CSIDH:

An Efficient Post-Quantum Commutative Group Action

https://csidh.isogeny.org

Wouter Castryck[1] Tanja Lange[2] Chloe Martindale[2]
Lorenz Panny[2] Joost Renes[3]

[1]KU Leuven [2]TU Eindhoven [3]RU Nijmegen

Oxford PQC Workshop, 22nd March 2019

[ˈsiːˌsaɪd]

History

1976	Diffie-Hellman: Key exchange using exponentiation in groups (DH)
1985	Koblitz-Miller: Diffie-Hellman style key exchange using multiplication in elliptic curve groups (ECDH)
1990	Brassard-Yung: Generalizes ‘group exponentiation’ to ‘groups acting on sets’ in a crypto context
1994	Shor: Polynomial-time quantum algorithm to break the discrete logarithm problem in any group, quantumly breaking DH and ECDH
1997	Couveignes: Post-quantum isogeny-based Diffie-Hellman-style key exchange using commutative group actions (not published at the time)
2003	Kuperberg: Subexponential-time quantum algorithm to attack cryptosystems based on a hidden shift

History

2004 Stolbunov-Rostovtsev independently rediscover Couveignes' scheme (CRS)

2006 Charles-Goren-Lauter: Build hash function from supersingular isogeny graph

2010 Childs-Jao-Soukharev: Apply Kuperberg's (and Regev's) hidden shift subexponential quantum algorithm to CRS

2011 Jao-De Feo: Build Diffie-Hellman style key exchange from supersingular isogeny graph (SIDH)

2018 De Feo-Kieffer-Smith: Apply new ideas to speed up CRS

2018 Castryck-Lange-Martindale-Panny-Renes: Apply ideas of De Feo, Kieffer, Smith to supersingular curves over $\mathbb{F}_p$ (CSIDH)

(History slides mostly stolen from Wouter Castryck)

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- Non-interactive key exchange (full public-key validation); previously an open problem post-quantumly

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- Non-interactive key exchange (full public-key validation); previously an open problem post-quantumly
- Small keys: 64 bytes at conjectured AES-128 security level

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- Non-interactive key exchange (full public-key validation); previously an open problem post-quantumly
- Small keys: 64 bytes at conjectured AES-128 security level
- Competitive speed: $\sim 25 - 32.5\,$ms per operation

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- Non-interactive key exchange (full public-key validation); previously an open problem post-quantumly
- Small keys: 64 bytes at conjectured AES-128 security level
- Competitive speed: $\sim 25 - 32.5\,\text{ms}$ per operation
- Flexible:
  - [DG] uses CSIDH for 'SeaSign' signatures
  - [DGOPS] uses CSIDH for oblivious transfer
  - [FTY] uses CSIDH for authenticated group key exchange

Post-quantum Diffie-Hellman?

Traditionally, Diffie-Hellman works in a group G via the map

$$\begin{array}{rcl} \mathbb{Z} \times G & \rightarrow & G \\ (x, g) & \mapsto & g^x. \end{array}$$

Post-quantum Diffie-Hellman?

Traditionally, Diffie-Hellman works in a group G via the map

$$\begin{array}{rcl} \mathbb{Z} \times G & \to & G \\ (x, g) & \mapsto & g^x. \end{array}$$

Shor's algorithm quantumly computes x from g^x in any group in polynomial time.

Post-quantum Diffie-Hellman!

Traditionally, Diffie-Hellman works in a group G via the map

$$\begin{array}{ccc} \mathbb{Z} \times G & \to & G \\ (x, g) & \mapsto & g^x. \end{array}$$

Shor's algorithm quantumly computes x from g^x in any group in polynomial time.

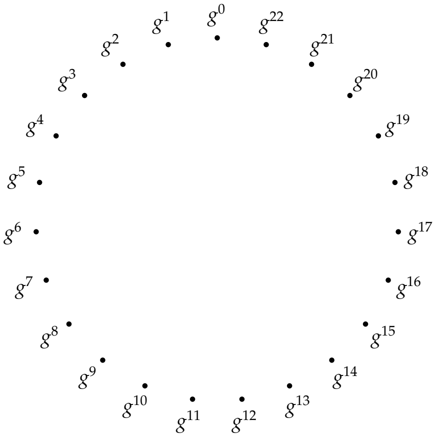
⇝ Idea:

Replace exponentiation on the group G by a group action of a group H on a set S:

$$H \times S \to S.$$

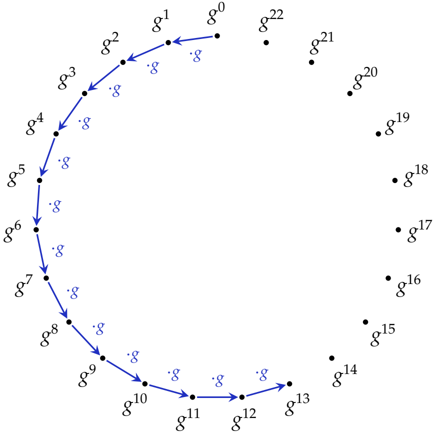
Square-and-multiply

Suppose $\#G = 23$ and that Alice computes g^{13}.



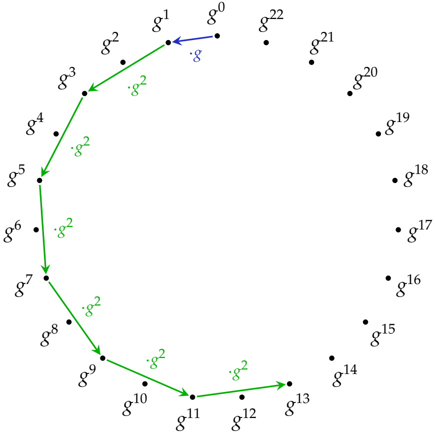
Square-and-multiply

Suppose $\#G = 23$ and that Alice computes g^{13}.



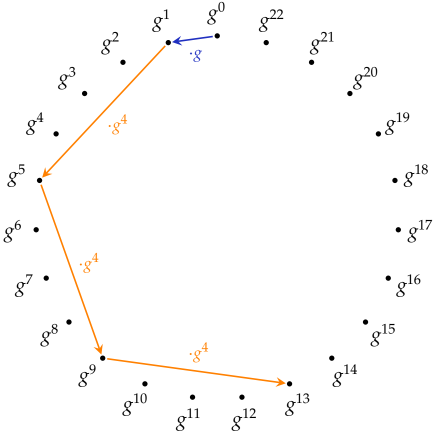
Square-and-multiply

Suppose $\#G = 23$ and that Alice computes g^{13}.



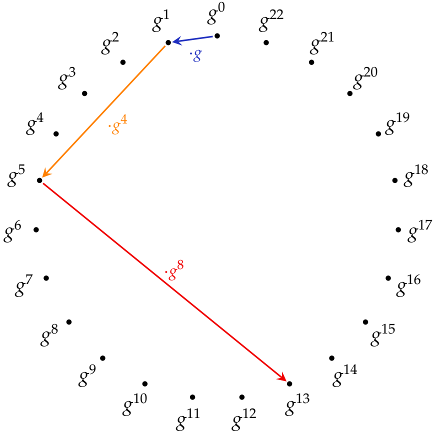
Square-and-multiply

Suppose $\#G = 23$ and that Alice computes g^{13}.

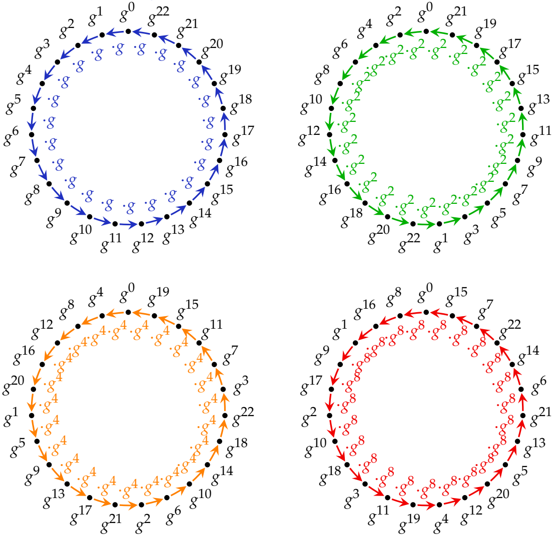


Square-and-multiply

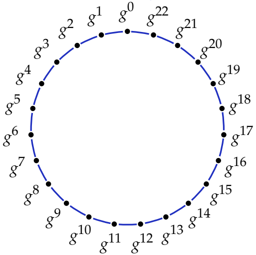
Suppose $\#G = 23$ and that Alice computes g^{13}.

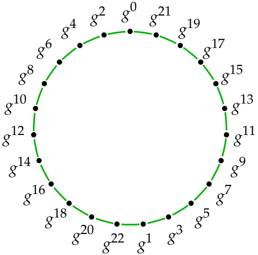


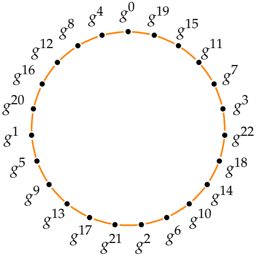
Square-and-multiply

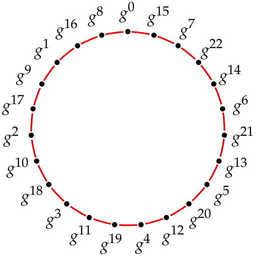


Square-and-multiply

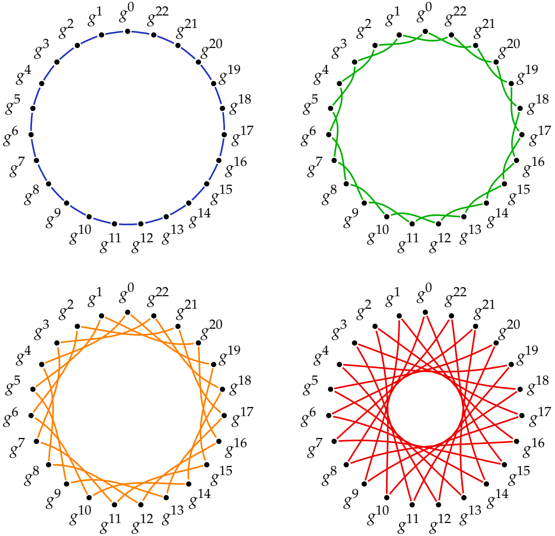




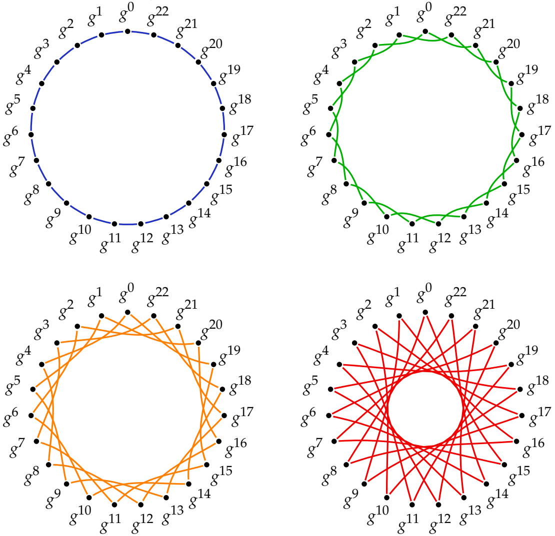




Square-and-multiply

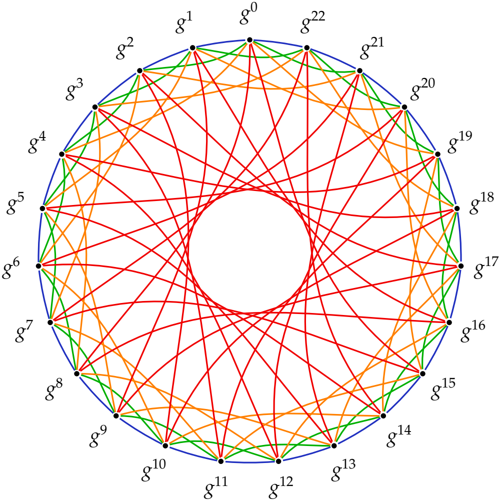


Square-and-multiply

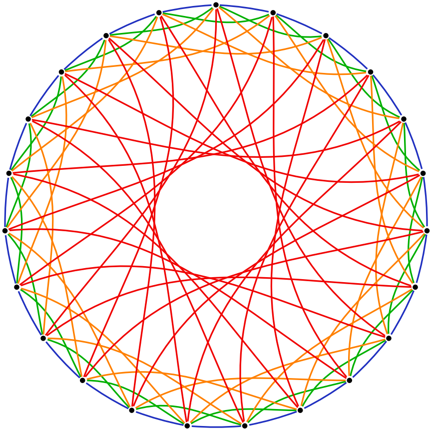


Cycles are compatible: [right, then left] = [left, then right], etc.

Union of cycles: rapid mixing

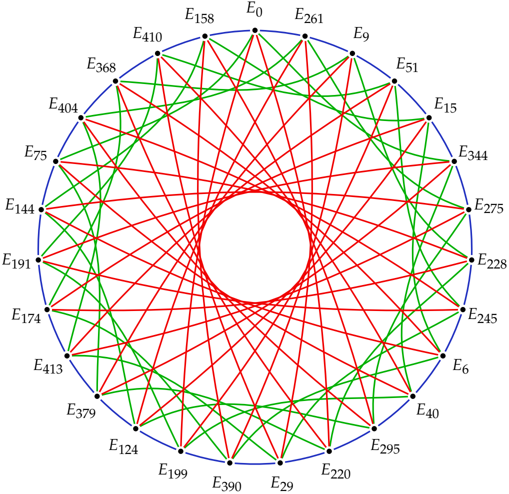


Union of cycles: rapid mixing

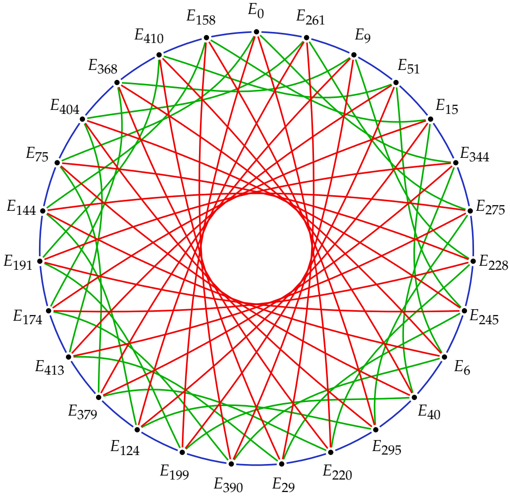


CSIDH: Nodes are now elliptic curves and edges are isogenies.

Graphs of elliptic curves

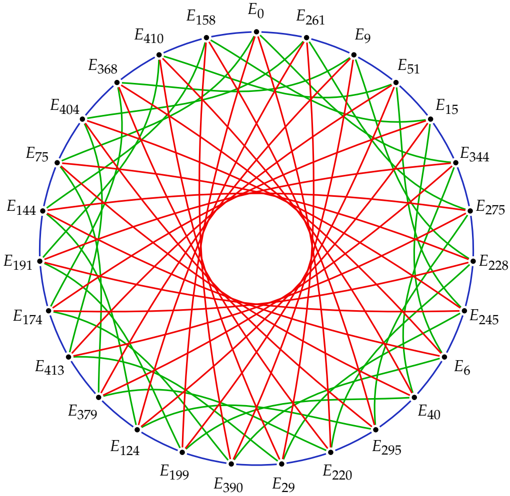


Graphs of elliptic curves



Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

Graphs of elliptic curves



Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
Edges: 3-, 5-, and 7-isogenies.

Interlude: supersingular elliptic curves and isogenies

Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

Edges: 3-, 5-, and 7-isogenies.

Interlude: supersingular elliptic curves and isogenies

Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

- If equation E_A is smooth (no self intersections or cusps) it represents an elliptic curve.

Edges: 3-, 5-, and 7-isogenies.

Interlude: supersingular elliptic curves and isogenies

Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

- If equation E_A is smooth (no self intersections or cusps) it represents an elliptic curve.
- The set of $\mathbb{F}_p$-rational solutions (x, y) to an elliptic curve equation $E_A/\mathbb{F}_p$, together with a 'point at infinity' P_∞, forms a group with identity P_∞, notated $E_A(\mathbb{F}_p)$.

Edges: 3-, 5-, and 7-isogenies.

Interlude: supersingular elliptic curves and isogenies

Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

- If equation E_A is smooth (no self intersections or cusps) it represents an elliptic curve.
- The set of $\mathbb{F}_p$-rational solutions (x, y) to an elliptic curve equation $E_A/\mathbb{F}_p$, together with a 'point at infinity' P_∞, forms a group with identity P_∞, notated $E_A(\mathbb{F}_p)$.
- An elliptic curve $E_A/\mathbb{F}_p$ with $p \geq 5$ such that $\#E_A(\mathbb{F}_p) = p + 1$ is supersingular.

Edges: 3-, 5-, and 7-isogenies.

Interlude: supersingular elliptic curves and isogenies

Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

- If equation E_A is smooth (no self intersections or cusps) it represents an elliptic curve.
- The set of $\mathbb{F}_p$-rational solutions (x, y) to an elliptic curve equation $E_A/\mathbb{F}_p$, together with a 'point at infinity' P_∞, forms a group with identity P_∞, notated $E_A(\mathbb{F}_p)$.
- An elliptic curve $E_A/\mathbb{F}_p$ with $p \geq 5$ such that $\#E_A(\mathbb{F}_p) = p + 1$ is supersingular.

Edges: 3-, 5-, and 7-isogenies.

- An isogeny $E_A \to E_B$ is a non-zero morphism ('nice map') that preserves P_∞.

Interlude: supersingular elliptic curves and isogenies

Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

- If equation E_A is smooth (no self intersections or cusps) it represents an elliptic curve.
- The set of $\mathbb{F}_p$-rational solutions (x, y) to an elliptic curve equation $E_A/\mathbb{F}_p$, together with a 'point at infinity' P_∞, forms a group with identity P_∞, notated $E_A(\mathbb{F}_p)$.
- An elliptic curve $E_A/\mathbb{F}_p$ with $p \geq 5$ such that $\#E_A(\mathbb{F}_p) = p + 1$ is supersingular.

Edges: 3-, 5-, and 7-isogenies.

- An isogeny $E_A \to E_B$ is a non-zero morphism ('nice map') that preserves P_∞.
- For $\ell \neq p$ (= 419 here), an ℓ-isogeny $f : E_A \to E_B$ is an isogeny with $\#\ker(f) = \ell$.

Interlude: supersingular elliptic curves and isogenies

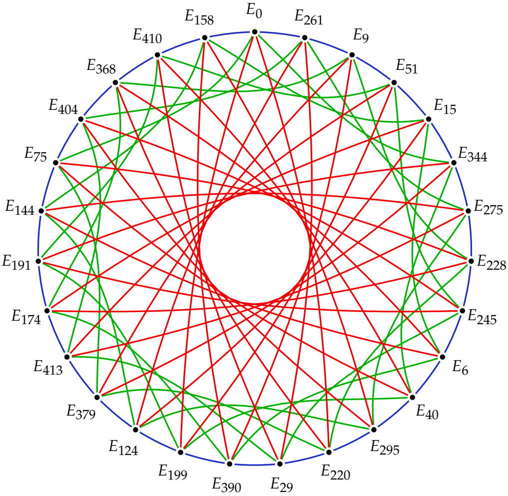
Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

- If equation E_A is smooth (no self intersections or cusps) it represents an elliptic curve.
- The set of $\mathbb{F}_p$-rational solutions (x, y) to an elliptic curve equation $E_A/\mathbb{F}_p$, together with a 'point at infinity' P_∞, forms a group with identity P_∞, notated $E_A(\mathbb{F}_p)$.
- An elliptic curve $E_A/\mathbb{F}_p$ with $p \geq 5$ such that $\#E_A(\mathbb{F}_p) = p + 1$ is supersingular.

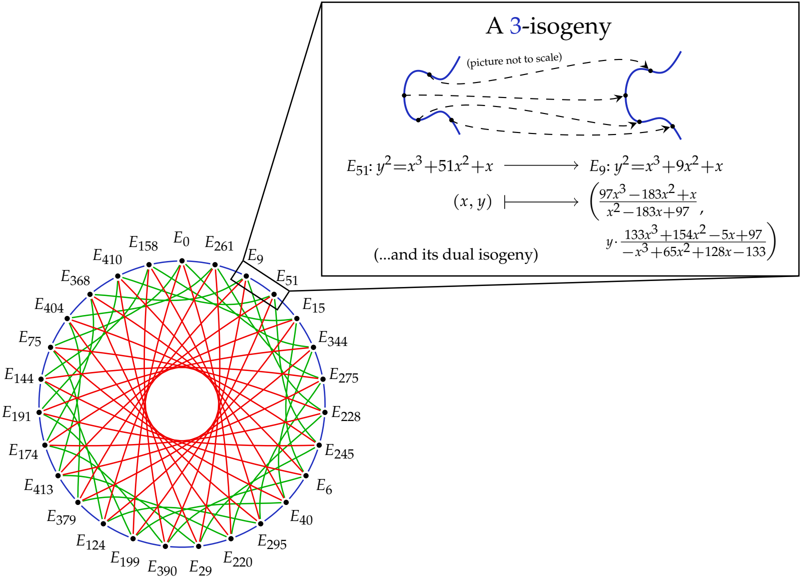
Edges: 3-, 5-, and 7-isogenies.

- An isogeny $E_A \to E_B$ is a non-zero morphism ('nice map') that preserves P_∞.
- For $\ell \neq p$ (= 419 here), an ℓ-isogeny $f : E_A \to E_B$ is an isogeny with $\#\ker(f) = \ell$.
- Every ℓ-isogeny $f : E_A \to E_B$ has a unique dual ℓ-isogeny $f : E_B \to E_A$. $\rightsquigarrow$ Undirected edges!

Graphs of elliptic curves



Graphs of elliptic curves



A 3-isogeny

$$E_{51}\colon y^2=x^3+51x^2+x \longrightarrow E_9\colon y^2=x^3+9x^2+x$$

$$(x,y) \longmapsto \left(\frac{97x^3-183x^2+x}{x^2-183x+97},\; y\cdot\frac{133x^3+154x^2-5x+97}{-x^3+65x^2+128x-133}\right)$$

(...and its dual isogeny)

Quantumifying Exponentiation

- Recall: we want to replace the exponentiation map

  $$\begin{array}{ccc} \mathbb{Z} \times G & \to & G \\ (x, g) & \mapsto & g^x \end{array}$$

  by a group action on a set.

Quantumifying Exponentiation

- Recall: we want to replace the exponentiation map

$$\begin{array}{ccc} \mathbb{Z} \times G & \to & G \\ (x, g) & \mapsto & g^x \end{array}$$

  by a group action on a set.

- Replace G by the set S of supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

Quantumifying Exponentiation

- Recall: we want to replace the exponentiation map

$$\begin{array}{ccc} \mathbb{Z} \times G & \to & G \\ (x, g) & \mapsto & g^x \end{array}$$

  by a group action on a set.
- Replace G by the set S of supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
- Replace $\mathbb{Z}$ by a commutative group $(H, *)$... more details to come!

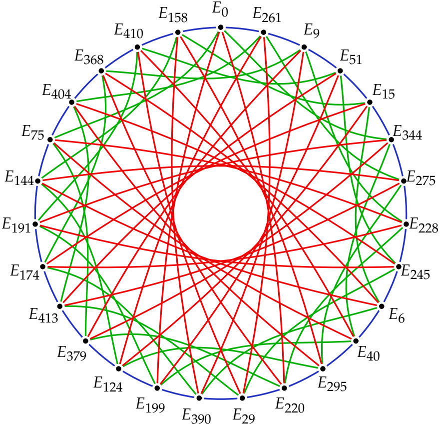
Quantumifying Exponentiation

- Recall: we want to replace the exponentiation map

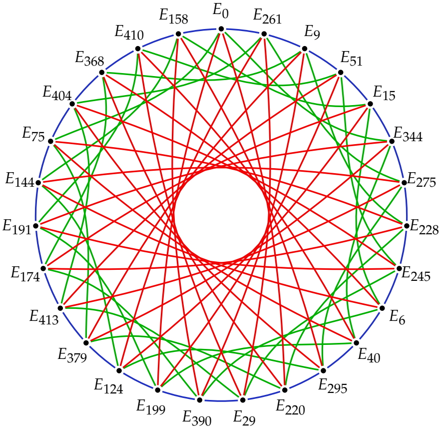
$$\begin{array}{ccc} \mathbb{Z} \times G & \to & G \\ (x, g) & \mapsto & g^x \end{array}$$

  by a group action on a set.
- Replace G by the set S of supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
- Replace $\mathbb{Z}$ by a commutative group $(H, *)$... more details to come!
- The action of a well-chosen h (or h^{-1}) $\in H$ on $E_A \in S$, written $h \cdot E_A$ (or $h^{-1} \cdot E_A$) gives an elliptic curve one step from E_A around one of the cycles in a $+$ (or $-$) direction.

Graphs of elliptic curves

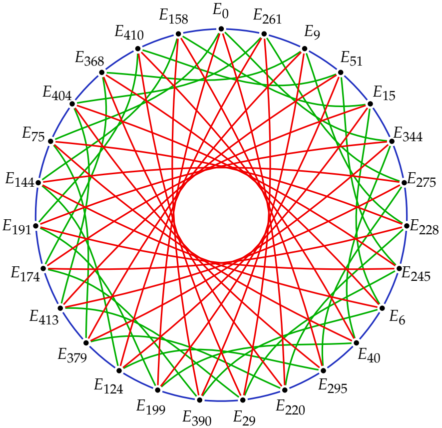


Graphs of elliptic curves



Nodes: Set S of supersingular $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

Graphs of elliptic curves

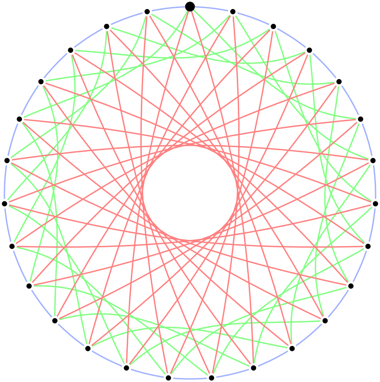


Nodes: Set S of supersingular $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
Edges: 3-, 5-, and 7-isogenies,
given (in clockwise direction) by action of $h_3, h_5, h_7 \in H$.

Diffie-Hellman on isogeny graphs

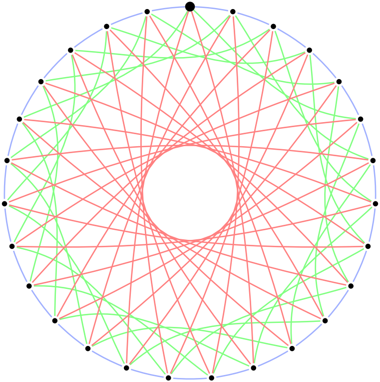
Alice

$[+, -, +, -]$



Bob

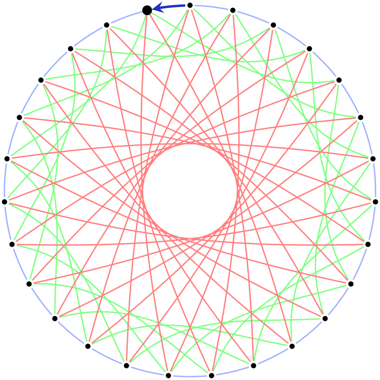
$[+, +, -, +]$



Diffie-Hellman on isogeny graphs

Alice

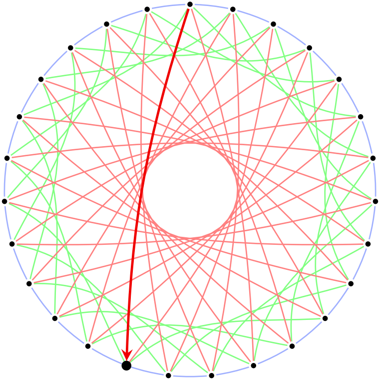
$[+, -, +, -]$
↑



$h_3 \cdot E_0$

Bob

$[+, +, -, +]$
↑

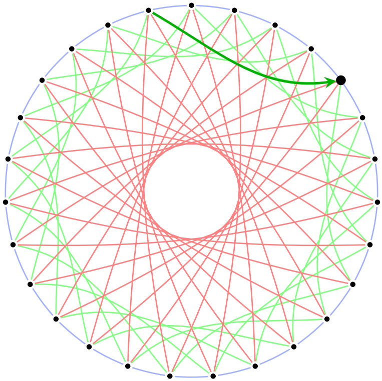


$h_5 \cdot E_0$

Diffie-Hellman on isogeny graphs

Alice

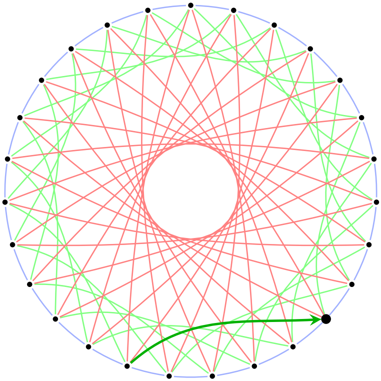
$[+, -, +, -]$
↑



$h_7^{-1} \cdot (\; h_3 \cdot E_0 \;)$

Bob

$[+, +, -, +]$
↑



$h_7 \cdot (\; h_5 \cdot E_0 \;)$

Diffie-Hellman on isogeny graphs

Alice

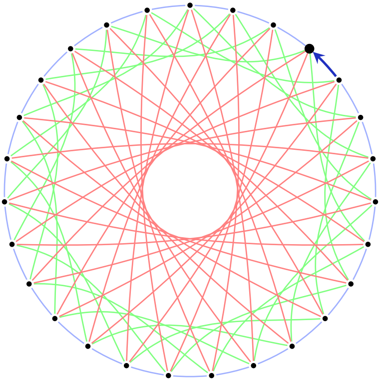
$[+, -, +, -]$

↑

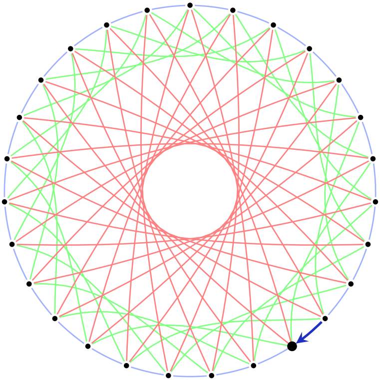
Bob

$[+, +, -, +]$

↑



$h_3 \cdot (h_7^{-1} \cdot (h_3 \cdot E_0))$



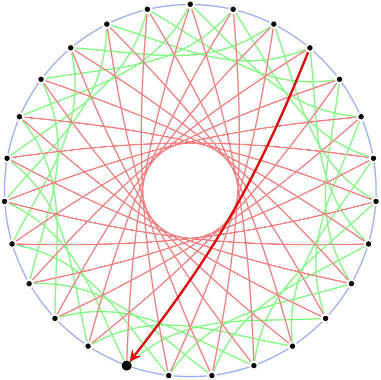
$h_3^{-1} \cdot (h_7 \cdot (h_5 \cdot E_0))$

Diffie-Hellman on isogeny graphs

Alice

$[+, -, +, -]$

↑

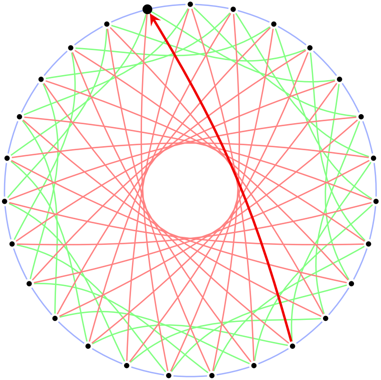


$E_A = h_5^{-1} \cdot (h_3 \cdot (h_7^{-1} \cdot (h_3 \cdot E_0)))$

Bob

$[+, +, -, +]$

↑



$E_B = h_5 \cdot (h_3^{-1} \cdot (h_7 \cdot (h_5 \cdot E_0)))$

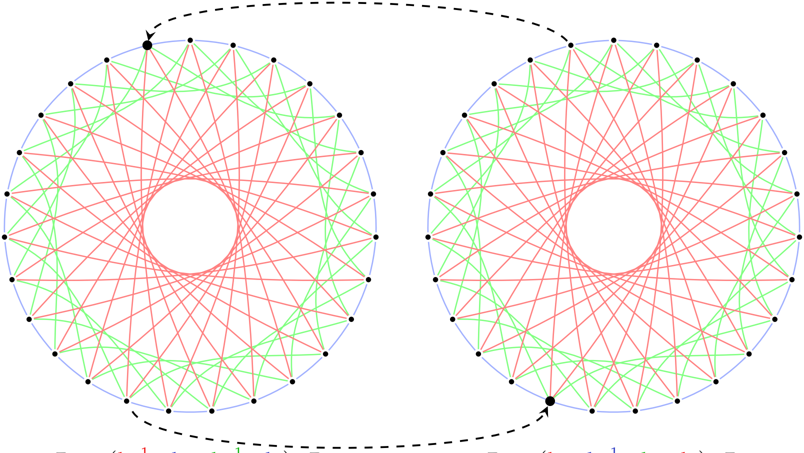
Diffie-Hellman on isogeny graphs

Alice

$[+, -, +, -]$

Bob

$[+, +, -, +]$



$E_A = (h_5^{-1} * h_3 * h_7^{-1} * h_3) \cdot E_0$

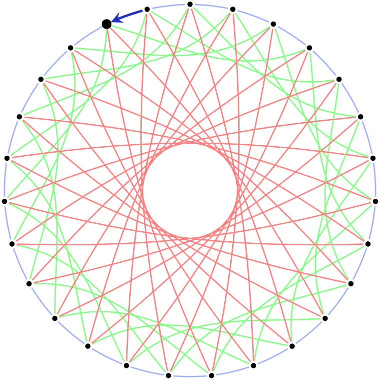
$E_B = (h_5 * h_3^{-1} * h_7 * h_5) \cdot E_0$

Diffie-Hellman on isogeny graphs

Alice

$[+, -, +, -]$

$\uparrow$

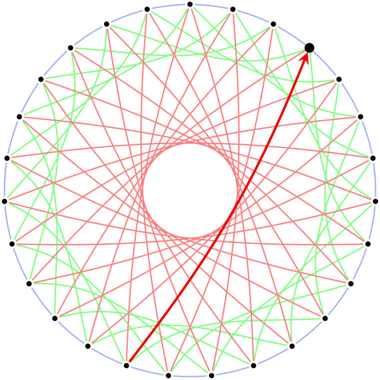


$h_3 \cdot E_B$

Bob

$[+, +, -, +]$

$\uparrow$

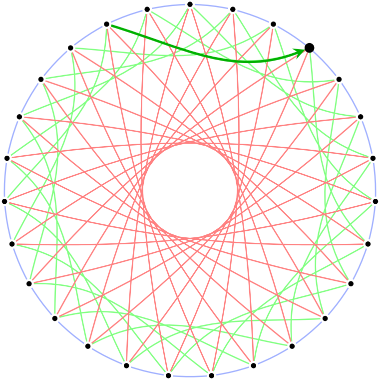


$h_5 \cdot E_A$

Diffie-Hellman on isogeny graphs

Alice

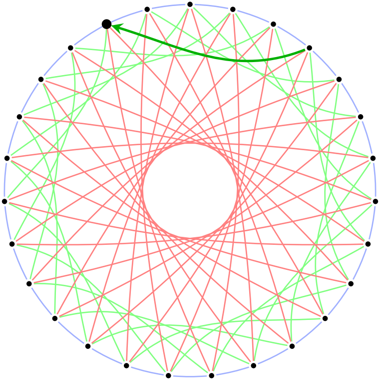
$[+, -, +, -]$
$\quad\uparrow$



$h_7^{-1} \cdot (\ h_3 \cdot E_B\ )$

Bob

$[+, +, -, +]$
$\quad\uparrow$

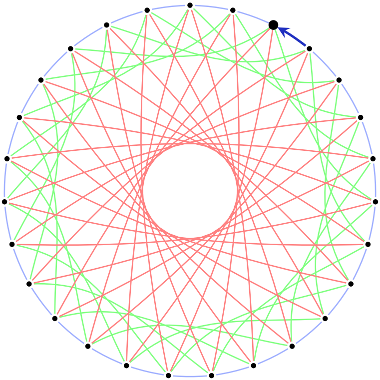


$h_7 \cdot (\ h_5 \cdot E_A\ )$

Diffie-Hellman on isogeny graphs

Alice

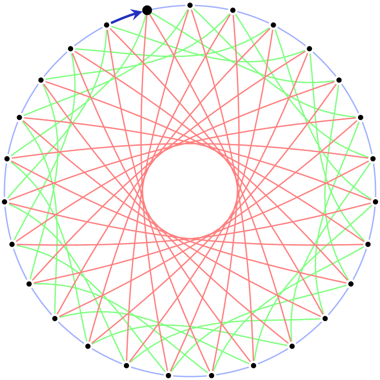
$[+, -, \underset{\uparrow}{+}, -]$



$h_3 \cdot (h_7^{-1} \cdot (h_3 \cdot E_B))$

Bob

$[+, +, \underset{\uparrow}{-}, +]$



$h_3^{-1} \cdot (h_7 \cdot (h_5 \cdot E_A))$

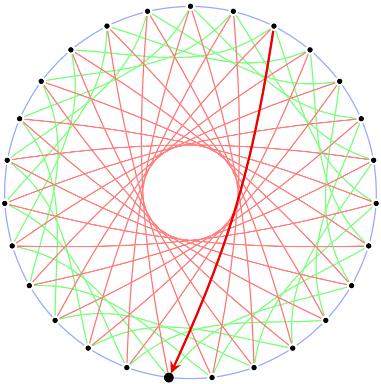
Diffie-Hellman on isogeny graphs

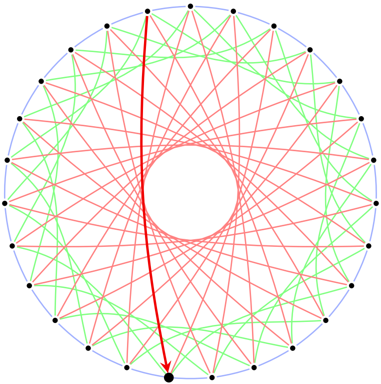
Alice

$[+, -, +, -]$
$\uparrow$

Bob

$[+, +, -, +]$
$\uparrow$





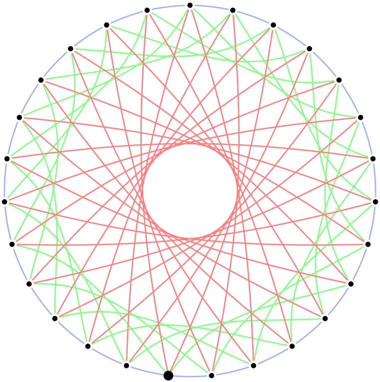
$E_{AB} = h_5^{-1} \cdot (\ h_3 \cdot (\ h_7^{-1} \cdot (\ h_3 \cdot E_B\ )))$

$E_{BA} = h_5 \cdot (\ h_3^{-1} \cdot (\ h_7 \cdot (\ h_5 \cdot E_A\ )))$

Diffie-Hellman on isogeny graphs

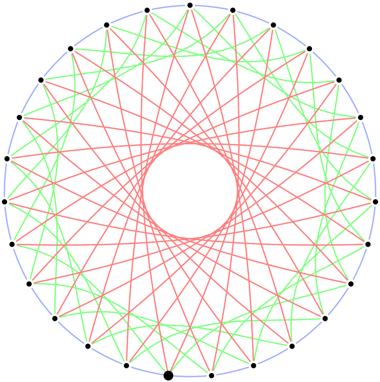
Alice

$[+, -, +, -]$



Bob

$[+, +, -, +]$



$$E_{AB} = E_{BA} = (h_5^{-1} * h_3 * h_7^{-1} * h_3 * h_5 * h_3^{-1} * h_7 * h_5) \cdot E_0$$

A walkable graph

- Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
- Edges: 3-, 5-, and 7-isogenies.

A walkable graph

- Nodes: Supersingular elliptic curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
- Edges: 3-, 5-, and 7-isogenies.

Important properties for such a walk:

IP1 ▸ The graph is a composition of compatible cycles.

IP2 ▸ We can compute neighbours in given directions.

Towards IP1: Isogeny graphs

Definition

Let p and ℓ be distinct primes. The isogeny graph G_ℓ containing $E/\mathbb{F}_p$ is the graph with:

- Nodes: elliptic curves $E'/\mathbb{F}_p$ (up to $\mathbb{F}_p$-isomorphism) with $\#E(\mathbb{F}_p) = \#E'(\mathbb{F}_p)$.
- Edges: we draw an edge $E - E'$ to represent an ℓ-isogeny $f : E \to E'$ together with its dual ℓ-isogeny.

Towards IP1: Isogeny graphs

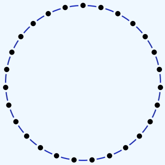
Definition

Let p and ℓ be distinct primes. The isogeny graph G_ℓ containing $E/\mathbb{F}_p$ is the graph with:

- Nodes: elliptic curves $E'/\mathbb{F}_p$ (up to $\mathbb{F}_p$-isomorphism) with $\#E(\mathbb{F}_p) = \#E'(\mathbb{F}_p)$.
- Edges: we draw an edge $E - E'$ to represent an ℓ-isogeny $f : E \to E'$ together with its dual ℓ-isogeny.

- In our example, these are

G_3:



Towards IP1: Isogeny graphs

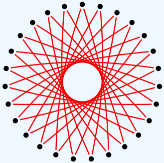
Definition

Let p and ℓ be distinct primes. The isogeny graph G_ℓ containing $E/\mathbb{F}_p$ is the graph with:

- Nodes: elliptic curves $E'/\mathbb{F}_p$ (up to $\mathbb{F}_p$-isomorphism) with $\#E(\mathbb{F}_p) = \#E'(\mathbb{F}_p)$.
- Edges: we draw an edge $E - E'$ to represent an ℓ-isogeny $f : E \to E'$ together with its dual ℓ-isogeny.

- In our example, these are

G_5:



Towards IP1: Isogeny graphs

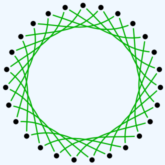
Definition

Let p and ℓ be distinct primes. The isogeny graph G_ℓ containing $E/\mathbb{F}_p$ is the graph with:

- Nodes: elliptic curves $E'/\mathbb{F}_p$ (up to $\mathbb{F}_p$-isomorphism) with $\#E(\mathbb{F}_p) = \#E'(\mathbb{F}_p)$.
- Edges: we draw an edge $E - E'$ to represent an ℓ-isogeny $f : E \to E'$ together with its dual ℓ-isogeny.

- In our example, these are

G_7:



Towards IP1: Isogeny graphs

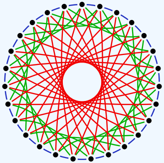
Definition

Let p and ℓ be distinct primes. The isogeny graph G_ℓ containing $E/\mathbb{F}_p$ is the graph with:

- Nodes: elliptic curves $E'/\mathbb{F}_p$ (up to $\mathbb{F}_p$-isomorphism) with $\#E(\mathbb{F}_p) = \#E'(\mathbb{F}_p)$.
- Edges: we draw an edge $E - E'$ to represent an ℓ-isogeny $f : E \to E'$ together with its dual ℓ-isogeny.

- In our example, these are

$G_3 \cup G_5 \cup G_7$:



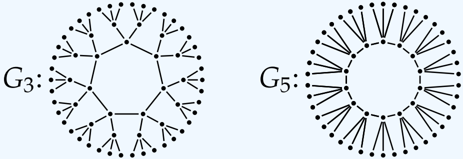
Towards IP1: Isogeny graphs

Definition

Let p and ℓ be distinct primes. The isogeny graph G_ℓ containing $E/\mathbb{F}_p$ is the graph with:

- Nodes: elliptic curves $E'/\mathbb{F}_p$ (up to $\mathbb{F}_p$-isomorphism) with $\#E(\mathbb{F}_p) = \#E'(\mathbb{F}_p)$.
- Edges: we draw an edge $E - E'$ to represent an ℓ-isogeny $f : E \to E'$ together with its dual ℓ-isogeny.

- Generally, the G_ℓ look something like



Towards IP1: Endomorphism rings

- We want to make sure G_ℓ is a disjoint union of cycles.

Towards IP1: Endomorphism rings

- We want to make sure G_ℓ is a disjoint union of cycles.
- Equivalently: every node in G_ℓ should be distance zero from a cycle.

Towards IP1: Endomorphism rings

- We want to make sure G_ℓ is a disjoint union of cycles.
- Equivalently: every node in G_ℓ should be distance zero from a cycle.
- If two nodes have the same endomorphism ring then they are at same distance from a cycle.

Towards IP1: Endomorphism rings

Definition

An endomorphism of an elliptic curve E is a morphism $E \to E$ (as abelian varieties).

Towards IP1: Endomorphism rings

Definition

An endomorphism of an elliptic curve E is a morphism $E \to E$ (as abelian varieties).

Example

Let $E/\mathbb{F}_p$ be an elliptic curve.

- For $n \in \mathbb{Z}$, the mulitplication-by-n map

$$[n] : \begin{array}{ccc} E & \to & E \\ P & \mapsto & nP \end{array}$$

  is an endomorphism.

Towards IP1: Endomorphism rings

Definition

An endomorphism of an elliptic curve E is a morphism $E \to E$ (as abelian varieties).

Example

Let $E/\mathbb{F}_p$ be an elliptic curve.

- For $n \in \mathbb{Z}$, the mulitplication-by-n map

$$[n]: \begin{array}{ccc} E & \to & E \\ P & \mapsto & nP \end{array}$$

  is an endomorphism.

- The Frobenius map

$$\pi: \begin{array}{ccc} E & \to & E \\ (x,y) & \mapsto & (x^p, y^p) \end{array}$$

  is an endomorphism.

Towards IP1: Endomorphism rings

Definition

The $\mathbb{F}_p$-rational endomorphism ring $\mathrm{End}_{\mathbb{F}_p}(E)$ of an elliptic curve $E/\mathbb{F}_p$ is the set of $\mathbb{F}_p$-rational endomorphisms.

Towards IP1: Endomorphism rings

Definition

The $\mathbb{F}_p$-rational endomorphism ring $\mathrm{End}_{\mathbb{F}_p}(E)$ of an elliptic curve $E/\mathbb{F}_p$ is the set of $\mathbb{F}_p$-rational endomorphisms.

Example

Let $p \geq 5$, let $E/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ be a supersingular elliptic curve, and let π be the Frobenius endomorphism. Then

$$\pi \circ \pi = [-p]$$

and

$$\begin{array}{ccc} \mathbb{Z}[\sqrt{-p}] & \to & \mathrm{End}_{\mathbb{F}_p}(E) \\ n & \mapsto & [n] \\ \sqrt{-p} & \mapsto & \pi \end{array}$$

extends $\mathbb{Z}$-linearly to a ring homomorphism.

Towards IP1: A composition of cycles

For $p \equiv 3 \pmod 8$ and $p \geq 5$, if $E_A/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ is supersingular, then $\mathrm{End}_{\mathbb{F}_p}(E_A) \cong \mathbb{Z}[\sqrt{-p}]$.

Towards IP1: A composition of cycles

For $p \equiv 3 \pmod 8$ and $p \geq 5$, if $E_A/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ is supersingular, then $\mathrm{End}_{\mathbb{F}_p}(E_A) \cong \mathbb{Z}[\sqrt{-p}]$.

Recall:

- We want to make sure the isogeny graph G_ℓ is a disjoint union of cycles.
- Equivalently: every node in G_ℓ should be distance zero from a cycle.
- If two nodes have the same endomorphism ring then they are at same distance from a cycle.

Towards IP1: A composition of cycles

For $p \equiv 3 \pmod 8$ and $p \geq 5$, if $E_A/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ is supersingular, then $\mathrm{End}_{\mathbb{F}_p}(E_A) \cong \mathbb{Z}[\sqrt{-p}]$.

Recall:

- We want to make sure the isogeny graph G_ℓ is a disjoint union of cycles.
- Equivalently: every node in G_ℓ should be distance zero from a cycle.
- If two nodes have the same endomorphism ring then they are at same distance from a cycle.

$\rightsquigarrow$ take G_ℓ to be the isogeny graph containing supersingular $E_A/\mathbb{F}_p$ with $p \equiv 3 \pmod 8$.

IP1: A composition of compatible cycles

- Remember: we wanted to replace exponentiation $\mathbb{Z} \times G \to G$ with a commutative group action $H \times S \to S$.
  $\rightsquigarrow$ cycles are compatible.

IP1: A composition of compatible cycles

- Remember: we wanted to replace exponentiation $\mathbb{Z} \times G \to G$ with a commutative group action $H \times S \to S$.
  $\rightsquigarrow$ cycles are compatible.
- The set S is the set of supersingular elliptic curves $E_A/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ with $p \equiv 3 \pmod 8$ and $p \geq 5$.

IP1: A composition of compatible cycles

- Remember: we wanted to replace exponentiation $\mathbb{Z} \times G \to G$ with a commutative group action $H \times S \to S$.
  $\rightsquigarrow$ cycles are compatible.
- The set S is the set of supersingular elliptic curves $E_A/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$ with $p \equiv 3 \pmod 8$ and $p \geq 5$.
- The group $H = \mathrm{Cl}(\mathbb{Z}[\sqrt{-p}])$ is the class group of $\mathrm{End}_{\mathbb{F}_p}(E_A)$ for (every) $E_A \in S$.

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
    - Let $E_A/\mathbb{F}_p$ be supersingular and $p \geq 5$. Then $E_A(\mathbb{F}_p) \cong C_{p+1}$ or $C_2 \times C_{(p+1)/2}$.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
  - Let $E_A/\mathbb{F}_p$ be supersingular and $p \geq 5$. Then $E_A(\mathbb{F}_p) \cong C_{p+1}$ or $C_2 \times C_{(p+1)/2}$.
  - Suppose we have found $Q = E_A(\mathbb{F}_p)$ of order $p + 1$ or $(p + 1)/2$.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
  - Let $E_A/\mathbb{F}_p$ be supersingular and $p \geq 5$. Then $E_A(\mathbb{F}_p) \cong C_{p+1}$ or $C_2 \times C_{(p+1)/2}$.
  - Suppose we have found $Q = E_A(\mathbb{F}_p)$ of order $p+1$ or $(p+1)/2$.
  - For every odd prime $\ell | (p+1)$, the point $P = \frac{p+1}{\ell} Q$ is a point of order ℓ.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
  - Let $E_A/\mathbb{F}_p$ be supersingular and $p \geq 5$. Then $E_A(\mathbb{F}_p) \cong C_{p+1}$ or $C_2 \times C_{(p+1)/2}$.
  - Suppose we have found $Q = E_A(\mathbb{F}_p)$ of order $p+1$ or $(p+1)/2$.
  - For every odd prime $\ell | (p+1)$, the point $P = \frac{p+1}{\ell} Q$ is a point of order ℓ.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
    - Let $E_A/\mathbb{F}_p$ be supersingular and $p \geq 5$. Then $E_A(\mathbb{F}_p) \cong C_{p+1}$ or $C_2 \times C_{(p+1)/2}$.
    - Suppose we have found $Q = E_A(\mathbb{F}_p)$ of order $p + 1$ or $(p + 1)/2$.
    - For every odd prime $\ell | (p + 1)$, the point $P = \frac{p+1}{\ell} Q$ is a point of order ℓ.
- Compute the isogeny with kernel $\{P, 2P, \dots, \ell P\}$ using Vélu's formulas (implemented in Sage).
    - Given a $\mathbb{F}_p$-rational point of order ℓ, the isogeny computations can be done over $\mathbb{F}_p$.

Towards IP2: Computing the neighbours (in given directions)

To compute a neighbour of E_A, we have to compute an ℓ-isogeny from E_A. To do this:

- Find a point P of order ℓ on E_A.
  - Let $E_A/\mathbb{F}_p$ be supersingular and $p \geq 5$. Then $E_A(\mathbb{F}_p) \cong C_{p+1}$ or $C_2 \times C_{(p+1)/2}$.
  - Suppose we have found $Q = E_A(\mathbb{F}_p)$ of order $p+1$ or $(p+1)/2$.
  - For every odd prime $\ell | (p+1)$, the point $P = \frac{p+1}{\ell} Q$ is a point of order ℓ.
- Compute the isogeny with kernel $\{P, 2P, \ldots, \ell P\}$ using Vélu's formulas (implemented in Sage).
  - Given a $\mathbb{F}_p$-rational point of order ℓ, the isogeny computations can be done over $\mathbb{F}_p$.

(The direction can be easily computed as well, but that requires a bit more background).

IP2: Computing neighbours in given directions

For which ℓ can we (efficiently) compute the neighbours of supersingular $E_A/\mathbb{F}_p$ in its ℓ-isogeny graph G_ℓ for odd $\ell | (p+1)$?

---

[1] You still need a little more to get computations for both the $+$ and $-$ directions to be over $\mathbb{F}_p$

IP2: Computing neighbours in given directions

For which ℓ can we (efficiently) compute the neighbours of supersingular $E_A/\mathbb{F}_p$ in its ℓ-isogeny graph G_ℓ for odd $\ell|(p+1)$? Choosing $p = 4\ell_1 \cdots \ell_n - 1$ ensures:

- Every $\ell_i|(p+1)$, so there is a rational basis point of the ℓ_i-torsion

---

[1] You still need a little more to get computations for both the $+$ and $-$ directions to be over $\mathbb{F}_p$

IP2: Computing neighbours in given directions

For which ℓ can we (efficiently) compute the neighbours of supersingular $E_A/\mathbb{F}_p$ in its ℓ-isogeny graph G_ℓ for odd $\ell|(p+1)$? Choosing $p = 4\ell_1 \cdots \ell_n - 1$ ensures:

- Every $\ell_i|(p+1)$, so there is a rational basis point of the ℓ_i-torsion
- $p \equiv 3 \pmod 8$, so G_{ℓ_i} is a cycle (we have our group action)

---

[1] You still need a little more to get computations for both the $+$ and $-$ directions to be over $\mathbb{F}_p$

IP2: Computing neighbours in given directions

For which ℓ can we (efficiently) compute the neighbours of supersingular $E_A/\mathbb{F}_p$ in its ℓ-isogeny graph G_ℓ for odd $\ell|(p+1)$?
Choosing $p = 4\ell_1 \cdots \ell_n - 1$ ensures:

- Every $\ell_i|(p+1)$, so there is a rational basis point of the ℓ_i-torsion
- $p \equiv 3 \pmod 8$, so G_{ℓ_i} is a cycle (we have our group action)

With our design choices all isogeny computations are over $\mathbb{F}_p$. [1]

---

[1] You still need a little more to get computations for both the $+$ and $-$ directions to be over $\mathbb{F}_p$

Representing nodes of the graph

- Every node of G_{ℓ_i} is

$$E_A : y^2 = x^3 + Ax^2 + x.$$

Representing nodes of the graph

- Every node of G_{ℓ_i} is

$$E_A\colon y^2 = x^3 + Ax^2 + x.$$

$\Rightarrow$ Can compress every node to a single value $A \in \mathbb{F}_p$.

Representing nodes of the graph

- Every node of G_{ℓ_i} is

$$E_A : y^2 = x^3 + Ax^2 + x.$$

⇒ Can compress every node to a single value $A \in \mathbb{F}_p$.

⇒ Tiny keys!

Does any A work?

---

[2] This algorithm has a small chance of false positives, but we actually use a variant that *proves* that E_A has $p+1$ points.

Does any A work?

No.

---

[2] This algorithm has a small chance of false positives, but we actually use a variant that *proves* that E_A has $p + 1$ points.

Does any A work?

No.

- About $\sqrt{p}$ of all $A \in \mathbb{F}_p$ are valid keys.

---

[2] This algorithm has a small chance of false positives, but we actually use a variant that *proves* that E_A has $p + 1$ points.

Does any A work?

No.

- About $\sqrt{p}$ of all $A \in \mathbb{F}_p$ are valid keys.
- Public-key validation: Check that E_A has $p + 1$ points.

  Easy Monte-Carlo algorithm: Pick random P on E_A and check $[p + 1]P = \infty$.[2]

---

[2] This algorithm has a small chance of false positives, but we actually use a variant that *proves* that E_A has $p + 1$ points.

Classical Security

- Security is based on the isogeny problem: given two elliptic curves, compute an isogeny between them.

Classical Security

- Security is based on the isogeny problem: given two elliptic curves, compute an isogeny between them.
- Say Alice's secret is isogeny is of degree $\ell_1^{e_1} \cdots \ell_n^{e_n}$. She knows the path, so can do only small degree isogeny computations, giving complexity $O(\sum e_i \ell_i)$.

Classical Security

- Security is based on the isogeny problem: given two elliptic curves, compute an isogeny between them.
- Say Alice's secret is isogeny is of degree $\ell_1^{e_1} \cdots \ell_n^{e_n}$. She knows the path, so can do only small degree isogeny computations, giving complexity $O(\sum e_i \ell_i)$. An attacker has to compute one isogeny of large degree.

Classical Security

- Security is based on the isogeny problem: given two elliptic curves, compute an isogeny between them.
- Say Alice's secret is isogeny is of degree $\ell_1^{e_1} \cdots \ell_n^{e_n}$. She knows the path, so can do only small degree isogeny computations, giving complexity $O(\sum e_i \ell_i)$. An attacker has to compute one isogeny of large degree.
- Alternative way of thinking about it: Alice has to compute the isogeny corresponding to one path from E_0 to E_A, whereas an attacker has compute all the possible paths from E_0.

Classical Security

- Security is based on the isogeny problem: given two elliptic curves, compute an isogeny between them.
- Say Alice's secret is isogeny is of degree $\ell_1^{e_1} \cdots \ell_n^{e_n}$. She knows the path, so can do only small degree isogeny computations, giving complexity $O(\sum e_i \ell_i)$. An attacker has to compute one isogeny of large degree.
- Alternative way of thinking about it: Alice has to compute the isogeny corresponding to one path from E_0 to E_A, whereas an attacker has compute all the possible paths from E_0.
- Best classical attacks are (variants of) meet-in-the-middle: Time $O(\sqrt[4]{p})$.

Quantum Security

Hidden-shift algorithms: Subexponential complexity (Kuperberg, Regev).

Quantum Security

Hidden-shift algorithms: Subexponential complexity (Kuperberg, Regev).

- Kuperberg's algorithm [Kup1] requires a subexponential number of queries, and a subexponential number of operations on a subexponential number of qubits.

Quantum Security

Hidden-shift algorithms: Subexponential complexity (Kuperberg, Regev).

- Kuperberg's algorithm [Kup1] requires a subexponential number of queries, and a subexponential number of operations on a subexponential number of qubits.
- Variant by Regev [Reg] uses polynomial number of qubits at the expense of time.

Quantum Security

Hidden-shift algorithms: Subexponential complexity (Kuperberg, Regev).

- Kuperberg's algorithm [Kup1] requires a subexponential number of queries, and a subexponential number of operations on a subexponential number of qubits.
- Variant by Regev [Reg] uses polynomial number of qubits at the expense of time.
- Kuperberg later [Kup2] gave more trade-off options for quantum and classical memory vs. time.

Quantum Security

Hidden-shift algorithms: Subexponential complexity (Kuperberg, Regev).

- Kuperberg's algorithm [Kup1] requires a subexponential number of queries, and a subexponential number of operations on a subexponential number of qubits.
- Variant by Regev [Reg] uses polynomial number of qubits at the expense of time.
- Kuperberg later [Kup2] gave more trade-off options for quantum and classical memory vs. time.
- Childs-Jao-Soukharev [CJS] applied Kuperberg/Regev to CRS – their attack also applies to CSIDH.

Quantum Security

Hidden-shift algorithms: Subexponential complexity (Kuperberg, Regev).

- Kuperberg's algorithm [Kup1] requires a subexponential number of queries, and a subexponential number of operations on a subexponential number of qubits.
- Variant by Regev [Reg] uses polynomial number of qubits at the expense of time.
- Kuperberg later [Kup2] gave more trade-off options for quantum and classical memory vs. time.
- Childs-Jao-Soukharev [CJS] applied Kuperberg/Regev to CRS – their attack also applies to CSIDH.
- Part of CJS attack computes many paths in superposition.

Quantum Security

- The exact cost of the Kuperberg/Regev/CJS attack is subtle – it depends on:
  - Choice of time/memory trade-off (Regev/Kuperberg)
  - Quantum evaluation of isogenies

(and much more).

---

[3] From [BLMP], using query count of [BS]. [BS] also study quantum evaluation of isogenies but their current preprint misses some costs.

Quantum Security

- The exact cost of the Kuperberg/Regev/CJS attack is subtle – it depends on:
  - Choice of time/memory trade-off (Regev/Kuperberg)
  - Quantum evaluation of isogenies

  (and much more).
- Most previous analysis focussed on asymptotics

---

[3] From [BLMP], using query count of [BS]. [BS] also study quantum evaluation of isogenies but their current preprint misses some costs.

Quantum Security

- The exact cost of the Kuperberg/Regev/CJS attack is subtle – it depends on:
  - Choice of time/memory trade-off (Regev/Kuperberg)
  - Quantum evaluation of isogenies

  (and much more).
- Most previous analysis focussed on asymptotics
- [BLMP] gives full computer-verified simulation of quantum evaluation of isogenies. Computes one query (i.e. CSIDH-512 group action) using $765325228976 \approx 0.7 \cdot 2^{40}$ nonlinear bit operations.

---

[3] From [BLMP], using query count of [BS]. [BS] also study quantum evaluation of isogenies but their current preprint misses some costs.

Quantum Security

- The exact cost of the Kuperberg/Regev/CJS attack is subtle – it depends on:
  - Choice of time/memory trade-off (Regev/Kuperberg)
  - Quantum evaluation of isogenies

  (and much more).
- Most previous analysis focussed on asymptotics
- [BLMP] gives full computer-verified simulation of quantum evaluation of isogenies. Computes one query (i.e. CSIDH-512 group action) using $765325228976 \approx 0.7 \cdot 2^{40}$ nonlinear bit operations.
- For fastest variant of Kuperberg (uses billions of qubits), total cost of CSIDH-512 attack is about 2^{81} qubit operations.[3]

---

[3] From [BLMP], using query count of [BS]. [BS] also study quantum evaluation of isogenies but their current preprint misses some costs.

Parameters

CSIDH-log p	intended NIST level	public key size	private key size	time (full exchange)	cycles (full exchange)	stack memory	classical security
CSIDH-512	1	64 b	32 b	65 ms	212e6	4368 b	128
CSIDH-1024	3	128 b	64 b				256
CSIDH-1792	5	224 b	112 b				448

CSIDH vs SIDH?

Apart from mathematical background, SIDH and CSIDH actually have very little in common, and are likely to be useful for different applications.

Here is a comparison for (conjectured) NIST level 1:

	CSIDH	SIDH
Speed (NIST 1)	65ms (can be improved)	$\approx$ 10ms[4]
Public key size (NIST 1)	64B	378B
Key compression (speed)		$\approx$ 15ms
Key compression (size)		222B
Constant-time slowdown	$\approx \times 3$ (can be improved)	$\approx \times 1$
Submitted to NIST	no	yes
Maturity	10 months	8 years
Best classical attack	$p^{1/4}$	$p^{1/4}$
Best quantum attack	$L_p[1/2]$	$p^{1/4}$
Key size scales	quadratically	linearly
Security assumption	isogeny walk problem	ad hoc
Non-interactive key exchange	yes	unbearably slow
Signatures (classical)	unbearably slow	seconds
Signatures (quantum)	seconds	still seconds?

---

[4] This is a very conservative estimate!

Work in progress & future work

- Fast and constant-time implementation. (For ideas on constant-time optimization, see [BLMP], [MCR]).

Work in progress & future work

- Fast and constant-time implementation. (For ideas on constant-time optimization, see [BLMP], [MCR]).
- Hardware implementation.

Work in progress & future work

- Fast and constant-time implementation. (For ideas on constant-time optimization, see [BLMP], [MCR]).
- Hardware implementation.
- More applications.

Work in progress & future work

- Fast and constant-time implementation. (For ideas on constant-time optimization, see [BLMP], [MCR]).
- Hardware implementation.
- More applications.
- [Your paper here!]

Thank you!

References

Mentioned in this talk:

BLMP Bernstein, Lange, Martindale, and Panny:
Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies
`https://quantum.isogeny.org`

BS Bonnetain, Schrottenloher:
Quantum Security Analysis of CSIDH and Ordinary Isogeny-based Schemes
`https://ia.cr/2018/537`

CLMPR Castryck, Lange, Martindale, Panny, Renes:
CSIDH: An Efficient Post-Quantum Commutative Group Action
`https://ia.cr/2018/383`

CJS Childs, Jao, and Soukharev:
Constructing elliptic curve isogenies in quantum subexponential time
`https://arxiv.org/abs/1012.4019`

DG De Feo, Galbraith:
SeaSign: Compact isogeny signatures from class group actions
`https://ia.cr/2018/824`

DKS De Feo, Kieffer, Smith:
Towards practical key exchange from ordinary isogeny graphs
`https://ia.cr/2018/485`

References

Mentioned in this talk (contd.):

DOPS	Delpech de Saint Guilhem, Orsini, Petit, and Smart: *Secure Oblivious Transfer from Semi-Commutative Masking* `https://ia.cr/2018/648`
FTY	Fujioka, Takashima, and Yoneyama: *One-Round Authenticated Group Key Exchange from Isogenies* `https://eprint.iacr.org/2018/1033`
MCR	Meyer, Campos, Reith: *On Lions and Elligators: An efficient constant-time implementation of CSIDH* `https://eprint.iacr.org/2018/1198`
Kup1	Kuperberg: *A subexponential-time quantum algorithm for the dihedral hidden subgroup problem* `https://arxiv.org/abs/quant-ph/0302112`
Kup2	Kuperberg: *Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem* `https://arxiv.org/abs/1112.3333`
Reg	Regev: *A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space* `https://arxiv.org/abs/quant-ph/0406151`

References

Further reading:

BIJ Biasse, Iezzi, Jacobson:
A note on the security of CSIDH
`https://arxiv.org/pdf/1806.03656`

DPV Decru, Panny, and Vercauteren:
Faster SeaSign signatures through improved rejection sampling
`https://eprint.iacr.org/2018/1109`

JLLR Jao, LeGrow, Leonardi, Ruiz-Lopez:
A polynomial quantum space attack on CRS and CSIDH
(MathCrypt 2018)

MR Meyer, Reith:
A faster way to the CSIDH
`https://ia.cr/2018/782`

Credits: thanks to Lorenz Panny for many of these slides, including all of the beautiful tikz pictures.