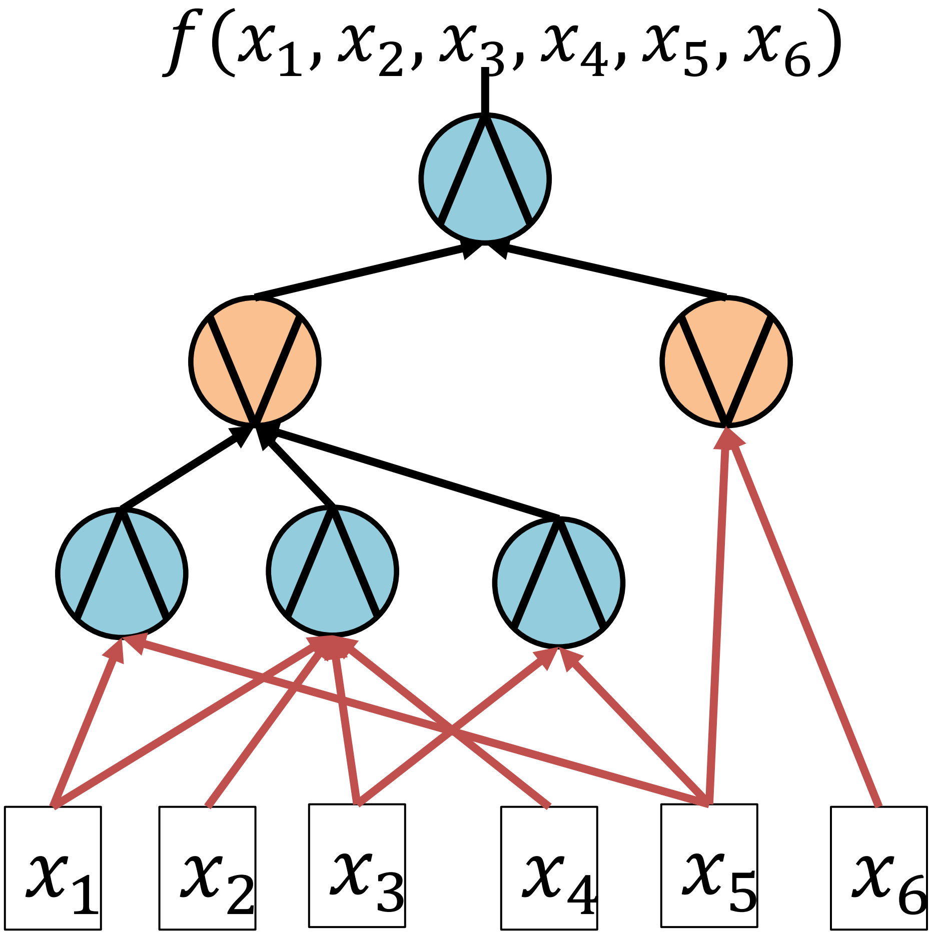
The Quantum Query Complexity of Read-Many Boolean Formulas

Andrew Childs[1], Shelby Kimmel[2], Robin Kothari[1]

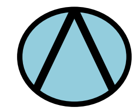
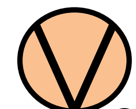
1. University of Waterloo

2. MIT

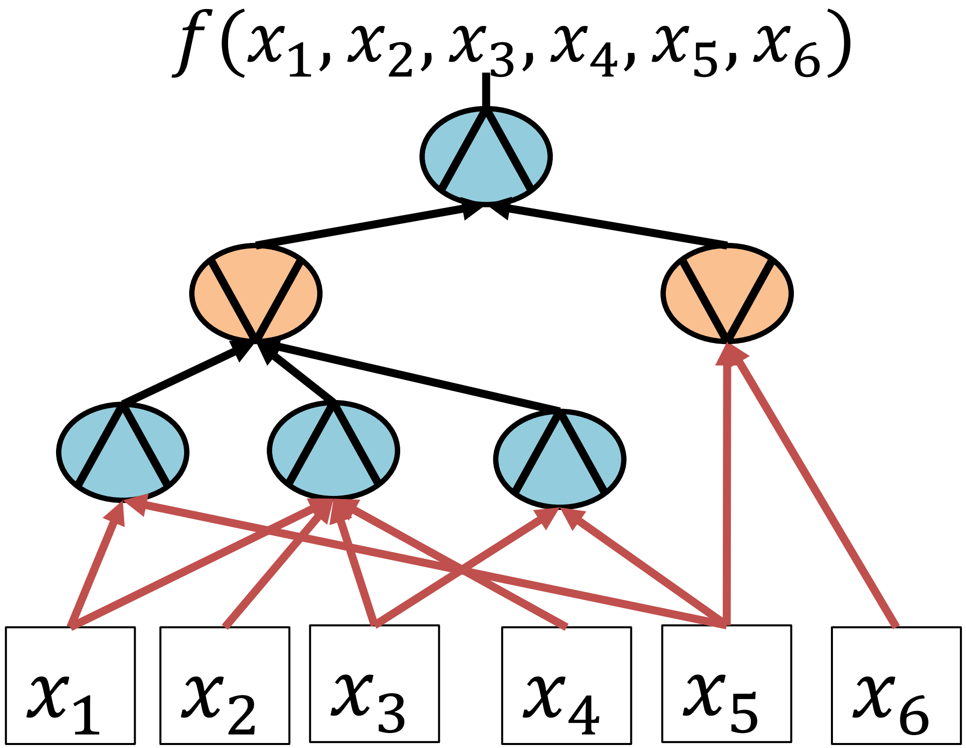
General Boolean Formula



Boolean: $x_i \in \{0,1\}$

- Unbounded Fan-in
  - AND 
  - OR 
- No fanout of gates
- Fanout of inputs OK
- n =# of inputs (6)
- S =# of input edges (formula size) (10)
- G =# of gates (6)

Want to Evaluate with Quantum Computer



- **Quantum Query Complexity**

Number of queries to the inputs x_i needed to evaluate the formula (with bounded error) with a quantum computer. Denoted by $Q(f)$.

New Bounds on Formula Quantum Query Complexity

Upper Bound: Algorithm to evaluate any Boolean formula w/ quantum query complexity

$$\boxed{\boldsymbol{O(min\{n, S^{1/2}, n^{1/2}G^{1/4}\})}}$$

Lower Bound: Given values for n, S, and G, $\exists$ a formula with n inputs, size $\leq S$, and gate count $\leq G$, with query complexity

$$\boxed{\boldsymbol{\Omega(min\{n, S^{1/2}, n^{1/2}G^{1/4}\})}}$$

n =# of inputs, S =# of input edges, G =# of gates

Application 1: Classical Formula Complexity

$$Q(f) = \mathrm{O}(n^{1/2}G^{1/4})$$

Upper bound on Query Complexity in terms of number of gates in the formula

$$G(f) = \Omega(n^{-2}Q^4)$$

Lower bound on the number of gates in a formula in terms of the query complexity.

Result:

- PARITY requires n^2gates (previous best result: $S = \Omega(n^2)$) [Khrapchenko, ’71]

Application 2: Classical Formula Complexity

Can circuits be efficiently expressed as formulas?

Result:

- Given $\epsilon > 0$, there exists a constant depth circuit of size $O(n)$, such that a formula representation requires $O(n^{2-\epsilon})$ ***gates***

Previous Result [Jukna '12, Nechiporuk, '66]

- There exists a constant depth circuit of size $O(n)$, such that a formula representation requires ***formula size*** $O\left(n^{2-o(1)}\right)$.

Proof Idea:

- Construct a constant depth, size $O(n)$ circuit with $Q(f)$ close to linear. Apply previous technique for lower bounding gates.

New Bounds on Formula Quantum Query Complexity

Upper Bound: Algorithm to evaluate any Boolean formula w/ quantum query complexity

$$\boxed{\mathbf{O}(\boldsymbol{min}\{\boldsymbol{n}, \boldsymbol{S}^{1/2}, \boldsymbol{n}^{1/2}\boldsymbol{G}^{1/4}\})}$$

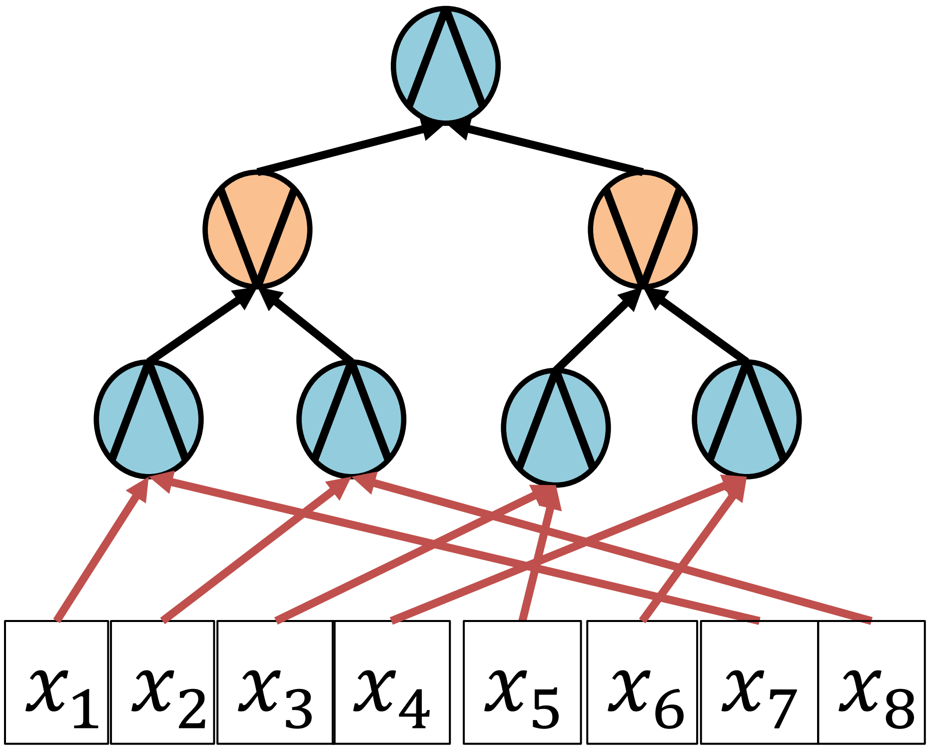
Lower Bound: Given values for n, S, and G, $\exists$ a formula with n inputs, size $\leq S$, and gate count $\leq G$, with query complexity

$$\boxed{\boldsymbol{\Omega}(\boldsymbol{min}\{\boldsymbol{n}, \boldsymbol{S}^{1/2}, \boldsymbol{n}^{1/2}\boldsymbol{G}^{1/4}\})}$$

n =# of inputs, S =# of input edges, G =# of gates

General vs. Read-Once Formulas

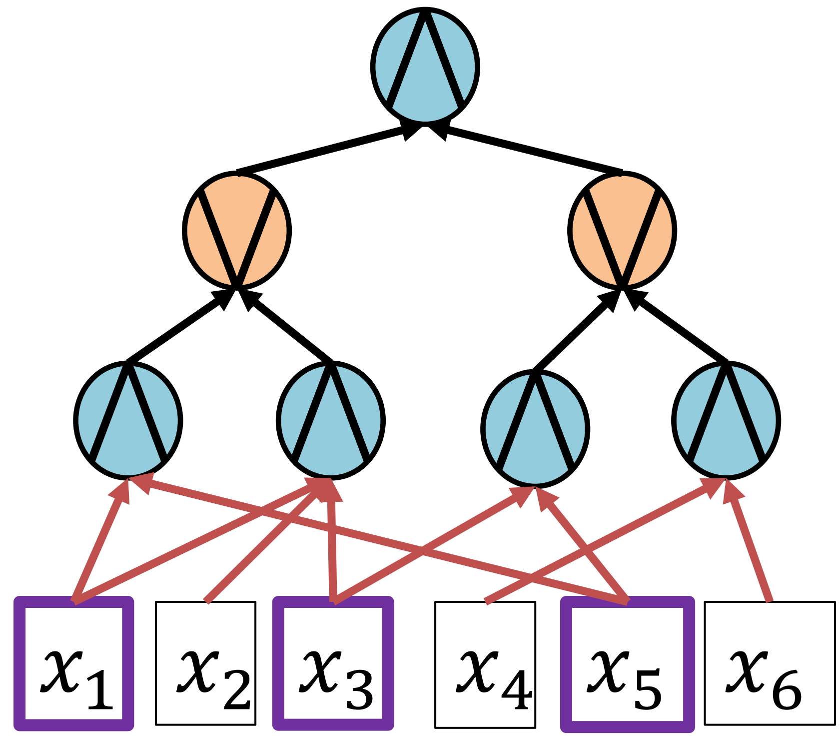
Read-Once



$Q(f) = \Theta(\sqrt{S})$ [Reichardt, '11]

S =# of input edges (formula size)

"Read-many" = general

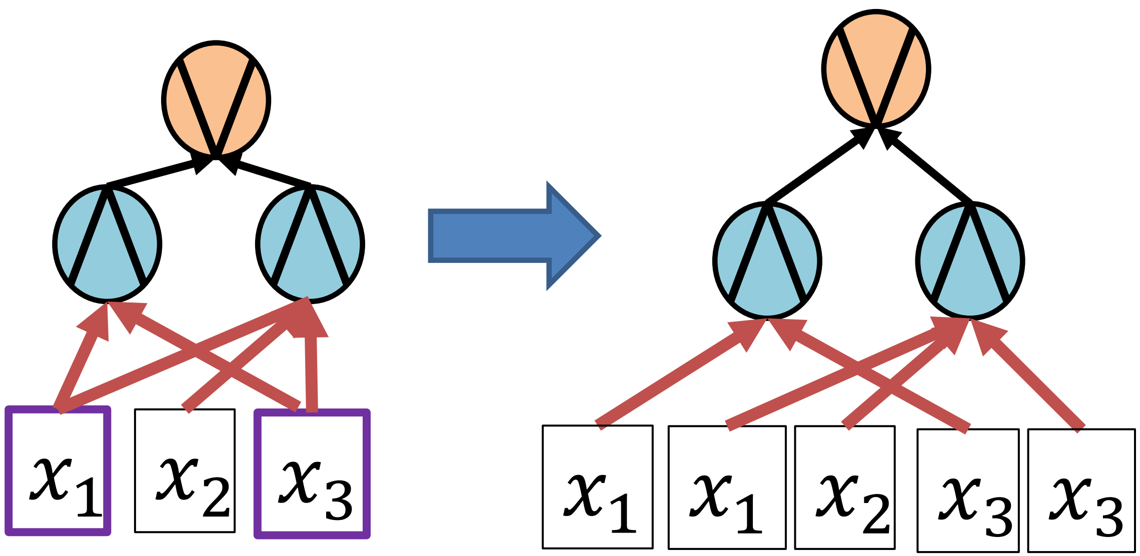


$Q(f) = O(\sqrt{S})$

Big Idea: Algorithm

$$Q(f) = O(\min\{n, S^{1/2}, n^{1/2}G^{1/4}\})$$

- n – Query all inputs (trivial)
- $S^{1/2}$ – convert to read-once

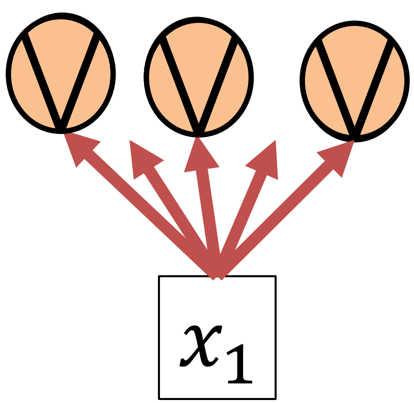


n =# of inputs, S =# of input edges, G =# of gates

Big Idea: Algorithm

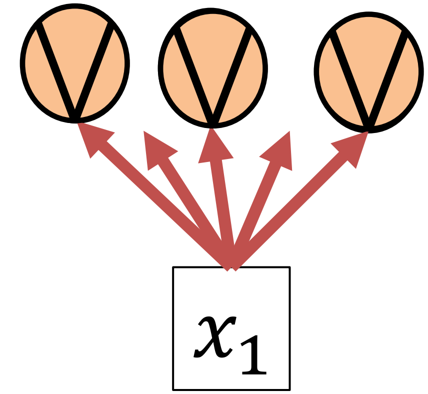
$$\mathrm{Q(f)} = \mathrm{O}\left(\min\left\{n, S^{1/2}, \boxed{n^{1/2}G^{1/4}}\right\}\right)$$

Strategy: Deal with "high degree" (deg $> G^{1/2}$) inputs separately before expanding.



If $x_1 = 1$, learn **a lot** (output of $> G^{1/2}$ OR gates)

Big Idea: Algorithm



$$Q(f) = O(\min\{n, S^{1/2}, \boxed{n^{1/2}G^{1/4}}\})$$

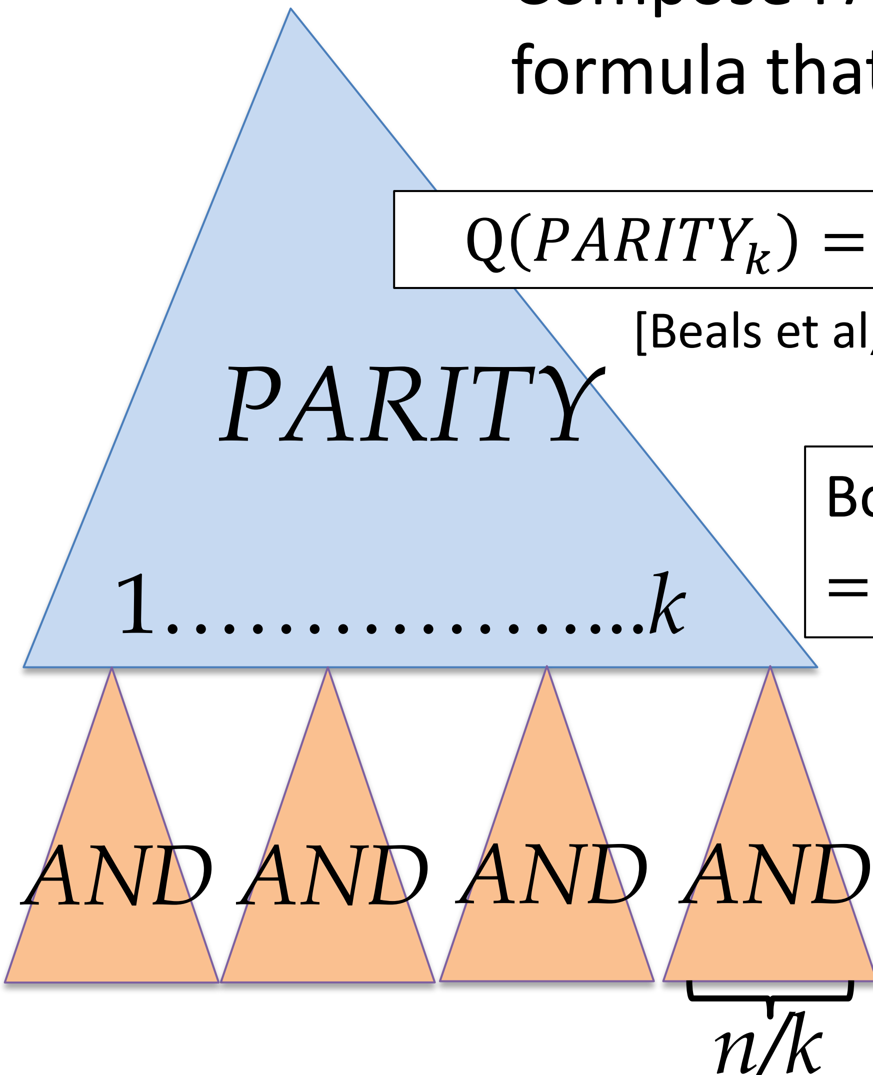
<u>Plan</u>

1. Learn all high deg (deg $> G^{1/2}$) nodes by Grover search:
   - Search among high deg "OR" inputs for value 1. Each time kills $G^{1/2}$ gates, so can't have too many rounds.
   - Repeat high deg "AND" inputs (search for 0).
2. Now S is small because no input is high degree
   - Expand (by repeating inputs) to Read-Once

Parts 1 & 2 each use $O\left(n^{1/2}G^{1/4}\right)$ queries.

Big Idea: Lower Bound

Compose PARITY and AND to get new formula that needs large query complexity



$Q(PARITY_k) = \Omega(k)$

[Beals et al, '98]

$Q\left(AND_{n/k}\right) = \Omega(\sqrt{n/k})$

Bound on composed: [Reichardt, '11]

$= \Omega(k) \times \Omega(\sqrt{n/k}) = \Omega(\sqrt{n \times k})$

By adjusting k, can get a formula w/ lower bound that matches $\Omega(\min\{n, S^{1/2}, n^{1/2} G^{1/4}\})$

Extensions: Constant Depth Formulas

What if also know depth $= d$?

Upper bound still holds, but need to create new worst case functions for lower bound

Depth	**Function Composed with AND**	**Upper/Lower Bounds on Q**
Not constant	PARITY	Tight
$d \geq 3$	ONTO	Tight up to log factors
$d = 2$	2-Distinctness	$O(n^{.75}), \Omega(n^{.555})$

Application 3: Boolean Matrix Product Verification

Boolean Matrix Product Verification
A, B, C are $n \times n$ Boolean matrices
$A \times B = C$?
($\vee \sim +$, $\wedge \sim \times$)

$Q(BMPV) = O(n^{3/2}), \Omega(n)$ [Buhrman and Spalek, '06]

Result:
- $Q(BMPV) = \Omega(n^{1.0555})$

Summary and Open Problems

Accomplished:

- Optimal algorithm for Boolean formulas (in terms of n, S, and G).
- New technique for lower bounding # of gates for functions.
- New result on gates needed to convert from circuit to formula
- Improved lower bound for BMPV

To Do:

- Tighter bounds on formulas w/ constant depth
- Create new algorithms with quantum subroutines

The Quantum Query Complexity of Read-Many Formulas

Andrew Childs, Shelby Kimmel, Robin Kothari

arXiv:1112.0548

Quantum Boolean Formula Problems

Triangle Problem

n vertex graph (n^2edges) $\rightarrow$ triangle?

[Belovs, '11]

k-distinctness

n integers $\rightarrow \geq k$ of them equal?

[Belovs , '12]

Boolean Matrix Product Problems

[Jeffrey, Kothari, Magniez, '12]

Outline

1. Intro to Boolean formulas and quantum query complexity
2. Optimal algorithm for Boolean formulas
3. Applications
   a) Boolean Matrix Product Verification
   b) Classical Formula Complexity

Applications: Boolean Matrix Product Verification

Boolean Matrix Product Verification

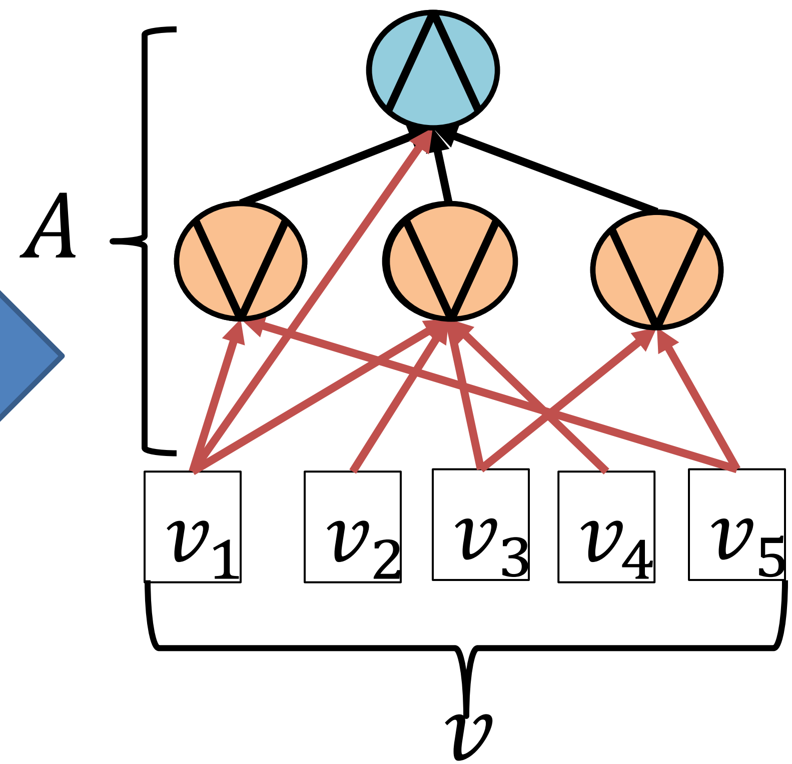
A, B, C are $n \times n$ Boolean matrices → $A \times B = C$? $(\vee \sim +,\ \wedge \sim \times)$

$Q(BMPV) = O(n^{3/2}), \Omega(n)$ [Buhrman and Spalek, ‘06]

Boolean ***Vector*** Product Verification

$$\begin{bmatrix} a_{11} & a_{12} & \cdots \\ a_{21} & \ddots & \vdots \\ \vdots & \cdots & \ddots \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \end{bmatrix} \stackrel{?}{=} \begin{bmatrix} 1 \\ 1 \\ \vdots \end{bmatrix}$$

A known $\qquad$ v unknown



Query Complexity = $\Omega(n^{.555})$

Applications: Boolean Matrix Product Verification

$$\begin{bmatrix} a_{11} & a_{12} & \cdots \\ a_{21} & \ddots & \vdots \\ \vdots & \cdots & \ddots \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & \cdots \\ v_{21} & \ddots & \vdots \\ \vdots & \cdots & \ddots \end{bmatrix} \stackrel{?}{=} \mathbb{I} \leq$$

Boolean Matrix Product Verification (all matrices unknown)

A known $\quad V$ unknown

$$f = (AND) \circ (Boolean\,Vector\;Product\;Verification)$$

$$Q(f) = \Omega(Q(AND)) \times \Omega(Q(BVPV))$$ [Reichardt, '11]

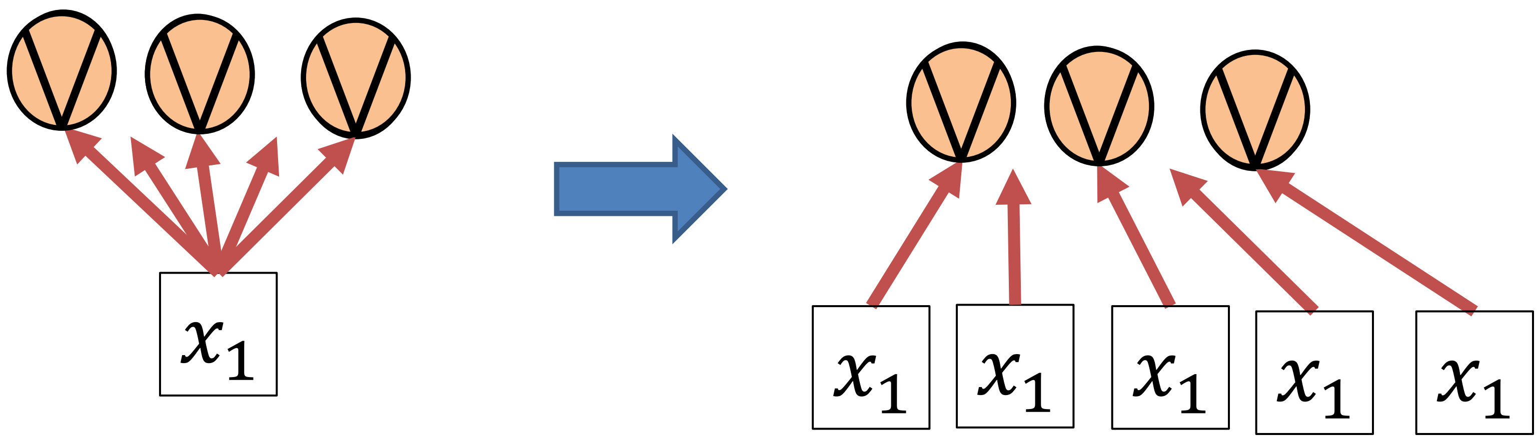
Lower bound = $\Omega(n^{1.0555})$

Previous best lower bound=$\Omega(n)$

Big Idea: Algorithm

$$Q(f) = O(\min\{n, S^{1/2}, n^{1/2}G^{1/4}\})$$

- n – Query all inputs (trivial)
- $S^{1/2}$ – convert to read-once



n =# of inputs, S =# of input edges, G =# of gates