ETH-Hardness of Learning Monotone Circuits and Approximating Their Size

Bruno P. Cavalar

University of Oxford

Joint work with

Susanna F. de Rezende
Lund University

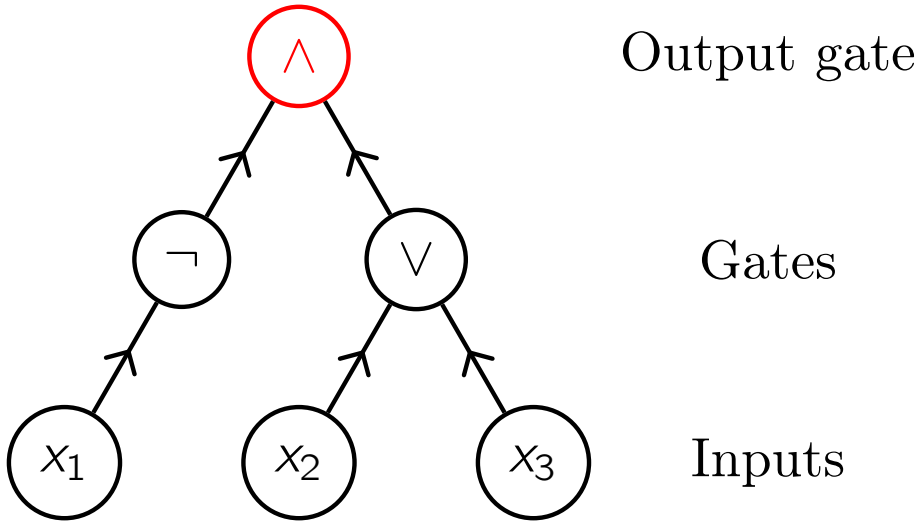
Matthew Gray
University of Oxford → SFSU

Rahul Santhanam
University of Oxford

CCC 2026

Circuit complexity

- $\mathsf{P} \neq \mathsf{NP} \equiv$ inexistence of polynomial-time algorithms for NP.



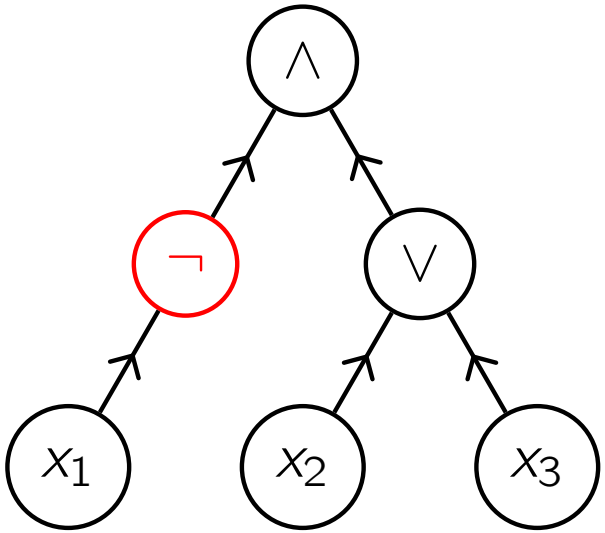
- Enough: lower bounds on the size of circuits computing a function in NP!
- MCSP: Decide if input f (as a truth-table) admits circuits of size s.
- NP-hardness of MCSP (under "natural" reductions) implies circuit lower bounds [Kabanets-Cai '00].

Meta-complexity

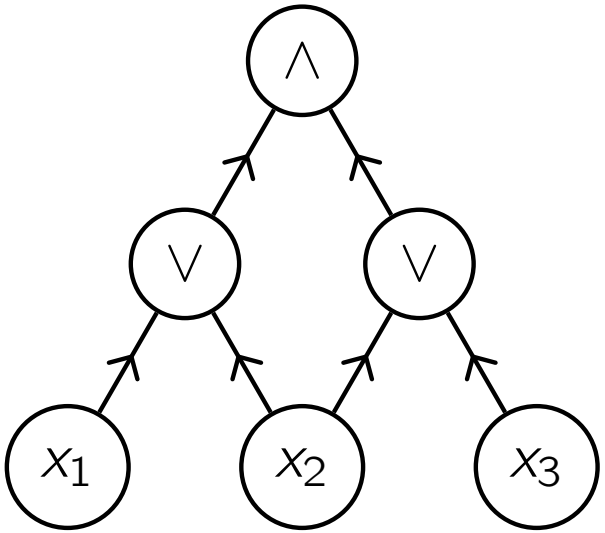
- A holy grail of meta-complexity: NP-hardness of MCSP.

- Easy with crypto, but...
- **NP-hardness of *strongly approximating* MCSP rules out Heuristica!** (Hirahara '18)
- Existing hardness of MCSP under worst-case assumptions do not give strong approximation factors
  - Except [HIR'25]: circuits with oracles.
- What about monotone circuits!?

Monotone circuits



A non-monotone circuit.



A monotone circuit.

$f : \{0,1\}^n \to \{0,1\}$ is *monotone* if $x \leqslant y \implies f(x) \leqslant f(y)$.

Relevance: cryptography, learning theory, proof complexity, hazard-free computation, communication complexity, TFNP, &c.

PAC-Learning

PAC-Learning a class of Boolean functions $\mathcal{C}$:

- Given samples $(x, f(x))$ where $x \sim \mathcal{D}$ for some distr. $\mathcal{D}$ and $f \in \mathcal{C}$
- Must output a hypothesis h that approximates f over $\mathcal{D}$.
- *Improper*: h can just be a poly-size circuit
- *Proper*: $h \in \mathcal{C}$
- *Semi-proper*: $h \in \mathcal{C}' \supseteq \mathcal{C}$, where $\mathcal{C}'$ is slightly bigger

Hardness of Learning

- A holy grail of learning theory: rule out *improper* learning under worst-case assumptions.

- Hardness of improper learning only known via crypto or average-case assumptions (Daniely 2016 onwards).

Applebaum-Barak-Xiao '08: NP-hardness of improper PAC-learning (under Karp reductions) implies PH collapse!

Hardness of Semi-Proper Learning

- Proper and mildly semi-proper learning have been shown to be NP-hard (for learning e.g., DNFs or halfspaces).
- What about monotone circuits!?
  - Only known under crypto assumptions
  - Or unconditionally in SQ-learning
- Parallel with MCSP. Indeed, $\approx$ search version thereof...

Occam Learning or Partial MCSP

Partial Monotone Minimum Circuit-Formula Size Problem (Partial-Monotone-MFCSP)

- Input: m pairs (x_i, b_i) with $x_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$.
- Input length $N = \Theta(mn)$.

Is there a monotone formula F of size n consistent with all examples?

OR

Must every monotone circuit of size n^c err on approx. half of the pairs?

- Shown to be NP-hard (Ilango-Loff-Oliveira'20), but no strong hardness of approximation (argument reduces from Set-Cover).

Result 1: Hardness for Partial MCSP

Recall rETH: takes time $2^{\Omega(n)}$ to solve SAT with randomised algorithms.

Theorem

Under rETH, it takes time $N^{\Omega(\log N)}$ to solve Partial-Monotone-MFCSP with $m = n^c \log n$ examples.

In particular, partial monotone circuit and formula size cannot be $N^{1-\varepsilon}$-multiplicatively approximated in $N^{o(\log N)}$ time.

Again: would rule out Heuristica if it were shown to be NP-hard for general circuits and for MCSP.

Result 2: Hardness for PAC-Learning

Theorem

Under rETH, it takes time $N^{\Omega(\log N)}$ to PAC-learn monotone formulas of size n by monotone circuits of size $n^{(\log n)^{1-\varepsilon}}$.

- Avoids ABX'08 barrier of PH collapse!
- Hardness of improper learning with even $n^{\Omega(1)}$ gap was previously unknown!
- Hirahara '22: Learning linear programs of size s_1 by linear programs of size $s_1 n^{o(1)}$. Smaller gap but under NP-hardness.
- Also: gap between learning $(\log n)^{\log\log n}$-juntas by n^c-size circuits.

A unifying problem: Monotone CGL

Monotone Circuit-Formula Gap Learning (mCFGL)

Input: A poly-size circuit D that samples pairs $(x, b) \in \{0,1\}^n \times \{0,1\}$.

Is there a monotone formula of size s_1 consistent with the support of D?

OR

Must every monotone circuit of size s_2 err with probability $1/2 - \varepsilon$?

Theorem

Under rETH, mCFGL takes time $N^{\Omega(\log N)}$ *for any* $s_1 \geqslant n$ *and* $s_2 \leqslant n^{\delta \log n}$.

A chain of connections

- Strong inapproximability of Resolution proof *size* (Atserias-Müller 2020) from NP-hardness (reducing from SAT)
- Tight relationship between Resolution *width/depth* and monotone circuit/formula complexity of a related Boolean function (lifting theorems, GGKS20).

Resolution

- A resolution refutation of a formula ϕ is a sequence of clauses $D_1, \ldots, D_s$ where $D_s = \bot$ and each clause D_i is
  1. Either one of the clauses of ϕ, or
  2. A *weakening* of a previous clause, or
  3. Obtained from two previous clauses by a Resolution rule:

$$\frac{A \vee x \qquad B \vee \neg x}{A \vee B} \text{ (Res)}$$

- *Size:* s
- *Width*: maximum number of literals appearing clauses $D_1, \ldots, D_s$
- *Depth*: Length of longest path in the proof DAG

Atserias-Müller: automatability

- Strong inapproximability of Resolution proof *size* (Atserias-Müller 2020) from NP-hardness
- Given a formula F, they construct $\mathsf{Ref}(F)$ such that:
  1. If F is sat, then $\mathsf{Ref}(F)$ admits a poly-size Res proof;
  2. If F is unsat, then any Res proof of $\mathsf{Ref}(F)$ must have size $2^{n^{\Omega(1)}}$.
- Informally, the formula encodes the statement "F has a small Resolution refutation."

Lifting to monotone circuit size

- Let ϕ be an unsatisfiable formula with n variables.
- Lifting associates with ϕ a function f_ϕ with monotone circuit complexity $n^{\Theta(\mathsf{ResWidth}(\phi))}$ and formula complexity $n^{\Theta(\mathsf{ResDepth}(\phi))}$
- *Idea:* Apply it to $\mathsf{Ref}(F)$!
  - Problem: their result is about size, not width.
  - Resolution size upper bounds the complexity of another function (feasible interpolation), but no converse known.

A new Ref formula

1. $\mathsf{Ref}^*(F)$ has $2^{O(\sqrt{n})}$ variables and $2^{O(\sqrt{n})}$ clauses.
2. $\mathsf{Ref}^*(F)$ admits Res proofs of depth at most $O(\sqrt{n})$ when F is satisfiable;
3. $\mathsf{Ref}^*(F)$ requires Res proofs of width at least $2^{\Omega(\sqrt{n})}$ when F is unsatisfiable.

Tradeoff: increase the *size* of the formula, but improve the width/depth upper bound. Reduction is now subexponential. Resulting function:

1. $f_{\mathsf{Ref}^*(F)}$ has monotone complexity $n^{\Theta(\sqrt{n})}$ when F is satisfiable;
2. $f_{\mathsf{Ref}^*(F)}$ has monotone complexity $2^{\Omega(n)}$ when F is unsatisfiable.

We just need to *construct* a sampler for the instances of f given F!

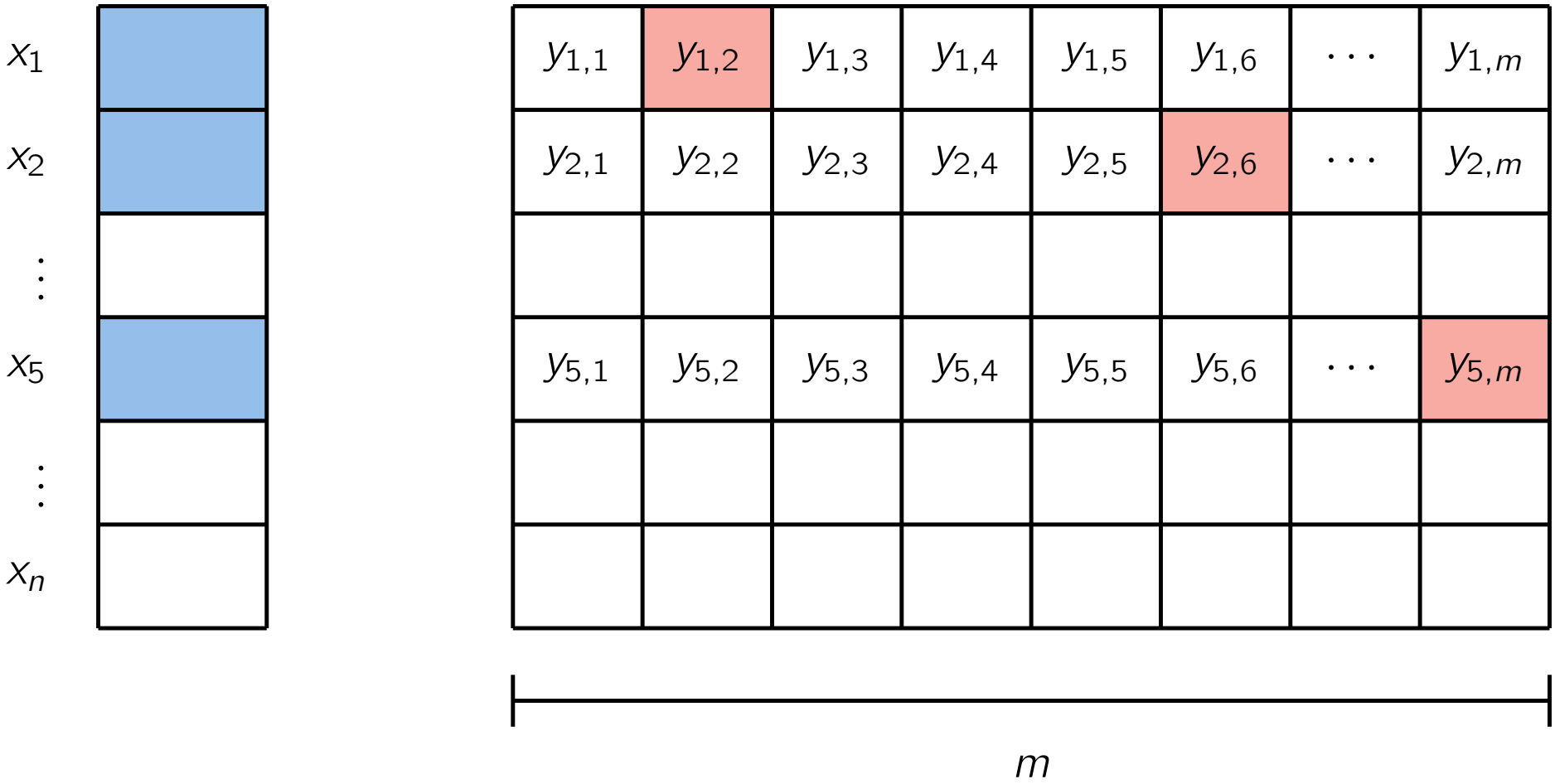
Lifted functions

Suppose ϕ has width 3.

- Let $g = \mathsf{Ind}_m : [m] \times \{0,1\}^m \to \{0,1\}$ be such that $g(x,y) = y[x]$.
- The lifted function $f = f_{\phi,m}$ has one variable for each pair (C, α), where C is a clause of ϕ and $\alpha \in [m]^3$.

Lifted functions: an example

$C = x_1 \vee x_2 \vee x_5$ $\alpha = (2, 6, m)$ $(C, \alpha) = y_{1,2} \vee y_{2,6} \vee y_{5,m}$



Number of input bits: $\#\text{clauses} \cdot m^3$.

Lifted function behaviour

1. **Accept (Output 1):** there exists $x : [n] \to [m]$ such that the input bit corresponding to $(C, x^{\mathsf{vars}(C)})$ is set to 1 for every clause C of ϕ, and everything else is set to 0;
2. **Reject (Output 0):** there exists $y \in \{0,1\}^{mn}$ such that all input bits associated with clauses (C, α) satisfied by y are set to 1, and everything else is set to 0.

Monotone as we cannot make an unsatisfiable system satisfiable by adding more clauses!

Karchmer-Wigderson game of $f_{\phi,m}$ finds a falsified clause of $\phi \circ g$.

Reduction to Computational Gap Learning

- Given ϕ, m, let the circuit $D := D^{\phi,m}$ receive as input a seed of $mn + 1$ bits.
- If the first bit is 0, interpret the remaining bits as $y \in \{0,1\}^{mn}$ and add all clauses (C, α) satisfied by y.
- Otherwise, interpret the remaining bits as $x \in [m]^n$ and add the clauses $(C, x^{\mathsf{vars}(C)})$.

Very simple to construct the circuit given ϕ!

Wrap-up of the algorithm

Input: 3CNF F on n variables.

- Get F, construct $\phi = \mathsf{Ref}(F)$.
  Takes time $N = 2^{O(\sqrt{n})}$.
- Construct $D := D^{\phi,m}$ for some appropriate $m = \mathsf{poly}(n)$.
  Takes time $\mathsf{poly}(N)$.
- Call mCFGL(D).

Theorem (Lifting – de Rezende-Vinyals '25)

- *If F is satisfiable, then there is a monotone formula L of size $\leqslant 2^{\sqrt{n}}$ s.t. $L(x) = b$ for every $(x, b) \in \mathsf{supp}(D)$;*
- *if F is unsatisfiable, then every monotone circuit K of size $\leqslant 2^n$ satisfies $\mathbb{P}_{(x,b)\leftarrow D}[K(x) = b] < \frac{1}{2} + \gamma$, where $\gamma \approx m^{-1}$.*

- Lifting of [dRV25] allows for smaller m, trading on the lower bound.
- Lifting is typically stated with $\gamma = 1/2$, but we can take $\gamma \approx m^{-1}$.

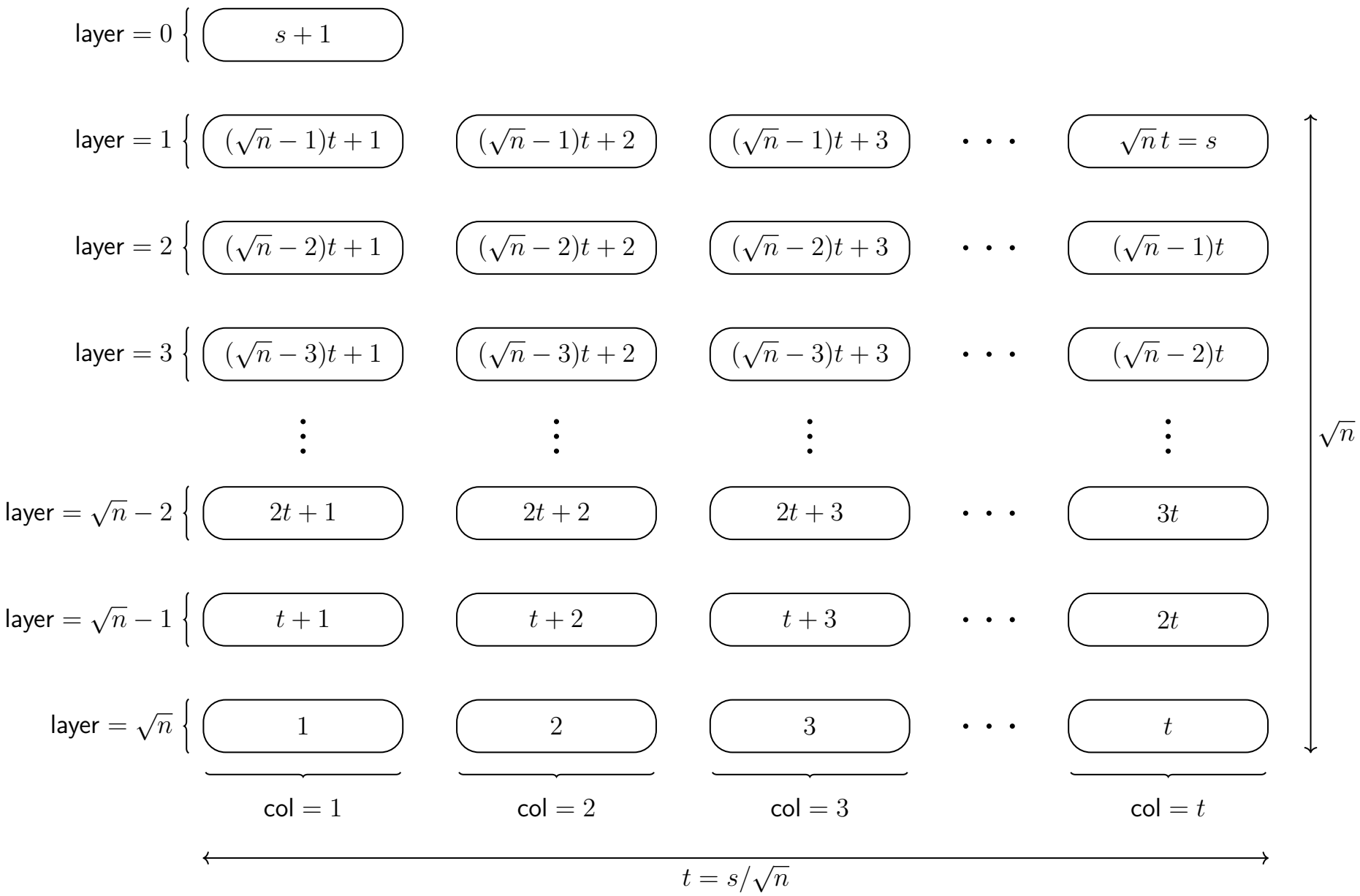
A new Ref formula: upper bound ideas

Prover-Adversary game: a Refuter with a satisfying assignment x^* to F needs to find a contradiction in the supposed refutation of F, by querying the variables of $\mathsf{Ref}(F)$.

Prover/Refuter strategy: Begin at $\bot$ and reach axiom of F.
Invariant: current clause is falsified by x^*.

- Require variables to be resolved in order
  - Given a clause, we know exactly which ones may have resolved to it
- Require each Resolution step to resolve $\sqrt{n}$ variables
  - So the depth of the proof encoded is $\sqrt{n}$
- Keep "summary" variables that encode information about sequences of $\sqrt{n}$ variables
  - So only need to query $O(1)$ variables at each "layer"

Layers and columns of $\mathsf{Ref}^*(F)$



Open problems

- Get NP-hardness!
  - No known barriers;
  - Lifting block-width to monotone circuit size?
    - Explicitly circumvented by [GKMP20] (ask Ian Mertz)
- Hardness of membership query learning?
  - Construct lifted function with monotone-vs-nonmonotone gap, and moreover construct the nonmonotone circuit.
- What about monotone MCSP?
  - Challenge: truth-table is too big!

Thanks!