Karlsruhe Institute of Technology

Quantum LLL

with an Application to Mersenne Number Cryptosystems

Marcel Tiepelt[1] Alan Szepieniec[2]

[1]Karlsruhe Institute of Technology
[2]Nervos Foundation

Latincrypt 2019
Santiago de Chile, Oct. 2-4

Overview

- Quantum circuit representation of LLL
  - for (textbook) rational numbers
  - for floating-point approximation
- Resource estimates of (sub)circuits, in Toffoli-gates
- Focus on qubits count

Why quantum translation of LLL?

Consider LLL as a subroutine, e.g., SVP oracle in cryptanalysis

- Assume 256 bits of classical security, for $O(2^{256})$ expected number of oracle calls

Why quantum translation of LLL?

Consider LLL as a subroutine, e.g., SVP oracle in cryptanalysis

- Assume 256 bits of classical security, for $O(2^{256})$ expected number of oracle calls

- Quantumly: 128 bits of security, Groverization promises improvement to $O(2^{128})$
  - $\rightarrow$ Requires efficient translation of LLL into quantum setting!

Why quantum translation of LLL?

Consider LLL as a subroutine, e.g., SVP oracle in cryptanalysis

- Assume 256 bits of classical security, for $O(2^{256})$ expected number of oracle calls

- Quantumly: 128 bits of security, Groverization promises improvement to $O(2^{128})$
  $\rightarrow$ Requires efficient translation of LLL into quantum setting!

- **But**: translation of (text-book) LLL results in large overhead w.r.t. the number of qubits!

Why quantum translation of LLL?

Consider LLL as a subroutine, e.g., SVP oracle in cryptanalysis

- Assume 256 bits of classical security, for $O(2^{256})$ expected number of oracle calls
- Quantumly: 128 bits of security, Groverization promises improvement to $O(2^{128})$
  - $\rightarrow$ Requires efficient translation of LLL into quantum setting!
- **But**: translation of (text-book) LLL results in large overhead w.r.t. the number of qubits!

Does Grover with a QLLL give us the desired improvement?

(Classical) LLL

Input: Basis $B = (b_1, b_2, ..., b_r)$
Output: Reduced Basis $\hat{B}$
$B^*, M \leftarrow$ GSO(B)
$k \leftarrow 2$
while $k \leq r$ **do**
    Size-reduce(b_k, b_{k-1})
    if Lovász condition holds on b_k, b_{k-1} **then**
        Size-reduce($b_k, \{b_j\}_{0 \leq j \leq k-1}$), update M
        $k{+}{+}$
    else
        swap b_k, b_{k-1}, update M
        $k := max(2, k-1)$
    end if
end while

Variants

- Rational M: **Lenstra1982**
- Floating-point approximation M: **Schnorr:1988:MEA:48880.48883**

  "Best" variant: L^2 **10.1007/11426639˙13**

  (many more)

Quantum LLL Setup

Registers

$|B\rangle$ Basis representing a superposition of integer lattices

$|M^{(i)}\rangle$ transformation M in iteration i s.t.: $B = MB^*$

$|K\rangle, |cntl\rangle$ counters, controls

Quantum LLL Setup

Karlsruhe Institute of Technology

Registers

$|B\rangle$ Basis representing a superposition of integer lattices

$|M^{(i)}\rangle$ transformation M in iteration i s.t.: $B = MB^*$

$|K\rangle, |cntl\rangle$ counters, controls

Operations

Arithmetic in $\mathbb{Q}$ or $\mathbb{R}$, *vector operations* in $\mathbb{Z}$

misc compare, round, $\max(x, y)$, ...

Quantum LLL Setup

Registers

$\lvert B\rangle$	Basis representing a superposition of integer lattices
$\lvert M^{(i)}\rangle$	transformation M in iteration i s.t.: $B = MB^*$
$\lvert K\rangle, \lvert cntl\rangle$	counters, controls

Operations

Arithmetic in $\mathbb{Q}$ or $\mathbb{R}$, *vector operations* in $\mathbb{Z}$

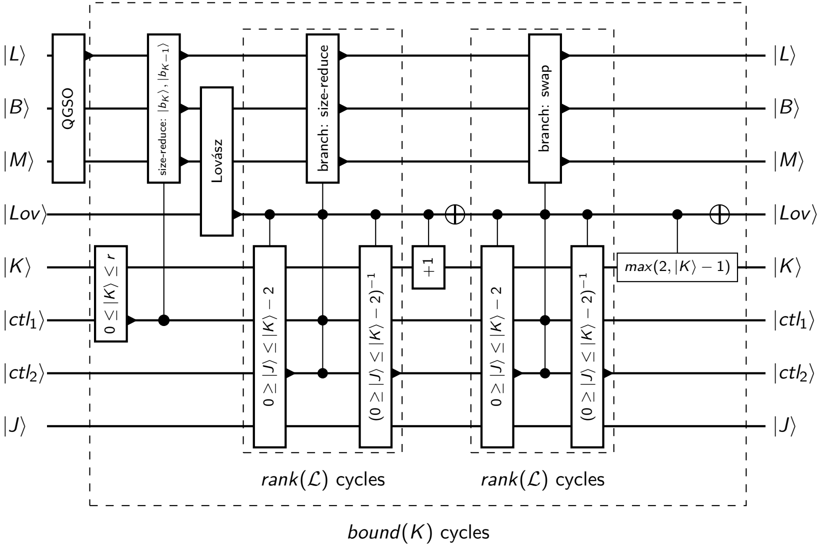
misc compare, round, $\max(x, y)$, ...

Notations

function $f(X)$

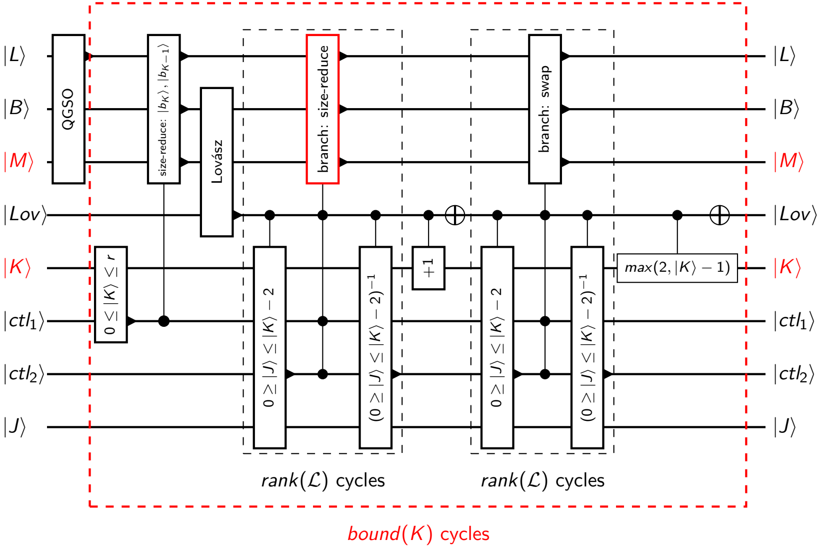
uncompute (run circuit backwards) $(f(X))^{-1}$

Quantum LLL



|L⟩
|B⟩
|M⟩
|Lov⟩
|K⟩
|ctl1⟩
|ctl2⟩
|J⟩
QGSO
size-reduce: |bK⟩, |bK−1⟩
Lovász
branch: size-reduce
branch: swap
0 ≤ |K⟩ ≤ r
0 ≥ |J⟩ ≤ |K⟩ − 2
(0 ≥ |J⟩ ≤ |K⟩ − 2)−1
+1
max(2, |K⟩ − 1)
rank(ℒ) cycles
rank(ℒ) cycles
bound(K) cycles

Quantum LLL



|L⟩
|B⟩
|M⟩
|Lov⟩
|K⟩
|ctl1⟩
|ctl2⟩
|J⟩
QGSO
0 ≤ |K⟩ ≤ r
size-reduce: |bK⟩, |bK−1⟩
Lovász
0 ≥ |J⟩ ≤ |K⟩ − 2
branch: size-reduce
(0 ≥ |J⟩ ≤ |K⟩ − 2)−1
+1
branch: swap
max(2, |K⟩ − 1)
rank(ℒ) cycles
rank(ℒ) cycles
bound(K) cycles

Pitfall I: unbounded loops

Classical

- Apply operation until loop terminates

Pitfall I: unbounded loops

Quantum

- Apply as often as necessary, but not *too* often

Classical

- Apply operation until loop terminates

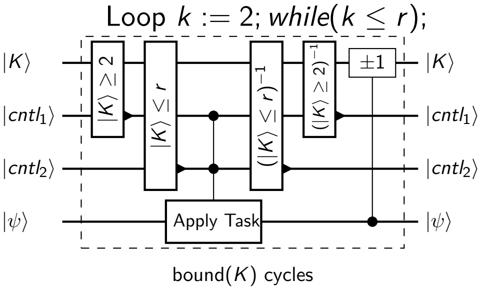
Pitfall I: unbounded loops

Quantum

- Apply as often as necessary, but not *too* often

Classical

- Apply operation until loop terminates



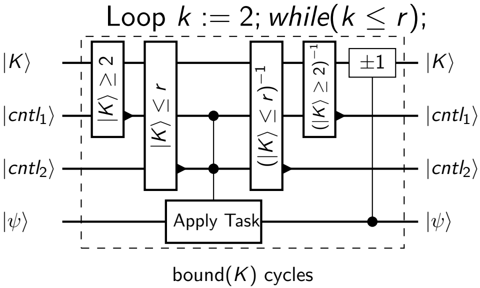
Pitfall I: unbounded loops

Quantum

- Apply as often as necessary, but not *too* often

Classical

- Apply operation until loop terminates



Quantum: worst-case running time for all (unbounded) loops

Pitfall Part II: size-reduction cleanup

Karlsruhe Institute of Technology

Size reduction: $b_i \xrightarrow{\textit{reduce by } b_j} \hat{b}_i$
Update M s.t. $\hat{B} = M\hat{B}^*$

Classical

Pitfall Part II: size-reduction cleanup

Size reduction: $b_i \xrightarrow{\text{reduce by } b_j} \hat{b}_i$
Update M s.t. $\hat{B} = M\hat{B}^*$

Classical

$$\lceil m_{ij} \rfloor \leftarrow \mathrm{round}(m_{ij})$$

$$\hat{b}_i \leftarrow b_i - \lceil m_{ij} \rfloor b_j$$

$$\hat{m}_{ij} \leftarrow m_{ij} - \lceil m_{ij} \rfloor$$

$$\mathit{free}(\lceil m_{ij} \rfloor),\ \mathit{free}(b_i),\ \mathit{free}(m_{ij})$$

Pitfall Part II: size-reduction cleanup

Size reduction: $b_i \xrightarrow{\text{reduce by } b_j} \hat{b}_i$
Update M s.t. $\hat{B} = M\hat{B}^*$

Classical

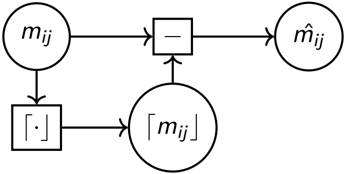
$$\lceil m_{ij} \rfloor \leftarrow \mathrm{round}(m_{ij})$$
$$\hat{b}_i \leftarrow b_i - \lceil m_{ij} \rfloor b_j$$
$$\hat{m}_{ij} \leftarrow m_{ij} - \lceil m_{ij} \rfloor$$
$$\mathit{free}(\lceil m_{ij} \rfloor),\ \mathit{free}(b_i),\ \mathit{free}(m_{ij})$$

m_{ij}, b_i can not be recomputed from $\hat{m}_{ij}$, $\hat{b}_{ij}$

$\Rightarrow$ information about *larger* basis is lost

Pitfall Part II: size-reduction cleanup

Quantum

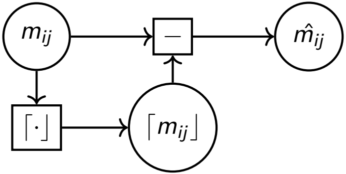


$|m_{ij}\rangle$, $|b_i\rangle$ can not be recomputed from $|\hat{m}_{ij}\rangle$, $|\hat{b_{ij}}\rangle$

$\Rightarrow$ $|b_i\rangle$, $|m_{ij}\rangle$ or $|\lceil m_{ij}\rfloor\rangle$ need to be preserved for reversibility

Pitfall Part II: size-reduction cleanup

Quantum



$$|m_{ij}\rangle, |b_i\rangle \text{ can not be recomputed from } |\hat{m}_{ij}\rangle, |\hat{b_{ij}}\rangle$$
$$\Rightarrow |b_i\rangle, |m_{ij}\rangle \text{ or } |\lceil m_{ij}\rfloor\rangle \text{ need to be preserved for reversibility}$$

Quantum: need *fresh* memory in every size-reduction

(similar issues arises from divisions/ preserving the remainder for fp-numbers)

Impact?

$$\begin{array}{c} |M^{(0)}\rangle|0\rangle\ldots|0\rangle \\ \big\downarrow \text{size-reduce} \\ |M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\ldots|0\rangle \end{array}$$

- Size reduction is conditionally applied to all vectors of $|M^{(i)}\rangle$
- Reversible size-reduction: $|M^{(i)}\rangle|B\rangle|0\rangle \Rightarrow |M^{(i)}\rangle|B\rangle|M^{(i+1)}\rangle$

Impact?

$$|M^{(0)}\rangle|0\rangle...|0\rangle \xrightarrow{\text{size-reduce}} |M^{(0)}\rangle|M^{(1)}\rangle|0\rangle...|0\rangle \xrightarrow{\text{size-reduce}} |M^{(0)}\rangle|M^{(1)}\rangle|M^{(2)}\rangle|0\rangle...|0\rangle \xrightarrow{\text{size-reduce}} ... \xrightarrow{\text{size-reduce}} |M^{(0)}\rangle|M^{(1)}\rangle...|M^{(bound(K))}\rangle$$

- Size reduction is conditionally applied to all vectors of $|M^{(i)}\rangle$
- Reversible size-reduction: $|M^{(i)}\rangle|B\rangle|0\rangle \Rightarrow |M^{(i)}\rangle|B\rangle|M^{(i+1)}\rangle$

Impact?

$$|M^{(0)}\rangle|0\rangle\ldots|0\rangle$$

$$\Big\downarrow \text{size-reduce}$$

$$|M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\ldots|0\rangle$$

$$\Big\downarrow \text{size-reduce}$$

$$|M^{(0)}\rangle|M^{(1)}\rangle|M^{(2)}\rangle|0\rangle\ldots|0\rangle$$

$$\Big\downarrow \text{size-reduce}$$

$$\ldots$$

$$\Big\downarrow \text{size-reduce}$$

$$|M^{(0)}\rangle|M^{(1)}\rangle\ldots|M^{(bound(K))}\rangle$$

- Size reduction is conditionally applied to all vectors of $|M^{(i)}\rangle$
- Reversible size-reduction: $|M^{(i)}\rangle|B\rangle|0\rangle \Rightarrow |M^{(i)}\rangle|B\rangle|M^{(i+1)}\rangle$
- How many qubits does this require?
  - sizeOf(M) qubits for each reduction
  - bound(K) many iterations

  $\rightarrow$ bound(K) $\times$ sizeOf(M)

Impact?

$$
\begin{array}{c}
|M^{(0)}\rangle|0\rangle\ldots|0\rangle \\
\Big\downarrow \text{size-reduce} \\
|M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\ldots|0\rangle \\
\Big\downarrow \text{size-reduce} \\
|M^{(0)}\rangle|M^{(1)}\rangle|M^{(2)}\rangle|0\rangle\ldots|0\rangle \\
\Big\downarrow \text{size-reduce} \\
\ldots \\
\Big\downarrow \text{size-reduce} \\
|M^{(0)}\rangle|M^{(1)}\rangle\ldots|M^{(bound(K))}\rangle
\end{array}
$$

- Size reduction is conditionally applied to all vectors of $|M^{(i)}\rangle$
- Reversible size-reduction: $|M^{(i)}\rangle|B\rangle|0\rangle \Rightarrow |M^{(i)}\rangle|B\rangle|M^{(i+1)}\rangle$

- How many qubits does this require?
  - sizeOf(M) qubits for each reduction
  - bound(K) many iterations

  $\rightarrow$ bound(K) $\times$ sizeOf(M)

Bad if bound(K) is large

Can we do better?

$$|M^{(0)}\rangle|0\rangle...|0\rangle$$

$$\Big\downarrow \text{size-reduce}$$

$$|M^{(0)}\rangle|M^{(1)}\rangle|0\rangle...|0\rangle$$

$$\Big\downarrow$$

$$...$$

$$\Big\downarrow$$

$$|M^{(0)}\rangle|M^{(1)}\rangle...|M^{(j-2)}\rangle|M^{(j-1)}\rangle|0\rangle$$

$$\searrow \text{size-reduce}$$

$$|M^{(0)}\rangle|M^{(1)}\rangle...|M^{(j-1)}\rangle|M^{(j)}\rangle$$

Can we do better?

$$
\begin{tikzcd}[column sep=small]
|M^{(0)}\rangle|0\rangle\dots|0\rangle \arrow[d, "\text{size-reduce}"'] & & |M^{(0)}\rangle|0\rangle\dots|0\rangle|M^{(j)}\rangle \\
|M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\dots|0\rangle \arrow[d] & & |M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\dots|0|M^{(j)}\rangle\rangle \arrow[u, dashed, "(\textit{size-reduce})^{-1}"'] \\
\dots \arrow[d] & & \dots \arrow[u, dashed] \\
|M^{(0)}\rangle|M^{(1)}\rangle\dots|M^{(j-2)}\rangle|M^{(j-1)}\rangle|0\rangle \arrow[dr, "\text{size-reduce}"'] & & |M^{(0)}\rangle|M^{(1)}\rangle\dots|M^{(j-3)}\rangle|M^{(j-2)}\rangle|0\rangle|M^{(j)}\rangle \arrow[u, dashed] \\
& |M^{(0)}\rangle|M^{(1)}\rangle\dots|M^{(j-1)}\rangle|M^{(j)}\rangle \arrow[ur, dashed, "(\textit{size-reduce})^{-1}"'] &
\end{tikzcd}
$$

Can we do better?

$$
\begin{tikzcd}
|M^{(0)}\rangle|0\rangle\ldots|0\rangle \arrow[d, "\text{size-reduce}"'] & & |M^{(0)}\rangle|0\rangle\ldots|0\rangle|M^{(j)}\rangle \\
|M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\ldots|0\rangle \arrow[d] & & |M^{(0)}\rangle|M^{(1)}\rangle|0\rangle\ldots|0|M^{(j)}\rangle\rangle \arrow[u, dashed, "(\mathit{size\text{-}reduce})^{-1}"'] \\
\cdots \arrow[d] & & \cdots \arrow[u, dashed] \\
|M^{(0)}\rangle|M^{(1)}\rangle\ldots|M^{(j-2)}\rangle|M^{(j-1)}\rangle|0\rangle \arrow[dr, "\text{size-reduce}"'] & & |M^{(0)}\rangle|M^{(1)}\rangle\ldots|M^{(j-3)}\rangle|M^{(j-2)}\rangle|0\rangle|M^{(j)}\rangle \arrow[u, dashed] \\
& |M^{(0)}\rangle|M^{(1)}\rangle\ldots|M^{(j-1)}\rangle|M^{(j)}\rangle \arrow[ur, dashed, "(\mathit{size\text{-}reduce})^{-1}"'] &
\end{tikzcd}
$$

$\rightarrow$ Requires at most: $j \times$sizeOf(M) qubits

Impact?

$$\left|M^{(0)}\right\rangle$$

Impact?

$$
\begin{aligned}
& |M^{(0)}\rangle \\
\rightarrow\ & |M^{(0)}\rangle|M^{(j)}\rangle
\end{aligned}
$$

Impact?

$$
\begin{aligned}
& |M^{(0)}\rangle \\
\rightarrow\ & |M^{(0)}\rangle|M^{(j)}\rangle \\
\rightarrow\ & \ldots \\
\rightarrow\ & |M^{(0)}\rangle|M^{(j)}\rangle\ldots|M^{(bound(K))}\rangle \\
& (\text{Optimal for } j = \sqrt{\text{bound}(K)})
\end{aligned}
$$

Impact?

$$
\begin{aligned}
& |M^{(0)}\rangle \\
\rightarrow\ & |M^{(0)}\rangle|M^{(j)}\rangle \\
\rightarrow\ & ... \\
\rightarrow\ & |M^{(0)}\rangle|M^{(j)}\rangle...|M^{(bound(K))}\rangle \\
& (\text{Optimal for } j = \sqrt{\text{bound}(K)})
\end{aligned}
$$

Trade-off:
(Maximal) number of qubits: $\sqrt{\text{bound}(K)} \times \text{sizeOf(M)}$
For # additional iterations: $\text{bound}(K)$

Resource Estimate

- Given basis $B := (b_1, b_2, ..., b_r)$, $b_i \in \mathbb{Z}^d$
- (qu)bit-length n in b_i
- $bound(K) := r^2 \log \hat{B}$, $\hat{B} :=$ bounds norm of initial basis

Resource Estimate

- Given basis $B := (b_1, b_2, ..., b_r)$, $b_i \in \mathbb{Z}^d$
- (qu)bit-length n in b_i
- $bound(K) := r^2 \log \hat{B}$, $\hat{B}$:= bounds norm of initial basis

	#Toffoli	#Qubits
QLLL	$O\left(2 \log \hat{B}(r^3 d + r^4)\left(\frac{n^2}{\log n} + 2n\right)\right)$	$max(d, r) \cdot n$

Resource Estimate

- Given basis $B := (b_1, b_2, ..., b_r),\ b_i \in \mathbb{Z}^d$
- (qu)bit-length n in b_i
- $bound(K) := r^2 \log \hat{B},\ \hat{B} :=$ bounds norm of initial basis

	#Toffoli	#Qubits
QLLL	$O\left(2 \log \hat{B}(r^3 d + r^4)\left(\frac{n^2}{\log n} + 2n\right)\right)$	$max(d, r) \cdot n$

	$\#\text{Qubits}_M$
text-book	$O\left(r^3 d \log \hat{B}(\log \hat{B})^{\frac{1}{2}}\right)$
Schnorr	$O\left(r^2 d \log \hat{B}(\log \hat{B})^{\frac{1}{2}}\right)$
L^2	$O\left(r(\log \hat{B})^{\frac{1}{2}}(1.6d + o(d))\right)$

Application: Groverization of Attack on Mersenne number cryptosystems

Problem

- Given $a, b \xleftarrow{\$} \mathbb{Z}_p$ with *low* Hamming weight , $G \xleftarrow{\$} \mathbb{Z}_p$
- Given $\mathrm{pk} := aG + b = H \mod p$, Find a, b

Application: Groverization of Attack on Mersenne number cryptosystems

Problem

- Given $a, b \xleftarrow{\$} \mathbb{Z}_p$ with *low* Hamming weight , $G \xleftarrow{\$} \mathbb{Z}_p$
- Given pk $:= aG + b = H \mod p$, Find a, b

(Best) approach due to **Beunardeau2017OnTH** applies lattice reduction after partitioning sparse a, b, such that each partition represents small number

Resource Estimate of Grover Oracle

Instantiation for 256-bits of security with $n = 756839$ the QLLL oracle requires:

Resource Estimate of Grover Oracle

Instantiation for 256-bits of security with $n = 756839$ the QLLL oracle requires:

	#Toffoli	#Qubits
text-book	$\approx 2^{85}$	$\approx 2^{52}$
Schnorr	$\approx 2^{65}$	$\approx 2^{44}$
L^2	$\approx 2^{55}$	$\approx 2^{33}$

Conclusions

Quantum vs. Classical

Quantum:

- Apply size-reduction **and** swap conditionally
- Average is worst-case, domain knowledge gives significant improvements!
- Split LLL reduction to improve qubit overhead $O\left(r^3 d \log \hat{B} (\log \hat{B})^{\frac{1}{2}}\right)$

Classical:

- Apply either size-reduction **or** swap
- Bad worst-case, good (empirical) average time