



Discrete log and factoring are cheaper than you think

Greg Meyer

a.k.a. Greg Kahanamoku-Meyer

October 22, 2026

How I felt in 2025

How I felt in 2025

Oh my god, logical circuits for breaking cryptography are
way cheaper than we thought.

How I felt in 2025

Oh my god, logical circuits for breaking cryptography are
way cheaper than we thought.

Why isn't everyone freaking out?

What happened in 2026

**Google Accelerates Q-Day
Migration Timeline to 2029**

What happened in 2026

Google Accelerates O-Day

Mi

Cloudflare targets 2029 for full post-quantum security

2026-04-07

What happened in 2026

Google Accelerates Q-Day

Cloudflare targets 2029 for full

Microsoft Accelerates Post-Quantum

2026+

Cryptography Shift to 2029

What happened in 2026

Google Accelerates Q-Day

Cloudflare targets 2029 for full

Microsoft Accelerates Post-Quantum

2026-1

pc

2026-1

Crypto

White House
drastically shortens
deadline for dropping
quantum-vulnerable
crypto

What happened in 2026

Google Accelerates Q-Day

Cloudflare targets 2029 for full

Microsoft Accelerates Post-Quantum

Mi

pc

2026-1

Crypto

White House
drastically shortens
deadline for dropping
quantum-vulnerable
crypto

People have started paying attention!

How I feel in 2026

How I feel in 2026

Yay, people are paying attention!

How I feel in 2026

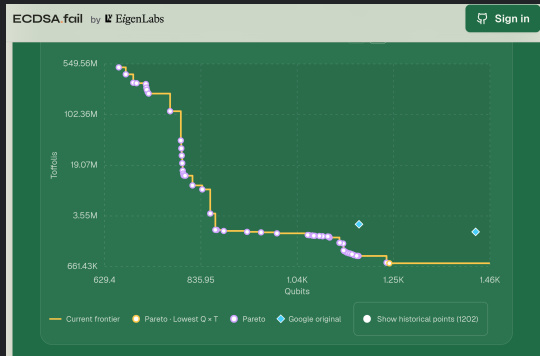
Yay, people are paying attention!

Are we at risk of being *further* surprised?

Circuit improvements via LLMs

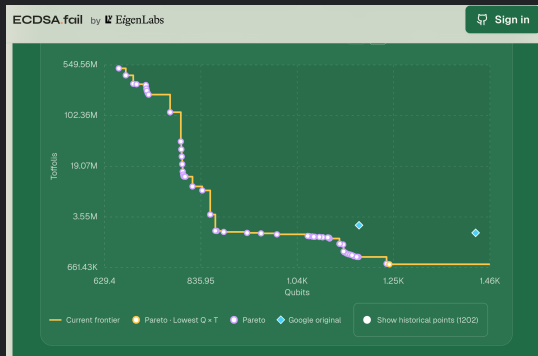
Circuit improvements via LLMs

- LLM-based circuit discovery has some problems, but seems to be **genuinely improving circuits**



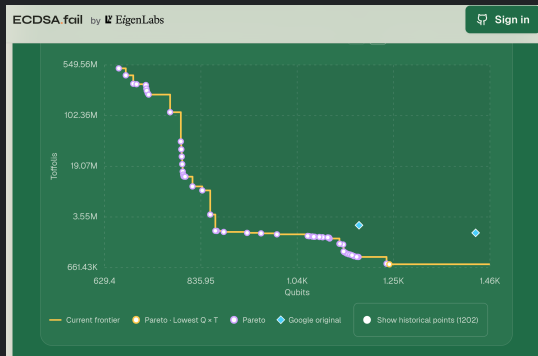
Circuit improvements via LLMs

- LLM-based circuit discovery has some problems, but seems to be **genuinely improving circuits**
- Can create extremely well-optimized circuits for a **given metric**



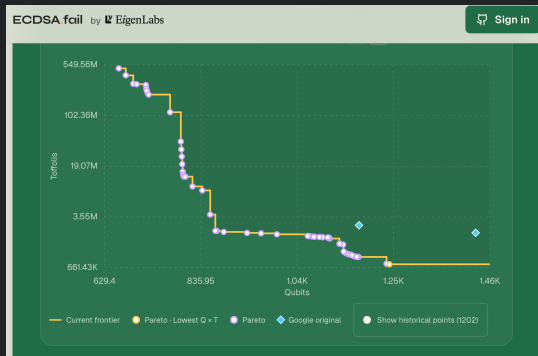
Circuit improvements via LLMs

- LLM-based circuit discovery has some problems, but seems to be **genuinely improving circuits**
- Can create extremely well-optimized circuits for a **given metric**



Circuit improvements via LLMs

- LLM-based circuit discovery has some problems, but seems to be **genuinely improving circuits**
- Can create extremely well-optimized circuits for a **given metric**



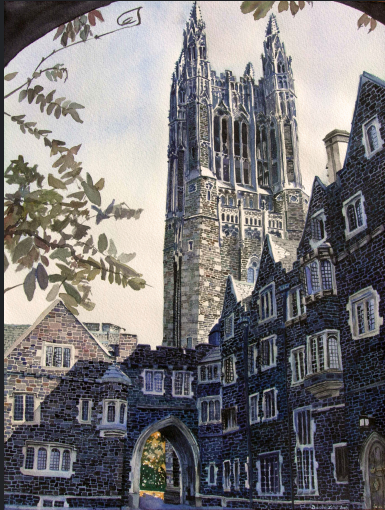
This talk: Given this, what is important for *us* to not miss?

Overview



Are we focusing on the right problems?

Overview



Are we focusing on the right problems?

Co-design of algorithms and architectures: heuristics are changing

Overview



Are we focusing on the right problems?

Co-design of algorithms and architectures: heuristics are changing

A case study: new factoring circuits

Overview



Are we focusing on the right problems?

Co-design of algorithms and architectures: heuristics are changing

A case study: new factoring circuits

Comparing factoring and discrete log

Observation: factoring is often used as a stand-in for “breaking cryptography”

Comparing factoring and discrete log

Observation: factoring is often used as a stand-in for “breaking cryptography”

Is the focus on factoring **appropriate?**

Comparison: use in modern classical cryptography

Example: digital signature schemes

Data: Cloudflare blog (blog.cloudflare.com/ml-dsa-will-have-to-do/)

Scheme	Ed25519	RSA-2048
Vuln. problem	EC discrete log	factoring
Public key size (bytes)	32	272
Signature size (bytes)	64	256
Time to sign (arb. units)	0.15	80
Time to verify (arb. units)	1.3	0.4

Comparison: use in modern classical cryptography

Example: digital signature schemes

Data: Cloudflare blog (blog.cloudflare.com/ml-dsa-will-have-to-do/)

Scheme	Ed25519	RSA-2048
Vuln. problem	EC discrete log	factoring
Public key size (bytes)	32	272
Signature size (bytes)	64	256
Time to sign (arb. units)	0.15	80
Time to verify (arb. units)	1.3	0.4

Schemes relying on the hardness of **elliptic curve discrete log** are used more widely on the internet.

Comparison: quantum cost to break cryptographic problem sizes

Note: these are **logical** costs, in the abstract circuit model.

Comparison: quantum cost to break cryptographic problem sizes

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

Comparison: quantum cost to break cryptographic problem sizes

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Comparison: quantum cost to break cryptographic problem sizes

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Elliptic curve discrete log has cheaper quantum circuits (as of Oct. 2026).

Comparing factoring and discrete log

Most attention given	factoring
Most widely used	discrete log
Cheapest quantum cost	discrete log

Comparing factoring and discrete log

Most attention given	factoring
Most widely used	discrete log
Cheapest quantum cost	discrete log

We're giving attention to the problem that is
less widely used and **more expensive**.

Comparing factoring and discrete log

Most attention given	factoring
Most widely used	discrete log
Cheapest quantum cost	discrete log

We're giving attention to the problem that is
less widely used and **more expensive**.

... but there's even another reason to focus on discrete log...

Let's see who you really are...



Let's see who you really are...



Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of
positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of
positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

★ Knowing $p + q$ is sufficient to factor N .

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

☀️ Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

🌟 Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$
2.  Compute $w = g^{N+1} \pmod{N}$

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

☀ Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$
2.  Compute $w = g^{N+1} \bmod N$
3.  Compute $\text{dlog}_g(w) = p + q$

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

★ Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$
2.  Compute $w = g^{N+1} \bmod N$
3.  Compute $\text{dlog}_g(w) = p + q$
4.  Factor N using $p + q$

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

★ Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$
2.  Compute $w = g^{N+1} \bmod N$
3.  Compute $\text{dlog}_g(w) = p + q$
4.  Factor N using $p + q$

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

★ Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$
2.  Compute $w = g^{N+1} \bmod N$
3.  Compute $\text{dlog}_g(w) = p + q$
4.  Factor N using $p + q$

So, we can factor via a discrete log. **Is this helpful?**

Factoring RSA integers via discrete logarithm

Foundational paper: Ekerå and Håstad '17 (arXiv:1702.00249)

Observation

Consider an integer $N = pq$.

For the vast majority of positive integers $g < N$:

$$g^{N+1} \equiv g^{p+q} \pmod{N}$$

★ Knowing $p + q$ is sufficient to factor N .

Plan

 = classical step,  = quantum step

1.  Pick an integer $g < N$
2.  Compute $w = g^{N+1} \bmod N$
3.  Compute $\text{dlog}_g(w) = p + q$
4.  Factor N using $p + q$

So, we can factor via a discrete log. **Is this helpful? Yes!**

Factoring RSA integers via discrete logarithm

Nowadays, all RSA factoring
circuits...

Factoring RSA integers via discrete logarithm

Nowadays, all RSA factoring
circuits...



Finding the period of
 $f(x) = a^x \bmod N$



Taking the discrete log
 $g^N \equiv g^{p+q-1} \pmod{N}$

Summary: are we focusing on the right problems?

Summary: are we focusing on the right problems?

- Elliptic curve discrete log is **more widely relied upon** than factoring

Summary: are we focusing on the right problems?

- Elliptic curve discrete log is **more widely relied upon** than factoring
- Elliptic curve discrete log has **cheaper quantum circuits** than factoring

Summary: are we focusing on the right problems?

- Elliptic curve discrete log is **more widely relied upon** than factoring
- Elliptic curve discrete log has **cheaper quantum circuits** than factoring
- Modern circuits for RSA factoring are **discrete log** circuits anyway

Summary: are we focusing on the right problems?

- Elliptic curve discrete log is **more widely relied upon** than factoring
- Elliptic curve discrete log has **cheaper quantum circuits** than factoring
- Modern circuits for RSA factoring are **discrete log** circuits anyway

Summary: are we focusing on the right problems?

- Elliptic curve discrete log is **more widely relied upon** than factoring
- Elliptic curve discrete log has **cheaper quantum circuits** than factoring
- Modern circuits for RSA factoring are **discrete log** circuits anyway

... but please **don't forget about factoring**, it's still used all over the place in practice!

Overview



Are we focusing on the right problems?

Co-design of algorithms and architectures: heuristics are changing

A case study: new factoring circuits

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?
- How do we tune the knobs of parametrized subroutines?

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?
- How do we tune the knobs of parametrized subroutines?
 - Example: carry-lookahead adders

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?
- How do we tune the knobs of parametrized subroutines?
 - Example: carry-lookahead adders
 - ▶ **Depth:** $O(\log n)$

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?
- How do we tune the knobs of parametrized subroutines?
 - Example: carry-lookahead adders
 - ▶ **Depth:** $O(\log n)$
 - ▶ **Ancillas:** $O(n/\log n)$

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?
- How do we tune the knobs of parametrized subroutines?
 - Example: carry-lookahead adders
 - ▶ **Depth:** $O(k)$
 - ▶ **Ancillas:** $O(n/k)$

Heuristics in logical circuit design

When designing quantum circuits, we are constantly making trade-offs at all scales.

Examples:

Macro-scale

- What algorithm should we use to compute a modular exponential in Shor's algorithm?
- What problems should we target for demonstrations (cryptographic proofs of quantumness)?
 - Discrete log?
 - Factoring?
 - Something else?

Micro-scale

- What quantum adder do we use as a subroutine?
- How do we tune the knobs of parametrized subroutines?
 - Example: carry-lookahead adders
 - ▶ **Depth:** $O(k)$
 - ▶ **Ancillas:** $O(n/k)$

How do we make these decisions? **Heuristics.**

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*?

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Metrics for cost

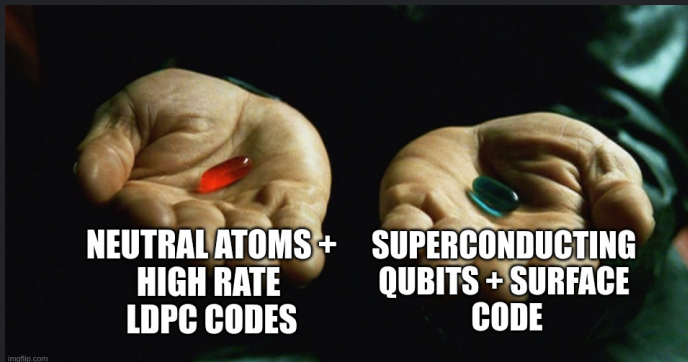
How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!



Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

**Surface code
+ superconducting**

**High rate LDPC
+ neutral atoms**

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

**Surface code
+ superconducting**

- Fast cycle time

**High rate LDPC
+ neutral atoms**

- Slow cycle time

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

**Surface code
+ superconducting**

- Fast cycle time
- Physical qubit count more constrained

**High rate LDPC
+ neutral atoms**

- Slow cycle time
- Physical qubit count less constrained

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

Surface code + superconducting

- Fast cycle time
- Physical qubit count more constrained
- Many physical qubits per logical

High rate LDPC + neutral atoms

- Slow cycle time
- Physical qubit count less constrained
- Fewer physical qubits per logical

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

Surface code + superconducting

- Fast cycle time
- Physical qubit count more constrained
- Many physical qubits per logical
- Parallelism has big space footprint

High rate LDPC + neutral atoms

- Slow cycle time
- Physical qubit count less constrained
- Fewer physical qubits per logical
- Parallelism more achievable (?)

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

Surface code + superconducting

- Fast cycle time
- Physical qubit count more constrained
- Many physical qubits per logical
- Parallelism has big space footprint
- **Minimize logical qubit count**

High rate LDPC + neutral atoms

- Slow cycle time
- Physical qubit count less constrained
- Fewer physical qubits per logical
- Parallelism more achievable (?)

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

Surface code + superconducting

- Fast cycle time
- Physical qubit count more constrained
- Many physical qubits per logical
- Parallelism has big space footprint
- **Minimize logical qubit count**

High rate LDPC + neutral atoms

- Slow cycle time
- Physical qubit count less constrained
- Fewer physical qubits per logical
- Parallelism more achievable (?)
- **Balance depth and qubit count**

Many algorithmic decisions made with **surface code in mind**.

Metrics for cost

How should we balance logical *qubits* vs *Toffolis*? (vs depth, other gates, ...)

Depends strongly on architecture!

Surface code + superconducting

- Fast cycle time
- Physical qubit count more constrained
- Many physical qubits per logical
- Parallelism has big space footprint
- **Minimize logical qubit count**

High rate LDPC + neutral atoms

- Slow cycle time
- Physical qubit count less constrained
- Fewer physical qubits per logical
- Parallelism more achievable (?)
- **Balance depth and qubit count**

Many algorithmic decisions made with **surface code in mind**.
We need to re-optimize under new heuristics!

Overview



Are we focusing on the right problems?

Co-design of algorithms and architectures: heuristics are changing

A case study: new factoring circuits

Very space-efficient discrete log modulo N (and thus RSA factoring)

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Very space-efficient discrete log modulo N (and thus RSA factoring)

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Very space-efficient discrete log modulo N (and thus RSA factoring)

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Result: quantum circuit to find $\text{dlog}_g(w) = s$
using $\sim \log_2 s$ qubits.

Very space-efficient discrete log modulo N (and thus RSA factoring)

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Result: quantum circuit to find $\text{dlog}_g(w) = s$
using $\sim \log_2 s$ qubits.

Recall: To factor n -bit RSA integers, take a discrete log where $\log_2 s \sim n/2$.

Very space-efficient discrete log modulo N (and thus RSA factoring)

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Result: quantum circuit to find $\text{dlog}_g(w) = s$
using $\sim \log_2 s$ qubits.

Recall: To factor n -bit RSA integers, take a discrete log where $\log_2 s \sim n/2$.

Corollary: Factoring n -bit RSA integers with
 $\sim n/2$ qubits.

Physical cost estimates for factoring RSA-2048

Physical cost estimates for factoring RSA-2048

	Conventional algorithm Shor '94	New algorithm Chevignard et al. '24, Gidney '25
Superconducting + surface code		
Neutral atoms + high rate LDPC		

Physical cost estimates for factoring RSA-2048

	Conventional algorithm Shor '94	New algorithm Chevignard et al. '24, Gidney '25
Superconducting + surface code	20 million qubits, 8 hours Gidney-Ekera '19	
Neutral atoms + high rate LDPC		

Physical cost estimates for factoring RSA-2048

	Conventional algorithm Shor '94	New algorithm Chevignard et al. '24, Gidney '25
Superconducting + surface code	20 million qubits, 8 hours Gidney-Ekera '19	1 million qubits, 1 week Gidney '25
Neutral atoms + high rate LDPC		

Physical cost estimates for factoring RSA-2048

	Conventional algorithm Shor '94	New algorithm Chevignard et al. '24, Gidney '25
Superconducting + surface code	20 million qubits, 8 hours Gidney-Ekera '19	1 million qubits, 1 week Gidney '25
Neutral atoms + high rate LDPC	< 100k qubits, 97 days Cain et al. '26	

Physical cost estimates for factoring RSA-2048

	Conventional algorithm Shor '94	New algorithm Chevignard et al. '24, Gidney '25
Superconducting + surface code	20 million qubits, 8 hours Gidney-Ekera '19	1 million qubits, 1 week Gidney '25
Neutral atoms + high rate LDPC	< 100k qubits, 97 days Cain et al. '26	13k qubits, 27 years Cain et al. '26

Physical cost estimates for factoring RSA-2048

	Conventional algorithm Shor '94	New algorithm Chevignard et al. '24, Gidney '25
Superconducting + surface code	20 million qubits, 8 hours Gidney-Ekera '19	1 million qubits, 1 week Gidney '25
Neutral atoms + high rate LDPC	< 100k qubits, 97 days Cain et al. '26	13k qubits, 27 years Cain et al. '26

Which algorithm is preferred **depends on the architecture.**

Digging a little deeper

Digging a little deeper

Does the new algorithm really take **27 years** on neutral atoms?

Digging a little deeper

Does the new algorithm really take **27 years** on neutral atoms?

- This estimate uses the logical circuit from Gidney '25

Digging a little deeper

Does the new algorithm really take **27 years** on neutral atoms?

- This estimate uses the logical circuit from Gidney '25
- That circuit has its knobs turned for **superconducting qubits**.

Parallelizing the new algorithm

To take a discrete log of $w = g^s \bmod N$:

$$\text{Bottleneck: } \sum_x |x\rangle |0\rangle \rightarrow \sum_x |x\rangle |f(x)\rangle$$

$|x\rangle$



$|f(x)\rangle$

Parallelizing the new algorithm

To take a discrete log of $w = g^s \bmod N$:

$$\text{Bottleneck: } \sum_x |x\rangle |0\rangle \rightarrow \sum_x |x\rangle |f(x)\rangle$$

$|x\rangle$

$|f(x)\rangle$ workspace



Parallelizing the new algorithm

To take a discrete log of $w = g^s \bmod N$:

$$\text{Bottleneck: } \sum_x |x\rangle |0\rangle \rightarrow \sum_x |x\rangle |f(x)\rangle$$



Original circuit:

- **Depth:** D
- **Space:** $\sim \log_2 s + O(\log \log N)$

Parallelizing the new algorithm

To take a discrete log of $w = g^s \bmod N$:

$$\text{Bottleneck: } \sum_x |x\rangle |0\rangle \rightarrow \sum_x |x\rangle |f(x)\rangle$$



Original circuit:

- **Depth:** D
- **Space:** $\sim \log_2 s + O(\log \log N)$

“2-threaded” version:

- **Depth:** $D/2$

Parallelizing the new algorithm

To take a discrete log of $w = g^s \bmod N$:

$$\text{Bottleneck: } \sum_x |x\rangle |0\rangle \rightarrow \sum_x |x\rangle |f(x)\rangle$$



Original circuit:

- **Depth:** D
- **Space:** $\sim \log_2 s + O(\log \log N)$

“2-threaded” version:

- **Depth:** $D/2$
- **Space:** $\sim \log_2 s + 2 \cdot O(\log \log N)$

Parallelizing the new algorithm

To take a discrete log of $w = g^s \bmod N$:

$$\text{Bottleneck: } \sum_x |x\rangle |0\rangle \rightarrow \sum_x |x\rangle |f(x)\rangle$$



Original circuit:

- **Depth:** D
- **Space:** $\sim \log_2 s + O(\log \log N)$

“ p -threaded” version:

- **Depth:** D/p
- **Space:** $\sim \log_2 s + p \cdot O(\log \log N)$

Parallelizing the new algorithm

p -threaded version:

- **Depth:** D/p
- **Space:** $\sim \log_2 s + p \cdot O(\log \log N)$

Parallelizing the new algorithm

p -threaded version:

- **Depth:** D/p
- **Space:** $\sim \log_2 s + p \cdot O(\log \log N)$

Why would you ever not do this?

Parallelizing the new algorithm

p -threaded version:

- **Depth:** D/p
- **Space:** $\sim \log_2 s + p \cdot O(\log \log N)$

Why would you ever not do this?

Because of the *architecture*.

Test-case takeaway 1

The 27 year runtime for the new algorithm on neutral atoms comes from leaving some “knobs” tuned to superconducting qubits.

Test-case takeaway 1

The 27 year runtime for the new algorithm on neutral atoms comes from leaving some “knobs” tuned to superconducting qubits.

But **those knobs were only in folklore!**

Test-case takeaway 1

The 27 year runtime for the new algorithm on neutral atoms comes from leaving some “knobs” tuned to superconducting qubits.

But **those knobs were only in folklore!**

Logical circuit designers: we need to publish
parametrized circuits.

Is factoring the right problem, redux

Is factoring the right problem, redux

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Result: quantum circuit to find $\text{dlog}_g(w) = s$
using $\sim \log_2 s$ qubits.

Is factoring the right problem, redux

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Result: quantum circuit to find $\text{dlog}_g(w) = s$
using $\sim \log_2 s$ qubits.

Proof of quantumness: forget about factoring...

Is factoring the right problem, redux

Paper 2024/222

Reducing the Number of Qubits in Quantum Factoring

Clémence Chevnard, Univ Rennes, Inria, CNRS, IRISA

Pierre-Alain Fouque , Univ Rennes, Inria, CNRS, IRISA

André Schrottenloher , Univ Rennes, Inria, CNRS, IRISA

eprint:2024/222

How to factor 2048 bit RSA integers with less than a million noisy qubits

Craig Gidney

Google Quantum AI, Santa Barbara, California 93117, USA

June 9, 2025

arXiv:2505.15917

Recall the *discrete log modulo N* :
 $\Rightarrow$ given g, N and $w = g^s \bmod N$, find s .

Result: quantum circuit to find $\text{dlog}_g(w) = s$
using $\sim \log_2 s$ qubits.

Proof of quantumness: forget about factoring...

What if we set up a discrete log problem in which $\log_2 s$ is as small as possible?

Recall: costs from beginning of talk

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Recall: costs from beginning of talk

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Finite field discrete log

Given $w \equiv g^s \pmod{N}$ find s .

For 2048-bit N and 225-bit s :

Recall: costs from beginning of talk

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Finite field discrete log

Given $w \equiv g^s \pmod{N}$ find s .

For 2048-bit N and 225-bit s :

- Qubits: ~ 350
- Toffolis: ~ 70 million

(Preliminary estimate!)

Recall: costs from beginning of talk

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Finite field discrete log

Given $w \equiv g^s \pmod{N}$ find s .

For 2048-bit N and 225-bit s :

- Qubits: ~ 350
- Toffolis: ~ 70 million

(Preliminary estimate!)

Good thing finite field discrete log isn't really used in practice, right?

Finite-field discrete log is used in standardized cryptography

RFC 7919: Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)

D. Gillmor

PROPOSED STANDARD

Internet Engineering Task Force (IETF)

Request for Comments: 7919

Updates: [2246](#), [4346](#), [4492](#), [5246](#)

Category: Standards Track

ISSN: 2070-1721

D. Gillmor

ACLU

August 2016

Finite-field discrete log is used in standardized cryptography

RFC 7919: Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)

D. Gillmor

PROPOSED STANDARD

Internet Engineering Task Force (IETF)

Request for Comments: 7919

Updates: [2246](#), [4346](#), [4492](#), [5246](#)

Category: Standards Track

ISSN: 2070-1721

D. Gillmor

ACLU

August 2016



Finite-field discrete log is used in standardized cryptography

RFC 7919: Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)

D. Gillmor

PROPOSED STANDARD

Internet Engineering Task Force (IETF)
Request for Comments: 7919
Updates: [2246](#), [4346](#), [4492](#), [5246](#)
Category: Standards Track
ISSN: 2070-1721

D. Gillmor
ACLU
August 2016



OK, but surely the parameters aren't set so the exponent is small, right?

The parameters are absolutely set like that

RFC 7919: Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)

D. Gillmor

PROPOSED STANDARD

Internet Engineering Task Force (IETF)
Request for Comments: 7919
Updates: [2246](#), [4346](#), [4492](#), [5246](#)
Category: Standards Track
ISSN: 2070-1721

D. Gillmor
ACLU
August 2016

“Peers using [a **2048-bit modulus**] ... should choose a **secret key of at least 225 bits.**”

The parameters are absolutely set like that

RFC 7919: Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)

D. Gillmor

PROPOSED STANDARD

Internet Engineering Task Force (IETF)
Request for Comments: 7919
Updates: [2246](#), [4346](#), [4492](#), [5246](#)
Category: Standards Track
ISSN: 2070-1721

D. Gillmor
ACLU
August 2016



“Peers using [a **2048-bit modulus**] ... should choose a **secret key of at least 225 bits.**”

Comparing costs

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Finite field discrete log

Given $w \equiv g^s \pmod{N}$ find s .

For 2048-bit N and 225-bit s :

- Qubits: ~ 350
- Toffolis: ~ 70 million

(Preliminary estimate!)

All of these correspond to real cryptographic standards!

A small comfort

From a recent conversation with a practitioner:

A small comfort

From a recent conversation with a practitioner:

“A big part of my job is understanding what cryptography is currently being used in real-world applications.

A small comfort

From a recent conversation with a practitioner:

“A big part of my job is understanding what cryptography is currently being used in real-world applications.
I don't think I've seen a single instance of finite-field discrete log being actively used in practice.”

A small comfort

From a recent conversation with a practitioner:

“A big part of my job is understanding what cryptography is currently being used in real-world applications.
I don't think I've seen a single instance of finite-field discrete log being actively used in practice.”

My response: “But it's part of the TLS 1.2 standard!”

A small comfort

From a recent conversation with a practitioner:

“A big part of my job is understanding what cryptography is currently being used in real-world applications.
I don't think I've seen a single instance of finite-field discrete log being actively used in practice.”

My response: “But it's part of the TLS 1.2 standard!”

“There's a lot of weird stuff in TLS 1.2”

Test-case takeaway 2

Finite-field discrete logarithm is perhaps the **most vulnerable** standardized cryptographic assumption.

Hot take

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Finite field discrete log

Given $w \equiv g^s \pmod{N}$ find s .

For 2048-bit N and 225-bit s :

- Qubits: ~ 350
- Toffolis: ~ 70 million

(Preliminary estimate!)

Hot take

Note: these are **logical** costs, in the abstract circuit model.

Elliptic curve discrete log

Given $w = [s]g$, find s .

For curve secp256k1:

- Qubits: ~ 900
- Toffolis: ~ 70 million

Source: ecdsa.fail

RSA integer factorization

Given $N = pq$, find p and q .

For RSA-2048:

- Qubits: ~ 1400
- Toffolis: $\sim 7,000$ million

Gidney '25 (arXiv:2505.15917)

Finite field discrete log

Given $w \equiv g^s \pmod{N}$ find s .

For 2048-bit N and 225-bit s :

- Qubits: ~ 350
- Toffolis: ~ 70 million

(Preliminary estimate!)

Do these differences in cost even *matter*?

Summary



Summary



- Until recently, overwhelming focus on **factoring** even though it's:

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically
- Rapidly evolving **architectures** have changed how we evaluate cost of logical circuits

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically
- Rapidly evolving **architectures** have changed how we evaluate cost of logical circuits
- Need to revisit:

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically
- Rapidly evolving **architectures** have changed how we evaluate cost of logical circuits
- Need to revisit:
 - **Macro-scale:** algorithms and target problems

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically
- Rapidly evolving **architectures** have changed how we evaluate cost of logical circuits
- Need to revisit:
 - **Macro-scale:** algorithms and target problems
 - **Micro-scale:** subroutines and adjustable parameters

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically
- Rapidly evolving **architectures** have changed how we evaluate cost of logical circuits
- Need to revisit:
 - **Macro-scale:** algorithms and target problems
 - **Micro-scale:** subroutines and adjustable parameters

Summary



- Until recently, overwhelming focus on **factoring** even though it's:
 - more expensive quantumly
 - less widely used classically
- Rapidly evolving **architectures** have changed how we evaluate cost of logical circuits
- Need to revisit:
 - **Macro-scale:** algorithms and target problems
 - **Micro-scale:** subroutines and adjustable parameters

Not only has the timeline for cryptographic vulnerability shifted,
but also the **uncertainty has increased**



INTELLIGENCE COMMUNITY
POSTDOCTORAL RESEARCH
FELLOWSHIP PROGRAM



SIMONS
INSTITUTE
for the Theory of Computing

Thank you!

Email: gkm@mit.edu

Website: <https://gmeyer.net/>



https://gmeyer.net/blog/posts/tqc_2026

Relevant series of blog posts

GDKM was supported by an appointment to the Intelligence Community Postdoctoral Research Fellowship Program at MIT administered by Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy and the Office of the Director of National Intelligence (ODNI).