Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies

Daniel J. Bernstein, Tanja Lange,
Chloe Martindale, Lorenz Panny

https://quantum.isogeny.org

Non-interactive key exchange

Alice: secret a, public aG. Bob: secret b, public bG.
Shared secret $a(bG) = (ab)G = (ba)G = b(aG)$.

Non-interactive key exchange

Alice: secret a, public aG. Bob: secret b, public bG.
Shared secret $a(bG) = (ab)G = (ba)G = b(aG)$.

DH: 1976 Diffie–Hellman.
ECDH: 1985 Miller, 1987 Koblitz.
Cost poly(λ) for pre-quantum security level 2^λ
(*assuming* that the best attacks known are optimal).

Non-interactive key exchange

Alice: secret a, public aG. Bob: secret b, public bG.
Shared secret $a(bG) = (ab)G = (ba)G = b(aG)$.

DH: 1976 Diffie–Hellman.
ECDH: 1985 Miller, 1987 Koblitz.
Cost poly(λ) for pre-quantum security level 2^λ
(*assuming* that the best attacks known are optimal).
Fast addition of public keys $\rightarrow$ post-quantum break.

Non-interactive key exchange

Alice: secret a, public aG. Bob: secret b, public bG.
Shared secret $a(bG) = (ab)G = (ba)G = b(aG)$.

DH: 1976 Diffie–Hellman.
ECDH: 1985 Miller, 1987 Koblitz.
Cost poly(λ) for pre-quantum security level 2^λ
(*assuming* that the best attacks known are optimal).
Fast addition of public keys $\rightarrow$ post-quantum break.

CRS: 2006 Rostovtsev–Stolbunov, 2006 Couveignes.
Slow. Not obviously not post-quantum.


[ˈsiːˌsaɪd]

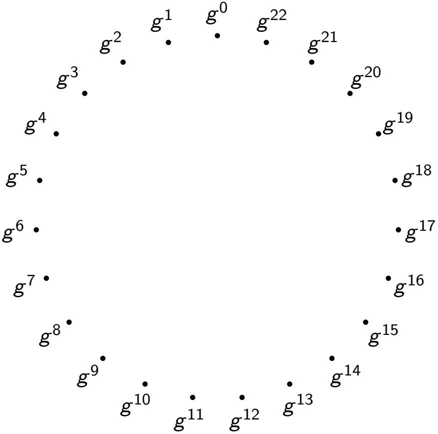
CSIDH: An Efficient Post-Quantum Commutative Group Action

Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, Joost Renes 2018

- Closest thing we have in PQC to normal DH key exchange: Keys can be reused, keys can be blinded; no difference between initiator & responder.
- Public keys are represented by some $A \in \mathbf{F}_p$; p fixed prime.
- Alice computes and distributes her public key A.
  Bob computes and distributes his public key B.
- Alice and Bob do computations on each other's public keys to obtain shared secret.
- Fancy math: computations start on some elliptic curve
  $$E_A : y^2 = x^3 + Ax^2 + x,$$
  use *isogenies* to move to a different curve.
- Computations need arithmetic (add, mult, div) modulo p and elliptic-curve computations.

Square-and-multiply

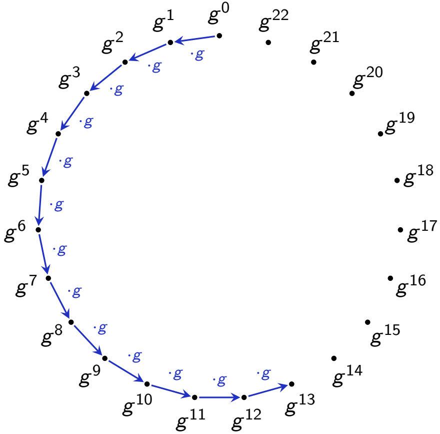
Reminder: DH in group with $\#G = 23$. Alice computes g^{13}.



Pretty pictures by Chloe Martindale and Lorenz Panny.

Square-and-multiply

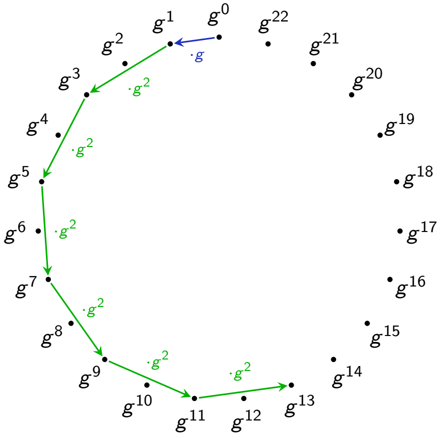
Reminder: DH in group with $\#G = 23$. Alice computes g^{13}.



Pretty pictures by Chloe Martindale and Lorenz Panny.

Square-and-multiply

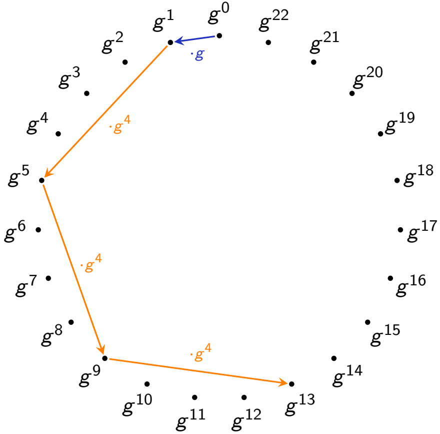
Reminder: DH in group with $\#G = 23$. Alice computes g^{13}.



Pretty pictures by Chloe Martindale and Lorenz Panny.

Square-and-multiply

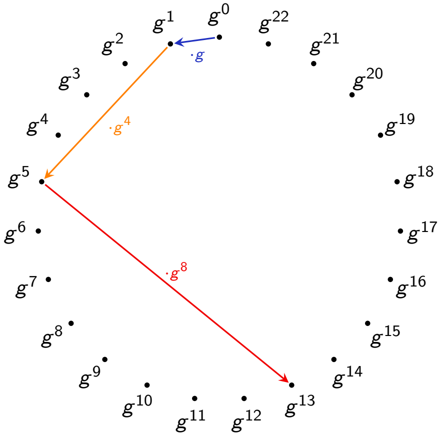
Reminder: DH in group with $\#G = 23$. Alice computes g^{13}.



Pretty pictures by Chloe Martindale and Lorenz Panny.

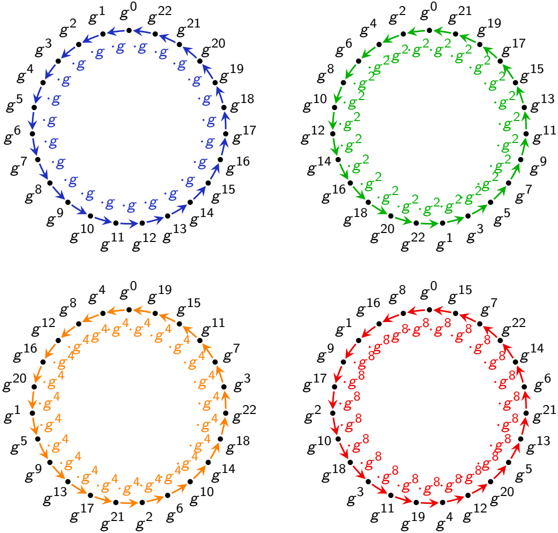
Square-and-multiply

Reminder: DH in group with $\#G = 23$. Alice computes g^{13}.



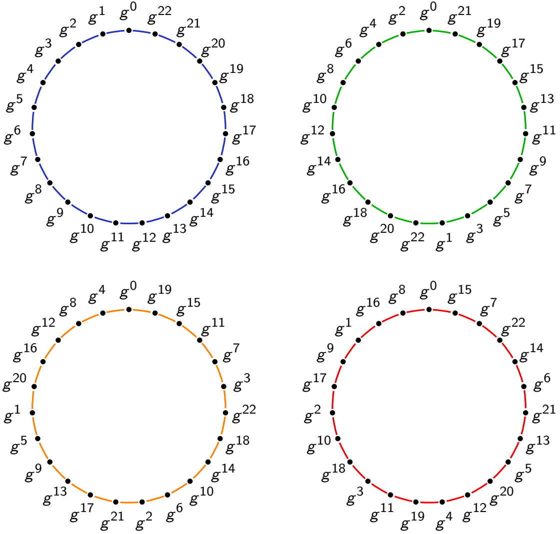
Pretty pictures by Chloe Martindale and Lorenz Panny.

Square-and-multiply



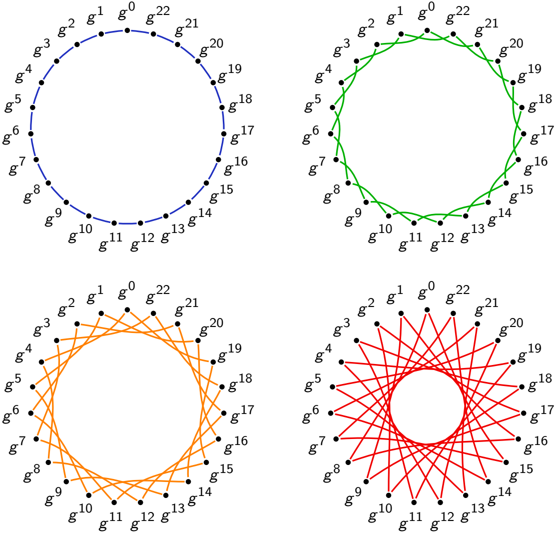
Pretty pictures by Chloe Martindale and Lorenz Panny.

Square-and-multiply



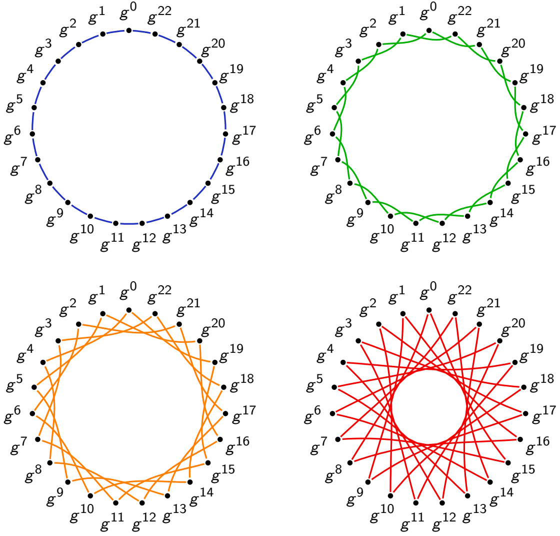
Pretty pictures by Chloe Martindale and Lorenz Panny.

Square-and-multiply



Pretty pictures by Chloe Martindale and Lorenz Panny.

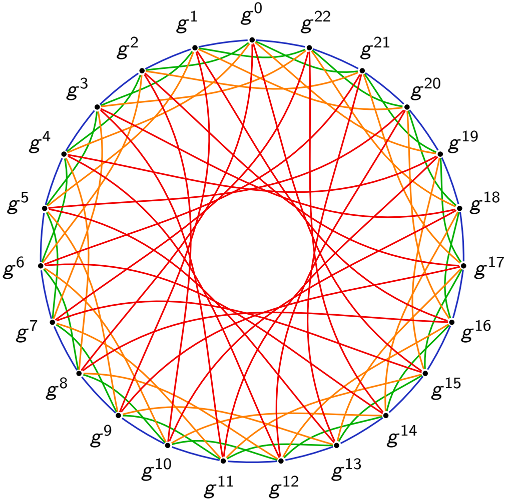
Square-and-multiply



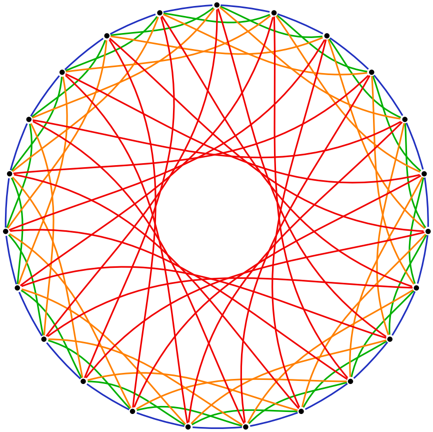
Cycles are *compatible*: [right, then left] = [left, then right], etc.

Pretty pictures by Chloe Martindale and Lorenz Panny.

Union of cycles: rapid mixing



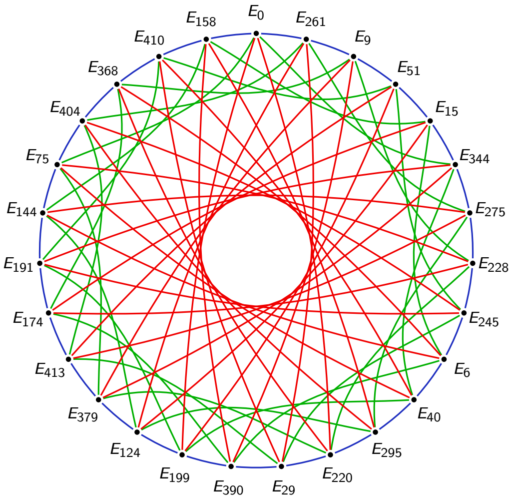
Union of cycles: rapid mixing



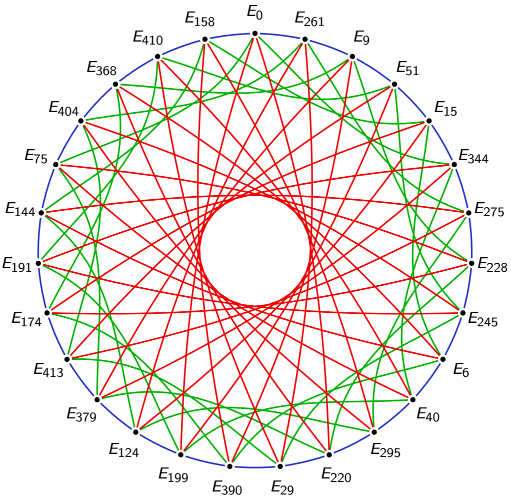
CSIDH: Nodes are now *elliptic curves* and edges are *isogenies*.

Pretty pictures by Chloe Martindale and Lorenz Panny.

Graphs of elliptic curves

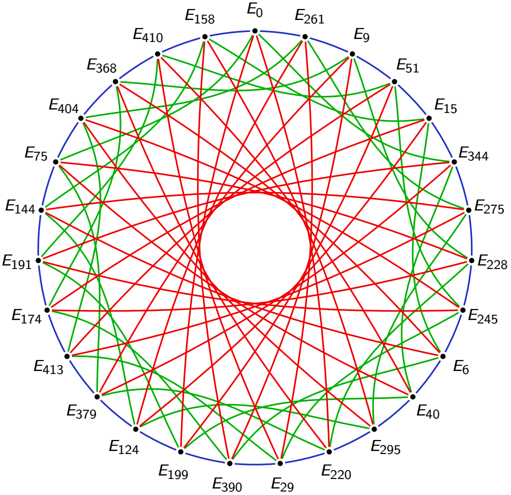


Graphs of elliptic curves



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

Graphs of elliptic curves



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

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

Pretty pictures by Chloe Martindale and Lorenz Panny.

Non-interactive key exchange

Alice: secret a, public aG. Bob: secret b, public bG.
Shared secret $a(bG) = (ab)G = (ba)G = b(aG)$.

DH: 1976 Diffie–Hellman.
ECDH: 1985 Miller, 1987 Koblitz.
Cost poly(λ) for pre-quantum security level 2^λ
(*assuming* that the best attacks known are optimal).
Fast addition of public keys $\to$ post-quantum break.

CRS: 2006 Rostovtsev–Stolbunov, 2006 Couveignes.
CSIDH: 2018 Castryck-Lange-Martindale-Panny-Renes.
Cost poly(λ) for pre-quantum security level 2^λ.
Cost poly(λ) for post-quantum security level 2^λ.

Encryption systems with small public keys

Key bits where all known attacks take 2^λ operations
(naive serial attack metric, ignoring memory cost):

	pre-quantum	post-quantum
SIDH, SIKE	$(24 + o(1))\lambda$	$(36 + o(1))\lambda$
compressed	$(14 + o(1))\lambda$	$(21 + o(1))\lambda$
CRS, CSIDH	$(4 + o(1))\lambda$	superlinear
ECDH	$(2 + o(1))\lambda$	exponential

Hard problem in CSIDH:
Given curves E_0 and $E = \varphi(E_0)$ find isogeny φ.
Also: φ needs to be quickly computable, $\varphi = [P_1]^{a_1} \cdots [P_d]^{a_d}$.

Encryption systems with small public keys

Key bits where all known attacks take 2^λ operations (naive serial attack metric, ignoring memory cost):

	pre-quantum	post-quantum
SIDH, SIKE	$(24 + o(1))\lambda$	$(36 + o(1))\lambda$
compressed	$(14 + o(1))\lambda$	$(21 + o(1))\lambda$
CRS, CSIDH	$(4 + o(1))\lambda$	superlinear
ECDH	$(2 + o(1))\lambda$	exponential

Hard problem in CSIDH:
Given curves E_0 and $E = \varphi(E_0)$ find isogeny φ.
Also: φ needs to be quickly computable, $\varphi = [P_1]^{a_1} \cdots [P_d]^{a_d}$.

Subexp 2010 Childs–Jao–Soukharev attack (on CRS):
This problem can be seen as a hidden-shift problem.
2003 Kuperberg or 2004 Regev or 2011 Kuperberg solves this in subexponentially many queries.

Attack works for any commutative group action, thus also CSIDH.

Major questions

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

Major questions

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

Subexp attack: many quantum CSIDH queries.

- How many queries do these attacks perform?
  2011 Kuperberg supersedes previous papers.

Major questions

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

Subexp attack: many quantum CSIDH queries.

- How many queries do these attacks perform?
  2011 Kuperberg supersedes previous papers.
- How is attack affected by occasional errors
  and non-uniform distributions over the group?

Major questions

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

Subexp attack: many quantum CSIDH queries.

- How many queries do these attacks perform?
  2011 Kuperberg supersedes previous papers.
- How is attack affected by occasional errors
  and non-uniform distributions over the group?
- How expensive is each CSIDH query?
  See our paper—full 56-page version online, with detailed analysis and many optimizations.

Major questions

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

Subexp attack: many quantum CSIDH queries.

- How many queries do these attacks perform?
  2011 Kuperberg supersedes previous papers.
- How is attack affected by occasional errors
  and non-uniform distributions over the group?
- How expensive is each CSIDH query?
  See our paper—full 56-page version online, with detailed analysis and many optimizations.
- What about memory, using parallel AT metric?

Verifying quantum costs on your laptop

We provide software to compute CSIDH group action using bit operations.
Automatic tallies of nonlinear ops (AND, OR), linear ops (XOR, NOT).

Verifying quantum costs on your laptop

We provide software to compute CSIDH group action using bit operations.
Automatic tallies of nonlinear ops (AND, OR), linear ops (XOR, NOT).

Generic conversions:
sequence of bit ops with $\leq B$ nonlinear ops
$\Rightarrow$ sequence of *reversible* ops with $\leq 2B$ Toffoli ops

Verifying quantum costs on your laptop

We provide software to compute CSIDH group action using bit operations.
Automatic tallies of nonlinear ops (AND, OR), linear ops (XOR, NOT).

Generic conversions:
sequence of bit ops with $\leq B$ nonlinear ops
$\Rightarrow$ sequence of *reversible* ops with $\leq 2B$ Toffoli ops
$\Rightarrow$ sequence of *quantum* gates with $\leq 14B$ T-gates.

Verifying quantum costs on your laptop

We provide software to compute CSIDH group action using bit operations.
Automatic tallies of nonlinear ops (AND, OR), linear ops (XOR, NOT).

Generic conversions:
sequence of bit ops with $\leq B$ nonlinear ops
$\Rightarrow$ sequence of *reversible* ops with $\leq 2B$ Toffoli ops
$\Rightarrow$ sequence of *quantum* gates with $\leq 14B$ T-gates.

Building confidence in correctness of output:
1. Compare output to Sage script for CSIDH.
2. Generating-function analysis of *exact* error rates.

Compare to experiments with noticeable error rates.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5, \ldots, 5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

$\approx 2^{51}$ by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5,\ldots,5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

$\approx 2^{51}$ by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.
$1118827416420 \approx 2^{40}$ by our Algorithm 7.1.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5, \ldots, 5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

	$\approx 2^{51}$	by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.
1118827416420	$\approx 2^{40}$	by our Algorithm 7.1.
765325228976	$\approx 0.7 \cdot 2^{40}$	by our Algorithm 8.1.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5, \ldots, 5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

$$\approx 2^{51} \quad \text{by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.}$$
$$1118827416420 \approx 2^{40} \quad \text{by our Algorithm 7.1.}$$
$$765325228976 \approx 0.7 \cdot 2^{40} \quad \text{by our Algorithm 8.1.}$$

$\Rightarrow \approx 2^{43.3}$ T-gates using $\approx 2^{40}$ qubits.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5, \ldots, 5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

$$\begin{array}{rll} & \approx 2^{51} & \text{by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.} \\ 1118827416420 & \approx 2^{40} & \text{by our Algorithm 7.1.} \\ 765325228976 & \approx 0.7 \cdot 2^{40} & \text{by our Algorithm 8.1.} \end{array}$$

$\Rightarrow \approx 2^{43.3}$ T-gates using $\approx 2^{40}$ qubits.
Can do $\approx 2^{45.3}$ T-gates using $\approx 2^{20}$ qubits.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5,\ldots,5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

	$\approx 2^{51}$	by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.
1118827416420	$\approx 2^{40}$	by our Algorithm 7.1.
765325228976	$\approx 0.7 \cdot 2^{40}$	by our Algorithm 8.1.

$\Rightarrow \approx 2^{43.3}$ T-gates using $\approx 2^{40}$ qubits.
Can do $\approx 2^{45.3}$ T-gates using $\approx 2^{20}$ qubits.
Total gates (T+Clifford): $\approx 2^{46.9}$.

Case study: one CSIDH-512 query

Consider query with exponents uniform over $\{-5,\ldots,5\}^{74}$ for the same 74 isogenies as in the constructive use.
For error rate of $<2^{-32}$ (maybe ok) this requires nonlinear bit ops:

	$\approx 2^{51}$	by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.
1118827416420	$\approx 2^{40}$	by our Algorithm 7.1.
765325228976	$\approx 0.7 \cdot 2^{40}$	by our Algorithm 8.1.

$\Rightarrow \approx 2^{43.3}$ T-gates using $\approx 2^{40}$ qubits.
Can do $\approx 2^{45.3}$ T-gates using $\approx 2^{20}$ qubits.
Total gates (T+Clifford): $\approx 2^{46.9}$.

Variations in 512, $\{-5,\ldots,5\}$, 2^{-32}: see paper.

Case study: full CSIDH-512 attack

CSIDH-512 user has inputs $[P_1]^{a_1} \cdots [P_d]^{a_d}$ with $(a_1, \ldots, a_d) \in \{-5, \ldots, 5\}^{74}$ but Kuperberg assumes $[P_1]^a$ with uniform $a \in \mathbf{Z}/N$.

- Approach 1: Compute lattice $L = \mathrm{Ker}(a_1, \ldots, a_d \mapsto [P_1]^{a_1} \cdots [P_d]^{a_d})$.
  Given $a \in \mathbf{Z}^d$, find close $v \in L$:
  distance $\exp((\log N)^{1/2+o(1)})$ using time $\exp((\log N)^{1/2+o(1)})$.
- Approach 2: Increase d up to $\exp((\log N)^{1/2+o(1)})$.
  Search randomly for small relations.
  Time $\exp((\log N)^{1/2+o(1)})$ to compute group action.
- Approach 3 (ours): Uniform $(a_1, \ldots, a_d)$ in $\{-c, \ldots, c\}^d$.
  Choose c somewhat larger than users do.
  Not much slowdown in action.
  Surely $g = [P_1]^{a_1} \cdots [P_d]^{a_d}$ is nearly uniformly distributed.
  Need more analysis of impact of these redundant representations upon Kuperberg's algorithm.