Implications of Non-Trivial Quantum Learning of Circuits

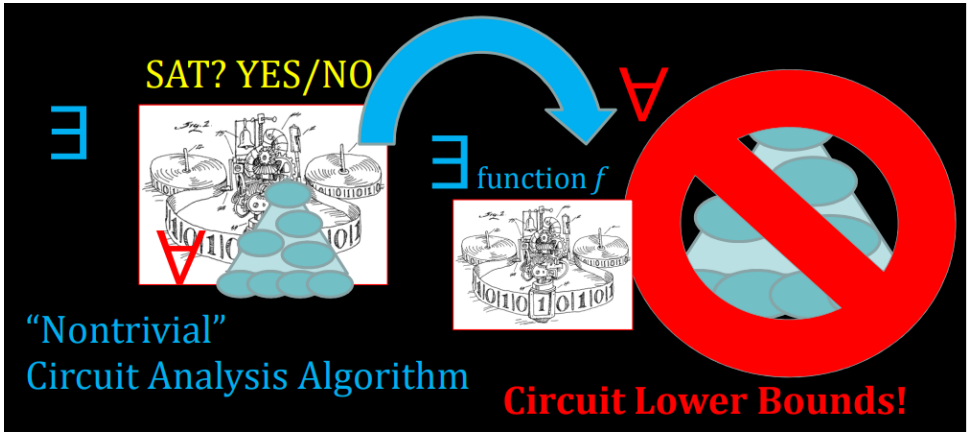
Pawan Paleja

MATH 277A: Quantum Complexity Theory
August 13, 2024

Presentation Overview

General Theme

There is a known interplay between the design of algorithms for a given circuit class and matching lower bounds for the same class.



Quantum Learning Definition

Definition

Let $\mathcal{C}_n$ be a family of Boolean functions on n input variables and $\mathcal{C} = \bigcup_{n \geq 1} \mathcal{C}_n$. For $G : \mathbb{N} \to \mathbb{N}$, we say that $\mathcal{C}_n$ can be $\varepsilon(n), \delta(n))$-learned in time $G(n)$ if there is a quantum learning algorithm $\mathcal{L}$ such that $\mathcal{L}$:

- ... has quantum oracle access to f
- ... uses at most $G(n)$ gates
- ... outputs with probability $1 - \delta(n)$ a quantum hypothesis circuit $\tilde{U}_f$ such that

$$\mathbb{E}_{x \sim 0,1^n} \left[\| (|f(x)\rangle\langle f(x)| \otimes I) \tilde{U}_f x \rangle |0\rangle \|^2 \right] \leq 1 - \varepsilon(n)$$

Main Result

Theorem

If $\exists\gamma$ such that $\mathcal{C}[poly(n)]$ can be learned with probability $\delta = \Omega(1)$ with error $\varepsilon \leq 1/2 - \gamma$ in quantum time $T = o(\gamma^2 * 2^n/n)$, then $BQE \not\subset \mathcal{C}[poly(n)]$

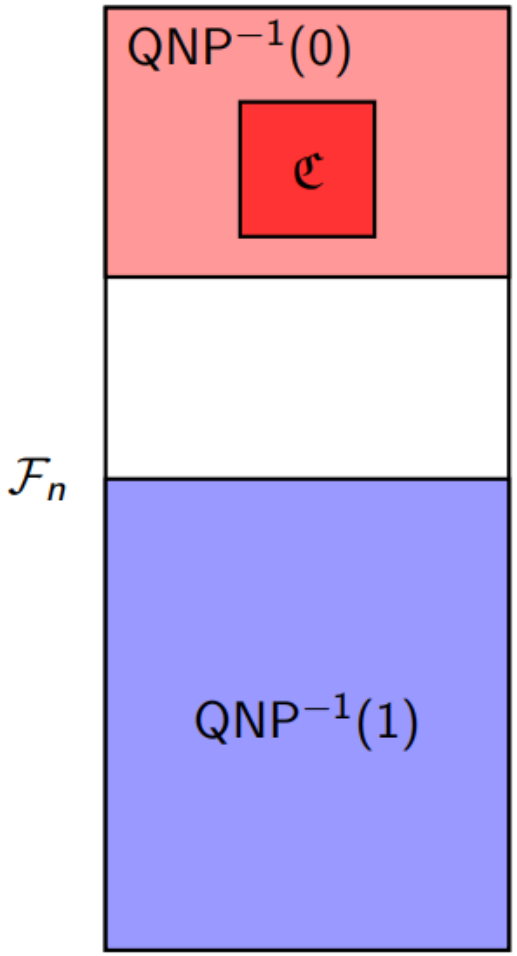
Quantum Learning Implies Circuit Lower Bounds

Framework introduced in [2] and extended to the quantum realm by Arunachala et al: [1]:
Learning algorithm $\mathcal{A}$ finds structure, and so can be used contrapositively against $\mathcal{C}$ to derive algorithm $\mathcal{D}$ which tests for functions that appear random to $\mathcal{C}$ (lack structure) $\rightarrow$
Use PRG with exponential stretch $G : \{0,1\}^{O(n)} \rightarrow \{0,1\}^{2^n}$, and use $\mathcal{D}$ with some constant number of inputs to G to get a hard function for $\mathcal{C}$.
Thus, if we write a PRG that can fool our quantum distinguisher, we can generate a hard function for $\mathcal{D}$.

Distinguisher: Quantum Natural Properties

BQP-Quantum Natural Property $\mathcal{P}$ against $\mathcal{C}$

1. Constructivity: "$tt(f)$ has property P?" is decidable in BQP by quantum algorithm $\mathcal{D}$
2. Largeness: There is a dense set $\mathcal{P}_n \in \mathcal{F}_n$, where the probability a function in $\mathcal{P}_{\setminus}$ is miscategorized as not having the property by $\mathcal{D}$ is $\leq 2^{-|tt(f)|}$.
3. Useful: For $f \in \mathcal{C}_n$, the probability that it has the probability is $2^{-|tt(f|}$.



QNP Distinguisher from non-trivial learning algorithm

Decide whether f is computable by a circuit $\mathcal{C}$ for which we have a learning algorithm $\mathcal{A}$ for

1. Run the learning algorithm on f to get hypothesis h
2. Test h on some constant number of inputs.
3. If it succeeded with high enough probability, output *inside* else output *outside*.

PRG Construction

- The PRG construction and analysis are quite involved but they follow the same steps as Carmosino et al [2] except it is made much more difficult in the quantum case.
- The idea is a 'win-win' argument based on the hardness relation of PSPACE and BQE. The more interesting case is with the hardness assumption that PSPACE $\not\subseteq$ BQE.

PRG Construction

The idea:

1. Start with some hard language $L \in$ PSPACE-complete
2. We define a family of PRGs based on L and assume they don't work, i.e. there is a *quantum distinguisher*
3. Use the direct product trick of [4] to combine multiple instances $\{f\}^k \in L$ into a much harder function. This trick is computed efficiently using the quantum analogue of the Goldreich-Levin Algorithm. [3].
4. Leverage the proof of the NW-generator [5] to reconstruct the hard function using the distinguisher from step (2)
5. This implies that our quantum distinguisher was enough to solve the PSPACE-complete problem, contradiction.

What's different in the quantum setting?

The main trickiness of the analysis comes from proving quantum analogues for all the aforementioned steps. One needs to prove a quantum analogue for NW, Goldreich-Levin, and the direct product proof.

Key Insight: Inherently Probabilistic Circuits I

In analyzing randomized algorithms, one can reduce to the deterministic case by considering a randomized algorithm as picking over the distribution of deterministic algorithms. In the quantum realm, this is not possible, but we can still reduce to a simpler model: Inherently Probabilistic Circuits.

Definition

$\mathcal{A}$ is an inherently probabilistic circuit if $A : 0,1^m \to \mathbf{F}$, where $F = \{D | D : \{0,1\}^{l \to [0,1]}\}$. $\mathcal{A}$ computes $g : \{0,1\}^m \to \{0,1\}^l$ with probability ε if

$$\Pr_{z,\nu}[\nu = g(z)] \geq \varepsilon$$

Key Insight: Inherently Probabilistic Circuits II

Lemma

If R is an IPC and Q a QC s.t. $\forall x$, the output distributions of $R(x)$ and $Q(x)$ are exactly the same. Then for every probabilistic algorithm A, which might have have some classical input w which is allowed to make classical queries to R or Q, we have that

$$\Pr_{A,R}\left[A^R(w) = y\right] = \Pr_{A}, Q\left[A^Q(w) = y\right]$$

Open Questions

- Speedup Lemma: Oliviera and Santhanam [6] proved that nontrivial learning is equivalent to strong learning in the classical sense, and given the analysis here, the speed up still holds. However, it might not hold for important classes like QAC, in fact if it didn't to this would provide evidence against separation between QAC0 and AC0 circuits.
- In the same vein, this paper extended much of the techniques in [2] to the quantum case. Could their result on AC0 not being able to compute NW designs be extended as well?

Bibliography I

[1] Srinivasan Arunachalam et al. “Quantum learning algorithms imply circuit lower bounds”. In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 562–573.

[2] Marco L Carmosino et al. “Learning algorithms from natural proofs”. In: *31st Conference on Computational Complexity (CCC 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2016.

[3] Oded Goldreich and Leonid Levin. “A Hard-Core Predicate for all One-Way Functions”. In: Jan. 1989, pp. 25–32. DOI: 10.1145/73007.73010.

[4] Russell Impagliazzo et al. “Uniform direct product theorems: simplified, optimized, and derandomized”. In: *Proceedings of the fortieth annual ACM symposium on Theory of computing* (2008).

Bibliography II

[5] Noam Nisan and Avi Wigderson. “Hardness vs randomness”. In: *Journal of computer and System Sciences* 49.2 (1994), pp. 149–167.

[6] Igor C Oliveira and Rahul Santhanam. “Conspiracies between learning algorithms, circuit lower bounds and pseudorandomness”. In: *arXiv preprint arXiv:1611.01190* (2016).

The End

Questions? Comments?