CSIDH:
An Efficient Post-Quantum Commutative Group Action

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

[1]KU Leuven [2]TU Eindhoven [3]Radboud Universiteit

Crypto Working Group, Utrecht, 14 September 2018


[ˈsiːˌsaɪd]

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 (w/ reasonable speed)

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- Non-interactive key exchange (full public-key validation); previously an open problem post-quantumly (w/ reasonable speed)
- 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 (w/ reasonable speed)
- Small keys: 64 bytes at conjectured AES-128 security level
- Competitive speed: $\sim$ 85 ms for a full key exchange

Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- Non-interactive key exchange (full public-key validation); previously an open problem post-quantumly (w/ reasonable speed)
- Small keys: 64 bytes at conjectured AES-128 security level
- Competitive speed: $\sim 85\,\mathrm{ms}$ for a full key exchange
- Flexible: compatible with 0-RTT protocols such as QUIC; recent preprint uses CSIDH for 'SeaSign' signatures

Post-quantum Diffie-Hellman?

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

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

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.

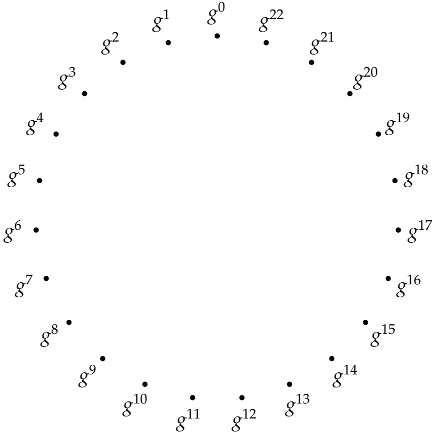
$\rightsquigarrow$ Idea:

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

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

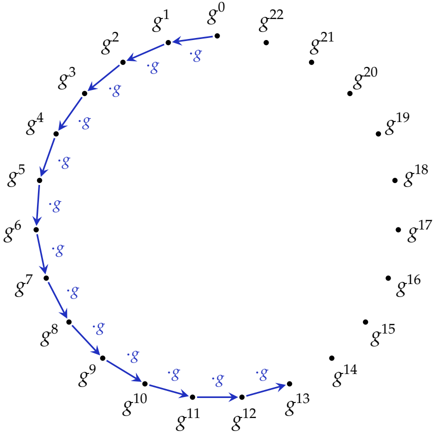
Square-and-multiply

Suppose $G \cong \mathbb{Z}/23$ and that Alice computes g^{13}.



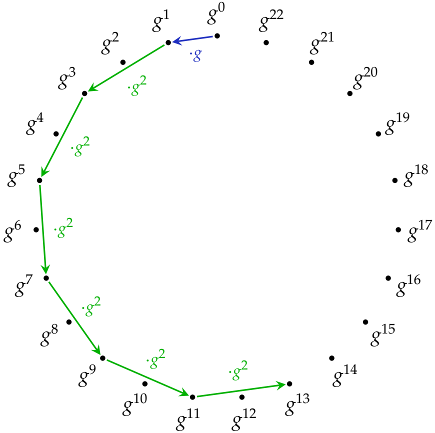
Square-and-multiply

Suppose $G \cong \mathbb{Z}/23$ and that Alice computes g^{13}.



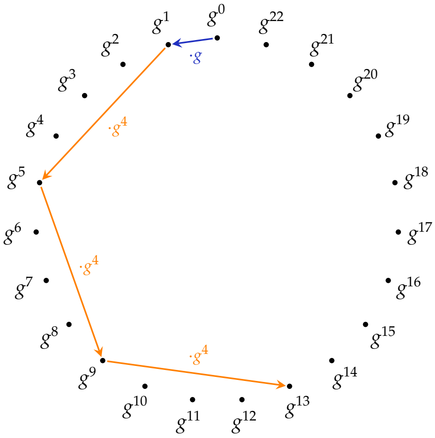
Square-and-multiply

Suppose $G \cong \mathbb{Z}/23$ and that Alice computes g^{13}.



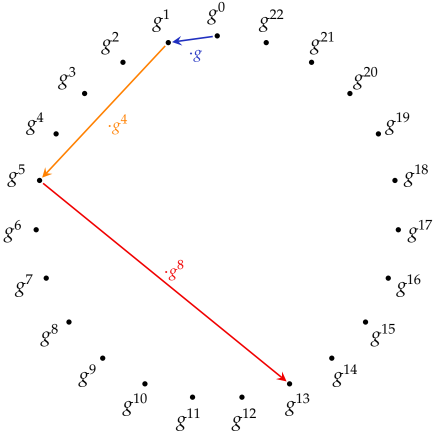
Square-and-multiply

Suppose $G \cong \mathbb{Z}/23$ and that Alice computes g^{13}.

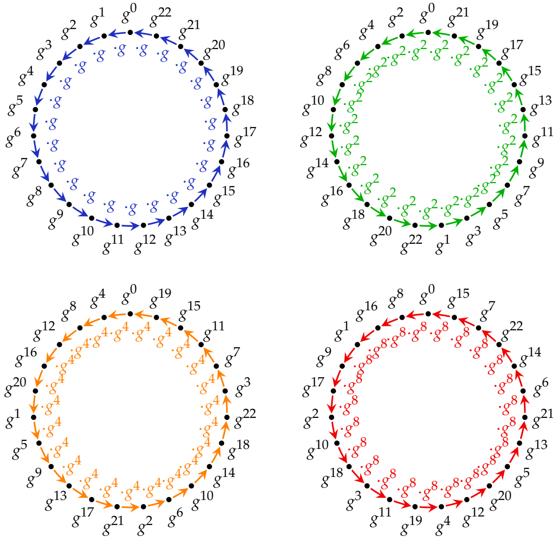


Square-and-multiply

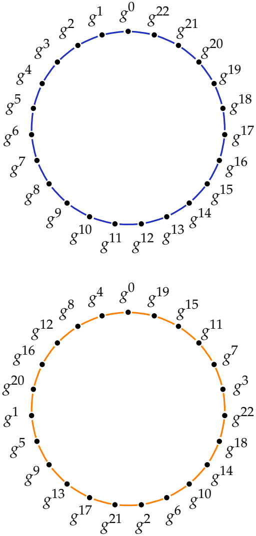
Suppose $G \cong \mathbb{Z}/23$ and that Alice computes g^{13}.



Cycles!

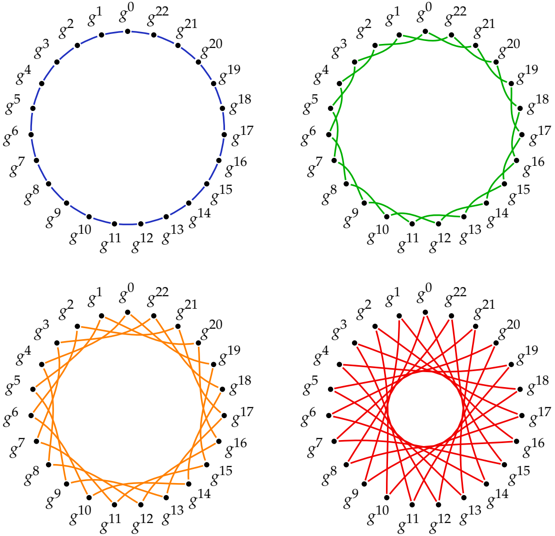


Cycles!

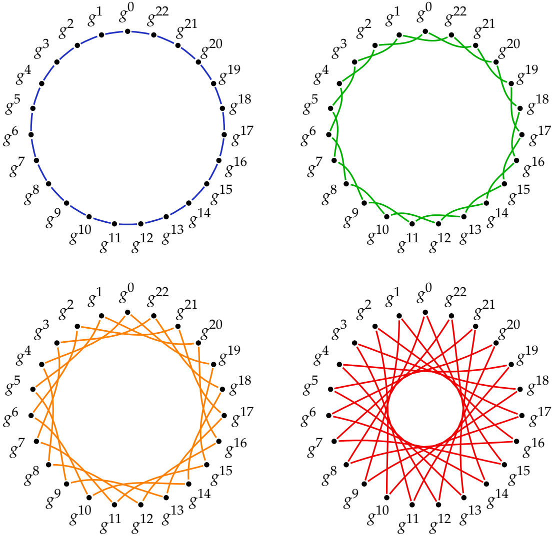


g^{0}
g^{22}
g^{21}
g^{20}
g^{19}
g^{18}
g^{17}
g^{16}
g^{15}
g^{14}
g^{13}
g^{12}
g^{11}
g^{10}
g^{9}
g^{8}
g^{7}
g^{6}
g^{5}
g^{4}
g^{3}
g^{2}
g^{1}
g^{0}
g^{21}
g^{19}
g^{17}
g^{15}
g^{13}
g^{11}
g^{9}
g^{7}
g^{5}
g^{3}
g^{1}
g^{22}
g^{20}
g^{18}
g^{16}
g^{14}
g^{12}
g^{10}
g^{8}
g^{6}
g^{4}
g^{2}
g^{0}
g^{19}
g^{15}
g^{11}
g^{7}
g^{3}
g^{22}
g^{18}
g^{14}
g^{10}
g^{6}
g^{2}
g^{21}
g^{17}
g^{13}
g^{9}
g^{5}
g^{1}
g^{20}
g^{16}
g^{12}
g^{8}
g^{4}
g^{0}
g^{15}
g^{7}
g^{22}
g^{14}
g^{6}
g^{21}
g^{13}
g^{5}
g^{20}
g^{12}
g^{4}
g^{19}
g^{11}
g^{3}
g^{18}
g^{10}
g^{2}
g^{17}
g^{9}
g^{1}
g^{16}
g^{8}

Cycles!

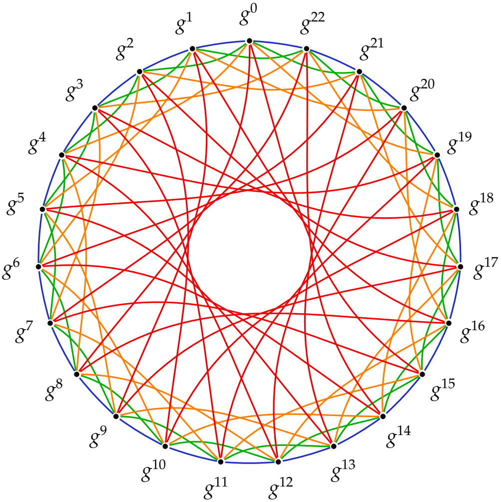

g^0 g^1 g^2 g^3 g^4 g^5 g^6 g^7 g^8 g^9 g^{10} g^{11} g^{12} g^{13} g^{14} g^{15} g^{16} g^{17} g^{18} g^{19} g^{20} g^{21} g^{22}
g^0 g^1 g^2 g^3 g^4 g^5 g^6 g^7 g^8 g^9 g^{10} g^{11} g^{12} g^{13} g^{14} g^{15} g^{16} g^{17} g^{18} g^{19} g^{20} g^{21} g^{22}
g^0 g^1 g^2 g^3 g^4 g^5 g^6 g^7 g^8 g^9 g^{10} g^{11} g^{12} g^{13} g^{14} g^{15} g^{16} g^{17} g^{18} g^{19} g^{20} g^{21} g^{22}
g^0 g^1 g^2 g^3 g^4 g^5 g^6 g^7 g^8 g^9 g^{10} g^{11} g^{12} g^{13} g^{14} g^{15} g^{16} g^{17} g^{18} g^{19} g^{20} g^{21} g^{22}

Cycles!

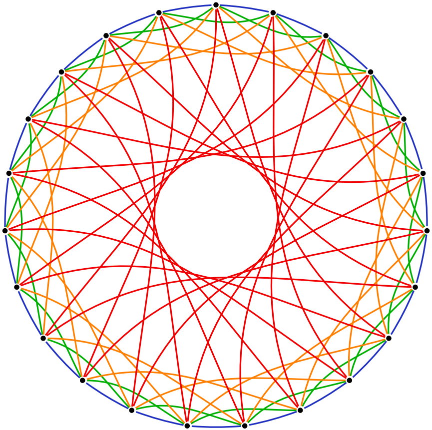


Cycles are compatible: [right, then left] = [left, then right], etc.

Union of cycles: rapid mixing

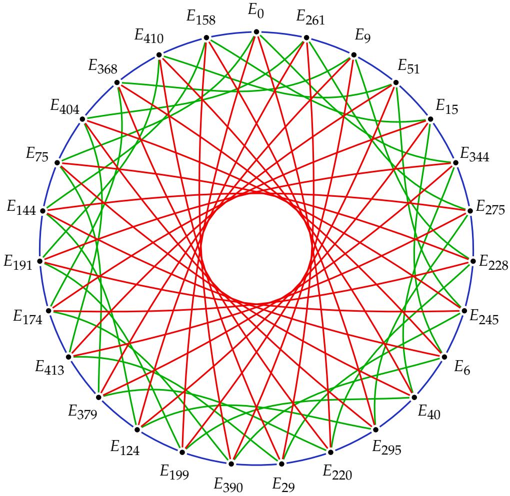


Union of cycles: rapid mixing

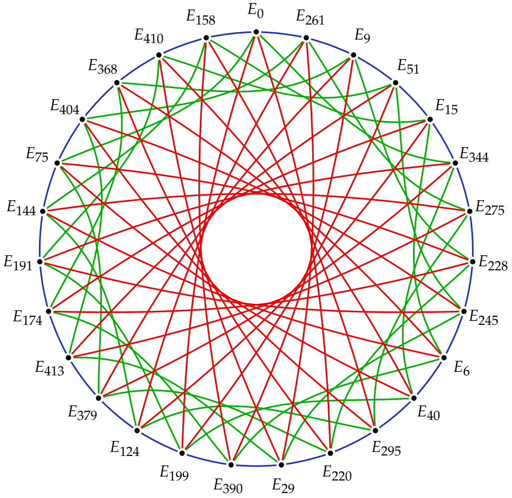


CSIDH: Nodes are now elliptic curves and edges are isogenies.

Graphs of elliptic curves

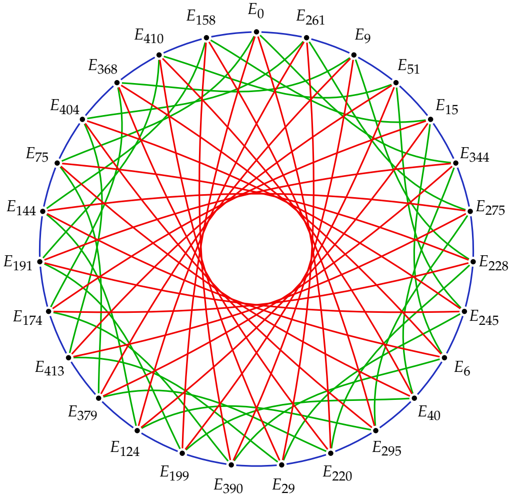


Graphs of elliptic curves



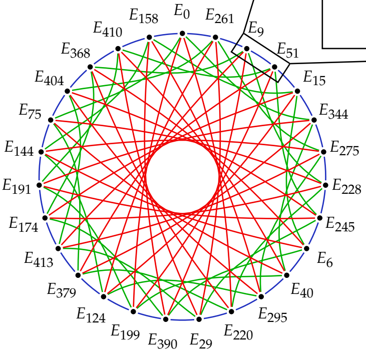
Nodes: Supersingular curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.

Graphs of elliptic curves

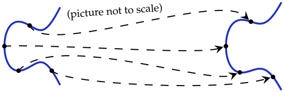


Nodes: Supersingular curves $E_A : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_{419}$.
Edges: 3-, 5-, and 7-isogenies (certain kinds of maps).

Graphs of elliptic curves



A 3-isogeny



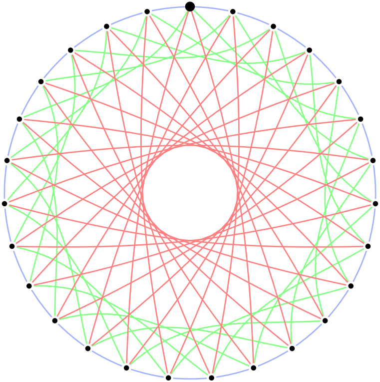
$$E_{51}\colon y^2 = x^3 + 51x^2 + x \longrightarrow E_9\colon y^2 = x^3 + 9x^2 + x$$

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

Diffie-Hellman on 'nice' graphs

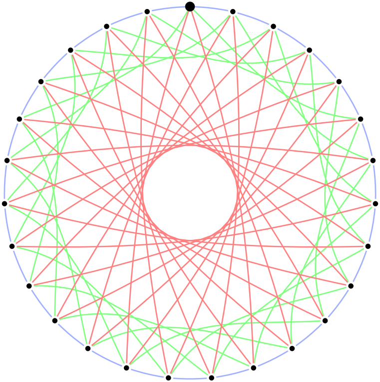
Alice

$[+, -, +, -]$



Bob

$[+, +, -, +]$

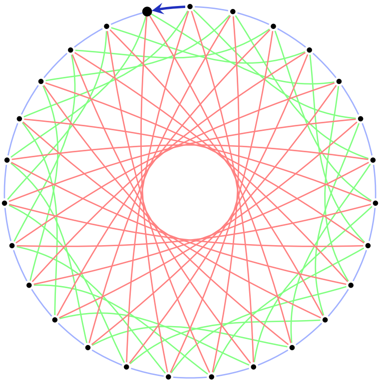


Diffie-Hellman on 'nice' graphs

Alice

$[+, -, +, -]$

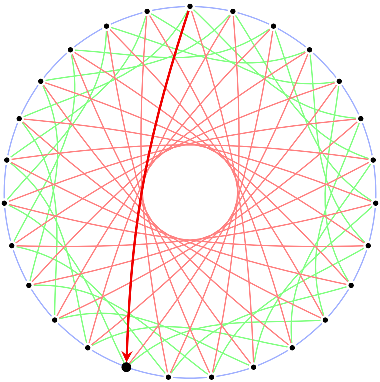
$\uparrow$



Bob

$[+, +, -, +]$

$\uparrow$



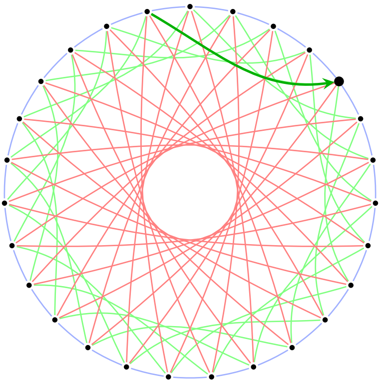
Diffie-Hellman on 'nice' graphs

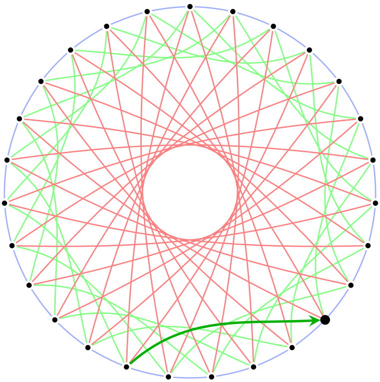
Alice

$[+, -, +, -]$

Bob

$[+, +, -, +]$





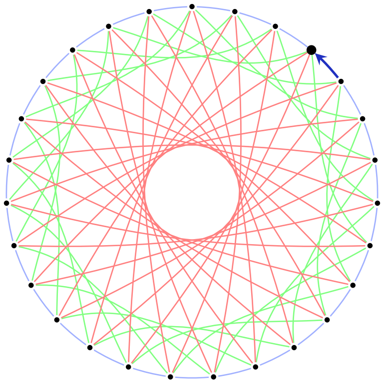
Diffie-Hellman on 'nice' graphs

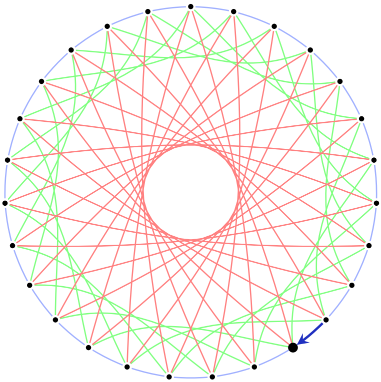
Alice

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

Bob

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

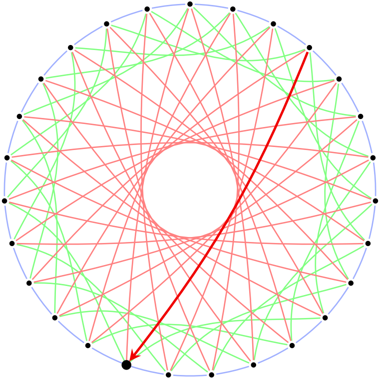




Diffie-Hellman on 'nice' graphs

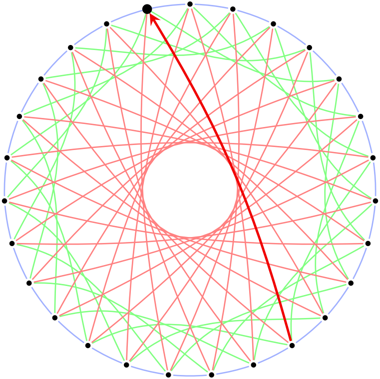
Alice

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



Bob

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



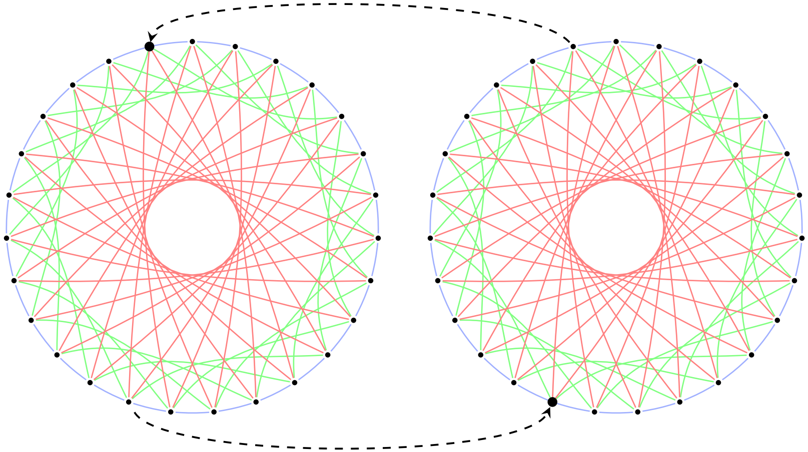
Diffie-Hellman on ‘nice’ graphs

Alice

$[+, -, +, -]$

Bob

$[+, +, -, +]$



Diffie-Hellman on ‘nice’ graphs

Alice

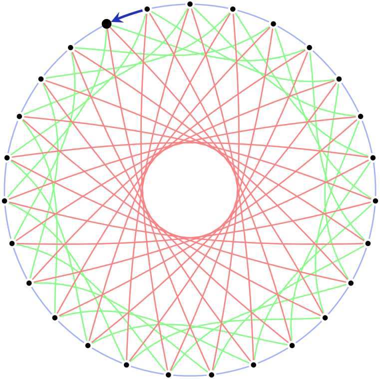
$[+, -, +, -]$

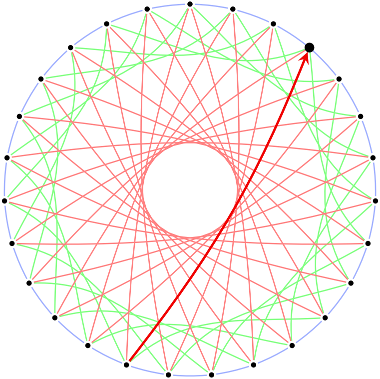
$\uparrow$

Bob

$[+, +, -, +]$

$\uparrow$





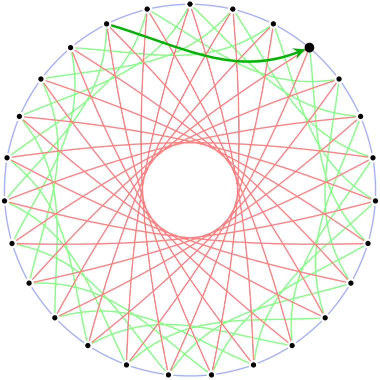
Diffie-Hellman on ‘nice’ graphs

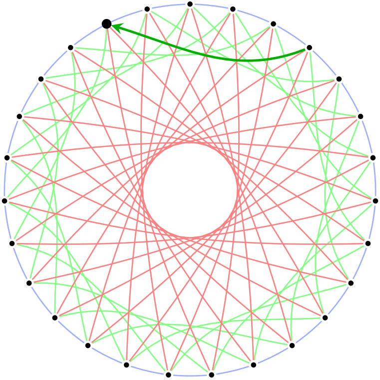
Alice

$[+, -, +, -]$

Bob

$[+, +, -, +]$

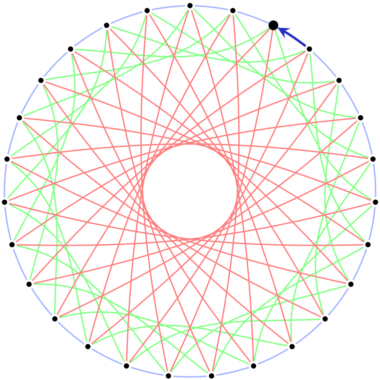




Diffie-Hellman on ‘nice’ graphs

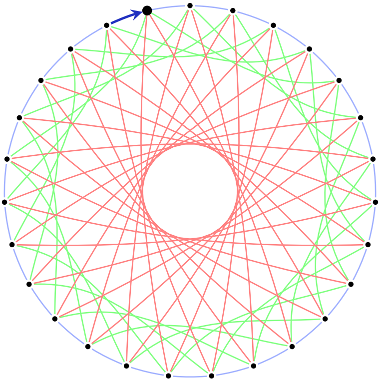
Alice

$[+, -, +, -]$



Bob

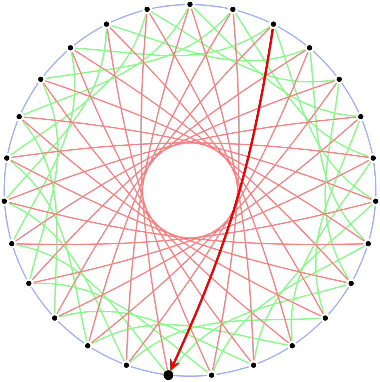
$[+, +, -, +]$



Diffie-Hellman on ‘nice’ graphs

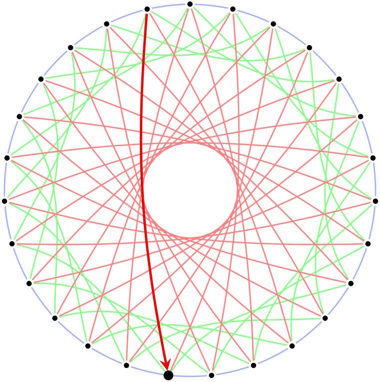
Alice

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



Bob

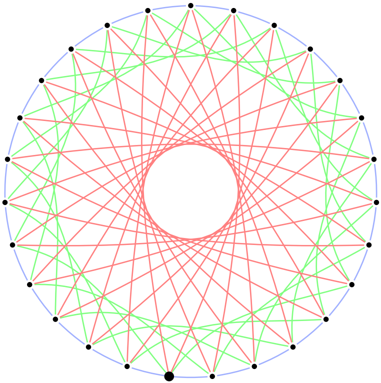
$[+, +, -, \underset{\uparrow}{+}]$



Diffie-Hellman on 'nice' graphs

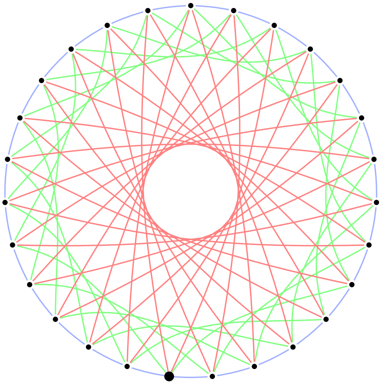
Alice

[+, −, +, −]



Bob

[+, +, −, +]



A walkable graph

Important properties for such a walk:

IP1 ▸ The graph is a composition of compatible cycles.

IP2 ▸ We can compute neighbours in given directions.

IP1: A composition of cycles

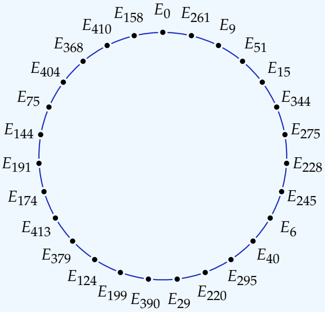
- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

IP1: A composition of cycles

- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

- In our example, these are

G_3:

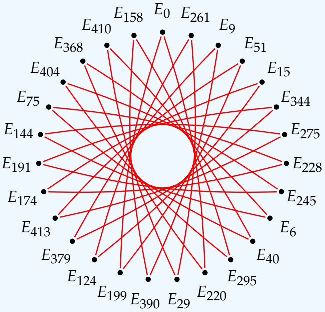


IP1: A composition of cycles

- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

- In our example, these are

G_5:

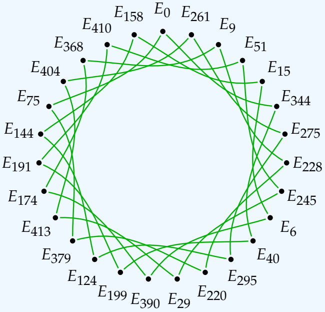


IP1: A composition of cycles

- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

- In our example, these are

G_7:

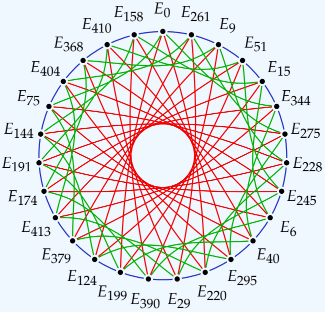


IP1: A composition of cycles

- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

- In our example, these are

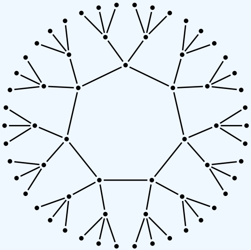
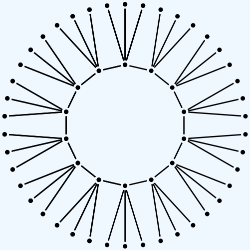
$G_3 \cup G_5 \cup G_7$:



IP1: A composition of cycles

- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

- Generally, the G_ℓ look something like

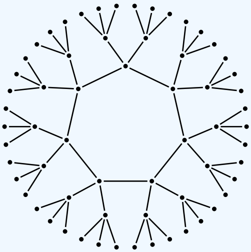
G_3:  G_5: 

IP1: A composition of cycles

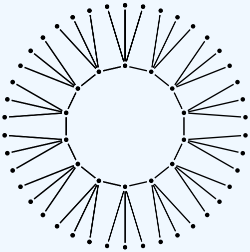
- The graph used in CSIDH is constructed as a composition of graphs G_ℓ of 'ℓ-isogenies'.

- Generally, the G_ℓ look something like

G_3:



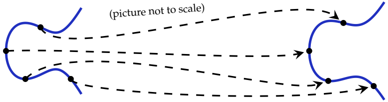
G_5:



- We want to make sure G_ℓ is just a cycle.

IP2: Compute neighbours in given directions

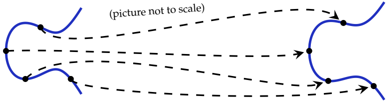
The edges of G_ℓ are ℓ-isogenies.



$$E_{51} : y^2 = x^3 + 51x^2 + x \longrightarrow E_9 : y^2 = x^3 + 9x^2 + x$$
$$(x, y) \longmapsto \left(\frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, y \cdot \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right)$$

IP2: Compute neighbours in given directions

The edges of G_ℓ are ℓ-isogenies.

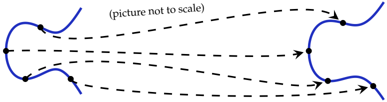


$$\begin{aligned} E_{51} : y^2 = x^3 + 51x^2 + x &\longrightarrow E_9 : y^2 = x^3 + 9x^2 + x \\ (x, y) &\longmapsto \left(\frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, y \cdot \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right) \end{aligned}$$

- The orientation of G_ℓ is mathematically well-defined (canonical way to compute the 'left' or 'right' isogeny).

IP2: Compute neighbours in given directions

The edges of G_ℓ are ℓ-isogenies.

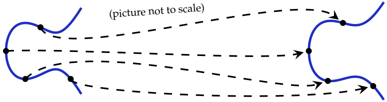


$$\begin{aligned} E_{51} : y^2 = x^3 + 51x^2 + x &\longrightarrow E_9 : y^2 = x^3 + 9x^2 + x \\ (x, y) &\longmapsto \left(\frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, y \cdot \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right) \end{aligned}$$

- The orientation of G_ℓ is mathematically well-defined (canonical way to compute the 'left' or 'right' isogeny).
- The cost grows with ℓ $\rightsquigarrow$ want small ℓ.

IP2: Compute neighbours in given directions

The edges of G_ℓ are ℓ-isogenies.



$$\begin{aligned} E_{51} : y^2 = x^3 + 51x^2 + x \quad &\longrightarrow \quad E_9 : y^2 = x^3 + 9x^2 + x \\ (x, y) \quad &\longmapsto \quad \left(\frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, y \cdot \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right) \end{aligned}$$

- The orientation of G_ℓ is mathematically well-defined (canonical way to compute the 'left' or 'right' isogeny).
- The cost grows with $\ell \rightsquigarrow$ want small ℓ.
- Generally needs big extension fields...

Point counting

Both ‘IP’s are connected to the number of points on the curves.

Point counting

Both ‘IP’s are connected to the number of points on the curves.

It seems difficult to find a curve with a given number of points (and such that the graph is big). [De Feo–Kieffer–Smith]

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1. ▶ Choose some small odd primes $\ell_1, \ldots, \ell_n$.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
- Choose some small odd primes $\ell_1, \ldots, \ell_n$.
- Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
- Choose some small odd primes $\ell_1, \ldots, \ell_n$.
- Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
- Fix the curve $E_0 \colon y^2 = x^3 + x$ over $\mathbb{F}_p$.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1. - Choose some small odd primes $\ell_1, \ldots, \ell_n$.
   - Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
   - Fix the curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$.

2. ~~magic~~ math happens!

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
   - Choose some small odd primes $\ell_1, \ldots, \ell_n$.
   - Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
   - Fix the curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$.

2.  ~~magic~~ math happens!

3.
   - E_0 has $p + 1$ points.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
   - Choose some small odd primes $\ell_1, \ldots, \ell_n$.
   - Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
   - Fix the curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$.

2.  ~~magic~~ math happens!

3.
   - E_0 has $p + 1$ points.
   - Let the nodes of G_{ℓ_i} be those E_A with $p + 1$ points.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
   - Choose some small odd primes $\ell_1, \ldots, \ell_n$.
   - Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
   - Fix the curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$.

2. ~~magic~~ math happens!

3.
   - E_0 has $p + 1$ points.
   - Let the nodes of G_{ℓ_i} be those E_A with $p + 1$ points.
   - Then every G_{ℓ_i} is a disjoint union of cycles.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
   - Choose some small odd primes $\ell_1, \dots, \ell_n$.
   - Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
   - Fix the curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$.

2. ~~magic~~ math happens!

3.
   - E_0 has $p + 1$ points.
   - Let the nodes of G_{ℓ_i} be those E_A with $p + 1$ points.
   - Then every G_{ℓ_i} is a disjoint union of cycles.
   - All G_{ℓ_i} are compatible.

Salt water (CSIDH, get it?) is a solution (do bad chemistry jokes belong in crypto talks?)

1.
- Choose some small odd primes $\ell_1, \ldots, \ell_n$.
- Make sure $p = 4 \cdot \ell_1 \cdots \ell_n - 1$ is prime.
- Fix the curve $E_0 \colon y^2 = x^3 + x$ over $\mathbb{F}_p$.

2. ~~magic~~ math happens!

3.
- E_0 has $p + 1$ points.
- Let the nodes of G_{ℓ_i} be those E_A with $p + 1$ points.
- Then every G_{ℓ_i} is a disjoint union of cycles.
- All G_{ℓ_i} are compatible.
- Computations need only $\mathbb{F}_p$-arithmetic.

Representing nodes of the graph

Side effect of magic:

- Every node of G_{ℓ_i} can be written as

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

Representing nodes of the graph

Side effect of magic:

- Every node of G_{ℓ_i} can be written as

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

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

Representing nodes of the graph

Side effect of magic:

- Every node of G_{ℓ_i} can be written as

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

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

$\Rightarrow$ Tiny keys!

Does any A work?

[1] 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.

[1] 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.

[1] 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 $[p+1]P = \infty$.[1]

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

Security

Classical:

- Meet-in-the-middle variants: Time $O(\sqrt[4]{p})$.

Security

Classical:

- Meet-in-the-middle variants: Time $O(\sqrt[4]{p})$.

Quantum:

- Hidden-shift algorithms: Subexponential complexity.

Security

Classical:

- Meet-in-the-middle variants: Time $O(\sqrt[4]{p})$.

Quantum:

- Hidden-shift algorithms: Subexponential complexity.
  - Literature contains mostly asymptotics.
  - Time-space trade-off: Fastest variants need huge memory.
  - Concrete estimates are to be done.

Security

Classical:

- Meet-in-the-middle variants: Time $O(\sqrt[4]{p})$.

Quantum:

- Hidden-shift algorithms: Subexponential complexity.
  - Literature contains mostly asymptotics.
  - Time-space trade-off: Fastest variants need huge memory.
  - Concrete estimates are to be done.
  - (Recent preprint [BS] ignores much of the cost!)

Parameters

CSIDH-log p	target NIST level	public key size	private key size	time (full exchange)	cycles (full exchange)	stack memory	classical security	quantum security claimed by [BS] (take *cum grano salis*)
CSIDH-512	1	64 b	32 b	85 ms	212e6	4368 b	128	71
CSIDH-1024	3	128 b	64 b				256	88
CSIDH-1792	5	224 b	112 b				448	104

Work in progress & future work

- Fast and constant-time implementation

Work in progress & future work

- Fast and constant-time implementation
- Reliable security analysis

Work in progress & future work

- Fast and constant-time implementation
- Reliable security analysis
- More applications

Work in progress & future work

- Fast and constant-time implementation
- Reliable security analysis
- More applications
- [Your paper here!]


Thank you!

References

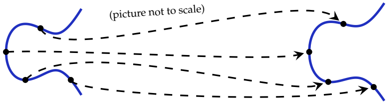
Mentioned in this talk:

- Castryck, Lange, Martindale, Panny, Renes:
  CSIDH: An Efficient Post-Quantum Commutative Group Action
  `https://ia.cr/2018/383` (to appear at ASIACRYPT 2018)
- De Feo, Kieffer, Smith:
  Towards practical key exchange from ordinary isogeny graphs
  `https://ia.cr/2018/485` (to appear at ASIACRYPT 2018)
- De Feo, Galbraith:
  SeaSign: Compact isogeny signatures from class group actions
  `https://ia.cr/2018/824`
- [BS] Bonnetain, Schrottenloher:
  Quantum Security Analysis of CSIDH and Ordinary Isogeny-based Schemes
  `https://ia.cr/2018/537`

Other related work:

- Delfs, Galbraith:
  Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$
  `https://arxiv.org/abs/1310.7789`
- Childs, Jao, Soukharev:
  Constructing elliptic curve isogenies in quantum subexponential time
  `https://arxiv.org/abs/1012.4019`
- Meyer, Reith:
  A faster way to the CSIDH
  `https://ia.cr/2018/782`
  (to appear at Indocrypt 2018)
- Jao, LeGrow, Leonardi, Ruiz-Lopez:
  A polynomial quantum space attack on CRS and CSIDH
  (MathCrypt 2018)
- Biasse, Iezzi, Jacobson:
  A note on the security of CSIDH
  `https://arxiv.org/abs/1806.03656`
  (to appear at Indocrypt 2018)

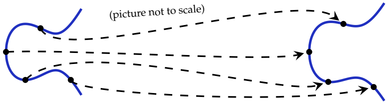
Where's the group?



$$
\begin{aligned}
E_{51} : y^2 = x^3 + 51x^2 + x &\longrightarrow E_9 : y^2 = x^3 + 9x^2 + x \\
(x, y) &\longmapsto \left(\frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, y \cdot \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right)
\end{aligned}
$$

- $E_9 = E_{51}/\mathfrak{a}$ where $\mathfrak{a}$ is the ideal $(3, \pi - 1)$ of $\mathrm{End}_{\mathbb{F}_p}(E_{51})$.

Where's the group?



$$\begin{aligned} E_{51} : y^2 = x^3 + 51x^2 + x &\longrightarrow E_9 : y^2 = x^3 + 9x^2 + x \\ (x, y) &\longmapsto \left(\frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, y \cdot \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right) \end{aligned}$$

- $E_9 = E_{51}/\mathfrak{a}$ where $\mathfrak{a}$ is the ideal $(3, \pi - 1)$ of $\mathrm{End}_{\mathbb{F}_p}(E_{51})$.
- For our choices of A, $\mathrm{End}_{\mathbb{F}_p}(E_A) \cong \mathbb{Z}[\sqrt{-p}]$.
- The group action is

$$\begin{aligned} \mathrm{cl}(\mathbb{Z}[\sqrt{-p}]) \times \{E_A\} &\longrightarrow \{E_A\} \\ ([\mathfrak{a}], E) &\longmapsto E/\mathfrak{a} \end{aligned}$$

(modulo details).