Quantum attacks on CSIDH: an overview

Chloe Martindale
University of Bristol
Based on joint work with
Daniel J. Bernstein, Tanja Lange, and Lorenz Panny
quantum.isogeny.org
Why CSIDH?

- Drop-in post-quantum replacement for (EC)DH
- **Non-interactive key exchange** (full public-key validation); previously an open problem post-quantumly
- **Smallest** keys of all post-quantum key exchange candidates
- Competitive **speed**: 50-60ms for a full key exchange
CSIDH: a picture

Secret key: path on the graph
Public key: end points of path.
Quantum complexity analysis

Recall Kuperberg’s algorithm from David Jao’s talk.

2011 Kuperberg estimates time complexity $2^{(\sqrt{2}+o(1))\sqrt{\log_2 p}}$, improvement on 2003 Kuperberg: $2^{(1.77+o(1))\sqrt{\log_2 p}}$. 

Main open questions on asymptotics:

▶ Can the power of $\log_2 p$ be reduced?
▶ If not, can the constant $\sqrt{2}$ be improved? (Last improvement: 2011).
▶ If not, what’s the smallest $o(1)$? Important for proposing parameters! (See next talk).
Quantum complexity analysis

Recall Kuperberg’s algorithm from David Jao’s talk.

2011 Kuperberg estimates time complexity $2^{(\sqrt{2}+o(1))\sqrt{\log_2 p}}$, improvement on 2003 Kuperberg: $2^{(1.77+o(1))\sqrt{\log_2 p}}$.

Main open questions on asymptotics:
- Can the power of $\log_2 p$ be reduced?
Recall Kuperberg’s algorithm from David Jao’s talk.

2011 Kuperberg estimates time complexity $2^{(\sqrt{2}+o(1))\sqrt{\log_2 p}}$, improvement on 2003 Kuperberg: $2^{(1.77+o(1))\sqrt{\log_2 p}}$.

Main open questions on asymptotics:

- Can the power of $\log_2 p$ be reduced?
- If not, can the constant $\sqrt{2}$ be improved? (Last improvement: 2011).
Quantum complexity analysis

Recall Kuperberg’s algorithm from David Jao’s talk.

2011 Kuperberg estimates time complexity $2^{(\sqrt{2}+o(1))\sqrt{\log_2 p}}$, improvement on 2003 Kuperberg: $2^{(1.77+o(1))\sqrt{\log_2 p}}$.

Main open questions on asymptotics:

▶ Can the power of $\log_2 p$ be reduced?
▶ If not, can the constant $\sqrt{2}$ be improved? (Last improvement: 2011).
▶ If not, what’s the smallest $o(1)$?
  Important for proposing parameters! (See next talk).
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?

Kuperberg’s attack: many quantum CSIDH queries.
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?

Kuperberg’s attack: many quantum CSIDH queries.

- Not covered in this talk: how many queries needed?
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?

Kuperberg’s attack: many quantum CSIDH queries.

- Not covered in this talk: how many queries needed?
- How is attack affected by occasional errors and non-uniform distributions over the group?
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?

Kuperberg’s attack: many quantum CSIDH queries.
  ▶ Not covered in this talk: how many queries needed?
  ▶ How is attack affected by occasional errors and non-uniform distributions over the group?
  ▶ How expensive is each CSIDH query?
    Studied in joint work with Bernstein, Lange, and Panny.
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?

Kuperberg’s attack: many quantum CSIDH queries.

▶ Not covered in this talk: how many queries needed?
▶ How is attack affected by occasional errors and non-uniform distributions over the group?
▶ How expensive is each CSIDH query?
  Studied in joint work with Bernstein, Lange, and Panny.
▶ What about memory, using parallel AT metric?
  Trade-offs possible: (theoretically) fastest variant uses billions of qubits.
Concrete quantum complexity analysis

What CSIDH key sizes are needed for post-quantum security level $2^{64}$? $2^{96}$? $2^{128}$?

Kuperberg’s attack: many quantum CSIDH queries.

- Not covered in this talk: how many queries needed?
- How is attack affected by occasional errors and non-uniform distributions over the group?
- How expensive is each CSIDH query?

Studied in joint work with Bernstein, Lange, and Panny.

- What about memory, using parallel AT metric? Trade-offs possible: (theoretically) fastest variant uses billions of qubits.
One CSIDH query: isogenies

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

A 3-isogeny

\[ E_{51}: y^2 = x^3 + 51x^2 + x \rightarrow E_9: y^2 = x^3 + 9x^2 + x \]

\[(x, y) \mapsto \left( \frac{97x^3 - 183x^2 + x}{x^2 - 183x + 97}, \frac{133x^3 + 154x^2 - 5x + 97}{-x^3 + 65x^2 + 128x - 133} \right) \]
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the isogeny graph

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

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

Edges: 3-isogenies.
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$. 
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$.
- $P$ has order dividing 420.
- With probability $\frac{2}{3}$, $140 \cdot P$ has order 3.
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.  
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
- With probability $\frac{2}{3}$, $140 \cdot P$ has order 3
- Find map with kernel $= \langle 140 \cdot P \rangle$
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 3-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
- With probability $\frac{2}{3}$, $140 \cdot P$ has order 3
- Find map with kernel $= \langle 140 \cdot P \rangle$
- Image of map is a neighbour
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 5-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
- With probability $\frac{2}{3}$, $140 \cdot P$ has order 3
- Find map with kernel $= \langle 140 \cdot P \rangle$
- Image of map is a neighbour
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 5-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
- With probability $\frac{4}{5}$, $84 \cdot P$ has order 5
- Find map with kernel $= \langle 84 \cdot P \rangle$
- Image of map is a neighbour
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 7-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
- With probability $\frac{4}{5}$, $84 \cdot P$ has order 5
- Find map with kernel $= \langle 84 \cdot P \rangle$
- Image of map is a neighbour
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the 7-isogeny graph

- Recall: $E_{51}/\mathbb{F}_{419} : y^2 = x^3 + 51x^2 + x$.
- Choose a random $\mathbb{F}_{419}$-point $P = (x, y)$ on $E_{51}$
- $P$ has order dividing 420.
- With probability $\frac{6}{7}$, $60 \cdot P$ has order 7
- Find map with kernel $= \langle 60 \cdot P \rangle$
- Image of map is a neighbour
Computing isogenies

Aim: given curve $E_A$, find a neighbour in the $\ell$-isogeny graph

- Recall: $E_A/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$
- Choose a random $\mathbb{F}_p$-point $P = (x, y)$ on $E_A$
- $P$ has order dividing $p + 1$.
- With probability $\frac{\ell-1}{\ell} \cdot \frac{p+1}{\ell} \cdot P$ has order $\ell$.*
- Find map with kernel $\langle \frac{p+1}{\ell} \cdot P \rangle$
- Image of map is a neighbour

* assuming $\ell|(p + 1)$. 
Computing a query

- A query computes paths in superposition.
A query computes paths in superposition.
A path is a sequence of isogenies (of varying degrees).
Computing a query

- A query computes paths in superposition.
- A path is a sequence of isogenies (of varying degrees).
- Larger degree isogenies are more expensive.
Computing a query

- A query computes paths in superposition.
- A path is a sequence of isogenies (of varying degrees).
- Larger degree isogenies are more expensive. Different degrees computed in superposition \(\rightarrow\) bored qubits.
Computing a query

- A query computes paths in superposition.
- A path is a sequence of isogenies (of varying degrees).
- Larger degree isogenies are more expensive. Different degrees computed in superposition $\leadsto$ bored qubits.
- Isogeny computation fails often for small $\ell$. 

[BLMP] Gives many optimizations / more complex variants – trying to mitigate these problems.
Computing a query

- A query computes paths in superposition.
- A path is a sequence of isogenies (of varying degrees).
- Larger degree isogenies are more expensive. Different degrees computed in superposition $\rightarrow$ bored qubits.
- Isogeny computation fails often for small $\ell$. $\rightarrow$ problematic for quantum implementation.
Computing a query

- A query computes paths in superposition.
- A path is a sequence of isogenies (of varying degrees).
- Larger degree isogenies are more expensive. Different degrees computed in superposition ⇝ bored qubits.
- Isogeny computation fails often for small $\ell$. ⇝ problematic for quantum implementation.

[BLMP] Gives many optimizations / more complex variants–trying to mitigate these problems.
Computing a query

[BLMP] provides software to compute a path using basic bit operations: automatic tallies of nonlinear ops (AND, OR) and linear ops (XOR, NOT).
Computing a query

[BLMP] provides software to compute a path using basic bit operations: automatic tallies of nonlinear ops (AND, OR) and linear ops (XOR, NOT).

We then apply a generic conversion:
Computing a query

[BLMP] provides software to compute a path using basic bit operations: automatic tallies of nonlinear ops (AND, OR) and linear ops (XOR, NOT).

We then apply a generic conversion:

sequence of basic bit ops with $\leq B$ $\leadsto$ sequence of reversible ops with $\leq 2B$ $\leadsto$ sequence of reversible ops with $\leq 14B$
sequential nonlinear ops Toffoli ops T-gates

Why this generic conversion?
Unknown expense of extra $O(B)$ measurements in context of surface-code error correction

Open question: How much faster than the generic conversion is possible?
Computing a query

[BLMP] provides software to compute a path using basic bit operations: automatic tallies of nonlinear ops (AND, OR) and linear ops (XOR, NOT).

We then apply a generic conversion:

sequence of basic bit ops with $\leq B \sim$ sequence of reversible ops with $\leq 2B \sim$ sequence of reversible ops with $\leq 14B$

nonlinear ops Toffoli ops T-gates

Why this generic conversion?
Unknown expense of extra $O(B)$ measurements in context of surface-code error correction
Computing a query

[BLMP] provides software to compute a path using basic bit operations: automatic tallies of nonlinear ops (AND, OR) and linear ops (XOR, NOT).

We then apply a generic conversion:

sequence of basic bit ops with $\leq B$ \(\leadsto\) sequence of reversible ops with $\leq 2B$ \(\leadsto\) sequence of reversible ops with $\leq 14B$

Why this generic conversion?
Unknown expense of extra $O(B)$ measurements in context of surface-code error correction

Open question:
How much faster than the generic conversion is possible?
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I
(based on asymptotic complexities for Kuperberg’s algorithm).

Here the finite field is $\mathbb{F}_p$ with $p = 4 \cdot \ell_1 \cdots \ell_{74} - 1$, where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.

Note that each $\ell_i$ divides $p + 1$.

For an error rate of $< 2^{-32}$, our best algorithm requires $\approx 765325228976 \approx 0.7 \cdot 2^{40}$ nonlinear bit operations. Previous record was $2^{51}$.

Generic conversion gives $\approx 2^{43.3} T$-gates using $2^{40}$ qubits.

Can do $\approx 2^{45.3} T$-gates using $\approx 2^{20}$ qubits.

Total gates for one query (T+Clifford): $\approx 2^{46.9}$.

Number of queries: see next talk.
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I (based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

$$p = 4 \cdot \ell_1 \cdots \ell_{74} - 1,$$

where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I
(based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

$$p = 4 \cdot \ell_1 \cdots \ell_{74} - 1,$$

where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.
- Note that each $\ell_i$ divides $p + 1$. 
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I (based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

  \[ p = 4 \cdot \ell_1 \cdots \ell_{74} - 1, \]

  where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.

- Note that each $\ell_i$ divides $p + 1$.

- For an error rate of $< 2^{-32}$, our best algorithm requires

  \[ 765325228976 \approx 0.7 \cdot 2^{40} \]

  nonlinear bit operations. Previous record was $2^{51}$. 
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I
(based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

$$p = 4 \cdot \ell_1 \cdots \ell_{74} - 1,$$

where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.

- Note that each $\ell_i$ divides $p + 1$.

- For an error rate of $< 2^{-32}$, our best algorithm requires

$$765325228976 \approx 0.7 \cdot 2^{40}$$

nonlinear bit operations. Previous record was $2^{51}$.

- Generic conversion gives $\approx 2^{43.3}$ T-gates using $2^{40}$ qubits.
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I (based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

\[ p = 4 \cdot \ell_1 \cdots \ell_{74} - 1, \]

where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.

- Note that each $\ell_i$ divides $p + 1$.

- For an error rate of $< 2^{-32}$, our best algorithm requires

\[ 765325228976 \approx 0.7 \cdot 2^{40} \]

nonlinear bit operations. Previous record was $2^{51}$.

- Generic conversion gives $\approx 2^{43.3}$ T-gates using $2^{40}$ qubits.

- Can do $\approx 2^{45.3}$ T-gates using $\approx 2^{20}$ qubits.
Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I
(based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

$$p = 4 \cdot \ell_1 \cdots \ell_{74} - 1,$$

where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.

- Note that each $\ell_i$ divides $p + 1$.

- For an error rate of $< 2^{-32}$, our best algorithm requires

$$765325228976 \approx 0.7 \cdot 2^{40}$$

nonlinear bit operations. Previous record was $2^{51}$.

- Generic conversion gives $\approx 2^{43.3}$ T-gates using $2^{40}$ qubits.

- Can do $\approx 2^{45.3}$ T-gates using $\approx 2^{20}$ qubits.

- Total gates for one query (T+Clifford): $\approx 2^{46.9}$. 

Case study: CSIDH-512

[CLMPR]: proposes CSIDH-512 for NIST level I (based on asymptotic complexities for Kuperberg’s algorithm).

- Here the finite field is $\mathbb{F}_p$ with

$$p = 4 \cdot \ell_1 \cdots \ell_{74} - 1,$$

where $\ell_1, \ldots, \ell_{74}$ are small distinct primes.

- Note that each $\ell_i$ divides $p + 1$.

- For an error rate of $< 2^{-32}$, our best algorithm requires

$$765325228976 \approx 0.7 \cdot 2^{40}$$

nonlinear bit operations. Previous record was $2^{51}$.

- Generic conversion gives $\approx 2^{43.3}$ T-gates using $2^{40}$ qubits.

- Can do $\approx 2^{45.3}$ T-gates using $\approx 2^{20}$ qubits.

- Total gates for one query (T+Clifford): $\approx 2^{46.9}$.

- Number of queries: see next talk.
Oracle errors

BLMP gives oracle costs for error rates $2^{-1}$, $2^{-32}$, and $2^{-256}$. 
Oracle errors

BLMP gives oracle costs for error rates $2^{-1}, 2^{-32},$ and $2^{-256}$.

- Understanding the error tolerance of Kuperberg’s algorithm is essential to obtain accurate concrete numbers.
Oracle errors

BLMP gives oracle costs for error rates $2^{-1}$, $2^{-32}$, and $2^{-256}$.

- Understanding the error tolerance of Kuperberg’s algorithm is essential to obtain accurate concrete numbers.
- Advances in quantum error correction would also massively change the complexity.
Open questions: summary

▶ How do oracle errors interact with Kuperberg’s algorithm?
▶ What kind of overheads come from handling large numbers of qubits?
▶ Is there a quantum algorithm that does better than L(1/2)?
▶ Can we decrease the cost of one query?
Open questions: summary

- How do oracle errors interact with Kuperberg’s algorithm?
- What kind of overheads come from handling large numbers of qubits?
- Is there a quantum algorithm that does better than $L(1/2)$?
- Can we decrease the cost of one query?

Thank you!
References

BLMP  Bernstein, Lange, Martindale, and Panny,
Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies,
Eurocrypt 2019, quantum.isogeny.org.

CLMPR  Castryck, Lange, Martindale, Panny, and Renes,
CSIDH: An efficient post-quantum commutative group action,

Credits to my coauthors Daniel J. Bernstein, Tanja Lange, and
Lorenz Panny for many of the contents of this presentation.