Eliminating Intermediate Measurements in Space-Bounded Quantum Computation

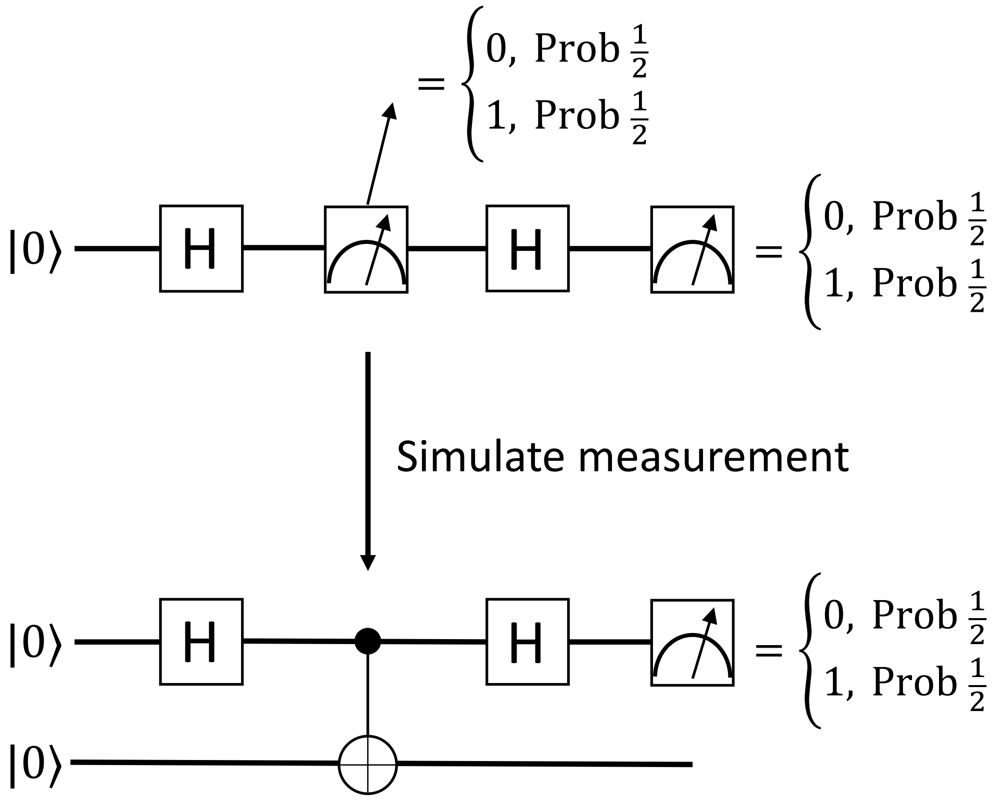
Bill Fefferman and Zack Remscrim (University of Chicago)

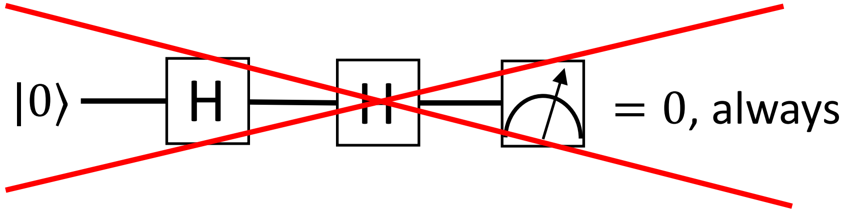
+

Quantum Logspace Algorithm for Powering Matrices with Bounded Norm

Uma Girish, Ran Raz and Wei Zhan (Princeton University)

Principle of Deferred Measurement





Okay if only care about time

Not okay if also care about space

Cost = 1 ancilla (space) per measurement

Ex: $\log(n)$-space and $poly(n)$-time

May have $poly(n)$ measurements

⇒ exponential blowup to $poly(n)$-space

Possible without space blowup? [Watrous’01, Watrous’03, van Melkebeek-Watson’12, Ta-Shma’13, Fefferman-Lin’16, etc.]

Main Results

Can eliminate intermediate measurement without space blowup or time blowup

Quantum logspace algorithms and matching hardness results for many natural linear algebraic problems

For well-conditioned matrix A, approximate

$\det(A)$

A^{-1}

A^m

etc.

$\Rightarrow$ can eliminate intermediate measurements

Why is Space Important?

Current experimental quantum computers: Noisy Intermediate Scale era [Preskill'18]

Intermediate Scale = few qubits

IBM Q System
53 Qubits
Cost: $\gg$ \$400

Apple iPhone 7
128 GB
Cost: \$400

(somewhat unfair comparison)

Near-term quantum computers have few qubits $\Rightarrow$ space is important

Why Eliminate Intermediate Measurements?

Measurements are a natural resource, just like time and space

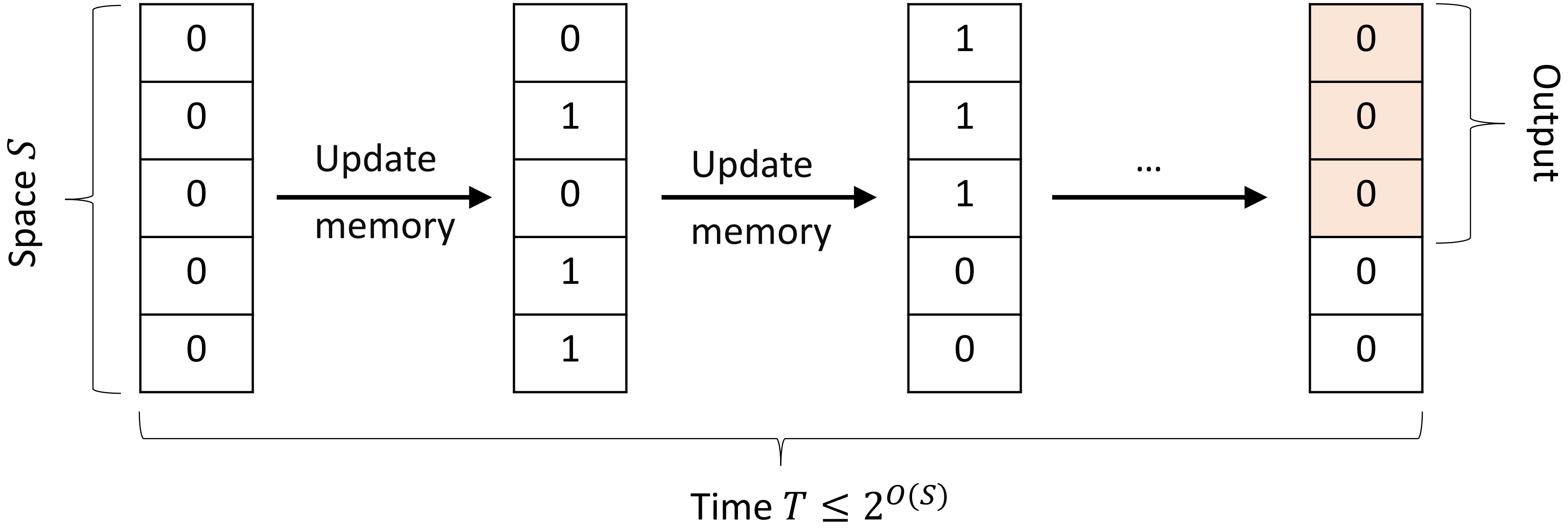
Unitary computations (i.e., without intermediate measurements) are reversible

- Undo computation: useful in design/analysis of algorithms
  - "Tidy" subroutine calling [Bennet-Bernstein-Brassard-Vazirani'97]
  - Quantum rewinding [Watrous'09]
- No heat generation (Landauer's Principle)

Shows definition of "quantum space $s(n)$" is robust

- general $=$ unitary
- For probabilistic space
  - $\subseteq$ general quantum (easy)
  - $\subseteq$ unitary quantum (previously unknown)

Our Model of Space Bounded Computation



We require that there is a classical **deterministic space S** Turing Machine which on input $x \in \{0,1\}^*$, outputs the description of the update operations.

An algorithm **computes** a function $f: \{0,1\}^n \to \{0,1\}^m$ if the output of the algorithm is $f(x)$ (with probability at least $\frac{2}{3}$ if the algorithm is randomized/quantum).

Logspace Algorithms: Space $O(\log n)$ and Time $\mathbf{poly}(n)$

Classical Logspace Algorithms:

$\boldsymbol{L}$: Classical algorithms where the updates are deterministic transition matrices.

$\boldsymbol{BPL}$: Classical randomized algorithms where the updates are stochastic matrices.

Quantum Logspace Algorithms:

$\boldsymbol{BQ_U L}$: (Unitary Quantum algorithms)

- Store qubits.
- Updates are by unitary quantum channels.
- The output is the outcome of measuring some qubits at the end of the computation.

$\boldsymbol{BQ_Q L}$: (Pure Quantum algorithms)

- Updates are by unital quantum channels.

$\boldsymbol{BQL}$: (Quantum algorithms)

- Updates are by general quantum channels.

Logspace Algorithms:
Space $O(\log n)$ and Time $\mathrm{poly}(n)$

$\boldsymbol{BQ_UL}$ Unitary Logspace	$\boldsymbol{BQ_QL}$ Pure Quantum Logspace	$\boldsymbol{BQL}$ General Quantum Logspace
Unitary quantum channels	Unital quantum channels	General quantum channels
✓ Unitary operations.	✓ Unitary operations. ✓ Intermediate Measurements.	✓ Unitary operations. ✓ Intermediate Measurements. ✓ Reset qubits.
Purely quantum memory, Unitary Operations	Purely quantum memory, Unitary Operations + Intermediate Measurements	Quantum Memory, Unitary Operations, Intermediate Measurements + Classical Memory + Copy measurement outcome to classical memory

Our Result:
Contraction Matrix Powering is in $\boldsymbol{BQ_UL}$.

Given: An $n \times n$ real contraction matrix A (i.e., $\|A\| \leq 1$),
$T \leq \text{poly}(n), \epsilon \geq \frac{1}{\text{poly}(n)}$ and indices $s, t \in \{1, \dots, n\}$
Estimate $\mathrm{A}^{\mathrm{T}}[s, t]$ up to ϵ additive error.

We [Girish-Raz-Zhan'20] show that this problem can be solved in $\boldsymbol{BQ_UL}$.
Equivalently, Iterated Contraction Matrix Multiplication is in $\boldsymbol{BQ_UL}$.

It is not known if this problem can be solved in $\boldsymbol{BPL}$.

Applications:
Eliminating Intermediate Measurements

We [Girish-Raz-Zhan'20] show that $\boldsymbol{BQ_U L = BQ_Q L}$.

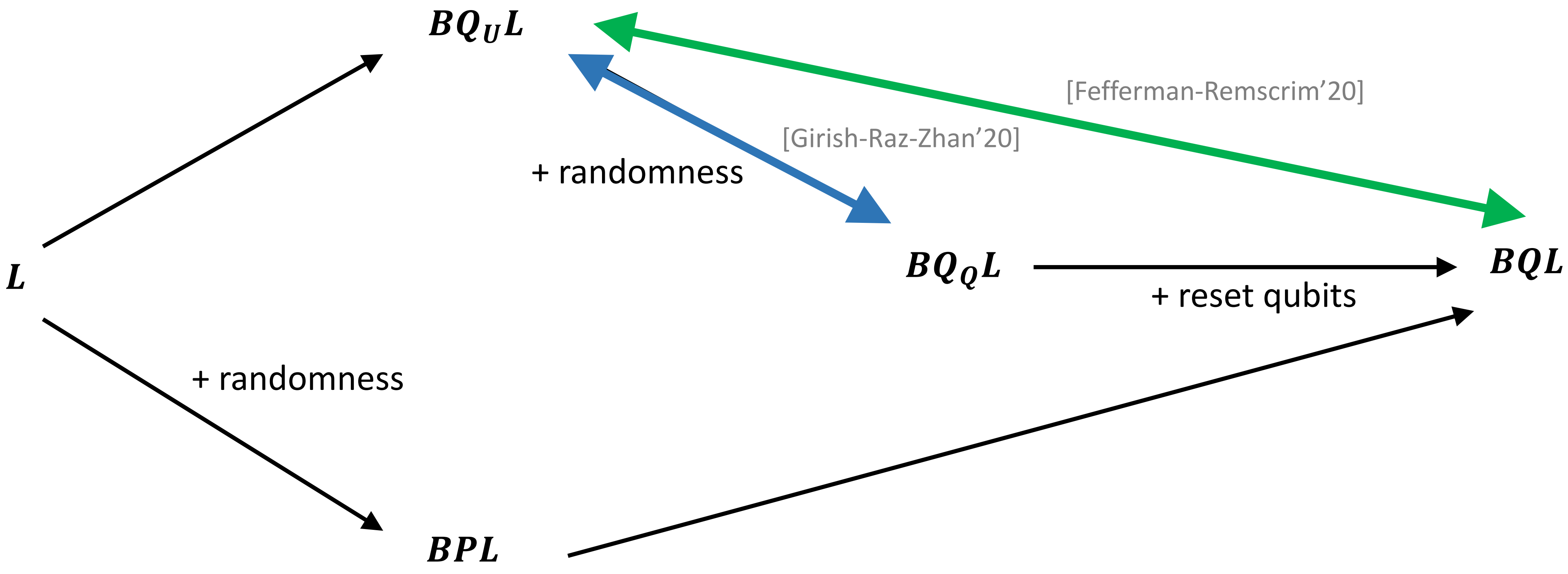
Measurements during computation don't give more power to quantum logspace algorithms with only quantum memory.

In general, we show that a pure quantum algorithm of space S and time T *with intermediate measurements,*
can be simulated in space $O(S + \log T)$ and time $\mathrm{poly}(T, 2^{O(S)})$ *without intermediate measurements.*
$\Rightarrow \boldsymbol{BQ_U SPACE}(s(n)) = \boldsymbol{BQ_Q SPACE}(s(n))$ for any space-constructible $s(n) = \Omega(\log(n))$.

[Fefferman-Remscrim'20] show that $\boldsymbol{BQ_U L = BQ_Q L = BQL}$.
$\Rightarrow \boldsymbol{BQ_U SPACE}(s(n)) = \boldsymbol{BQSPACE}(s(n))$ for any space-constructible $s(n) = \Omega(\log(n))$.

The ability to do intermediate measurements or reset qubits doesn't give more power to quantum logspace algorithms.

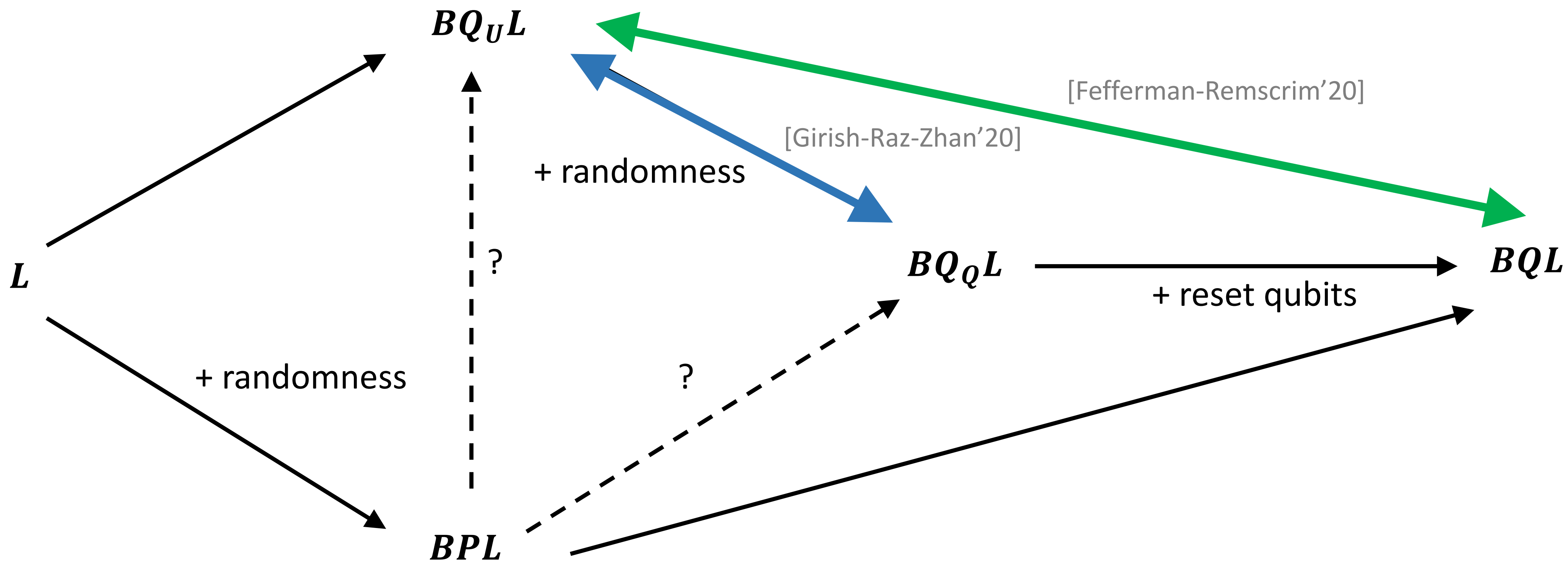
Applications: Derandomizing $\boldsymbol{BQ_QL}$



$\boldsymbol{A} \rightarrow \boldsymbol{B}$ denotes that $\boldsymbol{B}$ is at least as powerful as $\boldsymbol{A}$.

We show that $\boldsymbol{BQ_QL}$ algorithms don’t require randomness.

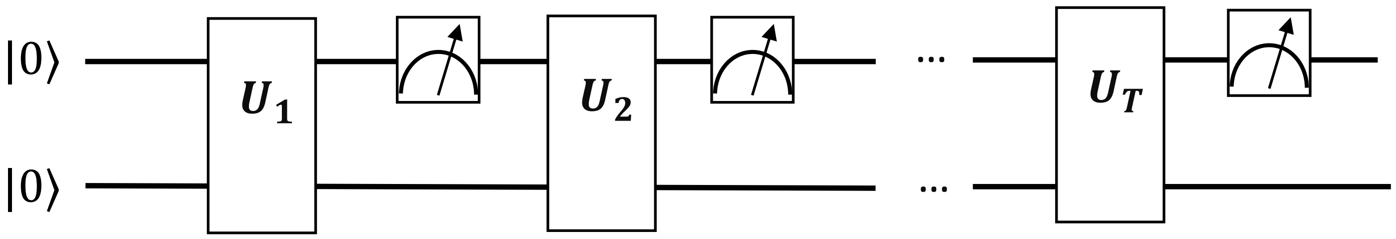
Applications:
$\boldsymbol{BPL}$ versus $\boldsymbol{BQ_UL}$



$\boldsymbol{A}$ → $\boldsymbol{B}$ denotes that $\boldsymbol{B}$ is at least as powerful as $\boldsymbol{A}$.

Connection between Contraction Matrix Multiplication and $\boldsymbol{BQ_QL}$

A generic $\boldsymbol{BQ_QL}$ algorithm



The probability that this circuit outputs 1 is precisely

Unitary matrices

$$\mathrm{vec}(\Pi_1\Pi_1^\dagger)^\dagger(U_T \otimes \overline{U_T})\, M \ldots (U_2 \otimes \overline{U_2})\, M\, (U_1 \otimes \overline{U_1})(v_0 \otimes v_0) \quad = a^\dagger(A_{\mathrm{poly}(n)} \times \cdots \times A_1)b$$

Vector† Contraction matrices Unit vector

$\Pi_1 =$ projection matrix onto $\{\, |i\rangle : i_1 = 1 \,\}$
$v_0 = |0^S\rangle$
$M =$ a diagonal matrix with diagonal entries in $\{0,1\}$.

Our Approach

Step 1. Embed each of the contraction matrices $\boldsymbol{A_1}, \ldots, \boldsymbol{A_T}$ inside *unitary* logspace quantum circuits.

$$A \hookrightarrow \begin{bmatrix} A & \sqrt{\mathbb{I} - AA^\dagger} \\ \sqrt{\mathbb{I} - A^\dagger A} & -A^\dagger \end{bmatrix}$$

Step 2. Compose the quantum circuits carefully.

$$\begin{bmatrix} A & * & 0 \\ * & * & 0 \\ 0 & 0 & \mathbb{I} \end{bmatrix} \times \begin{bmatrix} B & 0 & * \\ 0 & \mathbb{I} & 0 \\ * & 0 & * \end{bmatrix} = \begin{bmatrix} AB & * & * \\ * & * & * \\ * & * & * \end{bmatrix}$$

Step 3. Estimate $\boldsymbol{a}^\dagger \boldsymbol{M} \boldsymbol{b}$ using a quantum circuit that embeds $\boldsymbol{M}$.

An argument similar to Grover's search.

Well-Conditioned Matrix Inversion

Given: invertible $n \times n$ matrix A and indices $i, j \in \{1, \ldots, n\}$

Approx $A^{-1}[i, j]$ to precision $1/poly(n)$

Difficulty depends on condition number $\kappa(A) = \frac{\sigma_1(A)}{\sigma_n(A)}$

$\sigma_1(A)$ = largest singular value, $\sigma_n(A)$ = smallest singular value

Well-conditioned: $\kappa(A) = poly(n)$,

Different params (sparse $2^n \times 2^n$ matrix) in $\boldsymbol{BQP}$ [Harrow-Hassidim-Lloyd'09]

in $\boldsymbol{BQL}$ [Ta-Shma'13]

is $\boldsymbol{BQ_U L}$-complete [Fefferman-Lin'16]

Determinant

$\boldsymbol{DET} =$ problems reducible to computing $\det(A)$ [Cook'85]

$n \times n$ matrix A, entries are n-bit integers

Natural complete problems: Determinant, Matrix Inversion, Matrix Powering, Iterated Matrix Multiplication, etc.

$poly$-conditioned matrix inversion is $\boldsymbol{BQ_UL}$-complete [Fefferman-Lin'16]

We generalize: $poly$-conditioned $\boldsymbol{DET}$-complete problems are $\boldsymbol{BQ_UL}$-complete

$\Rightarrow$ can eliminate intermediate measurements

Difficulty: "standard" reductions generally do not preserve being well-conditioned

Well-Conditioned Matrix Powering

Given: $n \times n$ matrix A, $t \leq poly(n)$, and indices $i, j \in \{1, \ldots, n\}$

Approx $A^t[i,j]$ to precision $1/poly(n)$

Promise: $\|A^k\| \leq poly(n), \forall k \in \{1, \ldots, t\}$ (analogue of "poly-conditioned" for powering)

Simple reduction to $poly$-conditioned matrix inversion $\Rightarrow \in \boldsymbol{BQ_UL}$

$B =$

I	$-A$	0	0	0	0
0	I	$-A$	0	0	0
0	0	I	$-A$	0	0
0	0	0	I	$-A$	0
0	0	0	0	I	$-A$
0	0	0	0	0	I

$B^{-1} =$

I	A	A^2	A^3	A^4	A^5
0	I	A	A^2	A^3	A^4
0	0	I	A	A^2	A^3
0	0	0	I	A	A^2
0	0	0	0	I	A
0	0	0	0	0	I

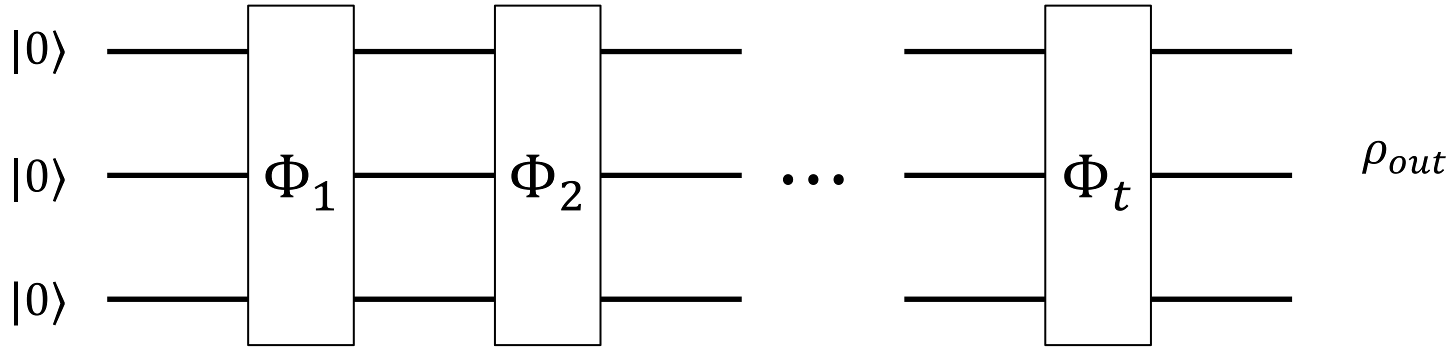
A^t appears in top-right block of B^{-1} and B is $poly$-conditioned

Well-Conditioned Iterated Matrix Product

Similar to powering: Now approx entry of $A_1 \cdots A_t$, $t \leq poly(n)$, $\|A_k \cdots A_h\| \leq poly(n), \forall\ 1 \leq k \leq h \leq t$

Can reduce to $poly$-conditioned matrix inversion $\Rightarrow \in \boldsymbol{BQ_UL}$

Is $\boldsymbol{BQL}$-hard: General quantum circuit is sequence of (arbitrary) quantum channels



$\Rightarrow \boldsymbol{BQL} = \boldsymbol{BQ_UL}$

$\Rightarrow \boldsymbol{BQSPACE}\big(s(n)\big) = \boldsymbol{BQ_USPACE}(s(n))$, any space-constructible $s(n) = \Omega(\log(n))$

Also show results for $\boldsymbol{RQL}$ and $\boldsymbol{NQL}$, and for "$\boldsymbol{QMA}$" and "$\boldsymbol{DQC1}$" versions of $\boldsymbol{BQL}$, $\boldsymbol{RQL}$, and $\boldsymbol{NQL}$

Well-Conditioned Determinant

Given: $n \times n$ matrix A, $\kappa(A) = poly(n)$

- Approx $\log(|\det(A)|)$ to precision $1/poly(n)$ (Approx $|\det(A)|$ to multiplicative factor $1 + 1/poly(n)$)

We show: is $\boldsymbol{BQL}(= \boldsymbol{BQ_U L})$-complete

- Other well-conditioned $\boldsymbol{DET}$-complete also $\boldsymbol{BQL}$-complete
- "Standard" reductions do not preserve well-conditioned (e.g. Berkowitz's algorithm)
- Our reductions: various power series approximations, quantum "instance compression"

[Boix-Adsera, Eldar, and Mehraban'19]: $\in \boldsymbol{DSPACE}(\log^2 n \, poly(\log\log n))$

- Used (different) power series approximations
  - Suggests source of quantum advantage
- We show $\in \boldsymbol{DSPACE}(\log^2 n)$
  - Recall: $\boldsymbol{BQL} \subseteq \boldsymbol{DSPACE}(\log^2 n)$ [Watrous'03]
- Improve dependence on $\kappa(A)$?
  - Would show $\boldsymbol{BQL} \subseteq \boldsymbol{DSPACE}(\log^{2-\delta} n)$