Query and Depth Upper Bounds for Quantum Unitaries via Grover Search

Gregory Rosenthal

University of Toronto

TQC 2022

The unitary synthesis problem

Can every n-qubit unitary U be approximately implemented in poly(n) time using an appropriate classical oracle O_U? [AK'07]

- If yes, then upper bound for $O_U \Rightarrow$ upper bound for U.
  - Interesting because we know more about how to compute boolean functions than unitaries.
- Trivial $\tilde{O}(4^n)$ time solution: oracle encodes a circuit for U.
- $\tilde{O}(2^{n/2})$ time solution [R'21].

Implementing unitaries in low depth

What's the minimum depth required to exactly implement any n-qubit unitary using one- and two-qubit gates (and ancillae)?

- Depth = parallel computation time.
- Trivial $\tilde{O}(4^n)$ upper bound.
- $\tilde{O}(2^n)$ upper bound [STYYZ'21].
- $\tilde{O}(2^{n/2})$ upper bound (with $\tilde{O}(4^n)$ ancillae) [R'21].

Constructing states $\Rightarrow$ implementing unitaries

- **Main definition**: If U is an n-qubit unitary, call a $2n$-qubit unitary A a *U-qRAM* if for all $x \in \{0,1\}^n$,

  $$A|x, 0^n\rangle = |x\rangle \otimes U|x\rangle.$$

  $A|x, y\rangle$ is unspecified for $y \neq 0^n$.
- Think of A as *constructing the state* $U|x\rangle$ controlled on the classical key x, while preserving x.
- Can implement U in $\tilde{O}(2^{n/2})$ time with A and $A^\dagger$ oracles [R'21].

How this all fits together

- Right column follows from the left column:

	Constructing states	Implementing unitaries
Runtime with a classical oracle	poly(n) [Aaronson'16]	$\tilde{O}\left(2^{n/2}\right)$ [R'21]
Circuit depth	$O(n)$ [R'21, STYYZ'21, ZLY'22]	$\tilde{O}\left(2^{n/2}\right)$ [R'21]

- Also: matching $\Omega(2^{n/2})$ query lower bound for approximately implementing Haar random U given A and $A^\dagger$ oracles [R'21].

Implementing U with A and $A^\dagger$ oracles

- By linearity, assume the input is a standard basis state $|x\rangle$.
- First apply A to obtain $|x\rangle \otimes U|x\rangle$.
  - (We can't just trace out x because in general these registers are entangled.)
- $G := A(I_n \otimes (I_n - 2|0^n\rangle\langle 0^n|))A^\dagger$ can be efficiently implemented.
- $G(I_n \otimes U|x\rangle) = (I_n - 2|x\rangle\langle x|) \otimes U|x\rangle$.
- Run exact Grover search in reverse to uncompute x.

Lower bound warmup: permutation matrices

- Grover is optimal for unstructured search, but can we do better than simulating unstructured search?
- For a permutation σ of $\{0,1\}^n$, let $U_\sigma|x\rangle = |\sigma(x)\rangle$ and $A_\sigma|x,y\rangle = |x, y \oplus \sigma(x)\rangle$.
- It takes $\Omega\left(2^{n/2}\right)$ quantum queries to A_σ $(= A_\sigma^\dagger)$ to implement U_σ for random σ [Ambainis'02, Nayak'11].
- Unsatisfying because U_σ is easy to implement in other models.

Why is the Haar random case interesting?

- For fixed U and Haar random R,

$$U = \underbrace{UR}_{\text{Haar random}} \cdot \underbrace{R^\dagger}_{\text{Haar random}}.$$

- $\Rightarrow$ If Haar random unitaries have "low complexity" w.h.p. then *all* unitaries have low (nonuniform) complexity.
- Contrapositive: If *any* unitary has high complexity, then so does a Haar random unitary w.h.p.

Lower bound for Haar random unitaries

Theorem: Let C be such that w.h.p. over Haar random R, for all R-qRAMs A, the circuit $C^{(A,A^\dagger)}$ approx. implements R. Then C makes $\Omega(2^{n/2})$ queries.

- Proof overview: combine previous two slides.
- Fix U, let A be a U-qRAM (e.g. $U = U_\sigma, A = A_\sigma$).
- $(I_n \otimes R)A$ is an RU-qRAM.
- If R is Haar random then so is RU.
- $\Rightarrow C^{((I_n \otimes R)A, A^\dagger(I_n \otimes R^\dagger))}$ approx. implements RU w.h.p. over R.
- Prepending $R^\dagger$ yields an implementation of U using the same number of A and $A^\dagger$ queries as C.

Warmup: sampling $\mathbf{s} \sim \{0,1\}^n$ in $O(n)$ depth

- For each string x of length $< n$, independently sample

$$\mathbf{b}_x \sim \text{Bernoulli}(\mathbb{P}(\mathbf{s} \text{ begins with } x1 \mid \mathbf{s} \text{ begins with } x)).$$

- For k from 1 to n, if the first $k-1$ output bits are the string x, then the k'th output bit is $\mathbf{b}_x$.
- Computing the output in $O(n)$ depth:

$$i\text{'th output bit} = \bigvee_{\substack{t \in \{0,1\}^n \\ t_i = 1}} \bigwedge_{1 \le k \le n} \left(\mathbf{b}_{t_1 \cdots t_{k-1}} = t_k\right).$$

Constructing $\sum_{x\in\{0,1\}^n} \alpha_x|x\rangle$ in $O(n)$ depth

- Replace independent Bernoulli random variables with unentangled one-qubit states.
- Results in $\sum_x \alpha_x|x\rangle|\text{garbage}_x\rangle$.
- $|\text{garbage}_x\rangle$ factors as a tensor product of one-qubit states, each of which has a succinct description as a function of x.
- $\Rightarrow$ can efficiently uncompute $|\text{garbage}_x\rangle$ controlled on x.
- *Remark*: construction works in $\mathsf{QAC}^0_\mathsf{f}$.