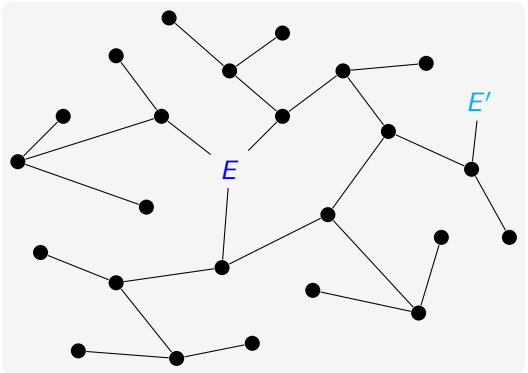
$$E_1 \underset{\ell\text{-isogenous}}{\text{---}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),
- **rapidly mixing**,
- **connected**.

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

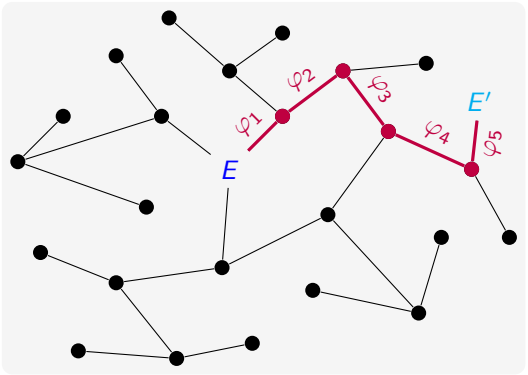
$$E_1 \underset{\ell\text{-isogenous}}{\text{——}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),
- **rapidly mixing**,
- **connected**.

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

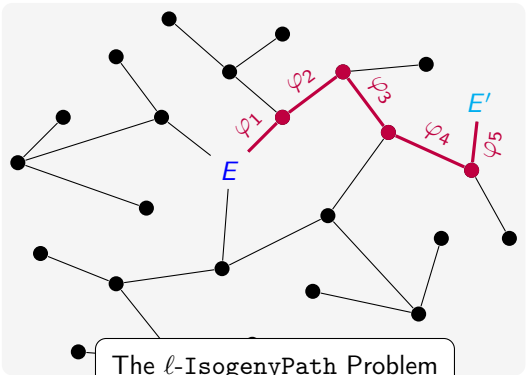
$$E_1 \underset{\ell\text{-isogenous}}{\text{---}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),
- **rapidly mixing**,
- **connected**.

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

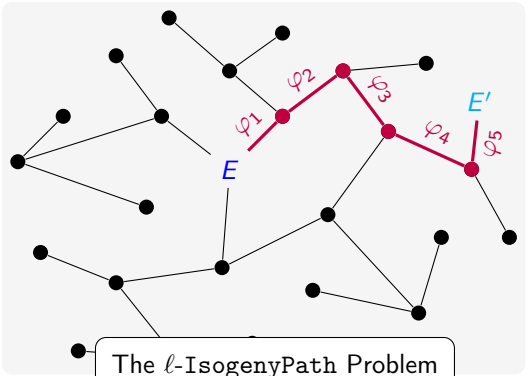
$$E_1 \underset{\ell\text{-isogenous}}{\text{---}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),
- **rapidly mixing**,
- **connected**.

The ℓ-IsogenyPath Problem

Let $\ell \neq p$ be a prime, e.g. $\ell = 2$.



The ℓ-IsogenyPath Problem

⇕ [Wes22] under GRH

The Isogeny Problem

The ℓ-**isogeny graph** has

vertices: supersingular elliptic curves up to **isomorphism**

edges: isogenies of degree ℓ

Modular Polynomials

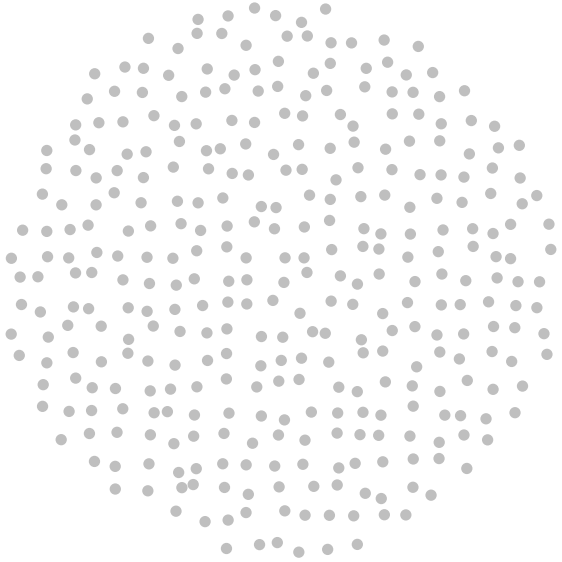
The modular polynomial $\Phi_\ell(X, Y)$ of level ℓ parametrizes ℓ-isogenies between elliptic curves:

$$E_1 \underset{\ell\text{-isogenous}}{\text{——}} E_2 \quad \text{iff} \quad \Phi_\ell(j(E_1), j(E_2)) = 0.$$

The ℓ-**isogeny graph** is

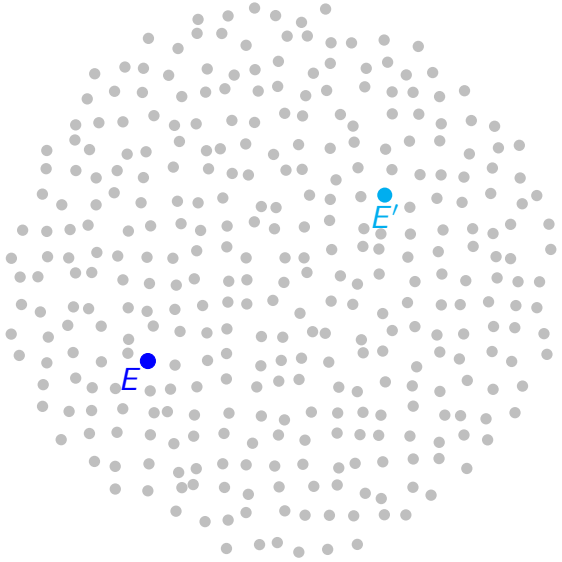
- $(\ell + 1)$-**regular**,
- **huge** ($\approx p/12$ vertices),
- **rapidly mixing**,
- **connected**.

The naive meet-in-the-middle approach



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

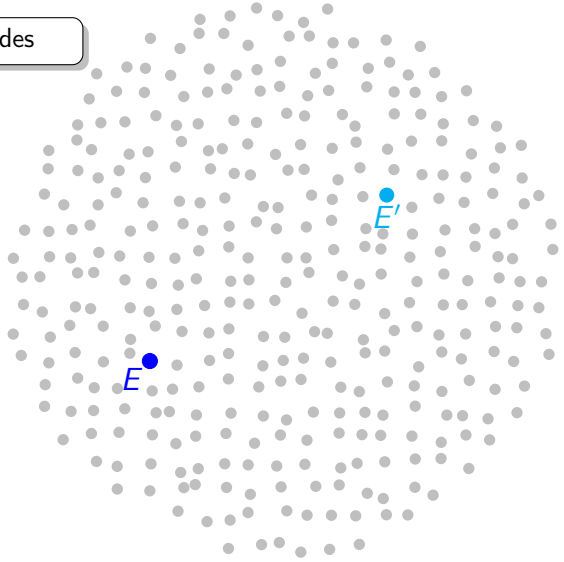
The naive meet-in-the-middle approach



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

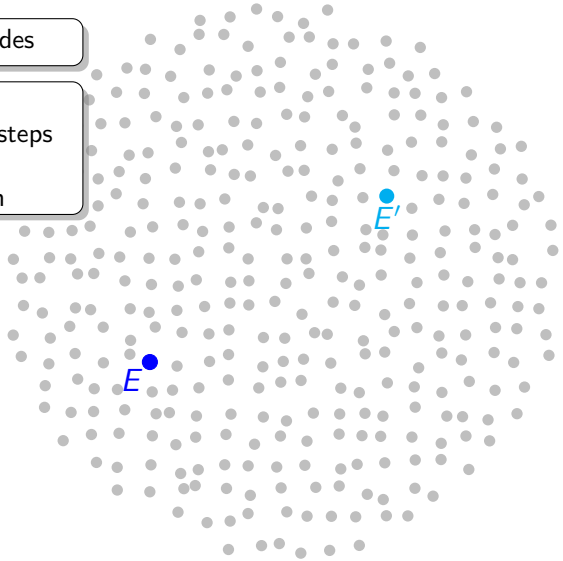


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

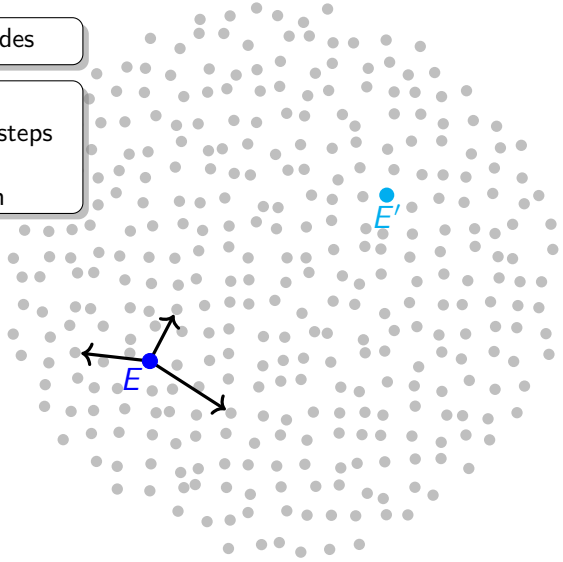


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

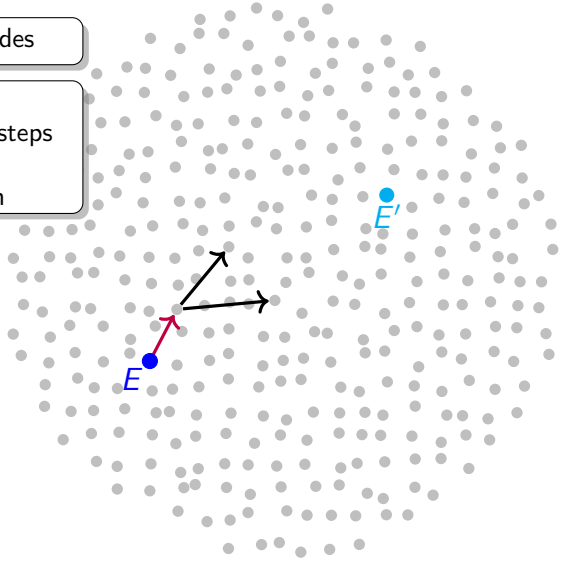


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

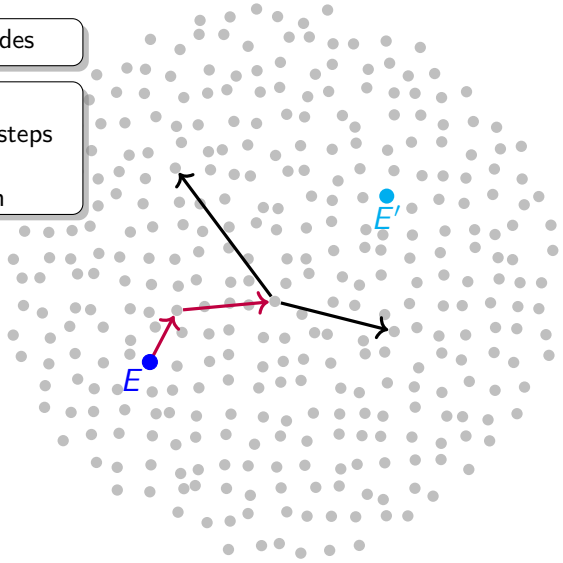


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

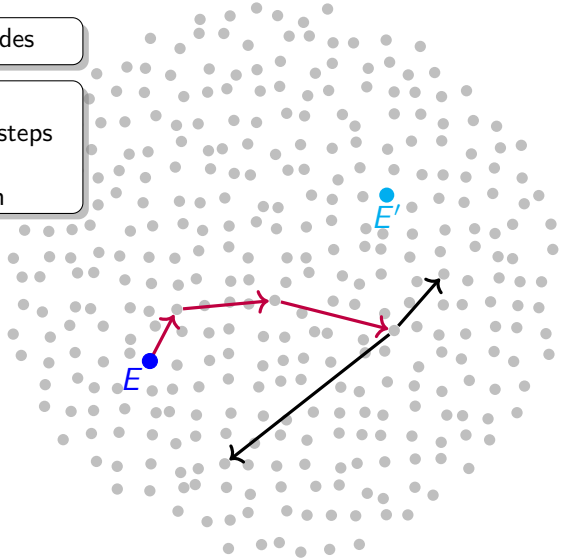


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

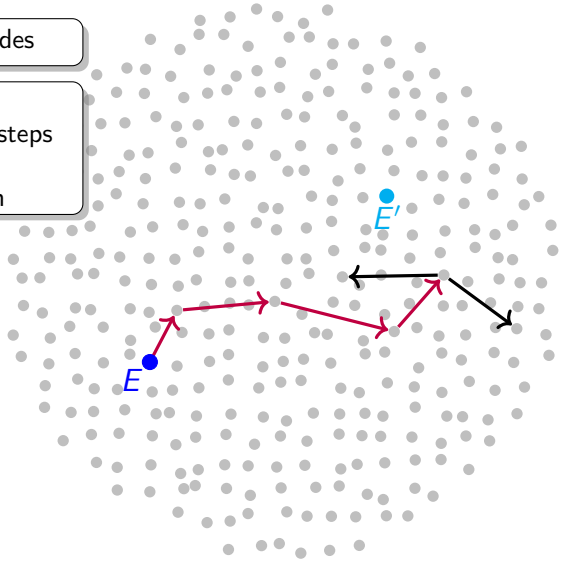


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

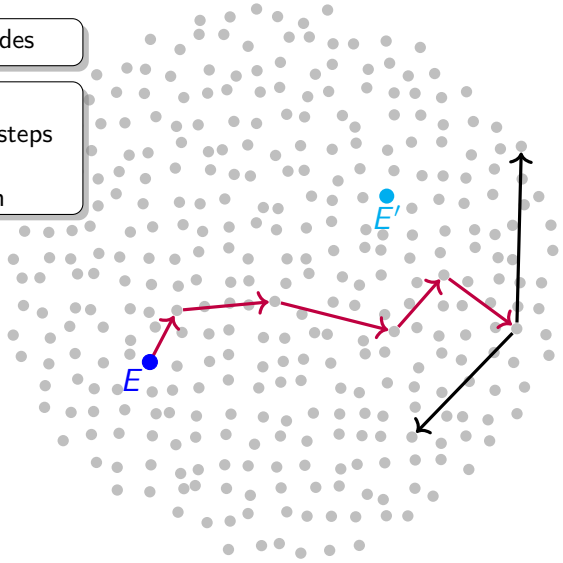


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

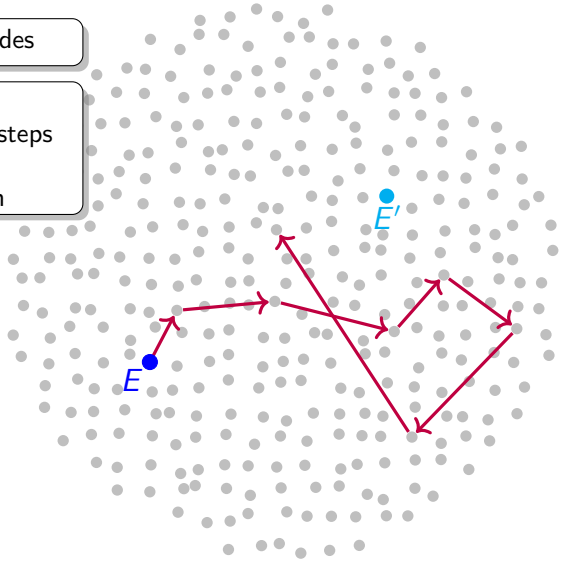


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

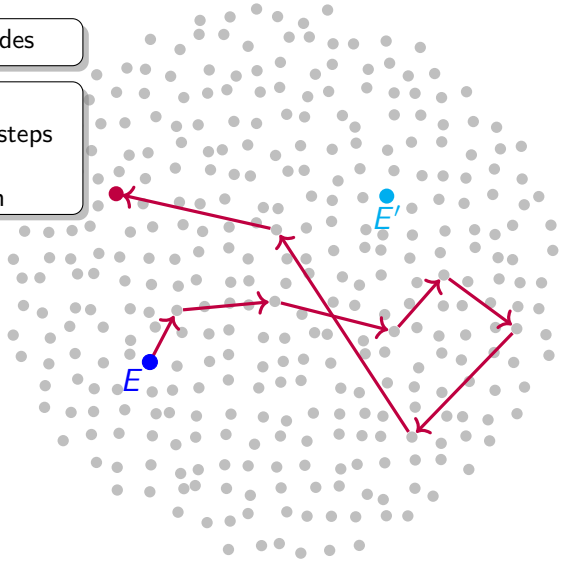


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

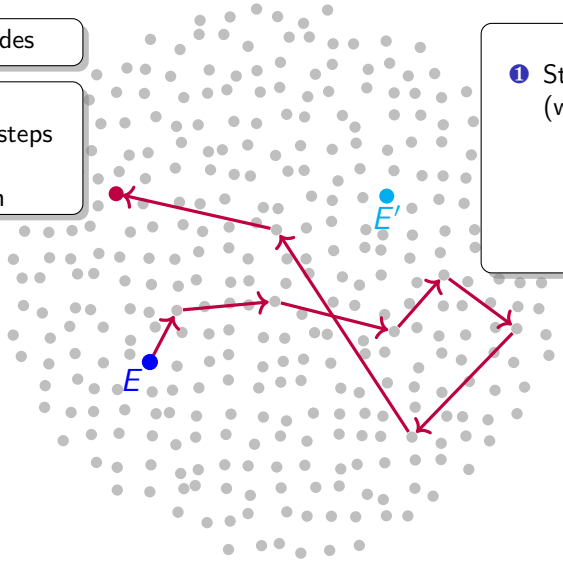
The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

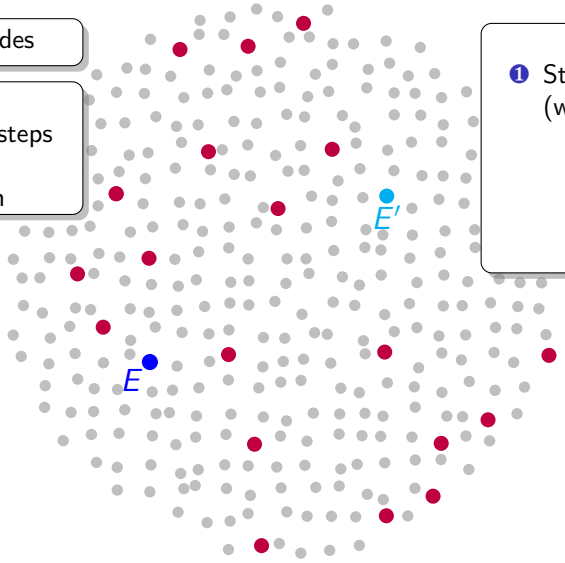
The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E

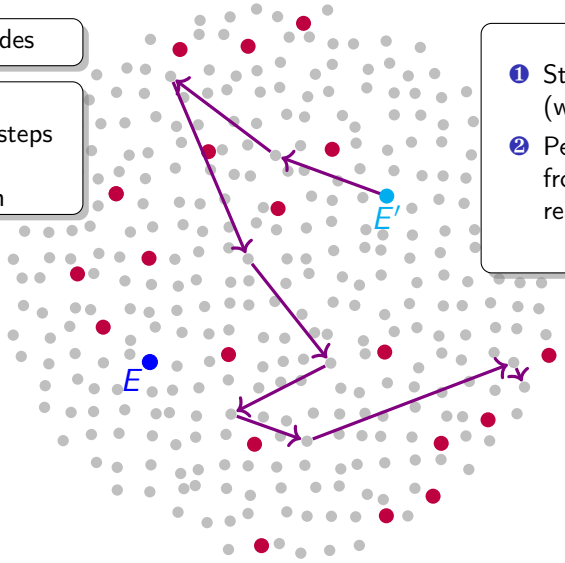


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

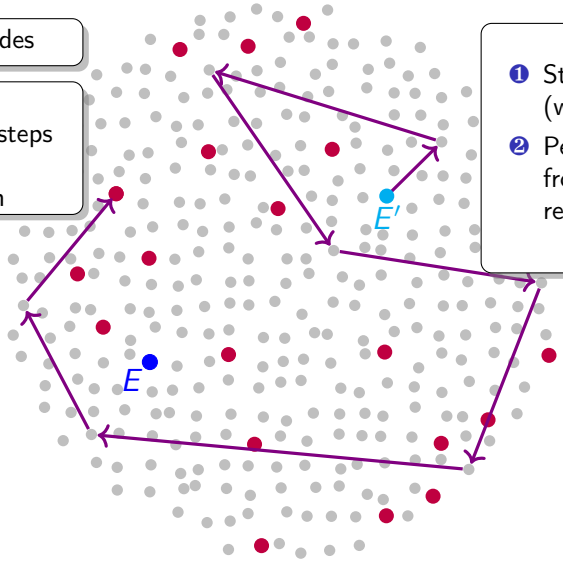
1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

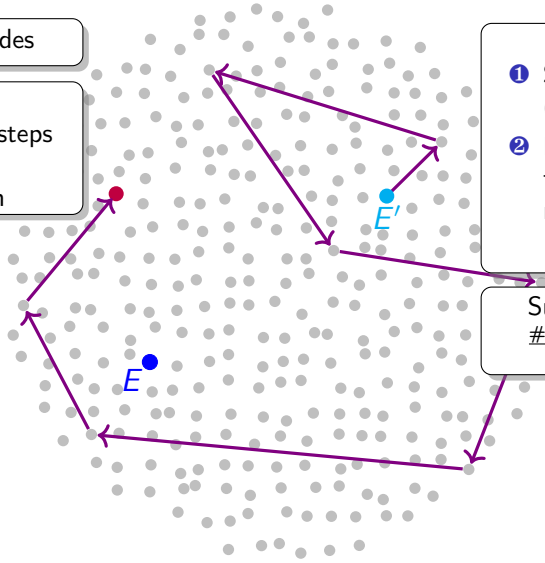
1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve

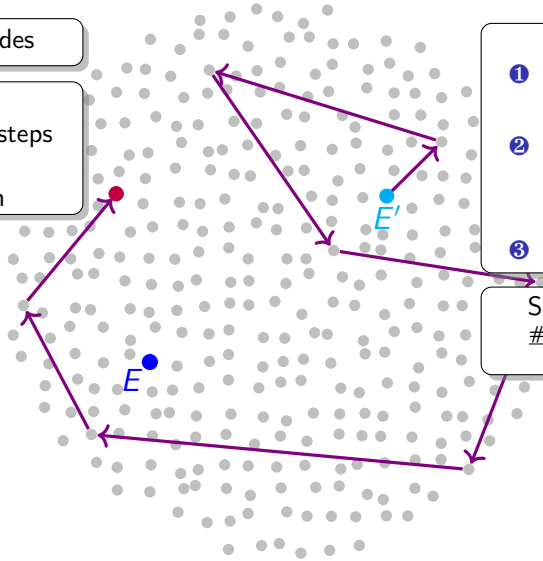
Success probability for 2 is
$\frac{\#\text{ stored curves}}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve
3. Combine the paths

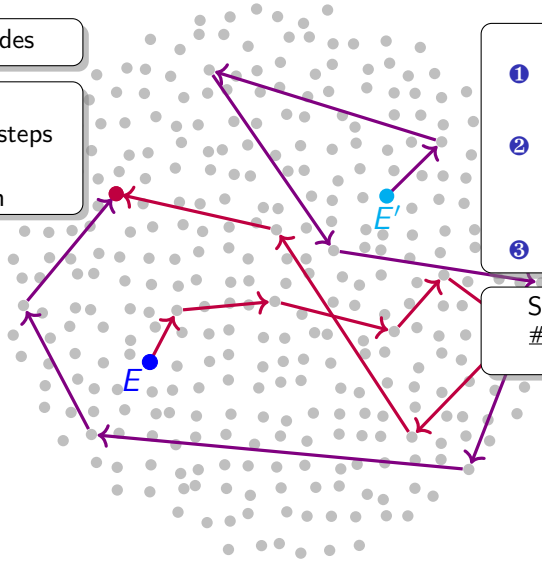
Success probability for 2 is
$\frac{\#\text{ stored curves}}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve
3. Combine the paths

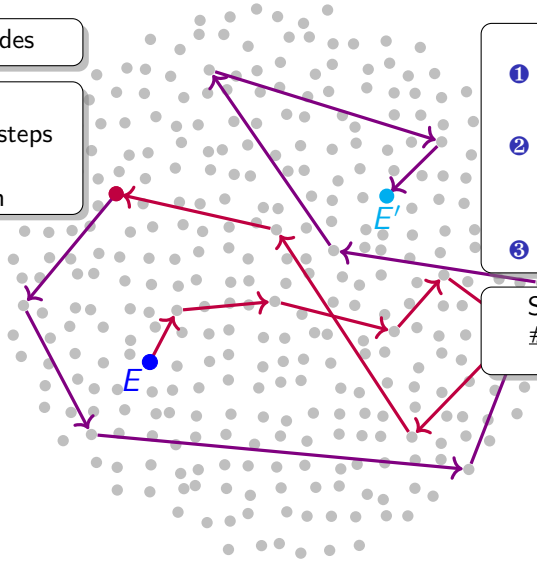
Success probability for 2 is
$\frac{\#\text{ stored curves}}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve
3. Combine the paths

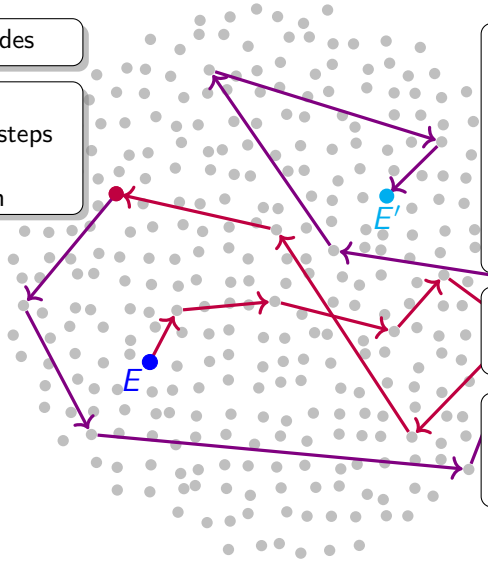
Success probability for 2 is
$\frac{\#\text{ stored curves}}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve
3. Combine the paths

Success probability for ❷ is
$\frac{\#\text{ stored curves}}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

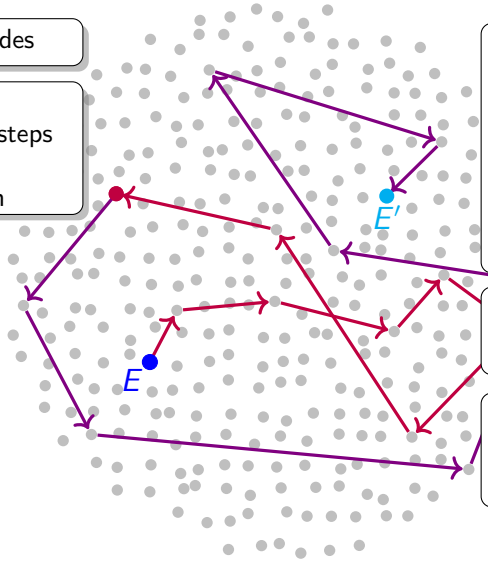
Complexity:
Time: $\tilde{O}(\sqrt{p})$
Memory: $\tilde{O}(\sqrt{p})$

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The naive meet-in-the-middle approach

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



Method:

1. Store $\approx \sqrt{p}$ random curves (with their path) from E
2. Perform random walks from E' until reaching a stored curve
3. Combine the paths

Success probability for 2 is
$\frac{\#\text{ stored curves}}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

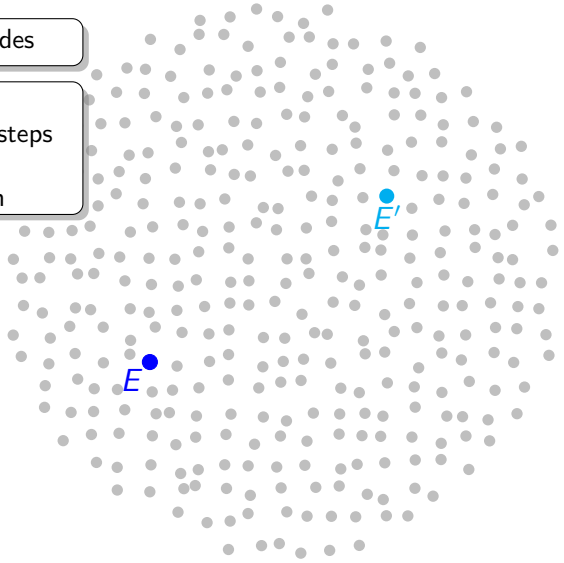
Complexity:
Time: $\tilde{O}(\sqrt{p})$
Memory: $\tilde{O}(\sqrt{p})$

A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution



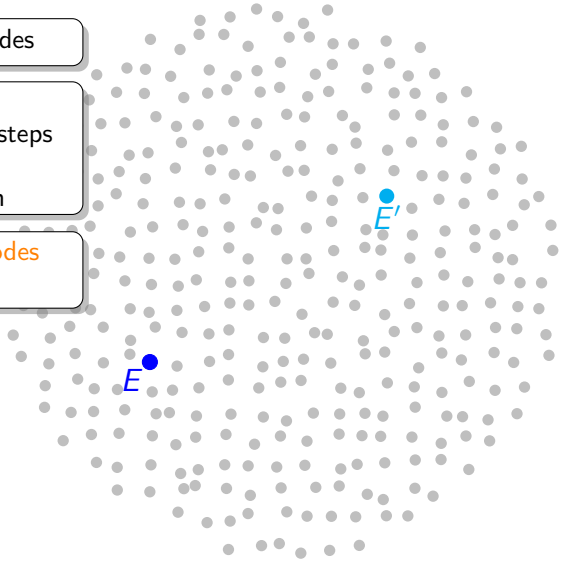
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



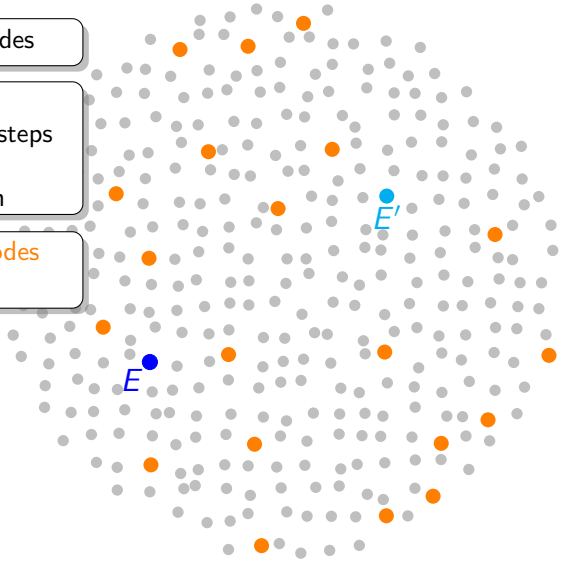
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

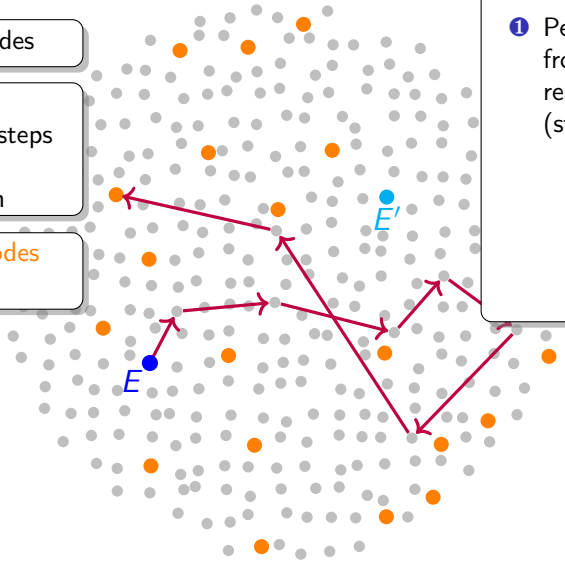
There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$

Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)



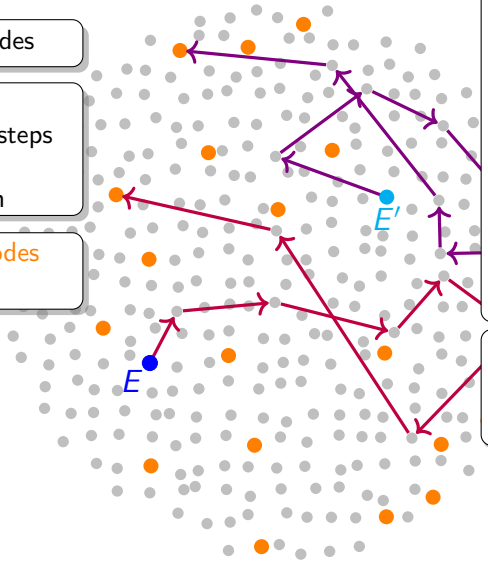
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)
2. Perform random walks from E' until reaching a curve over $\mathbb{F}_p$ (store the path)

Success probabilities for 1 and 2 are
$\frac{\#\text{ curves over } \mathbb{F}_p}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

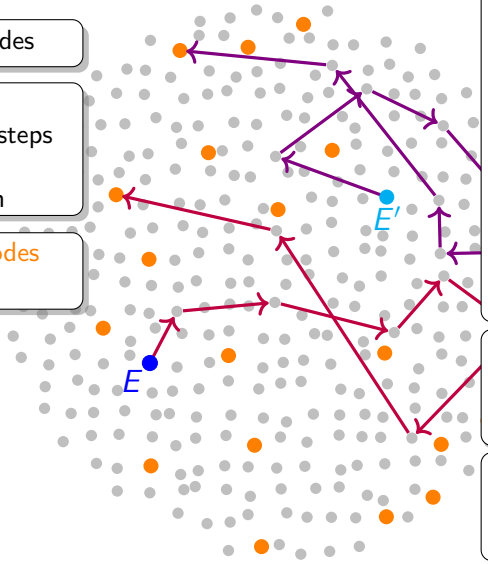
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)
2. Perform random walks from E' until reaching a curve over $\mathbb{F}_p$ (store the path)

Success probabilities for 1 and 2 are
$\frac{\#\text{ curves over } \mathbb{F}_p}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

Complexity (so far):
Time: $\tilde{O}(\sqrt{p})$
Memory: $\text{polylog}(p)$

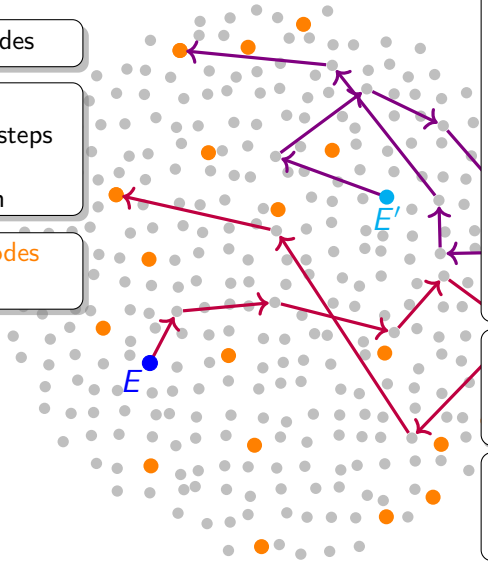
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)
2. Perform random walks from E' until reaching a curve over $\mathbb{F}_p$ (store the path)

Success probabilities for 1 and 2 are
$\frac{\#\text{ curves over } \mathbb{F}_p}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

Complexity (so far):
Time: $\tilde{O}(\sqrt{p})$
Memory: $\text{polylog}(p)$

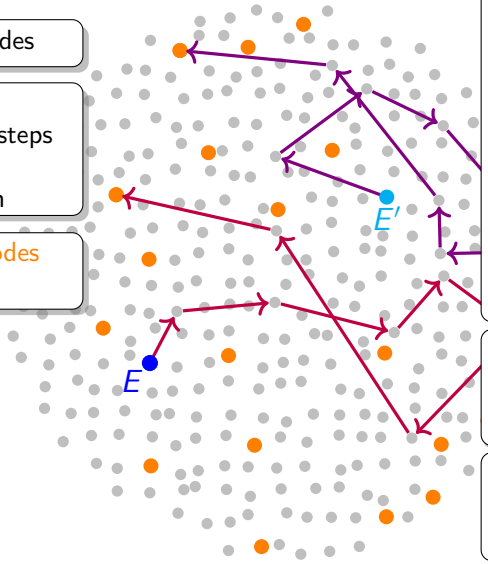
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)
2. Perform random walks from E' until reaching a curve over $\mathbb{F}_p$ (store the path)
3. Combine the paths

Success probabilities for 1 and 2 are
$\frac{\#\text{ curves over } \mathbb{F}_p}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

Complexity (so far):
Time: $\tilde{O}(\sqrt{p})$
Memory: $\text{polylog}(p)$

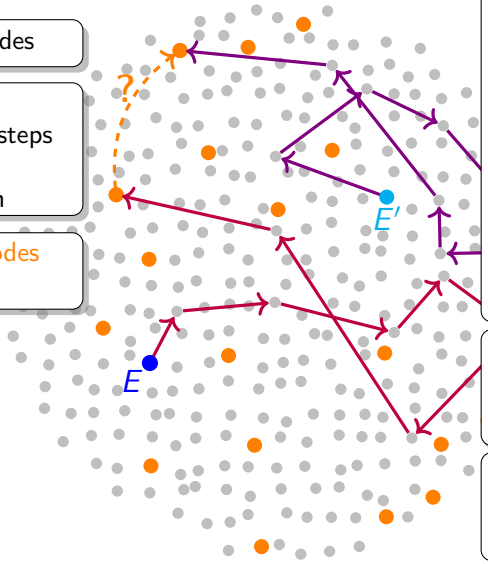
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)
2. Perform random walks from E' until reaching a curve over $\mathbb{F}_p$ (store the path)
3. Combine? the paths

Success probabilities for 1 and 2 are
$\frac{\#\text{ curves over } \mathbb{F}_p}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

Complexity (so far):
Time: $\tilde{O}(\sqrt{p})$
Memory: $\text{polylog}(p)$

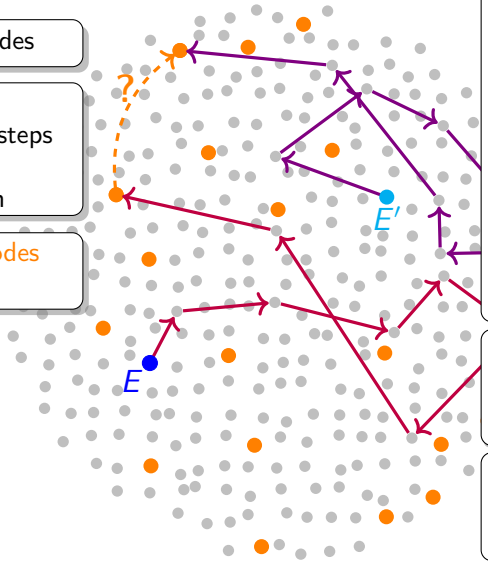
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm [DG16]

There are around $\frac{p}{12}$ nodes

Ramanujan graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution

There are around $\sqrt{p}$ nodes
defined over $\mathbb{F}_p$



Method:

1. Perform random walks from E until reaching a curve over $\mathbb{F}_p$ (store the path)
2. Perform random walks from E' until reaching a curve over $\mathbb{F}_p$ (store the path)
3. **Connect** the paths

Success probabilities for 1 and 2 are
$\frac{\#\text{ curves over } \mathbb{F}_p}{\#\text{ curves}} \approx \frac{\sqrt{p}}{p} = \frac{1}{\sqrt{p}}$

Complexity (so far):
Time: $\tilde{O}(\sqrt{p})$
Memory: $\text{polylog}(p)$

A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Endomorphism ring

An isogeny $\varphi : E \to E$ is an **endomorphism**.
We denote $\mathrm{End}(E) := \{\varphi : E \to E\} \cup \{0\}$.

Endomorphism ring

An isogeny $\varphi : E \to E$ is an **endomorphism**.
We denote $\mathrm{End}(E) := \{\varphi : E \to E\} \cup \{0\}$.

- $(\mathrm{End}(E), +, \circ)$ is the **endomorphism ring** of E, where for every $P \in E$:

$$(\varphi+\psi)(P) = \varphi(P) + \psi(P) \qquad \text{and} \qquad (\varphi\circ\psi)(P) = \varphi(\psi(P)).$$

Endomorphism ring

An isogeny $\varphi : E \to E$ is an **endomorphism**.
We denote $\mathrm{End}(E) := \{\varphi : E \to E\} \cup \{0\}$.

- $(\mathrm{End}(E), +, \circ)$ is the **endomorphism ring** of E, where for every $P \in E$:

$$(\varphi+\psi)(P) = \varphi(P) + \psi(P) \qquad \text{and} \qquad (\varphi\circ\psi)(P) = \varphi(\psi(P)).$$

- $\mathbb{Z} \hookrightarrow \mathrm{End}(E)$ as **subring**. For every $n \in \mathbb{Z}$, we have the endomorphism

$$\begin{aligned} [n] : E &\to E \\ P &\mapsto [n]P := \underbrace{P + \cdots + P}_{n \text{ times}}. \end{aligned}$$

Endomorphism ring

An isogeny $\varphi : E \to E$ is an **endomorphism**.
We denote $\mathrm{End}(E) := \{\varphi : E \to E\} \cup \{0\}$.

- $(\mathrm{End}(E), +, \circ)$ is the **endomorphism ring** of E, where for every $P \in E$:

  $$(\varphi+\psi)(P) = \varphi(P) + \psi(P) \qquad \text{and} \qquad (\varphi\circ\psi)(P) = \varphi(\psi(P)).$$

- $\mathbb{Z} \hookrightarrow \mathrm{End}(E)$ as **subring**. For every $n \in \mathbb{Z}$, we have the endomorphism

  $$\begin{aligned} [n] : E &\to E \\ P &\mapsto [n]P := \underbrace{P + \cdots + P}_{n \text{ times}}. \end{aligned}$$

- $(\mathrm{End}(E), +)$ is a lattice of dimension
  - **2** i.e. $\mathrm{End}(E) \simeq \mathbb{Z} \oplus \alpha\mathbb{Z}$,
    or
  - **4** i.e. $\mathrm{End}(E) \simeq \mathbb{Z} \oplus \alpha\mathbb{Z} \oplus \beta\mathbb{Z} \oplus \gamma\mathbb{Z}$.

Endomorphism ring

An isogeny $\varphi : E \to E$ is an **endomorphism**.
We denote $\mathrm{End}(E) := \{\varphi : E \to E\} \cup \{0\}$.

- $(\mathrm{End}(E), +, \circ)$ is the **endomorphism ring** of E, where for every $P \in E$:

$$(\varphi+\psi)(P) = \varphi(P) + \psi(P) \qquad \text{and} \qquad (\varphi\circ\psi)(P) = \varphi(\psi(P)).$$

- $\mathbb{Z} \hookrightarrow \mathrm{End}(E)$ as **subring**. For every $n \in \mathbb{Z}$, we have the endomorphism

$$\begin{aligned} [n] : E &\to E \\ P &\mapsto [n]P := \underbrace{P + \cdots + P}_{n \text{ times}}. \end{aligned}$$

- $(\mathrm{End}(E), +)$ is a lattice of dimension
  - **2** i.e. $\mathrm{End}(E) \simeq \mathbb{Z} \oplus \alpha\mathbb{Z}$, } E is **ordinary**.
    or
  - **4** i.e. $\mathrm{End}(E) \simeq \mathbb{Z} \oplus \alpha\mathbb{Z} \oplus \beta\mathbb{Z} \oplus \gamma\mathbb{Z}$. } E is **supersingular**.

Orientations

Orientations

$\alpha \in \mathrm{End}(E) \backslash \mathbb{Z}$ can be viewed as a **quadratic integer** with minimal polynomial

$$P_\alpha(X) := X^2 + (\alpha + \hat{\alpha})X + \alpha \circ \hat{\alpha} \in \mathbb{Z}[X].$$

Orientations

$\alpha \in \mathrm{End}(E) \backslash \mathbb{Z}$ can be viewed as a **quadratic integer** with minimal polynomial

$$P_\alpha(X) := X^2 + (\alpha + \hat{\alpha})X + \alpha \circ \hat{\alpha} \in \mathbb{Z}[X].$$

For any root ω of $P_\alpha(X)$, there is an **embedding**

$$\mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E), \omega \mapsto \alpha.$$

Orientations

$\alpha \in \mathrm{End}(E) \backslash \mathbb{Z}$ can be viewed as a **quadratic integer** with minimal polynomial

$$P_\alpha(X) := X^2 + (\alpha + \hat{\alpha})X + \alpha \circ \hat{\alpha} \in \mathbb{Z}[X].$$

For any root ω of $P_\alpha(X)$, there is an **embedding**

$$\mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E), \omega \mapsto \alpha.$$

Orientation [CK20; Onu21]

Let ω be a quadratic integer. An embedding

$$\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$$

is called an $\mathbb{Z}[\omega]$-**orientation** on E. We say that (E, ι) is an $\mathbb{Z}[\omega]$-oriented curve.

Orientations

$\alpha \in \mathrm{End}(E) \backslash \mathbb{Z}$ can be viewed as a **quadratic integer** with minimal polynomial

$$P_\alpha(X) := X^2 + (\alpha + \hat{\alpha})X + \alpha \circ \hat{\alpha} \in \mathbb{Z}[X].$$

For any root ω of $P_\alpha(X)$, there is an **embedding**

$$\mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E), \omega \mapsto \alpha.$$

Orientation [CK20; Onu21]

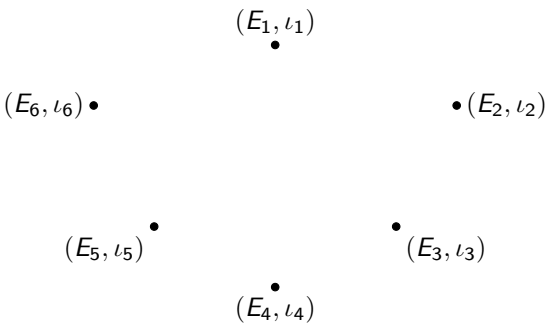
Let ω be a quadratic integer. An embedding

$$\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$$

is called an $\mathbb{Z}[\omega]$-**orientation** on E. We say that (E, ι) is an $\mathbb{Z}[\omega]$-oriented curve.
We say that ι is a **primitive orientation**,
if for any quadratic ring $\mathfrak{O} \supseteq \mathbb{Z}[\omega]$ such that $\mathfrak{O} \hookrightarrow \mathrm{End}(E)$, we have that $\mathfrak{O} = \mathbb{Z}[\omega]$.

Class group action

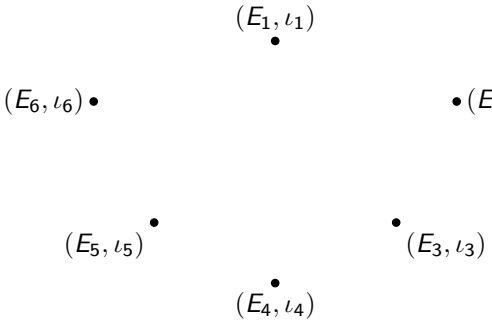
Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.

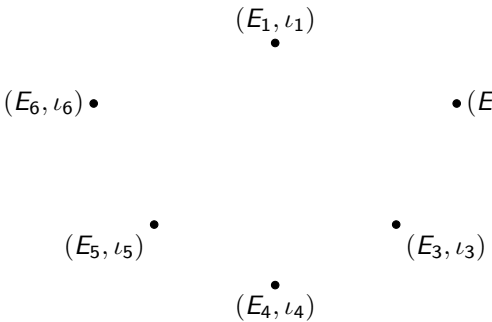


Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



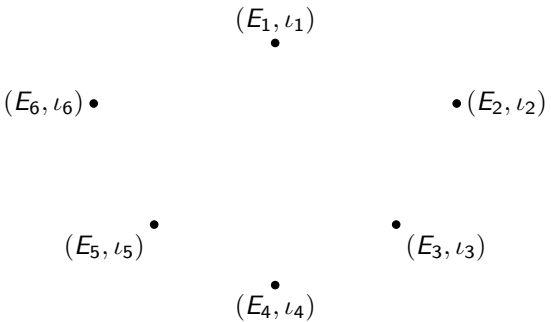
Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

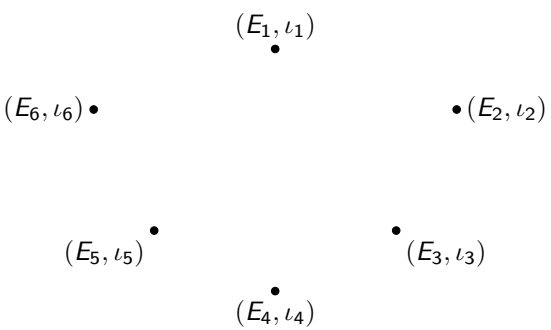
Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

$\forall \alpha \in \mathfrak{a}$, $\iota(\alpha) \in \mathrm{End}(E)$.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

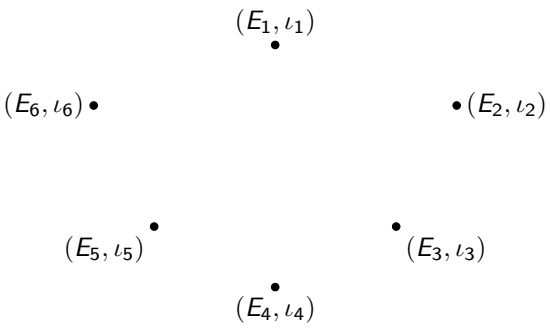
Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

$$\forall \alpha \in \mathfrak{a},\ \ker \iota(\alpha) \leqslant E.$$

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

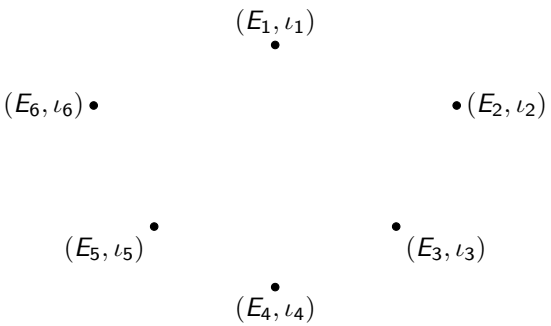
Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

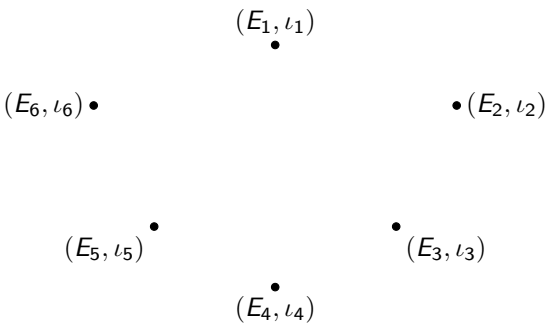
(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

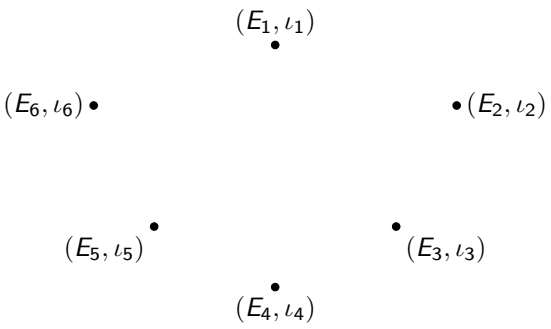
$$E[\mathfrak{a}] := \bigcap_{\alpha \in \mathfrak{a}} \ker(\iota(\alpha)) \leqslant E.$$

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

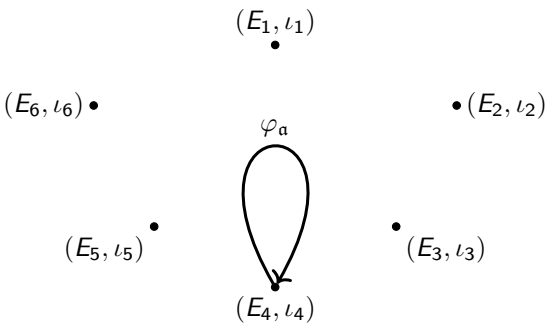
The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Proposition [CK20; Onu21]

The class group $\mathcal{C}l(\mathbb{Z}[\omega])$ acts freely on $SS_{\mathbb{Z}[\omega]}$ and has at most two orbits.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

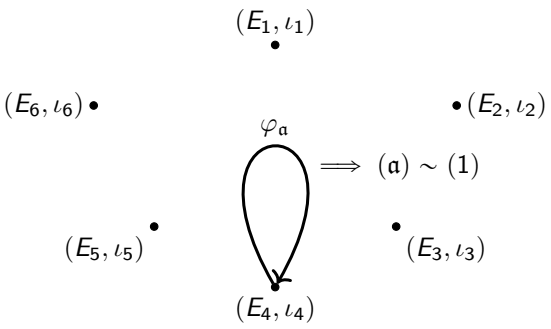
The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Proposition [CK20; Onu21]

The class group $\mathcal{C}l(\mathbb{Z}[\omega])$ acts **freely** on $SS_{\mathbb{Z}[\omega]}$ and has at most two orbits.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

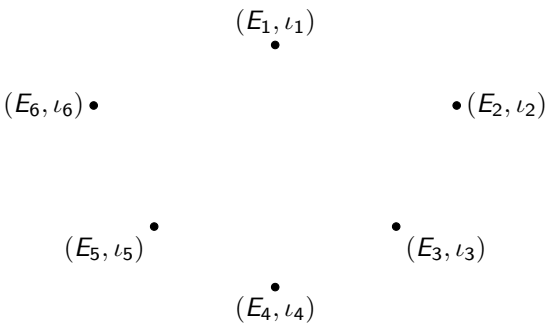
The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Proposition [CK20; Onu21]

The class group $\mathcal{C}l(\mathbb{Z}[\omega])$ acts **freely** on $SS_{\mathbb{Z}[\omega]}$ and has at most two orbits.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

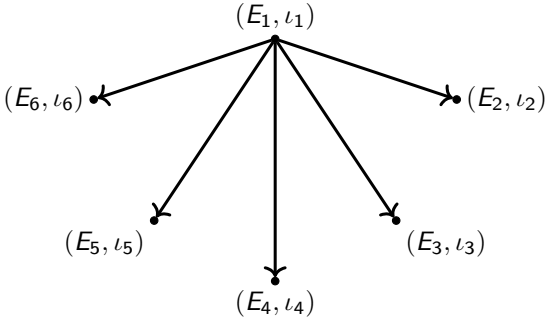
The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Proposition [CK20; Onu21]

The class group $\mathcal{C}l(\mathbb{Z}[\omega])$ acts **freely** on $SS_{\mathbb{Z}[\omega]}$ and has **at most two orbits**.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

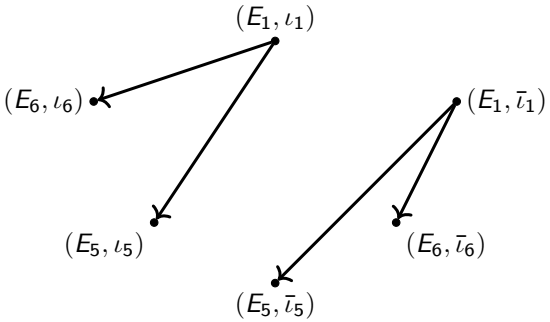
The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Proposition [CK20; Onu21]

The class group $\mathcal{C}l(\mathbb{Z}[\omega])$ acts **freely** on $SS_{\mathbb{Z}[\omega]}$ and has **at most two orbits**.

Class group action

Let $SS_{\mathbb{Z}[\omega]}$ be the set of primitively $\mathbb{Z}[\omega]$-oriented elliptic curves, up to isomorphism.



Example when $|SS_{\mathbb{Z}[\omega]}| = 6$

Let $(E, \iota) \in SS_{\mathbb{Z}[\omega]}$ and $\mathfrak{a}$ be an $\mathbb{Z}[\omega]$-ideal

(recall $\iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E)$)

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

Using Vélu's formulas [Vé71],
we get an isogeny $\varphi_{\mathfrak{a}}$ with kernel $E[\mathfrak{a}]$.

The isogeny $\varphi_{\mathfrak{a}}$ is **horizontal**,
i.e. it sends (E, ι) to another curve in $SS_{\mathbb{Z}[\omega]}$.

Proposition [CK20; Onu21]

The class group $\mathcal{C}l(\mathbb{Z}[\omega])$ acts **freely** on $SS_{\mathbb{Z}[\omega]}$ and has **at most two orbits**.

Orientation over $\mathbb{F}_p$

Proposition [DG16]

If $p > 3$ then

$$E \text{ is defined over } \mathbb{F}_p \iff \iota : \mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E), \quad \sqrt{-p} \mapsto \pi,$$

where π is the Frobenius map

$$\pi : E \to E^{(p)}, \qquad (x, y) \mapsto (x^p, y^p).$$

Orientation over $\mathbb{F}_p$

Proposition [DG16]

If $p > 3$ then

$$E \text{ is defined over } \mathbb{F}_p \iff \iota : \mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E), \quad \sqrt{-p} \mapsto \pi,$$

where π is the Frobenius map

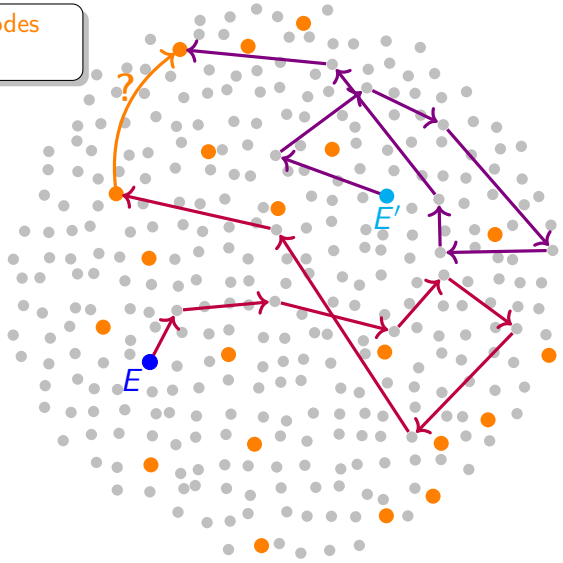
$$\pi : E \to E^{(p)}, \qquad (x, y) \mapsto (x^p, y^p).$$

Corollary from [JMV09] (under GRH)

One can efficiently perform random walks over $SS_{\mathbb{Z}[\omega]}$ following a distribution close to uniform.

The Delfs-Galbraith algorithm (Step 2)

There are around $\sqrt{p}$ nodes defined over $\mathbb{F}_p$

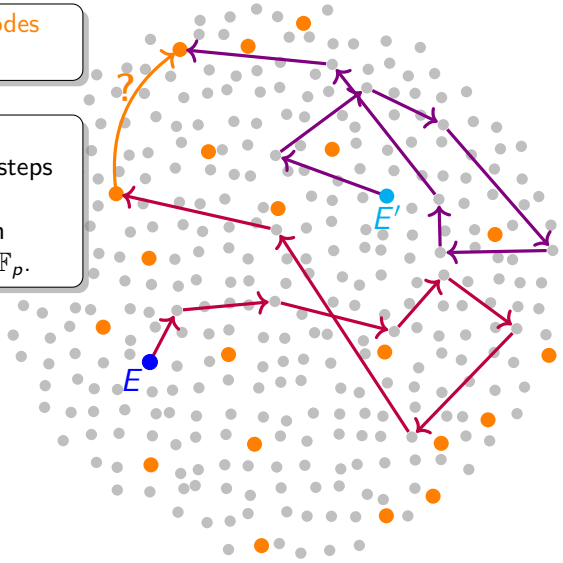


A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm (Step 2)

There are around $\sqrt{p}$ nodes defined over $\mathbb{F}_p$

Expander graph:
Taking $O(\log p)$ random steps *almost* gives a uniform distribution on curves defined over $\mathbb{F}_p$.

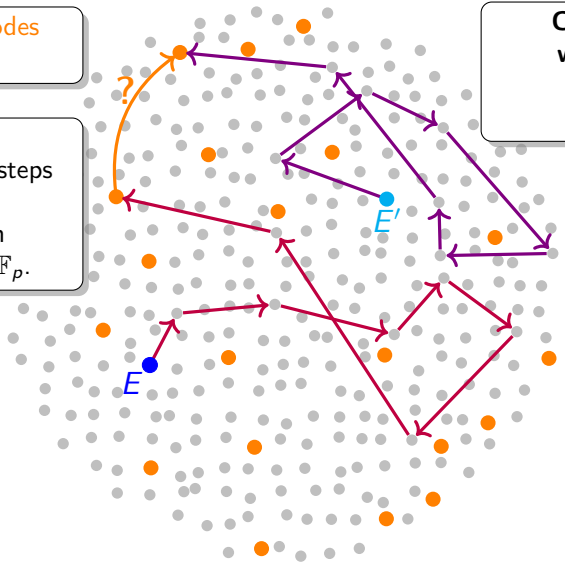


A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm (Step 2)

There are around $\sqrt{p}$ nodes defined over $\mathbb{F}_p$

Expander graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution
on curves defined over $\mathbb{F}_p$.



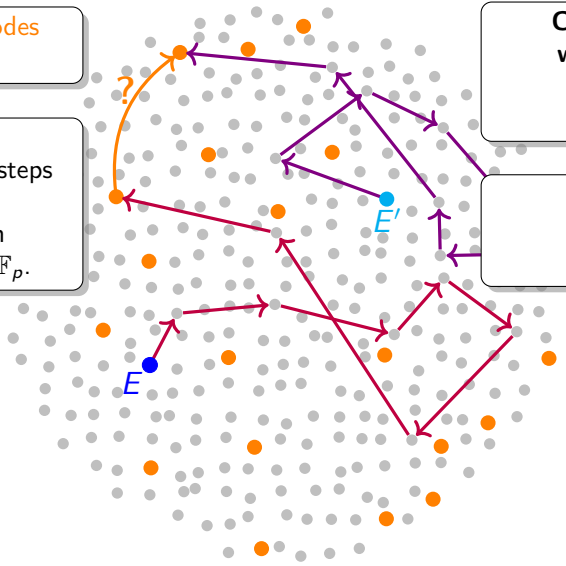
Complexity of Step 2 with a naive MITM:
Time: $\tilde{O}(p^{1/4})$
Memory: $\tilde{O}(p^{1/4})$

A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm (Step 2)

There are around $\sqrt{p}$ nodes defined over $\mathbb{F}_p$

Expander graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution
on curves defined over $\mathbb{F}_p$.



Complexity of Step 2 with a naive MITM:
Time: $\tilde{O}(p^{1/4})$
Memory: $\tilde{O}(p^{1/4})$

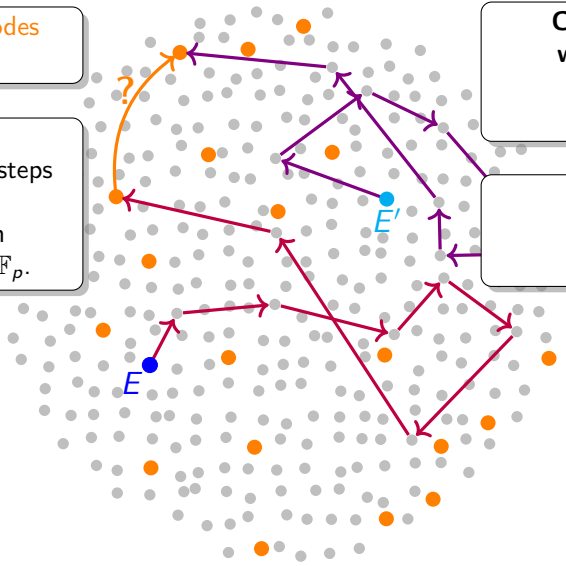
Total complexity:
Time: $\tilde{O}(\sqrt{p})$
Memory: $\tilde{O}(p^{1/4})$

A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The Delfs-Galbraith algorithm (Step 2)

There are around $\sqrt{p}$ nodes defined over $\mathbb{F}_p$

Expander graph:
Taking $O(\log p)$ random steps
almost gives
a uniform distribution
on curves defined over $\mathbb{F}_p$.

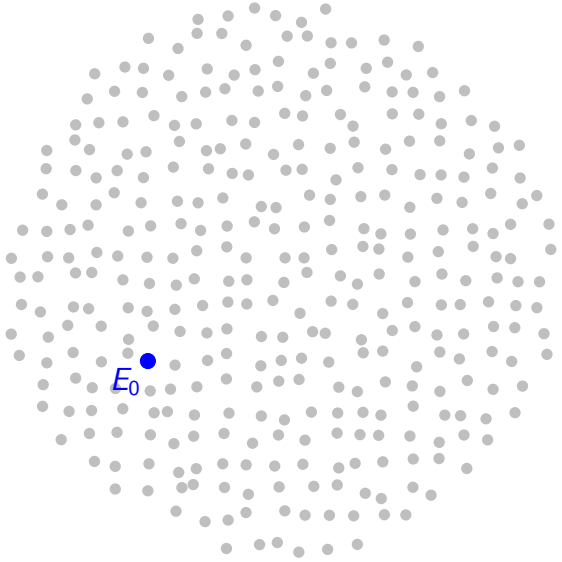


Complexity of Step 2 with a naive MITM:
Time: $\tilde{O}(p^{1/4})$
Memory: $\tilde{O}(p^{1/4})$

Total complexity:
Time: $\tilde{O}(\sqrt{p})$
Memory: $\tilde{O}(p^{1/4})$

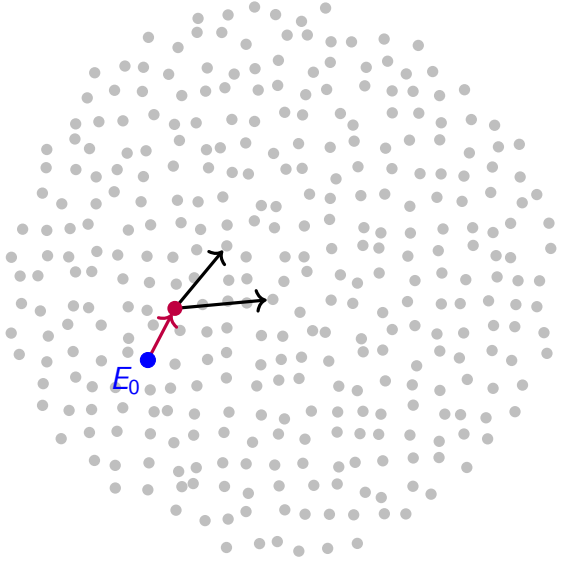
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

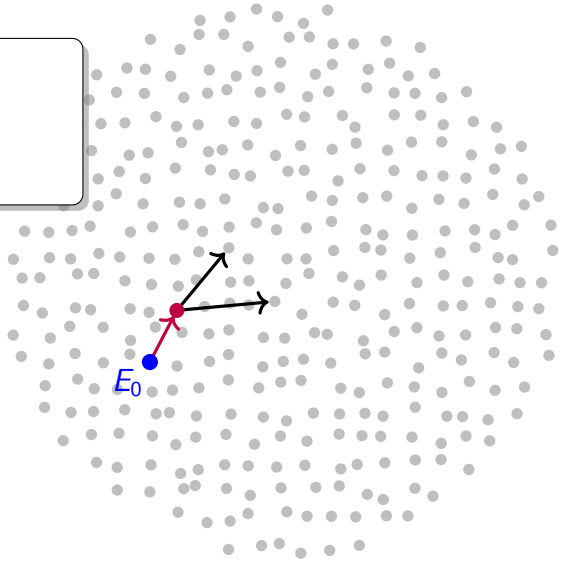


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. Sample a random root

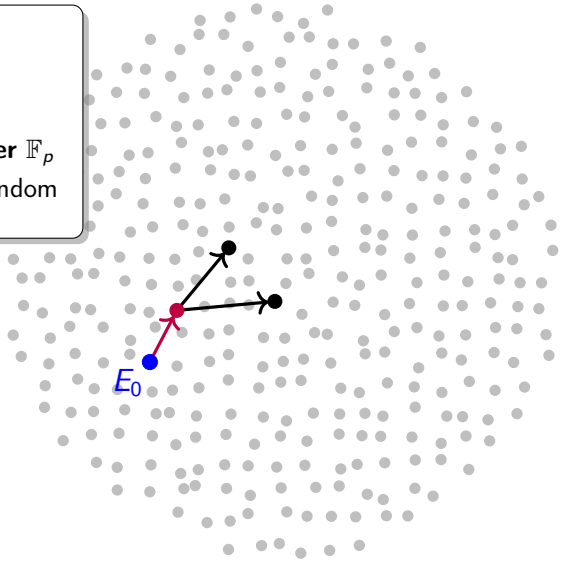


A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root



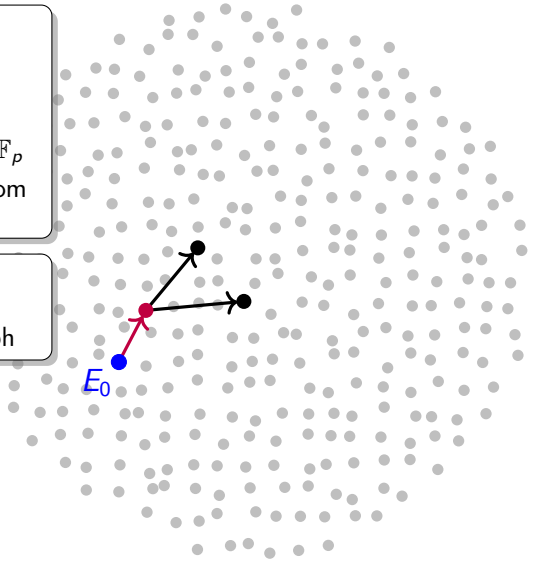
A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

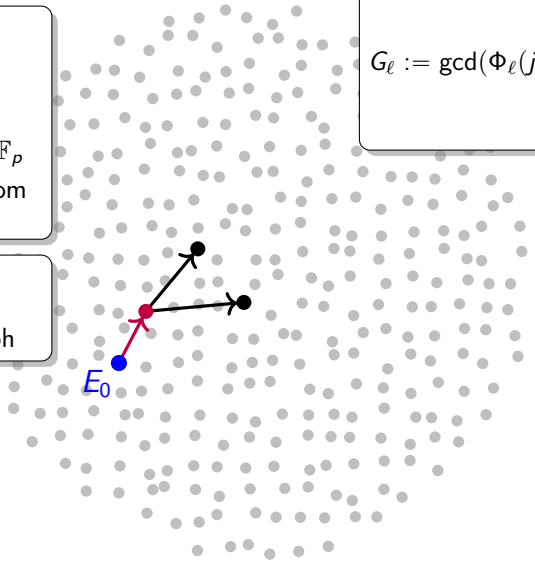
Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph

Lemma:

$G_\ell := \gcd(\Phi_\ell(j(E), Y), \Phi_\ell(j(E), Y)^p)$



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

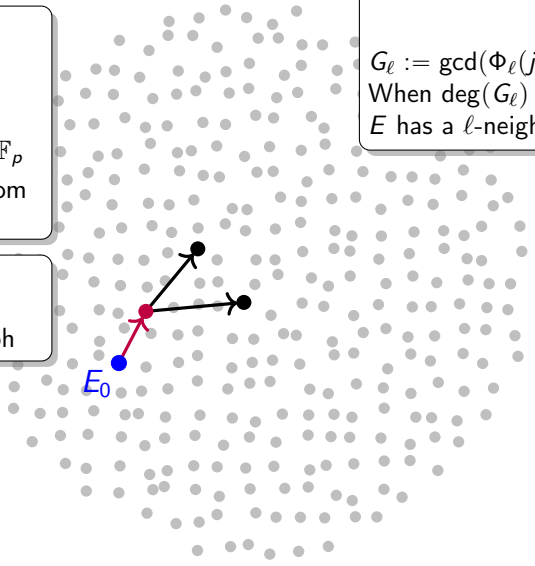
Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph

Lemma:

$G_\ell := \gcd(\Phi_\ell(j(E), Y), \Phi_\ell(j(E), Y)^p)$
When $\deg(G_\ell) = 1$,
E has a ℓ-neighbor defined over $\mathbb{F}_p$.



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

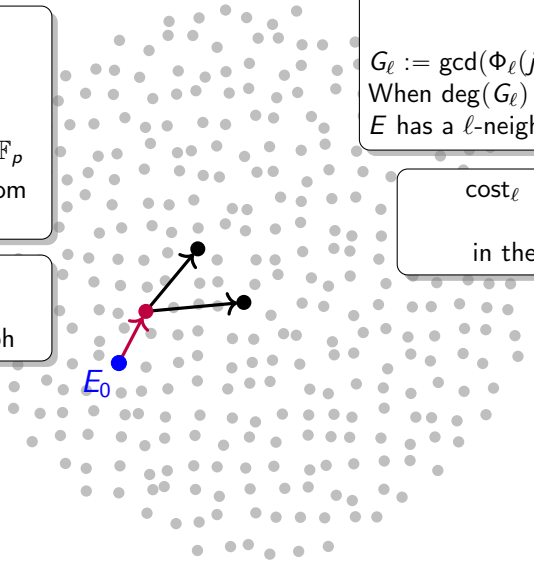
1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph

Lemma:

$G_\ell := \gcd(\Phi_\ell(j(E), Y), \Phi_\ell(j(E), Y)^p)$
When $\deg(G_\ell) = 1$,
E has a ℓ-neighbor defined over $\mathbb{F}_p$.

cost_ℓ := cost per node **inspected** in the ℓ-isogeny graph



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

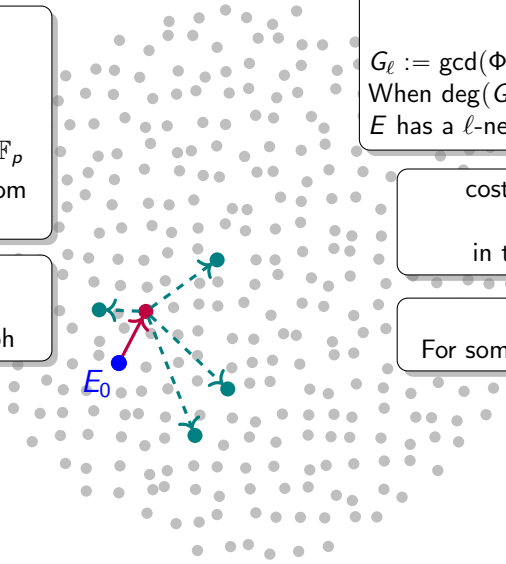
cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph

Lemma:

$G_\ell := \gcd(\Phi_\ell(j(E), Y), \Phi_\ell(j(E), Y)^p)$
When $\deg(G_\ell) = 1$,
E has a ℓ-neighbor defined over $\mathbb{F}_p$.

cost_ℓ := cost per node **inspected** in the ℓ-isogeny graph

Observation:
For some $\ell > 2$, $\text{cost}_2 > \text{cost}_\ell$.



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

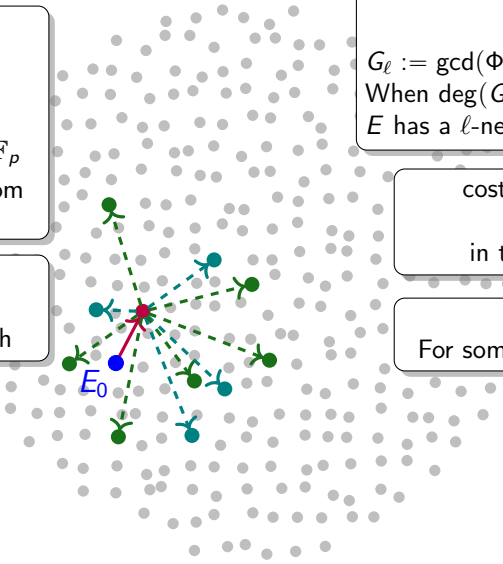
cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph

Lemma:

$G_\ell := \gcd(\Phi_\ell(j(E), Y), \Phi_\ell(j(E), Y)^p)$
When $\deg(G_\ell) = 1$,
E has a ℓ-neighbor defined over $\mathbb{F}_p$.

cost_ℓ := cost per node **inspected** in the ℓ-isogeny graph

Observation:
For some $\ell > 2$, $\text{cost}_2 > \text{cost}_\ell$.



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

The SuperSolver algorithm [CCS22]

Precompute $\Phi_2(X, Y)$

1. Compute roots of $\Phi_2(j(E), Y)$
2. **Check if a root is over $\mathbb{F}_p$**
3. Otherwise sample a random root

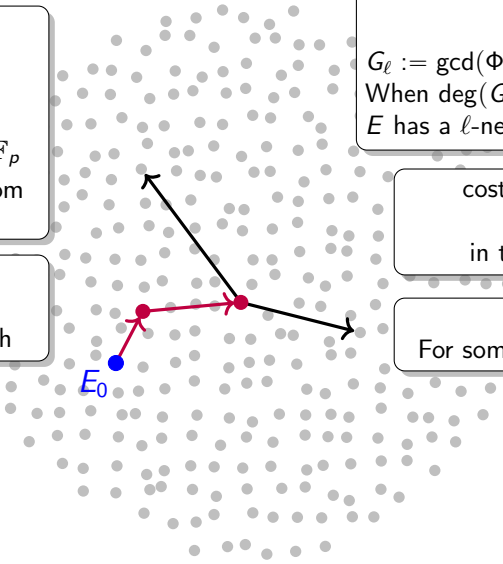
cost_2 := cost per node **revealed** of taking a step in the 2-isogeny graph

Lemma:

$G_\ell := \gcd(\Phi_\ell(j(E), Y), \Phi_\ell(j(E), Y)^p)$
When $\deg(G_\ell) = 1$,
E has a ℓ-neighbor defined over $\mathbb{F}_p$.

cost_ℓ := cost per node **inspected** in the ℓ-isogeny graph

Observation:
For some $\ell > 2$, $\text{cost}_2 > \text{cost}_\ell$.



A (non-contractual) 2-isogeny graph over $\bar{\mathbb{F}}_p$.

SuperSolver's results [CCS22]

Core idea

Perform **fast subfield root detection** for all ℓ in the subset of $L := \{\ell \text{ prime such that } \text{cost}_2 > \text{cost}_\ell\}$ that minimizes the total cost of each step

$$\text{cost} := \frac{\#\text{ of multiplications over } \mathbb{F}_p \text{ at each step}}{\#\text{ of nodes inspected}}.$$

SuperSolver's results [CCS22]

Core idea

Perform **fast subfield root detection** for all ℓ in the subset of
$L := \{\ell \text{ prime such that } \text{cost}_2 > \text{cost}_\ell\}$ that minimizes the total cost of each step

$$\text{cost} := \frac{\#\text{ of multiplications over } \mathbb{F}_p \text{ at each step}}{\#\text{ of nodes inspected}}.$$

Result

The SuperSolver algorithm has the **same asymptotic complexity** as the Delfs-Galbraith algorithm but is more than 17 times **more efficient in practice** (for large p).

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

SuperSolver Detection

For some ℓ such that $\mathrm{cost}_\ell < \mathrm{cost}_2$.
Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\ell\text{-isogenous}]{} E' \ ?$$

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

SuperSolver Detection

For some ℓ such that $\mathrm{cost}_\ell < \mathrm{cost}_2$.
Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\ell\text{-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

SuperSolver Detection

For some ℓ such that $\mathrm{cost}_\ell < \mathrm{cost}_2$.
Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\ell\text{-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

More general configuration:

$$E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E''$$

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

SuperSolver Detection

For some ℓ such that $\mathrm{cost}_\ell < \mathrm{cost}_2$.
Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\ell\text{-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

More general configuration:

$$E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E''$$

Interesting detection:

- Is E' defined over $\mathbb{F}_p$? $\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

SuperSolver Detection

For some ℓ such that $\mathrm{cost}_\ell < \mathrm{cost}_2$.
Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\ell\text{-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

More general configuration:

$$E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E''$$

Interesting detection:

- Is E' defined over $\mathbb{F}_p$? $\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$
- Is E'' defined over $\mathbb{F}_p$? $\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E'')$

Starting point of this **ongoing** work

From now on, this is joint work with Maria Corte Real Santos, David Jao, Joseph Macula, Michael Meyer, Travis Morrison, and Eli Orvis.

Delfs-Galbraith Detection

Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\text{2-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

SuperSolver Detection

For some ℓ such that $\mathrm{cost}_\ell < \mathrm{cost}_2$.
Is there an E' defined over $\mathbb{F}_p$ such that

$$E \xrightarrow[\ell\text{-isogenous}]{} E' \ ?$$

$\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$.

More general configuration:

$$E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E''$$

Interesting detection:

- Is E' defined over $\mathbb{F}_p$? $\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E')$
- Is E'' defined over $\mathbb{F}_p$? $\implies$ We detect $\mathbb{Z}[\sqrt{-p}] \hookrightarrow \mathrm{End}(E'')$
- Is $E'' \simeq E'^{(p)}$? $\implies$ We detect $\mathbb{Z}[\sqrt{-\ell_2 p}] \hookrightarrow \mathrm{End}(E')$

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

Remarks: it also gives an oriented curve by $\mathbb{Z}[\sqrt{-p}]$ (Delf-Galbraith) or $\mathbb{Z}[\sqrt{-\ell_2 p}]$ ("New")

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

Remarks: it also gives an oriented curve by $\mathbb{Z}[\sqrt{-p}]$ (Delf-Galbraith) or $\mathbb{Z}[\sqrt{-\ell_2 p}]$ ("New")
The latter corresponds to $E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E'^{(p)}$

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

Remarks: it also gives an oriented curve by $\mathbb{Z}[\sqrt{-p}]$ (Delf-Galbraith) or $\mathbb{Z}[\sqrt{-\ell_2 p}]$ ("New")
The latter corresponds to $E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E'^{(p)}$,i.e. $\exists \varphi : E' \to E'^{(p)}$ s.t. $\deg \varphi = \ell_2$.

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

Remarks: it also gives an oriented curve by $\mathbb{Z}[\sqrt{-p}]$ (Delf-Galbraith) or $\mathbb{Z}[\sqrt{-\ell_2 p}]$ ("New")

The latter corresponds to $E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E'^{(p)}$,i.e. $\exists \varphi : E' \to E'^{(p)}$ s.t. $\deg \varphi = \ell_2$.

Thus, the explicit orientation is $\mathbb{Z}[\sqrt{-p\ell_2}] \hookrightarrow \mathrm{End}(E'), \sqrt{-p\ell_2} \mapsto \varphi \circ \pi$.

New detection tool: OrientationFinder

Theorem

For some primes ℓ_1, ℓ_2, there exists a precomputable polynomial $R_{\ell_1,\ell_2}(X, Y) \in \mathbb{Z}[X, Y]$ such that

$$R_{\ell_1,\ell_2}(j(E), Y) \text{ has an } \mathbb{F}_p\text{-rational root} \iff \iota : \mathbb{Z}[\omega] \hookrightarrow \mathrm{End}(E) \text{ is a primitive orientation,}$$

where $\omega \in \{\sqrt{-p}, \frac{1+\sqrt{-p}}{2}, \sqrt{-\ell_2 p}, \frac{1+\sqrt{-\ell_2 p}}{2}, \ell_1\sqrt{-p}, \ell_1\frac{1+\sqrt{-p}}{2}, \ell_1\sqrt{-\ell_2 p}, \ell_1\frac{1+\sqrt{-\ell_2 p}}{2}\}$.

Remarks: it also gives an oriented curve by $\mathbb{Z}[\sqrt{-p}]$ (Delf-Galbraith) or $\mathbb{Z}[\sqrt{-\ell_2 p}]$ ("New")

The latter corresponds to $E \xrightarrow[\ell_1\text{-isogenous}]{} E' \xrightarrow[\ell_2\text{-isogenous}]{} E'^{(p)}$,i.e. $\exists \varphi : E' \to E'^{(p)}$ s.t. $\deg \varphi = \ell_2$.

Thus, the explicit orientation is $\mathbb{Z}[\sqrt{-p\ell_2}] \hookrightarrow \mathrm{End}(E'), \sqrt{-p\ell_2} \mapsto \varphi \circ \pi$.

These orientations were already studied in [CS22], where the question of such Generalized Delfs–Galbraith algorithms is raised.

What are the advantages?

Pros:

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1 + 1)(\ell_2 + 1)$ versus $\ell_1 + 1$),

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

- Detection is **more costly** than in SuperSolver,

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

- Detection is **more costly** than in SuperSolver,
- As in SuperSolver, there is **no asymptotic complexity improvement**.

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

- Detection is **more costly** than in SuperSolver,
- As in SuperSolver, there is **no asymptotic complexity improvement**.

Conclusion:

- To obtain concrete improvements, we need to compute the optimal set of primes to consider.

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

- Detection is **more costly** than in SuperSolver,
- As in SuperSolver, there is **no asymptotic complexity improvement**.

Conclusion:

- To obtain concrete improvements, we need to compute the optimal set of primes to consider.

New problems

- The previous metric is no longer suitable.

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

- Detection is **more costly** than in SuperSolver,
- As in SuperSolver, there is **no asymptotic complexity improvement**.

Conclusion:

- To obtain concrete improvements, we need to compute the optimal set of primes to consider.

New problems

- The previous metric is no longer suitable.

In SuperSolver, we **minimize** cost $:= \frac{\#\text{ of multiplications over } \mathbb{F}_p \text{ at each step}}{\#\text{ of nodes inspected}}$, so

What are the advantages?

Pros:

- The SuperSolver method is **included** in OrientedSolver,
- For each fast subfield root detection, **more curves are inspected** than in SuperSolver ($(\ell_1+1)(\ell_2+1)$ versus ℓ_1+1),
- We consider **multiple special sets** of curves, all of cardinality around $\sqrt{p}$, instead of only one in SuperSolver.

Cons:

- Detection is **more costly** than in SuperSolver,
- As in SuperSolver, there is **no asymptotic complexity improvement**.

Conclusion:

- To obtain concrete improvements, we need to compute the optimal set of primes to consider.

New problems

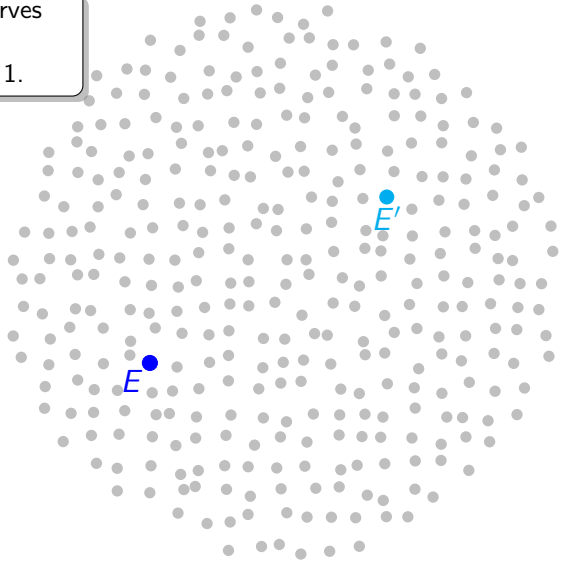
- The previous metric is no longer suitable.

In SuperSolver, we **minimize** cost $:= \frac{\#\text{ of multiplications over } \mathbb{F}_p \text{ at each step}}{\#\text{ of nodes inspected}}$, so

$$\#\text{ of multiplications over } \mathbb{F}_p \text{ in Step 1 } = \text{cost} \cdot \frac{\#\text{ of curves over } \mathbb{F}_{p^2}}{\#\text{ of curves over } \mathbb{F}_p} \text{ is } \textbf{minimal}.$$

Does it really work?

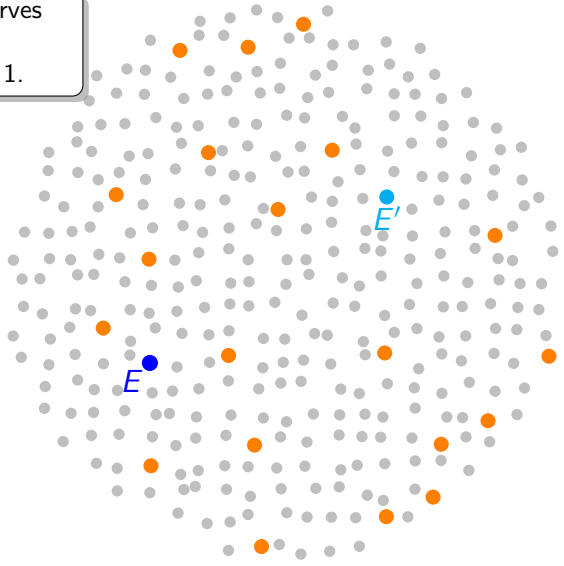
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

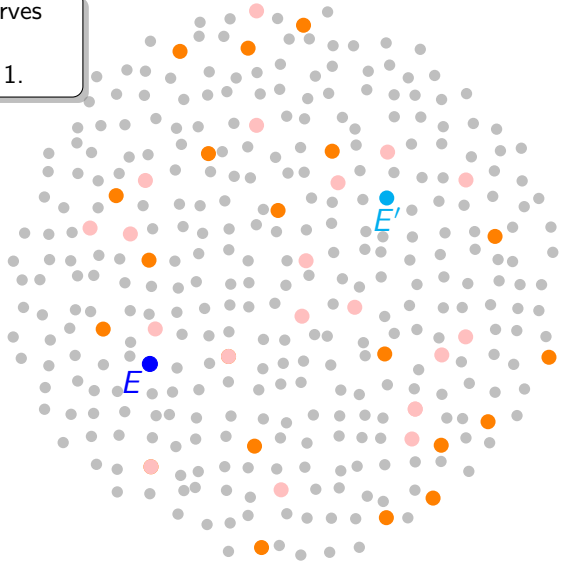
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

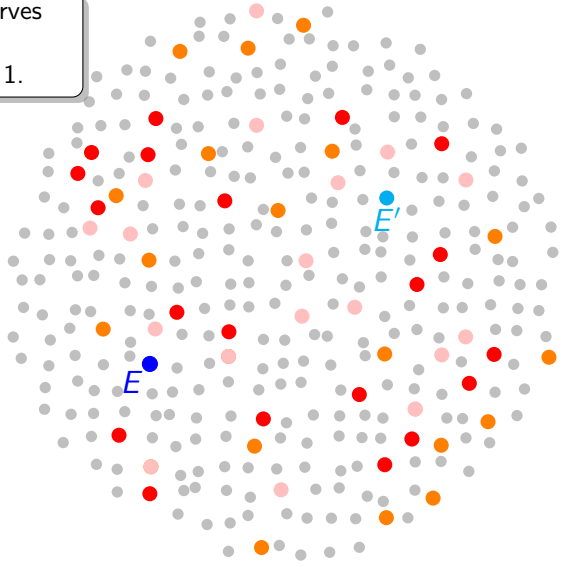
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

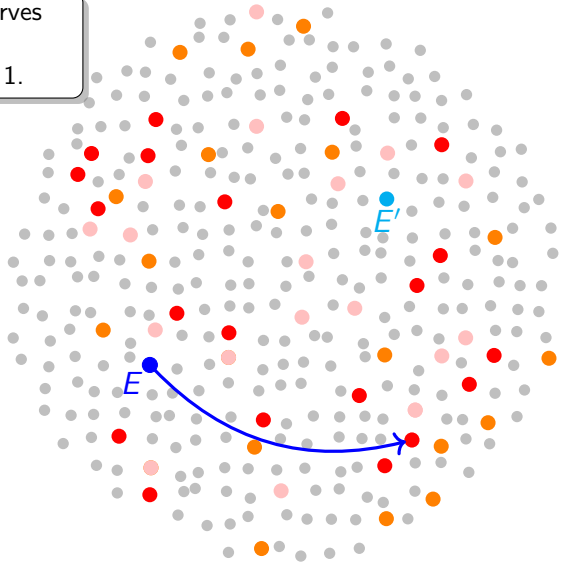
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

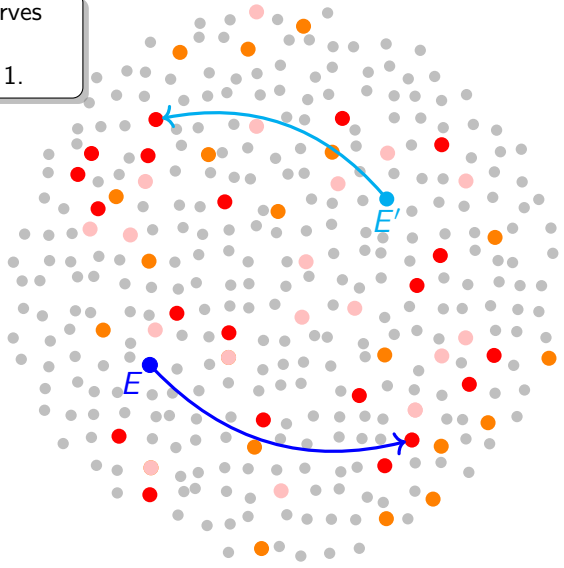
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

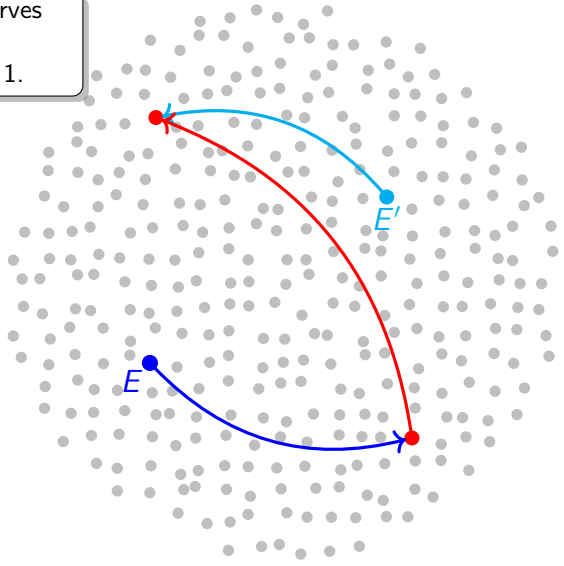
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

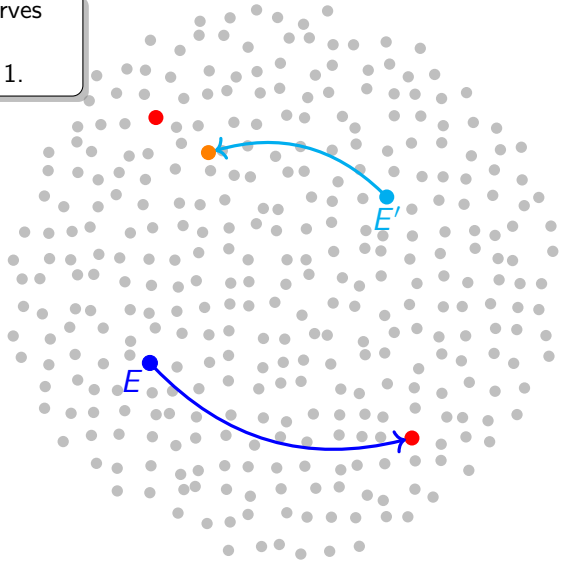
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

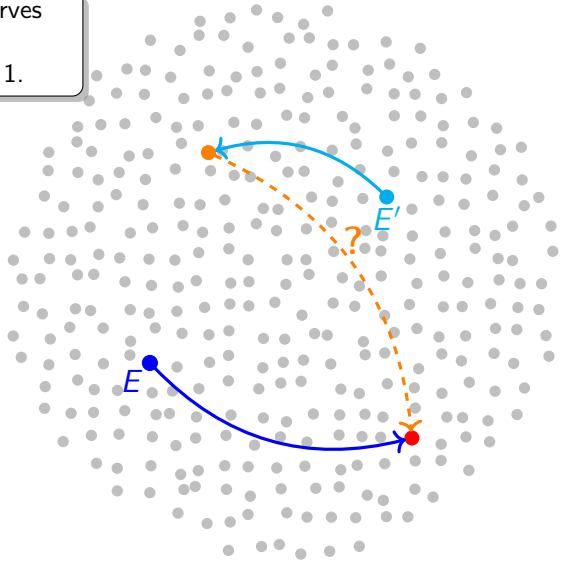
There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

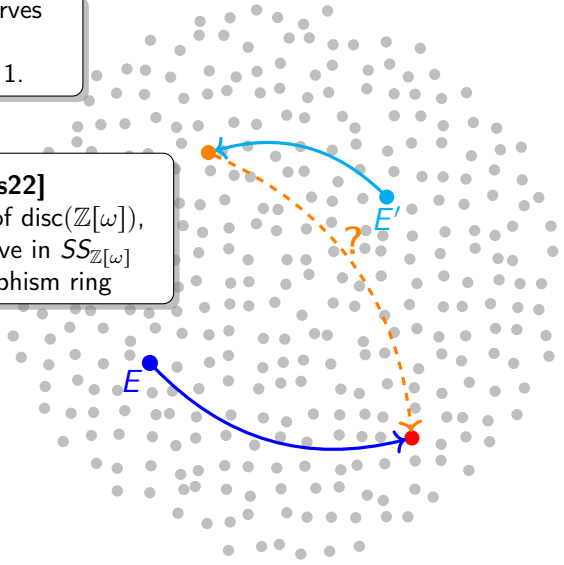


A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring

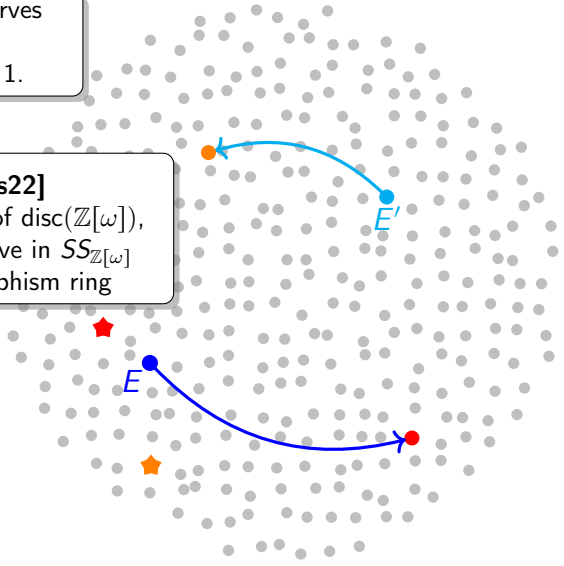


A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$.

Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring

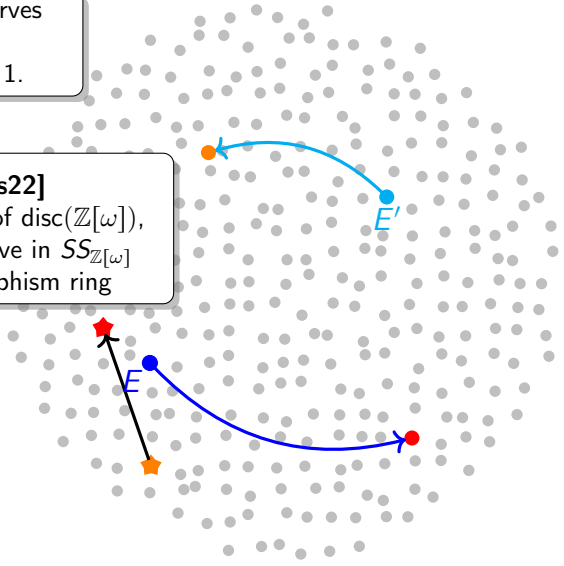


A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$. Star nodes are curves with known endomorphism ring.

Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring

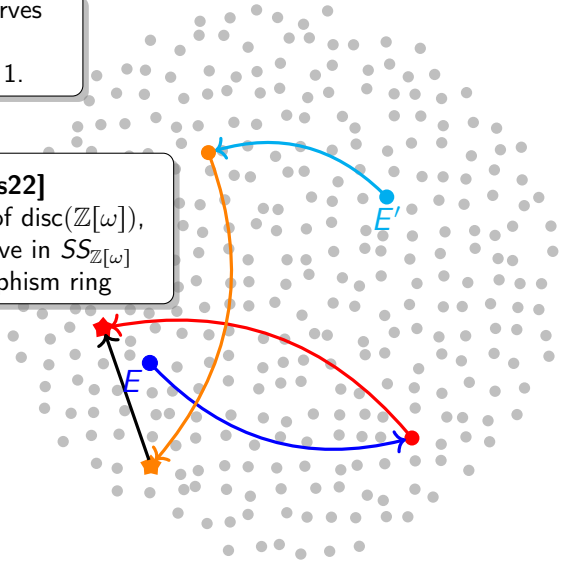


A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$. Star nodes are curves with known endomorphism ring.

Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring



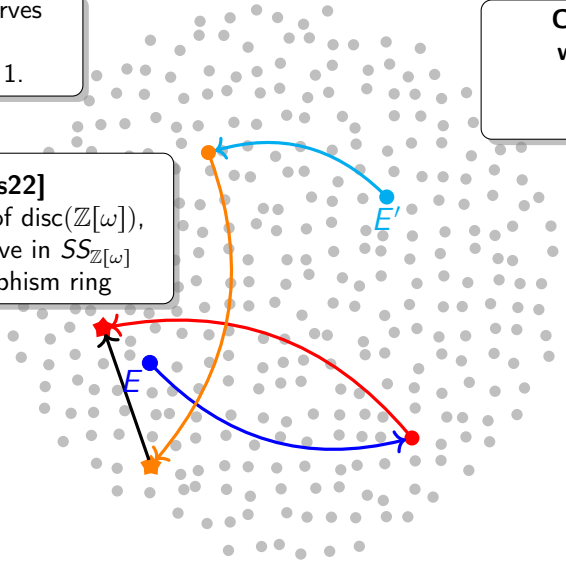
A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$. Star nodes are curves with known endomorphism ring.

Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring

Complexity of Step 2 with a naive MITM:
Time: $\tilde{O}(p^{1/4})$
Memory: $\tilde{O}(p^{1/4})$



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$. Star nodes are curves with known endomorphism ring.

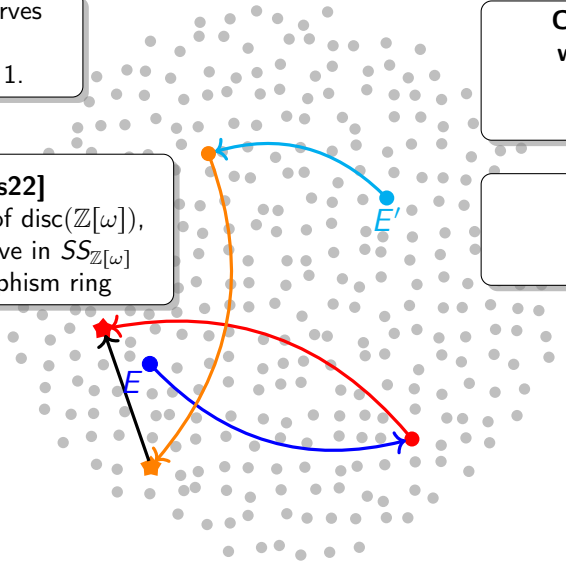
Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring

Complexity of Step 2 with a naive MITM:
Time: $\tilde{O}(p^{1/4})$
Memory: $\tilde{O}(p^{1/4})$

Total complexity:
Time: $\tilde{O}(\sqrt{p})$
Memory: $\tilde{O}(p^{1/4})$



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$. Star nodes are curves with known endomorphism ring.

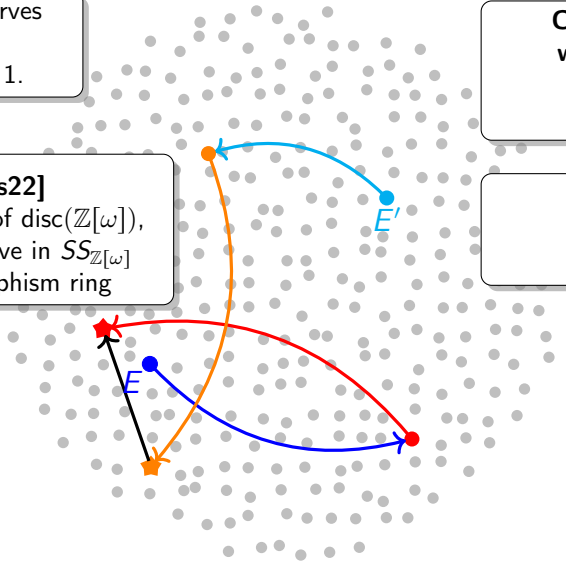
Does it really work?

There are around $\sqrt{p}$ curves oriented by $\mathbb{Z}[\sqrt{\ell_2 p}]$, for ℓ_2 a small prime or 1.

Theorem [Wes22]
Given the factorisation of $\mathrm{disc}(\mathbb{Z}[\omega])$, one can compute a curve in $SS_{\mathbb{Z}[\omega]}$ with known endomorphism ring

Complexity of Step 2 with a naive MITM:
Time: $\tilde{O}(p^{1/4})$
Memory: $\tilde{O}(p^{1/4})$

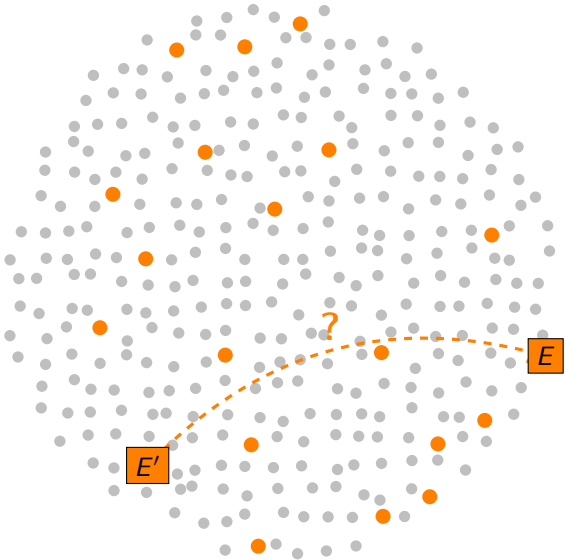
Total complexity:
Time: $\tilde{O}(\sqrt{p})$
Memory: $\tilde{O}(p^{1/4})$



A (non-contractual) ℓ-isogeny graph over $\bar{\mathbb{F}}_p$. Star nodes are curves with known endomorphism ring.

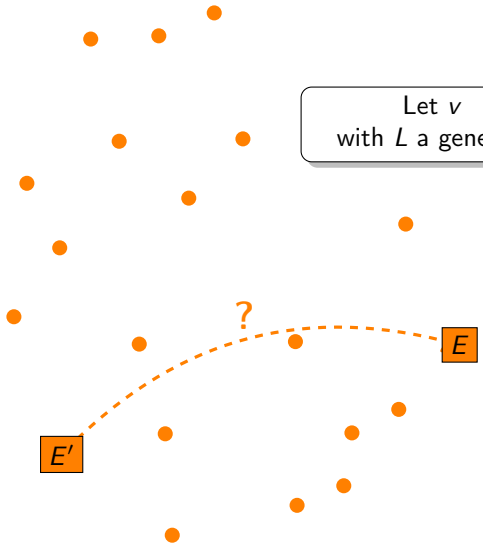
What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].



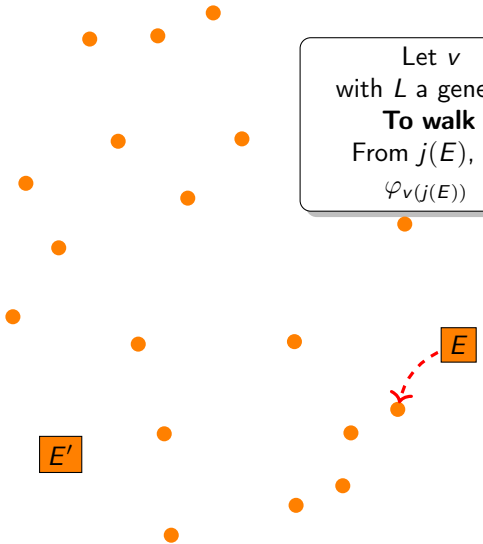
What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].



What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].



Let $v : SS_{\mathbb{Z}[\omega]} \to L$,
with L a generating set of $Cl(\mathbb{Z}[\omega])$.
To walk deterministically:
From $j(E)$, go to $j(E')$ such that
$\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

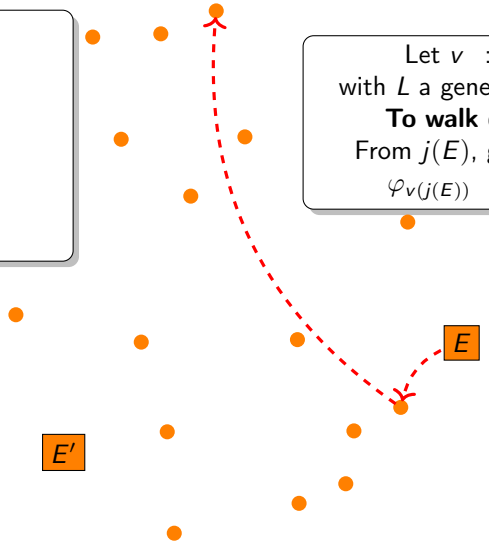
Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)

Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $\mathcal{C}l(\mathbb{Z}[\omega])$.

To walk deterministically:

From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

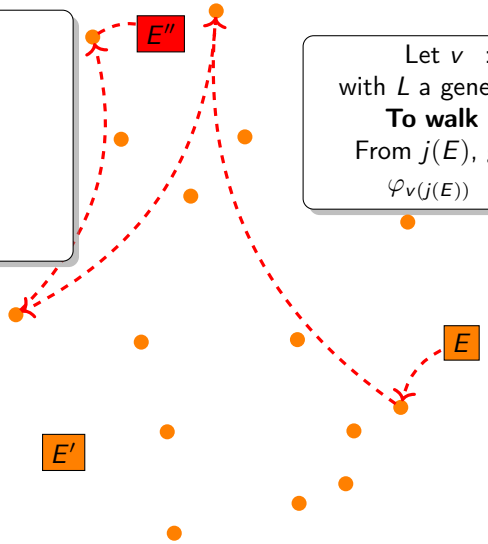


What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.

To walk deterministically:

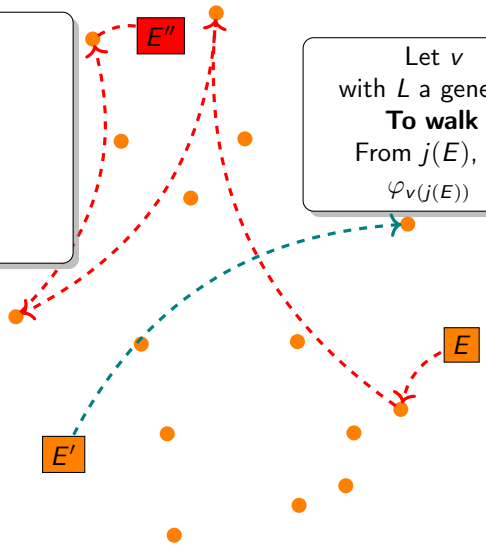
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $\mathcal{C}l(\mathbb{Z}[\omega])$.

To walk deterministically:

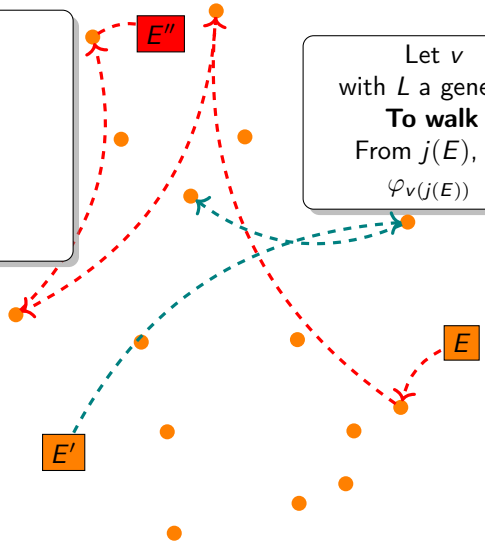
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.

To walk deterministically:

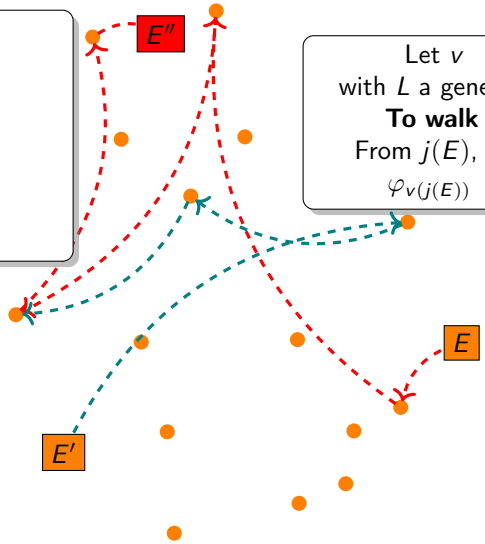
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $\mathcal{C}l(\mathbb{Z}[\omega])$.

To walk deterministically:

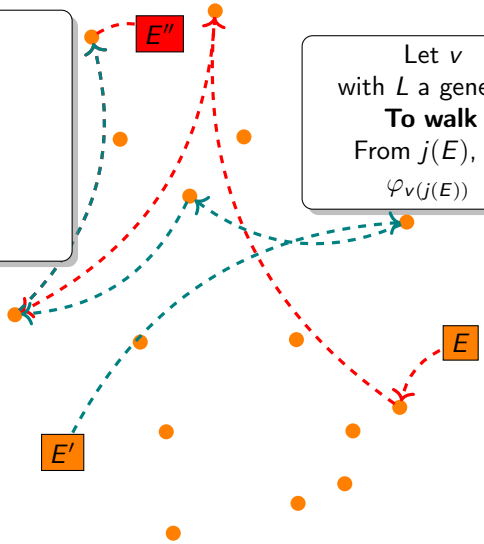
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.

To walk deterministically:

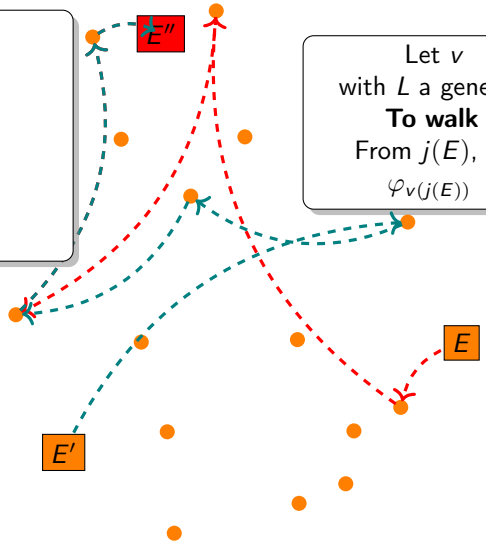
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.

To walk deterministically:

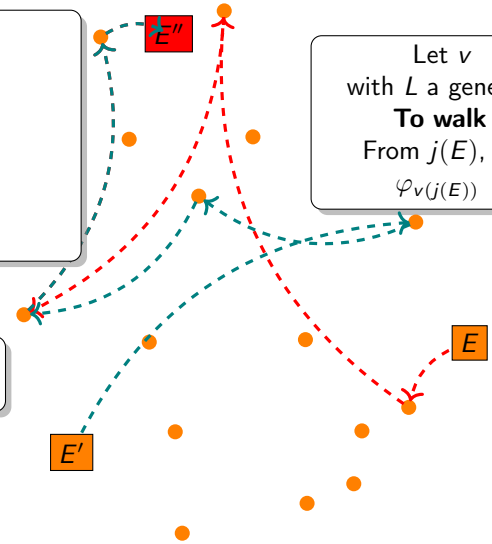
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

What about the memory?

Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)



Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.
To walk deterministically:
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

From $\mathfrak{a}$ s.t. $\varphi_{\mathfrak{a}} : E \to E''$ and $\mathfrak{b}$ s.t. $\varphi_{\mathfrak{b}} : E' \to E''$,

What about the memory?

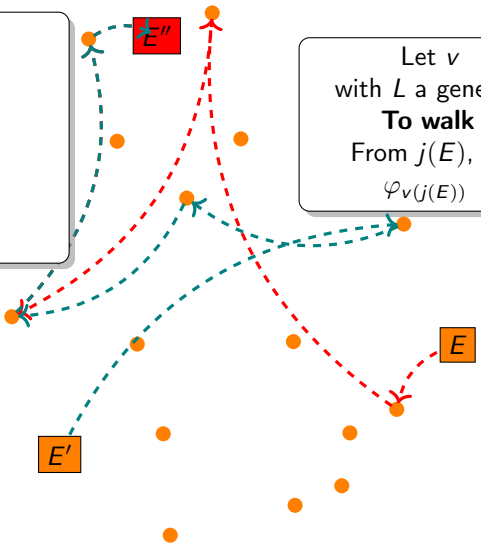
Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)

Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $\mathcal{C}l(\mathbb{Z}[\omega])$.
To walk deterministically:
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

From $\mathfrak{a}$ s.t. $\varphi_{\mathfrak{a}} : E \to E''$ and $\mathfrak{b}$ s.t. $\varphi_{\mathfrak{b}} : E' \to E''$, we know $\varphi_{\mathfrak{a}\bar{\mathfrak{b}}} : E \to E'$.



What about the memory?

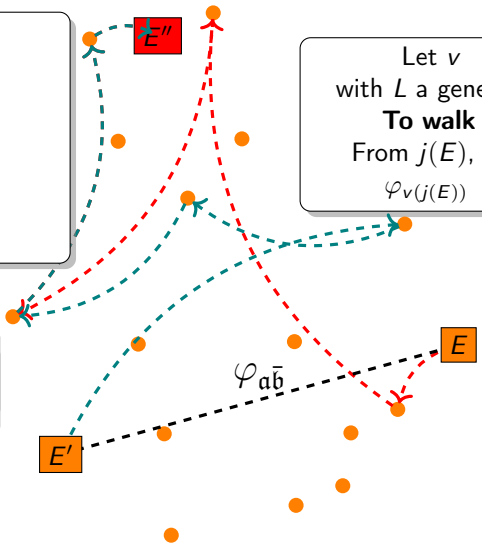
Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)

Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.
To walk deterministically:
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

From $\mathfrak{a}$ s.t. $\varphi_{\mathfrak{a}} : E \to E''$ and $\mathfrak{b}$ s.t. $\varphi_{\mathfrak{b}} : E' \to E''$, we know $\varphi_{\mathfrak{a}\bar{\mathfrak{b}}} : E \to E'$.



What about the memory?

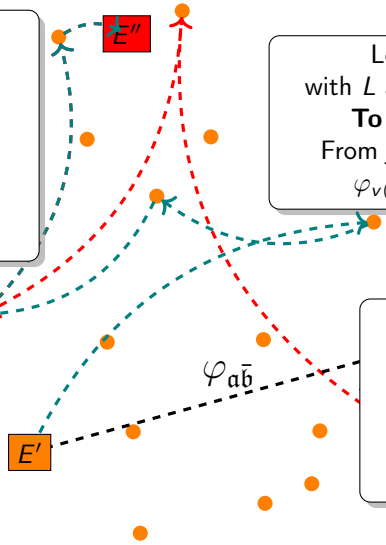
Let's adapt low-storage algorithms to this context [GHS02; GS13].

Method:

- Take $O(p^{1/4})$ steps from E (store the path as an ideal $\mathfrak{a}$)
- Store the last curve E''
- Walk from E' until reaching the stored curve E'' (store the path as an ideal $\mathfrak{b}$)

Let $v : SS_{\mathbb{Z}[\omega]} \to L$, with L a generating set of $Cl(\mathbb{Z}[\omega])$.
To walk deterministically:
From $j(E)$, go to $j(E')$ such that $\varphi_{v(j(E))} : j(E) \to j(E')$.

From $\mathfrak{a}$ s.t. $\varphi_{\mathfrak{a}} : E \to E''$ and $\mathfrak{b}$ s.t. $\varphi_{\mathfrak{b}} : E' \to E''$, we know $\varphi_{\mathfrak{a}\bar{\mathfrak{b}}} : E \to E'$.



Complexity of this step:
Time: $\tilde{O}(p^{1/4})$
Memory: polylog(p)
Total complexity:
Time: $\tilde{O}(p^{1/2})$
Memory: polylog(p)

Conclusion

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm!

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?
Current state of the project:

- A proven detection tool for orientation
- An implementation of the oriented search (Step 1)
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2)
- A suitable metric to compute the optimal primes to consider

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?

Current state of the project:

- A proven detection tool for orientation ✓
- An implementation of the oriented search (Step 1)
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2)
- A suitable metric to compute the optimal primes to consider

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?

Current state of the project:

- A proven detection tool for orientation ✓
- An implementation of the oriented search (Step 1) ✓
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2)
- A suitable metric to compute the optimal primes to consider

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?

Current state of the project:

- A proven detection tool for orientation ✓
- An implementation of the oriented search (Step 1) ✓
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2) ✓
- A suitable metric to compute the optimal primes to consider

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?
Current state of the project:

- A proven detection tool for orientation ✓
- An implementation of the oriented search (Step 1) ✓
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2) ✓
- A suitable metric to compute the optimal primes to consider ×

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?
Current state of the project:

- A proven detection tool for orientation ✓
- An implementation of the oriented search (Step 1) ✓
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2) ✓
- A suitable metric to compute the optimal primes to consider ×

Conclusion

Orientations can be used to improve the concrete complexity of the Delfs–Galbraith algorithm?

Current state of the project:

- A proven detection tool for orientation ✓
- An implementation of the oriented search (Step 1) ✓
- A proof of concept for the deterministic oriented meet-in-the-middle (Step 2) ✓
- A suitable metric to compute the optimal primes to consider ×

Thank you for your attention!

Bibliography I

[CCS22] Maria Corte-Real Santos, Craig Costello, and Jia Shi. "Accelerating the Delfs-Galbraith Algorithm with Fast Subfield Root Detection". In: 2022, pp. 285–314. DOI: 10.1007/978-3-031-15982-4_10.

[CK20] Leonardo Colò and David Kohel. "Orienting supersingular isogeny graphs". In: Journal of Mathematical Cryptology 14.1 (2020), pp. 414–437. DOI: doi:10.1515/jmc-2019-0034. URL: https://doi.org/10.1515/jmc-2019-0034.

[CS22] Mathilde Chenu and Benjamin Smith. "Higher-degree supersingular group actions". In: Mathematical Cryptology 1.2 (Mar. 2022), pp. 85–101. URL: https://inria.hal.science/hal-03288075.

[DG16] Christina Delfs and Steven D. Galbraith. "Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$". In: 78.2 (2016), pp. 425–440. DOI: 10.1007/s10623-014-0010-1.

[GHS02] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. "Extending the GHS Weil Descent Attack". In: 2002, pp. 29–44. DOI: 10.1007/3-540-46035-7_3.

Bibliography II

[GS13] Steven Galbraith and Anton Stolbunov. "Improved algorithm for the isogeny problem for ordinary elliptic curves". In: Appl. Algebra Engrg. Comm. Comput. 24.2 (2013), pp. 107–131. ISSN: 0938-1279,1432-0622. DOI: 10.1007/s00200-013-0185-0. URL: https://doi-org.ezproxy.lib.vt.edu/10.1007/s00200-013-0185-0.

[JMV09] David Jao, Stephen D Miller, and Ramarathnam Venkatesan. "Expander graphs based on GRH with an application to elliptic curve cryptography". In: Journal of Number Theory 129.6 (2009), pp. 1491–1504.

[Onu21] Hiroshi Onuki. "On oriented supersingular elliptic curves". In: Finite Fields and Their Applications 69 (2021). Publisher: Elsevier, p. 101777.

[Sho94] Peter W. Shor. "Algorithms for Quantum Computation: Discrete Logarithms and Factoring". In: 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[Vé71] Jacques Vélu. "Isogénies entre courbes elliptiques". In: C. R. Acad. Sc. Paris, Série A t. 273 (1971), pp. 238–241.

[Wes22] Benjamin Wesolowski. "The supersingular isogeny path and endomorphism ring problems are equivalent". In: 2022, pp. 1100–1111. DOI: 10.1109/FOCS52979.2021.00109.