(Some) quantum speedups are...

$|\text{alive}\rangle + |\text{dead}\rangle$

Fernando Virdia

NOVA.ID.FCT
Universidade NOVA de Lisboa

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

- Does Grover key-search really work? [JNRV20]

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

- Does Grover key-search really work? [JNRV20]
- Does quantum lattice sieving really work? [AGPS20]

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

- Does Grover key-search really work? [JNRV20]
- Does quantum lattice sieving really work? [AGPS20]
- Does quantum lattice enumeration really work? [BBTV23]

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

- Does Grover key-search really work? [JNRV20]
- Does quantum lattice sieving really work? [AGPS20]
- Does quantum lattice enumeration really work? [BBTV23]

Disclaimer

I don't know.

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

- Does Grover key-search really work? [JNRV20]
- Does quantum lattice sieving really work? [AGPS20]
- Does quantum lattice enumeration really work? [BBTV23]

Disclaimer

I don't know. **Opinion**: I think as currently stated, no.

I have a big defect, I'm a contrarian. This whole talk is me going "well, actually".

- Does Grover key-search really work? [JNRV20]
- Does quantum lattice sieving really work? [AGPS20]
- Does quantum lattice enumeration really work? [BBTV23]

Disclaimer

I don't know. **Opinion**: I think as currently stated, no.
However we never know, these are just arguments against them.

Let's step back. There are mostly two kinds of quantum cryptanalysis:

Let's step back. There are mostly two kinds of quantum cryptanalysis:

- Algorithms turning hard problems into easy ones (e.g., Shor's)

Let's step back. There are mostly two kinds of quantum cryptanalysis:

- Algorithms turning hard problems into easy ones (e.g., Shor's)
- Algorithms turning hard problems into $\sqrt{\text{hard}}$ problems (e.g., Grover's)

Let's step back. There are mostly two kinds of quantum cryptanalysis:

- Algorithms turning hard problems into easy ones (e.g., Shor's)
- Algorithms turning hard problems into $\sqrt{\text{hard}}$ problems (e.g., Grover's)
- The first kind usually looks entirely different from the classical known attacks

Let's step back. There are mostly two kinds of quantum cryptanalysis:

- Algorithms turning hard problems into easy ones (e.g., Shor's)
- Algorithms turning hard problems into $\sqrt{\text{hard}}$ problems (e.g., Grover's)
- The first kind usually looks entirely different from the classical known attacks
- The second kind are usually used as "black-box" subroutines to classical attacks

Let's step back. There are mostly two kinds of quantum cryptanalysis:

- Algorithms turning hard problems into easy ones (e.g., Shor's)
- Algorithms turning hard problems into $\sqrt{\text{hard}}$ problems (e.g., Grover's)
- The first kind usually looks entirely different from the classical known attacks
- The second kind are usually used as "black-box" subroutines to classical attacks

I will be talking about the latter.

Quantum computation

Let $\mathcal{X}$ be a finite set. Attacks often use three "operations":

Quantum computation

Let $\mathcal{X}$ be a finite set. Attacks often use three "operations":

Evaluating f

$$U_f \cdot \sum_{x \in \mathcal{X}} c_x |x\rangle |0\rangle \mapsto \sum_{x \in \mathcal{X}} c_x |x\rangle |f(x)\rangle \text{, for } c_x \in \mathbb{C} \text{ and } \sum_{x \in \mathcal{X}} |c_x|^2 = 1$$

Quantum computation

Let $\mathcal{X}$ be a finite set. Attacks often use three "operations":

Evaluating f

$$U_f \cdot \sum_{x\in\mathcal{X}} c_x \left|x\right\rangle \left|0\right\rangle \mapsto \sum_{x\in\mathcal{X}} c_x \left|x\right\rangle \left|f(x)\right\rangle, \text{ for } c_x \in \mathbb{C} \text{ and } \sum_{x\in\mathcal{X}} |c_x|^2 = 1$$

Modifying the amplitudes c_x

$$U_{\text{amp}} \cdot \sum_{x\in\mathcal{X}} c_x \left|x\right\rangle \left|f(x)\right\rangle \mapsto \sum_{x\in\mathcal{X}} d_x \left|x\right\rangle \left|f(x)\right\rangle, \text{ for } d_x \in \mathbb{C} \text{ and } \sum_{x\in\mathcal{X}} |d_x|^2 = 1$$

and some x such that $c_x \neq d_x$.

Quantum computation

Let $\mathcal{X}$ be a finite set. Attacks often use three "operations":

Evaluating f

$$U_f \cdot \sum_{x\in\mathcal{X}} c_x \left|x\right\rangle \left|0\right\rangle \mapsto \sum_{x\in\mathcal{X}} c_x \left|x\right\rangle \left|f(x)\right\rangle, \text{ for } c_x \in \mathbb{C} \text{ and } \sum_{x\in\mathcal{X}} |c_x|^2 = 1$$

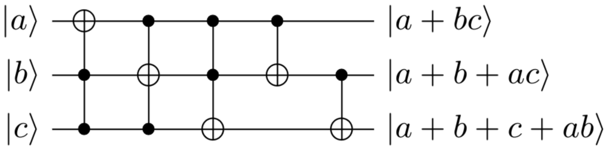
Modifying the amplitudes c_x

$$U_{\text{amp}} \cdot \sum_{x\in\mathcal{X}} c_x \left|x\right\rangle \left|f(x)\right\rangle \mapsto \sum_{x\in\mathcal{X}} d_x \left|x\right\rangle \left|f(x)\right\rangle, \text{ for } d_x \in \mathbb{C} \text{ and } \sum_{x\in\mathcal{X}} |d_x|^2 = 1$$

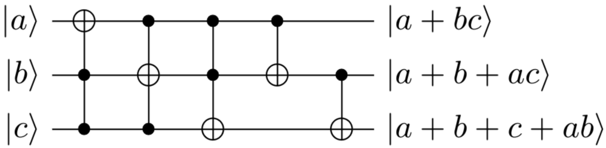
and some x such that $c_x \neq d_x$.

Measuring the register

$$\sum_{x\in\mathcal{X}} d_x \left|x\right\rangle \left|f(x)\right\rangle \mapsto \left|x_0\right\rangle \left|f(x_0)\right\rangle, \text{ for some } x_0 \in \mathcal{X} \text{ with probability } |d_{x_0}|^2$$



This is a quantum circuit of width 3, depth 5 and gate count 5.



This is a quantum circuit of width 3, depth 5 and gate count 5.

Comparing cost with classical circuits

We can compare the # of quantum gates with classical cycles [JS19] (G metric).
If we assume active memory correction, we can use depth $\times$ width (DW metric).

AES key search using Grover’s algorithm

Unstructured search

(N. M)-unstructured search problem

Given a randomly sorted list L of size N and a property $f(\cdot)$ such that exactly M elements of L satisfy $f(\cdot)$, find one such element.

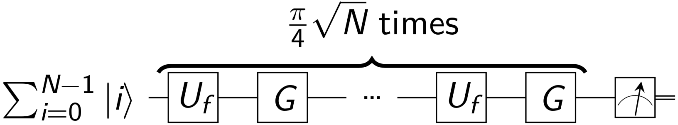


Figure: Grover search circuit when $M = 1$.

Unstructured search

(N. M)-unstructured search problem

Given a randomly sorted list L of size N and a property $f(\cdot)$ such that exactly M elements of L satisfy $f(\cdot)$, find one such element.

$\implies$ Classically this requires $O(N/M)$ steps, Grover's solves it in $O(\sqrt{N/M})$ steps.

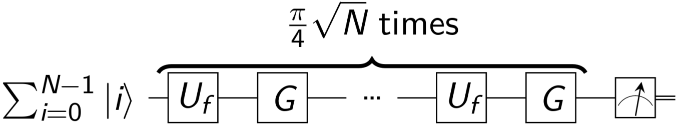


Figure: Grover search circuit when $M = 1$.

AES block cipher

Block cipher with encryption function $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$.
$E(\cdot, m)$ considered indistinguishable from a random function over $\{0,1\}^k \mapsto \{0,1\}^n$.

AES block cipher

Block cipher with encryption function $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$.
$E(\cdot, m)$ considered indistinguishable from a random function over $\{0,1\}^k \mapsto \{0,1\}^n$.

Attacking AES: given (m, c), find k such that $c \leftarrow E(k, m)$.

AES block cipher

Block cipher with encryption function $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$.
$E(\cdot, m)$ considered indistinguishable from a random function over $\{0,1\}^k \mapsto \{0,1\}^n$.

Attacking AES: given (m, c), find k such that $c \leftarrow E(k, m)$.

Since $E(\cdot, m) \sim \$$, this is an unstructured search in $\{0,1\}^k$.
$\implies$ Classical runtime $\approx 2^k$ encryptions, one per key
$\implies$ Quantum runtime $\approx 2^{k/2}$ Grover steps

Asymptotically, we are “done” with cryptanalysis: 2^k vs $2^{k/2}$ means doubling the key length k is enough.

Asymptotically, we are “done” with cryptanalysis: 2^k vs $2^{k/2}$ means doubling the key length k is enough.

Why attempt a non-asymptotic cryptanalysis?

Asymptotically, we are "done" with cryptanalysis: 2^k vs $2^{k/2}$ means doubling the key length k is enough.

Why attempt a non-asymptotic cryptanalysis?

- General reason: doubling keys may be practically inconvenient (and overkill).

Asymptotically, we are “done” with cryptanalysis: 2^k vs $2^{k/2}$ means doubling the key length k is enough.

Why attempt a non-asymptotic cryptanalysis?

- General reason: doubling keys may be practically inconvenient (and overkill).
- Particular reason: the hardness of AES is being used as a definition of security.

Asymptotically, we are “done” with cryptanalysis: 2^k vs $2^{k/2}$ means doubling the key length k is enough.

Why attempt a non-asymptotic cryptanalysis?

- General reason: doubling keys may be practically inconvenient (and overkill).
- Particular reason: the hardness of AES is being used as a definition of security.

NIST Post-Quantum Cryptography standardisation

- Since 2017, the US NIST has been running a process to standardise post-quantum public-key cryptographic schemes.
- To qualify for ”category 5” security, a scheme should be as secure as AES-256.

Where should we start with non-asymptotyc cryptanalysis?

Where should we start with non-asymptotyc cryptanalysis?

- First, an asymptotically smaller issue: we have been ignoring the cost of U_f.

Where should we start with non-asymptotyc cryptanalysis?

- First, an asymptotically smaller issue: we have been ignoring the cost of U_f.
  - Our implementations suggest $\approx 2^{20}$ gates [JNRV20]
  - Follow up work reduces this somewhat [ZWS⁺20, JBK⁺22, HS22] ($\approx$ 2 bits smaller)

Where should we start with non-asymptotyc cryptanalysis?

- First, an asymptotically smaller issue: we have been ignoring the cost of U_f.
  - Our implementations suggest $\approx 2^{20}$ gates [JNRV20]
  - Follow up work reduces this somewhat [ZWS+20, JBK+22, HS22] ($\approx$ 2 bits smaller)
- Second, a bigger issue: the quantum computation model is too generous to the attacker.

Where should we start with non-asymptotyc cryptanalysis?

- First, an asymptotically smaller issue: we have been ignoring the cost of U_f.
  - Our implementations suggest $\approx 2^{20}$ gates [JNRV20]
  - Follow up work reduces this somewhat [ZWS$^+$20, JBK$^+$22, HS22] ($\approx$ 2 bits smaller)
- Second, a bigger issue: the quantum computation model is too generous to the attacker.

Let's talk quantum state decoherence

Quantum state decoherence

- Classical memory is easy to error-correct, quantum memory not at all

Quantum state decoherence

- Classical memory is easy to error-correct, quantum memory not at all
- Current qubits need near-absolute-zero temperatures; yet, operating on them quickly leads to signal loss

Quantum state decoherence

- Classical memory is easy to error-correct, quantum memory not at all
- Current qubits need near-absolute-zero temperatures; yet, operating on them quickly leads to signal loss

New constraint: max-depth (MD)

Consider limiting the depth of quantum circuit [Nat16]:

Quantum state decoherence

- Classical memory is easy to error-correct, quantum memory not at all
- Current qubits need near-absolute-zero temperatures; yet, operating on them quickly leads to signal loss

New constraint: max-depth (MD)

Consider limiting the depth of quantum circuit [Nat16]:

- $MD = 2^{40} \approx$ "gates that presently envisioned quantum computing architectures are expected to serially perform in a year"

Quantum state decoherence

- Classical memory is easy to error-correct, quantum memory not at all
- Current qubits need near-absolute-zero temperatures; yet, operating on them quickly leads to signal loss

New constraint: max-depth (MD)

Consider limiting the depth of quantum circuit [Nat16]:

- $MD = 2^{40} \approx$ "gates that presently envisioned quantum computing architectures are expected to serially perform in a year"
- $MD = 2^{64} \approx$ "gates that current classical computing architectures can perform serially in a decade"

Quantum state decoherence

- Classical memory is easy to error-correct, quantum memory not at all
- Current qubits need near-absolute-zero temperatures; yet, operating on them quickly leads to signal loss

New constraint: max-depth (MD)

Consider limiting the depth of quantum circuit [Nat16]:

- $MD = 2^{40} \approx$ "gates that presently envisioned quantum computing architectures are expected to serially perform in a year"
- $MD = 2^{64} \approx$ "gates that current classical computing architectures can perform serially in a decade"
- $MD = 2^{96} \approx$ "gates that atomic scale qubits with speed of light propagation times could perform in a millennium"

Consequences of *MD*

- NIST considers a hard limit $MD \in \{2^{40}, 2^{64}, 2^{96}\}$.
- AES-256: $MD < 2^{k/2} = 2^{128}$, what is naively required by Grover's

Consequences of MD

- NIST considers a hard limit $MD \in \{2^{40}, 2^{64}, 2^{96}\}$.
- AES-256: $MD < 2^{k/2} = 2^{128}$, what is naively required by Grover's
- Grover search almost certainly fails if stopped early; can't rinse-and-repeat

Consequences of MD

- NIST considers a hard limit $MD \in \{2^{40}, 2^{64}, 2^{96}\}$.
- AES-256: $MD < 2^{k/2} = 2^{128}$, what is naively required by Grover's
- Grover search almost certainly fails if stopped early; can't rinse-and-repeat
  $\implies$ We need to account for Grover's parallelisation.

Consequences of MD

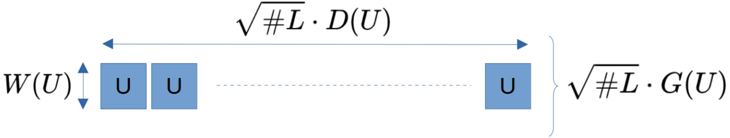
- NIST considers a hard limit $MD \in \{2^{40}, 2^{64}, 2^{96}\}$.
- AES-256: $MD < 2^{k/2} = 2^{128}$, what is naively required by Grover's
- Grover search almost certainly fails if stopped early; can't rinse-and-repeat
  $\implies$ We need to account for Grover's parallelisation.

Issue

Grover parallelises badly [Zal99]. Rule of thumb: need S machines for $\sqrt{S}$ speed-up.

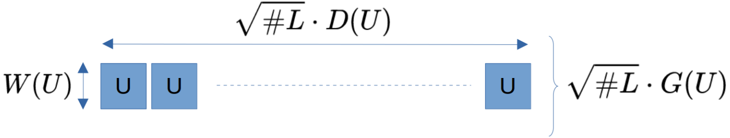
Example: Parallel Grover

Let L be a list to search and U a “Grover step”



Example: Parallel Grover

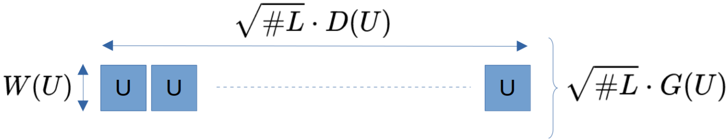
Let L be a list to search and U a "Grover step"



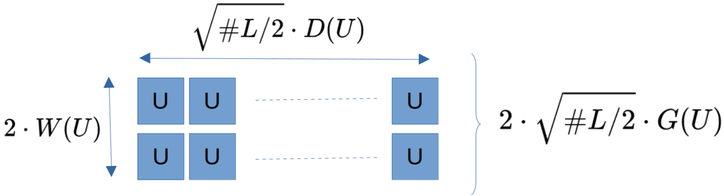
Divide $L = L_1 \cup L_2$ with $\#L_1 = \#L_2 = \#L/2$,

Example: Parallel Grover

Let L be a list to search and U a "Grover step"



Divide $L = L_1 \cup L_2$ with $\#L_1 = \#L_2 = \#L/2$,



Example: Parallel Grover

In general, using S machines,

- The circuit width $\mapsto S \cdot W(U)$
- The circuit depth $\mapsto \sqrt{\#L} \cdot D(U)/\sqrt{S}$
- The circuit gate count $\mapsto \sqrt{\#L} \cdot G(U) \cdot \sqrt{S}$

Example: Parallel Grover

In general, using S machines,

- The circuit width $\mapsto S \cdot W(U)$
- The circuit depth $\mapsto \sqrt{\#L} \cdot D(U)/\sqrt{S}$
- The circuit gate count $\mapsto \sqrt{\#L} \cdot G(U) \cdot \sqrt{S}$

This leads to gate counts. For a fully analysis in our setting, see [JNRV20].

Resulting estimates

Cipher	Gate-count for *MD*				
	∞, query	∞, gates	2^{40}	2^{64}	2^{96}
AES-128	2^{64}	2^{83}	2^{117}	2^{93}	$*2^{83}$
AES-192	2^{96}	2^{114}	2^{181}	2^{157}	2^{126}
AES-256	2^{128}	2^{148}	2^{245}	2^{221}	2^{190}

Slightly smaller numbers have since been obtained in the same computational model.

Resulting estimates

Cipher	Gate-count for *MD*				
	∞, query	∞, gates	2^{40}	2^{64}	2^{96}
AES-128	2^{64}	2^{83}	2^{117}	2^{93}	$*2^{83}$
AES-192	2^{96}	2^{114}	2^{181}	2^{157}	2^{126}
AES-256	2^{128}	2^{148}	2^{245}	2^{221}	2^{190}

$\Longrightarrow$ Quantum speed-ups with depth limit not as dramatic for symmetric crypto.

Slightly smaller numbers have since been obtained in the same computational model.

An interlude: Quantum lattice sieving

Lattice sieving using Grover's algorithm

- Lattice point sieving is the currently fastest Short Vector Problem solver available at experimental size

Lattice sieving using Grover's algorithm

- Lattice point sieving is the currently fastest Short Vector Problem solver available at experimental size
- To find short vectors in a lattice Λ, sieving
  - samples a list L of exponentially many vectors $v_i \in \Lambda$

Lattice sieving using Grover's algorithm

- Lattice point sieving is the currently fastest Short Vector Problem solver available at experimental size
- To find short vectors in a lattice Λ, sieving
  - samples a list L of exponentially many vectors $v_i \in \Lambda$
  - performs nearest neighbour search (NNS) on L to create a list L' of shorter vectors

Lattice sieving using Grover's algorithm

- Lattice point sieving is the currently fastest Short Vector Problem solver available at experimental size
- To find short vectors in a lattice Λ, sieving
  - samples a list L of exponentially many vectors $v_i \in \Lambda$
  - performs nearest neighbour search (NNS) on L to create a list L' of shorter vectors
  - repeats NNS multiple times, if L is long enough, a short vector is found

Lattice sieving using Grover's algorithm

- Lattice point sieving is the currently fastest Short Vector Problem solver available at experimental size
- To find short vectors in a lattice Λ, sieving
  - samples a list L of exponentially many vectors $v_i \in \Lambda$
  - performs nearest neighbour search (NNS) on L to create a list L' of shorter vectors
  - repeats NNS multiple times, if L is long enough, a short vector is found
- NNS internally performes unstructured search! $\implies$ "Groverise" (really, "filtered quantum search")

- Many lattice sieves exist [AKS01, NV08, Laa15, ADH^{+}19]

- Many lattice sieves exist [AKS01, NV08, Laa15, ADH+19]
- At the time of publication of [AGPS20], the asymptotically faster quantum sieve was from [BDGL16]
  - Classical complexity $2^{0.292n+o(1)}$, quantum complexity $2^{0.265n+o(1)}$

- Many lattice sieves exist [AKS01, NV08, Laa15, ADH+19]
- At the time of publication of [AGPS20], the asymptotically faster quantum sieve was from [BDGL16]
  - Classical complexity $2^{0.292n+o(1)}$, quantum complexity $2^{0.265n+o(1)}$
- Theoretically, using the quantum sped-up version should save $\approx 2^{(0.292-0.265)n}$ effort

- Many lattice sieves exist [AKS01, NV08, Laa15, ADH+19]
- At the time of publication of [AGPS20], the asymptotically faster quantum sieve was from [BDGL16]
  - Classical complexity $2^{0.292n+o(1)}$, quantum complexity $2^{0.265n+o(1)}$
- Theoretically, using the quantum sped-up version should save $\approx 2^{(0.292-0.265)n}$ effort

Forget max-depth. [AGPS20] ask: how does error correction overhead impact the quantum advantage?

Albrecht, Gheorghiu, Postlethwaite and Schanck consider using four cost metrics:

Albrecht, Gheorghiu, Postlethwaite and Schanck consider using four cost metrics:

- count gates: assumes idle qubits don't require error correction

Albrecht, Gheorghiu, Postlethwaite and Schanck consider using four cost metrics:

- count gates: assumes idle qubits don't require error correction
- count depth-width: assumes idle qubits require error correction, costing $\Theta(1)$ ops.

Albrecht, Gheorghiu, Postlethwaite and Schanck consider using four cost metrics:

- count gates: assumes idle qubits don't require error correction
- count depth-width: assumes idle qubits require error correction, costing $\Theta(1)$ ops.
- count DW in the surface code: idle qubit error correction costs $\Omega(\log^2(DW))$ ops.

Albrecht, Gheorghiu, Postlethwaite and Schanck consider using four cost metrics:

- count gates: assumes idle qubits don't require error correction
- count depth-width: assumes idle qubits require error correction, costing $\Theta(1)$ ops.
- count DW in the surface code: idle qubit error correction costs $\Omega(\log^2(DW))$ ops.
- surface code beyond asymptotics (Gidney-Ekerå, [GE21]): under mild engineering assumptions, choose attack parameters minimising estimated concrete overhead

Albrecht, Gheorghiu, Postlethwaite and Schanck consider using four cost metrics:

- count gates: assumes idle qubits don't require error correction
- count depth-width: assumes idle qubits require error correction, costing $\Theta(1)$ ops.
- count DW in the surface code: idle qubit error correction costs $\Omega(\log^2(DW))$ ops.
- surface code beyond asymptotics (Gidney-Ekerå, [GE21]): under mild engineering assumptions, choose attack parameters minimising estimated concrete overhead

They adapt the code of [GE21] to their quantum NNS circuits, and compare with asymptotic gate cost.

What's the impact of error correction?

Quantum metric	n	log time$_c$	log depth$_Q$	advantage factor
Asymptotic # of gates	312	91	83	2^8
Gidney-Ekerå	312	119	119	2^0
Asymptotic # of gates	352	103	93	2^{10}
Gidney-Ekerå	352	130	128	2^2
Asymptotic # of gates	544	159	144	2^{15}
Gidney-Ekerå	544	189	182	2^7
Asymptotic # of gates	824	241	218	2^{23}
Gidney-Ekerå	824	270	256	2^{14}

Observation

Error-correction considerations practically reduce the advantage by about 2^8 throughout all cryptanalytically interesting dimensions.
$\Longrightarrow$ The larger the dimension, the more appealing is quantum sieving.

Observation

Error-correction considerations practically reduce the advantage by about 2^8 throughout all cryptanalytically interesting dimensions.
$\Longrightarrow$ The larger the dimension, the more appealing is quantum sieving.

This is opposite to the effect of applying max-depth constraints. For fixed MD, the larger the key space, the smaller the advantage of running Grover search.

Observation

Error-correction considerations practically reduce the advantage by about 2^8 throughout all cryptanalytically interesting dimensions.
$\Longrightarrow$ The larger the dimension, the more appealing is quantum sieving.

This is opposite to the effect of applying max-depth constraints. For fixed *MD*, the larger the key space, the smaller the advantage of running Grover search.

Open follow-up: Would combining both kill advantages at both ends?

New result: Quantum lattice enumeration

Lattice enumeration

- The other main Short Vector Problem solver
- In dimension n, poly(n) memory, $2^{\frac{1}{8}n\log n + o(n)}$ time

Lattice enumeration

- The other main Short Vector Problem solver
- In dimension n, poly(n) memory, $2^{\frac{1}{8} n \log n + o(n)}$ time
- Given a lattice basis $(b_1, \ldots, b_n)$, it proceeds by identifying all short-enough vectors in $\langle b_n \rangle$, then $\langle b_{n-1}, b_n \rangle, \ldots$ via depth-first search

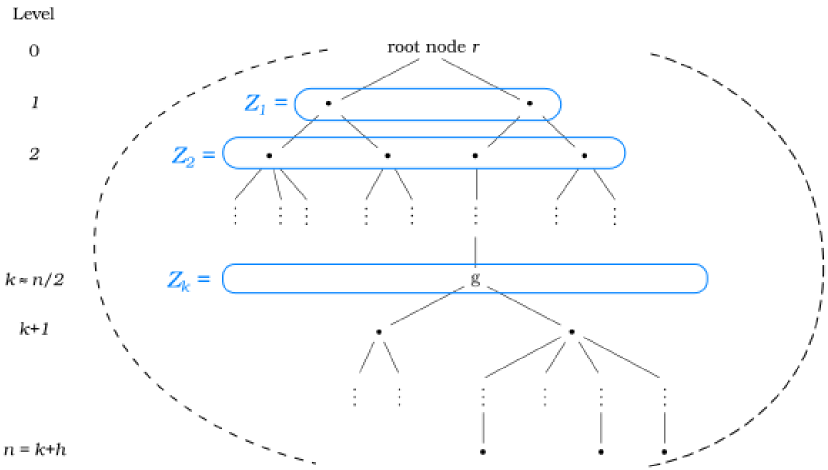
Lattice enumeration

- The other main Short Vector Problem solver
- In dimension n, poly(n) memory, $2^{\frac{1}{8}n\log n + o(n)}$ time
- Given a lattice basis $(b_1, \ldots, b_n)$, it proceeds by identifying all short-enough vectors in $\langle b_n \rangle$, then $\langle b_{n-1}, b_n \rangle, \ldots$ via depth-first search
- It terminates when a returning a short vector in $\langle b_1, \ldots, b_n \rangle$

Lattice enumeration

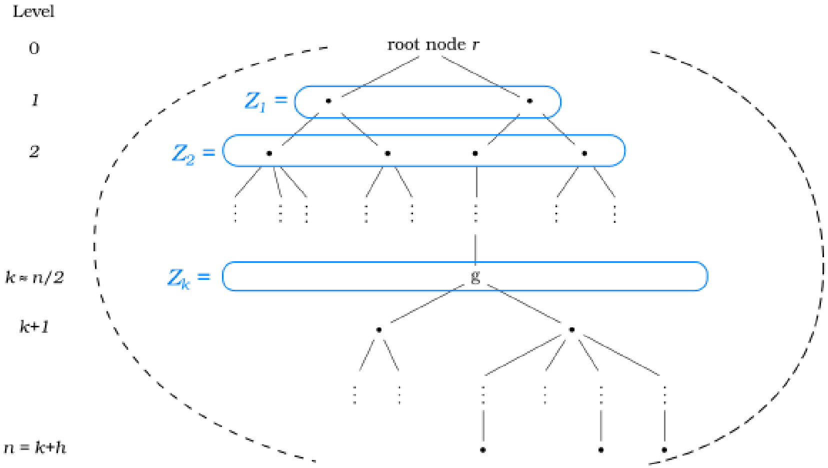
- The other main Short Vector Problem solver
- In dimension n, poly(n) memory, $2^{\frac{1}{8}n\log n+o(n)}$ time
- Given a lattice basis $(b_1, \ldots, b_n)$, it proceeds by identifying all short-enough vectors in $\langle b_n \rangle$, then $\langle b_{n-1}, b_n \rangle, \ldots$ via depth-first search
- It terminates when a returning a short vector in $\langle b_1, \ldots, b_n \rangle$
- It is naturally interpreted as searching for a "marked leaf" in a tree, where "marked" = "short"

A look at the enumeration tree



- Nodes divided on levels

A look at the enumeration tree



- Nodes divided on levels
- "Middle" levels super-exponentially large [GNR10]:

$$\#T \approx \#Z_{n/2}$$

Quantum tree search

- In 2018, Montanaro introduces two quantum tree-search algorithm, DetectMV and FindMV [Mon18]

Quantum tree search

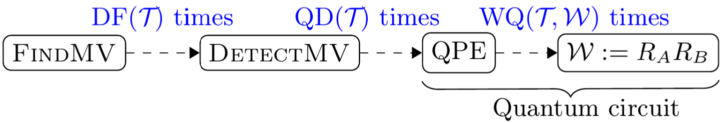
- In 2018, Montanaro introduces two quantum tree-search algorithm, DetectMV and FindMV [Mon18]
- Given a tree T and a predicate P, DetectMV returns whether $\exists x \in T$ such that $P(x) = \top$

Quantum tree search

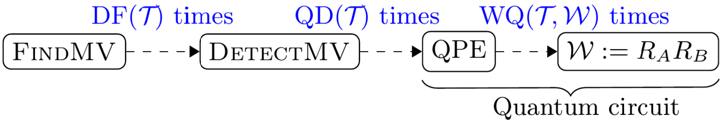
- In 2018, Montanaro introduces two quantum tree-search algorithm, DetectMV and FindMV [Mon18]
- Given a tree T and a predicate P, DetectMV returns whether $\exists x \in T$ such that $P(x) = \top$
- By performing "depth-first decision", $\mathit{DetectMV} \mapsto \mathit{FindMV}$, which returns x

Quantum tree search

- In 2018, Montanaro introduces two quantum tree-search algorithm, DetectMV and FindMV [Mon18]
- Given a tree T and a predicate P, DetectMV returns whether $\exists x \in T$ such that $P(x) = \top$
- By performing "depth-first decision", *DetectMV* $\mapsto$ *FindMV*, which returns x
- Classical worst-case runtime $O(\#T) \mapsto$ quantum worst case $O(\sqrt{\#T \cdot n})$, n the height of T



- DetectMV consists of repeating multiple quantum phase estimations of an operator W
- Under conservative assumptions, we evaluate $\sqrt{\#T \cdot n}$ times W



- DetectMV consists of repeating multiple quantum phase estimations of an operator W
- Under conservative assumptions, we evaluate $\sqrt{\#T \cdot n}$ times W

Does quantum enumeration fit within max-depth?

- For the sake of thought experiment, let's choose $\text{Depth}(W) = \text{Gates}(W) = 1$
- Using lower bounds for the cost of enumeration [ANSS18], we pick a block size β for using BKZ against Kyber

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } T}}[\text{Depth}(\mathit{FindMV})] \approx \mathbb{E}[\sqrt{\#T \cdot \beta}] \approx \sqrt{\mathbb{E}[\#T] \cdot \beta} \approx \begin{cases} 2^{90.3} & \text{for Kyber-512,} \\ 2^{166.2} & \text{for Kyber-768,} \\ 2^{263.7} & \text{for Kyber-1024,} \end{cases}$$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } T}}[\mathrm{Depth}(\mathit{FindMV})] \approx \mathbb{E}[\sqrt{\#T \cdot \beta}] \approx \sqrt{\mathbb{E}[\#T] \cdot \beta} \approx \begin{cases} 2^{90.3} & \text{for Kyber-512,} \\ 2^{166.2} & \text{for Kyber-768,} \\ 2^{263.7} & \text{for Kyber-1024,} \end{cases}$$


APTN / AP

- Wait, don't drag me down the podium

$$\underset{\substack{\text{random}\\\text{tree } T}}{\mathbb{E}}[\text{Depth}(\mathit{FindMV})] \approx \mathbb{E}[\sqrt{\#T\cdot\beta}] \approx \sqrt{\mathbb{E}[\#T]\cdot\beta} \approx \begin{cases} 2^{90.3} & \text{for Kyber-512,}\\ 2^{166.2} & \text{for Kyber-768,}\\ 2^{263.7} & \text{for Kyber-1024,}\end{cases}$$


APTN / AP

- Wait, don't drag me down the podium
- I do know Jensen's inequality!

$$\mathbb{E}[\sqrt{\#T}] \leq \sqrt{\mathbb{E}[\#T]}$$

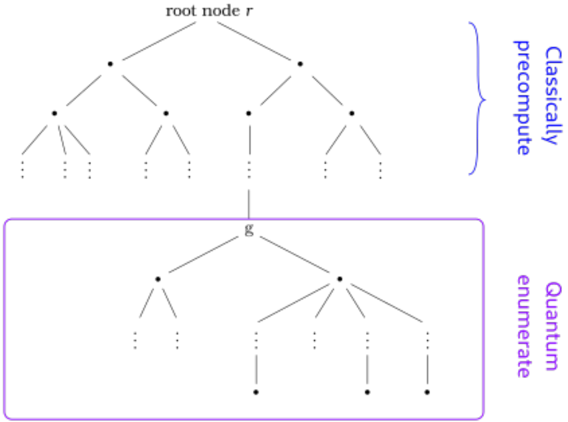
- Just wait a handful of slides

- We plausibly don't fit within $MD = 2^{96}$
- We need find ourselves smaller trees

- We plausibly don't fit within $MD = 2^{96}$
- We need find ourselves smaller trees

Classic trick from parallel enumeration

Precompute nodes up to level $k > 1$, run FindMV on the subtrees



Would this work?

Would this work?

- $k \approx 1$: in this case most of the tree fits within the quantum enumeration subroutine $\rightarrow$ a quadratic speedup without pre-computation, but maybe not our case

Would this work?

- $k \approx 1$: in this case most of the tree fits within the quantum enumeration subroutine $\rightarrow$ a quadratic speedup without pre-computation, but maybe not our case
- $k \approx n/2$: we run $\approx H_{n/2}$ quantum enumeration calls

Would this work?

- $k \approx 1$: in this case most of the tree fits within the quantum enumeration subroutine $\rightarrow$ a quadratic speedup without pre-computation, but maybe not our case

- $k \approx n/2$: we run $\approx H_{n/2}$ quantum enumeration calls
  $\implies$ total gate-count $\approx H_{n/2} \approx$ cost of classical enumeration

Would this work?

- $k \approx 1$: in this case most of the tree fits within the quantum enumeration subroutine $\rightarrow$ a quadratic speedup without pre-computation, but maybe not our case
- $k \approx n/2$: we run $\approx H_{n/2}$ quantum enumeration calls
  $\Longrightarrow$ total gate-count $\approx H_{n/2} \approx$ cost of classical enumeration
- $k \approx n$: we run some quantum enumeration, we precomputed more than $H_{n/2}$ classically, no advantage over classical enumeration

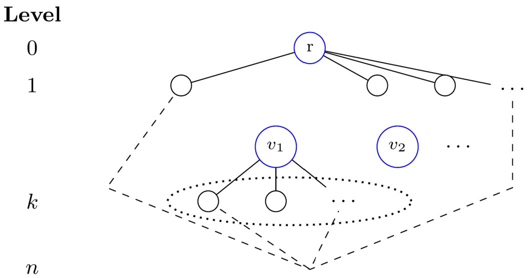
Running FindMV for every element in H_k may be too much: try bundling!

Running FindMV for every element in H_k may be too much: try bundling!

- Assume 2^y qRAM available

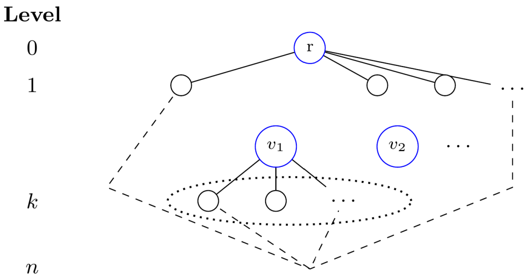
Running FindMV for every element in H_k may be too much: try bundling!

- Assume 2^y qRAM available
- Precompute sets of 2^y elements in H_k, collect them under a 'virtual' node v, run FindMV over the tree $T(v)$ with root v



Running FindMV for every element in H_k may be too much: try bundling!

- Assume 2^y qRAM available
- Precompute sets of 2^y elements in H_k, collect them under a 'virtual' node v, run FindMV over the tree $T(v)$ with root v



Disclaimer

qRAM (a.k.a. QRACM) may be extremely costly to access [JR23]. Many (most?) quantum-classical speedups assume it.

One last step

- Remember $\mathbb{E}[\sqrt{\#T}] \leq \sqrt{\mathbb{E}[\#T]}$?

One last step

- Remember $\mathbb{E}[\sqrt{\#T}] \leq \sqrt{\mathbb{E}[\#T]}$?
- We want to argue this quantum enumeration won't work, we need lower bounds, not upper bounds!

One last step

- Remember $\mathbb{E}[\sqrt{\#T}] \leq \sqrt{\mathbb{E}[\#T]}$?
- We want to argue this quantum enumeration won't work, we need lower bounds, not upper bounds!

Definition: Multiplicative Jensen's gap

Let X be a random variable. We say X has multiplicative Jensen's gap 2^z if

$$\sqrt{\mathbb{E}[X]} = 2^z\, \mathbb{E}[\sqrt{X}].$$

One last step

- Remember $\mathbb{E}[\sqrt{\# T}] \leq \sqrt{\mathbb{E}[\# T]}$?
- We want to argue this quantum enumeration won't work, we need lower bounds, not upper bounds!

Definition: Multiplicative Jensen's gap

Let X be a random variable. We say X has multiplicative Jensen's gap 2^z if

$$\sqrt{\mathbb{E}[X]} = 2^z \, \mathbb{E}[\sqrt{X}].$$

Let's find some lower bounds! . . . as a function of z

Classical pre-computation cost – well understood

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[\text{Classical Gates}] \approx \frac{1}{2}\sum_{i=1}^{k} H_i \approx \max_{1\leq \ell \leq k} H_\ell$$

Classical pre-computation cost – well understood

$$\underset{\substack{\text{random}\\ \text{tree } T}}{\mathbb{E}}[\text{Classical Gates}] \approx \frac{1}{2}\sum_{i=1}^{k} H_i \approx \max_{1\leq \ell \leq k} H_\ell$$

Quantum gate-cost

$$\begin{aligned}\underset{\substack{\text{random}\\ \text{tree } T}}{\mathbb{E}}[\text{Quantum Gates}] &\approx \frac{H_k}{2^y} \cdot \mathbb{E}\left[\text{Gates}(\text{FindMV}(T(g)))\right] \\ &\geq \frac{H_k}{2^y} \cdot \mathbb{E}\left[\sqrt{\#T(v)\cdot(n-k+1)}\right] \cdot \text{Gates}(W) \\ &= \frac{H_k}{2^y} \cdot \frac{1}{2^z}\sqrt{\mathbb{E}\left[\#T(v)\cdot(n-k+1)\right] \cdot \text{Gates}(W)}\end{aligned}$$

Classical pre-computation cost – well understood

$$\underset{\substack{\text{random}\\\text{tree } T}}{\mathbb{E}}[\text{Classical Gates}] \approx \frac{1}{2}\sum_{i=1}^{k} H_i \approx \max_{1\leq\ell\leq k} H_\ell$$

Quantum gate-cost

$$\begin{aligned}\underset{\substack{\text{random}\\\text{tree } T}}{\mathbb{E}}[\text{Quantum Gates}] &\approx \frac{H_k}{2^y}\cdot\mathbb{E}\,[\text{Gates}(\text{FindMV}(T(g)))]\\ &\geq \frac{H_k}{2^y}\cdot\mathbb{E}\left[\sqrt{\#T(v)\cdot(n-k+1)}\right]\cdot\text{Gates}(W)\\ &= \frac{H_k}{2^y}\cdot\frac{1}{2^z}\sqrt{\mathbb{E}\,[\#T(v)\cdot(n-k+1)]}\cdot\text{Gates}(W)\end{aligned}$$

Quantum depth

$$\mathbb{E}\,[\text{Depth}(\text{QPE}(W))] \geq \frac{1}{2^z}\sqrt{\mathbb{E}\,[\#T(v)\cdot(n-k+1)]}\cdot\text{Depth}(W).$$

We can now try computing some numbers.

We can now try computing some numbers.

- We assume both $\text{Depth}(W) = \text{Gates}(W) = 1$ ("query-model") and a lower bound based on best-known quantum arithmetic circuits ("circuit-model", recent work may help [BvHJ+23])

We can now try computing some numbers.

- We assume both $\text{Depth}(W) = \text{Gates}(W) = 1$ ("query-model") and a lower bound based on best-known quantum arithmetic circuits ("circuit-model", recent work may help [BvHJ+23])
- We use the LWE-estimator to find the enumeration dimension β

We can now try computing some numbers.

- We assume both $\text{Depth}(W) = \text{Gates}(W) = 1$ ("query-model") and a lower bound based on best-known quantum arithmetic circuits ("circuit-model", recent work may help [BvHJ+23])
- We use the LWE-estimator to find the enumeration dimension β
- We estimate sub-tree sizes using cylinder pruning lower-bounds [ANSS18]

We can now try computing some numbers.

- We assume both $\text{Depth}(W) = \text{Gates}(W) = 1$ ("query-model") and a lower bound based on best-known quantum arithmetic circuits ("circuit-model", recent work may help [BvHJ+23])
- We use the LWE-estimator to find the enumeration dimension β
- We estimate sub-tree sizes using cylinder pruning lower-bounds [ANSS18]
- We estimate costs for every $k \leq n$, $y \leq 80$, $z \leq 64$

We can now try computing some numbers.

- We assume both $\mathrm{Depth}(W) = \mathrm{Gates}(W) = 1$ (“query-model”) and a lower bound based on best-known quantum arithmetic circuits (“circuit-model”, recent work may help [BvHJ+23])
- We use the LWE-estimator to find the enumeration dimension β
- We estimate sub-tree sizes using cylinder pruning lower-bounds [ANSS18]
- We estimate costs for every $k \leq n$, $y \leq 80$, $z \leq 64$
- We report z, k minimising classical + quantum gate-cost

more likely to be feasible — less likely to be feasible

MD	Kyber	$\log \mathbb{E}[\text{GCost}]$ (with $\mathcal{W}$ as in § 4.1) below... Target security	Grover on $\text{AES}_{\{128,192,256\}}$	Quasi-Sqrt $^1/_b\sqrt{\#\mathcal{T}\cdot h}$	$\log \mathbb{E}[\text{GCost}]$ (with $\mathcal{W}$ as in § 4.2) below... Target security	Grover on $\text{AES}_{\{128,192,256\}}$	Quasi-Sqrt $^1/_b\sqrt{\#\mathcal{T}\cdot h}$
2^{40}	-512	$z \geq 7,\ k \leq 92$	$z \geq 13,\ k \leq 83$	$z \geq 26,\ k \leq 59$	$z \geq 23,\ k \leq 96$	$z \geq 29,\ k \leq 79$	$z \geq 42,\ k \leq 63$
	-768	$z \geq 51,\ k \leq 114$	$z \geq 57,\ k \leq 106$	$z \geq 64,\ k \leq 77$	$z > 64$	$z > 64$	$z > 64$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
2^{64}	-512	$z \geq 0,\ k \leq 83$	$z \geq 13,\ k \leq 64$	$z \geq 14,\ k \leq 59$	$z \geq 11,\ k \leq 96$	$z \geq 29,\ k \leq 63$	$z \geq 30,\ k \leq 63$
	-768	$z \geq 39,\ k \leq 114$	$z \geq 57,\ k \leq 77$	$z \geq 52,\ k \leq 77$	$z \geq 55,\ k \leq 111$	$z > 64$	$z > 64$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
2^{96}	-512	$z \geq 0,\ k \leq 58$	$z \geq 8,\ k \leq 53$	$z \geq 1,\ k \leq 58$	$z \geq 0,\ k \leq 63$	$z \geq 33,\ k \leq 54$	$z \geq 25,\ k \leq 58$
	-768	$z \geq 23,\ k \leq 106$	$z \geq 56,\ k \leq 62$	$z \geq 36,\ k \leq 77$	$z \geq 40,\ k \leq 77$	$z > 64$	$z \geq 52,\ k \leq 77$
	-1024	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$	$z > 64$
∞	-512	$z \geq 0,\ k = 0$	$z \geq 9,\ k = 0$	$z \geq 1,\ k = 0$	$z \geq 0,\ k = 0$	$z \geq 33,\ k = 0$	$z \geq 26,\ k = 0$
	-768	$z \geq 0,\ k = 0$	$z \geq 52,\ k = 0$	$z \geq 1,\ k = 0$	$z \geq 1,\ k = 0$	$z > 64$	$z \geq 27,\ k = 0$
	-1024	$z \geq 9,\ k = 0$	$z > 64$	$z \geq 1,\ k = 0$	$z \geq 35,\ k = 0$	$z > 64$	$z \geq 28,\ k = 0$

- Kyber-768 and -1024 seem out of reach

- Kyber-768 and -1024 seem out of reach
- Kyber-512 within the “query-model” reach, less clear for “circuit-model”

- Kyber-768 and -1024 seem out of reach
- Kyber-512 within the “query-model” reach, less clear for “circuit-model”
  - However AES-128 also within reach of Grover key-search in some settings...
  - And we are being quite strict in various parts of the computation

- Kyber-768 and -1024 seem out of reach
- Kyber-512 within the “query-model” reach, less clear for “circuit-model”
  - However AES-128 also within reach of Grover key-search in some settings...
  - And we are being quite strict in various parts of the computation
- Hard to claim this attack obviously works

- Kyber-768 and -1024 seem out of reach
- Kyber-512 within the “query-model” reach, less clear for “circuit-model”
  - However AES-128 also within reach of Grover key-search in some settings...
  - And we are being quite strict in various parts of the computation
- Hard to claim this attack obviously works

Disclaimer

Yet, these are **opinions** without a clear understanding of the Jensen gap!

- Kyber-768 and -1024 seem out of reach
- Kyber-512 within the "query-model" reach, less clear for "circuit-model"
  - However AES-128 also within reach of Grover key-search in some settings...
  - And we are being quite strict in various parts of the computation
- Hard to claim this attack obviously works

Disclaimer

Yet, these are **opinions** without a clear understanding of the Jensen gap!

Can we say anything about it?

Reasons to hope

- The cost reduces smoothly as a funciton of z (approximate estimates may already help)

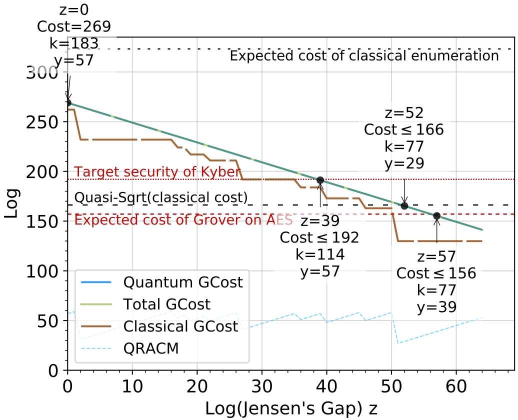


Figure: Kyber-768, $MD = 2^{64}$, unit W.

Reasons to hope

- The cost reduces smoothly as a funciton of z (approximate estimates may already help)
- Experimental evidence up $\beta = 70$ say $z \approx 1$

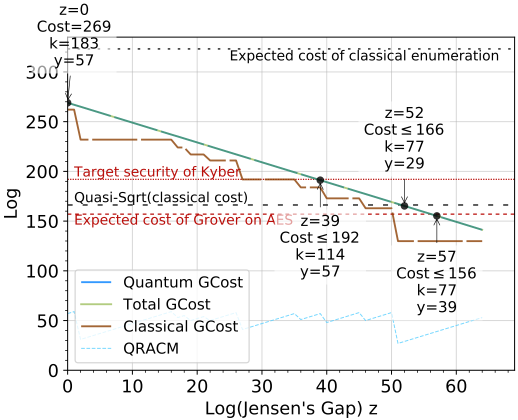


Figure: Kyber-768, $MD = 2^{64}$, unit W.

Reasons to hope

- The cost reduces smoothly as a funciton of z (approximate estimates may already help)
- Experimental evidence up $\beta = 70$ say $z \approx 1$
- Can prove lower bounds:
$$z \leq \frac{1}{2 \ln 2} \sqrt[4]{\mathbb{V}[\# T]}.$$

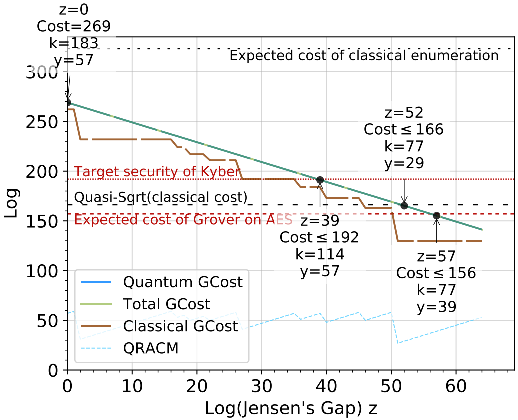


Figure: Kyber-768, $MD = 2^{64}$, unit W.

Open issues

- Not much analysis on $\mathbb{V}[\#T]$

Open issues

- Not much analysis on $\mathbb{V}[\#T]$

$$\mathop{\mathbb{E}}_{\substack{\text{random} \\ \text{tree } T}}[\#T] = \frac{1}{2}\sum_{k=1}^{n} \mathop{\mathbb{E}}_{\substack{\text{random} \\ \text{tree } T}}[|Z_k|],$$

Open issues

- Not much analysis on $\mathbb{V}[\#T]$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } T}}[\#T] = \frac{1}{2}\sum_{k=1}^{n} \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } T}}[|Z_k|]\,,$$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } T}}[\#Z_k] = \mathop{\mathbb{E}}_{\substack{\text{random}\\ \text{tree } T}}[|\mathrm{Ball}_k(\mathbf{0}, R) \cap \pi_{n-k+1}(\Lambda)|] \approx \frac{\mathrm{vol}(\mathrm{Ball}_k(\mathbf{0}, R))}{\mathrm{covol}(\pi_{n-k+1}(\Lambda))}$$

Open issues

- Not much analysis on $\mathbb{V}[\#T]$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[\#T] = \frac{1}{2}\sum_{k=1}^{n} \mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[|Z_k|],$$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[\#Z_k] = \mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[|\mathrm{Ball}_k(\mathbf{0}, R) \cap \pi_{n-k+1}(\Lambda)|] \approx \frac{\mathrm{vol}(\mathrm{Ball}_k(\mathbf{0}, R))}{\mathrm{covol}(\pi_{n-k+1}(\Lambda))}$$

$$\mathop{\mathbb{V}}_{\substack{\text{random}\\\text{tree } T}}[|\mathrm{Ball}_k(\mathbf{0}, R) \cap \pi_{n-k+1}(\Lambda)|]? \qquad \mathop{\mathbb{V}}_{\substack{\text{random}\\\text{tree } T}}[\#T]?$$

There's only some results for random real lattices [AEN]

Open issues

- Not much analysis on $\mathbb{V}[\#T]$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[\#T] = \frac{1}{2}\sum_{k=1}^{n} \mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[|Z_k|]\,,$$

$$\mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[\#Z_k] = \mathop{\mathbb{E}}_{\substack{\text{random}\\\text{tree } T}}[|\mathrm{Ball}_k(\mathbf{0}, R) \cap \pi_{n-k+1}(\Lambda)|] \approx \frac{\mathrm{vol}(\mathrm{Ball}_k(\mathbf{0}, R))}{\mathrm{covol}(\pi_{n-k+1}(\Lambda))}$$

$$\mathop{\mathbb{V}}_{\substack{\text{random}\\\text{tree } T}}[|\mathrm{Ball}_k(\mathbf{0}, R) \cap \pi_{n-k+1}(\Lambda)|]? \qquad \mathop{\mathbb{V}}_{\substack{\text{random}\\\text{tree } T}}[\#T]?$$

There's only some results for random real lattices [AEN]

We only covered cylinder pruning. Discrete pruning? Ad-hoc pruning for quantum enumeration?

Conclusions

Conclusions

- Conservative estimates are good in general

Conclusions

- Conservative estimates are good in general
- But mild limitations to quantum computers may incur in large penalties

Conclusions

- Conservative estimates are good in general
- But mild limitations to quantum computers may incur in large penalties
- It is quite difficult to tell if many proposed quantum speedups to classical algorithms actually hold

Conclusions

- Conservative estimates are good in general
- But mild limitations to quantum computers may incur in large penalties
- It is quite difficult to tell if many proposed quantum speedups to classical algorithms actually hold
- Can we do better by designing quantum attacks optimised for these limitations?

Conclusions

- Conservative estimates are good in general
- But mild limitations to quantum computers may incur in large penalties
- It is quite difficult to tell if many proposed quantum speedups to classical algorithms actually hold
- Can we do better by designing quantum attacks optimised for these limitations?

Thank you

Slides @ `https://fundamental.domains`

Martin R. Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W. Postlethwaite, and Marc Stevens.
The general sieve kernel and new records in lattice reduction.
In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 717–746. Springer, Heidelberg, May 2019.

Yoshinori Aono, Thomas Espitau, and Phong Q. Nguyen.
Random lattices: Theory and practice.
Preprint, available at https://espitau.github.io/bin/random_lattice.pdf.

Martin R. Albrecht, Vlad Gheorghiu, Eamonn W. Postlethwaite, and John M. Schanck.
Estimating quantum speedups for lattice sieves.
In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 583–613. Springer, Heidelberg, December 2020.

Miklós Ajtai, Ravi Kumar, and D. Sivakumar.
A sieve algorithm for the shortest lattice vector problem.
In *33rd ACM STOC*, pages 601–610. ACM Press, July 2001.

Yoshinori Aono, Phong Q. Nguyen, Takenobu Seito, and Junji Shikata.
Lower bounds on lattice enumeration with extreme pruning.
In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 608–637. Springer, Heidelberg, August 2018.

Nina Bindel, Xavier Bonnetain, Marcel Tiepelt, and Fernando Virdia.
Quantum lattice enumeration in limited depth.

Cryptology ePrint Archive, Paper 2023/1423, 2023.
https://eprint.iacr.org/2023/1423.

Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven.
New directions in nearest neighbor searching with applications to lattice sieving.
In Robert Krauthgamer, editor, *27th SODA*, pages 10–24. ACM-SIAM, January 2016.

Shi Bai, Maya-Iggy van Hoof, Floyd B. Johnson, Tanja Lange, and Tran Ngo.
Concrete analysis of quantum lattice enumeration.
In *Advances in Cryptology - ASIACRYPT 2023*, Lecture Notes in Computer Science. Springer-Verlag, 2023.

Craig Gidney and Martin Ekerå.
How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits.
Quantum, 5:433, 2021.

Nicolas Gama, Phong Q. Nguyen, and Oded Regev.
Lattice enumeration using extreme pruning.
In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 257–278. Springer, Heidelberg, May / June 2010.

Zhenyu Huang and Siwei Sun.
Synthesizing quantum circuits of aes with lower t-depth and less qubits.
In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 614–644. Springer, 2022.

Kyungbae Jang, Anubhab Baksi, Hyunji Kim, Gyeongju Song, Hwajeong Seo, and Anupam Chattopadhyay.

Quantum analysis of aes.
Cryptology ePrint Archive, Paper 2022/683, 2022.
https://eprint.iacr.org/2022/683.

Samuel Jaques, Michael Naehrig, Martin Roetteler, and Fernando Virdia.
Implementing grover oracles for quantum key search on AES and LowMC.
In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 280–310. Springer, Heidelberg, May 2020.

Samuel Jaques and Arthur G. Rattew.
Qram: A survey and critique, 2023.

Samuel Jaques and John M. Schanck.
Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE.
In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 32–61. Springer, Heidelberg, August 2019.

Thijs Laarhoven.
Sieving for shortest vectors in lattices using angular locality-sensitive hashing.
In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 3–22. Springer, Heidelberg, August 2015.

Ashley Montanaro.
Quantum-walk speedup of backtracking algorithms.
Theory Comput., 14(1):1–24, 2018.

National Institute of Standards and Technology.
Submission requirements and evaluation criteria for the Post-Quantum Cryptography standardization process.
http://csrc.nist.gov/groups/ST/post-quantum-crypto/documents/call-for-proposals-final-dec-2016.pdf, December 2016.

Phong Q. Nguyen and Thomas Vidick.
Sieve algorithms for the shortest vector problem are practical.
J. Math. Cryptol., 2(2):181–207, 2008.

Christof Zalka.
Grover's quantum searching algorithm is optimal.
Phys. Rev. A, 60:2746–2751, Oct 1999.

Jian Zou, Zihao Wei, Siwei Sun, Ximeng Liu, and Wenling Wu.
Quantum circuit implementations of aes with fewer qubits.
In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*, pages 697–726. Springer, 2020.