Estimating quantum speedups for lattice sieves

Martin R. Albrecht[1], Vlad Gheorghiu[2], *Eamonn W. Postlethwaite*[1], John M. Schanck[2]

[1]Information Security Group, Royal Holloway, University of London,
[2]Institute for Quantum Computing, University of Waterloo, Canada

What are we doing? We are

- trying to understand 'quantum lattice sieves' non-asymptotically,

What are we doing? We are

- trying to understand ‘quantum lattice sieves’ non-asymptotically,
- trying to glance behind the query model – e.g. no longer counting Grover oracle queries,

What are we doing? We are

- trying to understand 'quantum lattice sieves' non-asymptotically,
- trying to glance behind the query model – e.g. no longer counting Grover oracle queries,
- trying to understand the quantum overhead of these sieves, and compare to their classical variants.

How to do this? We need to

- decide exactly what a query to our oracle is,

How to do this? We need to

- decide exactly what a query to our oracle is,
- build a circuit for it,

How to do this? We need to

- decide exactly what a query to our oracle is,
- build a circuit for it,
- describe some cost metrics to optimise controllable parameters under,

How to do this? We need to

- decide exactly what a query to our oracle is,
- build a circuit for it,
- describe some cost metrics to optimise controllable parameters under,
- build some software to perform this optimisation.

Why is this interesting? (Good question) because

- a great deal of cryptography, some close to standardisation, uses lattice based assumptions,

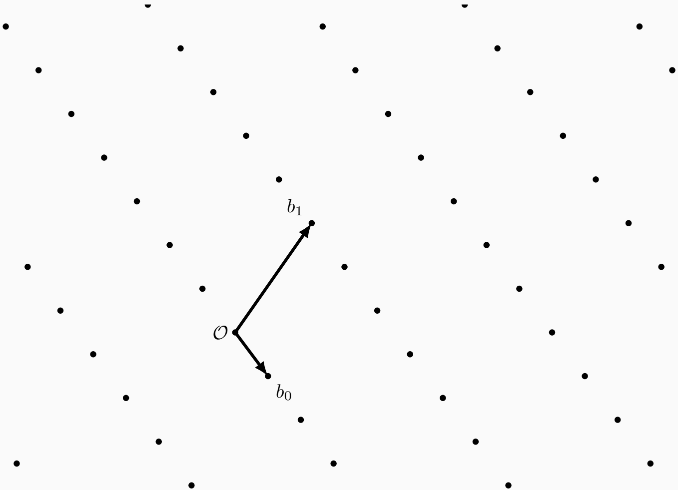
Why is this interesting? (Good question) because

- a great deal of cryptography, some close to standardisation, uses lattice based assumptions,
- classically it is lattice sieves that currently power the best cryptanalysis,

Why is this interesting? (Good question) because

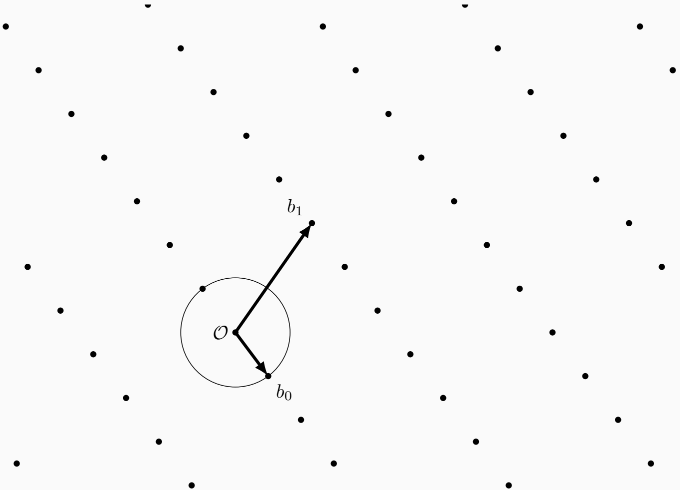
- a great deal of cryptography, some close to standardisation, uses lattice based assumptions,
- classically it is lattice sieves that currently power the best cryptanalysis,
- what if a large fault tolerant quantum computer appeared at CWI tomorrow?

What: lattices



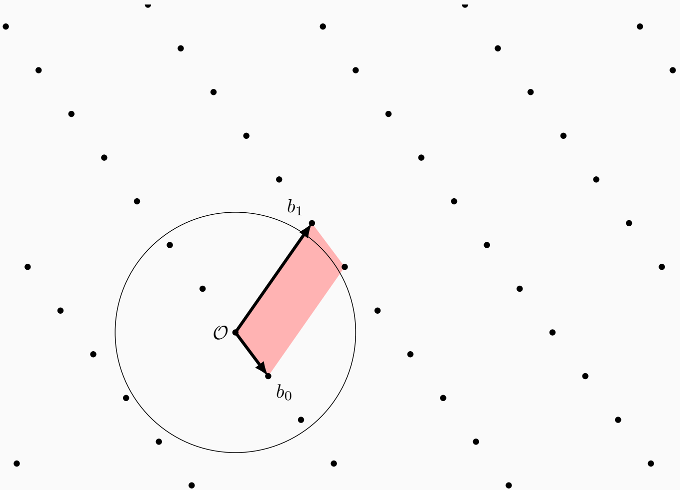
$\Lambda = \mathsf{Span}_{\mathbb{Z}}(b_0, \ldots, b_{d-1})$, $B = \{b_0, \ldots, b_{d-1}\} \subset \mathbb{R}^d$ a basis

What: lattices



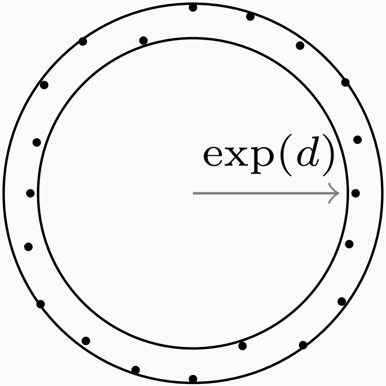
SVP: find $v \in \Lambda \setminus \{0\}$ such that $\|v\|_2 \leq \|w\|_2$ for all $w \in \Lambda \setminus \{0\}$

What: lattices



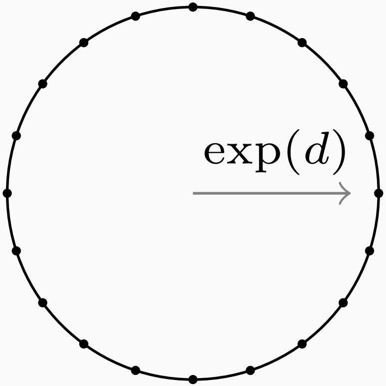
α-HSVP: find $v \in \Lambda \setminus \{0\}$ such that $\|v\|_2 \leq \alpha \cdot \mathrm{vol}(\Lambda)^{1/d}$

What: lattice sieves



We can cheaply sample a long lattice vector with uniform direction in some thin annulus.

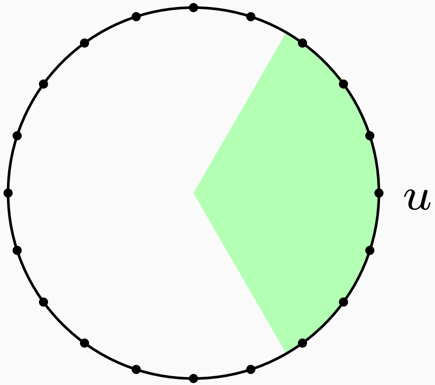
What: lattice sieves



We can cheaply sample a long lattice vector with uniform direction in some thin annulus.

Heuristic sieves are analysed as if we have many uniform vectors in S^{d-1}.

What: lattice sieves

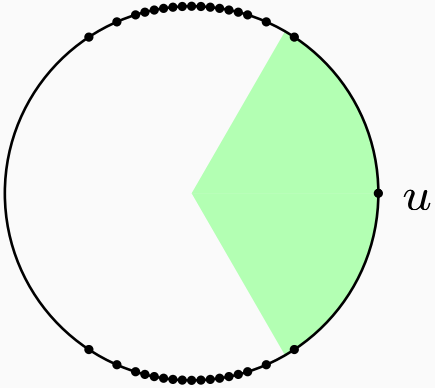


We can cheaply sample a long lattice vector with uniform direction in some thin annulus.

Heuristic sieves are analysed as if we have many uniform vectors in S^{d-1}.

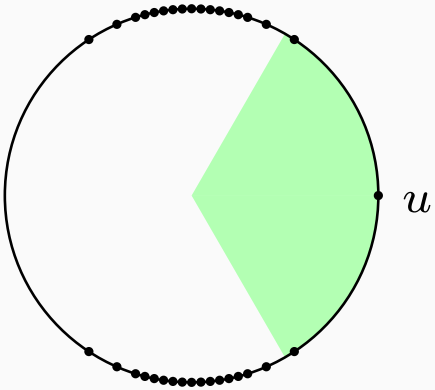
In this model $u - v$ is shorter (within the circle) iff $\theta(u, v) < \pi/3$.

What: lattice sieves (high dimensions)



As the dimension grows the distribution of $\theta(u, v)$ becomes concentrated around $\pi/2$.

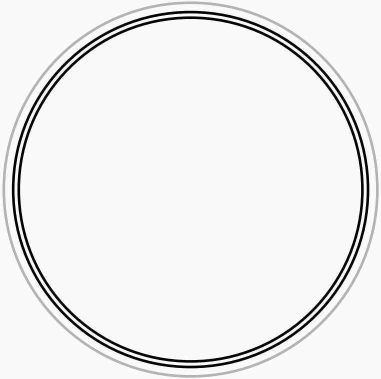
What: lattice sieves (high dimensions)



As the dimension grows the distribution of $\theta(u, v)$ becomes concentrated around $\pi/2$.

To find sufficiently many pairs with $\theta(u, v) < \pi/3$ we require $\exp(d)$ vectors; most reductions will have have $\theta(u, v) \approx \pi/3$.

What: lattice sieves (high dimensions)

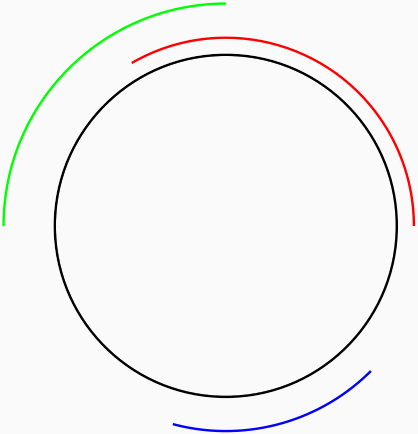


As the dimension grows the distribution of $\theta(u, v)$ becomes concentrated around $\pi/2$.

To find sufficiently many pairs with $\theta(u, v) < \pi/3$ we require $\exp(d)$ vectors; most reductions will have have $\theta(u, v) \approx \pi/3$.

We begin anew with some thin annulus of vectors an $\varepsilon \in (0, 1)$ factor shorter.

What: lattice sieves (bucketing)



Calculating $\theta(u, v)$ is effectively an inner product, the number of which we want to minimise.

Lattice sieves therefore bucket vectors in various manners and check $\theta(u, v)$ only within these buckets.

One can also filter further within buckets (spoiler: we do this).

What: different lattice sieves

Sieve (NNS subroutine)[1]	$\log_2 \text{time}_C$	$\log_2 \text{time}_Q$
NV style [NV08]	$0.415d$	$0.311d$
RandomBucket [BGJ15, ADH$^+$19]	$0.349d$	$0.301d$
ListDecoding [BDGL16, DSvW21]	$0.292d$	$0.265d$

[1] All complexities are missing $+o(d)$ terms.

What: different lattice sieves

Sieve (NNS subroutine)[1]	$\log_2 \text{time}_C$	$\log_2 \text{time}_Q$
NV style [NV08]	$0.415d$	$0.311d$
RandomBucket [BGJ15, ADH$^+$19]	$0.349d$	$0.301d$
ListDecoding [BDGL16, DSvW21]	$0.292d$	$0.265d$

The quantum variants of these sieves use Grover's search algorithm to instantiate the search for reducing pairs (within buckets, when appropriate).

All require exponential space, $2^{\Theta(d)}$.

[1] All complexities are missing $+o(d)$ terms.

How: classical and quantum search

Let $[N] = \{1, \ldots, N\}$ and $f\colon [N] \to \{0, 1\}$ be an unstructured predicate, with *roots*

$$\mathrm{Ker}(f) = \{x\colon f(x) = 0\}.$$

How: classical and quantum search

Let $[N] = \{1, \ldots, N\}$ and $f\colon [N] \to \{0,1\}$ be an unstructured predicate, with *roots*

$$\mathrm{Ker}(f) = \{x\colon f(x) = 0\}.$$

We can find a root

- classically by evaluating $f(1), \ldots, f(N)$,
- quantumly by measuring $\mathbf{G}(f)^j \mathbf{D}|0\rangle$.

How: classical and quantum search

Let $[N] = \{1, \ldots, N\}$ and $f \colon [N] \to \{0, 1\}$ be an unstructured predicate, with *roots*

$$\mathrm{Ker}(f) = \{x \colon f(x) = 0\}.$$

We can find a root

- classically by evaluating $f(1), \ldots, f(N)$,
- quantumly by measuring $\mathbf{G}(f)^j \mathbf{D}|0\rangle$.

If $|\mathrm{Ker}(f)| \in o(N)$ then, to succeed with constant probability, we expect $O(N)$ queries to f classically, and $j \in O(\sqrt{N})$ queries to $\mathbf{G}(f)$ quantumly.

How: filtered search

Classically, a potentially cheaper way is to use a filter, some predicate

$$g\colon [N] \to \{0,1\}, |\mathrm{Ker}(g) \cap \mathrm{Ker}(f)| \geq 1.$$

Then we can

How: filtered search

Classically, a potentially cheaper way is to use a filter, some predicate

$$g\colon [N] \to \{0,1\}, |\mathrm{Ker}(g) \cap \mathrm{Ker}(f)| \geq 1.$$

Then we can

$$\text{evaluate } g(1), \text{if } g(1) = 0 \text{ evaluate } f(1),$$

How: filtered search

Classically, a potentially cheaper way is to use a filter, some predicate

$$g\colon [N] \to \{0,1\}, |\mathrm{Ker}(g) \cap \mathrm{Ker}(f)| \geq 1.$$

Then we can

$$\text{evaluate } g(1), \text{if } g(1) = 0 \text{ evaluate } f(1), \dots, \text{evaluate } g(N), \text{if } g(N) = 0 \text{ evaluate } f(N).$$

How: filtered search

Classically, a potentially cheaper way is to use a filter, some predicate

$$g\colon [N] \to \{0,1\}, |\mathrm{Ker}(g) \cap \mathrm{Ker}(f)| \geq 1.$$

Then we can

$$\text{evaluate } g(1), \text{if } g(1) = 0 \text{ evaluate } f(1), \ldots, \text{evaluate } g(N), \text{if } g(N) = 0 \text{ evaluate } f(N).$$

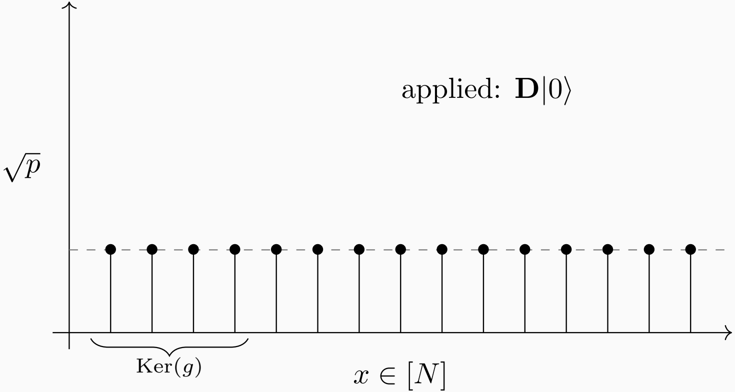
What makes a good filter? Cheaper than f to evaluate, and

$$\rho_f(g) = 1 - \frac{|\mathrm{Ker}(f) \cap \mathrm{Ker}(g)|}{|\mathrm{Ker}(g)|}, \quad \eta_f(g) = 1 - \frac{|\mathrm{Ker}(f) \cap \mathrm{Ker}(g)|}{|\mathrm{Ker}(f)|}$$

the false positive and negative rate, are both small.

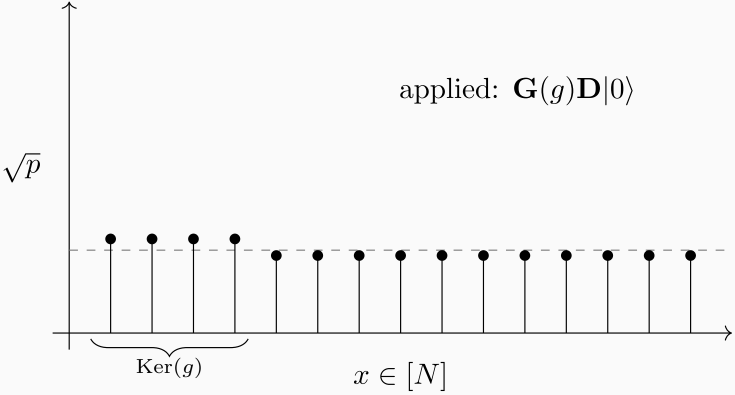
How: filtered quantum search

Branching based on g is not possible within Grover's algorithm. However we can use *amplitude amplification* to achieve something conceptually similar. First, Grover's algorithm over g:



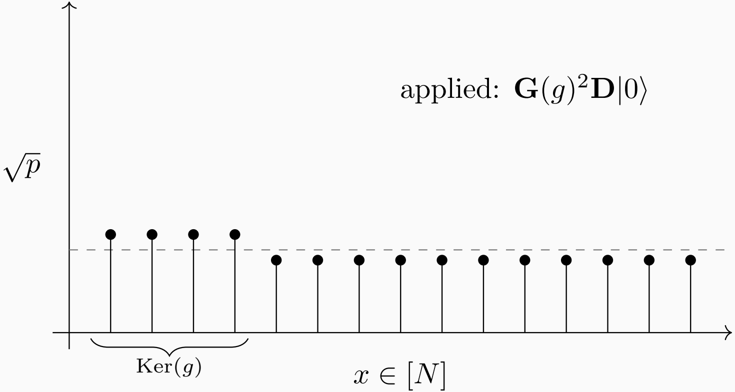
How: filtered quantum search

Branching based on g is not possible within Grover's algorithm. However we can use *amplitude amplification* to achieve something conceptually similar. First, Grover's algorithm over g:



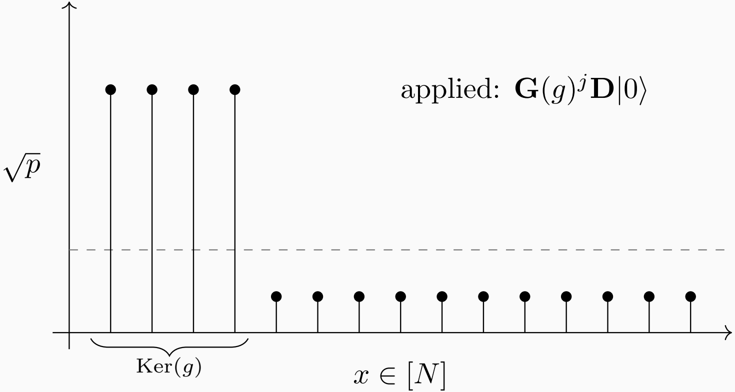
How: filtered quantum search

Branching based on g is not possible within Grover's algorithm. However we can use *amplitude amplification* to achieve something conceptually similar. First, Grover's algorithm over g:



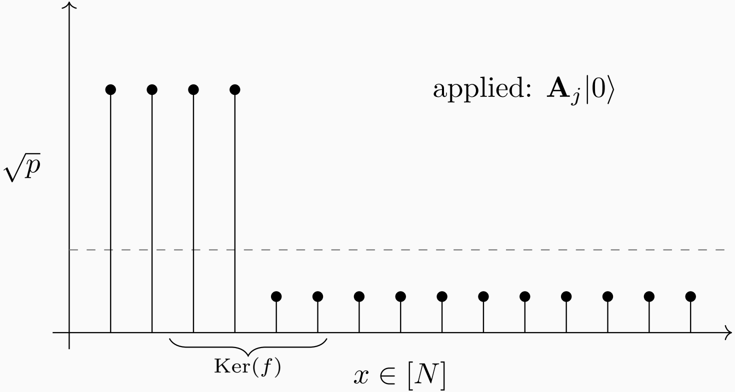
How: filtered quantum search

Branching based on g is not possible within Grover's algorithm. However we can use *amplitude amplification* to achieve something conceptually similar. First, Grover's algorithm over g:



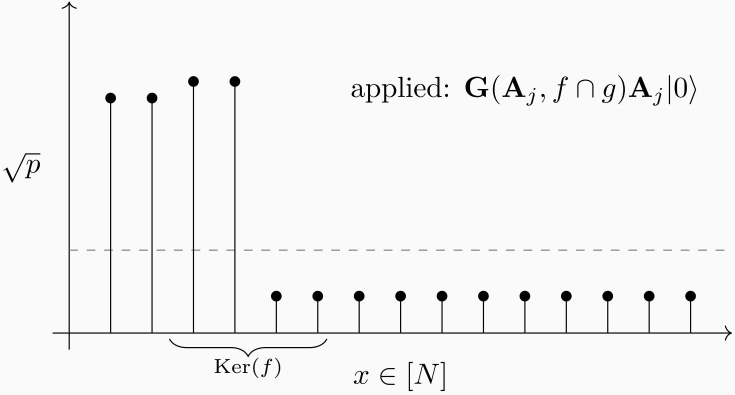
How: filtered quantum search (amplitude amplification)

Amplitude amplification can replace $\mathbf{D}$ with $\mathbf{A}_j = \mathbf{G}(g)^j\mathbf{D}$. Then amplitude amplification for the predicate $f \cap g$:



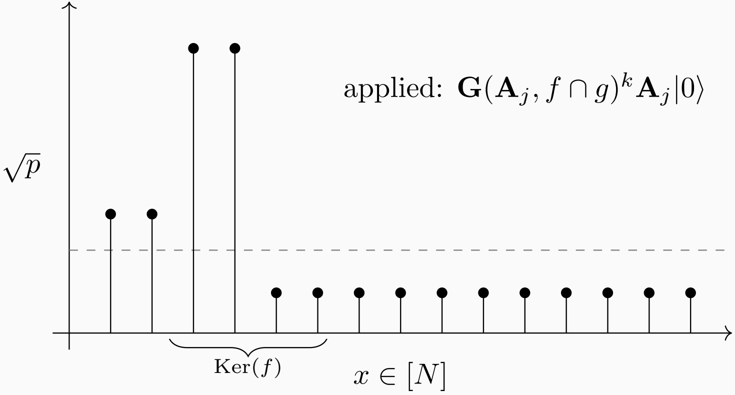
How: filtered quantum search (amplitude amplification)

Amplitude amplification can replace $\mathbf{D}$ with $\mathbf{A}_j = \mathbf{G}(g)^j\mathbf{D}$. Then amplitude amplification for the predicate $f \cap g$:



How: filtered quantum search (amplitude amplification)

Amplitude amplification can replace $\mathbf{D}$ with $\mathbf{A}_j = \mathbf{G}(g)^j\mathbf{D}$. Then amplitude amplification for the predicate $f \cap g$:



How: filtered quantum search (with partial information)

We give some technical results that (roughly) say, let

How: filtered quantum search (with partial information)

We give some technical results that (roughly) say, let

- g be a filter for predicate $f : [N] \to \{0, 1\}$,
- $P, Q, \gamma \in \mathbb{R}$ such that
  - $P/\gamma \leq |\mathrm{Ker}(g)| \leq \gamma P$, and
  - $1 \leq Q \leq |\mathrm{Ker}(f) \cap \mathrm{Ker}(g)|$.

How: filtered quantum search (with partial information)

We give some technical results that (roughly) say, let

- g be a filter for predicate $f : [N] \to \{0, 1\}$,
- $P, Q, \gamma \in \mathbb{R}$ such that
    - $P/\gamma \leq |\mathrm{Ker}(g)| \leq \gamma P$, and
    - $1 \leq Q \leq |\mathrm{Ker}(f) \cap \mathrm{Ker}(g)|$.

Then we can find a root of f with constant probability and a cost dominated by $\frac{\gamma}{2}\sqrt{N/Q}$ calls to $\mathbf{G}(g)$.

How: filtered quantum search (with partial information)

We give some technical results that (roughly) say, let

- g be a filter for predicate $f : [N] \to \{0, 1\}$,
- $P, Q, \gamma \in \mathbb{R}$ such that
  - $P/\gamma \leq |\mathrm{Ker}(g)| \leq \gamma P$, and
  - $1 \leq Q \leq |\mathrm{Ker}(f) \cap \mathrm{Ker}(g)|$.

Then we can find a root of f with constant probability and a cost dominated by $\frac{\gamma}{2}\sqrt{N/Q}$ calls to $\mathbf{G}(g)$.

The idea: the cost of a Grover query encoding the filter, $\mathbf{G}(g)$, is the crucial quantity.

$$\Rightarrow \text{specify } g, \text{ design } \mathbf{G}(g), \text{ and understand } (P, Q, \gamma).$$

How: `popcount` is our filter

For lattice vectors $u, v_1, \dots, v_N$, the reduction predicate of u is

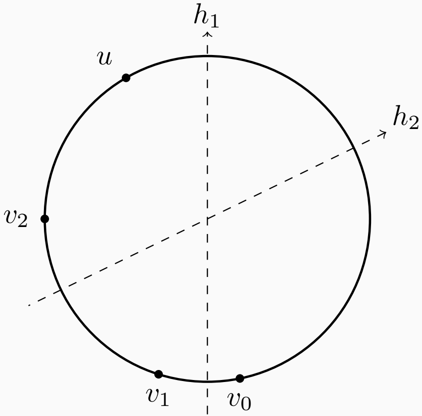
$$f_u \colon \{v_1, \dots, v_N\} \to \{0, 1\},\ f_u(v_i) = 0 \iff \langle u, v_i \rangle > \cos(\pi/3).$$

How: `popcount` is our filter

For lattice vectors $u, v_1, \ldots, v_N$, the reduction predicate of u is

$$f_u \colon \{v_1, \ldots, v_N\} \to \{0, 1\},\ f_u(v_i) = 0 \iff \langle u, v_i \rangle > \cos(\pi/3).$$

For the filter g we use 'XOR and popcount' [FBB+14], i.e. $g_u(\,\cdot\,) = \texttt{popcount}_{k,n}(u, \,\cdot\,)$.



$$\texttt{popcount}_{1,2}(u, v_0) = 1$$

$$\texttt{popcount}_{1,2}(u, v_1) = 0$$

$$\texttt{popcount}_{1,2}(u, v_2) = 0$$

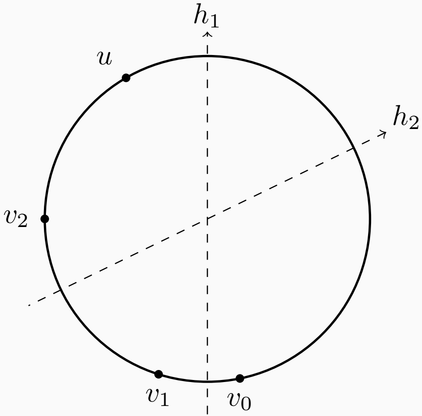
$$(k, n) = (1, 2)$$

How: `popcount` is our filter

For lattice vectors $u, v_1, \ldots, v_N$, the reduction predicate of u is

$$f_u \colon \{v_1, \ldots, v_N\} \to \{0, 1\}, \; f_u(v_i) = 0 \iff \langle u, v_i \rangle > \cos(\pi/3).$$

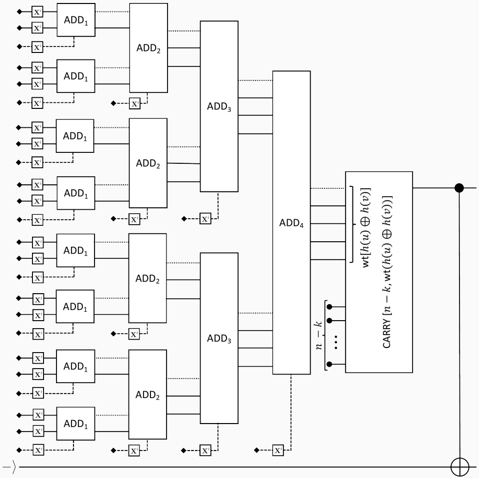
For the filter g we use 'XOR and popcount' [FBB+14], i.e. $g_u(\,\cdot\,) = \texttt{popcount}_{k,n}(u, \cdot\,)$.



$$h_i : \mathbb{R}^d \to \{0, 1\}$$

$$\text{check, over } \mathbb{Z} : \; \sum_i h_i(u) \oplus h_i(v_j) \le k$$

How: circuits for $\mathbf{G}(\mathtt{popcount}_{k,n})$



Basically a (reversible) tree of in place quantum adders ending with a comparison.

How: a probabilistic study of `popcount`

Given i.i.d. uniform $\{h_i\}_{i=1}^{n}$, some threshold k, and pair (u, v) on S^{d-1}, let $P_{k,n}(u, v)$ be the probability the pair pass $\texttt{popcount}_{k,n}$. Then

How: a probabilistic study of `popcount`

Given i.i.d. uniform $\{h_i\}_{i=1}^{n}$, some threshold k, and pair (u, v) on S^{d-1}, let $P_{k,n}(u, v)$ be the probability the pair pass $\texttt{popcount}_{k,n}$. Then

$$\Pr[P_{k,n}(u, v)] = \sum_{i=0}^{k} \binom{n}{i} \cdot \left(\frac{\theta(u, v)}{\pi}\right)^{i} \cdot \left(1 - \frac{\theta(u, v)}{\pi}\right)^{n-i}.$$

How: a probabilistic study of `popcount`

Given i.i.d. uniform $\{h_i\}_{i=1}^{n}$, some threshold k, and pair (u, v) on S^{d-1}, let $P_{k,n}(u, v)$ be the probability the pair pass $\texttt{popcount}_{k,n}$. Then

$$\Pr[P_{k,n}(u,v)] = \sum_{i=0}^{k} \binom{n}{i} \cdot \left(\frac{\theta(u,v)}{\pi}\right)^{i} \cdot \left(1 - \frac{\theta(u,v)}{\pi}\right)^{n-i}.$$

Ultimately it is $\theta = \theta(u, v)$ that matters, so we consider $P_{k,n}(\theta)$.

How: a simple example

The pdf of two uniform $u, v \in S^{d-1}$ having $\theta(u, v) = \theta$ is

$$A_d(\theta) = C(d) \cdot \sin^{d-2}(\theta),$$

and the probability of u, v passing $\texttt{popcount}_{k,n}$ is then given by

$$\int_0^\pi P_{k,n}(\theta) \cdot A_d(\theta)\, \mathrm{d}\theta.$$

How: a simple example

The pdf of two uniform $u, v \in S^{d-1}$ having $\theta(u, v) = \theta$ is

$$A_d(\theta) = C(d) \cdot \sin^{d-2}(\theta),$$

and the probability of u, v passing $\texttt{popcount}_{k,n}$ is then given by

$$\int_0^\pi P_{k,n}(\theta) \cdot A_d(\theta) \, \mathrm{d}\theta.$$

For the false negative rate and the different bucketing strategies we integrate $P_{k,n}(\theta)$ over the relevant spherical sections.

How: cost metrics

Following [JS19] we measure the cost of running a quantum circuit in terms of the classical control required to run it. Here (G, D, W) are (gate count, depth, width) of a quantum circuit.

How: cost metrics

Following [JS19] we measure the cost of running a quantum circuit in terms of the classical control required to run it. Here (G, D, W) are (gate count, depth, width) of a quantum circuit.

- gates: quantum gates cost $\Theta(1) \overset{\text{total}}{\Longrightarrow} \Theta(G)$,

How: cost metrics

Following [JS19] we measure the cost of running a quantum circuit in terms of the classical control required to run it. Here (G, D, W) are (gate count, depth, width) of a quantum circuit.

- gates: quantum gates cost $\Theta(1) \xRightarrow{\text{total}} \Theta(G)$,
- depth-width: {quantum gates, identity wires} cost $\Theta(1) \xRightarrow{\text{total}} \Theta(DW)$,
- error: {quantum gates, identity wires} cost $\Theta(\log^2(DW)) \xRightarrow{\text{total}} \Omega(DW \log^2(DW))$.

How: cost metrics

Following [JS19] we measure the cost of running a quantum circuit in terms of the classical control required to run it. Here (G, D, W) are (gate count, depth, width) of a quantum circuit.

- gates: quantum gates cost $\Theta(1) \xRightarrow{\text{total}} \Theta(G)$,
- depth-width: $\{$quantum gates, identity wires$\}$ cost $\Theta(1) \xRightarrow{\text{total}} \Theta(DW)$,
- error: $\{$quantum gates, identity wires$\}$ cost $\Theta(\log^2(DW)) \xRightarrow{\text{total}} \Omega(DW \log^2(DW))$.

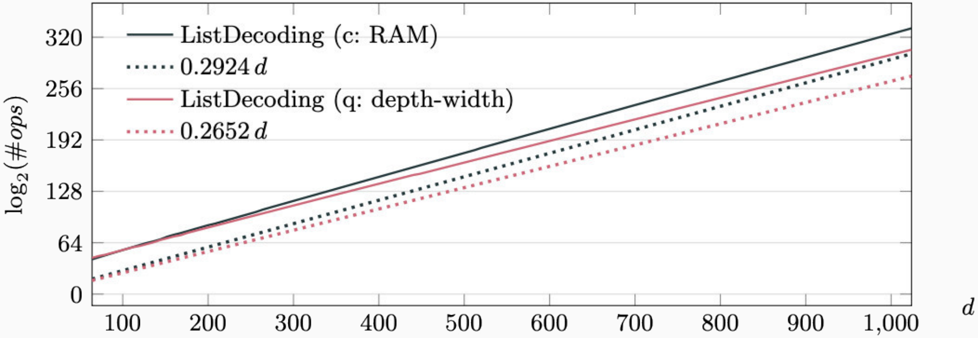
In particular we use the error correction model of Gidney–Ekerå [GE19] and the Clifford+T gate set. We compliment it with a *unit cost* qRAM lookup operation.

How: bringing it all together

So in toto

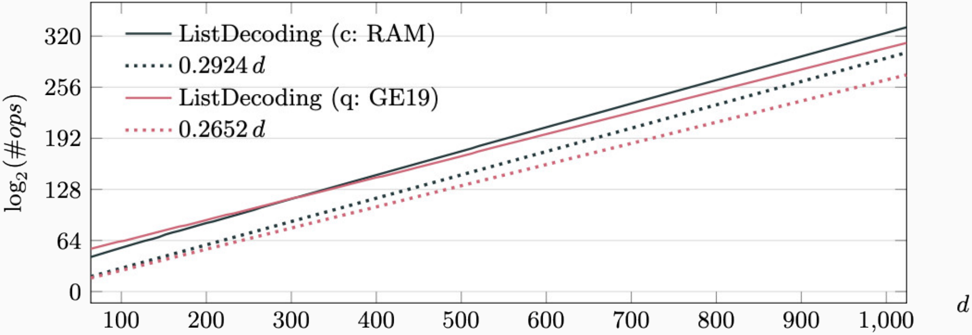
- pick your lattice sieve,
- determine its operation in terms of $(k, n), d$ and internal sieve parameters,
- determine the quantum circuit for amplitude amplification,
- pick your cost metric for quantum computation,
- minimise the cost under chosen metric in terms of (k, n) and internal sieve parameters...

Estimates: ListDecoding depth-width



ListDecodingSearch. Comparing c: (RAM) with q: (depth-width), and the leading terms of the asymptotic complexities.

Estimates: ListDecoding Gidney–Ekerå error correction



ListDecodingSearch. Comparing c: (RAM) with q: (GE19), and the leading terms of the asymptotic complexities.

Discussion I

Our estimates suggest less advantage for this quantum sieve than the asymptotic $2^{(0.292-0.265)d+o(d)}$, without entirely ruling out their relevance.

Discussion I

Our estimates suggest less advantage for this quantum sieve than the asymptotic $2^{(0.292-0.265)d+o(d)}$, without entirely ruling out their relevance.

Quantum Metric	d	$\log_2$ time$_C$	$\log_2$ time$_Q$	asym	$\log_2$ memory
Gidney–Ekerå	312	119	119	8	78
Gidney–Ekerå	352	130	128	10	87
Gidney–Ekerå	824	270	256	22	187
Depth-Width	544	189	176	15	128
Gidney–Ekerå	544	189	182	15	128

All classical costs are in a simple RAM model, the above table is for ListDecoding.

Discussion II

Our analyses do not account for the cost of qRAM and RAM, required in $\mathbf{G}(g)$ and g respectively, to which we assign unit cost. Neither has unit cost in practice, but qRAM is expected to have a much higher cost.

We also do not capture the natural clock speed error correction implies: after each layer of quantum circuit depth non-trivial classical processing must occur.

Finally, we do not apply depth constraints, the impact of which on quantum search is more than classical search, which can be trivially parallelised.

NNS $\leftrightarrow$ SVP?

The NNS search routine we cost is *not* the same as SVP.

NNS $\leftrightarrow$ SVP?

The NNS search routine we cost is *not* the same as SVP.

Under?

- the NNS subroutine is iterated $\mathrm{poly}(d)$ times,
- other subroutines, e.g. bucketing or lattice sampling, are not accounted for.

NNS $\leftrightarrow$ SVP?

The NNS search routine we cost is *not* the same as SVP.

Under?

- the NNS subroutine is iterated poly(d) times,
- other subroutines, e.g. bucketing or lattice sampling, are not accounted for.

Over?

- using ‘dimensions for free’ techniques [Duc18], NNS in dimension d solves SVP in dimension $d' > d$,
- many heuristic tricks [DSvW21, ADH$^+$19, FBB$^+$14] are not captured.

Thanks

All data and our software can be found at

`https://github.com/jschanck/eprint-2019-1161`

The paper can be found at

`https://eprint.iacr.org/2019/1161`

References i

- 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*, EUROCRYPT, 2019.
- Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven, *New directions in nearest neighbor searching with applications to lattice sieving*, Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, 2016.
- Anja Becker, Nicolas Gama, and Antoine Joux, *Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest neighbor search*, Cryptology ePrint Archive, Report 2015/522, 2015, `https://eprint.iacr.org/2015/522`.
- Léo Ducas, Marc Stevens, and Wessel van Woerden, *Advanced lattice sieving on gpus, with tensor cores*, Cryptology ePrint Archive, Report 2021/141, 2021, `https://eprint.iacr.org/2021/141`.
- Léo Ducas, *Shortest vector from lattice sieving: A few dimensions for free*, EUROCRYPT, 2018.
- Robert Fitzpatrick, Christian Bischof, Johannes Buchmann, Özgür Dagdelen, Florian Göpfert, Artur Mariano, and Bo-Yin Yang, *Tuning gausssieve for speed*, LATINCRYPT, 2014.

References ii

- Craig Gidney and Martin Ekerå, *How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits*, 2019.
- Samuel Jaques and John M. Schanck, *Quantum cryptanalysis in the ram model: Claw-finding attacks on sike*, CRYPTO, 2019.
- Phong Q. Nguyen and Thomas Vidick, *Sieve algorithms for the shortest vector problem are practical*, Journal of Mathematical Cryptology **2** (2008), no. 2, 181–207.