Constructing elliptic curve isogenies in quantum subexponential time

Andrew Childs
IQC, C&O

David Jao
C&O

Vladimir Soukharev
C&O

University of Waterloo
Public-key cryptography in the quantum world

Shor 94: Quantum computers can efficiently
• factor integers
• calculate discrete logarithms (in any group)

This breaks two common public-key cryptosystems:
• RSA
• elliptic curve cryptography
Public-key cryptography in the quantum world

Shor 94: Quantum computers can efficiently
• factor integers
• calculate discrete logarithms (in any group)

This breaks two common public-key cryptosystems:
• RSA
• elliptic curve cryptography

How do quantum computers affect the security of PKC in general?

Practical question: we’d like to be able to send confidential information even after quantum computers are built

Theoretical question: crypto is a good setting for exploring the potential strengths/limitations of quantum computers
Isogeny-based elliptic curve cryptography

Not all elliptic curve cryptography is known to be quantumly broken!

Couveignes 97, Rostovstev-Stolbunov 06, Stolbunov 10: Public-key cryptosystems based on the assumption that it is hard to construct an isogeny between given elliptic curves over $\mathbb{F}_q$

Best known classical algorithm: $O(q^{1/4})$ [Galbraith, Hess, Smart 02]
Isogeny-based elliptic curve cryptography

Not all elliptic curve cryptography is known to be quantumly broken!

Couveignes 97, Rostovstev-Stolbunov 06, Stolbunov 10: Public-key cryptosystems based on the assumption that it is hard to construct an isogeny between given elliptic curves over $\mathbb{F}_q$

Best known classical algorithm: $O(q^{1/4})$ [Galbraith, Hess, Smart 02]

Main result of this talk:
Quantum algorithm that constructs an isogeny in time $L_q\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ (assuming GRH), where

$$L_q(\alpha, c) := \exp\left[(c + o(1))(\ln q)^\alpha (\ln \ln q)^{1-\alpha}\right]$$
Elliptic curves

Let $\mathbb{F}$ be a field of characteristic different from 2 or 3

An elliptic curve $E$ is the set of points in $\mathbb{P}\mathbb{F}^2$ satisfying an equation of the form $y^2 = x^3 + ax + b$

Example ($\mathbb{F} = \mathbb{R}$):

$$y^2 = x^3 - x + 1$$
Elliptic curve group

Geometric definition of a binary operation on points of $E$:

This defines an abelian group with additive identity $\infty$
Elliptic curve group

Geometric definition of a binary operation on points of $E$:

![Diagram of an elliptic curve with points P, Q, and R showing the geometric definition of addition.]

Algebraic definition:

For $x_P \neq x_Q$,

$$\lambda := \frac{y_Q - y_P}{x_Q - x_P}$$

$$x_{P+Q} = \lambda^2 - x_P - x_Q$$

$$y_{P+Q} = \lambda(x_P - x_{P+Q}) - y_P$$

(similar expressions for other cases)

This defines an abelian group with additive identity $\infty$
Elliptic curves over finite fields

Cryptographic applications use a finite field $\mathbb{F}_q$

Example: $y^2 = x^3 + 2x + 2$

$\mathbb{F} = \mathbb{R}$

$\mathbb{F} = \mathbb{F}_{109}$
Elliptic curve isogenies

Let $E_0, E_1$ be elliptic curves

An isogeny $\phi : E_0 \to E_1$ is a rational map

$$\phi(x, y) = \left( \frac{f_x(x, y)}{g_x(x, y)}, \frac{f_y(x, y)}{g_y(x, y)} \right)$$

($f_x, f_y, g_x, g_y$ are polynomials) that is also a group homomorphism:

$$\phi((x, y) + (x', y')) = \phi(x, y) + \phi(x', y')$$
Elliptic curve isogenies

Let $E_0, E_1$ be elliptic curves

An isogeny $\phi : E_0 \to E_1$ is a rational map

$$\phi(x, y) = \left( \frac{f_x(x, y)}{g_x(x, y)}, \frac{f_y(x, y)}{g_y(x, y)} \right)$$

($f_x, f_y, g_x, g_y$ are polynomials) that is also a group homomorphism:

$$\phi((x, y) + (x', y')) = \phi(x, y) + \phi(x', y')$$

Example ($\mathbb{F} = \mathbb{F}_{109}$):

$$E_0 : y^2 = x^3 + 2x + 2 \quad \overset{\phi}{\longrightarrow} \quad E_1 : y^2 = x^3 + 34x + 45$$

$$\phi(x, y) = \left( \frac{x^3 + 20x^2 + 50x + 6}{x^2 + 20x + 100}, \frac{x^3 + 30x^2 + 23x + 52)y}{x^3 + 30x^2 + 82x + 19} \right)$$
Deciding isogeny

Theorem [Tate 66]: Two elliptic curves over a finite field are isogenous if and only if they have the same number of points.

There is a polynomial-time classical algorithm that counts the points on an elliptic curve [Schoof 85].

Thus a classical computer can decide isogeny in polynomial time.
The endomorphism ring

The set of isogenies from $E$ to itself (over $\overline{F}$) is denoted $\text{End}(E)$.
The endomorphism ring

The set of isogenies from $E$ to itself (over $\overline{F}$) is denoted $\text{End}(E)$.

We assume $E$ is ordinary (i.e., not supersingular), which is the typical case; then $\text{End}(E) \cong \mathcal{O}_\Delta = \mathbb{Z}[\frac{\Delta + \sqrt{\Delta}}{2}]$ is an imaginary quadratic order of discriminant $\Delta < 0$. 

The endomorphism ring

The set of isogenies from $E$ to itself (over $\overline{F}$) is denoted $\text{End}(E)$.

We assume $E$ is ordinary (i.e., not supersingular), which is the typical case; then $\text{End}(E) \cong \mathcal{O}_\Delta = \mathbb{Z}[\frac{\Delta + \sqrt{\Delta}}{2}]$ is an imaginary quadratic order of discriminant $\Delta < 0$.

If $\text{End}(E_0) = \text{End}(E_1)$ then we say $E_0$ and $E_1$ are endomorphic.
The endomorphism ring

The set of isogenies from $E$ to itself (over $\overline{F}$) is denoted $\text{End}(E)$

We assume $E$ is ordinary (i.e., not supersingular), which is the typical case; then $\text{End}(E) \cong \mathcal{O}_\Delta = \mathbb{Z}[\frac{\Delta + \sqrt{\Delta}}{2}]$ is an imaginary quadratic order of discriminant $\Delta < 0$

If $\text{End}(E_0) = \text{End}(E_1)$ then we say $E_0$ and $E_1$ are endomorphic

Let $\text{Ell}_{q,n}(\mathcal{O}_\Delta)$ denote the set of elliptic curves over $\mathbb{F}_q$ with $n$ points and endomorphism ring $\mathcal{O}_\Delta$ (up to isomorphism of curves)
Representing isogenies

The degree of an isogeny can be exponential (in $\log q$)

**Example:** The multiplication by $m$ map,

$$(x, y) \mapsto (x, y) + \cdots + (x, y)$$

is an isogeny of degree $m^2$

Thus we cannot even write down the rational map explicitly in polynomial time
Representing isogenies

The degree of an isogeny can be exponential (in $\log q$)

**Example:** The multiplication by $m$ map,

$$(x, y) \mapsto (x, y) + \cdots + (x, y)$$

is an isogeny of degree $m^2$

Thus we cannot even write down the rational map explicitly in polynomial time

**Fact:** Isogenies between endomorphic elliptic curves can be represented by elements of a finite abelian group, the *ideal class group* of the endomorphism ring, denoted $\text{Cl}(\mathcal{O}_\Delta)$
A group action

Thus we can view isogenies in terms of a group action

\[ *[b] \in \text{Cl}(\mathcal{O}_\Delta) \times \text{Ell}_{q,n}(\mathcal{O}_\Delta) \to \text{Ell}_{q,n}(\mathcal{O}_\Delta) \]

\[ [b] \ast E = E_b \]

where \( E_b \) is the elliptic curve reached from \( E \) by an isogeny corresponding to the ideal class \([b]\).
A group action

Thus we can view isogenies in terms of a group action

\[ * : \text{Cl}(\mathcal{O}_\Delta) \times \text{Ell}_{q,n}(\mathcal{O}_\Delta) \to \text{Ell}_{q,n}(\mathcal{O}_\Delta) \]

\[ [b] * E = E_b \]

where \( E_b \) is the elliptic curve reached from \( E \) by an isogeny corresponding to the ideal class \([b]\)

This action is regular [Waterhouse 69]:
for any \( E_0, E_1 \) there is a unique \([b]\) such that \([b] * E_0 = E_1\)
The abelian hidden shift problem

Let $A$ be a known finite abelian group

Let $f_0 : A \rightarrow R$ be an injective function (for some finite set $R$)

Let $f_1 : A \rightarrow R$ be defined by $f_1(x) = f_0(xs)$ for some unknown $s \in A$

Problem: find $s$
The abelian hidden shift problem

Let $A$ be a known finite abelian group

Let $f_0 : A \rightarrow R$ be an injective function (for some finite set $R$)

Let $f_1 : A \rightarrow R$ be defined by $f_1(x) = f_0(xs)$ for some unknown $s \in A$

Problem: find $s$

For $A$ cyclic, this is equivalent to the dihedral hidden subgroup problem

More generally, this is equivalent to the HSP in the generalized dihedral group $A \times \mathbb{Z}_2$
Isogeny construction as a hidden shift problem

Define $f_0, f_1 : \text{Cl}(\mathcal{O}_\Delta) \to \text{Ell}_{q,n}(\mathcal{O}_\Delta)$ by

$$f_0([b]) = [b] \ast E_0$$
$$f_1([b]) = [b] \ast E_1$$

$E_0, E_1$ are isogenous, so there is some $[s]$ such that $[s] \ast E_0 = E_1$
Isogeny construction as a hidden shift problem

Define $f_0, f_1 : \text{Cl}(\mathcal{O}_\Delta) \to \text{Ell}_{q,n}(\mathcal{O}_\Delta)$ by

$$f_0([b]) = [b] \ast E_0$$
$$f_1([b]) = [b] \ast E_1$$

$E_0, E_1$ are isogenous, so there is some $[s]$ such that $[s] \ast E_0 = E_1$

Therefore this is an instance of the hidden shift problem in $\text{Cl}(\mathcal{O}_\Delta)$ with hidden shift $[s]$:

- Since $\ast$ is regular, $f_0$ is injective
- Since $\ast$ is a group action, $f_1([b]) = f_0([b][s])$
Kuperberg’s algorithm

Theorem [Kuperberg 03]: There is a quantum algorithm that solves the abelian hidden shift problem in a group of order $N$ with running time $\exp[O(\sqrt{\ln N})] = L_N(\frac{1}{2}, 0)$. 
Kuperberg’s algorithm

Theorem [Kuperberg 03]: There is a quantum algorithm that solves the abelian hidden shift problem in a group of order $N$ with running time $\exp[O(\sqrt{\ln N})] = L_N(\frac{1}{2}, 0)$.

Thus there is a quantum algorithm to construct an isogeny with running time

$$L_N(\frac{1}{2}, 0) \times c(N)$$

where $c(N)$ is the cost of evaluating the action
Kuperberg’s algorithm

Theorem [Kuperberg 03]: There is a quantum algorithm that solves the abelian hidden shift problem in a group of order $N$ with running time $\exp[O(\sqrt{\ln N})] = L_N(\frac{1}{2}, 0)$.

Thus there is a quantum algorithm to construct an isogeny with running time

$$L_N(\frac{1}{2}, 0) \times c(N)$$

where $c(N)$ is the cost of evaluating the action

But previously it was not known how to compute the action in subexponential time.
Computing the action

**Problem:** Given $E, \Delta, b \in \mathcal{O}_\Delta$, compute $[b] \ast E$
Computing the action

Problem: Given $E, \Delta, b \in \mathcal{O}_\Delta$, compute $[b] \ast E$

Direct computation (using modular polynomials) takes time $O(\ell^3)$ for an ideal of norm $\ell$
Computing the action

Problem: Given $E$, $\Delta$, $b \in \mathcal{O}_\Delta$, compute $[b] \ast E$

Direct computation (using modular polynomials) takes time $O(\ell^3)$ for an ideal of norm $\ell$

Instead we use an indirect approach:

- Choose a factor base of small prime ideals $p_1, \ldots, p_f$
- Find a factorization $[b] = [p_1^{e_1} \cdots p_f^{e_f}]$ where $e_1, \ldots, e_f$ are small
- Compute $[b] \ast E$ one small prime at a time
Computing the action

Problem: Given $E, \Delta, b \in \mathcal{O}_\Delta$, compute $[b] * E$

Direct computation (using modular polynomials) takes time $O(\ell^3)$ for an ideal of norm $\ell$

Instead we use an indirect approach:

• Choose a factor base of small prime ideals $p_1, \ldots, p_f$
• Find a factorization $[b] = [p_1^{e_1} \cdots p_f^{e_f}]$ where $e_1, \ldots, e_f$ are small
• Compute $[b] * E$ one small prime at a time

By optimizing the size of the factor base, this approach can be made to work in time $L\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ (assuming GRH)
Computing the action

Problem: Given $E, \Delta, b \in \mathcal{O}_\Delta$, compute $[b] \ast E$

Direct computation (using modular polynomials) takes time $O(\ell^3)$ for an ideal of norm $\ell$

Instead we use an indirect approach:

- Choose a factor base of small prime ideals $p_1, \ldots, p_f$
- Find a factorization $[b] = [p_1^{e_1} \cdots p_f^{e_f}]$ where $e_1, \ldots, e_f$ are small
- Compute $[b] \ast E$ one small prime at a time

By optimizing the size of the factor base, this approach can be made to work in time $L\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ (assuming GRH)

Note: This assumes only GRH (previous related algorithms required stronger heuristic assumptions)
Polynomial space

Kuperberg’s algorithm uses space $\exp[\Theta(\sqrt{\ln N})]$.

Regev 04 presented a modified algorithm using only polynomial space for the case $A = \mathbb{Z}_{2^n}$, with running time

$$\exp[O(\sqrt{n \ln n})] = L_{2^n}(\frac{1}{2}, O(1))$$

Combining Regev’s ideas with techniques used by Kuperberg for the case of a general abelian group (of order $N$), and performing a careful analysis, we find an algorithm with running time $L_N(\frac{1}{2}, \sqrt{2})$.

Thus there is a quantum algorithm to construct elliptic curve isogenies using only polynomial space in time $L_q(\frac{1}{2}, \frac{\sqrt{3}}{2} + \sqrt{2})$. 
Conclusions

Given two isogenous, endomorphic, ordinary elliptic curves over $\mathbb{F}_q$, there is a quantum algorithm that constructs an isogeny between them in time $L_q\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ (or in time $L_q\left(\frac{1}{2}, \frac{\sqrt{3}}{2} + \sqrt{2}\right)$) using $\text{poly}(\log q)$ space.
Conclusions

Given two isogenous, endomorphic, ordinary elliptic curves over $\mathbb{F}_q$, there is a quantum algorithm that constructs an isogeny between them in time $L_q(\frac{1}{2}, \frac{\sqrt{3}}{2})$ (or in time $L_q(\frac{1}{2}, \frac{\sqrt{3}}{2} + \sqrt{2})$ using $\text{poly}(\log q)$ space)

Consequences:

• Isogeny-based cryptography may be less secure than more mainstream cryptosystems (e.g., lattices)
Conclusions

Given two isogenous, endomorphic, ordinary elliptic curves over $\mathbb{F}_q$, there is a quantum algorithm that constructs an isogeny between them in time $L_q\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ (or in time $L_q\left(\frac{1}{2}, \frac{\sqrt{3}}{2} + \sqrt{2}\right)$ using $\text{poly(}\log q\text{)}$ space)

Consequences:

- Isogeny-based cryptography may be less secure than more mainstream cryptosystems (e.g., lattices)
- Computing properties of algebraic curves may be a fruitful direction for new quantum algorithms
  - Can we break isogeny-based cryptography in polynomial time?
  - Computing properties of a single curve (e.g., endomorphism ring)
  - Generalizations: non-endomorphic curves, supersingular curves