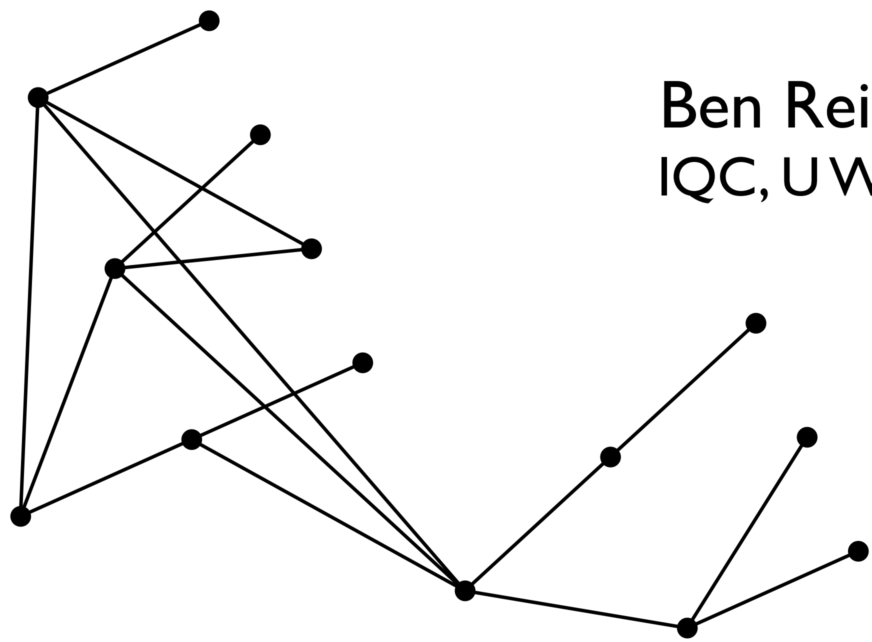
Quantum algorithms based on span programs



Ben Reichardt
IQC, U Waterloo

[arXiv:0904.2759]

Model:	**Quantum algorithms**	$\approx$	**Span programs**	[KW '93]
Complexity measure:	Black-box query complexity		Witness size	[RŠ '08]

- Optimal span program witness size is characterized by an SDP (Adv±)

  ⇒ quantum query complexity characterized by the same SDP

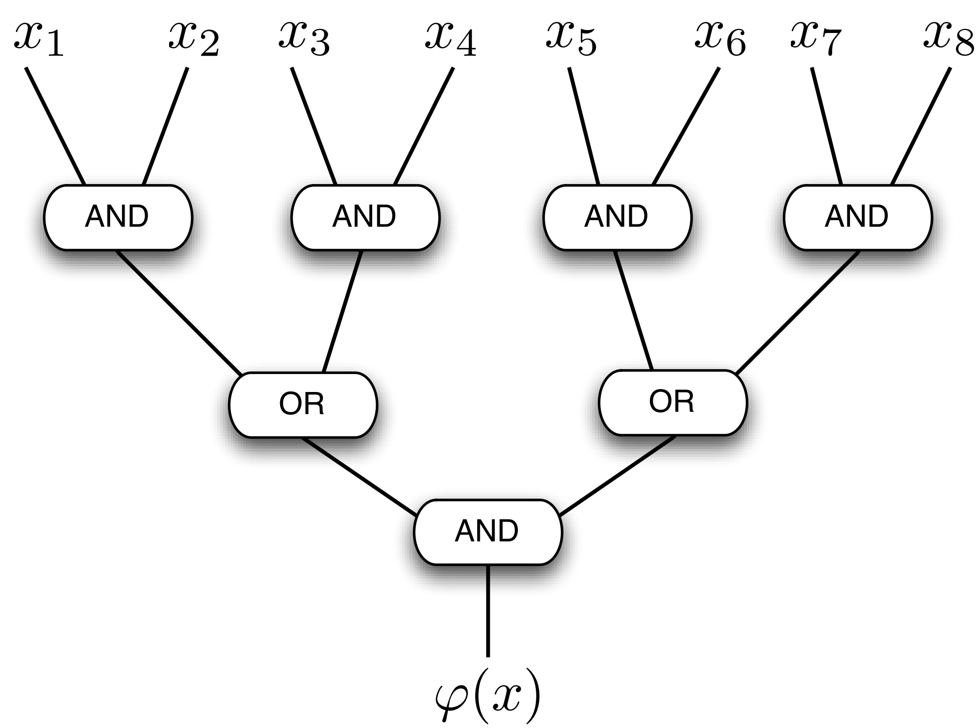
- Span programs compose easily

  ⇒ quantum algorithms compose easily

  ⇒ optimal quantum algorithms for many formula-evaluation problems

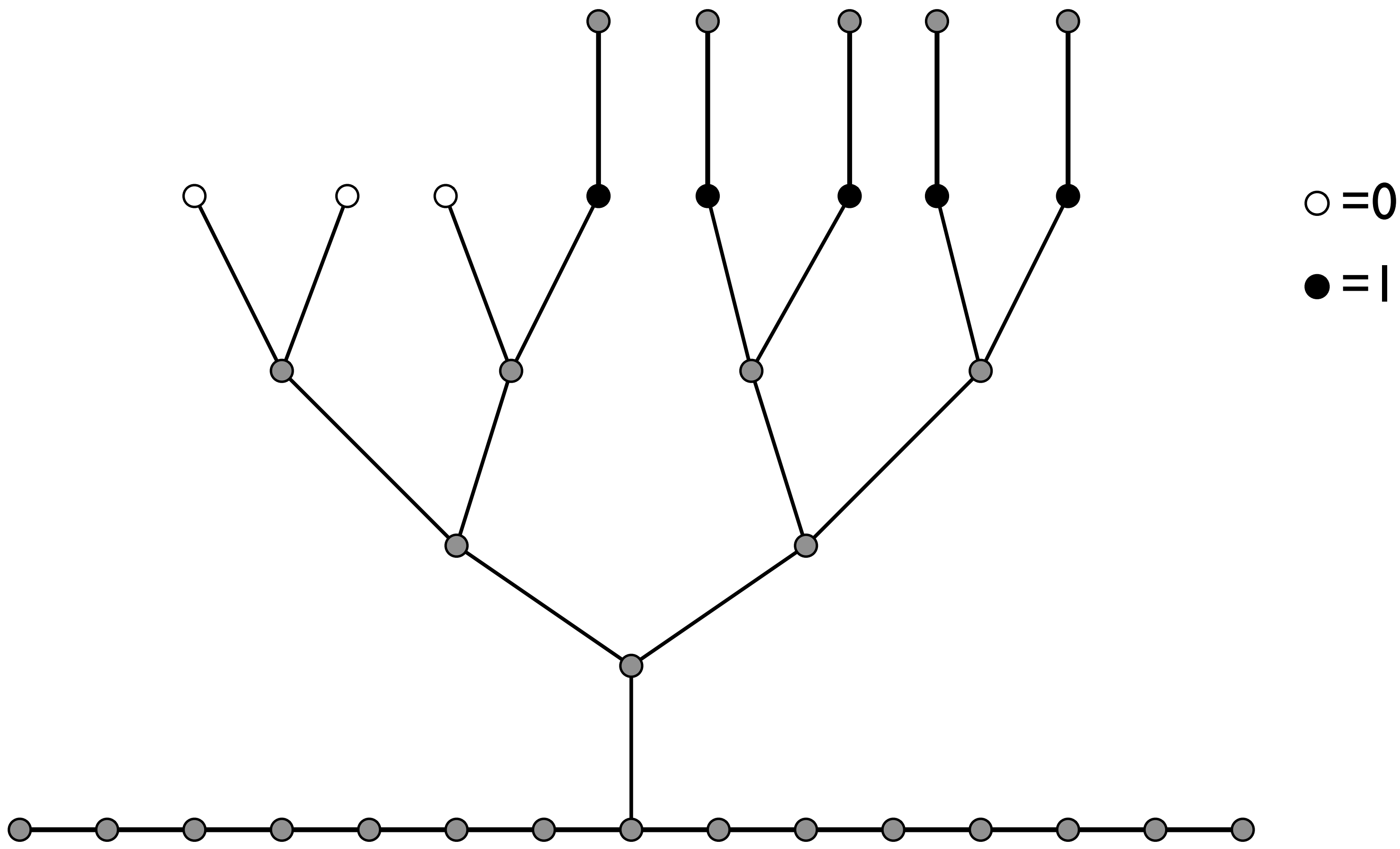
Farhi, Goldstone, Gutmann '07 algorithm

- **Theorem** ([FGG '07]): A balanced binary AND-OR formula can be evaluated in time $N^{½+o(1)}$.



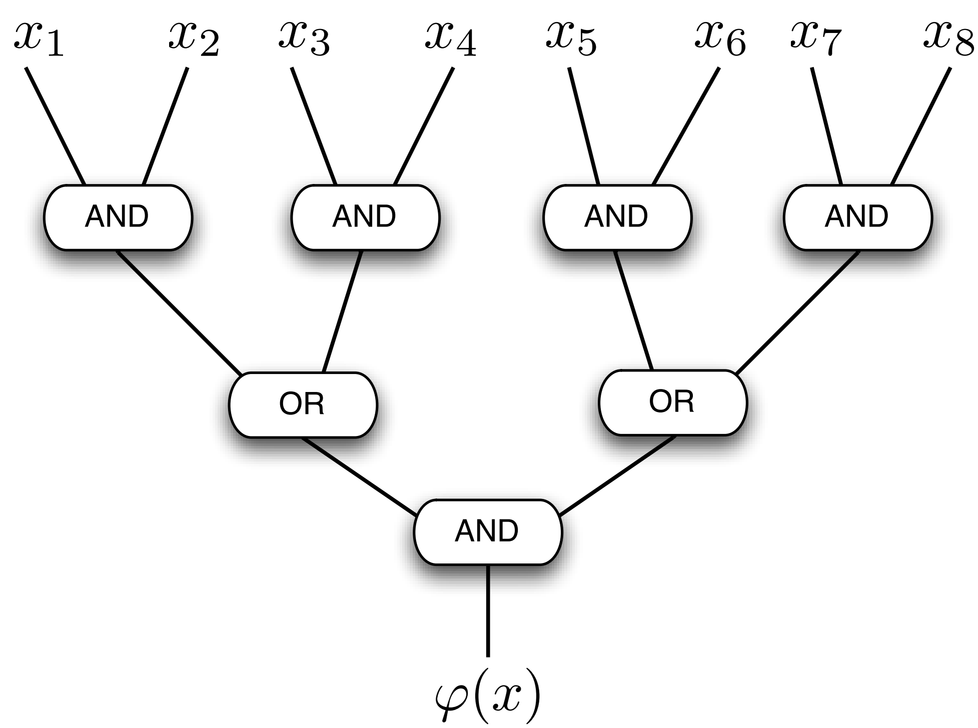
Farhi, Goldstone, Gutmann '07 algorithm

- **Theorem** ([FGG '07]): A balanced binary AND-OR formula can be evaluated in time $N^{½+o(1)}$.
  - Convert formula to a tree, and attach a line to the root
  - Add edges above leaf nodes evaluating to one

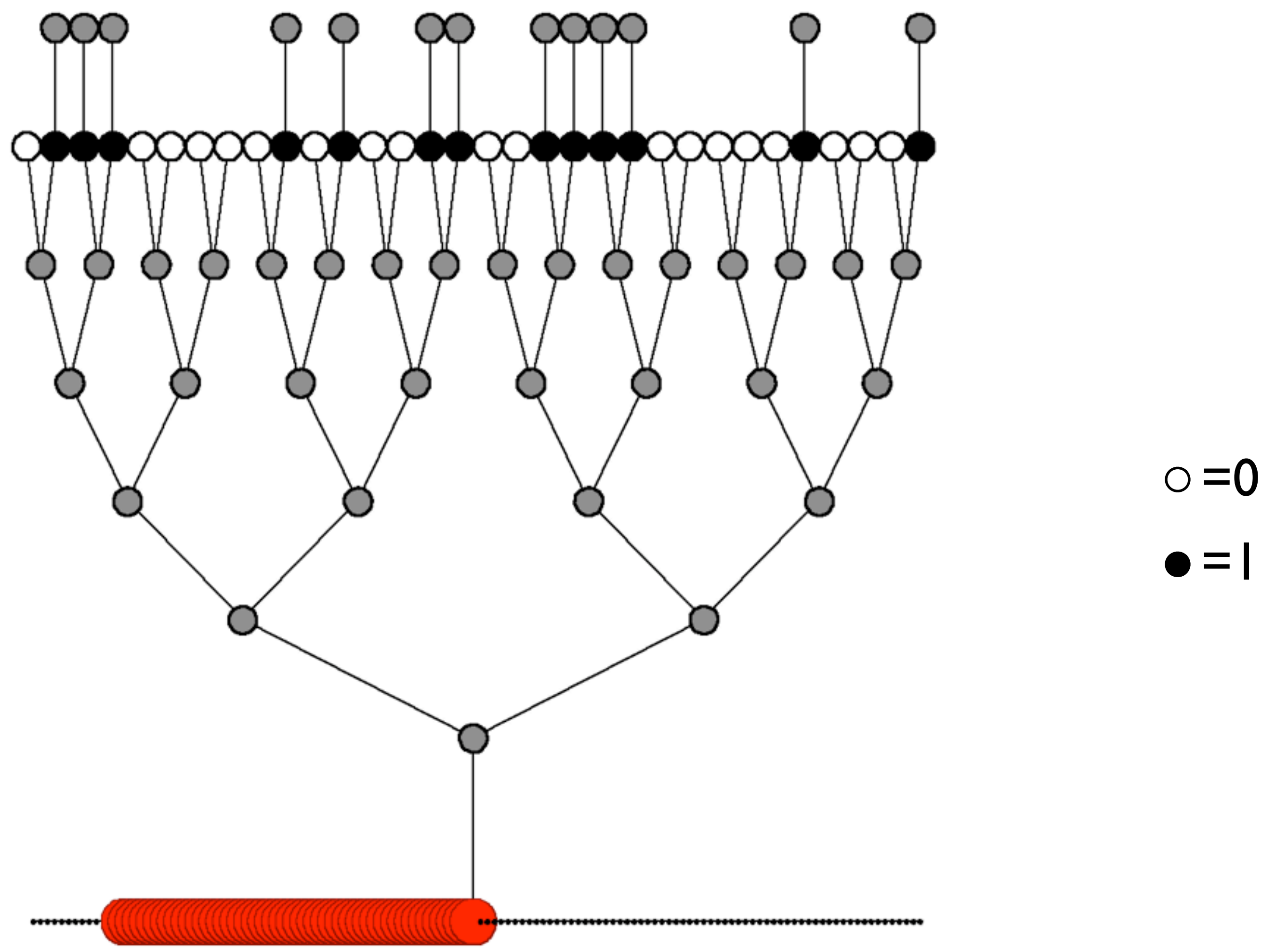


Farhi, Goldstone, Gutmann '07 algorithm

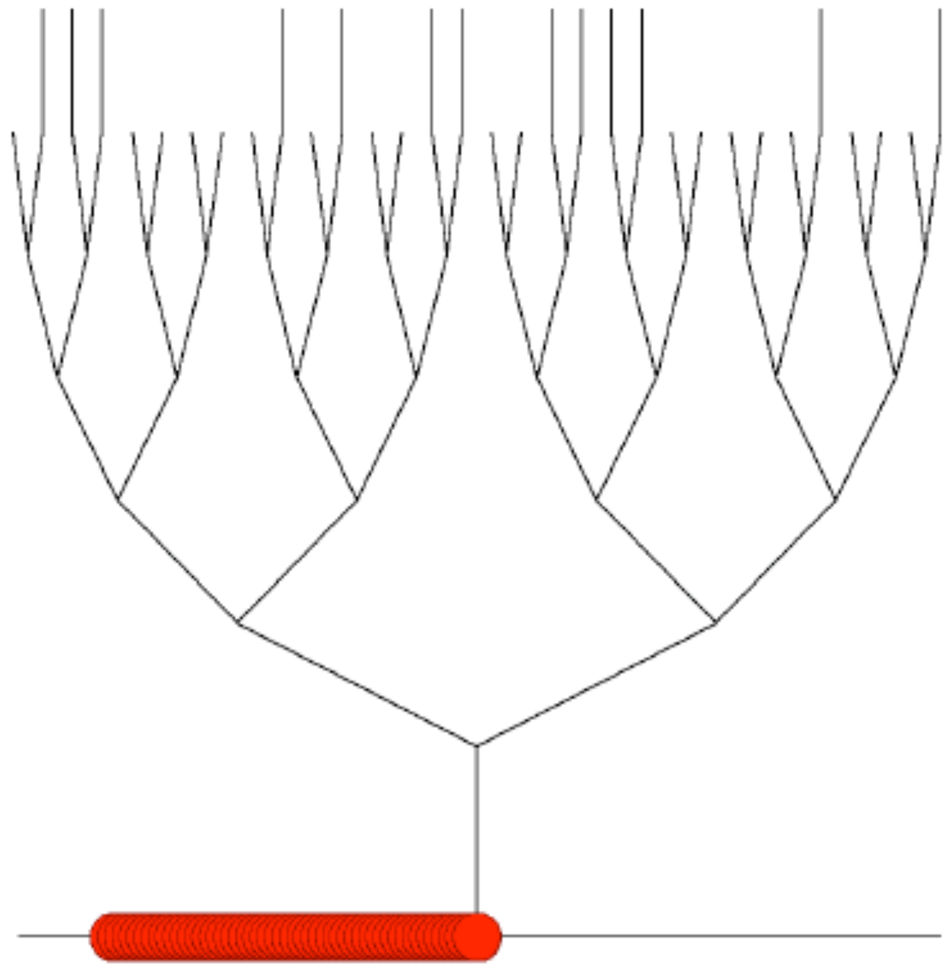
- **Theorem** ([FGG '07]): A balanced binary AND-OR formula can be evaluated in time $N^{½+o(1)}$.
  - Convert formula to a tree, and attach a line to the root
  - Add edges above leaf nodes evaluating to one



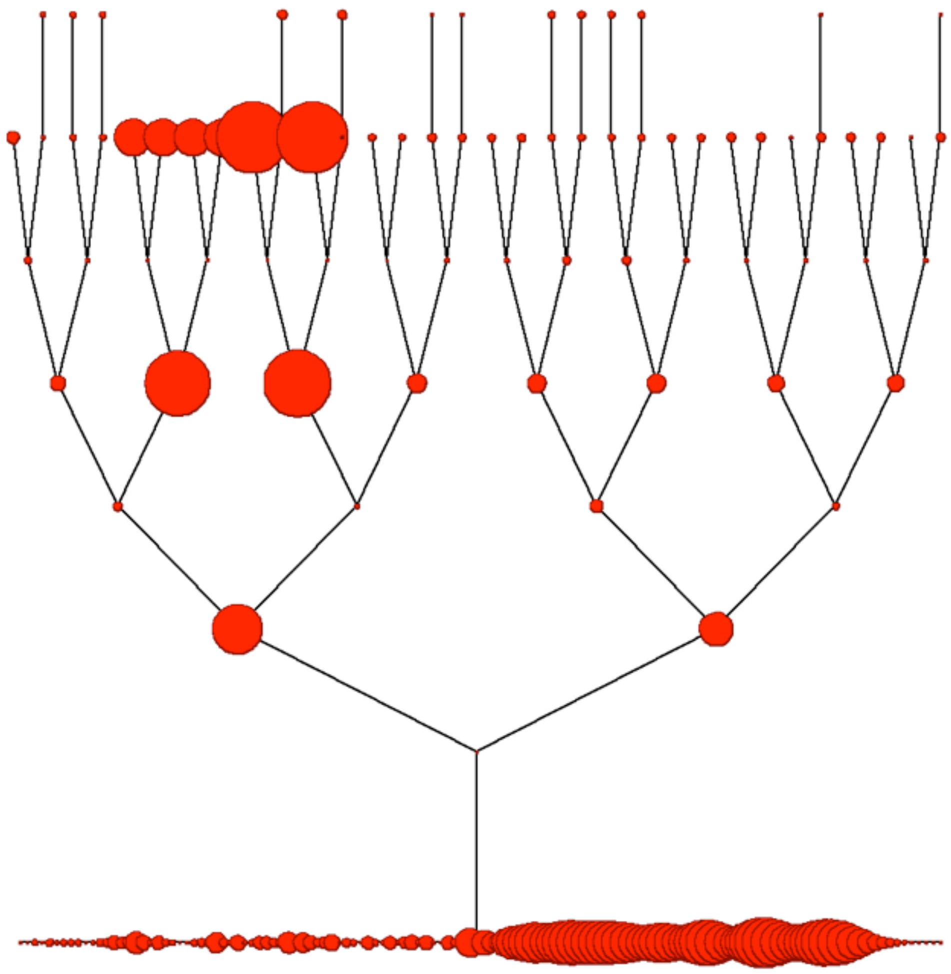
Continuous-time quantum walk [FGG '07]



Quantum walk $|\psi_t\rangle = e^{iA_G t}|\psi_0\rangle$



Quantum walk $|\psi_t\rangle = e^{iA_G t}|\psi_0\rangle$



Wave transmits! $\Rightarrow$ output $\varphi(x) = 1$

Two generalizations:

- **Theorem** ([FGG ’07]): A balanced binary AND-OR formula can be evaluated in time $N^{\frac{1}{2}+o(1)}$.

Unbalanced AND-OR

- **Theorem** ([ACRŠZ ’07]):
  - An “approximately balanced” AND-OR formula can be evaluated with $O(\sqrt{N})$ queries (optimal!).
  - A general AND-OR formula can be evaluated with $N^{\frac{1}{2}+o(1)}$ queries.

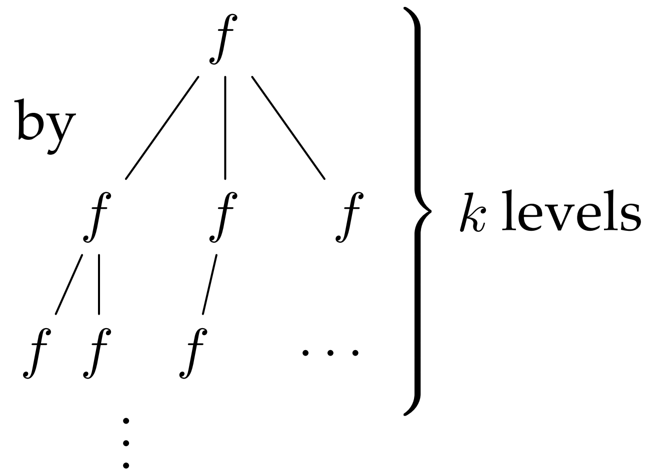
Balanced, More gates

- **Theorem** ([RŠ ’08]): A balanced formula ϕ over a gate set including all three-bit gates can be evaluated in $O(\mathrm{Adv}(\phi))$ queries (optimal!).

(Running time is poly-logarithmically slower in each case, after preprocessing.)

Span programs [Karchmer, Wigderson '93]

- **Theorem** ([R, Špalek '08]): Let $f : \{0,1\}^n \to \{0,1\}$
  Define $f^k : \{0,1\}^{n^k} \to \{0,1\}$ by



Let P be a *span program* computing f.
Then

$$\underbrace{Q(f^k)}_{\text{quantum query complexity}} = O\big(\underbrace{\mathrm{wsize}(P)}_{\text{span program complexity measure}}{}^k\big)$$

➡ Many optimal algorithms: for *f* any ≤3-bit function (e.g., AND, OR, PARITY, MAJ_3), and ~70 of ~200 different 4-bit functions…

- Open problems:
  - How can we find more good span programs?
  - Are span programs useful for developing other qu. algorithms?

Quantum query complexity Q(f)

- Time complexity = number of gates
- Query complexity = number of input bits that must be looked at
    - e.g., Search:
        - classical query complexity of OR_n is $\Theta(n)$ for both deterministic & randomized algorithms
        - quantum query complexity is $Q(OR_n)=\Theta(\sqrt{n})$, by Grover search
    - Most quantum algorithms are based on good qu. query algorithms
    - *Provable* lower bounds

Two methods to lower-bound Q(f)

- **Polynomial method**: $Q(f) = \Omega\big(\widetilde{\deg}(f)\big)$
  - for total functions $Q(f) \leq D(f) = O\big(\widetilde{\deg}(f)^6\big)$
- **Adversary method**: "How much can be learned from a single query?"

$$\mathrm{Adv}(f) = \max_{\substack{\text{adversary matrices } \Gamma: \\ \Gamma \geq 0}} \frac{\|\Gamma\|}{\max_{j \in [n]} \|\Gamma_j\|}$$

- Incomparable lower bounds:

	$\widetilde{\deg}$	Adv	
Element Distinctness:	$n^{2/3}$	$n^{1/3}$	
Ambainis formula:	$\leq 2^d$	2.5^d	$(n=4^d)$

Polynomial method $\quad Q(f) = \Omega(\widetilde{\deg}(f))$

Adversary bound $\quad Q(f) = \Omega(\mathrm{Adv}(f))$

$$\mathrm{Adv}(f) = \max_{\substack{\text{adversary matrices } \Gamma: \\ \Gamma \geq 0}} \frac{\|\Gamma\|}{\max_{j \in [n]} \|\Gamma_j\|}$$

- **General adversary bound** [Høyer, Lee, Špalek '07] $\quad Q(f) = \Omega(\mathrm{Adv}^{\pm}(f))$

$$\mathrm{Adv}^{\pm}(f) = \max_{\text{adversary matrices } \Gamma} \frac{\|\Gamma\|}{\max_{j \in [n]} \|\Gamma_j\|}$$

	$\widetilde{\deg}$	Adv	$\mathrm{Adv}^{\pm}$	Q	
Element Distinctness:	$n^{2/3}$	$n^{1/3}$	??	$n^{2/3}$	
Ambainis formula:	$\leq 2^d$	2.5^d	2.513^d	$\leq 2.774^d$	$(n=4^d)$

The general adversary bound is nearly tight

- **Theorem 1:** For any $f : \{0,1\}^n \to \{0,1\}$
$$Q(f) = \Omega(\mathrm{Adv}^{\pm}(f)) \quad \text{[HLŠ '07]}$$
and $$Q(f) = O\left(\mathrm{Adv}^{\pm}(f)\frac{\log \mathrm{Adv}^{\pm}(f)}{\log\log \mathrm{Adv}^{\pm}(f)}\right)$$

- Nearly tight characterization of quantum query complexity; the general adversary bound is always (almost) optimal

	Adv	$\widetilde{\deg}$	$\mathrm{Adv}^{\pm}$	Q	
Element Distinctness:	$n^{1/3}$	$n^{2/3}$	$\geq n^{2/3}/\log n$	$n^{2/3}$	
Ambainis formula:	2.5^d	$\leq 2^d$	2.513^d	2.513^d	($n=4^d$)

- Simpler, "greedier" semi-definite program than [Barnum, Saks, Szegedy '03]

Two steps to proving Theorem 1…

- **Theorem 1:** For any $f : \{0,1\}^n \to \{0,1\}$

$$Q(f) = O\left(\mathrm{Adv}^{\pm}(f)\frac{\log \mathrm{Adv}^{\pm}(f)}{\log\log \mathrm{Adv}^{\pm}(f)}\right)$$

- **Theorem 2:** For any boolean function f,

$$\inf_{P \text{ computing } f} \mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(f)$$

- **Theorem 3:** For any span program P computing f,

$$Q(f) = O\left(\mathrm{wsize}(P)\frac{\log \mathrm{wsize}(P)}{\log\log \mathrm{wsize}(P)}\right)$$

- **Theorem 2:** $\inf_{P \text{ computing } f} \mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(f) = O(Q(f))$

- **Theorem 3:** If P computes f, $Q(f) = O\left(\mathrm{wsize}(P)\frac{\log \mathrm{wsize}(P)}{\log\log \mathrm{wsize}(P)}\right)$

Span programs are equivalent to quantum computers!

(up to a log factor)

Model: Complexity measure:	Quantum algorithms query complexity	$\approx$	Span programs witness size

- **Theorem 2:** $\displaystyle \inf_{P \text{ computing } f} \mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(f) = O(Q(f))$

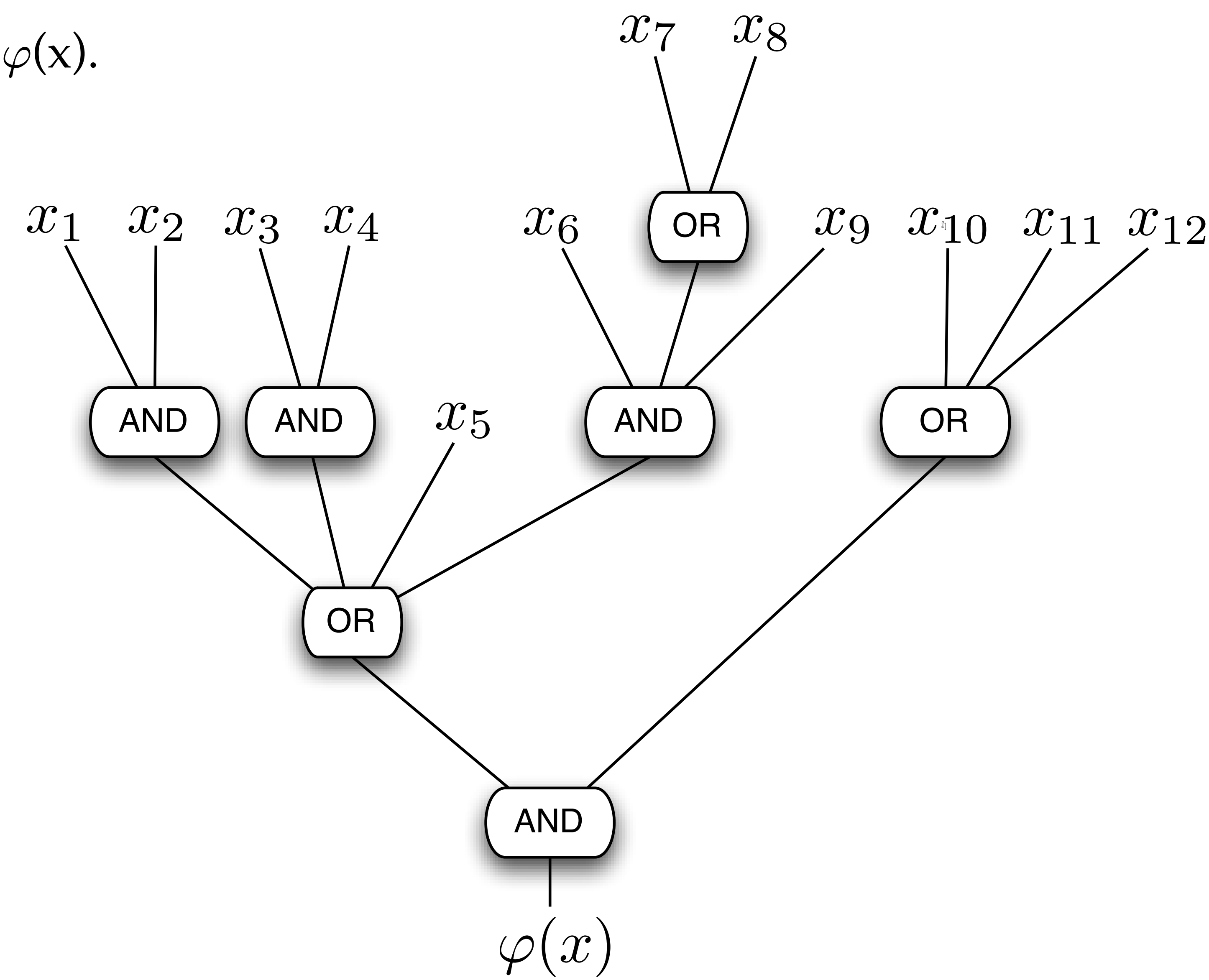
- **Theorem 3:** If P computes f, $\displaystyle Q(f) = O\left(\mathrm{wsize}(P) \frac{\log \mathrm{wsize}(P)}{\log \log \mathrm{wsize}(P)}\right)$

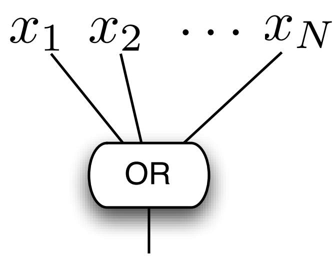
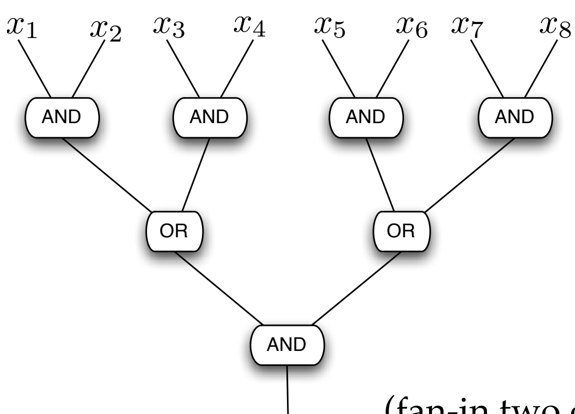
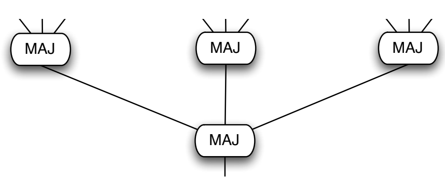
- **Thm.** [RŠ '08]:
  $$Q(f_P^k) = O\big(\mathrm{wsize}(P)^k\big)$$
- **Thm.** [HLŠ '07, R '09]:
  $$\mathrm{Adv}^{\pm}(f^k) = O\big(\mathrm{Adv}^{\pm}(f)^k\big)$$

Using Theorem 2, implies optimal qu. algorithm for evaluating balanced formulas over *any finite* gate set

Def.: Read-once formula φ on gate set $\mathcal{S}$
= Tree of nested gates from $\mathcal{S}$, with each input appearing once

Problem: Evaluate φ(x).



	Classical	Quantum
	$\Theta(n)$	$\Theta(\sqrt{n})$ [Grover '96]
Balanced AND-OR  (fan-in two case)	$\Theta(n^{0.753\ldots})$ [Snir '85] [Saks, Wigderson '86] [Santha '95]	$\Theta(\sqrt{n})$ [Farhi, Goldstone, Gutmann '07] [Ambainis, Childs, R, Špalek, Zhang '07]
General read-once AND-OR ⋮	$\Omega(n^{0.51})$ [Heiman, Wigderson '91]	$\Omega(\sqrt{n})$, [Barnum, Saks '04] $\sqrt{n} \cdot 2^{O(\sqrt{\log n})}$ [ACRŠZ '07]
Balanced MAJ_3 	$\Omega(2.333^d)$, $O(2.654^d)$ [Jayram, Kumar, Sivakumar '03]	$\Theta(2^d)$ ⋮ [RŠ '08]

	Classical	**Quantum**
OR_n (Search)	$\Theta(n)$	$\Theta(\sqrt{n})$
Balanced AND-OR	$\Theta(n^{0.753\ldots})$	$\Theta(\sqrt{n})$
General read-once AND-OR	$\Omega(n^{0.51})$	$\Omega(\sqrt{n})$, $\sqrt{n} \cdot 2^{O(\sqrt{(\log n)})}$
Balanced MAJ_3 …	$\Omega(2.333^d)$, $O(2.654^d)$	$\Theta(2^d)$
"Approximately balanced" formula over an arbitrary finite gate set	???	$\Theta(\mathrm{Adv}^{\pm}(f))$ [R '09]
Unbalanced formulas		Query complexity now understood, but not time-complexity

Two steps to proving Theorem 1...

- **Theorem 1:** For any $f : \{0,1\}^n \to \{0,1\}$

$$Q(f) = O\left(\mathrm{Adv}^{\pm}(f) \frac{\log \mathrm{Adv}^{\pm}(f)}{\log\log \mathrm{Adv}^{\pm}(f)}\right)$$

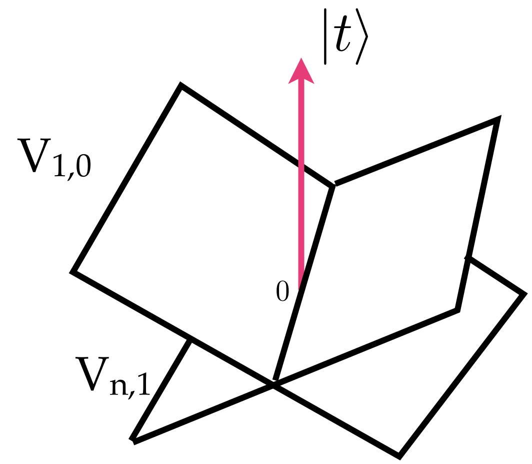
- **Theorem 2:** For any boolean function f,

$$\inf_{P \text{ computing } f} \mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(f)$$

- **Theorem 3:** For any span program P computing f,

$$Q(f) = O\left(\mathrm{wsize}(P) \frac{\log \mathrm{wsize}(P)}{\log\log \mathrm{wsize}(P)}\right)$$

- **Definition:** **Span program** P on n bits



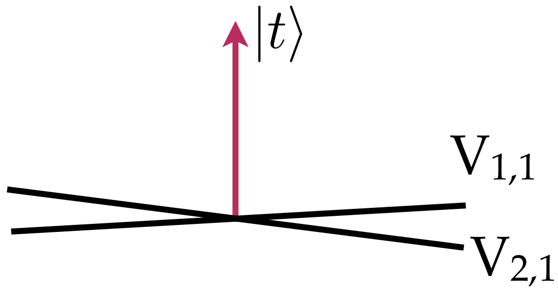
  - vector space V
  - target vector $|t\rangle$
  - subspaces $V_{1,0}$ $V_{1,1}$ $V_{2,0}$ $V_{2,1}$ … $V_{n,0}$ $V_{n,1}$

  - P "computes" $f_P : \{0,1\}^n \to \{0,1\}$

$$f_P(x) = 1 \iff |t\rangle \in \text{Span} \cup_j V_{j,x_j}$$

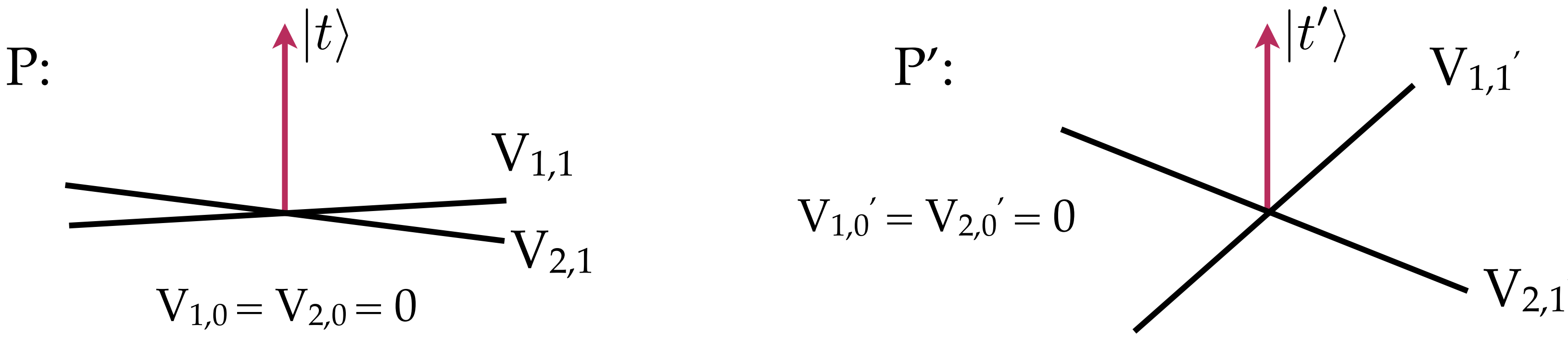
- **Example:** V=$\mathbf{C}^2$

$V_{1,0} = V_{2,0} = 0$



$\implies f_P = \text{AND}_2$

Example



- $f_P = f_{P'} = \mathrm{AND}_2$ but P′ seems better…

$$\mathrm{wsize}(P, 11) > \mathrm{wsize}(P', 11)$$

Span programs in coordinates

- Span program P: target $|t\rangle$

$$A = \begin{array}{c} \begin{array}{ccccc} \overbrace{\hphantom{|v_{1,0,1}\rangle \cdots |v_{1,0,m}\rangle}}^{V_{1,0}} & \overbrace{\hphantom{|v_{1,1,1}\rangle \cdots |v_{1,1,m}\rangle}}^{V_{1,1}} & \cdots & \overbrace{\hphantom{|v_{n,0,1}\rangle \cdots |v_{n,0,m}\rangle}}^{V_{n,0}} & \overbrace{\hphantom{|v_{n,1,1}\rangle \cdots |v_{n,1,m}\rangle}}^{V_{n,1}} \end{array} \\ \begin{pmatrix} \vert & & \vert & \vert & & \vert & & \vert & & \vert & \vert & & \vert \\ |v_{1,0,1}\rangle & \cdots & |v_{1,0,m}\rangle & |v_{1,1,1}\rangle & \cdots & |v_{1,1,m}\rangle & & |v_{n,0,1}\rangle & \cdots & |v_{n,0,m}\rangle & |v_{n,1,1}\rangle & \cdots & |v_{n,1,m}\rangle \\ \vert & & \vert & \vert & & \vert & & \vert & & \vert & \vert & & \vert \end{pmatrix} \end{array}$$

$\Pi(x)$ = projection onto available coordinates

Then

$$f_P(x) = 1 \iff |t\rangle \in \mathrm{Range}(A\Pi(x))$$

- Def.: If $f(x) = 1$, let $\mathrm{wsize}(P, x) = \min_{|w\rangle :\, A\Pi(x)|w\rangle = |t\rangle} \| |w\rangle \|^2$

(intuition: want a short witness)

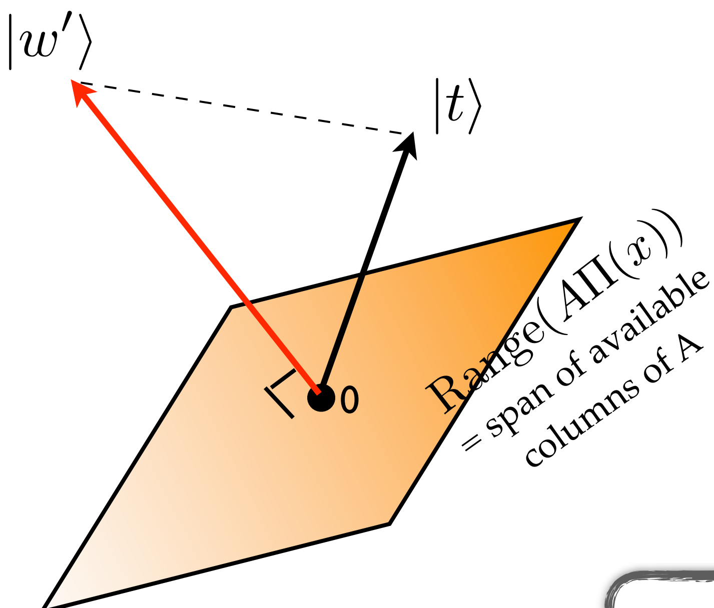
$$f_P(x) = 1 \implies |t\rangle \in \mathrm{Range}(A\Pi(x))$$

$$\mathrm{wsize}(P, x) = \min_{|w\rangle :\, A\Pi(x)|w\rangle = |t\rangle} \| |w\rangle \|^2$$

(intuition: want a short witness)

$$f_P(x) = 0 \implies |t\rangle \notin \mathrm{Range}(A\Pi(x))$$

$$\iff \exists\, |w'\rangle \text{ orthogonal to } \mathrm{Range}(A\Pi(x)) \text{ with } \langle t|w'\rangle \neq 0$$



$$\mathrm{wsize}(P, x) = \min_{\substack{|w'\rangle :\, \langle t|w'\rangle = 1 \\ \langle w'|\Pi(x) A = 0}} \| A^\dagger |w'\rangle \|^2$$

(intuition: if $|t\rangle$ is *close* to the span of available columns of A, then wsize should be *large*)

Definition: $\mathrm{wsize}(P) = \max_x \mathrm{wsize}(P, x)$

Example: Search (OR)

- Define a span program P as follows:
  - Vector space V = **C**
  - Target vector $|t\rangle = n^{1/4}$

$$A = \begin{array}{cccccccc} V_{1,0} & V_{1,1} & V_{2,0} & V_{2,1} & & V_{n,0} & V_{n,1} \\ \end{array}$$
$$\begin{pmatrix} 0 & 1 & 0 & 1 & \cdots & 0 & 1 \end{pmatrix}$$

➡ $f_P = \mathrm{OR}_n$

$$\mathrm{wsize}(P, 0^n) = \sqrt{n} \qquad |w'\rangle = 1/n^{1/4}$$

$$\mathrm{wsize}(P, 10\ldots0) = \sqrt{n} \qquad |w\rangle = (0, n^{1/4}, 0, \ldots, 0) \qquad \cdots$$

➡ $\mathrm{wsize}(P) = \sqrt{n}$

$$f_P(x) = 1 \implies \mathrm{wsize}(P, x) = \min_{|w\rangle :\, A\Pi(x)|w\rangle = |t\rangle} \| |w\rangle \|^2$$

(intuition: want a short witness)

$$f_P(x) = 0 \implies \mathrm{wsize}(P, x) = \min_{\substack{|w'\rangle :\, \langle t|w'\rangle = 1 \\ \langle w'|\Pi(x)A = 0}} \| A^\dagger |w'\rangle \|^2$$

(intuition: if $|t\rangle$ is *close* to the span of available columns of A, then wsize should be *large*)

Definition: $\mathrm{wsize}(P) = \max_x \mathrm{wsize}(P, x)$

- Why is this the right definition?

1. *Negating* a span program leaves wsize invariant
2. *Composing* span programs: wsize is multiplicative
3. Leads to quantum *algorithms* $Q(f_P^k) = O\big(\mathrm{wsize}(P)^k\big)$ [RŠ'08]

$Q(f_P) = \tilde{O}\big(\mathrm{wsize}(P)\big)$ (Theorem 3)

Proof of Theorem 2

- **Theorem 2:** For any boolean function f, $\inf_{P:\, f_P = f} \mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(f)$

- **Theorem 2:** For any boolean function f, $\inf_{P:\, f_P = f} \mathrm{wsize}(P) \leq \mathrm{Adv}^{\pm}(f)$

  Proof:

We look for span programs where the *rows* of A correspond to $\{x : f(x) = 0\}$

$$
\begin{array}{cc}
\begin{array}{c}\text{target is all}\\ \text{1s vector}\end{array} & \\
|t\rangle = \begin{pmatrix}1\\1\\\vdots\\1\end{pmatrix} &
\begin{array}{c}
\begin{array}{ccccc} V_{1,0} & V_{1,1} & \cdots & V_{n,0} & V_{n,1} \end{array}\\
\left(\begin{array}{cc|cc|c|cc|cc}
| & | & | & | & & | & | & | & | \\
|v_{1,0,1}\rangle \cdots & |v_{1,0,m}\rangle & |v_{1,1,1}\rangle \cdots & |v_{1,1,m}\rangle & & |v_{n,0,1}\rangle \cdots & |v_{n,0,m}\rangle & |v_{n,1,1}\rangle \cdots & |v_{n,1,m}\rangle \\
| & | & | & | & & | & | & | & | \\
| & | & \multicolumn{2}{c|}{0} & & \multicolumn{2}{c|}{0} & | & |
\end{array}\right)
\begin{array}{l} \left.\begin{array}{c}\\ \\ \\ \end{array}\right\} f^{-1}(0) \\ \leftarrow x = 10\ldots0 \end{array}
\end{array}
\end{array}
$$

…and in the row corresponding to *x*,
the columns available for input *x* are all *zero*

(Such span programs are said to be in “canonical form” [KW’93].)

This form guarantees that $f(x) = 0 \;\Rightarrow\; f_P(x) = 0$

($|w'\rangle = |x\rangle$ itself is the witness)

Example: AND

$$
|t\rangle = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}
\quad
\begin{array}{c}
\begin{array}{cccc} V_{1,1} & V_{2,1} & \cdots & V_{n,1} \end{array} \\
\begin{pmatrix} 1 & 0 & & 0 \\ 0 & 1 & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & 1 \end{pmatrix}
\end{array}
\longleftarrow x = 011\ldots1
$$

➡ $f_P = \mathrm{AND}_n$

$$|t\rangle = \begin{pmatrix}1\\1\\\vdots\\1\end{pmatrix} \quad \left.\left(\begin{array}{ccccc} \overbrace{|v_{1,0,1}\rangle \cdots |v_{1,0,m}\rangle}^{V_{1,0}} & \overbrace{|v_{1,1,1}\rangle \cdots |v_{1,1,m}\rangle}^{V_{1,1}} & \cdots & \overbrace{|v_{n,0,1}\rangle \cdots |v_{n,0,m}\rangle}^{V_{n,0}} & \overbrace{|v_{n,1,1}\rangle \cdots |v_{n,1,m}\rangle}^{V_{n,1}} \\ \langle v_{x1}| & 0 & & 0 & \langle v_{xn}| \end{array}\right)\right\} f^{-1}(0) \quad \leftarrow x = 10\ldots0$$

in the xth row, the columns available for input x are all 0; hence

$$f(x) = 0 \;\Rightarrow\; f_P(x) = 0$$

Now consider a $y \in f^{-1}(1)$

We want to find vectors $|v_{y1}\rangle, \ldots, |v_{yn}\rangle$

such that $\forall\, x \in f^{-1}(0), \quad 1 = \sum_{j : x_j \neq y_j} \langle v_{xj} | v_{yj} \rangle$

The witness size is $\max_x \sum_j \| |v_{xj}\rangle \|^2$

$$
|t\rangle = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}
\begin{array}{c}
\begin{array}{ccccc} \qquad V_{1,0} \qquad & \qquad V_{1,1} \qquad & \cdots & \qquad V_{n,0} \qquad & \qquad V_{n,1} \qquad \end{array} \\
\left.\begin{pmatrix}
| \quad\quad | & | \quad\quad | & & | \quad\quad | & | \quad\quad | \\
|v_{1,0,1}\rangle \cdots |v_{1,0,m}\rangle & |v_{1,1,1}\rangle \cdots |v_{1,1,m}\rangle & & |v_{n,0,1}\rangle \cdots |v_{n,0,m}\rangle & |v_{n,1,1}\rangle \cdots |v_{n,1,m}\rangle \\
| \quad\quad | & | \quad\quad | & & | \quad\quad | & | \quad\quad | \\
-\langle v_{x1}|- & -0- & & -0- & -\langle v_{xn}|-
\end{pmatrix} \leftarrow x = 10\ldots0 \;\right\} f^{-1}(0)
\end{array}
$$

$$
\implies \inf_{P:\, f_P = f} \mathrm{wsize}(P) \le \inf_{\substack{\{|v_{xj}\rangle\}: \\ \text{if } f(x) \neq f(y), \sum_{j: x_j \neq y_j} \langle v_{xj} | v_{yj} \rangle = 1}} \max_x \sum_j \| |v_{xj}\rangle \|^2
$$

$$
= \min_{\substack{X \succeq 0: \\ \forall (x,y) \in \Delta, \sum_{j: x_j \neq y_j} \langle x,j | X | y,j \rangle = 1}} \max_x \sum_j \langle x, j | X | x, j \rangle
$$

(Cholesky decomposition)

$$
= \mathrm{Adv}^{\pm}(f)
$$

(SDP duality) $\square$

Proof of Theorem 3

- **Theorem 3:** For any span program P, $Q(f_P) = O\left(\mathrm{wsize}(P)\frac{\log \mathrm{wsize}(P)}{\log\log \mathrm{wsize}(P)}\right)$

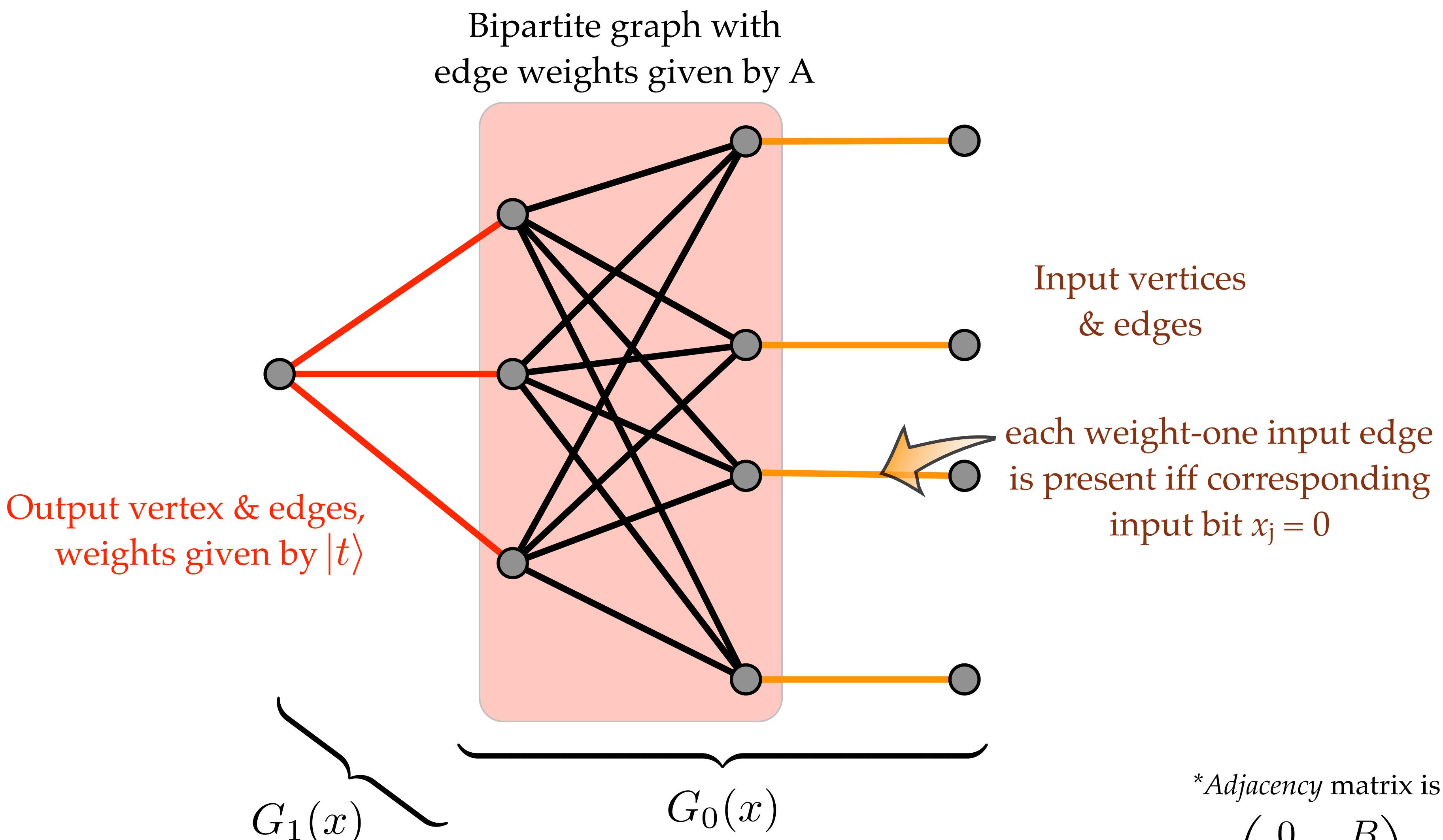
1.
Correspondence between P and bipartite graphs $G_P(x)$

2.
Eigenvalue-zero eigenvectors imply an "effective" spectral gap around zero

3.
Quantum algorithm for detecting eigenvectors of structured graphs

- **Def.:** Let $G_0(x)$ be the bipartite graph with *biadjacency* matrix* $\begin{pmatrix} A \\ \mathbf{1} - \Pi(x) \end{pmatrix}$

  Let $G_1(x)$ " " $\begin{pmatrix} |t\rangle & A \\ 0 & \mathbf{1} - \Pi(x) \end{pmatrix}$



**Adjacency* matrix is $\begin{pmatrix} 0 & B \\ B^\dagger & 0 \end{pmatrix}$

- **Def.:** Let $G_0(x)$ be the bipartite graph with *biadjacency* matrix* $\begin{pmatrix} & A \\ & \mathbf{1} - \Pi(x) \end{pmatrix}$

  Let $G_1(x)$ " " $\begin{pmatrix} |t\rangle & A \\ 0 & \mathbf{1} - \Pi(x) \end{pmatrix}$

- **Lemma:** $G_1(x)$ has an eigenvalue-zero eigenvector overlapping the output vertex $\Leftrightarrow f_P(x) = 1$

  $G_0(x)$ has an eigenvalue-zero eigenvector overlapping $|t\rangle \Leftrightarrow f_P(x) = 0$

  Proof: For $G_1(x)$: $$0 = \begin{pmatrix} |t\rangle & A \\ 0 & \mathbf{1} - \Pi(x) \end{pmatrix} \begin{pmatrix} 1 \\ |v\rangle \end{pmatrix}$$

  $$\Leftrightarrow \quad \Pi(x)|v\rangle = |v\rangle \quad \text{and} \quad |t\rangle = -A|v\rangle$$

  $$\Leftrightarrow \quad f_P(x) = 1 \qquad \square$$

Note: After normalizing the eigenvector, squared overlap on the output vertex is 1/(1+wsize(P,x)). Small wsize $\Leftrightarrow$ large overlap

② Small-eigenvalue analysis

When $f_P(x) = 0$, $G_0(x)$ having an eigenvalue-zero eigenvector with large (δ) squared *overlap* on $|t\rangle$ implies that $G_1(x)$ has a large (√δ) "effective" spectral *gap* around zero.

$$G_0(x) \qquad\qquad\qquad G_1(x)$$

$$\begin{pmatrix} A \\ \mathbf{1} - \Pi(x) \end{pmatrix} \qquad\qquad \begin{pmatrix} |t\rangle & A \\ 0 & \mathbf{1} - \Pi(x) \end{pmatrix}$$

Idea: Think of $G_1(x)$ as a perturbation of $G_0(x)$.

② Small-eigenvalue analysis

Theorem. *Let G be a weighted bipartite graph on $V = T \sqcup U$. Assume that for some $\delta > 0$ and $|t\rangle \in \mathbf{C}^T$, the adjacency matrix has an eigenvalue-zero eigenvector $|\psi\rangle$ with*

$$\frac{|\langle t|\psi_T\rangle|^2}{\||\psi\rangle\|^2} \geq \delta$$

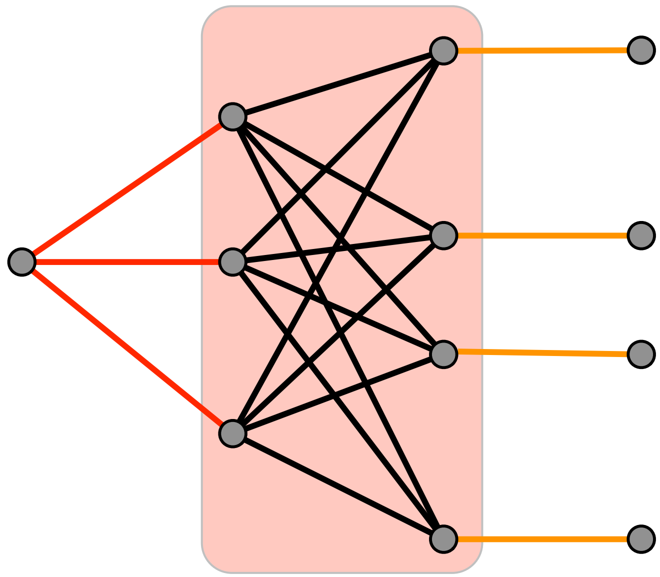
Let G' be the same as G except with a new vertex v added to the U side, and for $i \in T$ the new edge (τ_i, v) weighted by $\langle i|t\rangle$. Take $\{|\lambda\rangle\}$ a complete set of orthonormal eigenvectors of $A_{G'}$. Then for all $\Lambda \geq 0$,

$$\sum_{\lambda:\, |\lambda| \leq \Lambda} |\langle v|\lambda\rangle|^2 \leq \frac{8\Lambda^2}{\delta}$$

Good eigenvalue-zero eigenvector $\Rightarrow$ Large effective spectral gap

(Think $\delta = 1/\mathrm{wsize}(P)^2,\ \Lambda = 1/\mathrm{wsize}(P)$)

3 Quantum algorithm



Scale the target vector down by $1/\sqrt{\text{wsize}(P)}$.

- When $f_P(x) = 1$, there is an eigenvector with large ($\geq \frac{1}{2}$) squared overlap on the output vertex
- When $f_P(x) = 0$, there is a spectral gap of $1/\text{wsize}(P)$ around zero

Algorithm: Start at the output vertex and "measure" the adjacency matrix (as a Hamiltonian). Output 1 iff the measurement returns 0.

Key technical step: Since we have no control over the norm of the matrix, need [Cleve, Gottesman, Mosca, Somma, Yonge-Mallo '09] to simulate the measurement with a log / log log overhead factor. $\square$

Summary

- **Theorem 2:** For any boolean function f,

$$\inf_{P \text{ computing } f} \mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(f)$$

- **Theorem 3:** For any span program P,

$$Q(f) = O\left(\mathrm{wsize}(P) \frac{\log \mathrm{wsize}(P)}{\log \log \mathrm{wsize}(P)}\right)$$

Main corollaries

1. The general adversary bound is (almost) optimal for every total or partial function

$$f : \{0,1\}^n \to \{0,1\}^{\mathrm{poly}(\log n)}$$

2. Optimal quantum algorithm for evaluating balanced formulas over *any finite* gate set

3. Span programs are (almost) equivalent to quantum computers

Recipe for finding optimal quantum query algorithms

- Find a solution to: $$\mathrm{Adv}^{\pm}(f) = \min_{\substack{\{X_j \succeq 0\}: \\ \forall (x,y)\in\Delta,\ \sum_{j:x_j\neq y_j} \langle x|X_j|y\rangle = 1}} \max_x \sum_j \langle x|X_j|x\rangle \quad (*)$$

- Take the Cholesky decomposition: $\{|v_{xj}\rangle\} : \langle v_{xj}|v_{yj}\rangle = \langle x|X_j|y\rangle$

- Use the entries of the vectors to weight the edges of a graph, and run phase estimation on the quantum walk…

$$B_G = \left(\begin{array}{c|c} 1 & \\ \vdots & \sum_{x:f(x)=0} \sum_{j=1}^{n} |x\rangle\langle \overline{x_j}| \otimes \langle v_{xj}| \\ 1 & \end{array} \right)$$

- But how can we find good solutions to $(*)$?

Open problems

- Can the log overhead factor be removed? Is Adv± tight in the *continuous-query* model?
- Functions with non-binary domains? $f : \{1, 2, \ldots, k\}^n \to \{0, 1\}$
- Is there a good *classical* algorithm for evaluating span programs? Any algorithm faster than $O(\mathrm{wsize}(P)^6)$ would give a better relationship between classical and quantum query complexities. $D(f) = O(Q(f)^6)$
  - Our results apply to both total and *partial* functions, though (e.g., Simon's prob.)
- Robustness?
- More explicit and *time-efficient* algorithms

Thank you!