Graphs and cycles in curve-based collision walks

Monika Trimoska

with Sebastiano Boscardin, Jolijn Cottaar

Mathematics of Post-Quantum Cryptography

Raitenhaslach, September 7, 2026

TU/e

Three problems

The ECDLP problem

Input: Two points $P, Q \in E(\mathbb{F}_q)$ with $Q \in \langle P \rangle$.
Question: Find an integer x such that $xP = Q$.

The Isogeny Pathfinding Problem over $\mathbb{F}_{p^2}$

Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_{p^2}$ on an ℓ-isogeny graph.
Question: Find an isogeny φ such that $\varphi : E_0 \to E_1$.

The Isogeny Pathfinding Problem over $\mathbb{F}_p$

Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_p$.
Question: Find a vector $(e_1, \ldots, e_n) \in \mathbb{Z}^n$ such that $\varphi : E_0 \to E_1$ is an isogeny of degree $\prod \ell_i^{e_i}$.

Elliptic curves

What is an elliptic curve?

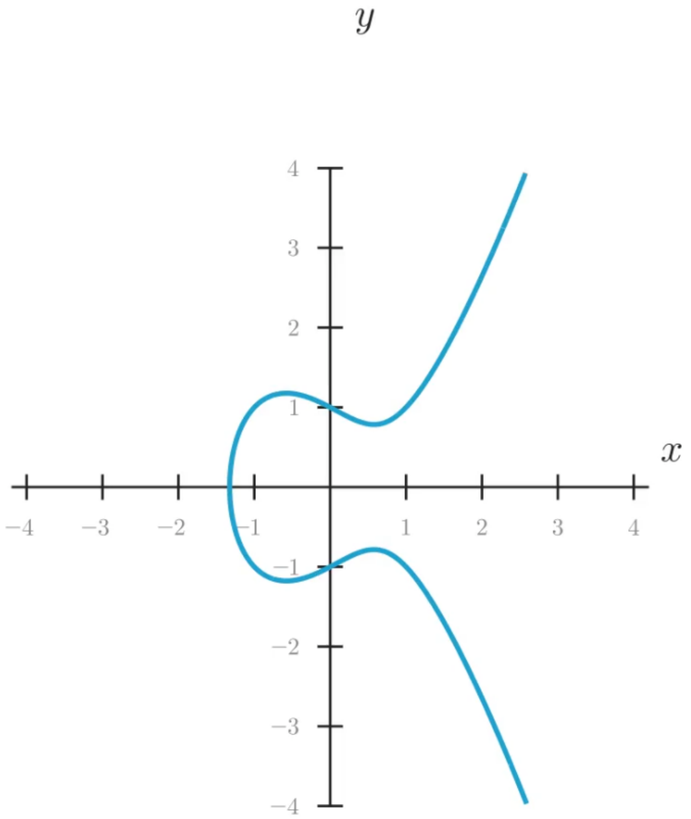
An elliptic curve is an algebraic curve that admits an affine equation of the form

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,$$

(general form of a Weierstrass curve)

with $a_i \in k$, where k is the field where the point is defined.

Example. $y^2 = x^3 - x + 1$



→ A point on E means that the point (x, y) satisfies the curve equation.

Elliptic curves over $\mathbb{F}_q$

In cryptography, we use elliptic curves over finite fields $\mathbb{F}_q$, $q = p^k$ (but we draw the figures over $\mathbb{R}$ because it's nicer).

→ We denote by $E(k)$ the set of k-rational points on E.

↳ defined over k

Hasse bound

$$\#E(\mathbb{F}_q) \in [q + 1 - 2\sqrt{(q)}, q + 1 + 2\sqrt{(q)}]$$

Elliptic curves in cryptography

- Short Weierstrass form

$$y^2 = x^3 + c_4 x + c_6$$

- The set of points on E with the addition law form a group.

- The group law is constructed geometrically.

Elliptic curves in cryptography

- Short Weierstrass form

$$y^2 = x^3 + c_4 x + c_6$$

- The set of points on E with the addition law form a group.

- The group law is constructed geometrically.

The ECDLP problem

Input: Two points $P, Q \in E(\mathbb{F}_q)$ with $Q \in \langle P \rangle$.
Question: Find an integer x such that $xP = Q$.

Elliptic curves in cryptography

A curve is

- non-singular (or smooth) if it does not have a singular point.

Jacobi criterion: a point on E is singular if (x, y) also satisfies the two partial derivatives $2y + a_1x + a_3 = 0$ and $a_1y = 3x^2 + 2a_2x + a_4$.

Elliptic curves in cryptography

A curve is

- non-singular (or smooth) if it does not have a singular point.

  Jacobi criterion: a point on E is singular if (x, y) also satisfies the two partial derivatives $2y + a_1x + a_3 = 0$ and $a_1y = 3x^2 + 2a_2x + a_4$.

- supersingular (also non-singular) if and only if $\#E(\mathbb{F}_p) = p + 1$ (for $p > 3$).

  equivalently: iff $E[p] = \{\infty\}$

  $E[n] = \{P \in E(\overline{\mathbb{F}_p}) \mid nP = \infty\}$ (the n-torsion group)

Elliptic curves in cryptography

A curve is

- non-singular (or smooth) if it does not have a singular point.

  Jacobi criterion: a point on E is singular if (x, y) also satisfies the two partial derivatives $2y + a_1x + a_3 = 0$ and $a_1y = 3x^2 + 2a_2x + a_4$.

- supersingular (also non-singular) if and only if $\#E(\mathbb{F}_p) = p + 1$ (for $p > 3$).

  equivalently: iff $E[p] = \{\infty\}$

  $E[n] = \{P \in E(\overline{\mathbb{F}_p}) \mid nP = \infty\}$ (the n-torsion group)

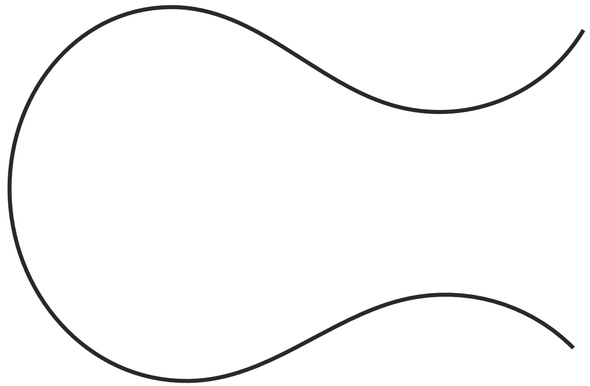
Supersingular curves and cyclic groups

- $E(\mathbb{F}_p) \cong \mathbb{Z}/(p+1)$
- $E(\mathbb{F}_{p^2}) \cong \mathbb{Z}/(p+1) \times \mathbb{Z}/(p+1)$

Isogenies

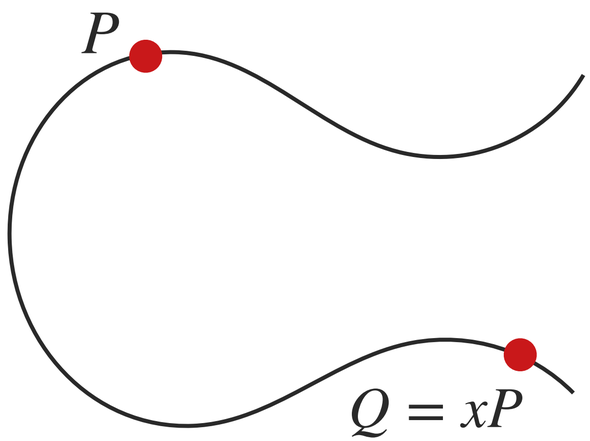
A familiar isogeny

The multiplication-by-d map



A familiar isogeny

The multiplication-by-d map

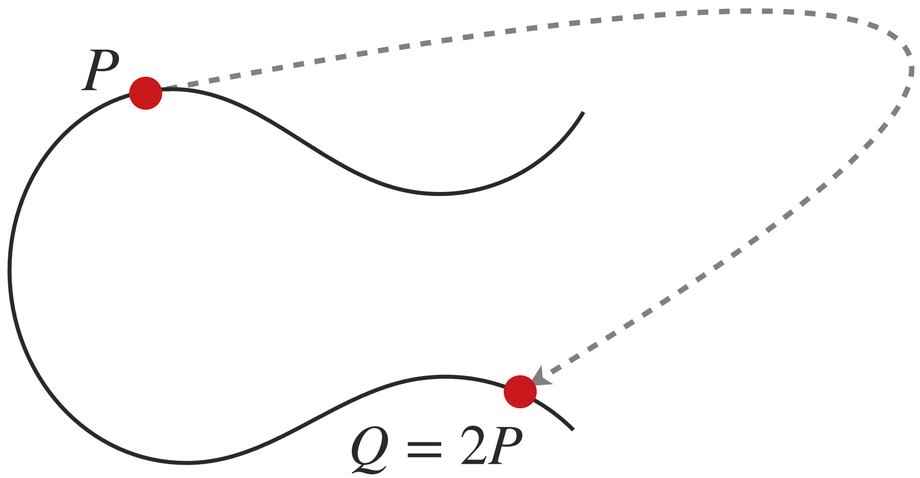


A familiar isogeny

The multiplication-by-d map

$$(x, y) \mapsto (\lambda^2 - 2x, \lambda x + y),$$

$$\lambda = \frac{3x^2 + a}{2y}$$



Isomorphisms

An isomorphism is a map between elliptic curves that is defined everywhere,
i.e., that is given by polynomials in x and y.

Isomorphisms

An isomorphism is a map between elliptic curves that is defined everywhere,
i.e., that is given by polynomials in x and y.

it is a degree-1 isogeny (will make sense on the next slide)
(can be viewed as a change of coordinates)

Isomorphisms

An isomorphism is a map between elliptic curves that is defined everywhere,
i.e., that is given by polynomials in x and y.

it is a degree-1 isogeny (will make sense on the next slide)
(can be viewed as a change of coordinates)

j-invariant

- An invariant under isomorphisms.
- For a curve in short Weierstrass form $y^2 = x^3 + c_4 x + c_6$, we have

$$j = 1728 \cdot 4c_4^3/(4c_4^3 + 27c_6^2)$$

Isogenies

An isogeny φ of elliptic curves is a non-zero map $E \to E'$ that is

- given by rational functions
- that is a group homomorphism

Isogenies

An isogeny φ of elliptic curves is a non-zero map $E \to E'$ that is

- given by rational functions
- that is a group homomorphism

$\frac{f(x,y)}{g(x,y)}$, where f, g are polynomials

Isogenies

An isogeny φ of elliptic curves is a non-zero map $E \to E'$ that is

- given by rational functions
- that is a group homomorphism

$$\varphi(P + Q) = \varphi(P) + \varphi(Q)$$

$$\frac{f(x, y)}{g(x, y)}, \text{ where } f, g \text{ are polynomials}$$

Isogenies

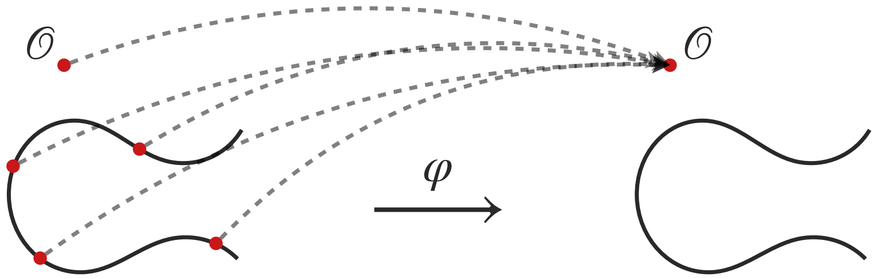
An isogeny φ of elliptic curves is a non-zero map $E \to E'$ that is

- given by rational functions
- that is a group homomorphism

$$\varphi(P+Q) = \varphi(P) + \varphi(Q)$$

$$\frac{f(x,y)}{g(x,y)}, \text{ where } f, g \text{ are polynomials}$$

→ An isogeny is uniquely defined by its kernel: $\{P \in E \,|\, \varphi(P) = \mathcal{O}_{E'}\}$.



Isogenies

An isogeny φ of elliptic curves is a non-zero map $E \to E'$ that is

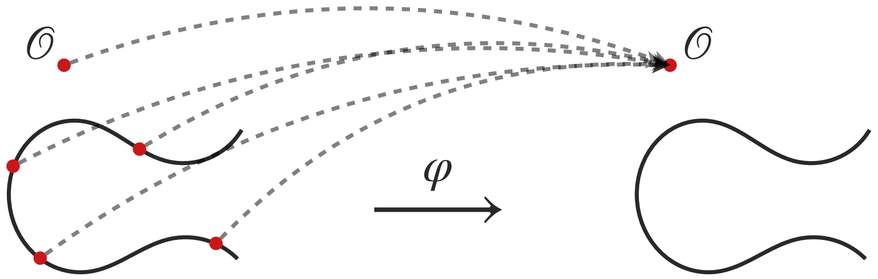
- given by rational functions
- that is a group homomorphism

$\varphi(P+Q) = \varphi(P) + \varphi(Q)$

$\frac{f(x,y)}{g(x,y)}$, where f, g are polynomials

→ An isogeny is uniquely defined by its kernel: $\{P \in E \,|\, \varphi(P) = \mathcal{O}_{E'}\}$.

→ The degree of a (separable) isogeny is the size of its kernel.



Isogenies

Example.

$$(x, y) \mapsto \left(\frac{x^3 - 4x^2 + 30x - 12}{(x-2)^2}, \frac{x^3 - 6x^2 - 14x + 35}{(x-2)^3} \cdot y\right)$$

defines a degree-3 isogeny of the elliptic curves

$$\{y^2 = x^3 + x\} \to \{y^2 = x^3 - 3x + 3\}$$

over $\mathbb{F}_{71}$. Its kernel is $\{(2,9), (2, -9), \mathcal{O}\}$.

Isogenies

Example.

$$(x, y) \mapsto \left(\frac{x^3 - 4x^2 + 30x - 12}{(x-2)^2}, \frac{x^3 - 6x^2 - 14x + 35}{(x-2)^3} \cdot y\right)$$

defines a degree-3 isogeny of the elliptic curves

$$\{y^2 = x^3 + x\} \to \{y^2 = x^3 - 3x + 3\}$$

over $\mathbb{F}_{71}$. Its kernel is $\{(2,9), (2,-9), \mathcal{O}\}$.

Isogenies

Example.

$$(x, y) \mapsto \left(\frac{x^3 - 4x^2 + 30x - 12}{(x-2)^2}, \frac{x^3 - 6x^2 - 14x + 35}{(x-2)^3} \cdot y\right)$$

defines a degree-3 isogeny of the elliptic curves

$$\{y^2 = x^3 + x\} \to \{y^2 = x^3 - 3x + 3\}$$

over $\mathbb{F}_{71}$. Its kernel is $\{(2,9), (2,-9), \mathcal{O}\}$.

ℓ-isogeny:

- $x \to \frac{f(x)}{g(x)}$, with $\deg(f) = \ell, \deg(g) = \ell - 1$
- $y \to \ldots$

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

Compute an ℓ-isogeny

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

Obtain a kernel of an ℓ-isogeny

↳ Compute an ℓ-isogeny

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

Obtain a subgroup of order ℓ

↳ Obtain a kernel of an ℓ-isogeny

↳ Compute an ℓ-isogeny

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

Take the cyclic group generated by P

↳ Obtain a subgroup of order ℓ

↳ Obtain a kernel of an ℓ-isogeny

↳ Compute an ℓ-isogeny

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

Find a point P on E of order ℓ

↳ Take the cyclic group generated by P

↳ Obtain a subgroup of order ℓ

↳ Obtain a kernel of an ℓ-isogeny

↳ Compute an ℓ-isogeny

Computing isogenies

*We consider only supersingular curves from now on.

→ **Goal**: Compute an ℓ-isogeny from E.

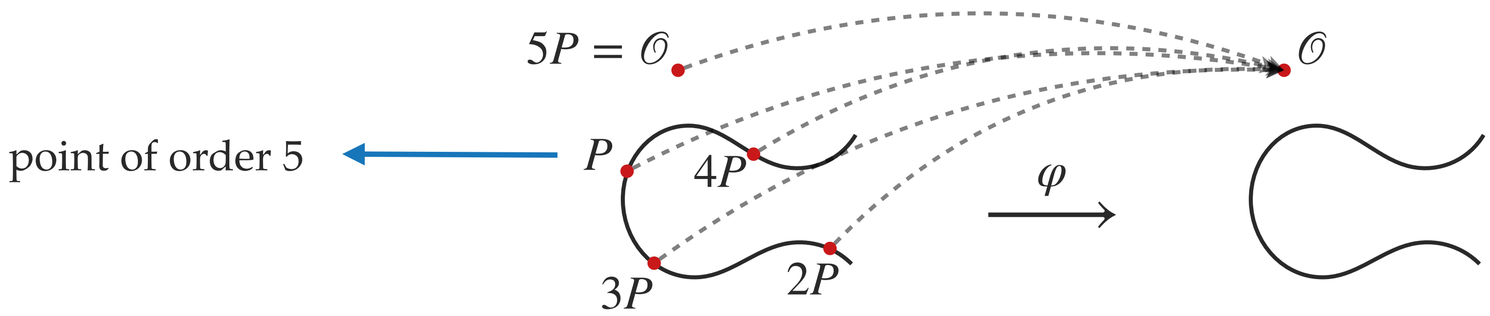
Find a point P on E of order ℓ

↳ Take the cyclic group generated by P

↳ Obtain a subgroup of order ℓ

↳ Obtain a kernel of an ℓ-isogeny

↳ Compute an ℓ-isogeny



Vélu's formulas

- For any finite subgroup G of E, there exists a unique (up to isomorphism) separable isogeny $\varphi_G : E \to E'$ with kernel G.

Vélu's formulas

→ For any finite subgroup G of E, there exists a unique (up to isomorphism) separable isogeny $\varphi_G : E \to E'$ with kernel G.

Vélu '71

- Formulas for computing E' and evaluating φ_G at a point.
- Complexity: $\Theta(\#G)$ → only suitable for small degrees.

Vélu's formulas

→ For any finite subgroup G of E, there exists a unique (up to isomorphism) separable isogeny $\varphi_G : E \to E'$ with kernel G.

Vélu '71

- Formulas for computing E' and evaluating φ_G at a point.
- Complexity: $\Theta(\#G)$ → only suitable for small degrees.

Let P have prime order ℓ on E_A.
For $1 \leq i < \ell$ let x_i be the x-coordinate of iP.
Let

$$\tau = \prod_{i=1}^{\ell-1} x_i, \quad \sigma = \sum_{i=1}^{\ell-1} \left(x_i - \frac{1}{x_i} \right), \quad f(x) = x \prod_{i=1}^{\ell-1} \frac{x x_i - 1}{x - x_i}.$$

Then the ℓ-isogeny with kernel $\langle P \rangle$ is given by

$$\varphi_\ell : E_A \to E_B, (x, y) \mapsto (f(x), c_0 y f'(x))$$

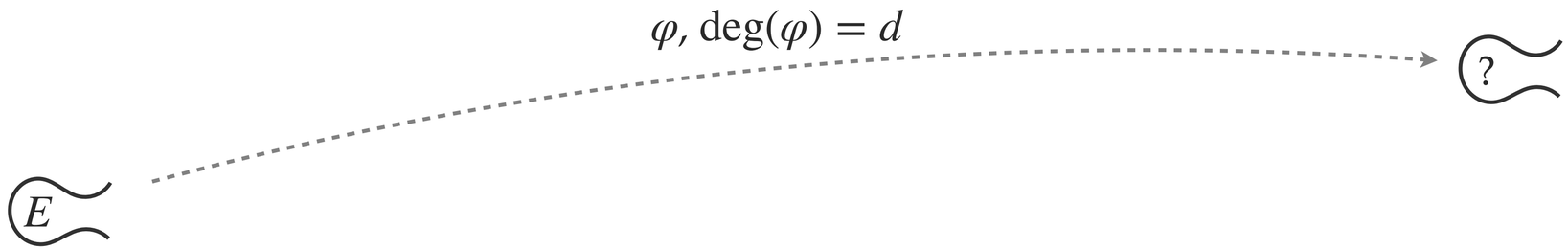
where $B = \tau(A - 3\sigma)$, and $c_0^2 = \tau$.

Composing isogenies

→ **Goal**: Compute a d-isogeny from E.

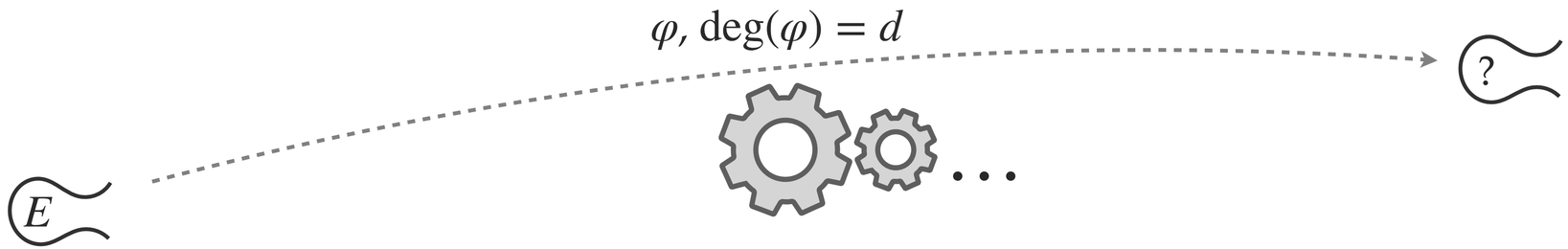
Composing isogenies

Goal: Compute a d-isogeny from E.



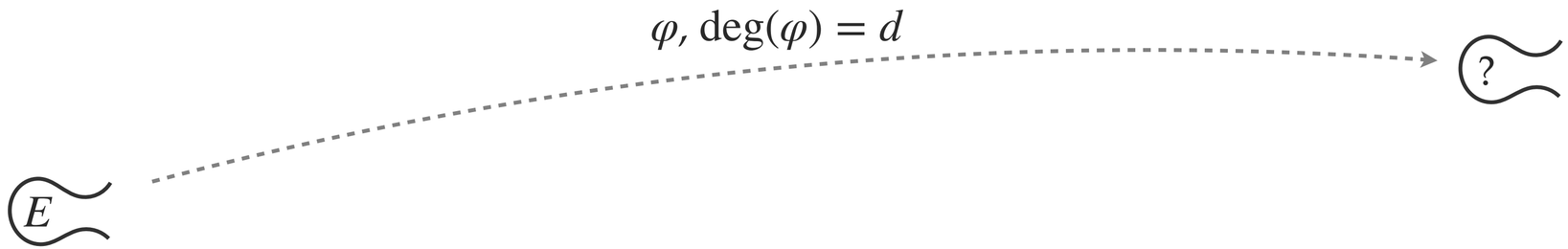
Composing isogenies

Goal: Compute a d-isogeny from E.



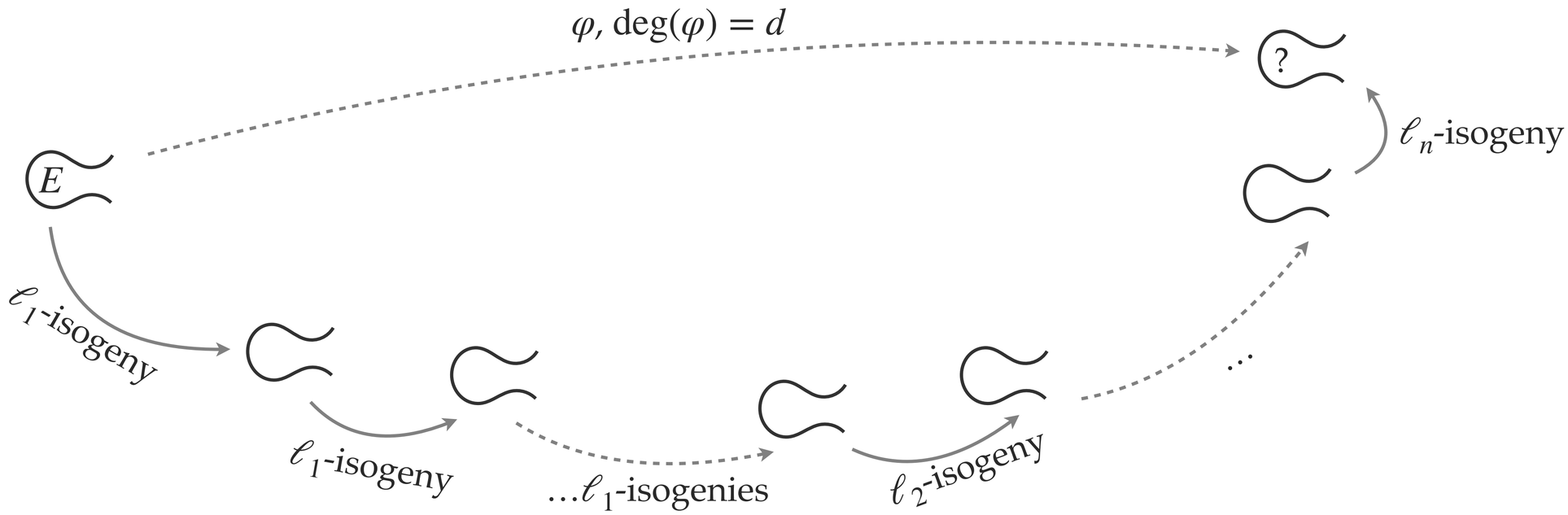
Composing isogenies

Goal: Compute a d-isogeny from E, with d a smooth integer ($d = \ell_1^{e_1} \cdot \ell_2^{e_2} \cdots \ell_n^{e_n}$).



Composing isogenies

Goal: Compute a d-isogeny from E, with d a smooth integer ($d = \ell_1^{e_1} \cdot \ell_2^{e_2} \cdots \ell_n^{e_n}$).



Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.

Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.
- The composition $\hat{\varphi} \circ \varphi$ is the multiplication-by-d map on E and $\varphi \circ \hat{\varphi}$ the multiplication-by-d map on E', where $d = \deg(\varphi) = \deg(\hat{\varphi})$.

Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.
- The composition $\hat{\varphi} \circ \varphi$ is the multiplication-by-d map on E and $\varphi \circ \hat{\varphi}$ the multiplication-by-d map on E', where $d = \deg(\varphi) = \deg(\hat{\varphi})$.

The multiplication-by-d map

- The multiplication-by-d map $[d] : E \to E$ is a degree-d^2 isogeny from E to E.

Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.
- The composition $\hat{\varphi} \circ \varphi$ is the multiplication-by-d map on E and $\varphi \circ \hat{\varphi}$ the multiplication-by-d map on E', where $d = \deg(\varphi) = \deg(\hat{\varphi})$.

The multiplication-by-d map

- The multiplication-by-d map $[d] : E \to E$ is a degree-d^2 isogeny from E to E.

  → It is an endomorphism.

Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.
- The composition $\hat{\varphi} \circ \varphi$ is the multiplication-by-d map on E and $\varphi \circ \hat{\varphi}$ the multiplication-by-d map on E', where $d = \deg(\varphi) = \deg(\hat{\varphi})$.

The multiplication-by-d map

- The multiplication-by-d map $[d] : E \to E$ is a degree-d^2 isogeny from E to E.
  - It is an endomorphism.
- Its kernel is $E[d] \cong \mathbb{Z}/d \times \mathbb{Z}/d$.

Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.
- The composition $\hat{\varphi} \circ \varphi$ is the multiplication-by-d map on E and $\varphi \circ \hat{\varphi}$ the multiplication-by-d map on E', where $d = \deg(\varphi) = \deg(\hat{\varphi})$.

The multiplication-by-d map

- The multiplication-by-d map $[d] : E \to E$ is a degree-d^2 isogeny from E to E.
- Its kernel is $E[d] \cong \mathbb{Z}/d \times \mathbb{Z}/d$.

It is an endomorphism.

End(E)

- An endomorphism is an isogeny from a curve E to itself.

Endomorphism rings

Dual isogeny

- For isogeny $\varphi : E \to E'$ there exists a unique dual isogeny $\hat{\varphi} : E' \to E$.
- The composition $\hat{\varphi} \circ \varphi$ is the multiplication-by-d map on E and $\varphi \circ \hat{\varphi}$ the multiplication-by-d map on E', where $d = \deg(\varphi) = \deg(\hat{\varphi})$.

The multiplication-by-d map

- The multiplication-by-d map $[d] : E \to E$ is a degree-d^2 isogeny from E to E.
- Its kernel is $E[d] \cong \mathbb{Z}/d \times \mathbb{Z}/d$.

It is an endomorphism.

$\mathrm{End}(E)$

- An endomorphism is an isogeny from a curve E to itself.
- The set of endomorphisms forms a ring $\mathrm{End}(E)$ under $+$ and $\circ$.

Hard problems and reductions

The isogeny path problem

Input: Two supersingular curves E and E'.
Question: Find an isogeny φ from E to E'.

Hard problems and reductions

The isogeny path problem

Input: Two supersingular curves E and E'.
Question: Find an isogeny φ from E to E'.

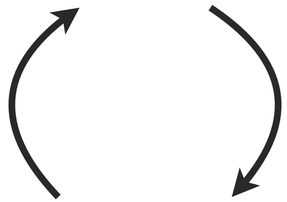
The EndRing problem

Input: A super singular curve E.
Question: Find a basis of $\mathrm{End}(E)$.

Hard problems and reductions

The isogeny path problem

Input: Two supersingular curves E and E'.
Question: Find an isogeny φ from E to E'.



The EndRing problem

Input: A super singular curve E.
Question: Find a basis of $\mathrm{End}(E)$.

Isogeny graphs

Supersingular curves and cyclic groups (recall)

- $E(\mathbb{F}_p) \cong \mathbb{Z}/(p+1)$
- $E(\mathbb{F}_{p^2}) \cong \mathbb{Z}/(p+1) \times \mathbb{Z}/(p+1)$

Let ℓ be a prime factor of p.

There are exactly $(\ell+1)$ $\mathbb{F}_{p^2}$-rational ℓ-isogenies from each E.

There is exactly one $\mathbb{F}_p$-rational ℓ-isogeny from each E.

Isogeny graphs

- Vertices are (isomorphism classes) of supersingular elliptic curves.
- Edges are prime-degree isogenies between them.

The vOW framework

Three problems for van Oorschot and Wiener

The ECDLP problem

Input: Two points $P, Q \in E(\mathbb{F}_q)$ with $Q \in \langle P \rangle$.
Question: Find an integer x such that $xP = Q$.

The Isogeny Pathfinding Problem over $\mathbb{F}_{p^2}$

Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_{p^2}$ on an ℓ-isogeny graph.
Question: Find an isogeny φ such that $\varphi : E_0 \to E_1$.

The Isogeny Pathfinding Problem over $\mathbb{F}_p$

Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_p$.
Question: Find a vector $(e_1, \ldots, e_n) \in \mathbb{Z}^n$ such that $\varphi : E_0 \to E_1$ is an isogeny of degree $\prod \ell_i^{e_i}$.

The multi-user setting

→ We have many users, say L, with the same system parameters.

Example. (ECDLP)

User L_j has public key Q_j, secret key x_j such that $Q_j = x_j P$.

The multi-user setting

→ We have many users, say L, with the same system parameters.

Example. (ECDLP)

User L_j has $\begin{matrix}\text{public key } Q_j,\\ \text{secret key } x_j\end{matrix}$ such that $Q_j = x_j P$.

Goal. If we can recover the key of one user in $\mathcal{O}($😺$)$,

we recover the key of all users in $< \mathcal{O}(L$ 😺$)$.

Another three problems for van Oorschot and Wiener

The MU ECDLP problem

Input: $L+1$ points $P, Q_1, \ldots, Q_L \in E(\mathbb{F}_q)$ with $Q_j \in \langle P \rangle$.
Question: Find integers $a_1, \ldots, a_L$ such that $Q_j = a_j P$ for $j \in \{1,\ldots,L\}$.

The MU Isogeny Pathfinding Problem over $\mathbb{F}_{p^2}$

Input: $L+1$ supersingular elliptic curves $E_0, E_1, \ldots, E_L$ defined over $\mathbb{F}_{p^2}$ on an ℓ-isogeny graph.
Question: Find L isogenies $\varphi_1, \ldots, \varphi_L$ such that $\varphi_j : E_0 \to E_j$.

The MU Isogeny Pathfinding Problem over $\mathbb{F}_p$

Input: $L+1$ supersingular elliptic curves $E_0, E_1, \ldots, E_L$ defined over $\mathbb{F}_p$.
Question: Find vectors $(e_{1,1}\ldots, e_{1,n}), \ldots, (e_{L,1}\ldots, e_{L,n}) \in \mathbb{Z}^n$ such that $\varphi_j : E_0 \to E_j$ is an isogeny of degree $\prod \ell_i^{e_{j,i}}$.

vOW for ECDLP

Collision search

What is a collision? Why does a collision help us solve the (EC)DLP?

Collision search

What is a collision? Why does a collision help us solve the (EC)DLP?

Having two different linear combinations of a random point $R \in E(\mathbb{F}_q)$

$$R = aP + bQ$$
$$R = a'P + b'Q$$

Collision search

What is a collision? Why does a collision help us solve the (EC)DLP?

Having two different linear combinations of a random point $R \in E(\mathbb{F}_q)$

$$R = aP + bQ$$
$$R = a'P + b'Q$$

we infer that

$$aP + bQ = a'P + b'Q$$
$$(a - a')P = (b' - b)xP$$

Collision search

What is a collision? Why does a collision help us solve the (EC)DLP?

Having two different linear combinations of a random point $R \in E(\mathbb{F}_q)$

$$R = aP + bQ$$
$$R = a'P + b'Q$$

we infer that

$$aP + bQ = a'P + b'Q$$
$$(a - a')P = (b' - b)xP$$

and we compute

$$x = \frac{a - a'}{b' - b} \pmod{N}.$$

Collision search

Collision

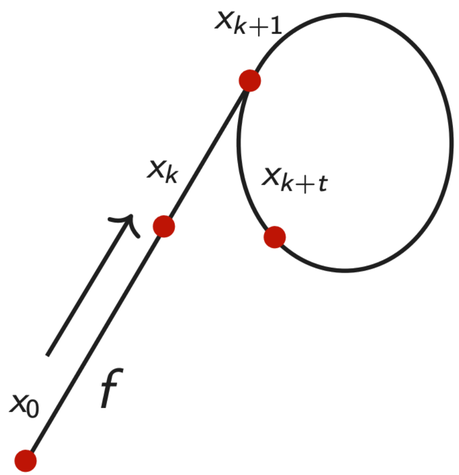
Given a random map $f : S \to S$ on a finite set S of cardinality N, we call collision any pair R, R' of elements in S such that $f(R) = f(R')$.

Collision search

Collision

Given a random map $f : S \to S$ on a finite set S of cardinality N, we call collision any pair R, R' of elements in S such that $f(R) = f(R')$.

Pollard's Rho method

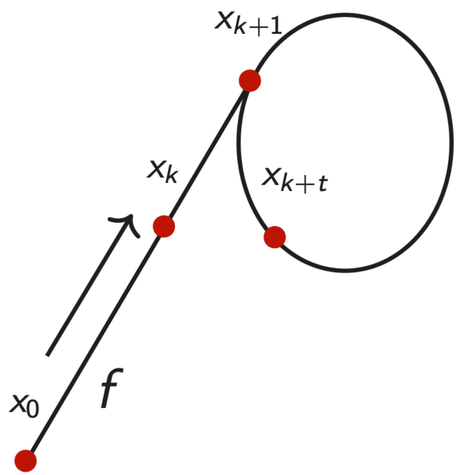


Collision search

Collision

Given a random map $f : S \to S$ on a finite set S of cardinality N, we call collision any pair R, R' of elements in S such that $f(R) = f(R')$.

Pollard's Rho method



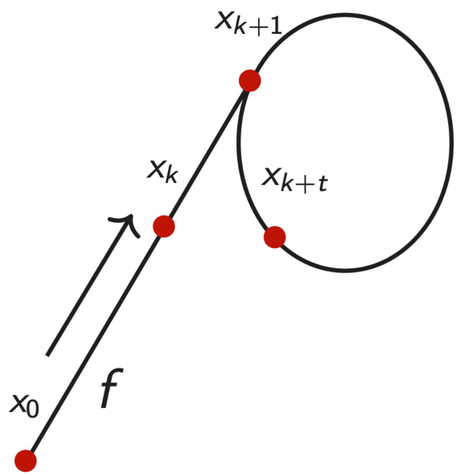
- Ideally, f is a random mapping.

Collision search

Collision

Given a random map $f : S \to S$ on a finite set S of cardinality N, we call collision any pair R, R' of elements in S such that $f(R) = f(R')$.

Pollard's Rho method



- Ideally, f is a random mapping.
- Expected number of steps until the collision is found

$$\sqrt{\frac{\pi N}{2}}$$

Collision search

$$f(R) = \begin{cases} R + c_1 P + d_1 Q & \text{if } R \in S_1 \\ R + c_2 P + d_2 Q & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P + d_k Q & \text{if } R \in S_k, \end{cases}$$

Collision search

$$f(R) = \begin{cases} R + c_1 P + d_1 Q & \text{if } R \in S_1 \\ R + c_2 P + d_2 Q & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P + d_k Q & \text{if } R \in S_k, \end{cases}$$

Property of f

Input $(aP + bQ) \rightarrow$ Output $(a'P + b'Q)$.
(If the input of f is linear combination of P and Q, the output of f is also a linear combination of P and Q.)

Collision search

$$f(R) = \begin{cases} R + c_1 P + d_1 Q & \text{if } R \in S_1 \\ R + c_2 P + d_2 Q & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P + d_k Q & \text{if } R \in S_k, \end{cases}$$

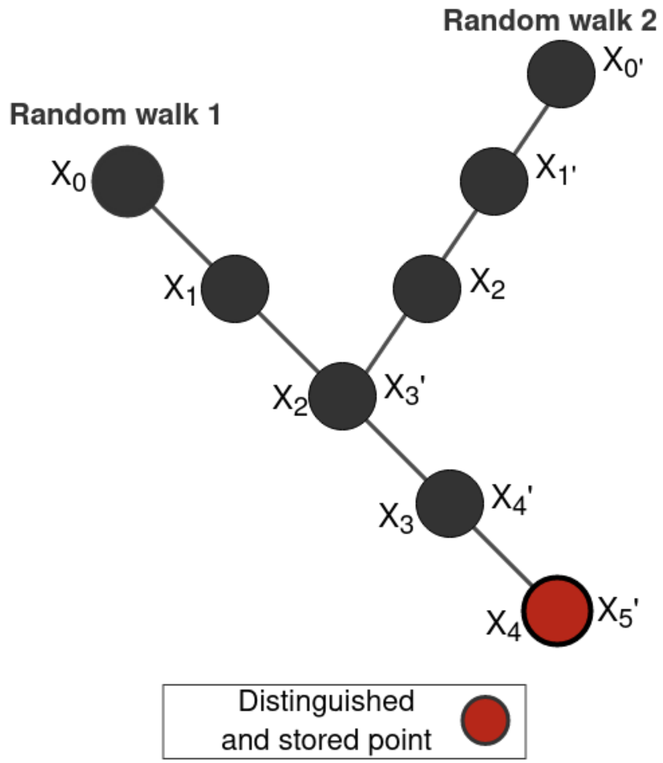
Property of f

Input $(aP + bQ) \rightarrow$ Output $(a'P + b'Q)$.
(If the input of f is linear combination of P and Q, the output of f is also a linear combination of P and Q.)

Intuitively:

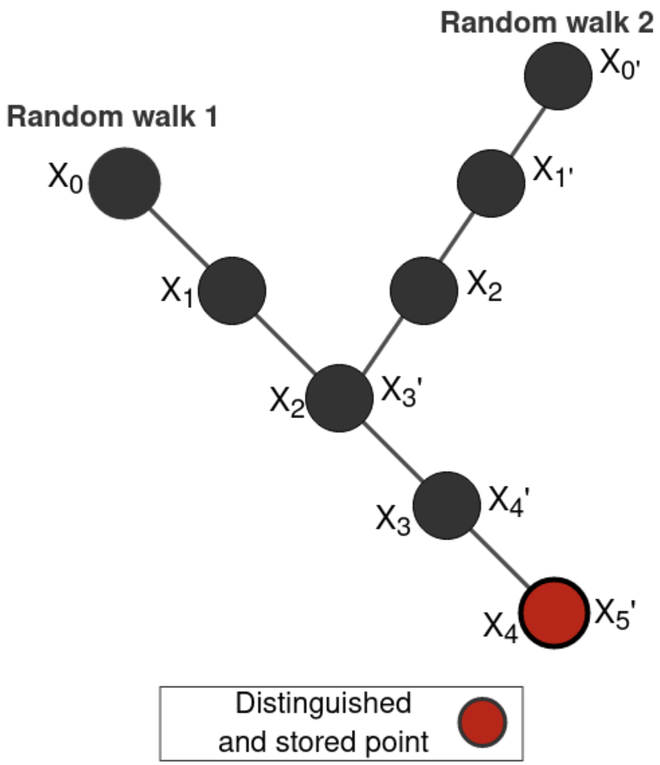
- Start from $R = aP + bQ$ for some random a and b.
- *Walk* the random walk until we find the same point twice.
  ↪ To discover the collision, we need to store *all** the points that we compute.

Parallel Collision Search



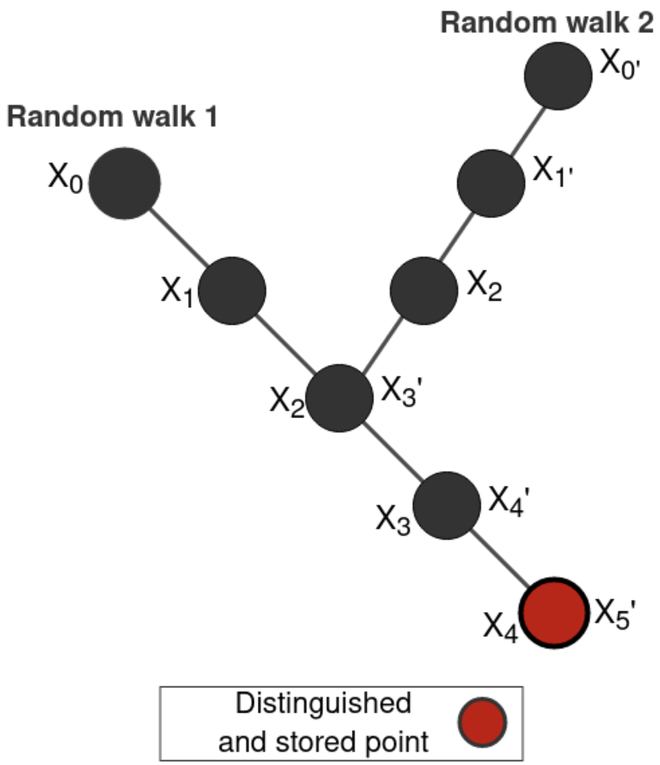
- Proposed by van Oorschot & Wiener (1996).

Parallel Collision Search



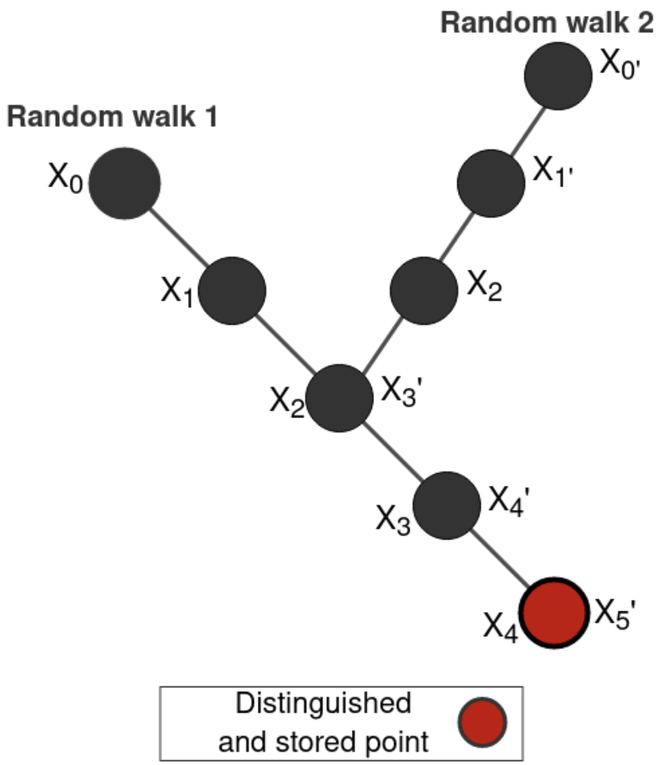
- Proposed by van Oorschot & Wiener (1996).
- Distinguished points: a set of points having an easily testable property.
  ex. The x-coordinate has 3 trailing zero bits:
  10101101000.

Parallel Collision Search



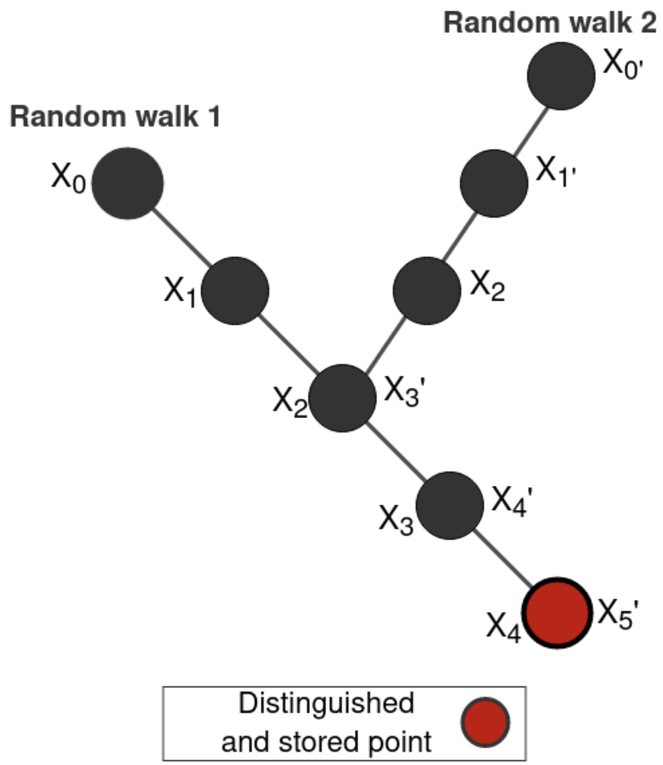
- Proposed by van Oorschot & Wiener (1996).
- Distinguished points: a set of points having an easily testable property.
  ex. The x-coordinate has 3 trailing zero bits:
  10101101000.
- Only distinguished points are stored in memory.

Parallel Collision Search



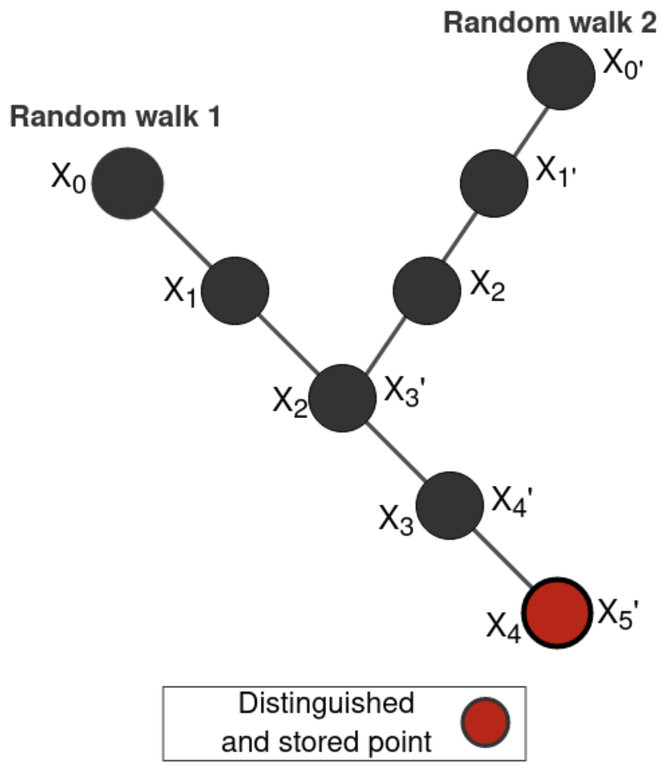
- Proposed by van Oorschot & Wiener (1996).
- Distinguished points: a set of points having an easily testable property.
  ex. The x-coordinate has 3 trailing zero bits:
  10101101000.
- Only distinguished points are stored in memory.
- θ - the proportion of distinguished points in a set S.

Parallel Collision Search



- Proposed by van Oorschot & Wiener (1996).
- Distinguished points: a set of points having an easily testable property.
  ex. The x-coordinate has 3 trailing zero bits: 10101101000.
- Only distinguished points are stored in memory.
- θ - the proportion of distinguished points in a set S.
- Complexity ? How many points do we expect to compute (store) before a collision is found ?

Parallel Collision Search



- Proposed by van Oorschot & Wiener (1996).
- Distinguished points: a set of points having an easily testable property.
  ex. The x-coordinate has 3 trailing zero bits: 10101101000.
- Only distinguished points are stored in memory.
- θ - the proportion of distinguished points in a set S.
- Complexity ? How many points do we expect to compute (store) before a collision is found ?

  ↳ The Birthday paradox

  ↳ (recall) $\sim \sqrt{N}$

Multi-user ECDLP

The multi-user setting

→ We have many users, say L, with the same system parameters.

Example. (ECDLP)

User L_j has $\begin{matrix}\text{public key } Q_j,\\ \text{secret key } x_j\end{matrix}$ such that $Q_j = x_j P$.

Goal. If we can recover the key of one user in $\mathcal{O}($😺$)$,
we recover the key of all users in $< \mathcal{O}(L$ 😺$)$.

The multi-user setting

→ We have many users, say L, with the same system parameters.

Example. (ECDLP)

User L_j has $\begin{matrix}\text{public key } Q_j,\\ \text{secret key } x_j\end{matrix}$ such that $Q_j = x_j P$.

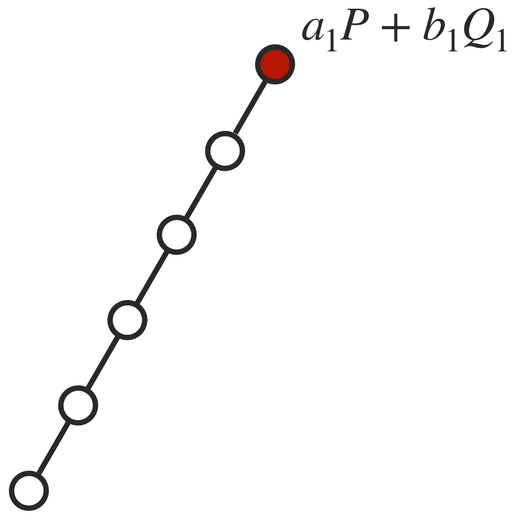
Goal. If we can recover the key of one user in $\mathcal{O}($😼$)$,
we recover the key of all users in $< \mathcal{O}(L$ 😼$)$.

Example. (ECDLP)

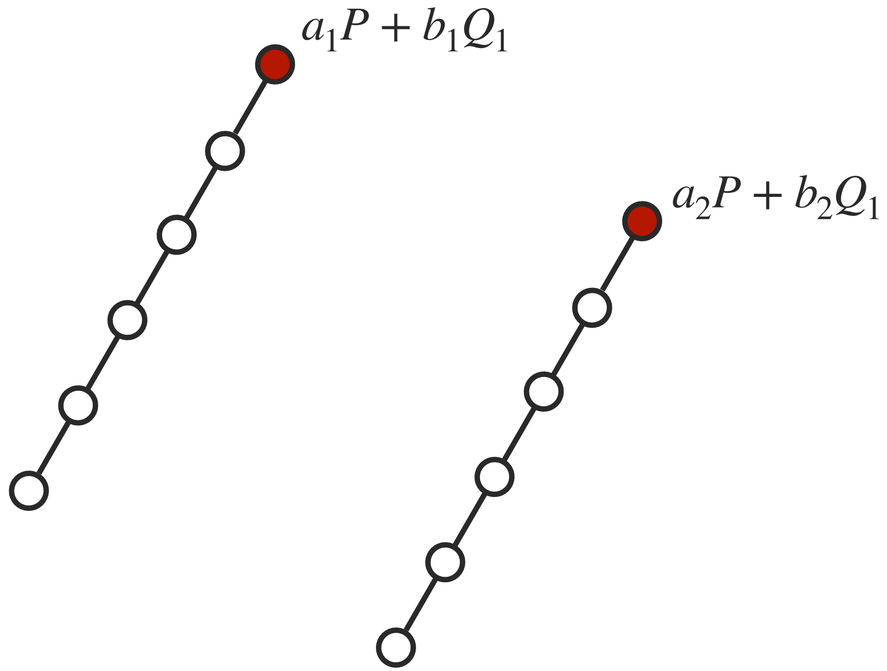
We recover the key of L users in $\mathcal{O}(\sqrt{L}$ 😼$)$.

ECDLP instances are friends

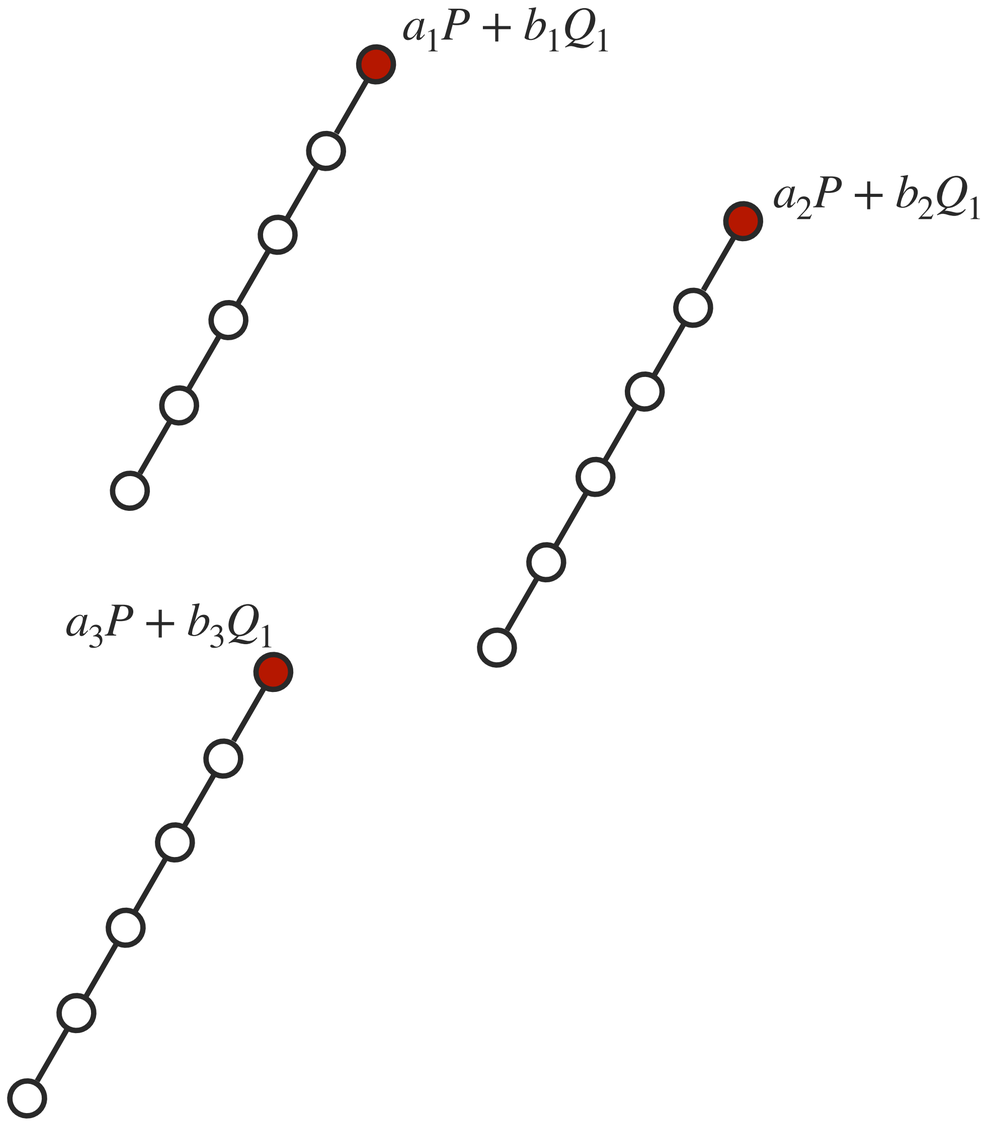
ECDLP instances are friends



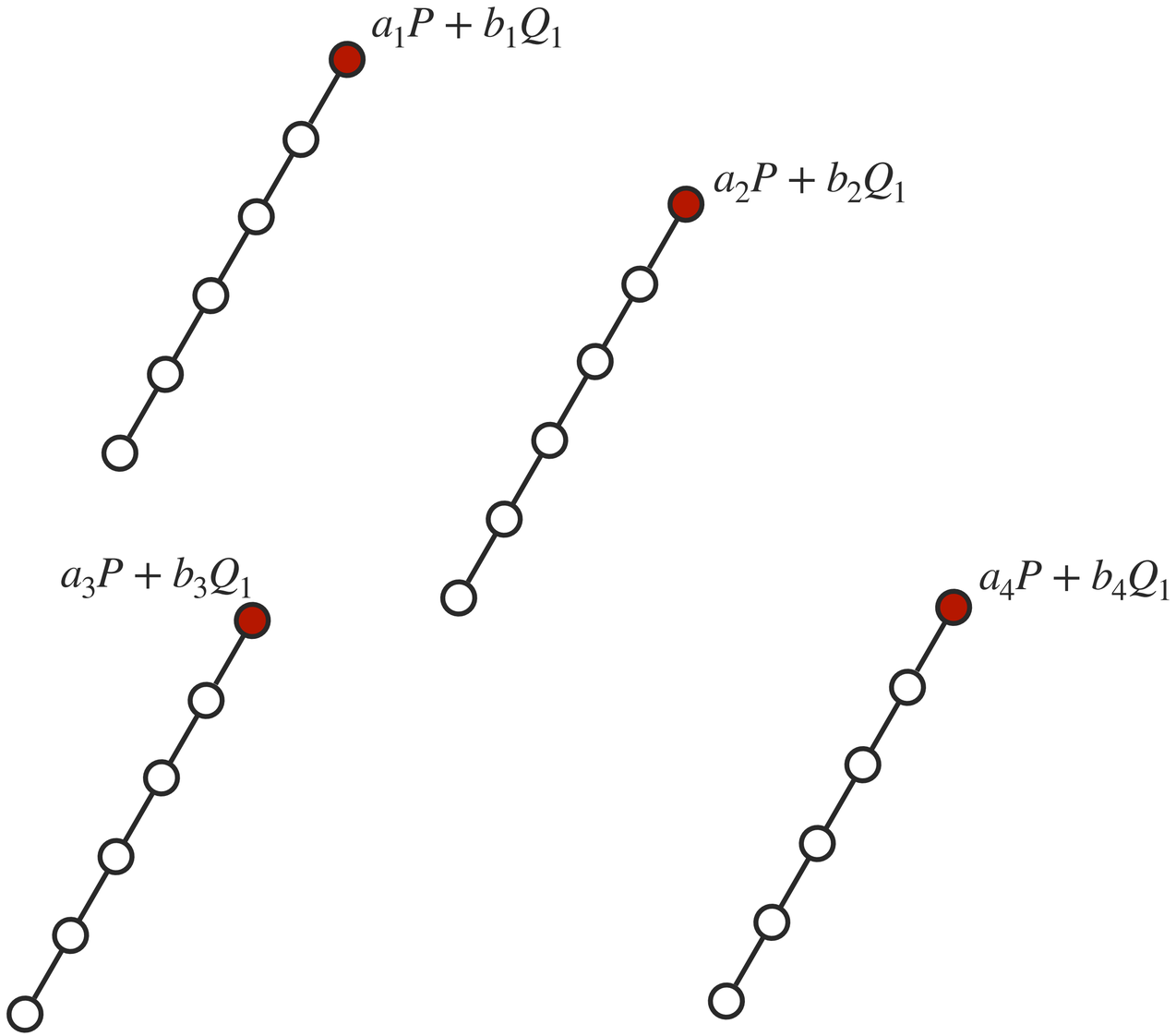
ECDLP instances are friends



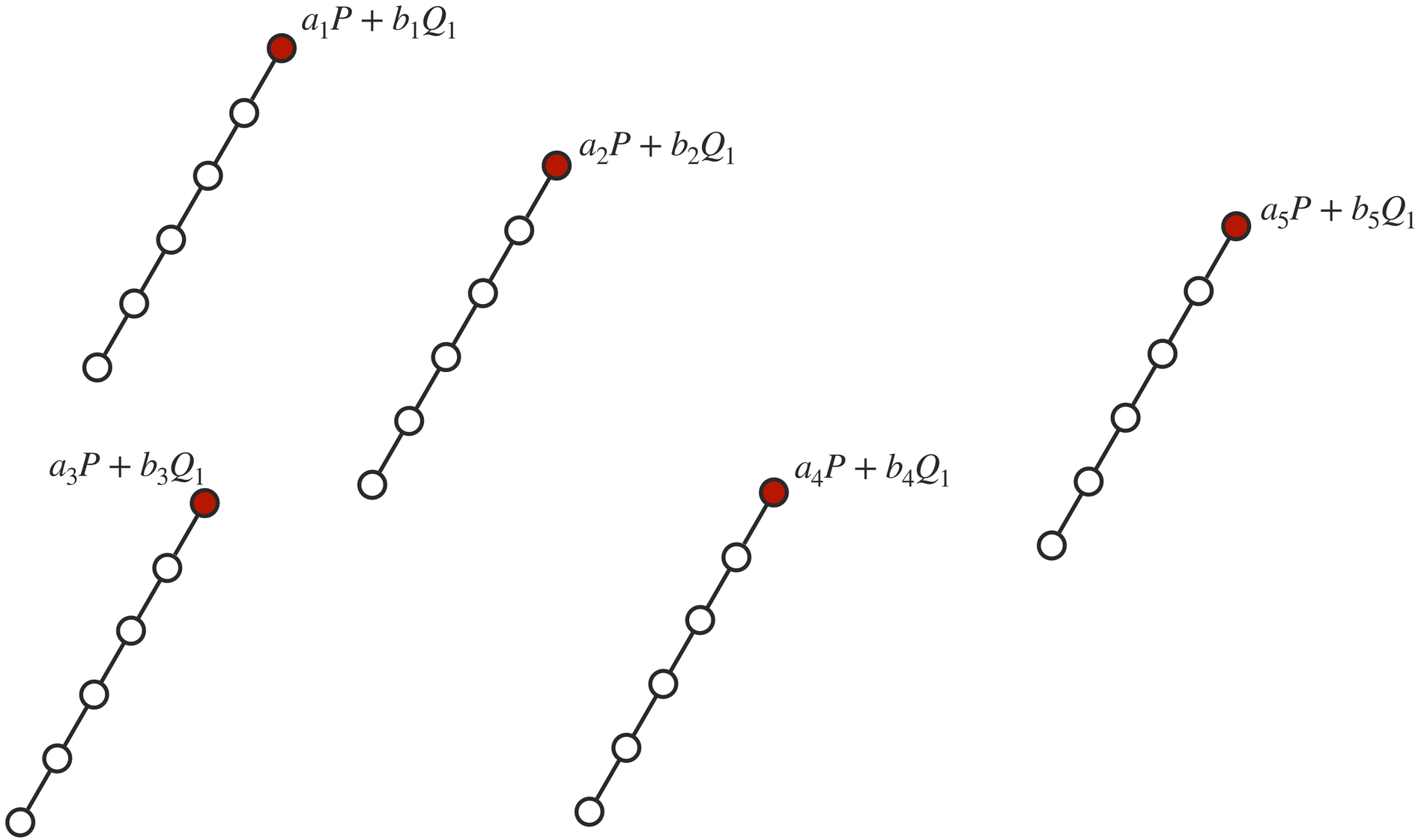
ECDLP instances are friends



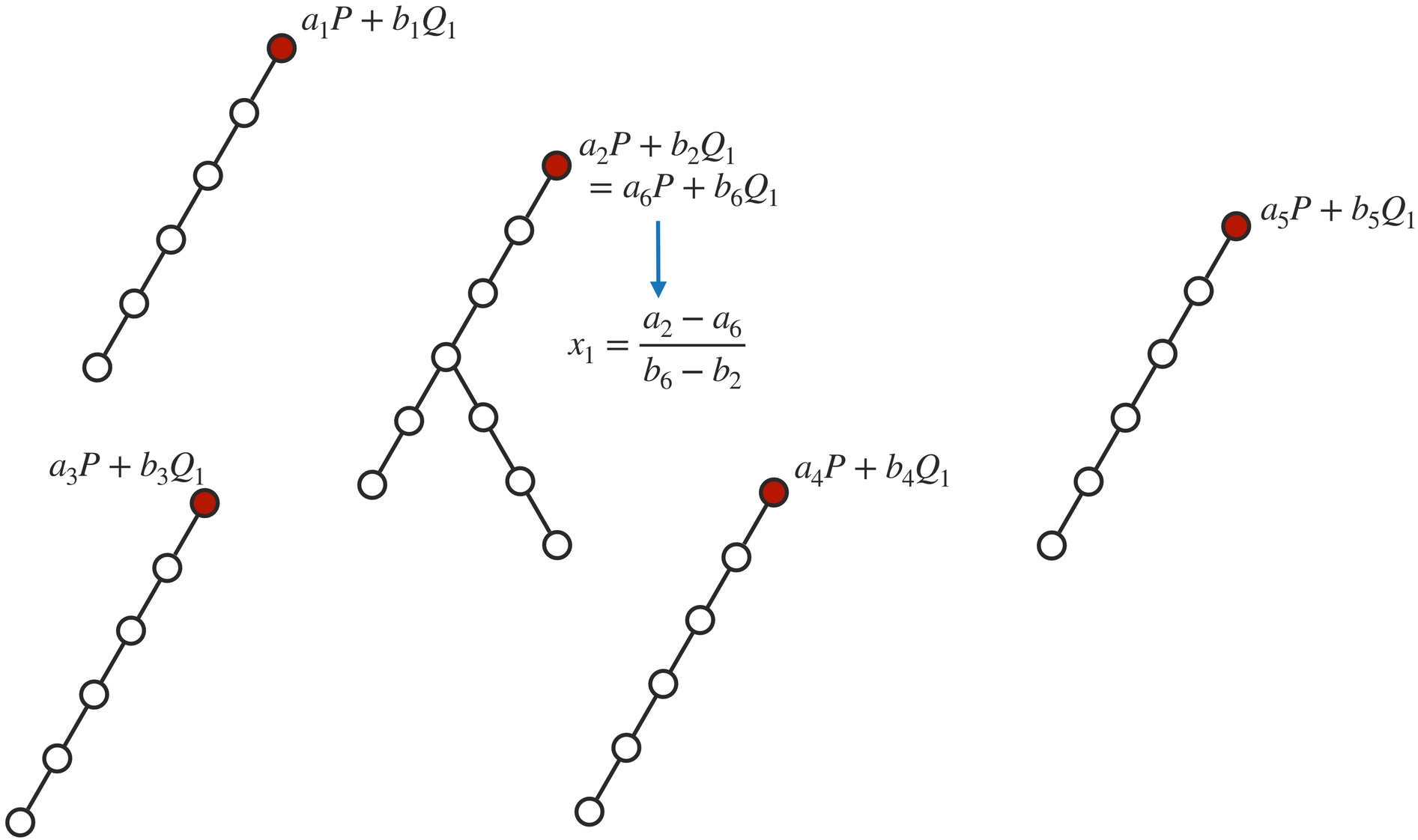
ECDLP instances are friends



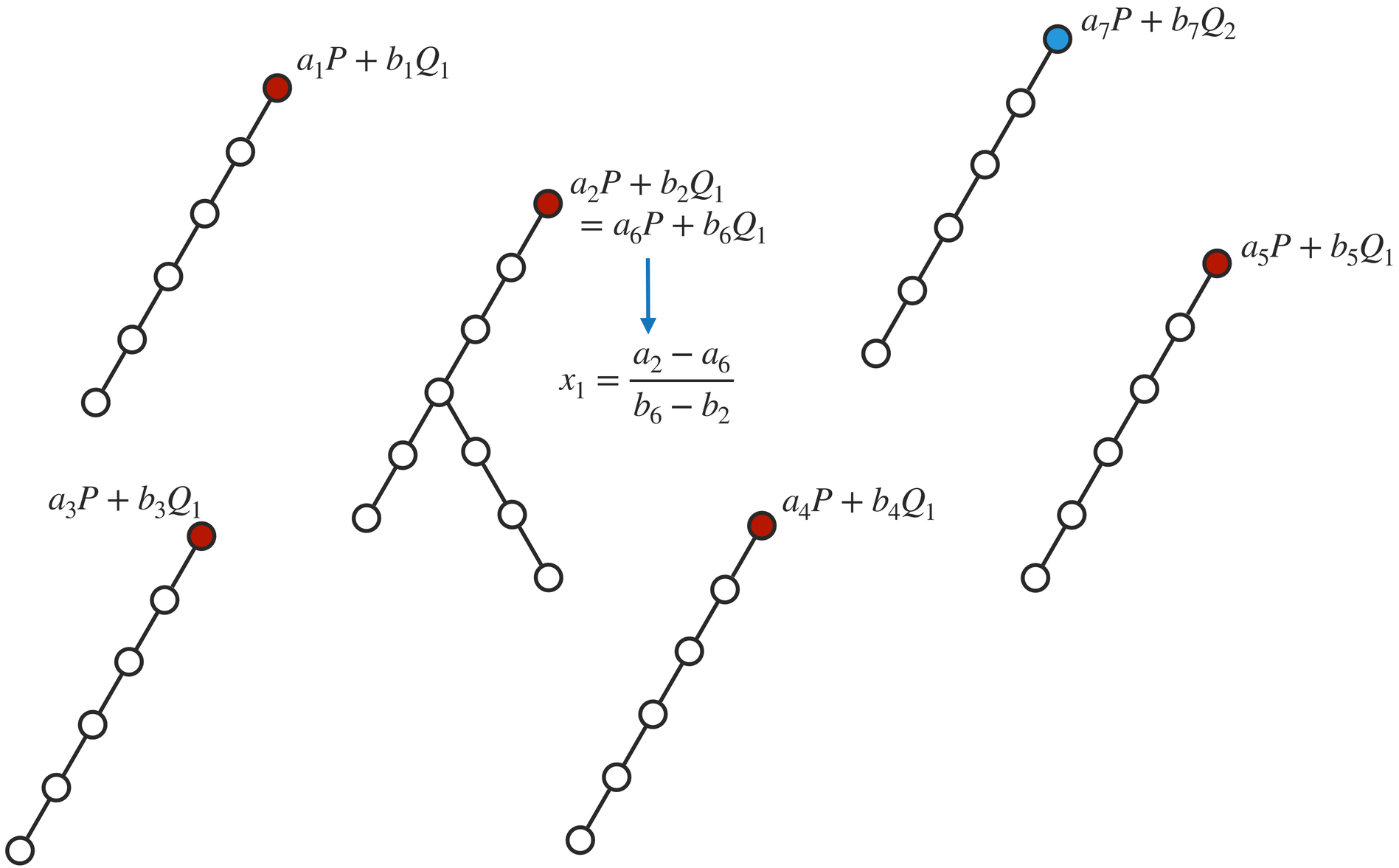
ECDLP instances are friends



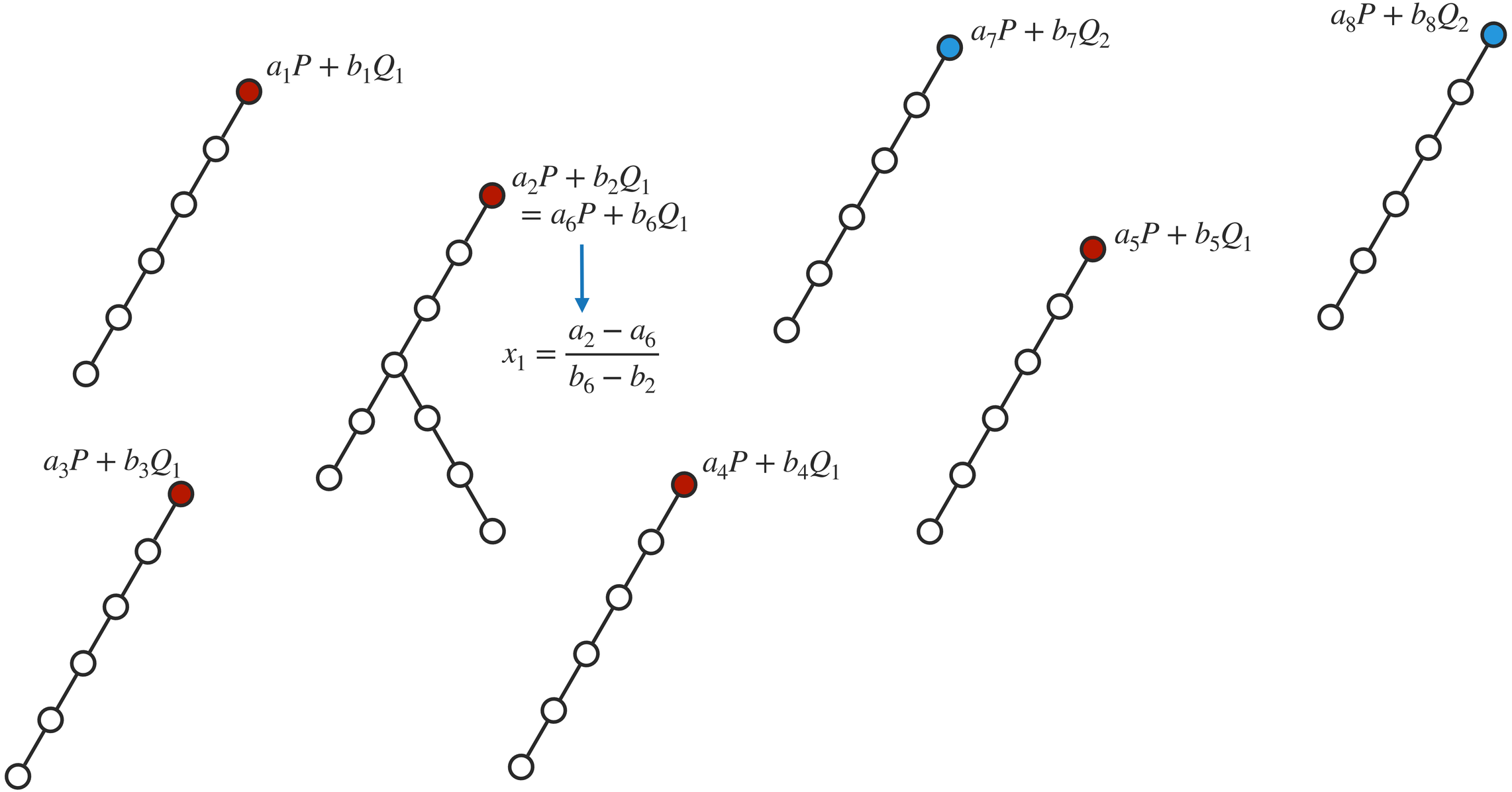
ECDLP instances are friends



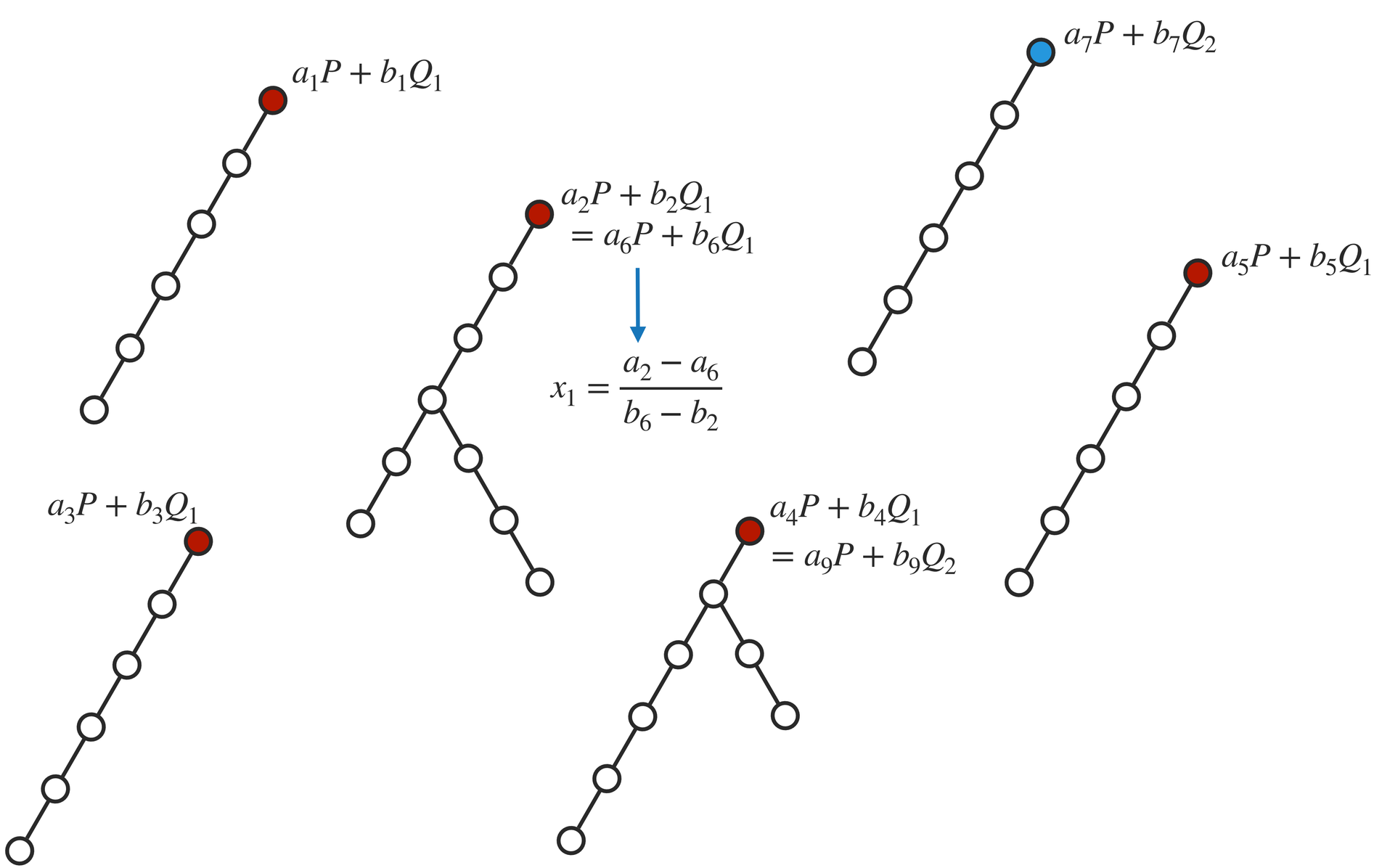
ECDLP instances are friends



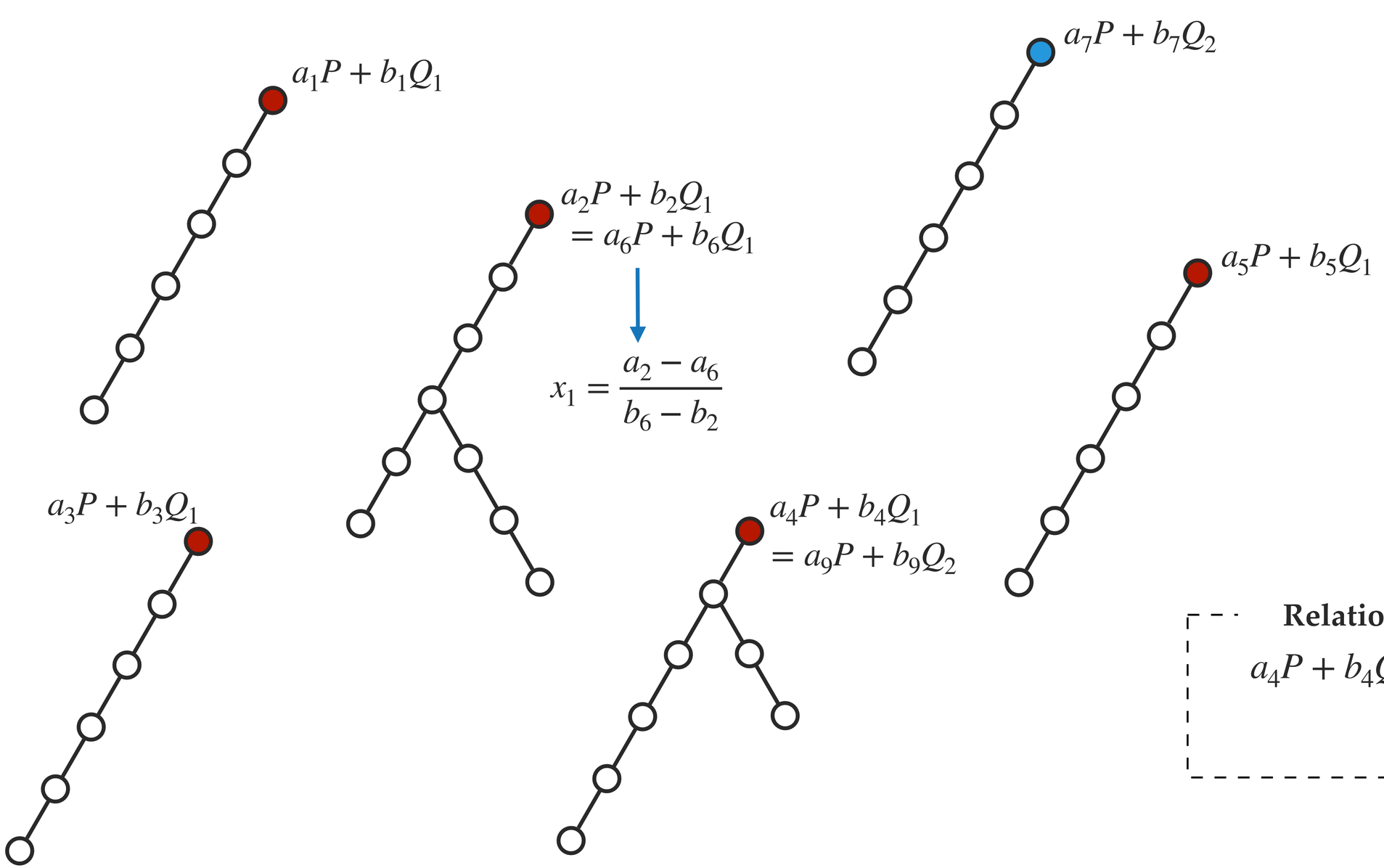
ECDLP instances are friends



ECDLP instances are friends



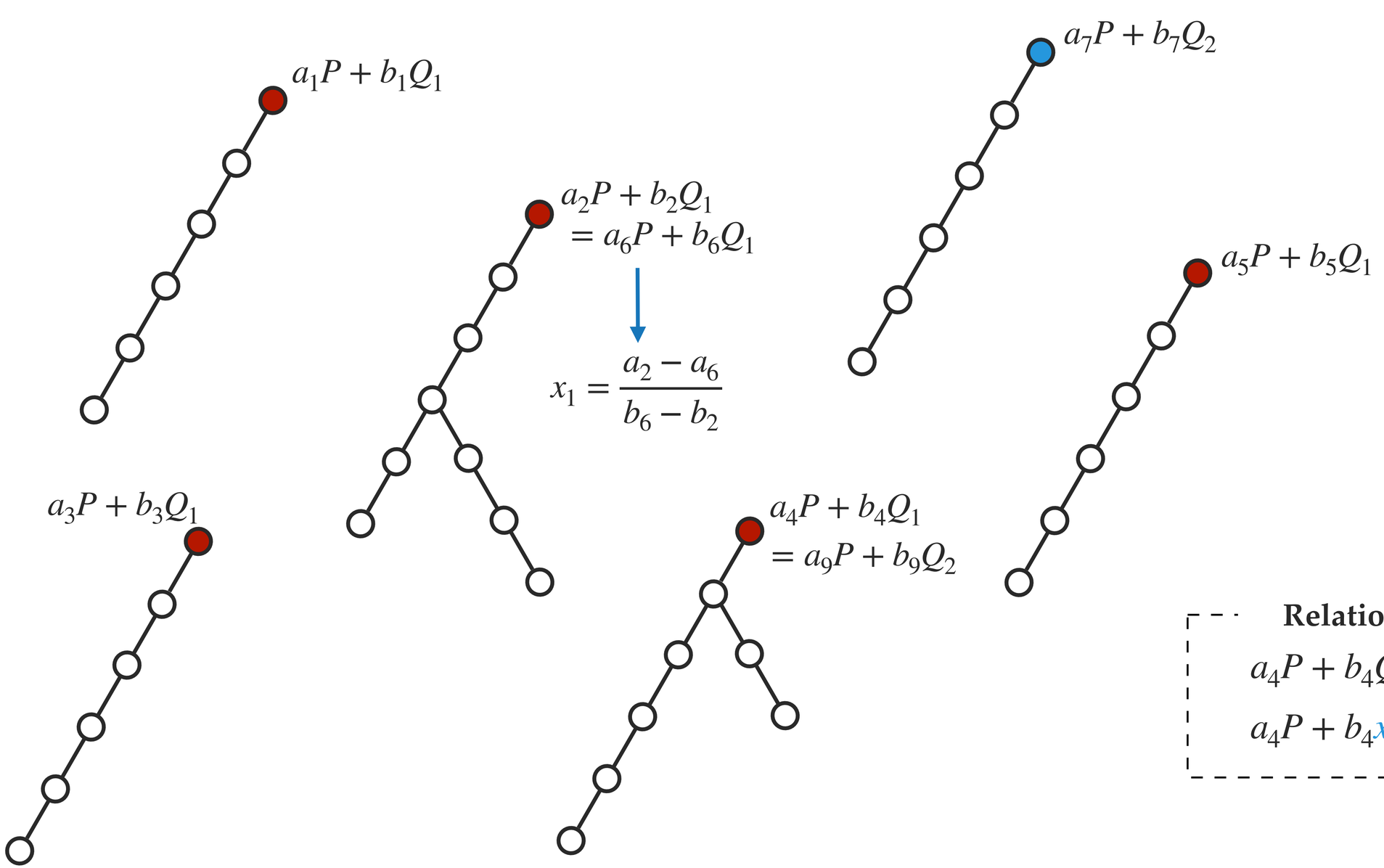
ECDLP instances are friends



Relation between users

$$a_4P + b_4Q_1 = a_9P + b_9Q_2$$

ECDLP instances are friends



Relation between users

$$a_4P + b_4Q_1 = a_9P + b_9Q_2$$

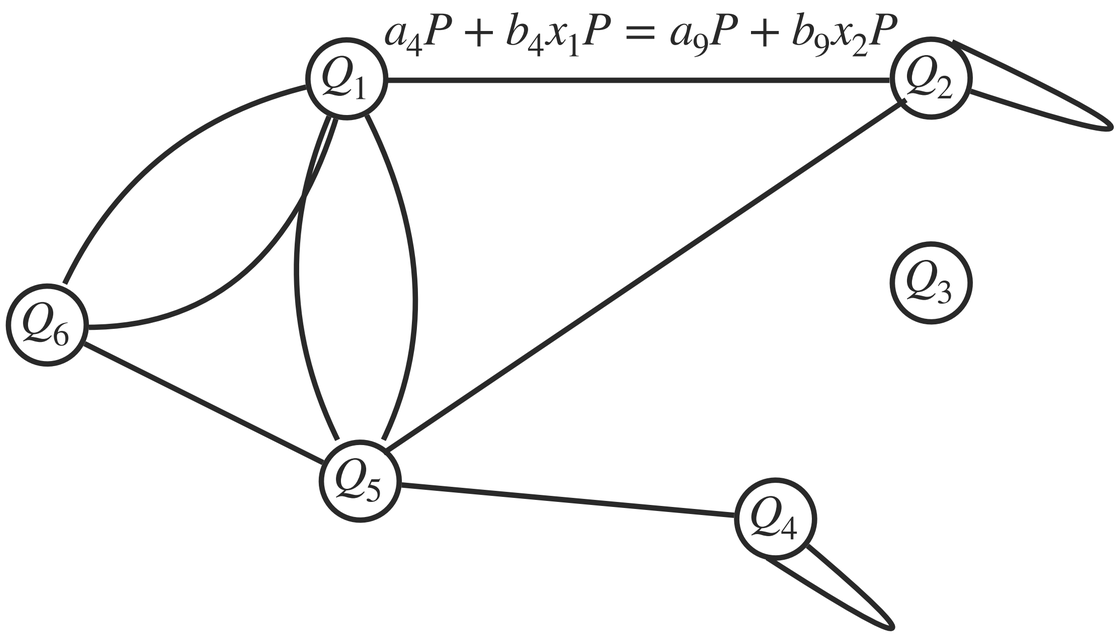
$$a_4P + b_4x_1P = a_9P + b_9x_2P$$

The multi-user graph

(not to be confused with the isogeny graph, seen previously)

Vertices: the public keys of users.

Edges: linear relations between the corresponding secret keys.

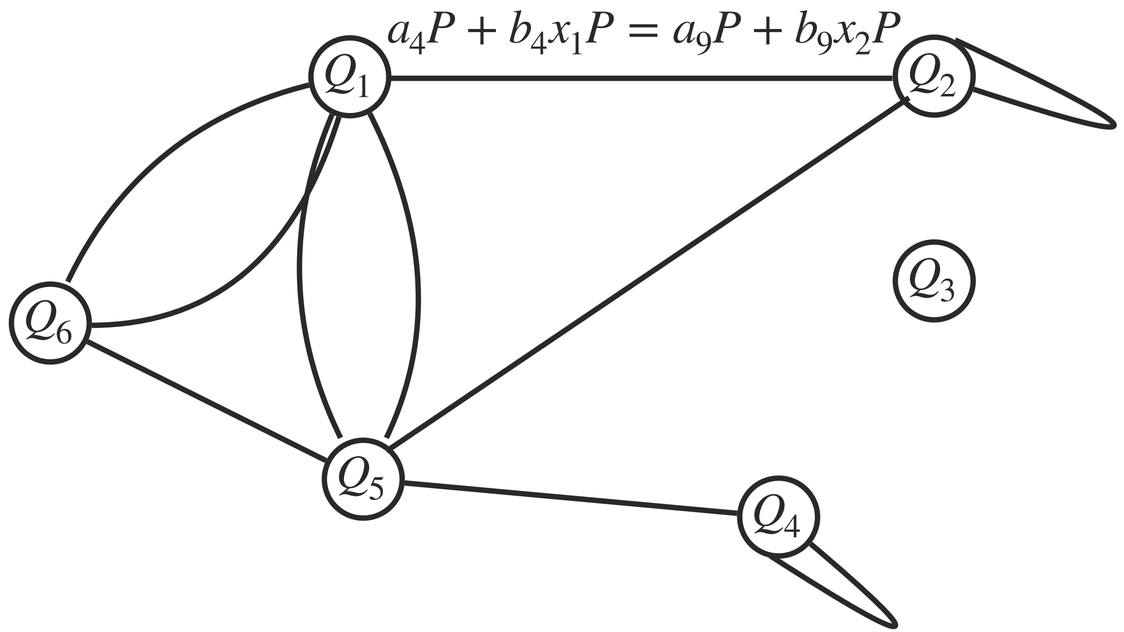


The multi-user graph

(not to be confused with the isogeny graph, seen previously)

Vertices: the public keys of users.

Edges: linear relations between the corresponding secret keys.



Algorithm:

A user of which we know the secret key

→ Add an infiltrator.

→ Produce cL of these relations.

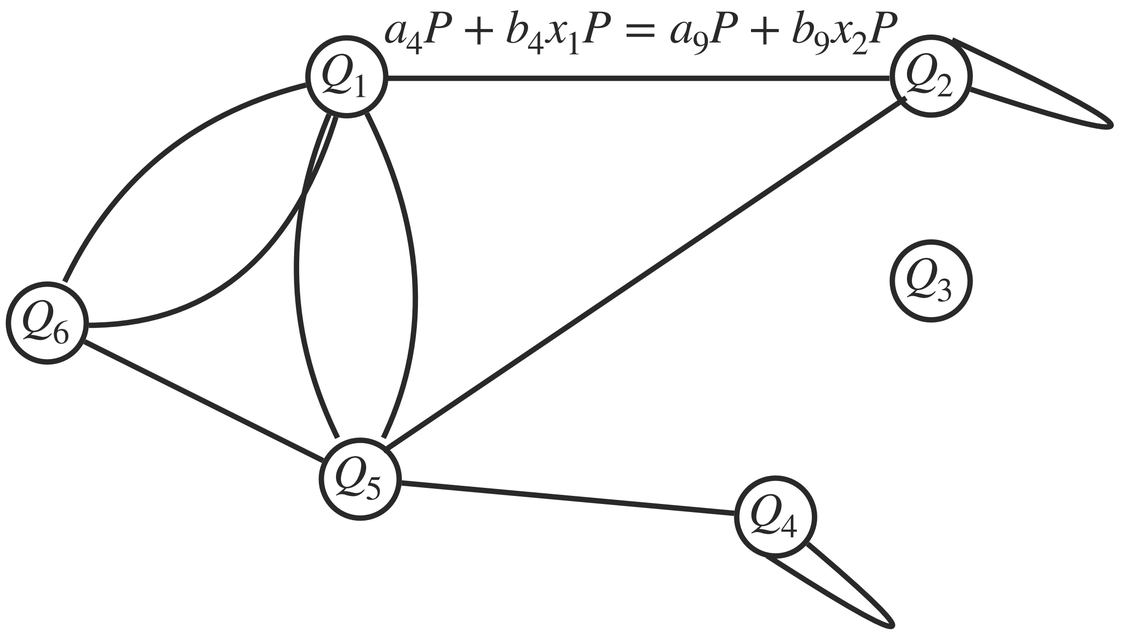
→ Recover the secret keys of all users in the same connected component as the infiltrator.

The multi-user graph

(not to be confused with the isogeny graph, seen previously)

Vertices: the public keys of users.

Edges: linear relations between the corresponding secret keys.



Algorithm:

A user of which we know the secret key

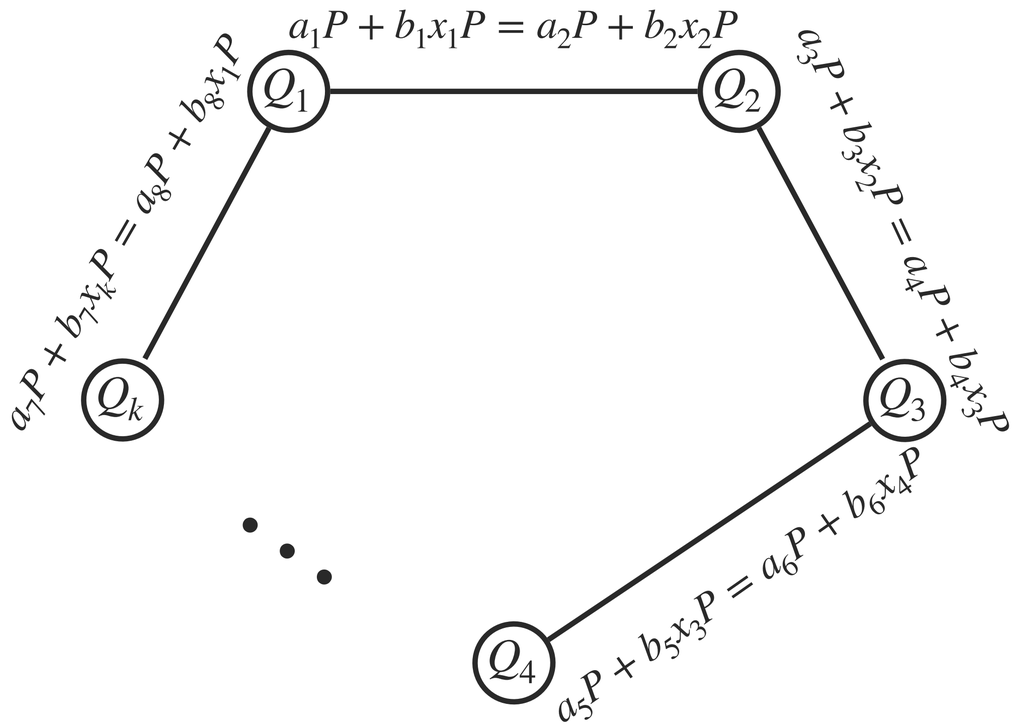
→ Add an infiltrator.

→ Produce cL of these relations.

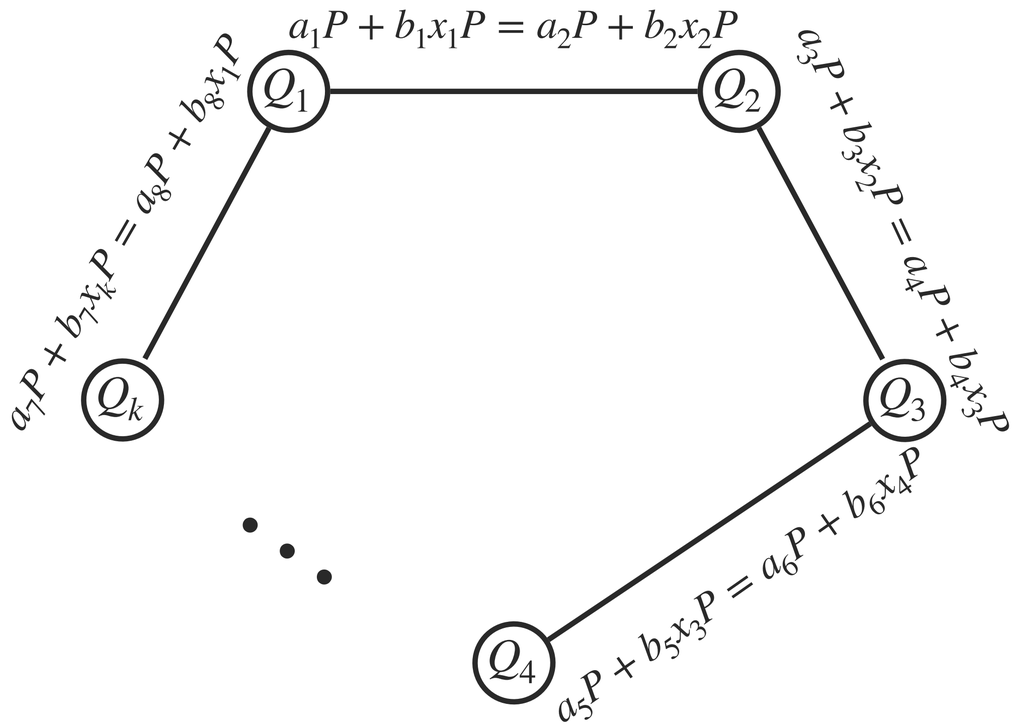
→ Recover the secret keys of all users in the same connected component as the infiltrator.

↳ After $2L$ relations, we expect that $0.98L$ vertices are in the largest connected component, including the infiltrator.

Cycles in the multi-user graph



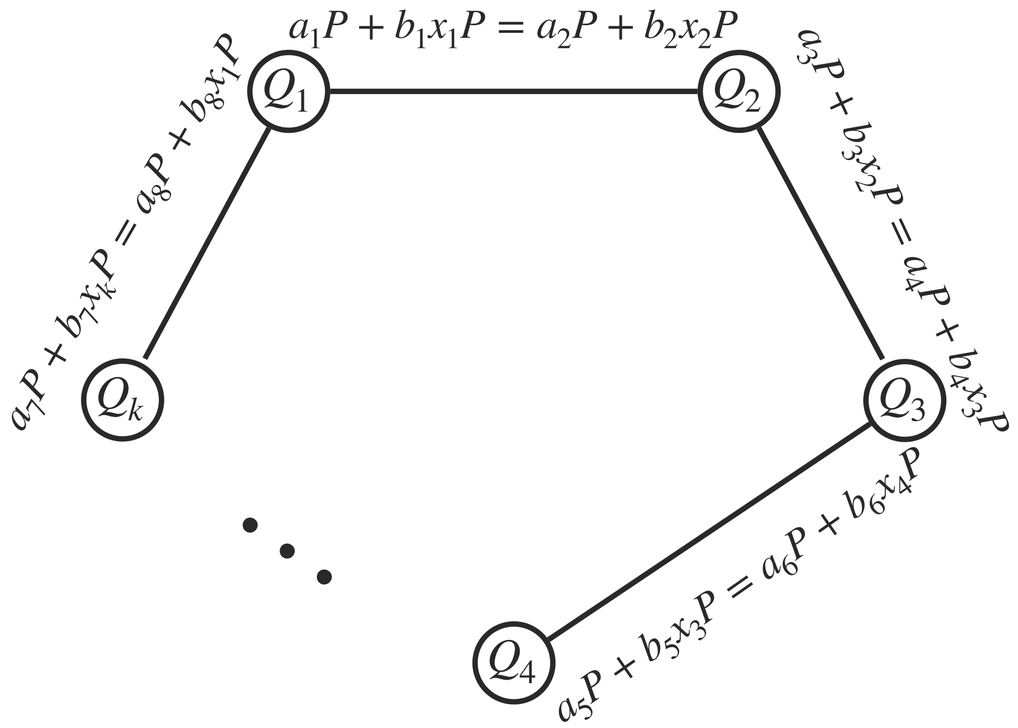
Cycles in the multi-user graph



$$
\begin{aligned}
&a_1P + b_1x_1P = a_2P + b_2x_2P \\
&a_3P + b_3x_2P = a_4P + b_4x_3P \\
&a_5P + b_5x_3P = a_6P + b_6x_4P \\
&\cdots \\
&a_7P + b_7x_kP = a_8P + b_8x_1P
\end{aligned}
$$

A cycle of length k yields a linear system of k equations in k unknowns.

Cycles in the multi-user graph



$$
\begin{aligned}
&a_1P + b_1x_1P = a_2P + b_2x_2P \\
&a_3P + b_3x_2P = a_4P + b_4x_3P \\
&a_5P + b_5x_3P = a_6P + b_6x_4P \\
&\cdots \\
&a_7P + b_7x_kP = a_8P + b_8x_1P
\end{aligned}
$$

A cycle of length k yields a linear system of k equations in k unknowns.

We solve the linear system and recover the secret key of all users in the cycle.

One collision per user is enough

≠ as many collisions as users

One collision per user is enough

≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

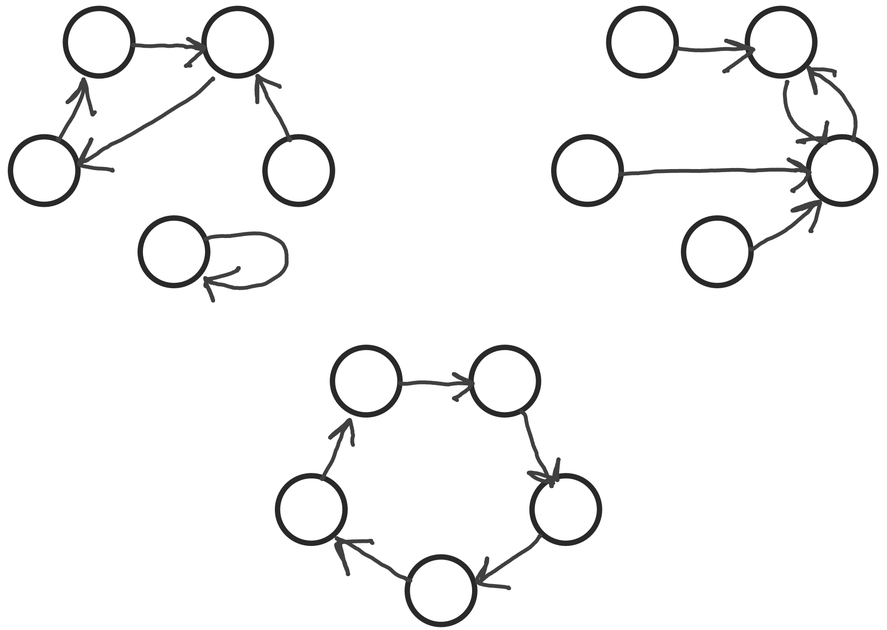
One collision per user is enough

≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

Example.



$L_i \longrightarrow L_j$
collision was discovered by L_i.

One collision per user is enough

≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

Having exactly one outgoing edge

$L_i \longrightarrow L_j$
collision was discovered by L_i.

One collision per user is enough

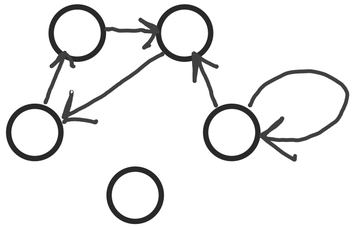
≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

Having exactly one outgoing edge

→ Avoids useless collisions.



$L_i \longrightarrow L_j$
collision was discovered by L_i.

One collision per user is enough

≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

Having exactly one outgoing edge

→ Avoids useless collisions.

→ Can be controlled algorithmically.

$L_i \longrightarrow L_j$
collision was discovered by L_i.

One collision per user is enough

≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

Having exactly one outgoing edge

→ Avoids useless collisions.

→ Can be controlled algorithmically.

! Requires the walk to be user-independent.

$L_i \longrightarrow L_j$
collision was discovered by L_i.

One collision per user is enough

≠ as many collisions as users

About the multi-user graph

Let G be a directed graph where each vertex has exactly one outgoing edge. Then all connected components of G contain a cycle.

Having exactly one outgoing edge

→ Avoids useless collisions.

→ Can be controlled algorithmically.

! Requires the walk to be user-independent.

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$ (starting point are still random $c_0 P + d_0 Q_j$)

$L_i \longrightarrow L_j$
collision was discovered by L_i.

To sum up

To sum up

→ We can use cycles to recover keys.

To sum up

→ We can use cycles to recover keys.

→ We can avoid useless collisions.

To sum up

→ We can use cycles to recover keys.

→ We have a user-independent walk.

→ We can avoid useless collisions.

To sum up

→ We have a user-independent walk.

→ We can use cycles to recover keys.

→ We can avoid useless collisions.

No need for an infiltrator.

We have an exact count on the number of collisions.

We recover all secret keys.

Fixed-degree isogeny pathfinding

A history lesson

The problem underlying SIDH

! We have $p = c2^e3^f - 1$, for small c

SSI-T (simplified)

Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_{p^2}$, the restriction of a 2^e-isogeny φ to the 3^f-torsion of E_0.
Question: Find the 2^e-isogeny φ such that $\varphi : E_0 \to E_1$.

A history lesson

The problem underlying SIDH

SSI-T (simplified)

! We have $p = c2^e3^f - 1$, for small c

Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_{p^2}$, the restriction of a 2^e-isogeny φ to the 3^f-torsion of E_0.
Question: Find the 2^e-isogeny φ such that $\varphi : E_0 \to E_1$.

poly time.

A history lesson

The problem underlying SIDH

! We have $p = c2^e3^f - 1$, for small c

SSI-T (simplified)

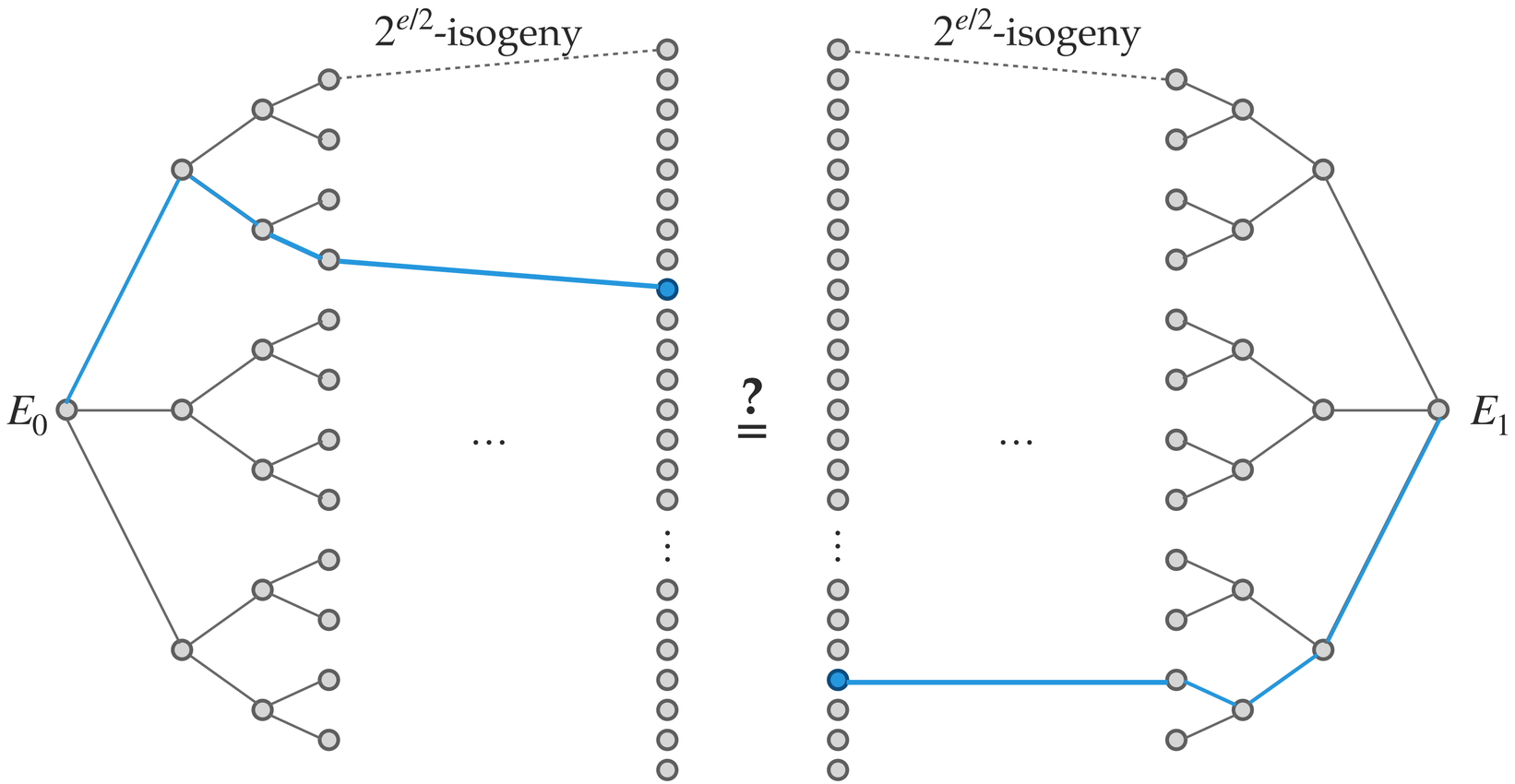
Input: Two supersingular elliptic curves E_0 and E_1 defined over $\mathbb{F}_{p^2}$, the restriction of a 2^e-isogeny φ to the 3^f-torsion of E_0.
Question: Find the 2^e-isogeny φ such that $\varphi : E_0 \to E_1$.

→ The vOW approach has no way of using torsion information.

↳ We look at the problem simply as a fixed-degree isogeny pathfinding problem.

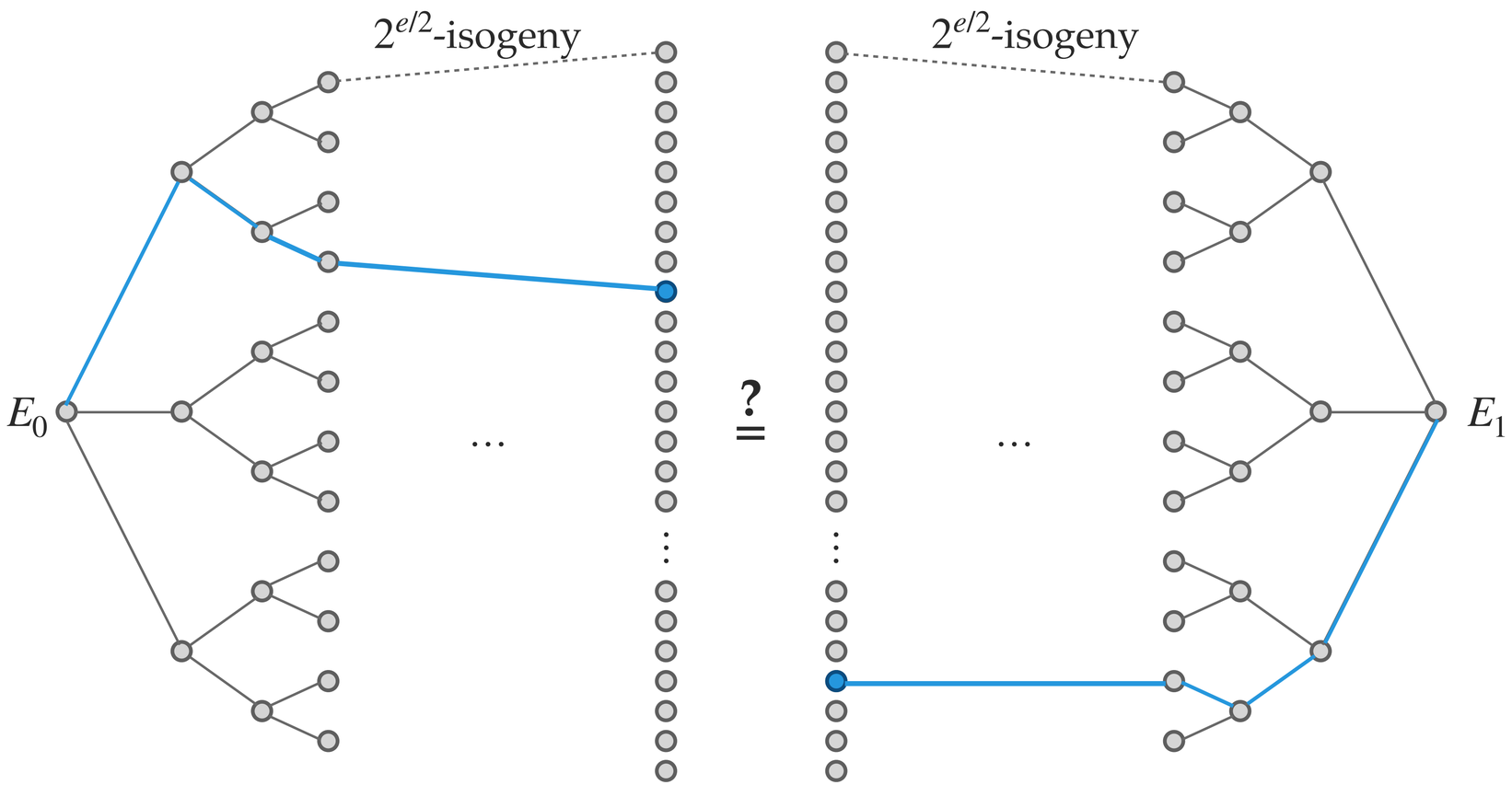
Meet-in-the-middle

Goal: Find a 2^e-isogeny from E_0 to E_1.



Meet-in-the-middle

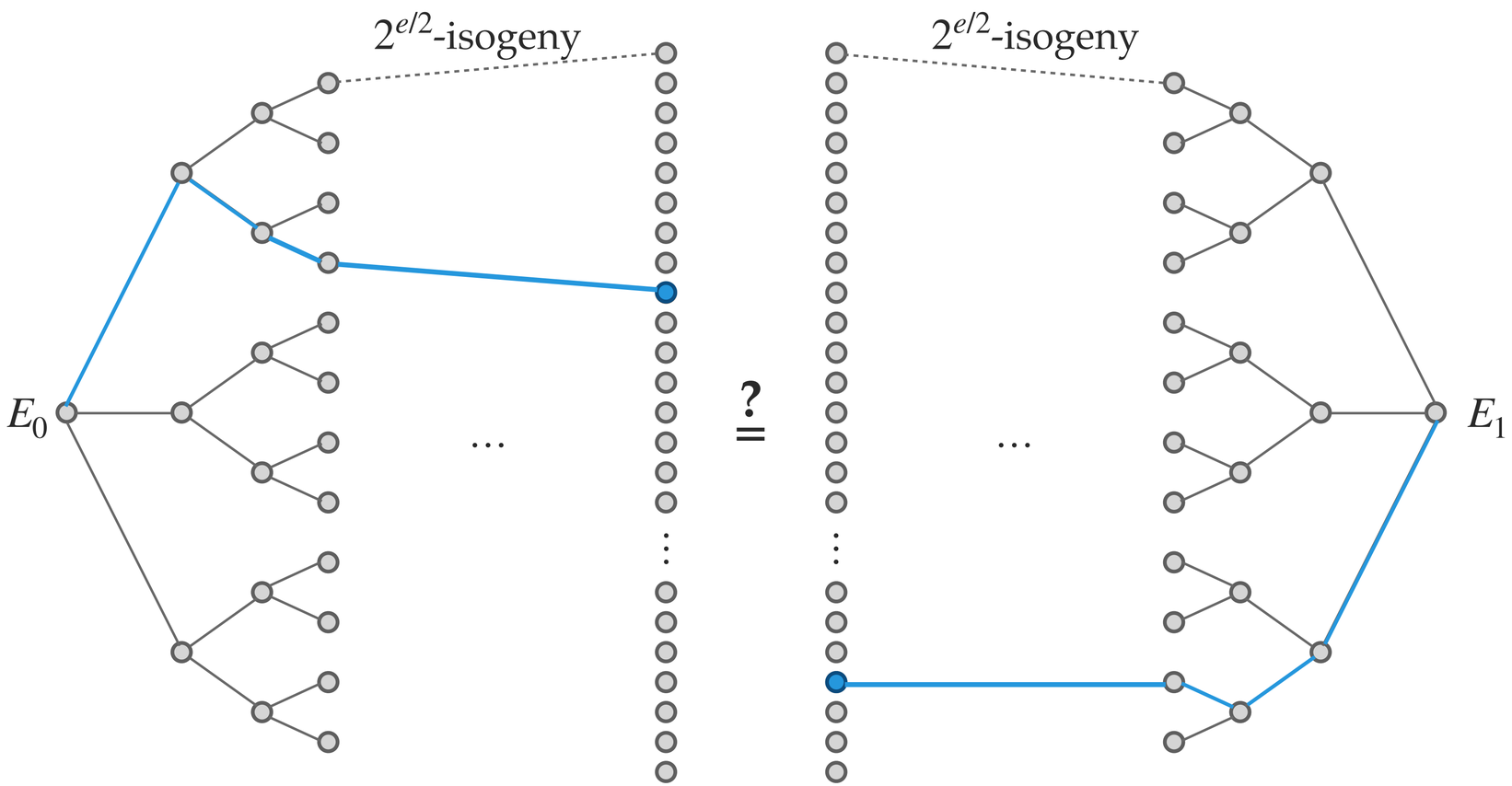
Goal: Find a 2^e-isogeny from E_0 to E_1.



Compute and store all curves on the left.

Meet-in-the-middle

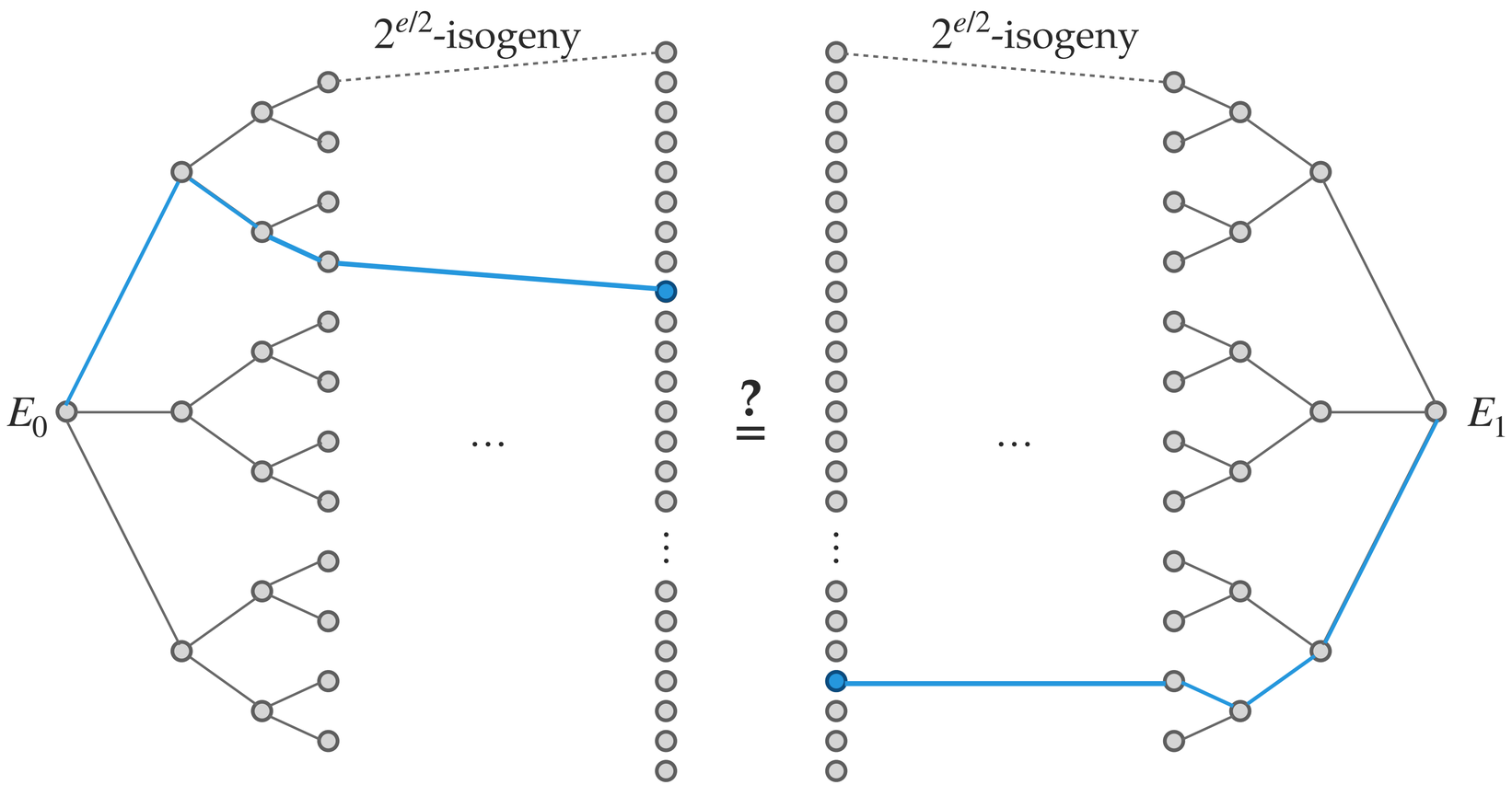
Goal: Find a 2^e-isogeny from E_0 to E_1.



- Compute and store all curves on the left.
- Compute a curve on the right and check for a collision until we find one.

Meet-in-the-middle

Goal: Find a 2^e-isogeny from E_0 to E_1.



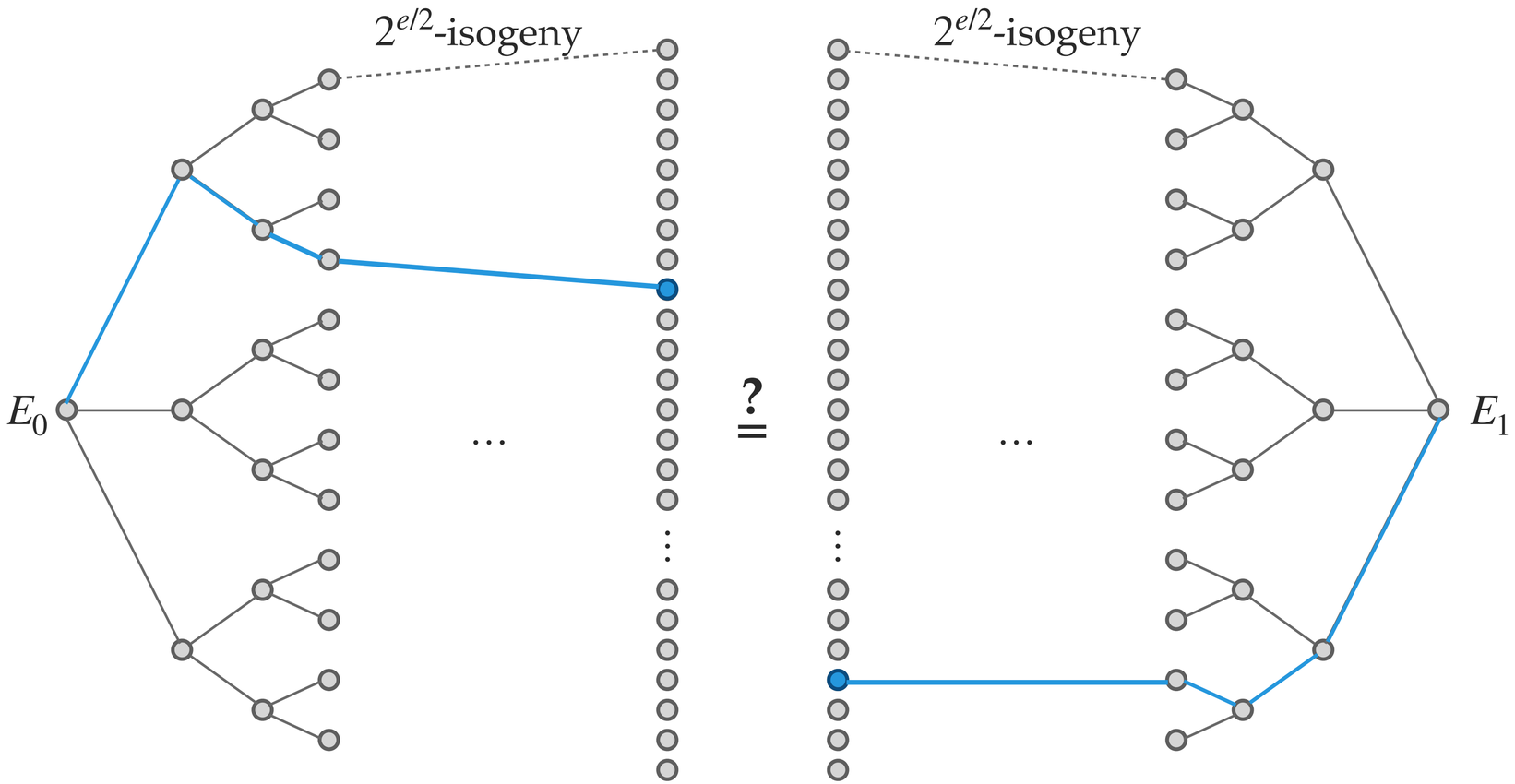
- Compute and store all curves on the left.
- Compute a curve on the right and check for a collision until we find one.

$\mathcal{O}(2^{e/2})$

in both time and memory.

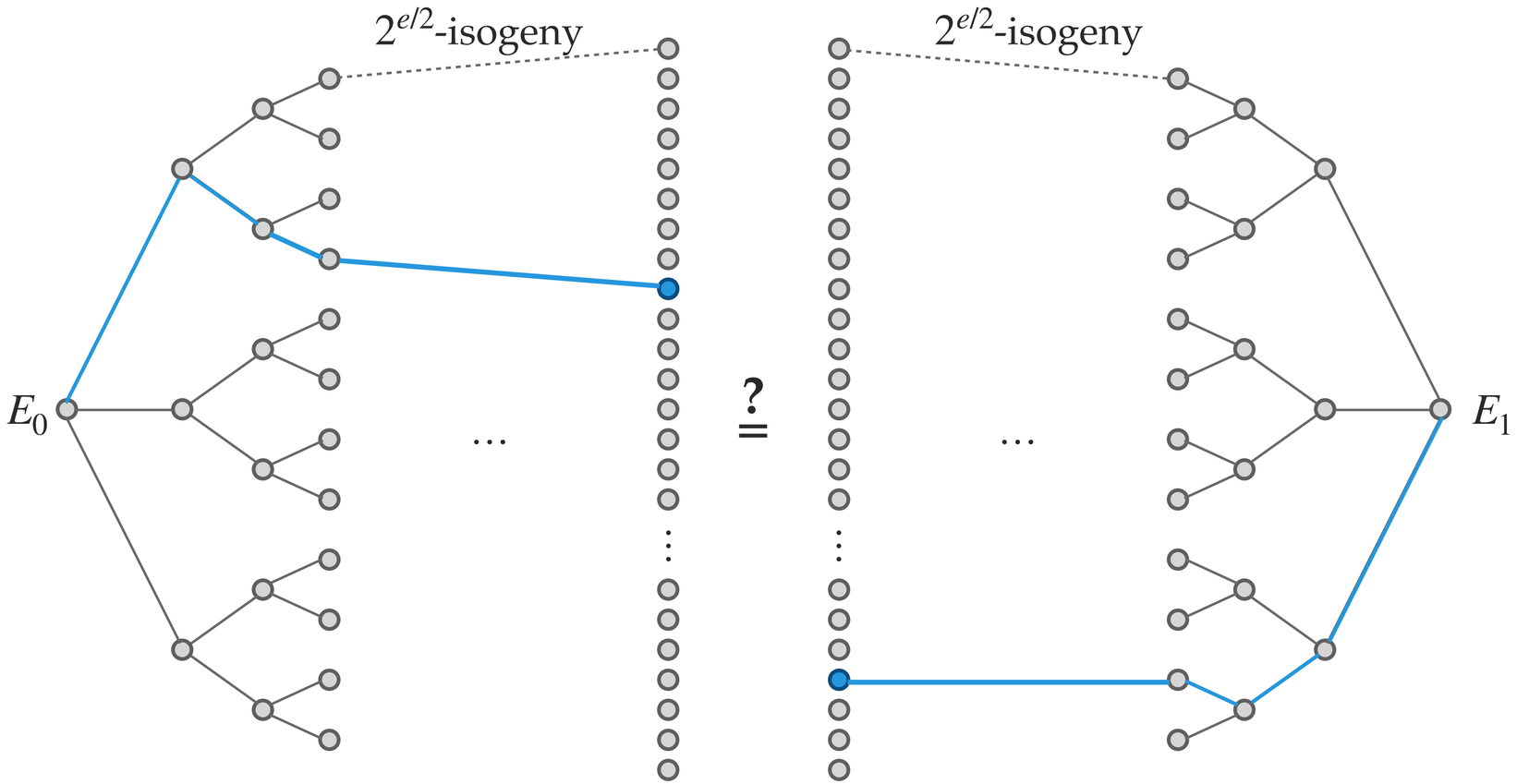
*Trade-offs are possible.

vOW for fixed-degree isogeny pathfinding



vOW for fixed-degree isogeny pathfinding

$I = \{1,\ldots,2^{e/2}\}$ - indices of possible paths
$A_{j,i}$ - the ith subgroup of order $\ell^{e/2}$ of $E_j[\ell^e]$
J - the set of j-invariants 'in the middle'

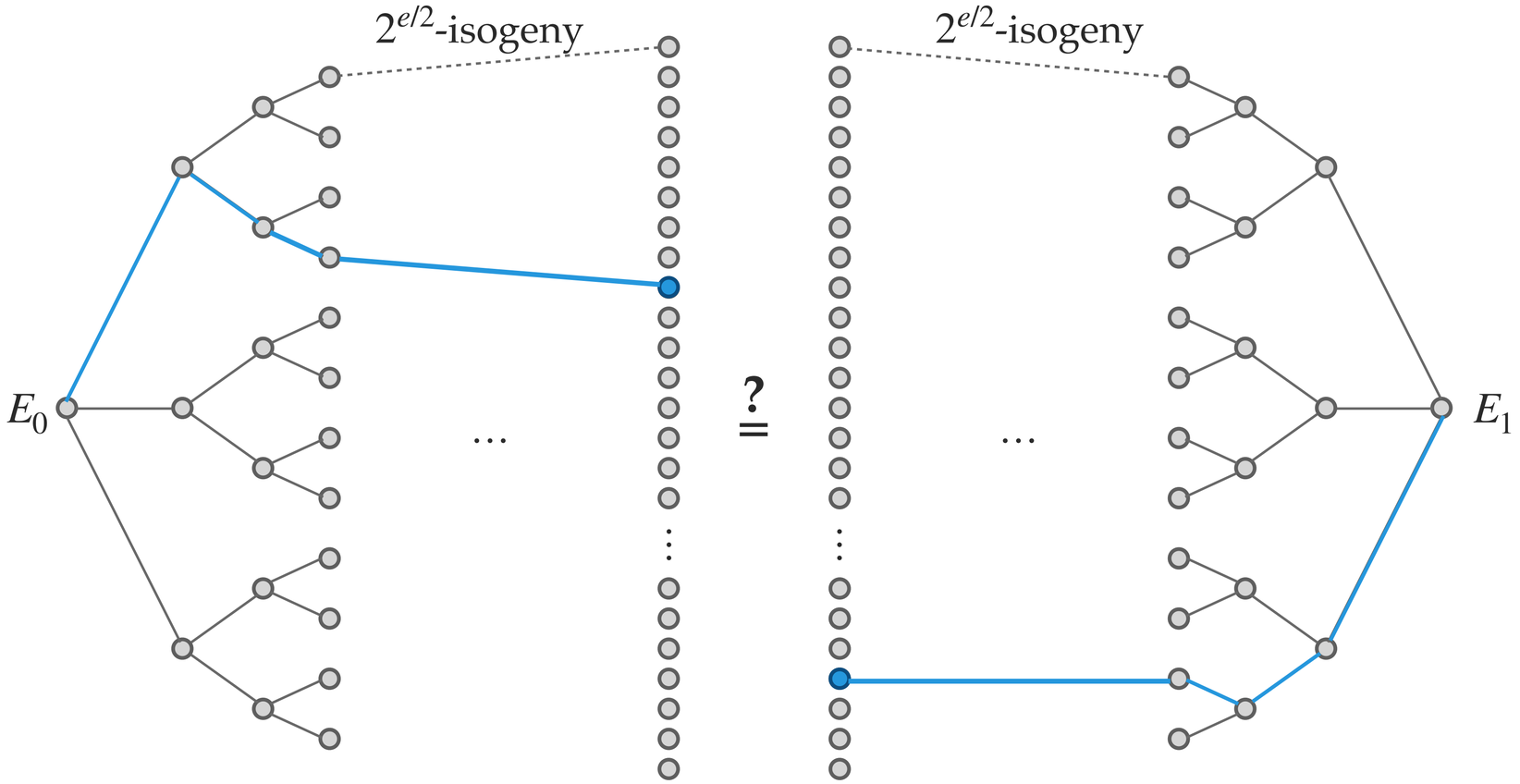


vOW for fixed-degree isogeny pathfinding

$I = \{1,\ldots,2^{e/2}\}$ - indices of possible paths
$A_{j,i}$ - the ith subgroup of order $\ell^{e/2}$ of $E_j[\ell^e]$
J - the set of j-invariants 'in the middle'

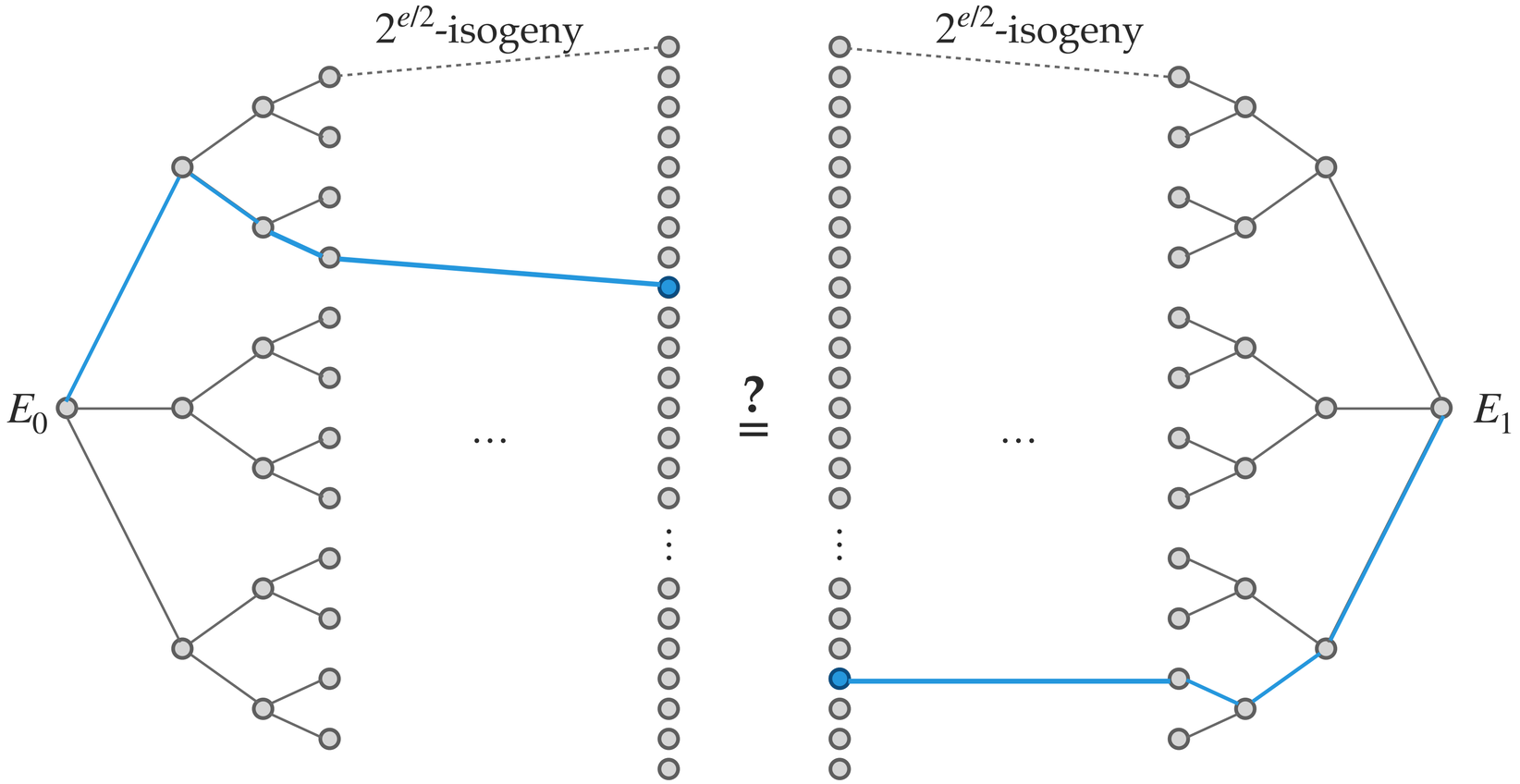
$$F : I \times \{1,2\} \to J$$
$$(i, j) \mapsto \mathrm{j}(E_j/A_{j,i})$$

Taking the j-invariant



vOW for fixed-degree isogeny pathfinding

$I = \{1,\ldots,2^{e/2}\}$ - indices of possible paths
$A_{j,i}$ - the ith subgroup of order $\ell^{e/2}$ of $E_j[\ell^e]$
J - the set of j-invariants 'in the middle'



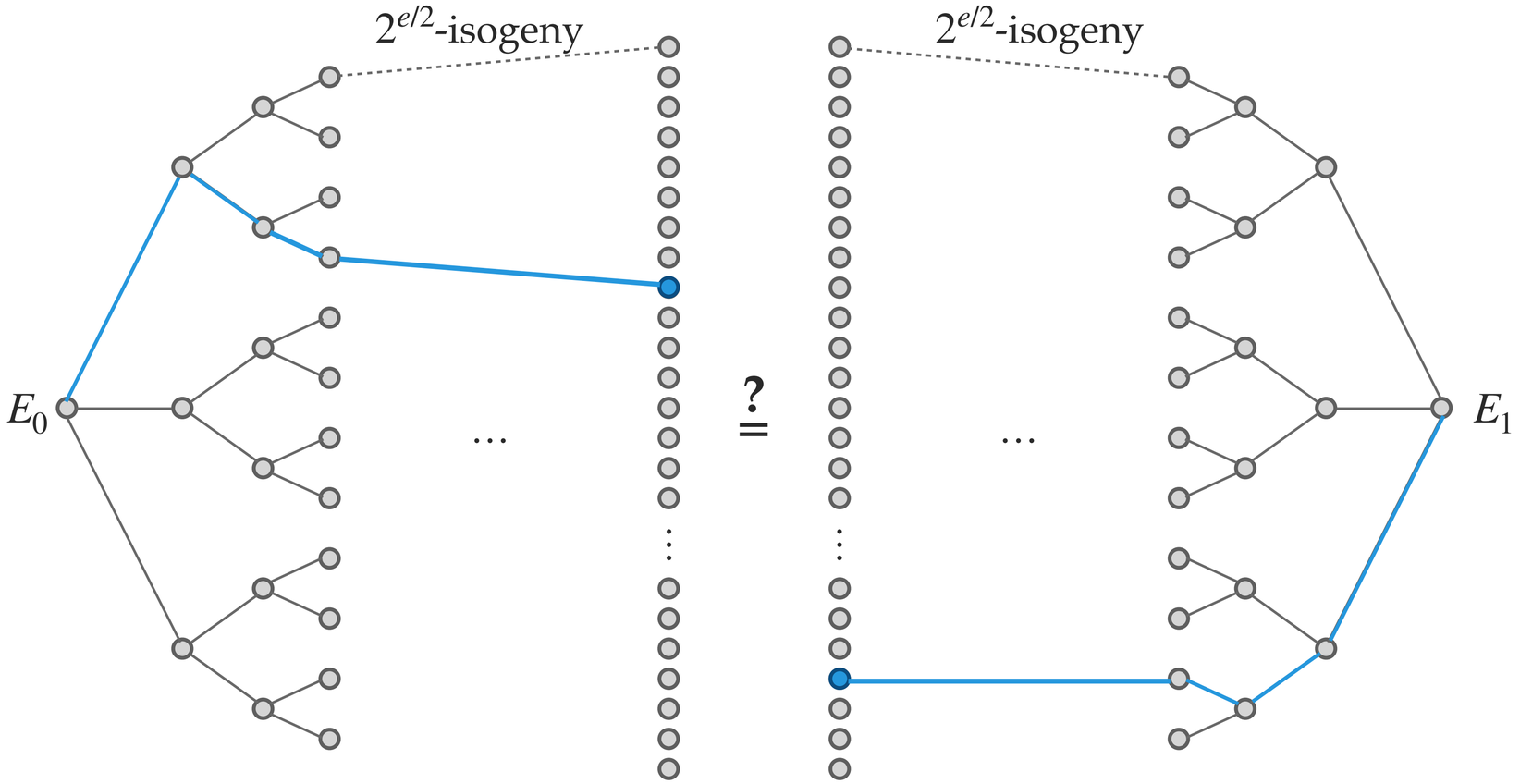
$F : I \times \{1,2\} \to J$
$(i, j) \mapsto \mathsf{j}(E_j/A_{j,i})$

Taking the j-invariant

$g : J \to I \times \{1,2\}$
$j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$ parsed into $I \times \{1,2\}$

vOW for fixed-degree isogeny pathfinding

$I = \{1,\ldots,2^{e/2}\}$ - indices of possible paths
$A_{j,i}$ - the ith subgroup of order $\ell^{e/2}$ of $E_j[\ell^e]$
J - the set of j-invariants 'in the middle'



$F : I \times \{1,2\} \to J$
$(i,j) \mapsto \mathsf{j}(E_j/A_{j,i})$

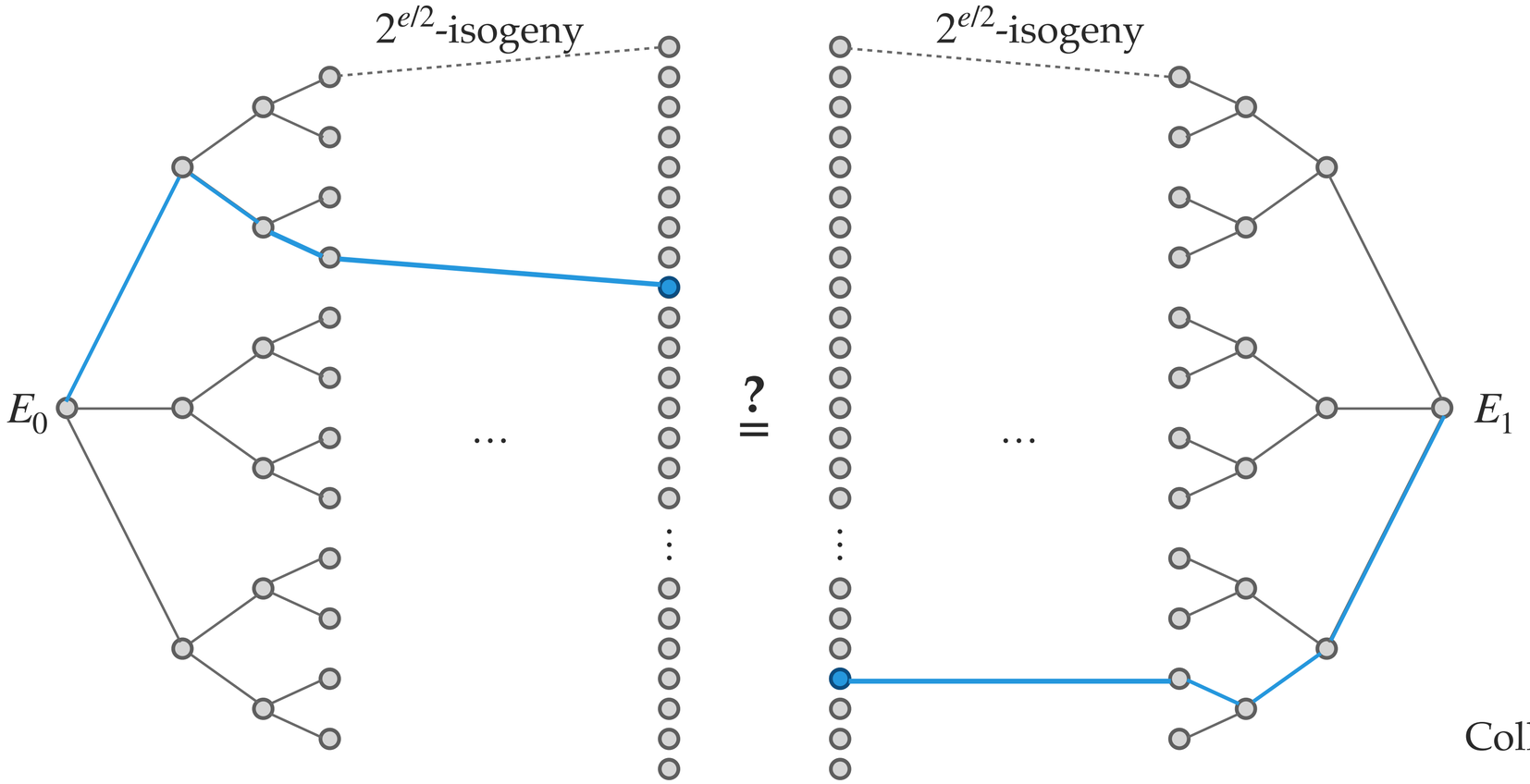
↳ Taking the j-invariant

$g : J \to I \times \{1,2\}$
$j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$ parsed into $I \times \{1,2\}$

$H : I \times \{1,2\} \to I \times \{1,2\}$
$(i,j) \mapsto g(F(i,j))$

vOW for fixed-degree isogeny pathfinding

$I = \{1,\ldots,2^{e/2}\}$ - indices of possible paths
$A_{j,i}$ - the ith subgroup of order $\ell^{e/2}$ of $E_j[\ell^e]$
J - the set of j-invariants 'in the middle'



$F : I \times \{1,2\} \to J$
$(i,j) \mapsto \mathsf{j}(E_j/A_{j,i})$

↳ Taking the j-invariant

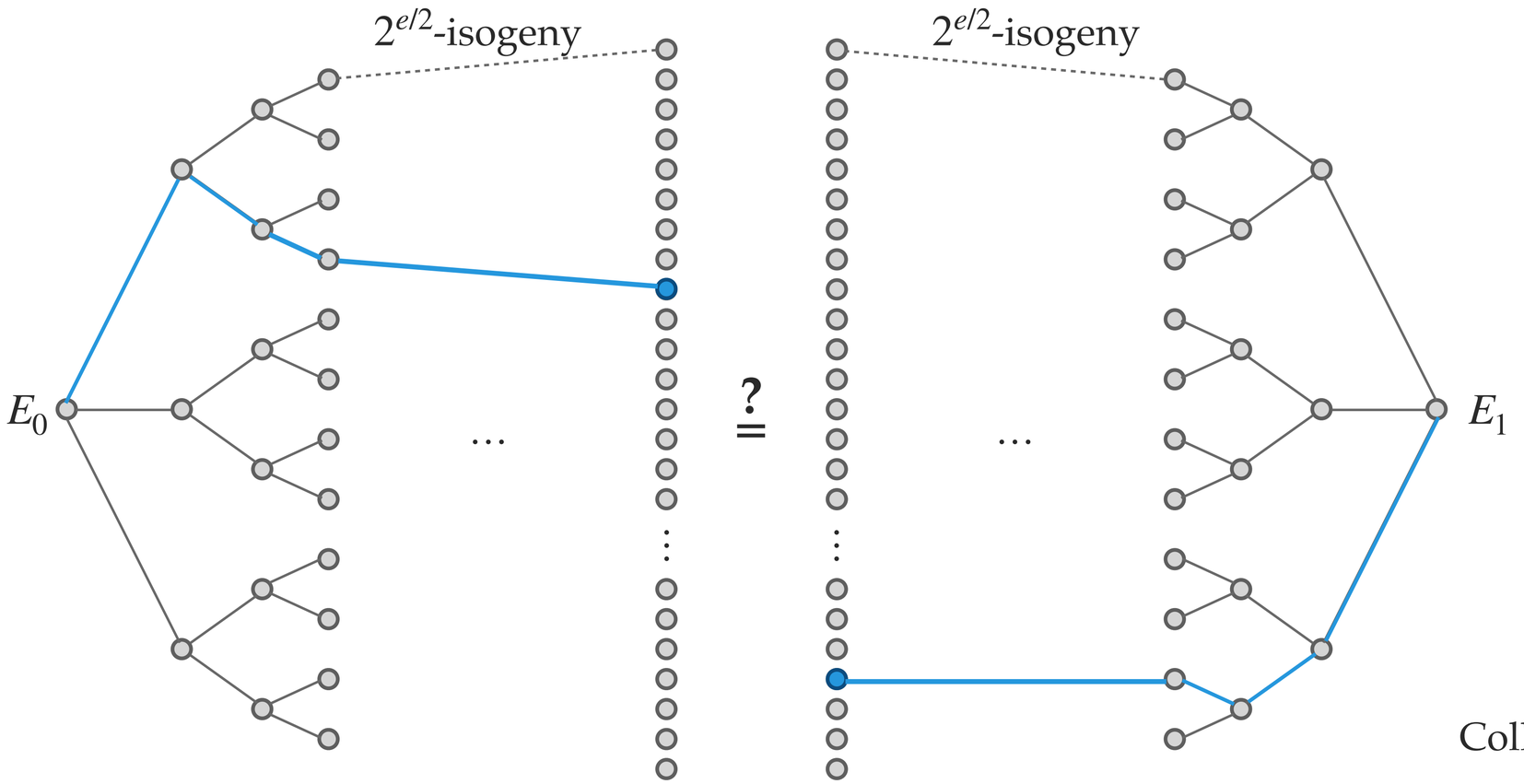
$g : J \to I \times \{1,2\}$
$j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$ parsed into $I \times \{1,2\}$

$H : I \times \{1,2\} \to I \times \{1,2\}$
$(i,j) \mapsto g(F(i,j))$

Collision of H

vOW for fixed-degree isogeny pathfinding

$I = \{1,\ldots,2^{e/2}\}$ - indices of possible paths
$A_{j,i}$ - the ith subgroup of order $\ell^{e/2}$ of $E_j[\ell^e]$
J - the set of j-invariants 'in the middle'



$F : I \times \{1,2\} \to J$
$(i,j) \mapsto \mathsf{j}(E_j/A_{j,i})$

↳ Taking the j-invariant

$g : J \to I \times \{1,2\}$
$j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$ parsed into $I \times \{1,2\}$

$H : I \times \{1,2\} \to I \times \{1,2\}$
$(i,j) \mapsto g(F(i,j))$

Collision of H
↗ collision of F : golden
↘ collision of g : a hash collision

A different view of vOW for ECDLP

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

A different view of vOW for ECDLP

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad g : J \to I \times \{1,2\}$$
$$(i,j) \mapsto \mathsf{j}(E_j / A_{j,i}) \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\}$$
$$(i,j) \mapsto g(F(i,j))$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad g : J \to I \times \{1,2\}$$
$$(i,j) \mapsto \mathsf{j}(E_j / A_{j,i}) \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\}$$
$$(i,j) \mapsto g(F(i,j))$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad g : J \to I \times \{1,2\}$$
$$(i,j) \mapsto \mathsf{j}(E_j / A_{j,i}) \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\}$$
$$(i,j) \mapsto g(F(i,j))$$

Rewrite vOW walk for ECDLP

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad g : J \to I \times \{1,2\}$$
$$(i,j) \mapsto \mathsf{j}(E_j / A_{j,i}) \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\}$$
$$(i,j) \mapsto g(F(i,j))$$

Rewrite vOW walk for ECDLP

$$F : S \times \{1,\ldots,k\} \to S$$
$$(R,i) \mapsto R + c_i P$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad (i, j) \mapsto \mathsf{j}(E_j / A_{j,i})$$

$$g : J \to I \times \{1,2\} \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\} \qquad (i, j) \mapsto g(F(i, j))$$

Rewrite vOW walk for ECDLP

$$F : S \times \{1, \ldots, k\} \to S \qquad (R, i) \mapsto R + c_i P$$

$$g : S \to S \times \{1, \ldots, k\} \qquad R \mapsto (R, R \bmod k)$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad (i, j) \mapsto \mathsf{j}(E_j / A_{j,i})$$

$$g : J \to I \times \{1,2\} \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\} \qquad (i, j) \mapsto g(F(i, j))$$

Rewrite vOW walk for ECDLP

$$F : S \times \{1, \ldots, k\} \to S \qquad (R, i) \mapsto R + c_i P$$

$$g : S \to S \times \{1, \ldots, k\} \qquad R \mapsto (R, R \bmod k)$$

$$H : S \times \{1, \ldots, k\} \to S \times \{1, \ldots, k\} \qquad (R, i) \mapsto g(F(R, i))$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad (i, j) \mapsto \mathsf{j}(E_j / A_{j,i})$$

$$g : J \to I \times \{1,2\} \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\} \qquad (i, j) \mapsto g(F(i, j))$$

Rewrite vOW walk for ECDLP

$$F : S \times \{1, \ldots, k\} \to S \qquad (R, i) \mapsto R + c_i P$$

$$g : S \to S \times \{1, \ldots, k\} \quad \text{! injective} \qquad R \mapsto (R, R \bmod k)$$

$$H : S \times \{1, \ldots, k\} \to S \times \{1, \ldots, k\} \qquad (R, i) \mapsto g(F(R, i))$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad (i,j) \mapsto \mathsf{j}(E_j / A_{j,i})$$

$$g : J \to I \times \{1,2\} \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\} \qquad (i,j) \mapsto g(F(i,j))$$

Rewrite vOW walk for ECDLP

$$F : S \times \{1,\ldots,k\} \to S \qquad (R,i) \mapsto R + c_i P$$

$$g : S \to S \times \{1,\ldots,k\} \qquad R \mapsto (R, R \bmod k)$$

! injective

no collisions

$$H : S \times \{1,\ldots,k\} \to S \times \{1,\ldots,k\} \qquad (R,i) \mapsto g(F(R,i))$$

A different view of vOW for ECDLP

Recall

collision of F : good
collision of g : bad

vOW walk for ECDLP

$$f(R) = \begin{cases} R + c_1 P & \text{if } R \in S_1 \\ R + c_2 P & \text{if } R \in S_2 \\ \cdots & \\ R + c_k P & \text{if } R \in S_k, \end{cases}$$

recall: $S = \{R \in E(\mathbb{F}_q)\}$

vOW walk for fixed-degree isogeny pathfinding

$$F : I \times \{1,2\} \to J \qquad (i,j) \mapsto \mathsf{j}(E_j / A_{j,i})$$

$$g : J \to I \times \{1,2\} \qquad j_{\mathsf{inv}} \mapsto \mathsf{hash}(j_{\mathsf{inv}})$$

$$H : I \times \{1,2\} \to I \times \{1,2\} \qquad (i,j) \mapsto g(F(i,j))$$

Rewrite vOW walk for ECDLP

$$F : S \times \{1,\ldots,k\} \to S \qquad (R,i) \mapsto R + c_i P$$

$$g : S \to S \times \{1,\ldots,k\} \qquad R \mapsto (R, R \bmod k)$$

! injective

↳ no collisions

$$H : S \times \{1,\ldots,k\} \to S \times \{1,\ldots,k\} \qquad (R,i) \mapsto g(F(R,i))$$

↳ Every collision of H is a collision of F. 😺

To sum up

g is injective ⟶ no collisions

To sum up

We did this by using the current element of S, not just to decide where to walk next, but also in the computation of the next step.

→ g is injective → no collisions

To sum up

We were able to do this because we are able to walk between elements of S while making sure we do not step out of S.

↳ We did this by using the current element of S, not just to decide where to walk next, but also in the computation of the next step.

↳ g is injective → no collisions

To sum up

We are able to do this simply because we are using the entire domain

↳ We were able to do this because we are able to walk between elements of S while making sure we do not step out of S.

↳ We did this by using the current element of S, not just to decide where to walk next, but also in the computation of the next step.

↳ g is injective → no collisions

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The starting points:

→ Choose randomly between E_0 and E_1

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The starting points:

- Choose randomly between E_0 and E_1
- Take a random ℓ^e-isogeny from the chosen E_j (this length is the diameter of the graph, so it should lead us to a uniformly random curve)

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The starting points:

- Choose randomly between E_0 and E_1
- Take a random ℓ^e-isogeny from the chosen E_j (this length is the diameter of the graph, so it should lead us to a uniformly random curve)

The vOW walk:

- $f(j_i) = \Phi_\ell(j_{i-1}, j_i, j_i \mod \ell)$
  - The modular polynomial
  - The previous j-invariant, so that we can avoid backtracking
  - The choice of a root

Why $\sqrt{}$?

- The Meet-in-the-middle algorithm
- The van Oorschot and Wiener algorithm
- The Delfs-Galbraith algorithm

Why $\sqrt{}$?

→ The Meet-in-the-middle algorithm

  ↳ because it is the *middle* aka. half of the length/exponent.

→ The van Oorschot and Wiener algorithm

→ The Delfs-Galbraith algorithm

Why $\sqrt{}$?

→ The Meet-in-the-middle algorithm

↳ because it is the *middle* aka. half of the length/exponent.

→ The van Oorschot and Wiener algorithm

↳ because of the birthday bound.

→ The Delfs-Galbraith algorithm

Why $\sqrt{}$?

→ The Meet-in-the-middle algorithm

↳ because it is the *middle* aka. half of the length/exponent.

→ The van Oorschot and Wiener algorithm

↳ because of the birthday bound.

→ The Delfs-Galbraith algorithm

↳ because of the ratio between $\mathbb{F}_p$-rational curves and all curves.

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The starting points:

- Choose randomly between E_0 and E_1
- Take a random ℓ^e-isogeny from the chosen E_j (this length is the diameter of the graph, so it should lead us to a uniformly random curve)

The vOW walk:

- $f(j_i) = \Phi_\ell(j_{i-1}, j_i, j_i \mod \ell)$
  - The modular polynomial
  - The previous j-invariant, so that we can avoid backtracking
  - The choice of a root

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The sta

The vC

The

To sum up

→ We have a user-independent walk.

→ We can use cycles to recover keys.

→ We can avoid useless collisions.

No need for an infiltrator.

We have an exact count on the number of collisions.

We recover all secret keys.

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The starting points:

- Choose randomly between E_0 and E_1
- Take a random ℓ^e-isogeny from the chosen E_j (this length is the diameter of the graph, so it should lead us to a uniformly random curve)

The vOW walk:

- $f(j_i) = \Phi_\ell(j_{i-1}, j_i, j_i \mod \ell)$
  - The modular polynomial
  - The previous j-invariant, so that we can avoid backtracking
  - The choice of a root

A walk for pathfinding in the $\mathbb{F}_{p^2}$ graph

(when $p = c\ell^e - 1$, for some small prime ℓ and a small factor c)

The starting points:

- Choose randomly between E_0 and E_1
- Take a random ℓ^e-isogeny from the chosen E_j (this length is the diameter of the graph, so it should lead us to a uniformly random curve)

The vOW walk:

- $f(j_i) = \Phi_\ell(j_{i-1}, j_i, j_i \mod \ell)$
  - The modular polynomial
  - The previous j-invariant, so that we can avoid backtracking
  - The choice of a root

User-independent walk ✓

A walk for pathfinding in the $\mathbb{F}_p$ graph

(when $p = 4 \cdot \ell_1 \cdots \ell_n - 1$, for some small primes $\ell_1 \cdots \ell_n$)

A walk for pathfinding in the $\mathbb{F}_p$ graph

(when $p = 4 \cdot \ell_1 \cdots \ell_n - 1$, for some small primes $\ell_1 \cdots \ell_n$)

The starting points:

→ Choose randomly between E_0 and E_1

A walk for pathfinding in the $\mathbb{F}_p$ graph

(when $p = 4 \cdot \ell_1 \cdots \ell_n - 1$, for some small primes $\ell_1 \cdots \ell_n$)

The starting points:

- Choose randomly between E_0 and E_1
- Apply a random full CSIDH key to the chosen E_j (so again, it should lead us to a uniformly random curve)

A walk for pathfinding in the $\mathbb{F}_p$ graph

(when $p = 4 \cdot \ell_1 \cdots \ell_n - 1$, for some small primes $\ell_1 \cdots \ell_n$)

The starting points:

- Choose randomly between E_0 and E_1
- Apply a random full CSIDH key to the chosen E_j (so again, it should lead us to a uniformly random curve)

The vOW walk:

- $$f(A) = \begin{cases} M(E_A/K_{\ell_1}) & \text{if } A \equiv 0 \mod n \\ M(E_A/K_{\ell_2}) & \text{if } A \equiv 1 \mod n \\ \vdots & \\ M(E_A/K_{\ell_n}) & \text{if } A \equiv n-1 \mod n\,. \end{cases}$$

Taking the Montgomery coefficient

The ℓ_n-order subgroup

A walk for pathfinding in the $\mathbb{F}_p$ graph

(when $p = 4 \cdot \ell_1 \cdots \ell_n - 1$, for some small primes $\ell_1 \cdots \ell_n$)

The starting points:

- Choose randomly between E_0 and E_1
- Apply a random full CSIDH key to the chosen E_j (so again, it should lead us to a uniformly random curve)

The vOW walk:

- $$f(A) = \begin{cases} M(E_A/K_{\ell_1}) & \text{if } A \equiv 0 \mod n \\ M(E_A/K_{\ell_2}) & \text{if } A \equiv 1 \mod n \\ \vdots & \\ M(E_A/K_{\ell_n}) & \text{if } A \equiv n-1 \mod n\,. \end{cases}$$

Taking the Montgomery coefficient

The ℓ_n-order subgroup

User-independent walk ✓

Multi-user isogeny pathfinding

What about cycles ?

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

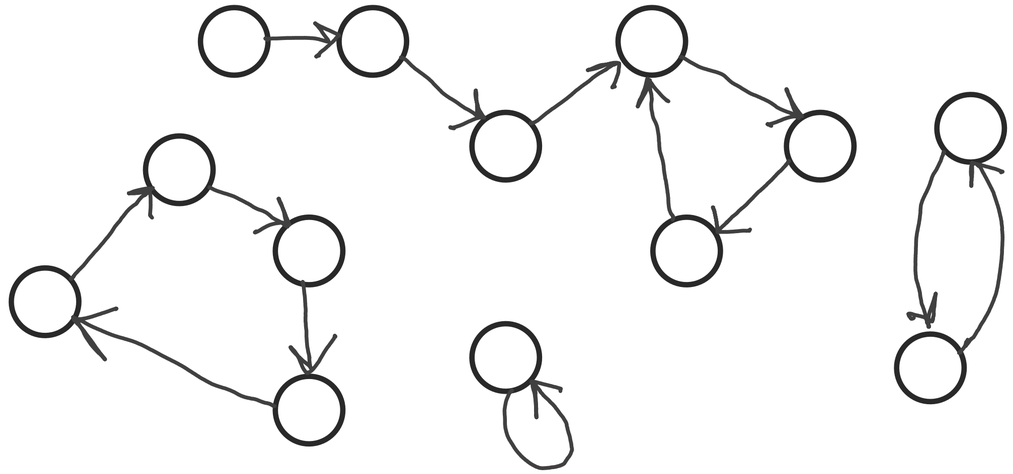
! We need at least two.

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

! We need at least two.

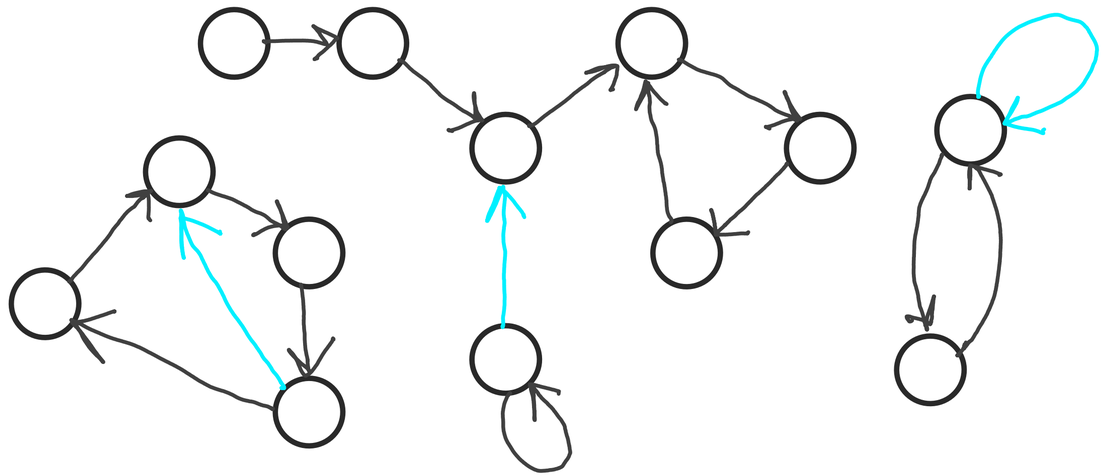


What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

! We need at least two.



→ Now, we add one more collision per component.

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

! We need at least two.

→ After adding L edges, we have approximately $\log(L)$ connected components, each containing a cycle.

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

! We need at least two.

→ After adding L edges, we have approximately $\log(L)$ connected components, each containing a cycle.

↳ $\sim L + \log(L)$ edges for connected components with two cycles (with this strategy).

What about cycles ?

1. In the $\mathbb{F}_{p^2}$ case: ✓

Cycles in the multi-user graph correspond to cycles in the isogeny graph, so they correspond to endomorphisms.

! We need at least two.

→ After adding L edges, we have approximately $\log(L)$ connected components, each containing a cycle.

↳ $\sim L + \log(L)$ edges for connected components with two cycles (with this strategy).

2. In the $\mathbb{F}_p$ case: ✗

A cycle is yet another $\mathbb{F}_p$-rational endomorphism.

From nb. of collisions to complexity

Theorem. Let S be a finite set of cardinality N, let $f : S \to S$ be an efficiently-computable semi-random function. The expected number of steps of a random walk using the f mapping before k collisions occur is asymptotic to

$$\sqrt{2kN}.$$

From nb. of collisions to complexity

Theorem. Let S be a finite set of cardinality N, let $f : S \to S$ be an efficiently-computable semi-random function. The expected number of steps of a random walk using the f mapping before k collisions occur is asymptotic to

$$\sqrt{2kN}.$$

→ Multi-user ECDLP: L collisions

$$\mathcal{O}(\sqrt{L}\ \text{😺}).$$

From nb. of collisions to complexity

Theorem. Let S be a finite set of cardinality N, let $f : S \to S$ be an efficiently-computable semi-random function. The expected number of steps of a random walk using the f mapping before k collisions occur is asymptotic to

$$\sqrt{2kN}.$$

→ Multi-user ECDLP: L collisions

$$\mathcal{O}(\sqrt{L}\ \text{😺}).$$

→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions

$$\mathcal{O}(\sqrt{L + \log(L)}\ \text{😺}).$$

From nb. of collisions to complexity

Theorem. Let S be a finite set of cardinality N, let $f : S \to S$ be an efficiently-computable semi-random function. The expected number of steps of a random walk using the f mapping before k collisions occur is asymptotic to

$$\sqrt{2kN}.$$

→ Multi-user ECDLP: L collisions

$$\mathcal{O}(\sqrt{L}\ \text{😺}).$$

→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions

$$\mathcal{O}(\sqrt{L + \log(L)}\ \text{😺}).$$

→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions

$$\mathcal{O}(\sqrt{L}\ \text{😺}).$$

To sum up (for the last time)

- Multi-user ECDLP: L collisions

$$\mathcal{O}(\sqrt{L}\ \text{🐱}).$$

- Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions

$$\mathcal{O}(\sqrt{L + \log(L)}\ \text{😼}).$$

- Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions

$$\mathcal{O}(\sqrt{L}\ \text{🐱}).$$

To sum up (for the last time)

Requires an infiltrator All keys recovered

- Multi-user ECDLP: L collisions

$$\mathcal{O}(\sqrt{L}\ \text{🐱}).$$

- Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions

$$\mathcal{O}(\sqrt{L + \log(L)}\ \text{🐱}).$$

- Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions

$$\mathcal{O}(\sqrt{L}\ \text{🐱}).$$

To sum up (for the last time)

Requires an infiltrator All keys recovered

- Multi-user ECDLP: L collisions

$$\mathcal{O}(\sqrt{L}\ \text{🐱}).$$

✘

- Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions

$$\mathcal{O}(\sqrt{L + \log(L)}\ \text{😼}).$$

- Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions

$$\mathcal{O}(\sqrt{L}\ \text{🐱}).$$

To sum up (for the last time)

	Requires an infiltrator	All keys recovered
→ Multi-user ECDLP: L collisions		
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✘	
→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions		
$\mathcal{O}(\sqrt{L + \log(L)}$ 😺$)$.	✘	
→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions		
$\mathcal{O}(\sqrt{L}$ 😺$)$.		

To sum up (for the last time)

	Requires an infiltrator	All keys recovered
→ Multi-user ECDLP: L collisions		
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✗	
→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions		
$\mathcal{O}(\sqrt{L + \log(L)}$ 😺$)$.	✗	
→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions		
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✓	

To sum up (for the last time)

	Requires an infiltrator	All keys recovered
→ Multi-user ECDLP: L collisions $\mathcal{O}(\sqrt{L}$ 😺$)$.	✗	✓
→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions $\mathcal{O}(\sqrt{L + \log(L)}$ 😺$)$.	✗	
→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions $\mathcal{O}(\sqrt{L}$ 😺$)$.	✓	

To sum up (for the last time)

	Requires an infiltrator	All keys recovered
→ Multi-user ECDLP: L collisions		
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✗	✓
→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions		
$\mathcal{O}(\sqrt{L + \log(L)}$ 😺$)$.	✗	✓
→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions		
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✓	

To sum up (for the last time)

	Requires an infiltrator	All keys recovered	
→ Multi-user ECDLP: L collisions			
$\mathcal{O}(\sqrt{L}$ 😼$)$.	✗	✓	
→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions			
$\mathcal{O}(\sqrt{L + \log(L)}$ 😼$)$.	✗	✓	
→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions			
$\mathcal{O}(\sqrt{L}$ 😼$)$.	✓	✗	$\sim 0.98L$

To sum up (for the last time)

Thank you !

	uires an infiltrator	All keys recovered	
→ Multi-user ECDLP: L collisions			
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✗	✓	
→ Multi-user isogeny pathfinding in $\mathbb{F}_{p^2}$: $L + \log(L)$ collisions			
$\mathcal{O}(\sqrt{L + \log(L)}$ 😺$)$.	✗	✓	
→ Multi-user isogeny pathfinding in $\mathbb{F}_p$: $2L$ collisions			
$\mathcal{O}(\sqrt{L}$ 😺$)$.	✓	✗	$\sim 0.98L$