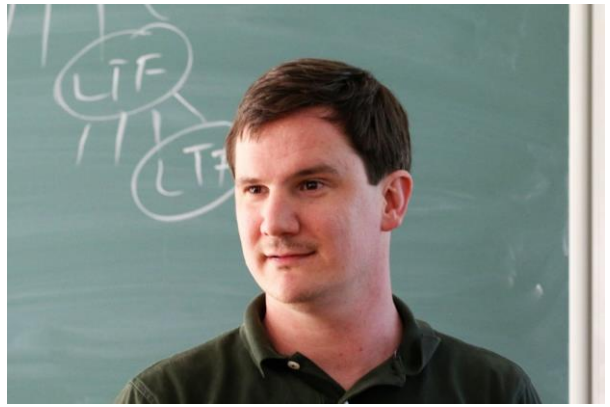
Stronger Connections Between Circuit Analysis and Circuit Lower Bounds, via PCPs of Proximity

Lijie Chen

Ryan Williams



Context: The Algorithmic Method for Proving Circuit Lower Bounds

Proving limitations on non-uniform circuits is **extremely** hard.

Prior approaches (**restrictions**, **polynomial approximations**, etc.) face barriers (**Relativization**, **Algebrization**, **Natural Proofs**).

Algorithmic Method

- **Non-trivial** circuit-analysis algorithm ⇒ **Circuit Lower Bounds**.
- Breakthroughs where previous approaches failed (**NEXP $\not\subset$ ACC^0**).
- Believed to be possible for strong circuits (even $P/poly$).

Context: A Frontier of Circuit Complexity, Depth-2 Threshold Circuits

THR gates : $f(x) = [w \cdot x \geq t]\ w \in Z^n, t \in Z.$

MAJ gates : when w_i's and t are bounded by poly(n).

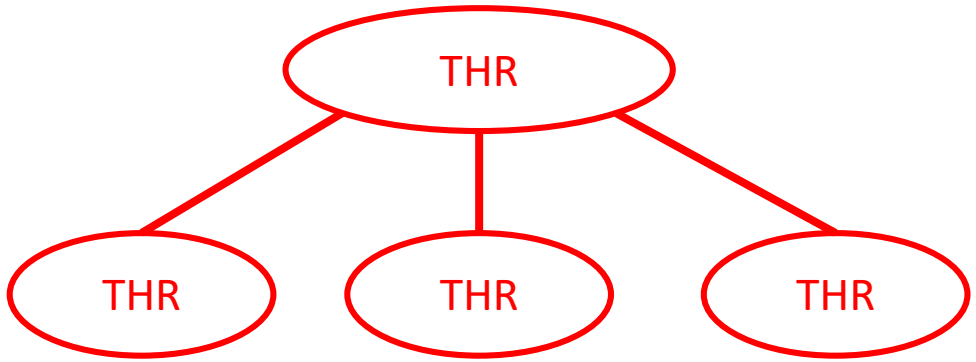
We can also define

$\boldsymbol{THR \circ MAJ}$

$\boldsymbol{MAJ \circ THR}$

$\boldsymbol{MAJ \circ MAJ}$

THR∘THR



Context: A Frontier of Circuit Complexity, Depth-2 Threshold Circuits

Exponential Lower Bounds are known for

$MAJ \circ MAJ$ **[Hajnal-Maass-Pudlák-Szegedy-Turán'93]**

$MAJ \circ THR$ **[Nisan'94]**

$THR \circ MAJ$ **[Forster-Krause-Lokam-Mubarakzjanov-Schmitt-Simon'01]**

NEXP

Non-deterministic Exponential Time.

Frontier Open Question: ***Is NEXP $\subseteq THR \circ THR$?***

Potential Approaches in this talk.

Motivation: Apply the Algorithmic Method to THR of THR?

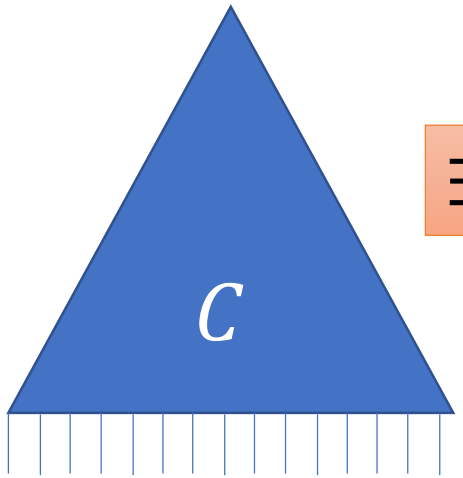
What Circuit-Analysis Tasks?

$\mathfrak{C}$-SAT

$\mathfrak{C}$-CAPP

Derandomization!!

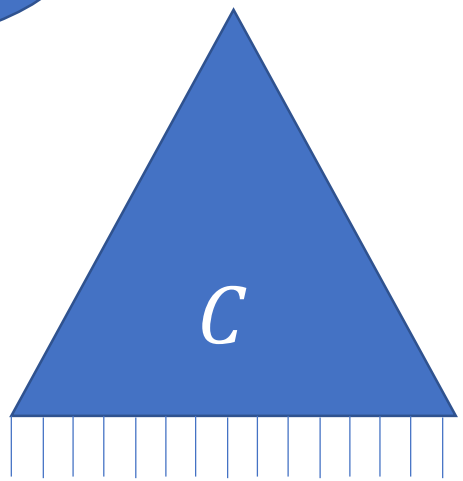
Non-trivial Circuit-Analysis Algorithms ⇒Circuit Lower Bounds



$\exists$ x s.t. $C(x) = 1$?

$2^n/n^{\omega(1)}$ time?

$\exists x$?



$x \sim U_n$

Estimate quantity $\Pr_{x \sim U_n}[C(x) = 1]$, with additive error ε

ε: constant or inverse polynomial

Motivation: Apply the Algorithmic Method to THR of THR?

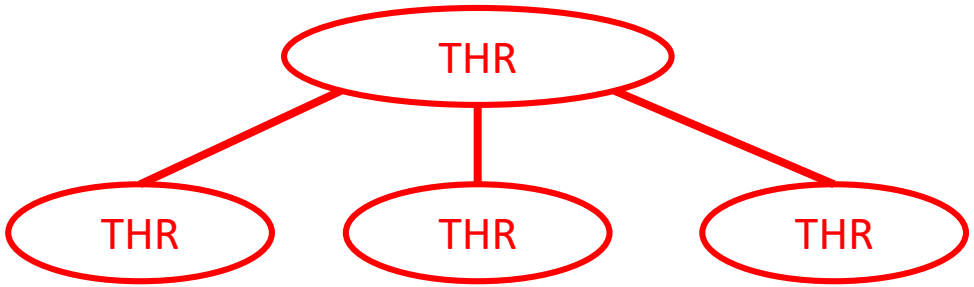
Most previous work on the algorithmic method exploits **SAT** algorithms.

Problem

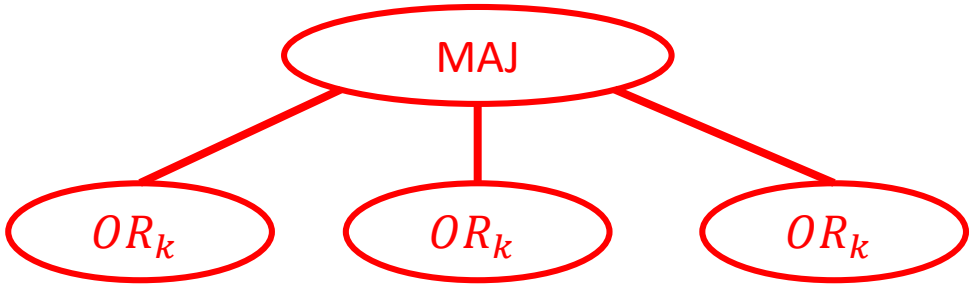
SAT of THR of THR is ***probably*** very hard.
A special case is MAX-k-SAT, for which no non-trivial ($2^n/n^{\omega(1)}$ time) algorithm is known for $\boldsymbol{k = \omega(\log n)}$ and $poly(n)$ clauses.

Considered to be a **barrier** for the Algorithmic Approach.

THR∘THR



MAX-k-SAT



Motivation: Apply the Algorithmic Method to THR of THR?

~~SAT of THR of THR~~ : *prob*

But derandomization is widely believed to be **possible**.

NQP
Non-deterministic Quasi-Polynomial Time. $(n^{polylog(n)})$

From Derandomization (CAPP)
⇒ Circuit Lower Bounds

For a circuit class $\mathfrak{C}$,

- $2^n/n^{\omega(1)}$-time CAPP for $(\mathbf{AND}_{\mathbf{poly}(n)} \circ \mathbf{OR}_3 \circ \mathfrak{C})$
  $\Rightarrow NEXP \not\subset \mathfrak{C}$ [Williams'13/14, Santhanam Williams'14, Ben-Sasson Viola'14]
- $2^n/n^{\omega(1)}$-time CAPP for $(\boldsymbol{AC_0} \circ \mathfrak{C})$
  $\Rightarrow NEXP$ can't be $\frac{1}{2} + o(1)$-approximated by $\mathfrak{C}$ [R. Chen Oliveira Santhanam'18]
- $2^{n-n^{\varepsilon}}$-time CAPP for $(\mathbf{AND}_{\mathbf{poly}(n)} \circ \mathbf{OR}_3 \circ \mathfrak{C})$
  $\Rightarrow NQP \not\subset \mathfrak{C}$ [Murray Williams'18]
- $2^{n-n^{\varepsilon}}$-time CAPP for $(\mathbf{AC_0} \circ \mathfrak{C})$
  $\Rightarrow NQP$ can't be $\frac{1}{2} + o(1)$-approximated by $\mathfrak{C}$ [L. Chen'19]

Back to THR of THR

~~SAT of THR of THR~~ : *probably* very hard

To show $NEXP \not\subset THR \circ THR$, we need to derandomize $\mathrm{AND}_{\mathrm{poly}(n)} \circ \mathrm{OR}_3 \circ THR \circ THR$, which could be harder.

Our result 1

It suffices to derandomize $THR \circ THR$.

Our result 2

Surprisingly, it indeed only suffices to derandomize $THR \circ MAJ$ or $MAJ \circ MAJ$!

General Result: A Stronger Connection Between Circuit-Analysis Algorithms and Circuit Lower Bounds

For a circuit class $\mathfrak{C}$:

- $\boldsymbol{2^n/n^{\omega(1)}}$**-time** CAPP for $\bigoplus_2 \circ \mathfrak{C}$, $AND_2 \circ \mathfrak{C}$, or $OR_2 \circ \mathfrak{C}$

$\Rightarrow NEXP \not\subset \mathfrak{C}$.

- $\boldsymbol{2^{n-n^{\varepsilon}}}$**-time** CAPP for $\bigoplus_2 \circ \mathfrak{C}$, $AND_2 \circ \mathfrak{C}$, or $OR_2 \circ \mathfrak{C}$

$\Rightarrow NQP \not\subset \mathfrak{C}$.

Why the constant "2"?

- **Short answer**: A PCP system needs to make at least 2 queries.
- **Long answer**: See the paper☺

Tighter Connections for Algorithms/Lower Bounds for THR of THR

Luckily, the **"2"** doesn't matter for $THR \circ THR$ ☺

$$\oplus_2 \circ THR \circ THR \subseteq THR \circ THR$$

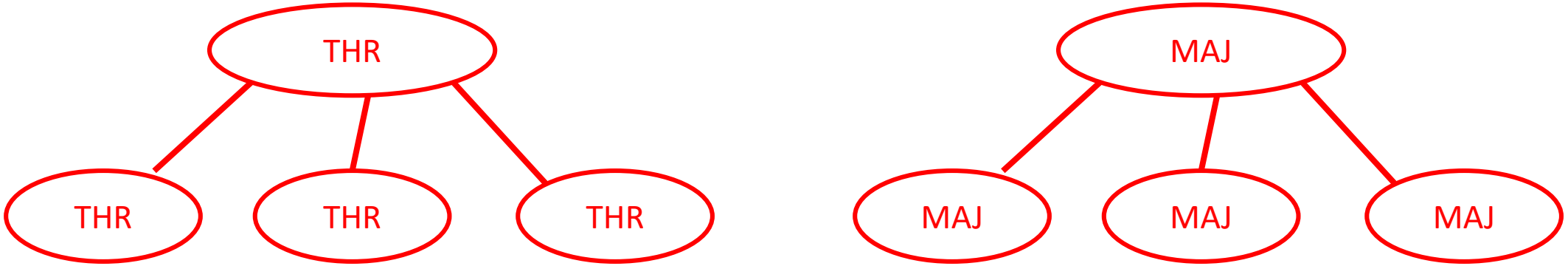
$2^n/n^{\omega(1)}$-time CAPP algorithm for $THR \circ THR$
$\Rightarrow NEXP \not\subset THR \circ THR$.

$2^n/n^{\omega(1)}$-time CAPP algorithm for TC_d
$\Rightarrow NEXP \not\subset TC_d$.

TC_d: depth-d, poly-size, linear threshold circuits

Let Us Make Our Life Even Easier

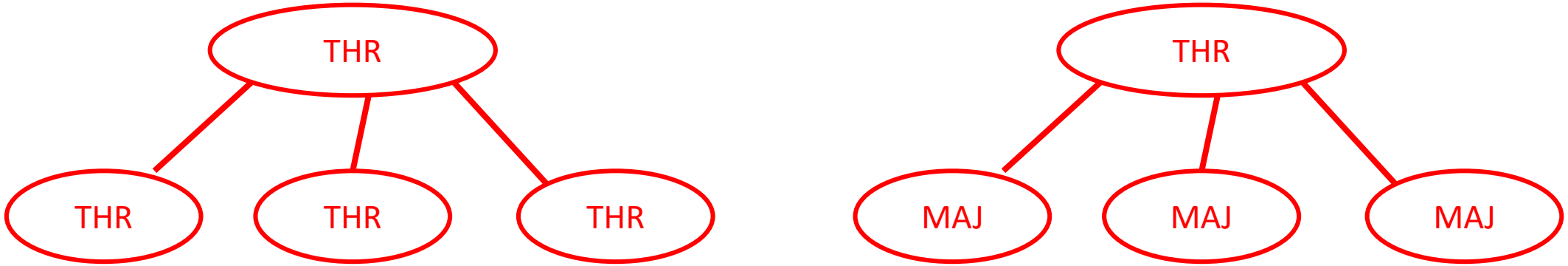
Poly-size $\boldsymbol{THR \circ THR}$ and $\boldsymbol{MAJ \circ MAJ}$ are **equivalent** for Non-Trivial ($2^n/n^{\omega(1)}$ time) CAPP Algorithms when $\varepsilon = 1/poly(n)$!



Proved by new structure lemmas for $THR \circ THR$

Let Us Make Our Life Even Easier

Poly-size $THR \circ THR$ and $THR \circ MAJ$ are **equivalent** for Non-Trivial ($2^n/n^{\omega(1)}$ time) CAPP Algorithms for any constant $\varepsilon > 0$!



Proved by new structure lemmas for $THR \circ THR$

Corollary

If there are

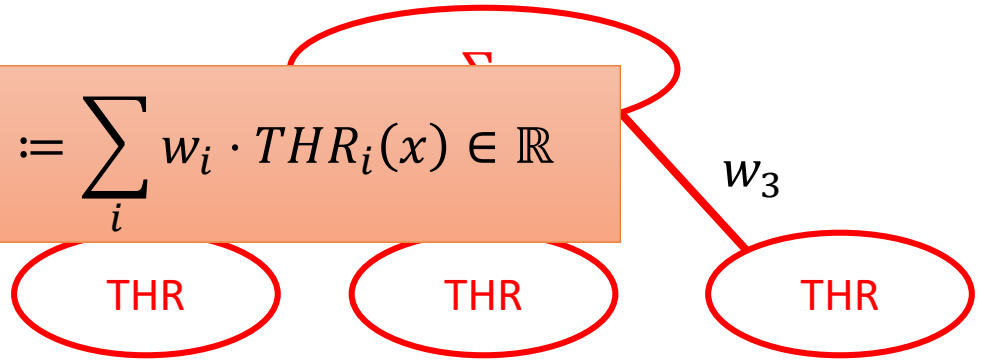
$2^n/n^{\omega(1)}$-time CAPP for $MAJ \circ MAJ$ with $\varepsilon = 1/poly(n)$, or a $2^n/n^{\omega(1)}$-time CAPP for $THR \circ MAJ$ with constant ε,

then $\boldsymbol{NEXP \not\subset THR \circ THR}$.

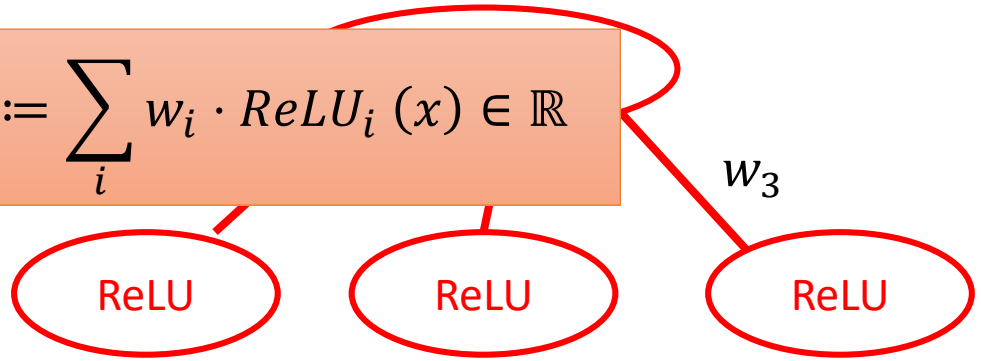
Another Application: Inapproximability by Depth-2 Neural Networks

Depth-2 Neural Network

$$N(x) := \sum_i w_i \cdot THR_i(x) \in \mathbb{R}$$



$$N(x) := \sum_i w_i \cdot ReLU_i(x) \in \mathbb{R}$$



Thm

For every k and constant $\delta < 1/2$, there is a function $f \in NP$ such that f cannot be δ **approximated** by Depth-2 Neural Networks of size n^k

Improved **[Wil'18]**, which proved that there is such an $f \in NP$ which cannot be **exactly computed** by Depth-2 Neural Networks of size n^k.

Philosophy

Using PCP *Algorithmically* to Prove Circuit Lower Bounds *(Remember: PCPs are algorithms!)*

If you want to prove $P = NP$, then PCPs should make your life **much easier** *(now you **only** need an algorithm for $(\frac{7}{8} + \varepsilon)$-approximation to 3-SAT!) [Håstad'97]*

*(Well, I don't really believe in $P = NP$.) We only want to **derandomize** circuits. But PCPs still make our life easier (though in a more indirect way)*

Starting Point: Non-deterministic Derandomization Suffices for Circuit Lower Bounds

$\mathfrak{C}$-GAP-TAUT (*tautology*)

Distinguish between

[Wil'13] $2^n/n^{\omega(1)}$ time non-deterministic

x

Non-deterministic Algorithm for GAP-TAUT

Given a general circuit C, we want a $2^n/n^{\omega(1)}$ time non-deterministic algo $\mathbb{A}$, such that:

1. If C is a tautology, then $\mathbb{A}$ accepts on some guesses.
2. If $\Pr_x[C(x) = 1] \leq 1/2$, $\mathbb{A}$ rejects on all guesses.

Proof Overview: Outline

[Wil'13]

$2^n/n^{\omega(1)}$ ... rithm for GAP-TAUT ... th $\varepsilon = 1/2$

$\Rightarrow \quad NEXP$ ...

Key point: make use of this assumption as much as possible!

Assume $NEXP \subset \mathfrak{C}$

Think of $\mathfrak{C}$ as $THR \circ THR$

Non-trivial CAPP on $\mathrm{OR}_3 \circ \mathfrak{C}$ with constant ε

$2^n/n^{\omega(1)}$ non-deterministic GAP-TAUT for $P/poly$

$NEXP \not\subset P/poly$ $\Rightarrow NEXP \not\subset \mathfrak{C}$ Contradiction!

Goal: Designing the Algorithm under Assumption

Assume $NEXP \subset \mathfrak{C}$

Think of $\mathfrak{C}$ as $THR \circ THR$

Non-trivial CAPP on $\mathrm{OR}_3 \circ \mathfrak{C}$ with constant ε

$2^n/n^{\omega(1)}$ non-deterministic GAP-TAUT on $P/poly$

$NAND(x,y) \coloneqq NOT(AND(x,y))$
It is **universal**

Goal

Given an $NAND$ circuit C, under the two assumptions, design a $2^n/n^{\omega(1)}$ time non-deterministic algo $\mathbb{A}$, such that:

1. If C is a tautology, then $\mathbb{A}$ accepts on some guesses.
2. If $\Pr_x[C(x) = 1] \leq 1/2$, $\mathbb{A}$ rejects on all guesses.

Review: Approach of [Wil'14]
Guess-and-Verify-Equivalence

$NEXP \subset \mathfrak{C}$ implies $P/poly$ collapses to $\mathfrak{C}$.
That is, **under assumption**, the given general circuit C has an **equivalent $\mathfrak{C}$ circuit D**.
If we can find D, then we can derandomize D instead, where we have algorithms!

Problem: How to find D?

Allowed to use **non-determinism** so one can guess D.
But still have to verify D is equivalent to C, which seems **HARD**.

Solution

Well, just guess more circuits!

Review: Approach of [Wil'14]
Guess-and-Verify-Equivalence

Suppose C has m gates, let $C_1, C_2, \cdots, C_m$ be the corresponding sub-circuits.

1. C_m is the output gate.
2. $C_1, \cdots, C_n$ are inputs.

$NEXP \subset \mathfrak{C}$ implies $P/poly$ collapses to $\mathfrak{C}$.
We guess $\mathfrak{C}$ circuits $D_1, D_2, \cdots, D_m$, hoping that $D_i \equiv C_i$.

We wish to **check** $D_m \equiv C \equiv C_m$. To do this, for each $i \in \{n+1, n+2, \cdots, m\}$, suppose gate-$i$ has inputs from gate-i_1 and gate-i_2. We **verify** $\boldsymbol{NAND\left(D_{i_1}(x), D_{i_2}(x)\right) \equiv D_i(x)}$. Then run CAPP on D_m.

Problem

Checking $\boldsymbol{NAND\left(D_{i_1}(x), D_{i_2}(x)\right) = D_i(x)}$ **for all** $\boldsymbol{x}$ requires solving **SAT** for $\boldsymbol{AND_3 \circ \mathfrak{C}}$.

A Local-checkable Proof System View

Problem: the previous approach requires solving **SAT** for $AND_3 \circ \mathbb{C}$.

Let $\pi(x) :=$
$\left(D_{n+1}(x), D_{n+2}(x), \cdots, D_m(x)\right)$.

*This is a **Claimed Proof** for $C_m(x) = 1$ by **giving values at all gates.***

Intuitively, it is supposed to be the computation history of C on input x.

What is **so good** about this proof $\pi(x)$?

Local checks on $x \circ \pi(x)$

- For each $i \in \{n+1, n+2, \cdots, m\}$, $NAND\left(D_{i_1}(x), D_{i_2}(x)\right) = D_i(x)$.
- $D_m(x) = 1$.

A Local-checkable Proof System View

Let $\pi(x) := \left(D_{n+1}(x), D_{n+2}(x), \cdots, D_m(x)\right)$.
*A **Claimed Proof** for $C_m(x) = 1$ by **giving values at all gates.***

One can get $\ell = O(m) = poly(n)$ functions $F_1, F_2, \cdots, F_\ell$ on $x \circ \pi(x)$, such that

- Each F_i is an $\boldsymbol{OR}$ **of 3 bits** (or their negations) from $x \circ \pi(x)$.
- If $C(x) = 1$, on the correct guesses $D_{n+1}, \cdots, D_m$, all F_i's are satisfied by $x \circ \pi(x)$. **(Completeness)**
- If $C(x) = 0$, for all possible $\pi(x)$, **at least one** F_i is **not** satisfied by $x \circ \pi(x)$. **(Soundness)**

An Attempt

Guess circuits $D_{n+1},,\cdots,D_m$, let $\pi(x) \coloneqq \big(D_{n+1}(x), D_{n+2}(x), \cdots, D_m(x)\big)$.
Estimate $\mathbb{E}_{i\in[\ell]}\mathbb{E}_x[F_i(x \circ \pi(x))]$. *($F_i\big(x \circ \pi(x)\big) \in OR_3 \circ \mathfrak{C}$.)*
(ℓ:number of F_i's)

- If C is a **tautology**. Then on the correct guess,

$$\mathbb{E}_{i\in[\ell]}\mathbb{E}_x\big[F_i\big(x \circ \pi(x)\big)\big] = 1.$$

- If $\Pr_x[C(x) = 1] \leq 1/2$, then on all guesses,

$$\mathbb{E}_{i\in[\ell]}\mathbb{E}_x\big[F_i\big(x \circ \pi(x)\big)\big] \leq 1 - \frac{1}{2\ell}.$$

- To distinguish the above two cases, we need a CAPP algo with error $\frac{1}{2\ell} = \frac{1}{poly(n)}$.
- But we **only** assume a CAPP algo with **constant error**!

What Went Wrong?

One can get $\ell = O(m)$ functions
- Each F_i is an OR of 3 bits (or
- If $C(x) = 1$, on the correct g
  (Completeness is 1)
- If $C(x) = 0$, for all possible $\pi(x)$, at least one F_i is not satisfied by $x \circ \pi(x)$.
  (Soundness is $1 - 1/\ell$)

Proof System View

$\pi(x)$: a **claimed proof** of $C(x) = 1$

F_i : **local check** of the **verifier**

If there is a verifier who picks a random $i \in [\ell]$, and checks whether $F_i\big(x \circ \pi(x)\big) = 1$.
She detects an error **only** with probability $1/\ell$ when $C(x) = 0$.

This is an ***extremely ``bad"*** PCP!
Why not just use the PCP theorem?

Issues When Applying PCPs Directly

Recall that in the end we want to estimat

$$\mathbb{E}_{i\in[\ell]}\mathbb{E}_x\left[F_i\left(x \circ \right.\right.$$

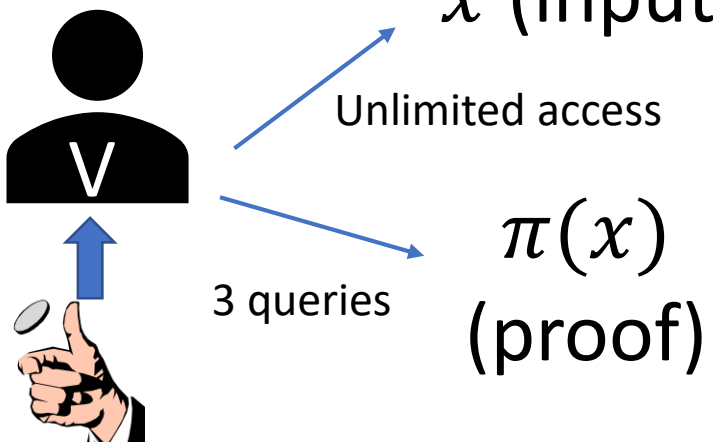
Key properties being used in **previous att**

These local checks F_i (verifier's queries positions) do not depend on the input x!

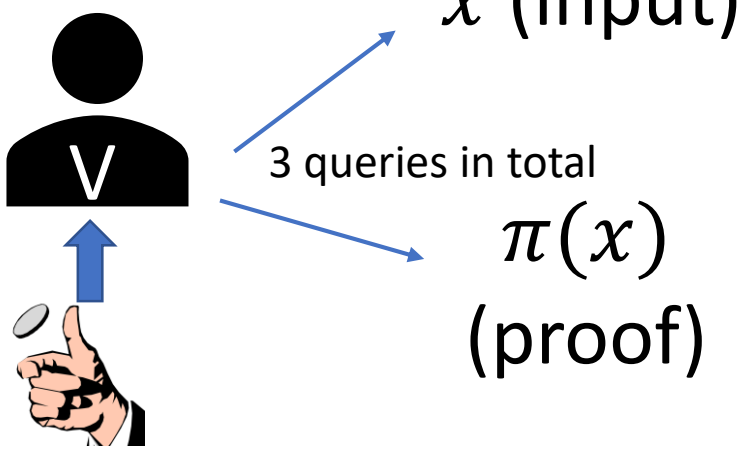
Use PCPs of Proximity!

Like PCPs but **both input and proof are given as oracles**.

PCPs



PCPs of Proximity



Now, $F_i(x \circ \pi(x))$ can depend on many bits of x.

$$F_i\big(x \circ \pi(x)\big) \in OR_3 \circ \mathfrak{C}$$

Issues When Applying PCP Directly

Therefore, we want a proof system for verifying $C(x) = 1$, such that given the random bits, verifier V queries **both input x** and **proof $\pi(x)$.**

1. If $C(x) = 1$, $\exists\ \pi(x)$, such that $V^{x \circ \pi(x)}$ **always accept**.
2. If $C(x) = 0$, $\forall\ \pi(x)$, $V^{x \circ \pi(x)}$ **rejects w.h.p**.

Counter-example?

Suppose C computes the **parity**.
Parity ***changes*** if we flip **a random bit** of x.
The verifier can't distinguish unless she queried that bit.

Solution

Give V access to an **error correcting code** of x!

Combing PCP of Proximity and ECCs

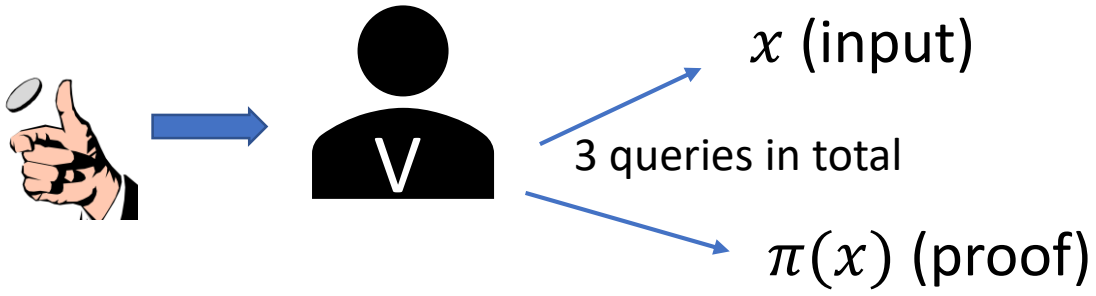
PCP of Proximity

Verifier V is given **both** the **input (x)** and
and makes 3 queries.

- $V^{x \circ \pi(x)}$ accepts w.p. 1, when $C(x) =$
- $V^{x \circ \pi(x)}$ accepts w.p. $\delta < 1$, when x m
  ($\boldsymbol{C}$ is zero in a small hamming ball around x). *(like property testing)*

How it avoids the parity counter example?

No inputs can make parity **robustly output 0**!



PCP of Proximity with ECCs

Verifier V is given both the encoded input ($ECC(x)$) and the proof $\pi(x)$ as oracles and makes 3 queries.

- $V^{ECC(x)\circ\pi(x)}$ **accepts w.p. 1**, when $C(x) = 1$;
- $V^{ECC(x)\circ\pi(x)}$ **accepts w.p.** $\boldsymbol{\delta < 1}$, when $C(x) = 0$.

Use $\boldsymbol{PCP}$ **of Proximity** for verifying $\boldsymbol{E(y) := C\big(DEC(y)\big) = 1}$, $ECC(x)$ makes $E(\cdot)$ **robustly output 0** when $C(x) = 0$!

DEC(corrupted $ECC(x)$) is still x ☺

Final Algorithm

Guess circuits $D_{n+1},,\cdots,D_m$, let $\pi(x) \coloneqq \big(D_{n+1}(x), D_{n+2}(x), \cdots, D_m(x)\big)$.

Fix ECC to be $\mathbb{F}_2$-linear. That is, $ECC(x)_i$ is a p

Suppose there is uniform parity circuit in $\mathfrak{C}$ for now (this assumption can be avoided)

Now **constant error** CAPP algo for $OR_3 \circ \mathfrak{C}$ suffices!

Estimate $\mathbb{E}_{i\in[\ell]}\mathbb{E}_x[F_i(ECC(x)\circ\pi(x))]$. *($F_i\big(x\circ\pi(x)\big) \in OR_3\circ\mathfrak{C}$.)*.

- If C is a tautology. Then on the correct guesses,

$$\mathbb{E}_{i\in[\ell]}\mathbb{E}_x\big[F_i\big(x\circ\pi(x)\big)\big] = 1.$$

- If $\Pr_x[C(x)=1] \le 1/2$, then on all guesses,

$$\mathbb{E}_{i\in[\ell]}\mathbb{E}_x\big[F_i\big(x\circ\pi(x)\big)\big] \le 1-\delta/2.$$

Future Work

NEW Building on the PCPP based approach, **[Alman Chen'19]** give a construction of Razborov-rigid matrices in P^{NP}.

Can we find non-trivial CAPP algorithms for $\boldsymbol{THR} \circ \boldsymbol{MAJ}$ or $\boldsymbol{MAJ} \circ \boldsymbol{MAJ}$ to prove circuit lower bounds for $\boldsymbol{THR} \circ \boldsymbol{THR}$?

Recall: we know exponential lower bounds for these two models! Can we ``**mine**'' some algorithms from these proofs?

Thank You