THEORY OF COMPUTING, Volume 8 (2012), pp. 291–319
www.theoryofcomputing.org

Span-Program-Based Quantum Algorithm for Evaluating Formulas

Ben W. Reichardt   Robert Špalek

Received: September 22, 2008; revised: October 5, 2011; published: July 7, 2012.

Abstract: We give a quantum algorithm for evaluating formulas over an extended gate set, including all two- and three-bit binary gates (e. g., NAND, 3-majority). The algorithm is optimal on read-once formulas for which each gate's inputs are balanced in a certain sense.

The main new tool is a correspondence between a classical linear-algebraic model of computation, "span programs," and weighted bipartite graphs. A span program's evaluation corresponds to an eigenvalue-zero eigenvector of the associated graph. A quantum computer can therefore evaluate the span program by applying spectral estimation to the graph.

For example, the classical complexity of evaluating the balanced ternary majority formula is unknown, and the natural generalization of randomized alpha-beta pruning is known to be suboptimal. In contrast, our algorithm generalizes the optimal quantum AND-OR formula evaluation algorithm and is optimal for evaluating the balanced ternary majority formula.

ACM Classification: F.1.1, F.2.2

AMS Classification: 81P68, 68Q05, 68Q12

Key words and phrases: span programs, quantum computation, query complexity, formula evaluation

1 Introduction

A formula φ on gate set $\mathcal{S}$ and of size N is a tree with N leaves, such that each internal node is a gate from $\mathcal{S}$ on its children. The read-once formula evaluation problem is to evaluate $\varphi(x)$ given oracle access to the input string $x = x_1 x_2 \ldots x_N$. An optimal, $O(\sqrt{N})$-query quantum algorithm is known to evaluate "approximately balanced" formulas over the gates $\mathcal{S} = \{\text{AND, OR, NOT}\}$ [4]. We develop an optimal quantum algorithm for evaluating balanced formulas over a greatly extended gate set. The notion of

DOI: 10.4086/toc.2012.v008a013

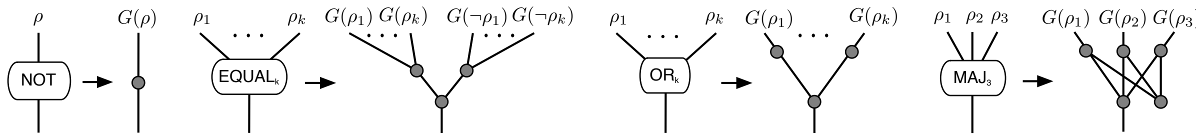


Figure 1: To convert a formula φ to the corresponding graph $G(\varphi)$, we recursively apply substitution rules starting at the root to convert each gate into a gadget subgraph. Some of the rules are shown here, except with the edge weights not indicated. The dangling edges at the top and bottom of each gadget are the input edges and output edge, respectively. To compose two gates, the output edge of one is identified with an input edge of the next (see Figure 2).

balance is technical, but includes as a special case layered formulas in which gates at the same depth are all equal.

The idea of our algorithm is to consider a weighted graph $G(\varphi)$ obtained by replacing each gate of the formula φ with a small gadget subgraph, and possibly also duplicating subformulas. Figure 1 has several examples. We relate the evaluation of φ to the presence or absence of small-eigenvalue eigenvectors of the weighted adjacency matrix $A_{G(\varphi)}$ that are supported on the root vertex of $G(\varphi)$. The quantum algorithm runs spectral estimation to either detect these eigenvectors or not, and therefore to evaluate φ.

As a special case, for example, our algorithm implies:

Theorem 1.1. *A balanced ternary majority* (MAJ_3) *formula of depth d, on $N = 3^d$ inputs, can be evaluated by a quantum algorithm with bounded error using $O(2^d)$ oracle queries, which is optimal.*

The classical complexity of evaluating this formula is $\Omega((5/2)^d)$, and the previous best quantum algorithm, from [4], uses $O(\sqrt{5}^d)$ queries.

The graph gadgets themselves are derived from "span programs" [26], establishing a new connection between span programs and quantum algorithms. The span program computational model has been used previously in classical complexity theory to prove lower bounds on formula size [26, 5] and monotone span programs are related to linear secret-sharing schemes [8]. (Most, though not all [1], applications are over finite fields, whereas we use the definition over $\mathbb{C}$.) Work subsequent to this paper has shown that span programs are equivalent to quantum query algorithms [35].

Classical and quantum background The formula evaluation problem has been well-studied in the classical computer model. Classically, the case $\mathcal{S} = \{\mathrm{NAND}\}$ is best understood. A formula with only NAND gates is equivalent to one with alternating levels of AND and OR gates, a so-called "AND-OR formula," also known as a read-once function or a two-player game tree. Using randomized alpha-beta pruning, one can compute the value of a balanced binary AND-OR formula with zero error in expected time $O(N^{\log_2[(1+\sqrt{33})/4]}) = O(N^{0.753\ldots})$ [42, 40], and this is optimal even for bounded-error algorithms [41]. Other formulas have different query complexities; for example, evaluating a single OR gate with N inputs needs $\Theta(N)$ oracle queries. The best general lower bound of $\Omega(N^{0.51})$ [22] has recently been improved to $\Omega(N^{0.746})$ [2].

If we allow the use of a quantum computer with coherent oracle access to the input, however, then the situation is simpler; $\Theta(\sqrt{N})$ queries are necessary and sufficient to evaluate *any* {AND, OR, NOT} formula with bounded error. The general lower bound is due to [6]. For the upper bound, in one extreme case, Grover search [20, 21] evaluates an OR gate of degree N using $O(\sqrt{N})$ oracle queries. In the other extreme case, Farhi, Goldstone and Gutmann devised a breakthrough algorithm for evaluating the depth-($\log_2 N$) balanced binary AND-OR formula using $N^{1/2+o(1)}$ queries [17, 15]. Ambainis et al. [4] improved this to $O(\sqrt{N})$ queries, even for "approximately balanced" formulas, and also extended the algorithm to arbitrary $\{\text{AND, OR, NOT}\}$ formulas with $N^{\frac{1}{2}+o(1)}$ queries and time after a preprocessing step. More recent work has given a general $O(\sqrt{N})$-query upper bound [37, 36].

This paper shows other nice features of the formula evaluation problem in the quantum computer model. Classically, with the exception of $\{\text{NAND}\}$, $\{\text{NOR}\}$ and a few trivial cases like $\{\text{PARITY}\}$, most gate sets are poorly understood. In 1986, Boppana asked the complexity of evaluating the balanced, depth-d ternary majority (MAJ_3) function [40], and today the complexity is only known to lie between $\Omega((5/2)^d)$ and $O((2.6537\ldots)^d)$ [25, 31]. In particular, the naïve generalization of randomized alpha-beta pruning—recursively evaluate two random immediate subformulas and then the third if they disagree—runs in expected time $O((8/3)^d)$ and is suboptimal. This suggests that the balanced ternary majority function is significantly different from the balanced k-ary NAND function, for which randomized alpha-beta pruning is known to be optimal [40]. In contrast, we show that the optimal quantum algorithm of [4] does extend to give an optimal $O(2^d)$-query algorithm for evaluating the balanced ternary majority formula. Moreover, the algorithm also generalizes to a significantly larger gate set $\mathcal{S}$.

Subsequent work The graph gadgets we use are derived from constant-size span programs, but in the conference version of this article [39] we speculated that "larger span programs could directly give useful new quantum algorithms." Later work has borne this out. Based on span programs, quantum algorithms have been developed for matrix rank [9], various subgraph-detection and related problems [10, 45, 29, 12], graph collision [18], the k-distinctness problem under a certain promise [11], and further formula-evaluation problems [36, 38, 37, 44, 27]. For formula evaluation, the quantum query complexity of arbitrary read-once formulas has been characterized, with no gate set restrictions or balance condition needed.

In fact, span programs have been shown to be equivalent to quantum query algorithms [35, 37, 30]. Formally, the general adversary lower bound on quantum query complexity exactly equals the least span program witness size, which via an algorithm upper bounds query complexity to within a constant factor. (The general adversary bound is a semi-definite program, which can be solved to find an optimal span program for an arbitrary gate; therefore, we have removed most of the hand-constructed examples present in the conference version of this article.)

Although the quantum query complexity of read-once formulas is settled, the quantum time complexity is not. The problem is that operations between queries to the input are not necessarily efficiently implementable. Our formula-evaluation algorithm is both query optimal and time efficient. Its strict balance condition has been relaxed, while keeping a time-efficient algorithm, in [38]. Compared to this follow-up work, our proof is more technical, but our conclusions are also stronger. The analysis in [38, 35] is based on showing that the output vertex of the graph has at most a small constant squared overlap on the span of the small-eigenvalue eigenspaces. Such an "effective spectral gap" can be proved relatively easily by studying the eigenvalue-zero eigenvectors. By contrast, in this article we prove a true spectral gap, i. e., that the output vertex has no overlap on small-eigenvalue eigenspaces. This requires studying eigenvalue-λ eigenvectors for small $\lambda \neq 0$, which is not easy (see Section 4.2.2 below). For the quantum algorithm, an effective spectral gap suffices for applying phase estimation, and only leads to a slightly higher error rate.

Organization In Section 2, we briefly recall the adversary lower bounds on quantum query complexity.

We introduce span programs and present optimal span programs for all three-bit Boolean functions in Section 3. A span program corresponds to a weighted bipartite graph, and its evaluation corresponds to eigenvalue-zero eigenvectors of the graph (Theorem 3.5). This theorem provides useful intuition.

We develop a quantitative version of Theorem 3.5 in Section 4. We bound from below the overlap of the eigenvalue-zero eigenvector with a known starting state. This lower bound will imply completeness of our quantum algorithm. To show soundness of the algorithm, we also analyze small-eigenvalue eigenvectors in order to prove a spectral gap around zero. Essentially, we solve the eigenvalue equations in terms of the eigenvalue λ, and expand a series around $\lambda = 0$. The results for small-eigenvalue and eigenvalue-zero eigenvectors are closely related, and are unified using a new complexity measure we term "span program witness size."

In Section 5, we present a quantum algorithm for evaluating formulas over the gate set $\mathcal{S}$ of all three-bit Boolean functions. On formulas that are "adversary-balanced" (Definition 2.3), the algorithm is optimal.

2 Adversary lower bounds on quantum query complexity

The query complexity or decision-tree complexity of a function $f : \mathcal{D} \to E$, with $\mathcal{D} \subseteq \{0,1\}^n$, is the least number of input bits one must look at in order to evaluate f on a worst-case input. The bounded-error quantum query complexity, $Q(f)$, is defined similarly except input queries are allowed to be made in quantum superposition and the output is allowed to be wrong with probability at most $1/3$. For a formal definition, see, e. g., the textbook [33].

To show the optimality of our formula-evaluation algorithm, we will use the general adversary bound:

Definition 2.1 ([24])**.** Let $f : \{0,1\}^k \to \{0,1\}$. Define the *general adversary bound* $\mathrm{Adv}^{\pm}(f)$ by

$$\mathrm{Adv}^{\pm}(f) = \max_{\Gamma} \left\{ \|\Gamma\| : \max_i \|\Gamma \circ D_i\| \leq 1 \right\} . \tag{2.1}$$

The maximum is over all $2^k \times 2^k$ symmetric matrices Γ satisfying $\langle x|\Gamma|y\rangle = 0$ if $f(x) = f(y)$. $\Gamma \circ D_i$ denotes the entrywise matrix product with D_i, a zero-one-valued matrix defined by $\langle x|D_i|y\rangle = 1$ if and only if $x_i \neq y_i$.

Theorem 2.2 ([24])**.** *The bounded-error quantum query complexity of f, $Q(f)$, is $\Omega(\mathrm{Adv}^{\pm}(f))$.*

The general adversary bound strengthens the adversary bound [3, 7], which is defined as in Equation (2.1) but with Γ required to have non-negative entries.

To match the lower bound of Theorem 2.2, our goal will be to use $O(\mathrm{Adv}^{\pm}(\varphi))$ queries to evaluate φ. We will prove the optimality of our algorithm only for formulas satisfying a strict balance condition:

Definition 2.3. A formula φ is *adversary-balanced* if for every gate, the general adversary lower bounds of its input subformulas are equal.

This balance condition is motivated by the following composition result:

Theorem 2.4 ([24])**.** *Let* $f = g \circ (h_1, \dots, h_k)$*, where* $\mathrm{Adv}^{\pm}(h_1) = \cdots = \mathrm{Adv}^{\pm}(h_k)$ *and* $\circ$ *denotes function composition. Then* $\mathrm{Adv}^{\pm}(f) \geq \mathrm{Adv}^{\pm}(g)\mathrm{Adv}^{\pm}(h_1)$.

Subsequent work, using the equivalence between the general adversary bound and span programs, has shown this to be an equality [35]. If φ is adversary-balanced, then by Theorem 2.4 $\mathrm{Adv}^{\pm}(\varphi_g)$ is at least the product of the gate adversary bounds along any non-self-intersecting path χ from g up to an input, $\mathrm{Adv}^{\pm}(\varphi_g) \geq \prod_{h \in \chi} \mathrm{Adv}^{\pm}(h)$. Note that $\mathrm{Adv}^{\pm}(\neg f) = \mathrm{Adv}^{\pm}(f)$, so NOT gates can be inserted anywhere in an adversary-balanced formula.

3 Span programs and eigenvalue-zero graph eigenvectors

A span program P is a linear-algebraic way of specifying a function f_P. In this section, we will associate P to a certain structured collection of weighted bipartite graphs $G_P(x)$, in such a way that on input x, $f_P(x) = 1$ if and only if there exists an eigenvalue-zero eigenvector of the adjacency matrix $A_{G_P(x)}$ supported on a distinguished output node (Theorem 3.5).

In turn, this will imply that specifying a span program P for a function f directly gives a quantum algorithm for evaluating f, or for evaluating formulas including f as a gate (Section 5). The algorithm works by spectral estimation on $A_{G_P(x)}$. Its running time depends on the span program's "witness size," a new complexity measure we define that bounds the squared length of the above eigenvalue-zero eigenvectors.

Definition 3.1 (Span program [26])**.** A span program $P = (n, d, |t\rangle, V_{\mathrm{free}}, \{V_{j,b}\})$ consists of a "target" vector $|t\rangle \in \mathbb{C}^d$ and finite sets V_{free} and $V_{1,0}, V_{1,1}, \dots, V_{n,0}, V_{n,1}$ of "input" vectors from $\mathbb{C}^d$.

To P corresponds a Boolean function $f_P : \{0,1\}^n \to \{0,1\}$, defined by $f_P(x) = 1$ if and only if $|t\rangle$ lies in the span of the vectors in $V_{\mathrm{free}} \cup \bigcup_{j=1}^n V_{j,x_j}$.

For manipulating a span program, it is convenient to index the input vectors so $V_{\mathrm{free}} = \{v_i : i \in I_{\mathrm{free}}\}$ and $V_{j,b} = \{v_i : i \in I_{j,b}\}$, for disjoint index sets I_{free} and $I_{j,b}$. Let $I = I_{\mathrm{free}} \cup \bigcup_{j,b} I_{j,b}$. Furthermore, let A be the matrix whose columns are the input vectors, i. e., in standard ket notation $A = \sum_{i \in I} |v_i\rangle\langle i|$. For an input $x \in \{0,1\}^n$, let $I(x) = I_{\mathrm{free}} \cup \bigcup_{j=1}^n I_{j,x_j}$ be the indices of the available input vectors, and let $\Pi(x)$ be the projection onto the corresponding columns of A, i. e., $\Pi(x) = \sum_{i \in I(x)} |i\rangle\langle i|$.

Definition 3.2 (Witness size)**.** Consider a span program P, and a vector $s \in [0, \infty)^n$. Define $S = \sum_{j,b;i \in I_{j,b}} \sqrt{s_j} |i\rangle\langle i|$. Define P's witness size with costs s as

$$\mathrm{wsize}_s(P) = \max_{x \in \{0,1\}^n} \mathrm{wsize}_s(P, x) \, ,$$

where $\mathrm{wsize}_s(P, x)$ is given by:

- If $f_P(x) = 1$, then $|t\rangle \in \mathrm{Range}(A\Pi(x))$, so there is a witness $|w\rangle \in \mathbb{C}^{|I|}$ satisfying $A\Pi(x)|w\rangle = |t\rangle$. Then $\mathrm{wsize}_s(P,x)$ is the minimum squared length of any such witness, weighted by the costs s:

$$\mathrm{wsize}_s(P,x) = \min_{|w\rangle : A\Pi(x)|w\rangle = |t\rangle} \| S|w\rangle \|^2 . \tag{3.1}$$

- If $f_P(x) = 0$, then $|t\rangle \notin \mathrm{Range}(A\Pi(x))$. Therefore there is a witness $|w'\rangle \in \mathbb{C}^d$ satisfying $\langle t|w'\rangle = 1$ and $\Pi(x)A^\dagger|w'\rangle = 0$. Then

$$\mathrm{wsize}_s(P,x) = \min_{\substack{|w'\rangle :\, \langle t|w'\rangle = 1 \\ \Pi(x)A^\dagger|w'\rangle = 0}} \| SA^\dagger|w'\rangle \|^2 . \tag{3.2}$$

When the subscript s is omitted, the costs are taken to be uniform, $s = \vec{1} = (1,1,\ldots,1)$, e. g., $\mathrm{wsize}(P) = \mathrm{wsize}_{\vec{1}}(P)$. In this case, note that $S = 1 - \sum_{i \in I_{\mathrm{free}}} |i\rangle\langle i|$.

Observe that the witness size is invariant under the application of any invertible linear transformation to the target and input vectors. Moreover, since the witness size does not charge for using vectors in $\mathrm{Span}(V_{\mathrm{free}})$, we may eliminate the free input vectors. (See [35, Lemma 4.12, Prop. 4.10].)

Lemma 3.3. *For any span program P, we may by an invertible linear transformation on the vectors in $\mathbb{C}^d$ change the target vector to $(1,0,0,\ldots,0)$. By projecting all vectors onto the space orthogonal to the span of V_{free}, we may assume $V_{free} = \emptyset$. These operations change neither f_P nor the witness size.*

3.1 Span program as an adjacency matrix

Definition 3.4 (Graphs G_P and $G_P(x)$)**.** For a span program P, let G_P be the complex-weighted, bipartite graph with biadjacency matrix

$$B_{G_P} = \begin{pmatrix} |t\rangle & A \end{pmatrix} . \tag{3.3}$$

That is, G_P has $1 + |I|$ column vertices and d row vertices. The first column vertex, called the "output vertex," has incident edges weighted by the coordinates of $|t\rangle$. Each of the remaining $|I|$ column vertices, or "input vertices," has incident edges weighted by an input vector. The adjacency matrix of G_P is

$$A_{G_P} = \begin{pmatrix} 0 & B_{G_P} \\ B_{G_P}^\dagger & 0 \end{pmatrix} . \tag{3.4}$$

For an input $x \in \{0,1\}^n$, let $G_P(x)$ be the same graph, but with weight-one dangling edges attached to the input vertices of $I \smallsetminus I(x)$. That is, the biadjacency matrix of $G_P(x)$ is

$$B_{G_P(x)} = \begin{pmatrix} |t\rangle & A \\ 0 & 1 - \Pi(x) \end{pmatrix} . \tag{3.5}$$

Theorem 3.5. *Let P be a span program, and x an input. Then there is a vector in the kernel of $B_{G_P(x)}$ supported on the output vertex if and only if $f_P(x) = 1$. There is a vector in the kernel of $\big(A^\dagger, 1 - \Pi(x)\big)$, the first d coordinates of which having nonzero inner product with $|t\rangle$, if and only if $f_P(x) = 0$.*

Proof. A vector $(a_O, |w\rangle) \in \mathbb{C} \times \mathbb{C}^{|I|}$ is in the kernel of $B_{G_P(x)}$ if and only if $a_O|t\rangle + A|w\rangle = 0$ and $(1 - \Pi(x))|w\rangle = 0$. The second condition says that the support of $|w\rangle$ contains only vertices corresponding to indices in $I(x)$. The first condition says that the corresponding linear combination of input vectors is $-a_O|t\rangle$. When $a_O \neq 0$, the existence of a $|w\rangle$ satisfying these conditions is equivalent to $f_P(x) = 1$.

Next, we have that $f_P(x) = 0$ if and only if $|t\rangle \notin \mathrm{Range}(A\Pi(x))$, which in turn holds if and only if there is a vector $|w'\rangle \in \mathbb{C}^d$ with $\langle t|w'\rangle \neq 0$ and $\Pi(x)A^\dagger|w'\rangle = 0$. This is equivalent to the existence of a vector $(|w'\rangle, |w''\rangle) \in \mathbb{C}^d \times \mathbb{C}^{|I|}$ in the kernel of $\big(A^\dagger, 1 - \Pi(x)\big)$ with $\langle t|w'\rangle \neq 0$ (for $i \in I \smallsetminus I(x)$, set $\langle i|w''\rangle = -\langle i|A^\dagger|w'\rangle$). □

Example 3.6. The span program P with $n = 3$, $d = 2$, $|t\rangle = (1, 0)$, $V_{\mathrm{free}} = V_{1,0} = V_{2,0} = V_{3,1} = \emptyset$, $V_{1,1} = \{(1, \alpha)\}$, $V_{2,1} = \{(1, \beta)\}$, $V_{3,0} = \{(1, \gamma)\}$ computes the function $\mathrm{MAJ}_3(x_1, x_2, \bar{x}_3)$, provided α, β, γ are distinct and nonzero. The biadjacency matrix $B_{G_P(101)}$ is

$$\left(\begin{array}{c|ccc} 1 & 1 & 1 & 1 \\ 0 & \alpha & \beta & \gamma \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right).$$

3.2 Dual span program

Observe that attaching a dangling weight-one edge to the output vertex complements the conditions of Theorem 3.5. Therefore a length-one path can be thought of as a NOT gate gadget. The construction of $G_P(x)$ from G_P is also based on adding NOT gates to false inputs.

Definition 3.7 (Dual span program)**.** For a span program P, define its dual $P^\dagger$ by complementing the output and all inputs with dangling weight-one edges, and also switching each input vector set $V_{j,b}$ with $V_{j,1-b}$.

Example 3.8. For distinct, nonzero α, β, γ, the span program

$$|t\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad A = \begin{array}{c} \begin{array}{ccccc} \multicolumn{2}{c}{\overbrace{\hphantom{1\;\alpha}}^{\top}} & \bar{x}_1 & \bar{x}_2 & x_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & \alpha & 1 & 0 & 0 \\ 1 & \beta & 0 & 1 & 0 \\ 1 & \gamma & 0 & 0 & 1 \end{pmatrix} \end{array}$$

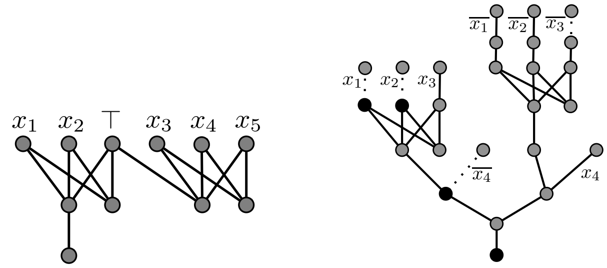
computes $\neg\mathrm{MAJ}_3(x_1, x_2, \bar{x}_3)$. Here, we have introduced the notation that $\top$ labels input vectors in V_{free}, and input vectors in $V_{j,1}$ or $V_{j,0}$ are indicated by the positive or negative literals x_j or $\bar{x}_j$, respectively; thus, e. g., $V_{3,1} = \{(0, 0, 0, 1)\}$. This span program is dual to the span program in Example 3.6.

From Definition 3.2, it is immediate that the dual span program $P^\dagger$ satisfies $f_{P^\dagger} = \neg f_P$ and $\mathrm{wsize}(P^\dagger) = \mathrm{wsize}(P)$. (See also [35, Lemma 4.1].)

Alternative constructions of dual span programs are given in [16, 34].

$$|t\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad A = \begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & \top & x_3 & x_4 & x_5 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ \alpha & \beta & \gamma & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & \alpha & \beta & \gamma \end{pmatrix} \end{array}$$

(a)



(b) (c)

Figure 2: (a) A composed span program that computes the function $\mathrm{MAJ}_3(x_1, x_2, \mathrm{MAJ}_3(x_3, x_4, x_5))$, provided α, β, γ are distinct and nonzero. (b) The associated composed graph, with labeled input vertices. (c) Graph for $\mathrm{MAJ}_3(x_1, x_2, x_3) \oplus x_4$, evaluated on input $x = 1100$. The span program for PARITY is from Section 3.4. The eigenvalue-zero eigenvector promised by Theorem 3.5 is supported on the black vertices.

Gate g	$\mathrm{Adv}^{\pm}(g)$
0	0
x_1	1
$x_1 \wedge x_2$	$\sqrt{2}$
$x_1 \oplus x_2$	2
$x_1 \wedge x_2 \wedge x_3$	$\sqrt{3}$
$x_1 \vee (x_2 \wedge x_3)$	$\sqrt{3}$
$x_1 \oplus (x_2 \wedge x_3)$	$1 + \sqrt{2}$

Gate g	$\mathrm{Adv}^{\pm}(g)$
$x_1 \vee (x_2 \oplus x_3)$	$\sqrt{5}$
$x_1 \oplus x_2 \oplus x_3$	3
$(x_1 \wedge x_2 \wedge x_3) \vee (\bar{x}_1 \wedge \bar{x}_2)$	$\sqrt{3 + \sqrt{3}}$
$\text{IF-THEN-ELSE}(x) = (x_1 \wedge x_2) \vee (\bar{x}_1 \wedge x_3)$	2
$\mathrm{MAJ}_3(x) = (x_1 \wedge x_2) \vee ((x_1 \vee x_2) \wedge x_3)$	2
$\mathrm{EQUAL}_3(x) = (x_1 \wedge x_2 \wedge x_3) \vee (\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3)$	$3/\sqrt{2}$
$\mathrm{EXACT}_{2 \text{ of } 3}(x) = \mathrm{MAJ}_3(x) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$	$\sqrt{7}$

Table 1: Inequivalent binary gates on up to three bits $x = x_1 x_2 x_3 \in \{0,1\}^3$, and their adversary bounds.

3.3 Span program composition

Definition 3.9 (Composed graph and span program)**.** Consider span program $P = (n, d, |t\rangle, V_{\mathrm{free}}, \{V_{j,b}\})$ and span programs Q_j and $Q_j^\dagger$ for $j = 1, \ldots, n$. The composed graph is defined by attaching G_P to copies of the G_{Q_j} and $G_{Q_j^\dagger}$ graphs as follows: each input vertex associated to $V_{j,1}$ should be identified with the output vertex of a copy of G_{Q_j}; and each input vertex associated to $V_{j,0}$ should be identified with the output vertex of a copy of $G_{Q_j^\dagger}$. The composed span program is the span program corresponding to this composed graph.

The composed span program computes the composed function $f_P \circ (f_{Q_1}, \ldots, f_{Q_n})$. Two examples are given in Figure 2. Span program composition is studied in more detail in [35, Sec. 4.2].

3.4 Examples: Span programs for three-bit gates

In order to make the above formalism more concrete, we now present span programs for every function $\{0,1\}^3 \to \{0,1\}$. Up to permutation of inputs and complementation of some or all inputs or output, there are fourteen inequivalent three-bit functions. Table 1 lists these functions together with their general adversary bounds, as computed by [23]. For all three-bit functions, the general adversary bound and the adversary bound are equal. Based on our algorithm in Section 5 below, it will follow that the general adversary bound is a lower bound on span program witness size. Therefore the span programs we give here are optimal.

Theorem 3.10. *For every three-bit function $g : \{0,1\}^3 \to \{0,1\}$, there exists a span program P with $f_P = g$, such that the witness size of P with uniform costs is*

$$\mathrm{wsize}(P) = \mathrm{Adv}^{\pm}(g)\,. \tag{3.6}$$

Proof. We begin by giving optimal span programs for the functions MAJ_3, following Example 3.6, and AND (equivalent to OR). We construct optimal span programs for the other functions by combining these two basic span programs.

Claim 3.11. *Let P_{MAJ_3} be the span program from Example 3.6 with $(\alpha,\beta,\gamma) = (1, e^{2\pi i/3}, e^{-2\pi i/3})$. Then if $\mathrm{MAJ}_3(x) = 0$, $\mathrm{wsize}(P_{\mathrm{MAJ}_3}, x) \le 6$, while if $\mathrm{MAJ}_3(x) = 1$, $\mathrm{wsize}(P_{\mathrm{MAJ}_3}, x) \le 2/3$.*

Proof. Substitute P_{MAJ_3} into Definition 3.2. Some of the witnesses are $|w'_{000}\rangle = (1,0)$, $|w'_{100}\rangle = (1,-1)$, and $|w_{110}\rangle = (e^{-i\pi/6}, e^{i\pi/6}, 0)/\sqrt{3}$, $|w_{111}\rangle = (1,1,1)/3$. The other witness vectors are symmetrical. □

The cases $\mathrm{MAJ}_3(x) = 0$ or 1 can be balanced by scaling the input vectors by a factor of $1/\sqrt{3}$, yielding a span program with witness size two.

For functions b and b' on disjoint inputs, the general adversary bound satisfies $\mathrm{Adv}^{\pm}(b \oplus b') = \mathrm{Adv}^{\pm}(b) + \mathrm{Adv}^{\pm}(b')$ and $\mathrm{Adv}^{\pm}(b \vee b') = (\mathrm{Adv}^{\pm}(b)^2 + \mathrm{Adv}^{\pm}(b')^2)^{1/2}$ [6, 24]; we obtain matching upper bounds for span program witness size.

Claim 3.12. *Let $s \in (0,\infty)^n$ and let P be the span program with target vector $|t\rangle = 1$, and input vectors*

$$V_{j,1} = \left\{ \frac{\sqrt{s_j}}{\sqrt[4]{\sum_i s_i^2}} \right\} \quad \textit{and} \quad V_{j,0} = V_{\mathit{free}} = \emptyset\,.$$

Then f_P is the n-bit OR function, and $\mathrm{wsize}_s(P) = \sqrt{\sum_j s_j^2}$.

Proof. On input $x = 0^n$, the unique witness $|w'\rangle = 1$ satisfies

$$\langle t|w'\rangle = 1 \quad \text{and} \quad \|SA^\dagger|w'\rangle\|^2 = \sqrt{\sum_j s_j^2}\,.$$

On input $x = 10^{n-1}$, the unique witness $|w\rangle$, with coefficient $\sqrt[4]{\sum_i s_i^2}/\sqrt{s_1}$ on the $V_{1,1}$ input vector, satisfies

$$A|w\rangle = |t\rangle \quad \text{and} \quad \|S|w\rangle\|^2 = \sqrt{\sum_j s_j^2}\,.$$

The other inputs with Hamming weight one are symmetrical, and the witness size only decreases for larger Hamming weight inputs. □

DeMorgan's laws allow for converting ANDs into ORs. For example, a span program P for $x_1 \wedge x_2$ has

$$|t\rangle = \frac{1}{\sqrt[4]{s_1^2 + s_2^2}} \left(\sqrt{s_1}, \sqrt{s_2}\right), \quad V_{j,1} = \{(1,0)\} \quad \text{and} \quad V_{j,2} = \{(0,1)\}.$$

Then $\mathrm{wsize}_s(P) = \sqrt{s_1^2 + s_2^2}$.

Other span programs can be written by combining the above span programs for AND and for OR. Our span program for PARITY is based on the identity $x_1 \oplus x_2 = (x_1 \wedge \bar{x}_2) \vee (\bar{x}_1 \wedge x_2)$, as follows:

$$|t\rangle = (1), \quad A = \begin{array}{c} \begin{array}{cc} x_1 \wedge \bar{x}_2 & \bar{x}_1 \wedge x_2 \end{array} \\ \begin{pmatrix} \quad 1 \quad & \quad 1 \quad \end{pmatrix} \end{array}.$$

Here, we have introduced a convenient shorthand notation of labeling input vectors by AND gates on subsets of the input bits (such input vectors are termed "grouped input vectors" in [39]). This should be interpreted as indicating a composition with an *unweighted* span program for the AND gate, i. e., as the span program

$$|t\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad A = \begin{array}{c} \begin{array}{cccccc} \top & x_1 & \bar{x}_2 & \top & \bar{x}_1 & x_2 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}. \tag{3.7}$$

Claim 3.13. *The span program P from Equation (3.7) has $f_P(x) = x_1 \oplus x_2$ and $\mathrm{wsize}_s(P) = s_1 + s_2$.*

The proof is an easy calculation.

Optimal span programs for most three-bit functions can be constructed using Claims 3.11, 3.12 and 3.13, e. g., the function $\mathrm{EXACT}_{2\text{ of }3}(x_1, x_2, x_3) = \mathrm{MAJ}_3(x_1, x_2, x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$, so there is a span program with witness size at most

$$\sqrt{\mathrm{wsize}(\mathrm{MAJ}_3)^2 + \mathrm{wsize}(\mathrm{OR}_3)^2} = \sqrt{7}.$$

The only exceptions are $(x_1 \wedge x_2 \wedge x_3) \vee (\bar{x}_1 \wedge \bar{x}_2)$ and the equality function. For the former function, use the span program

$$|t\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad A = \begin{array}{c} \begin{array}{ccc} x_1 \wedge x_2 & x_3 & \bar{x}_1 \wedge \bar{x}_2 \end{array} \\ \begin{pmatrix} \alpha_1 & 0 & \alpha_1 \alpha_2 \\ \alpha_2 & 1 & 0 \end{pmatrix} \end{array}$$

where $\alpha_1 = \sqrt[4]{1+\frac{1}{\sqrt{3}}}$ and $\alpha_2 = \sqrt{\sqrt{3}-1}$. This expands out to

$$|t\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad A = \begin{array}{c} \begin{array}{ccccccc} \top & x_1 & x_2 & x_3 & \top & \bar{x}_2 & \bar{x}_3 \end{array} \\ \begin{pmatrix} \alpha_1 & 0 & 0 & 0 & \alpha_1\alpha_2 & 0 & 0 \\ \alpha_2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array} .$$

The witness size is $\sqrt{3+\sqrt{3}}$, matching the adversary bound.[1] Finally,

Claim 3.14. *Using the condensed notation introduced for PARITY, an optimal span program for the equality function,* $\mathrm{EQUAL}_n(x) = (\wedge_{j=1}^n x_j) \vee (\wedge_{j=1}^n \bar{x}_j)$*, is given by*

$$|t\rangle = (1)\,, \quad A = \begin{array}{c} \begin{array}{cc} \wedge_j x_j & \wedge_j \bar{x}_j \end{array} \\ \begin{pmatrix} \sqrt[4]{n-1} & \sqrt[4]{n-1} \end{pmatrix} \end{array} .$$

This span program has witness size $n/\sqrt{n-1}$, which equals the adversary bound.[2]

Proof. The unweighted AND span program has witness size n on input $x = 1^n$, and $1/(n-|x|)$ otherwise.

For the above span program, therefore, on input $x = 1^n$, the witness $|w\rangle = (1/\sqrt[4]{n-1}, 0)$ gives a witness size of $n/\sqrt{n-1}$, where the factor of n is from the AND gate. The input $x = 0^n$ is symmetrical.

On a false input x, $|w'\rangle = (1)$ gives a witness size of

$$\sqrt{n-1}\left(\frac{1}{n-|x|} + \frac{1}{|x|}\right) \leq \frac{n}{\sqrt{n-1}} .$$

□

This completes the proof of Theorem 3.10. □

Many more optimal span programs, for various four-bit functions, can be found in an earlier arXiv version of this article [39].

[1] A calculation gives that the optimal adversary matrix Γ in Definition 2.1 comes from the matrix

$$\begin{pmatrix} 1 & a & \sqrt{3} \\ 1 & a & \sqrt{3} \\ b & 0 & \sqrt{3} \end{pmatrix}, \quad \text{where} \quad a = \left(\tfrac{1}{2}(5-\sqrt{13-6\sqrt{3}})\right)^{1/2}, \quad b = \tfrac{1}{2}\left(-1-\sqrt{3}+\sqrt{2(8+\sqrt{3})}\right),$$

and the rows correspond to inputs 011, 101, 110, the columns to inputs 000, 001, 111.

[2] The optimal adversary matrix Γ comes from the $2 \times 2n$ matrix

$$\begin{pmatrix} 1 & 1 & \cdots & a & a & \cdots \\ a & a & \cdots & 1 & 1 & \cdots \end{pmatrix},$$

where the rows correspond to inputs 0^n and 1^n, and the columns correspond to inputs of Hamming weight 1 and $n-1$, and $a = 1/(n-1)$.

4 Small-eigenvalue analysis of composed span programs

Definition 4.1 (Formula graph and span program)**.** Given span programs for each gate in a formula φ, span program $P(\varphi)$ is defined as their composition according to the formula. Let $G(\varphi,x)$ be the composed graph on input x, $G(\varphi,x) = G_{P(\varphi)}(x)$—except with length-two, weight-one paths attached to each input vertex in $I(x)$.

Theorem 3.5 establishes that $\varphi(x) = 1$ if and only if $G(\varphi,x)$ has an eigenvalue-zero eigenvector supported on the output vertex. (Attaching length-two paths to the $I(x)$ inputs clearly does not change this property.) Thus if $\varphi(x) = 0$, the graph has a spectral gap centered on zero. The goal of this section is to make Theorem 3.5 more quantitative, so that a phase estimation-based quantum algorithm for evaluating φ can be constructed and analyzed.

Since $G(\varphi,x)$ is a bipartite graph, it suffices to study eigenvectors with non-negative eigenvalues. Let $\lambda \geq 0$ and assume that $|\psi_\lambda\rangle$ is a unit-normalized eigenvalue-λ eigenvector of the adjacency matrix $A_{G(\varphi,x)}$—except possibly not satisfying the eigenvector constraint at the output vertex. The output vertex constraint is the only constraint that changes if we compose φ as an input to another span program, so leaving it out facilitates proofs by induction in the formula depth. Let a_O be the coefficient of $|\psi_\lambda\rangle$ on the output vertex and b_O be the coefficient of $A_{G(\varphi,x)}|\psi_\lambda\rangle$ on the output vertex.

- For $\lambda = 0$, we will place a lower bound on the achievable $|a_O|^2$ when $\varphi(x) = 1$. This will enable the algorithm to detect if $\varphi(x) = 1$ by starting a quantum walk at the output vertex.
- For small enough eigenvalues $\lambda > 0$, we will show that the evaluation $\varphi(x)$ corresponds to the *ratio* of the two output coefficients a_O and b_O. Let $r_O = a_O/b_O$. If $\varphi(x) = 1$, then r_O is large and negative, roughly of order $-1/\lambda$. If $\varphi(x) = 0$, then r_O is small and positive, of order λ. It will follow that if $\varphi(x) = 0$, then there is a significant spectral gap centered on zero of eigenvectors with any support on the output vertex. This spectral gap will prevent the algorithm from outputting false positives.

These two cases are considered in Sections 4.1 and 4.2 below. In our main theorem, Theorem 4.11 in Section 4.3, we will reimpose the output vertex constraint. For small enough $\lambda > 0$, it will give a contradiction when $\varphi(x) = 0$, implying that the graph has a significant spectral gap.

Assume for simplicity that the span programs used in φ, including dual span programs, have no free input vectors, i. e., $V_{\mathrm{free}} = \emptyset$. By Lemma 3.3, this assumption is without loss of generality. The span program $P(\varphi)$ will still have free input vectors at vertices between gates (as shown, e. g., in Figure 2). We do *not* eliminate these free inputs, since doing so would make the graph dense, preventing an efficient implementation of a quantum walk.

4.1 The case $\lambda = 0$

We begin the $\lambda = 0$ analysis by defining two subformula complexity measures. For each node v in a formula φ, let P_v be the associated span program, and let

$$\pi(v) = \max_{\xi} \prod_{w \in \xi} \mathrm{wsize}(P_w)\,, \qquad\qquad \sigma(v) = 1 + \max_{\xi} \sum_{w \in \xi} \frac{1}{\pi(w)}\,, \tag{4.1}$$

where the maximizations are both over simple paths in φ from v through its inputs to a leaf. The leaf itself contributes $+1$ to $\sigma(v)$. Let $\pi(\varphi), \sigma(\varphi)$ be the same quantities evaluated at the root of φ. The second quantity, $\sigma(\varphi)$, accounts for amplitude on free inputs, and will be a constant in our ultimate application.

Lemma 4.2. *If $\varphi(x) = 1$, then $A_{G(\varphi,x)}$ has an eigenvalue-zero eigenvector $|\psi_0\rangle$ with output coefficient a_O satisfying*

$$\frac{\||\psi_0\rangle\|^2}{|a_O|^2} \leq \pi(\varphi)\sigma(\varphi) . \tag{4.2}$$

Proof. The proof is by induction in the depth of the formula.

If the depth is zero, i. e., φ consists of a single variable x_j or its negation, then since $\varphi(x) = 1$, $G(\varphi,x)$ is just a length-two path, with eigenvector $|\psi_0\rangle = (1,0,-1)$. Thus $|a_O|^2 = \frac{1}{2}\||\psi_0\rangle\|^2$ and $\pi(\varphi) = 1$, $\sigma(\varphi) = 2$.

Now for the induction step, assume that for each available input i of the base span program P we have constructed an eigenvalue-zero eigenvector $|\psi_0^{(i)}\rangle$ of the corresponding subformula graph that satisfies the induction hypothesis:

$$\frac{\||\psi_0^{(i)}\rangle\|^2}{|a_O^{(i)}|^2} \leq \pi(v_i)\sigma(v_i) ,$$

where v_i is the gate in φ corresponding to input i. Let $|w\rangle$ be an optimal witness for P, achieving the minimum in Equation (3.1). Let $a_O = -1$ and scale the ith eigenvector by the coefficient $\langle i|w\rangle / a_O^{(i)}$, combining them to give $|\psi_0\rangle$, an eigenvalue-zero eigenvector for $G(\varphi,x)$. Then

$$\begin{aligned}
\frac{\||\psi_0\rangle\|^2}{|a_O|^2} = \||\psi_0\rangle\|^2 = 1 + \sum_i \frac{|\langle i|w\rangle|^2}{|a_O^{(i)}|^2}\||\psi_0^{(i)}\rangle\|^2 \\
&\leq 1 + \||w\rangle\|^2 \cdot \max_i \frac{\||\psi_0^{(i)}\rangle\|^2}{|a_O^{(i)}|^2} \\
&\leq 1 + \mathrm{wsize}(P) \cdot \max_i \pi(v_i) \cdot \max_i \sigma(v_i) \\
&= \pi(\varphi)\sigma(\varphi) .
\end{aligned}$$

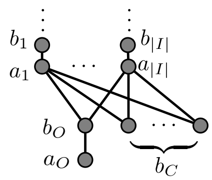
□

4.2 The case $\lambda > 0$

Now consider $\lambda > 0$. Similar to the proof of Lemma 4.2, our goal will be to construct an eigenvalue-λ eigenvector $|\psi_\lambda\rangle$ for $G(\varphi,x)$ inductively, based on joining together appropriately scaled eigenvectors for the input subformula graphs.

Let P be the root span program, with *unit-length* target vector $|t\rangle$ and matrix of input vectors A. Assume that for each input i, we have constructed states $|\psi_\lambda^{(i)}\rangle$ satisfying the eigenvalue-λ constraints for all the ith subgraph's vertices except at the output vertex. Let a_i and b_i be the output coefficients for the ith subgraph. Assume that there exists an $r_i \neq 0$ such that $a_i = r_i b_i$.[3]

[3]In Definition 4.1 we attached length-two paths to vertices in $I(x)$ entirely so that this r_i exists even when input i is a single literal, not another span program, and so that no special case in the analysis is needed.



$$\begin{array}{cc} & \begin{array}{cc} a_O & \overbrace{\quad a_I \quad} \end{array} \\ B_{G_P} = & \left(\begin{array}{c|c} 1 & \langle o| \\ \hline 0 & \\ \vdots & C \\ 0 & \end{array}\right) \begin{array}{l} b_O \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} b_C \end{array} \end{array}$$

Figure 3: The graph G_P with coefficients $a_O, b_O, |a_I\rangle, |b_I\rangle, |b_C\rangle$ indicated, in the case $|t\rangle = (1, 0, \ldots, 0)$.

Let $\langle o| = \langle t|A$ and let $C = (1 - |t\rangle\langle t|)A$, so $A = C + |t\rangle\langle o|$. Let $|a_I\rangle$ and $|b_I\rangle$ denote the vectors of scaled output coefficients for the input subgraphs, and let $|b_C\rangle$ be the other row-vertex coefficients of $|\psi_\lambda\rangle$ in the subgraph G_P. The notation is meant to be suggestive: O for "output," I for "input" and C for "constraints." Figure 3 places these coefficients on G_P. From Equation (3.3), the additional eigenvalue-λ eigenvector equations imposed by composition with G_P are

$$\lambda |b_C\rangle = C|a_I\rangle\,, \tag{4.3a}$$
$$\lambda b_O = \langle o|a_I\rangle + a_O\,, \tag{4.3b}$$
$$\lambda |a_I\rangle = |b_I\rangle + |o\rangle b_O + C^\dagger |b_C\rangle\,. \tag{4.3c}$$

Here, we have left off the eigenvalue-λ constraint at the output vertex, $\lambda a_O = b_O$, since this constraint will be modified by later composition steps, as in (4.3c) for the input subgraphs. Letting $r = \sum_i r_i |i\rangle\langle i|$, our goal is to solve Equations (4.3) with the subgraph output coefficients constrained by

$$|a_I\rangle = r|b_I\rangle\,. \tag{4.4}$$

Our main technical theorem is:

Theorem 4.3. *Fix an input x to φ, and let $\tilde{x}$ be the input to the root span program P. For each input index i, let $s_i = r_i/\lambda$ if the input evaluates to 0 and let $s_i = -r_i^{-1}/\lambda$ if it evaluates to 1. Assume that each $s_i > 0$.*

Then there exist $\varepsilon_P, \delta_P > 0$, constants depending only on the span program P, such that, provided $0 < \lambda \leq \varepsilon_P \min\{1, 1/\max_i s_i\}$, there exists an $r_O \neq 0$ such that any solution to Equations (4.3) and (4.4) satisfies $a_O = r_O b_O$. Moreover, if s_O is defined as be either r_O/λ or $-r_O^{-1}/\lambda$, depending on whether $\varphi(x)$ is 0 or 1, respectively, s_O satisfies

$$0 < s_O < \mathrm{wsize}(P, \tilde{x}) \cdot \max_i s_i \cdot (1 + \delta_P \lambda^2 \max_i s_i^2) + \delta_P\,. \tag{4.5}$$

The proof of this theorem has two main parts. First, in Section 4.2.1 below, we will derive several alternative expressions for span program witness size. Second, in Section 4.2.2, we will compute the Taylor series of the inverse of a certain matrix, in order to solve Equations (4.3) and (4.4). The lowest-order term in the solution will be related to the witness size of P, and we will bound the higher-order terms.

We will rely on the following notion of a matrix pseudoinverse:

Definition 4.4. For a matrix M with singular-value decomposition $M = \sum_k m_k |k\rangle\langle k'|$ with all $m_k > 0$ and sets of orthonormal vectors $\{|k\rangle\}$, $\{|k'\rangle\}$, let $M^+ = \sum_k m_k^{-1} |k'\rangle\langle k|$, the *Moore-Penrose pseudoinverse* of M.

If M is invertible, then $M^+ = M^{-1}$. In general, $MM^+ = \sum_k |k\rangle\langle k|$ is the orthogonal projection onto the range of M. If $|b\rangle$ is in the range of M, then

$$M^+|b\rangle = \arg\min_{|x\rangle : M|x\rangle = |b\rangle} \| |x\rangle \| .$$

A convenient identity, immediate from the singular-value decomposition, is $M^+ = (M^\dagger M)^+ M^\dagger = M^\dagger (MM^\dagger)^+$. The norms of pseudoinverses can be bounded using the following claim:

Claim 4.5. *For matrices A and B with* $\mathrm{Range}(B^\dagger) \subseteq \mathrm{Range}(A)$ *(i. e., $B = BAA^+$), $\|A(BA)^+\| \leq \|B^+\|$ and $\|(BA)^+\| \leq \|A^+\| \|B^+\|$.*

Proof. Since $A(BA)^+ = A(BA)^+ BB^+ = \big[A(BA)^+(BA)A^+\big] B^+$ and the bracketed term is a projection, $\|A(BA)^+\| \leq \|B^+\|$. The second claimed bound follows, since $(BA)^+ = A^+ \cdot A(BA)^+$. ☐

4.2.1 Equivalent expressions for the witness size

Lemma 4.6. *For S any positive-definite, diagonal matrix, let*

$$\mathrm{wsize}_S(P,x) = \begin{cases} \min_{|w\rangle : A\Pi|w\rangle = |t\rangle} \|S|w\rangle\|^2 & \text{if } f_P(x) = 1, \\ \min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi A^\dagger |w'\rangle = 0}} \|SA^\dagger |w'\rangle\|^2 & \text{if } f_P(x) = 0. \end{cases} \tag{4.6}$$

Then if $f_P(x) = 1$,

$$\begin{aligned} \mathrm{wsize}_S(P,x) &= \min_{\substack{|w\rangle : \langle w|\Pi|w\rangle = 1 \\ C\Pi S^{-1}|w\rangle = 0}} |\langle o|\Pi S^{-1}|w\rangle|^{-2} \\ &= \|(A\Pi S^{-1})^+|t\rangle\|^2 \\ &= \|(\Pi - (C\Pi S^{-1})^+ C\Pi S^{-1}) S^{-1}|o\rangle\|^{-2} \end{aligned} \tag{4.7}$$

and, if $f_P(x) = 0$, letting $\overline{\Pi} = 1 - \Pi$, $\Delta = C\Pi(C\Pi)^+$ and $\overline{\Delta} = 1 - \Delta$,

$$\begin{aligned} \mathrm{wsize}_S(P,x) &= \min_{\substack{|w'\rangle : \|SA^\dagger|w'\rangle\| = 1 \\ \Pi A^\dagger |w'\rangle = 0}} |\langle t|w'\rangle|^{-2} \\ &= \big\|\big(1 + (\overline{\Pi}(AS)^+ AS - 1)^+ \Pi\big)(AS)^+|t\rangle\big\|^{-2} \\ &= \|(1 - (\overline{\Delta}CS)^+ \overline{\Delta}CS) S (1 - C^\dagger (\Pi C^\dagger)^+)|o\rangle\|^2 . \end{aligned} \tag{4.8}$$

Moreover,

$$|w_S^*\rangle = \arg\min_{|w\rangle : A\Pi|w\rangle = |t\rangle} \|S|w\rangle\|^2 = S^{-1}(A\Pi S^{-1})^+|t\rangle \text{ has norm } \||w_S^*\rangle\| = O(1) \text{ and}$$

$$|w_S'^*\rangle = \arg\min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi A^\dagger |w'\rangle = 0}} \|SA^\dagger|w'\rangle\|^2 = |t\rangle - (\Pi C^\dagger)^+ \Pi |o\rangle - (SC^\dagger \overline{\Delta})^+ S (1 - C^\dagger (\Pi C^\dagger)^+)|o\rangle$$

has norm $\||w_S'^*\rangle\| = O(1)$.

Proof. Assume $f_P(x) = 1$. Since $|t\rangle$ is in the range of $A\Pi$,

$$\min_{|w\rangle : A\Pi S^{-1}|w\rangle = |t\rangle} \| |w\rangle \|^2 = \|(A\Pi S^{-1})^+ |t\rangle\|^2 .$$

This is the second of the three expressions for $\mathrm{wsize}_S(P,x)$ in Equation (4.7). The other two expressions follow by basic geometry: in general, for a vector $|v\rangle \in \mathbb{R}^m$ and a subspace V,

$$\min_{|w\rangle \in V : \langle v|w\rangle = 1} \| |w\rangle \|^2 = \min_{w \in V : \| |w\rangle \| = 1} |\langle v|w\rangle|^{-2} = \|\Pi_V |v\rangle\|^{-2} ,$$

where Π_V is the orthogonal projection onto V. Substitute $|v\rangle = S^{-1}\Pi|o\rangle$ and $V = \mathrm{Ker}(C\Pi S^{-1})$ to complete the proof of Equation (4.7).

Next assume $f_P(x) = 0$. As above,

$$\min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi A^\dagger |w'\rangle = 0}} \|SA^\dagger |w'\rangle\|^2 = \min_{\substack{|w'\rangle : \|SA^\dagger |w'\rangle\| = 1 \\ \Pi A^\dagger |w'\rangle = 0}} |\langle t|w'\rangle|^{-2} .$$

Now, without loss of generality, $|t\rangle \in \mathrm{Range}(A) = \mathrm{Range}(AS)$, since otherwise f_P is false on every input. Therefore, $\langle t|w'\rangle = \langle t|(SA^\dagger)^+(SA^\dagger)|w'\rangle = \langle t|(SA^\dagger)^+|w\rangle$ if $|w\rangle = SA^\dagger|w'\rangle$. We want to find the length-one vector $|w\rangle$ that is in the range of $SA^\dagger$ and also of $\overline{\Pi}$, and that maximizes $|\langle t|(SA^\dagger)^+|w\rangle|^2$. The answer is clearly the normalized projection of $(AS)^+|t\rangle$ onto the intersection $\mathrm{Range}(SA^\dagger) \cap \mathrm{Range}(\overline{\Pi})$. In general, given two projections Π_1 and Π_2, the projection onto the intersection of their ranges can be written $1 - (\Pi_1\Pi_2 - 1)^+(\Pi_1\Pi_2 - 1)$. Substituting $\Pi_1 = \overline{\Pi}$ and $\Pi_2 = (AS)^+AS$ gives the second claimed expression.

Finally, we show that

$$\min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi A^\dagger |w'\rangle = 0}} \|SA^\dagger |w'\rangle\|^2 = \|(1 - (\overline{\Delta}CS)^+\overline{\Delta}CS)S(1 - C^\dagger(\Pi C^\dagger)^+)|o\rangle\|^2 .$$

Since $f_P(x) = 0$, $|t\rangle$ does not lie in the span of the available input vectors, $|t\rangle \notin \mathrm{Range}(A\Pi)$, or equivalently $\Pi|o\rangle \in \mathrm{Range}(\Pi C^\dagger)$. Therefore, there exists a vector $|w'\rangle = |t\rangle + |b_C\rangle$ that is orthogonal to the span of the columns of $A\Pi$ and has inner product one with $|t\rangle$. Any such $|b_C\rangle$ has the form

$$|b_C\rangle = -(\Pi C^\dagger)^+\Pi|o\rangle + \overline{\Delta}|v\rangle ,$$

where $|v\rangle$ is an arbitrary vector with $\langle t|v\rangle = 0$. We want to choose $|v\rangle$ to minimize the squared length of

$$\begin{aligned} \overline{\Pi}SA^\dagger|w'\rangle &= S\overline{\Pi}(|o\rangle + C^\dagger|b_C\rangle) \\ &= S(1 - C^\dagger(\Pi C^\dagger)^+)|o\rangle + SC^\dagger\overline{\Delta}|v\rangle . \end{aligned}$$

The answer is clearly the squared length of $S(1 - C^\dagger(\Pi C^\dagger)^+)|o\rangle$ projected to the orthogonal complement of the range of $SC^\dagger\overline{\Delta}$, as claimed. This corresponds to setting $|v\rangle = -(SC^\dagger\overline{\Delta})^+S(1 - C^\dagger(\Pi C^\dagger)^+)|o\rangle$.

The norms of $|w^*_S\rangle$ and $|w'^*_S\rangle$ are bounded using Claim 4.5. □

It is immediate that $\mathrm{wsize}_s(P,x)$ is monotone increasing in each input complexity s_i. In the other direction, we can show the following estimates:

Lemma 4.7. *Let S and T be any positive-definite diagonal matrices. Then*

$$\operatorname{wsize}_{S\sqrt{1+T}}(P,x) \leq \operatorname{wsize}_S(P,x) \cdot (1 + \|T\|)\,, \tag{4.9}$$

$$\operatorname{wsize}_{\sqrt{S^2+T^2}}(P,x) \leq \operatorname{wsize}_S(P,x) + O(\|T\|^2)\,. \tag{4.10}$$

Proof. Equation (4.9) is immediate from the definition in Equation (4.6). To derive Equation (4.10), first note that $\|\sqrt{S^2+T^2}|v\rangle\|^2 = \|S|v\rangle\|^2 + \|T|v\rangle\|^2$. Then when $f_P(x) = 1$,

$$\min_{|w\rangle : A\Pi|w\rangle = |t\rangle} (\|S|w\rangle\|^2 + \|T|w\rangle\|^2) \leq \|S|w_S^*\rangle\|^2 + \|T|w_S^*\rangle\|^2 = \operatorname{wsize}_S(P,x) + O(\|T\|^2)\,,$$

where we have used that

$$|w_S^*\rangle = \operatorname*{arg\,min}_{|w\rangle : A\Pi|w\rangle = |t\rangle} \|S|w\rangle\|^2$$

has norm $\||w_S^*\rangle\| = O(1)$ by Lemma 4.6. The argument when $f_P(x) = 0$ is similar:

$$\min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi A^\dagger |w'\rangle = 0}} (\|SA^\dagger|w'\rangle\|^2 + \|TA^\dagger|w'\rangle\|^2) \leq \operatorname{wsize}_S(P,x) + \|TA^\dagger|w_S'^*\rangle\|^2\,,$$

where

$$|w_S'^*\rangle = \operatorname*{arg\,min}_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi A^\dagger |w'\rangle = 0}} \|SA^\dagger|w'\rangle\|^2$$

has $O(1)$ norm by Lemma 4.6. □

4.2.2 Proof of Theorem 4.3

Our proof will require two further matrix identities. First is a special case of the Woodbury matrix inversion formula [19]: for matrices A and B of the correct sizes, if A and $1 + B^\dagger A^{-1} B$ are invertible, then so is $A + BB^\dagger$, and

$$(A + BB^\dagger)^{-1} = A^{-1} - A^{-1}B(1 + B^\dagger A^{-1} B)^{-1} B^\dagger A^{-1}\,. \tag{4.11}$$

Multiply the right-hand side by $A + BB^\dagger$ to verify the identity. Second is the following claim:

Claim 4.8. *Let $A \succcurlyeq 0$, $B \succ 0$ be positive semi-definite and positive definite matrices, respectively, of the same size, and let $\Delta = AA^+$ and $\overline{\Delta} = 1 - \Delta$ be projections onto the range of A and its complement. Denote by $B_{\overline{\Delta}}^{-1}$ the pseudoinverse $(\overline{\Delta} B \overline{\Delta})^+$. Assume that $A^+ \succcurlyeq A^+(B - BB_{\overline{\Delta}}^{-1}B)A^+$. Then $B - A$ is invertible and*

$$(B - A)^{-1} = B_{\overline{\Delta}}^{-1} - (1 - B_{\overline{\Delta}}^{-1}B)\big(A - \Delta(B - BB_{\overline{\Delta}}^{-1}B)\Delta\big)^+(1 - BB_{\overline{\Delta}}^{-1})\,. \tag{4.12}$$

Proof. By the assumption $A^+ \succcurlyeq A^+(B - BB_{\overline{\Delta}}^{-1}B)A^+$, $1 - \sqrt{A^+}(B - BB_{\overline{\Delta}}^{-1}B)\sqrt{A^+}$ is invertible on Δ. To finish, multiply the claimed expression for $(B - A)^{-1}$ by $B - A$, and simplify using $B_{\overline{\Delta}}^{-1}B = \overline{\Delta} + B_{\overline{\Delta}}^{-1}B\Delta$. □

With the necessary technical tools in place, we are ready to begin our proof.

Proof of Theorem 4.3. Let $\Pi = \sum_{i \in I(\tilde{x})} |i\rangle\langle i|$ be the projection onto the available inputs i, and let

$$\overline{\Pi} = \sum_{i \in I \smallsetminus I(\tilde{x})} |i\rangle\langle i|$$

be the projection onto the other inputs. With $r = \sum_i r_i |i\rangle\langle i|$ and $s = \sum_i s_i |i\rangle\langle i|$, we have

$$s = \frac{1}{\lambda}(-\Pi r^{-1} + \overline{\Pi} r) .$$

Let

$$\tilde{r} = (r^{-1} - \lambda)^{-1} \quad \text{and} \quad \tilde{s} = \frac{1}{\lambda}(-\Pi \tilde{r}^{-1} + \overline{\Pi} \tilde{r}) .$$

Whereas r_i is the ratio between a_i and the amplitude b_i arriving from deeper in the formula, $-\tilde{r}_i$ can be seen to be the ratio between a_i and the amplitude arriving from shallower vertices. We first claim that $\tilde{r}$ and $\tilde{s}$ are well-defined:

Lemma 4.9. *Assume that $0 < \lambda \leq 1/\sqrt{2}$ and $0 < s_i \lambda \leq 1/\sqrt{2}$ for all $i \in I$. Then $\tilde{r} = (r^{-1} - \lambda)^{-1}$ exists. Moreover, for each input i, $s_i < \tilde{s}_i \leq s_i + 1$.*

Proof. If input $i \in I(\tilde{x})$, then using $0 < s_i = -1/(\lambda r_i)$, $\tilde{r}_i = (r_i^{-1} - \lambda)^{-1}$ exists, and $\tilde{s}_i = -1/(\lambda \tilde{r}_i) = s_i + 1$. If input $i \notin I(\tilde{x})$, then using $r_i = s_i \lambda < 1$, again $\tilde{r}_i$ exists, and $\tilde{s}_i = \tilde{r}_i / \lambda = s_i/(1 - s_i \lambda^2)$. It follows that $s_i < \tilde{s}_i \leq s_i + 1$. □

Next we solve Equations (4.3) and (4.4) to derive an exact expression for r_O:

Lemma 4.10. *Any solution to Equations (4.3) and (4.4) has $a_O = r_O b_O$, where*

$$r_O = \lambda + \langle o| \Big(\tilde{r} - \lambda \tilde{r} C^\dagger (\lambda^2 + \lambda C \tilde{r} C^\dagger)^{-1} C \tilde{r} \Big) |o\rangle , \tag{4.13}$$

provided that $\tilde{r} = (r^{-1} - \lambda)^{-1}$ exists and $\lambda^2 + \lambda C \tilde{r} C^\dagger$ is invertible.

Proof. Substituting Equations (4.4) and (4.3a) into (4.3c), and rearranging terms gives

$$\big(\lambda - r^{-1} - \frac{1}{\lambda} C^\dagger C \big) |a_I\rangle = |o\rangle b_O .$$

From Equation (4.3b),

$$a_O = \lambda b_O - \langle o|a_I\rangle = \lambda b_O + \langle o| \left(\tilde{r}^{-1} + \frac{1}{\lambda} C^\dagger C \right)^{-1} |o\rangle b_O .$$

Equation (4.13) then follows by the Woodbury identity, Equation (4.11), provided that $\tilde{r}$ and $(1 + (1/\lambda) C \tilde{r} C^\dagger)^{-1}$ exist. □

Let $S = \sqrt{\tilde{s}}$, $y = CS^{-1}\Pi$, $\bar{y} = CS\overline{\Pi}$, $Y = yy^\dagger$ and $\bar{Y} = \bar{y}\bar{y}^\dagger$. Observe that $\|\bar{Y}\| = O(\|\tilde{s}\|)$ and also, since $Y^+ = (y^\dagger)^+ y^+$, by Claim 4.5 $\|Y^+\| = O(\|\tilde{s}\|)$. Let

$$X = \lambda^2 + \lambda C\tilde{r}C^\dagger = \lambda^2(1+\bar{Y}) - Y\,.$$

Provided that X is invertible, we have from Equation (4.13) that

$$r_O = \lambda - \frac{1}{\lambda}\langle o|S^{-2}\Pi|o\rangle + \lambda\langle o|S^2\overline{\Pi}|o\rangle - \begin{pmatrix} \frac{1}{\lambda}\langle o|S^{-1}y^\dagger X^{-1} y S^{-1}|o\rangle + \lambda^3\langle o|S\bar{y}^\dagger X^{-1}\bar{y}S|o\rangle \\ -\lambda\big(\langle o|S^{-1}y^\dagger X^{-1}\bar{y}S|o\rangle + \langle o|S\bar{y}^\dagger X^{-1} y S^{-1}|o\rangle\big)\end{pmatrix}. \tag{4.14}$$

To evaluate X^{-1}, we will apply Claim 4.8 with $A = Y$ and $B = \lambda^2(1+\bar{Y})$. By a singular-value decomposition $\overline{\Delta}\bar{y} = \sum_k m_k |k\rangle\langle k'|$, we obtain

$$\bar{y}^\dagger(1+\bar{Y})_{\Delta}^{-1}\bar{y} = \sum_k \frac{m_k^2}{1+m_k^2}|k'\rangle\langle k'| \prec 1\,.$$

Therefore, using our bounds on $\|Y^+\|$ and $\|\bar{y}\|$, provided that $\lambda\|\tilde{s}\|$ is at most a sufficiently small constant,

$$1 - \sqrt{A^+}(B - BB_{\Delta}^{-1}B)\sqrt{A^+} = 1 - \lambda^2\sqrt{Y^+}(1+\bar{Y} - \bar{Y}(1+\bar{Y})_{\Delta}^{-1}\bar{Y})\sqrt{Y^+} \succ 0\,.$$

By Claim 4.8, $X = B - A$ is invertible.

Substitute Equation (4.12) and the series expansion

$$\big(A - \Delta(B - BB_{\Delta}^{-1}B)\Delta\big)^+ = \sum_{j=0}^{\infty}\sqrt{A^+}\Big[\sqrt{A^+}(B - BB_{\Delta}^{-1}B)\sqrt{A^+}\Big]^j\sqrt{A^+}$$

into Equation (4.14), and collect terms with like powers of λ. The lowest-order term is

$$-\frac{1}{\lambda}\langle o|S^{-2}\Pi|o\rangle + \frac{1}{\lambda}\langle o|S^{-1}y^\dagger A^+ y S^{-1}|o\rangle = -\frac{1}{\lambda}\|(1-y^+y)S^{-1}\Pi|o\rangle\|^2$$

since $y^\dagger A^+ y = y^\dagger(y^\dagger)^+y^+y = y^+y$. Including the other terms gives

$$\begin{aligned} r_O = &-\frac{1}{\lambda}\|(1-y^+y)S^{-1}\Pi|o\rangle\|^2 + \lambda\Big(1 + \langle v|V|v\rangle + \|(y^\dagger)^+S^{-1}|o\rangle\|^2\Big) \\ &+\sum_{j=0}^{\infty}\lambda^{3+2j}\Big\|\big(y^+(1+\bar{y}V\bar{y}^\dagger)(y^\dagger)^+\big)^{j/2}y^+\big(\bar{y}V|v\rangle - (y^\dagger)^+S^{-1}|o\rangle\big)\Big\|^2 \end{aligned} \tag{4.15}$$

where we have let $V = 1 - \bar{y}^\dagger(1+\bar{Y})_{\Delta}^{-1}\bar{y} \succ 0$ and $|v\rangle = (S\overline{\Pi} - \bar{y}^\dagger(y^\dagger)^+S^{-1})|o\rangle$. Observe that all coefficients, except the first, are non-negative, and that the coefficient of λ^{3+2j} is of order $O(\|\tilde{s}\|^{3+2j})$.

When $\varphi(x) = f_P(\tilde{x}) = 1$, the first term in (4.15) is $-1/(\lambda\,\mathrm{wsize}_S(P,\tilde{x}))$, using the last expression of Equation (4.7) for $\mathrm{wsize}_S(P,\tilde{x})$. Therefore,

$$-\lambda\|\tilde{s}\|r_O \geq \mathrm{wsize}(P)^{-1} - O(\lambda^2\|\tilde{s}\|^2) > 0\,,$$

so $s_O = -1/(\lambda r_O)$ satisfies

$$\mathrm{wsize}_S(P,\tilde{x}) \le s_O \le \mathrm{wsize}_S(P,\tilde{x})\big(1+O(\lambda^2\|\tilde{s}\|^2)\big)\,. \tag{4.16}$$

In particular, since each $\tilde{s}_i \le s_i + 1$ (Lemma 4.9), by Lemma 4.7,

$$s_O \le \mathrm{wsize}(P,\tilde{x})\|s\|\big(1+O(\lambda^2\|s\|^2\big)+O(1)\,.$$

Assume then that $\varphi(x) = f_P(\tilde{x}) = 0$, i. e., $\Pi|o\rangle \in \mathrm{Range}(\Pi C^\dagger)$. The first term in (4.15) is zero. Consider the second term. Using $\Pi C^\dagger(\Pi C^\dagger)^+|o\rangle = \Pi|o\rangle$ and $(y^\dagger)^+ S^{-1}|o\rangle = (\Pi S^{-1}C^\dagger)^+ S^{-1}\Pi|o\rangle = (\Pi C^\dagger)^+|o\rangle$,

$$\begin{aligned}|v\rangle &= (S\overline{\Pi} - \bar{y}^\dagger(\Pi C^\dagger)^+)|o\rangle + S\Pi(1-C^\dagger(\Pi C^\dagger)^+)|o\rangle\\ &= S(1-C^\dagger(\Pi C^\dagger)^+)|o\rangle\,.\end{aligned}$$

Furthermore, from the singular-value decomposition of $M = \overline{\Delta}\bar{y}$, one infers that

$$V = 1 - M^\dagger(1+MM^\dagger)^{-1}M = (1-M^+M) + M^+(1-(1+MM^\dagger)^{-1})(M^\dagger)^+\,.$$

Substituting this into Equation (4.15), the second term becomes

$$\lambda\Big(\|(1-(\overline{\Delta}\bar{y})^+(\overline{\Delta}\bar{y}))|v\rangle\|^2 + 1 + \|(\Pi C^\dagger)^+|o\rangle\|^2 + \|(1-(1+\overline{\Delta}\bar{Y}\overline{\Delta})^{-1})^{1/2}(\bar{y}^\dagger\overline{\Delta})^+|v\rangle\|^2\Big)\,.$$

Moreover, as $\overline{\Delta}C\Pi = 0$, $\overline{\Delta}\bar{y} = \overline{\Delta}CS\overline{\Pi} = \overline{\Delta}CS$. Therefore, $\|(1-(\overline{\Delta}\bar{y})^+(\overline{\Delta}\bar{y}))|v\rangle\|^2 = \mathrm{wsize}_S(P,\tilde{x})$, using the last expression of Equation (4.8) for $\mathrm{wsize}_S(P,\tilde{x})$. Also,

$$\|(1-(1+\overline{\Delta}\bar{Y}\overline{\Delta})^{-1})^{1/2}(\bar{y}^\dagger\overline{\Delta})^+|v\rangle\|^2 = O(1)\,,$$

since $\|(\bar{y}^\dagger\overline{\Delta})^+|v\rangle\| = O(\|S(\overline{\Delta}\bar{y})^+\|) = O(1)$ (Claim 4.5). Thus $s_O = r_O/\lambda$ satisfies

$$\begin{aligned}\mathrm{wsize}_S(P,\tilde{x}) + 1 \le s_O &\le \mathrm{wsize}_S(P,\tilde{x}) + O(1) + O(\lambda^2\|\tilde{s}\|^3)\\ &\le \mathrm{wsize}_s(P,\tilde{x}) + O(\lambda^2\|s\|^3) + O(1)\,,\end{aligned} \tag{4.17}$$

where we have used $\|\tilde{s}\| \le \|s\|+1$ (Lemma 4.9) and $\mathrm{wsize}_S(P,\tilde{x}) \le \mathrm{wsize}_s(P,\tilde{x}) + O(1)$ (Lemma 4.7).

Thus in either case $\varphi(x) = 1$ or $\varphi(x) = 0$ we have found

$$0 < s_O \le \mathrm{wsize}(P,\tilde{x})\|s\|\big(1+O(\lambda^2\|s\|^2)\big)+O(1)\,,$$

as desired. This concludes the proof of Theorem 4.3. □

4.3 Span program spectral analysis of φ

To study the spectra of the composed graphs $G(\varphi,x)$ of Definition 4.1, and argue that in the case $\varphi(x)=0$ there is an $\Omega(1/\pi(\varphi))$ spectral gap, we will solve the recursion set up by Theorem 4.3:

Theorem 4.11. *Let φ be a formula. For each node v, let P_v be the associated span program and assume that* $\mathrm{wsize}(P_v) \geq 1$. *Let $\pi(v)$, $\sigma(v)$ be defined as in Equation (4.1), and let $\bar{\sigma}(v)$ be given by*

$$\bar{\sigma}(v) = \max_{\xi} \sum_{w \in \xi} \pi(w)^2 , \tag{4.18}$$

where the maximization is over simple paths in φ from v through its inputs to a leaf. Let $\pi(\varphi), \sigma(\varphi), \bar{\sigma}(\varphi)$ be the same quantities evaluated at the root of φ. Let $C = \max\{\sigma(\varphi), \bar{\sigma}(\varphi)/\pi(\varphi)^2\}$. For a constant span program P, let δ_P, ε_P be the constants from Theorem 4.3. Let $\delta = \max\{1, \max_v \delta_{P_v}\}$ and $\varepsilon = 1/(8C^3\delta^2)$.

For an input x, let $G(\varphi,x)$ be the composed graph from Definition 4.1, and let $\tilde{G}(\varphi,x)$ be the same except with the weights of edges to the final output vertex scaled by a factor of $f = 1/\sqrt{C\pi(\varphi)}$. (Equivalently, scale the target vector of the root span program by the same factor f.) Then,

- *If $\varphi(x) = 1$, there exists an eigenvalue-zero eigenvector $|\tilde{\psi}_0\rangle$ of the adjacency matrix $A_{\tilde{G}(\varphi,x)}$ with at least half of its weight on the output vertex, i. e., $|\tilde{a}_O|^2/\||\tilde{\psi}_0\rangle\|^2 \geq 1/2$.*
- *If $\varphi(x) = 0$, then $A_{\tilde{G}(\varphi,x)}$ does not have any eigenvalue-λ eigenvectors with nonzero output coefficients $\tilde{a}_O$ or $\tilde{b}_O$ for*

$$|\lambda| \leq \frac{1}{\pi(\varphi)} \min\Big\{ \sqrt{\varepsilon/\delta}, \min\{1, \min_v \varepsilon_{P_v}\}/(2\delta C) \Big\} .$$

Proof. If $\varphi(x) = 1$, then by Lemma 4.2, $G(\varphi,x)$ has an eigenvalue-zero eigenvector $|\psi_0\rangle$ with output coefficient a_O satisfying

$$\frac{|a_O|^2}{\||\psi_0\rangle\|^2} \geq \frac{1}{\pi(\varphi)\sigma(\varphi)} \geq f^2 .$$

Let $|\tilde{\psi}_0\rangle$ be the same vector, except with output coefficient $\tilde{a}_O = \frac{1}{f} a_O$. Then $|\tilde{\psi}_0\rangle$ is an eigenvalue-zero eigenvector of $\tilde{G}(\varphi,x)$, and

$$\frac{|\tilde{a}_O|^2}{\||\tilde{\psi}_0\rangle\|^2} = \frac{\frac{1}{f^2}|a_O|^2}{\big(\frac{1}{f^2} - 1\big)|a_O|^2 + \||\psi_0\rangle\|^2} \geq \frac{1}{2 - f^2} .$$

Now assume that $\varphi(x) = 0$. By Theorem 3.5, there does not exist any eigenvalue-zero eigenvector with $\tilde{a}_O \neq 0$. Also $\tilde{b}_O = 0$ necessarily at $\lambda = 0$ by the constraint $\lambda \tilde{a}_O = f \tilde{b}_O$.

For $\lambda \neq 0$, we need to solve the recursion from Theorem 4.3, then impose the final output vertex constraint. The base cases are: at a false input, by the eigenvalue-λ constraint for a path of length one, $a_i = \lambda b_i$, so $s_i = 1$; and at a true input, by the eigenvalue-λ constraints for a path of length two, similarly $s_i = 1/(1-\lambda^2)$. From Equation (4.5), we have that provided $|\lambda| \leq \varepsilon_P \min\{1, 1/\max_i s_i\}$,

$$s_O < \mathrm{wsize}(P) \cdot \max_i s_i \cdot (1 + \delta_P \lambda^2 \max_i s_i^2) + \delta_P .$$

The following claim solves this recursion, somewhat conservatively, assuming that $|\lambda|$ is sufficiently small.

Claim 4.12. *Define a quantity $s(v)$ recursively by*

$$s(v) = \begin{cases} 2 & \text{if } v \text{ is a leaf of } \varphi, \\ \operatorname{wsize}(P_v) \max_c s(c) \Big(1 + \varepsilon \frac{\max_c s(c)^2}{\pi(\varphi)^2}\Big) + \delta & \text{otherwise,} \end{cases} \tag{4.19}$$

where in the second case the maximizations are over v's inputs in φ. Then $s(v) \le 2\delta C \pi(v)$.

Proof. For brevity, let $w_v = \operatorname{wsize}(P_v) \ge 1$. Define a quantity $s'(v)$ recursively by

$$s'(v) = w_v \max_c s'(c) + \delta \, ,$$

for an internal node v, and s' at a leaf is 2. Then by induction, one finds $s'(v) \le \delta \pi(v) \sigma(v)$. (For the induction step, use $s'(v)/\delta \le w_v \max_c \pi(c)\sigma(c) + 1 \le \pi(v) \max_c \big(\sigma(c) + 1/\pi(v)\big) = \pi(v)\sigma(v)$.) Moreover, from their definitions, clearly $s(v) \ge s'(v)$ always.

Let $D = 2\delta C$ and assume by induction that $s(c) \le D\pi(c)$ for each input c. Then we have from Equation (4.19)

$$s(v) \le \frac{w_v \max_c s(c) + \delta}{1 - \varepsilon D^2 \frac{\max_c \pi(c)^2}{\pi(\varphi)^2}} \le \frac{w_v \max_c s(c) \big(1 + \frac{\delta}{w_v \max_c s'(c)}\big)}{1 - \varepsilon D^2 \frac{\pi(v)^2}{\pi(\varphi)^2}} \le \frac{\max_c \frac{s(c)}{s'(c)} s'(v)}{1 - \varepsilon D^2 \frac{\pi(v)^2}{\pi(\varphi)^2}} \, .$$

The above inequality places a recursive bound on the ratio $s(v)/s'(v)$. Solve it, using $(1-a)(1-b) \ge 1 - a - b$ for $ab \ge 0$, to find

$$s(v) \le \frac{s'(v)}{1 - \varepsilon D^2 \frac{\bar\sigma(v)}{\pi(\varphi)^2}} \, .$$

Using $\bar\sigma(v) \le \bar\sigma(\varphi) \le C\pi(\varphi)^2$ and $\varepsilon = 1/(2D^2 C)$, therefore $s(v) \le 2s'(v) \le 2\delta C \pi(v)$, as desired. □

Therefore, provided that $|\lambda| \le \sqrt{\varepsilon / \max_v \delta_{P_v}} / \pi(\varphi)$ and $|\lambda| \le \min_v \varepsilon_{P_v} / (2\delta C \pi(\varphi))$, we obtain that either $a_O = b_O = 0$ or the final output ratio $s_O = a_O/(\lambda b_O)$ satisfies $0 < s_O < 2\delta C \pi(\varphi)$. Here a_O and b_O are the output coefficients for the unscaled graph $G(\varphi, x)$; the same solution works in $\tilde G(\varphi, x)$, except with output coefficients $\tilde a_O = a_O / f$ and $\tilde b_O = b_O$.

We have not yet used the eigenvector equation at the output vertex, $\lambda \tilde a_O = f \tilde b_O$. Combining this equation with $f \tilde a_O / (\lambda \tilde b_O) < 2\delta C \pi(\varphi)$ implies $|\lambda| > f/\sqrt{2\delta C \pi(\varphi)} = 1/(\sqrt{2\delta} C \pi(\varphi))$. This contradicts the assumption $|\lambda| \le 1/(2\delta C \pi(\varphi))$. Therefore, the adjacency matrix of $\tilde G_P(x)$ cannot have an eigenvalue-λ eigenvector with either output coefficient $\tilde a_O$ or $\tilde b_O$ nonzero. □

Proposition 4.13. *Let $\mathcal{P}$ be a finite set of span programs each with witness size greater than one, and let φ be a formula using only the associated gates. Then in the statement of Theorem 4.11, the parameters C and δ can be bounded above by a function only of $\mathcal{P}$, independent of φ, and similarly ε and $\min\{\sqrt{\varepsilon/\delta}, \min\{1, \min_v \varepsilon_{P_v}\}/(2\delta C)\}$ can be bounded below by a positive function of $\mathcal{P}$. In particular, when $\varphi(x) = 0$, $A_{\tilde G(\varphi,x)}$ has an $\Omega(1/\pi(\varphi))$ spectral gap for eigenvectors with $\tilde a_O$ or $\tilde b_O$ nonzero.*

Proof. Since $\mathcal{P}$ is finite, $\max_{v\in\varphi}\delta_{P_v} \le \max_{P\in\mathcal{P}}\delta_P < \infty$ and $\min_{v\in\varphi}\varepsilon_{P_v} \ge \min_{P\in\mathcal{P}}\varepsilon_P > 0$, bounds independent of φ. Since $C = \max\{\sigma(\varphi), \bar{\sigma}(\varphi)/\pi(\varphi)^2\}$, it remains to show that $\sigma(\varphi)$ and $\bar{\sigma}(\varphi)/\pi(\varphi)^2$ can each be bounded above by a function only of $\mathcal{P}$.

Recall from Equation (4.1) that $\sigma(\varphi) = 1 + \max_\xi \sum_{v\in\xi} 1/\pi(v)$, where the maximization is over simple paths ξ from the root to a leaf. Note that for a node $v\in\varphi$, having a child c, $\pi(v) \ge \mathrm{wsize}(P_v)\pi(c)$. Therefore, for any fixed path ξ, the sum $\sum_{w\in\xi} 1/\pi(w)$ is term-wise dominated by the geometric series $\sum_{j=0}^{\infty} 1/(\min_{P\in\mathcal{P}}\mathrm{wsize}(P))^j < \infty$.

From Equation (4.18), $\bar{\sigma}(\varphi)/\pi(\varphi)^2 = \max_\xi \sum_{w\in\xi}\pi(w)^2/\pi(\varphi)^2$. The first term in the series, for the root of φ, is $\pi(\varphi)^2/\pi(\varphi)^2 = 1$, and the ratio between the term for a node v and for its child c is bounded by $\pi(v)^2/\pi(c)^2 \le 1/\mathrm{wsize}(P_v)^2$. Therefore $\bar{\sigma}(\varphi)/\pi(\varphi)^2 < \sum_{j=0}^{\infty} 1/(\min_{P\in\mathcal{P}}\mathrm{wsize}(P))^{2j} < \infty$. □

In the following section, we will use this spectral gap, together with the completeness condition in Theorem 4.11, to develop an $O(\pi(\varphi))$-query quantum algorithm for evaluating φ.

5 Quantum formula-evaluation algorithm

Our quantum algorithm for evaluating φ will run a quantum walk starting at the output vertex of the graph $\tilde{G}(\varphi,x)$ from Theorem 4.11. In brief, Szegedy [43] has determined a correspondence between random walks and quantum walks, that can be reformulated as a correspondence between continuous-time, Hamiltonian-based quantum walks and discrete-time quantum walks. This correspondence relates the eigenvectors of the adjacency matrix $A_{\tilde{G}(\varphi,x)}$ to those of a discrete-time quantum walk, and relates the eigenvalues according to $\lambda \leftrightarrow e^{\pm i\arccos\lambda}$. If $\varphi(x) = 1$, then the initial state will have large overlap with walk eigenvectors with eigenvalues $e^{\pm i\pi/2} = \pm i$, whereas if $\varphi(x) = 0$, then the walk will have a spectral gap around phases $\pm\pi/2$. These two cases can therefore be distinguished by phase estimation [28, 32].

Our original algorithm from earlier versions of this paper [39] has since been generalized. We therefore only state the main theorem here, and refer the reader to [35] for a proof.

Theorem 5.1 ([35, Theorem 9.1])**.** *Let $G = (V,E)$ be a complex-weighted graph with Hermitian weighted adjacency matrix $A_G \in \mathcal{L}(\mathbb{C}^V)$ satisfying $\langle v|A_G|v\rangle \ge 0$ for all $v\in V$. Let V_{input} be a subset of degree-one vertices of G whose incident edges have weight one, and partition V_{input} as $V_{input} = \bigsqcup_{j\in\{1,\ldots,n\},b\in\{0,1\}} V_{j,b}$. For $x\in\{0,1\}^n$, define $G(x)$ from G by deleting all edges to vertices in $\cup_{j=1}^n V_{j,x_j}$. Let $A_{G(x)} \in \mathcal{L}(\mathbb{C}^V)$ be the weighted adjacency of matrix of $G(x)$. For any $\Lambda \ge 0$, let $\Pi_\Lambda(x)$ be the orthogonal projection onto the span of those eigenvectors of $A_{G(x)}$ with eigenvalues at most Λ in magnitude.*

Let $f : \mathcal{D} \to \{0,1\}$, with $\mathcal{D} \subseteq \{0,1\}^n$. Let $\mu \in V \smallsetminus V_{input}$, $\varepsilon = \Omega(1)$ and $\Lambda > 0$. Assume that for all inputs x with $f(x) = 1$, $\|\Pi_0(x)|\mu\rangle\|^2 \ge \varepsilon$, and that for all x with $f(x) = 0$, $\|\Pi_\Lambda(x)|\mu\rangle\|^2 \le \varepsilon/2$. Then f can be evaluated by a quantum algorithm with error probability at most $1/3$ using at most $Q = O(\|\mathrm{abs}(A_G)\|/\Lambda)$ input queries, where $\mathrm{abs}(A_G)$ is the entry-wise absolute value of A_G. Moreover, if the maximum degree of a vertex in G_P is d, then the time complexity of the algorithm for evaluating f_P is at most a factor of $(\log d)\big(\log(Q\log d)\big)^{O(1)}$ worse, after classical preprocessing and assuming constant-time coherent access to the preprocessed string.

Note that Theorem 4.11 shows a spectral gap in the case $\varphi(x) = 0$, whereas the soundness condition in Theorem 5.1 requires only the weaker assumption that the initial state have small overlap on the span of the small-eigenvalue eigenvectors, or an "effective spectral gap."

Our formula-evaluation algorithm follows from Theorems 4.11 and 5.1:

Theorem 5.2. *Let $\mathcal{S}$ be the set of all three-bit Boolean functions. There exists a quantum algorithm that evaluates an adversary-balanced formula $\varphi(x)$ over $\mathcal{S}$ using $O(\mathrm{Adv}^{\pm}(\varphi))$ input queries. After efficient classical preprocessing independent of the input x, and assuming $O(1)$-time coherent access to the preprocessed classical string, the running time of the algorithm is $\mathrm{Adv}^{\pm}(\varphi)(\log \mathrm{Adv}^{\pm}(\varphi))^{O(1)}$.*

Proof. By Theorem 4.11 and Proposition 4.13, the algorithm of Theorem 5.1 has query complexity $O(\pi(\varphi))$. By Equation (4.1), $\pi(\varphi)$ equals the maximum product of the span program witness sizes along a simple path from the root to a leaf. By Theorem 3.10, for optimal span programs this equals the product of the general adversary bounds of the gates along the path. By Theorem 2.4, the general adversary bound $\mathrm{Adv}^{\pm}(\varphi)$ is at least the same product, so $\pi(\varphi) = O(\mathrm{Adv}^{\pm}(\varphi))$.

During preprocessing, look up the optimal span programs to determine the input-independent weighted edges of $G(\varphi, x)$. Then factor the graph as required for implementing a discrete-time quantum walk. In general, for each vertex, store the unitary for one step of the quantum walk from that vertex. However, structured formulas may allow closed-form solutions, e. g., no preprocessing is needed for the balanced MAJ_3 formula of Theorem 1.1. The maximum degree d is a constant, since the span programs all have constant size. □

6 Extensions and open problems

Our formula-evaluation algorithm implies that the general adversary bound lower bounds the least span program witness size:

Corollary 6.1. *The general adversary bound lower bounds the least span program witness size for any total Boolean function f:*

$$\inf_{P: f_P = f} \mathrm{wsize}(P) \geq \mathrm{Adv}^{\pm}(f) . \tag{6.1}$$

Proof. Fix f and let f^d be the depth d balanced formula in which every gate is f. Let P be any span program with $f_P = f$. Then by Theorems 2.2 and 2.4, the quantum query complexity of evaluating f^d satisfies $Q(f^d) \geq \mathrm{Adv}^{\pm}(f^d) \geq \mathrm{Adv}^{\pm}(f)^d$, whereas by Theorems 4.11 and 5.1, $Q(f^d) = O(\mathrm{wsize}(P)^d)$, where the hidden constant may depend on f but not on d. Equation (6.1) follows by letting d tend to infinity. □

Subsequent work has shown that this inequality is tight; the least span program witness size equals the general adversary bound for any Boolean functions, total or partial [35]. Furthermore, by giving an algorithm for evaluating arbitrary span programs, and not only the repeatedly composed span programs that arise in formula evaluation, it has been shown that the general adversary bound equals the bounded-error quantum query complexity, up to constant factors [35, 37, 30].

Although the relationship between span programs, the general adversary bound and quantum query complexity has been resolved, there remain numerous open problems, including most of the problems from our earlier work [4]. Specifically, for evaluating formulas, the optimal quantum query algorithm may or may not also have a time-efficient implementation. Time-efficient and query-optimal or nearly query-optimal algorithms for formula evaluation have been studied further in [38, 36]. In [38], it is shown that our exact balance condition can be relaxed to allow a constant-factor imbalance in the adversary bounds of the inputs to any gate, and the optimal quantum query algorithm can still be implemented time efficiently. In [36], it is shown using a new span program composition technique that for arbitrary AND-OR formulas, without any balance condition, there exists a time-efficient evaluation algorithm that uses only $O(\log n)$ times as many queries as the query-optimal algorithm. Is this tradeoff between query-optimality and time-efficiency a general phenomenon? Can classical formula rebalancing results [13, 14] allow further relaxed balanced conditions? Finally, when can the preprocessing step of our algorithm be eliminated?

Span programs are also a promising tool for designing new quantum algorithms for problems beyond evaluating formulas.

Acknowledgements

We thank Troy Lee for pointing out that our Definition 3.1 corresponds to span programs. We thank Andrew Childs, Sean Hallgren, Cris Moore, David Yonge-Mallo and Shengyu Zhang for helpful conversations.

B.R. received support from NSF Grants CCF-0524828 and PHY-0456720, and from ARO Grant W911NF-05-1-0294. R.Š. received support from NSF Grant CCF-0524837 and ARO Grant DAAD 19-03-1-0082.

References

[1] ERIC ALLENDER, ROBERT BEALS, AND MITSUNORI OGIHARA: The complexity of matrix rank and feasible systems of linear equations. *Comput. Complexity*, 8(2):99–126, 1999. Preliminary version in STOC'96. [doi:10.1007/s000370050023] 292

[2] KAZUYUKI AMANO: Bounding the randomized decision tree complexity of read-once boolean functions. In *Proc. 22nd Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'11)*, pp. 1729–1744. ACM Press, 2011. [ACM:2133169] 292

[3] ANDRIS AMBAINIS: Polynomial degree vs. quantum query complexity. *J. Comput. System Sci.*, 72(2):220–238, 2006. Preliminary version in FOCS'03. [doi:10.1016/j.jcss.2005.06.006, arXiv:quant-ph/0305028] 294

[4] ANDRIS AMBAINIS, ANDREW M. CHILDS, BEN W. REICHARDT, ROBERT ŠPALEK, AND SHENGYU ZHANG: Any AND-OR formula of size N can be evaluated in time $N^{1/2+o(1)}$ on a quantum computer. *SIAM J. Comput.*, 39(6):2513–2530, 2010. Preliminary version in FOCS'07. [doi:10.1137/080712167] 291, 292, 293, 314

[5] LÁSZLÓ BABAI, ANNA GÁL, AND AVI WIGDERSON: Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19(3):301–319, 1999. Preliminary version in STOC'96. [doi:10.1007/s004930050058] 292

[6] HOWARD BARNUM AND MICHAEL SAKS: A lower bound on the quantum query complexity of read-once functions. *J. Comput. System Sci.*, 69(2):244–258, 2004. [doi:10.1016/j.jcss.2004.02.002, arXiv:quant-ph/0201007] 293, 299

[7] HOWARD BARNUM, MICHAEL SAKS, AND MARIO SZEGEDY: Quantum query complexity and semi-definite programming. In *Proc. 18th IEEE Conf. on Computational Complexity (CCC'03)*, pp. 179–193. IEEE Comp. Soc. Press, 2003. [doi:10.1109/CCC.2003.1214419] 294

[8] AMOS BEIMEL, ANNA GÁL, AND MIKE PATERSON: Lower bounds for monotone span programs. *Comput. Complexity*, 6(1):29–45, 1996. Preliminary version in FOCS'95. [doi:10.1007/BF01202040] 292

[9] ALEKSANDRS BELOVS: Span-program-based quantum algorithm for the rank problem. Technical report, 2011. [arXiv:1103.0842] 293

[10] ALEKSANDRS BELOVS: Span programs for functions with constant-sized 1-certificates: extended abstract. In *Proc. 44th STOC*. ACM Press, 2012. [doi:10.1145/2213977.2213985, arXiv:quant-ph/0611054] 293

[11] ALEKSANDRS BELOVS AND TROY LEE: Quantum algorithm for k-distinctness with prior knowledge on the input. Technical report, 2011. [arXiv:1108.3022] 293

[12] ALEKSANDRS BELOVS AND BEN W. REICHARDT: Span programs and quantum algorithms for st-connectivity and claw detection. Technical report, 2012. [arXiv:1203.2603] 293

[13] MARIA LUISA BONET AND SAMUEL R. BUSS: Size-depth tradeoffs for Boolean formulae. *Inform. Process. Lett.*, 49(3):151–155, 1994. [doi:10.1016/0020-0190(94)90093-0] 315

[14] NADER H. BSHOUTY, RICHARD CLEVE, AND WAYNE EBERLY: Size-depth tradeoffs for algebraic formulas. *SIAM J. Comput.*, 24(4):682–705, 1995. Preliminary version in FOCS'91. [doi:10.1137/S0097539792232586] 315

[15] ANDREW M. CHILDS, RICHARD CLEVE, STEPHEN P. JORDAN, AND DAVID L. YONGE-MALLO: Discrete-query quantum algorithm for NAND trees. *Theory of Computing*, 5(1):119–123, 2009. [doi:10.4086/toc.2009.v005a005] 293

[16] RONALD CRAMER AND SERGE FEHR: Optimal black-box secret sharing over arbitrary abelian groups. In *22nd Ann. Internat. Cryptology Conf. (CRYPTO'02)*, pp. 272–287. Springer, 2002. [doi:10.1007/3-540-45708-9_18] 297

[17] EDWARD FARHI, JEFFREY GOLDSTONE, AND SAM GUTMANN: A quantum algorithm for the Hamiltonian NAND tree. *Theory of Computing*, 4(1):169–190, 2008. [doi:10.4086/toc.2008.v004a008] 293

[18] DMITRY GAVINSKY AND TSUYOSHI ITO: A quantum query algorithm for the graph collision problem. Technical report, 2012. [arXiv:1204.1527] 293

[19] GENE H. GOLUB AND CHARLES F. VAN LOAN: *Matrix Computations*. Johns Hopkins, Baltimore, 3rd edition, 1996. 307

[20] LOV K. GROVER: A fast quantum mechanical algorithm for database search. In *Proc. 28th STOC*, pp. 212–219. ACM Press, 1996. [doi:10.1145/237814.237866, arXiv:quant-ph/9605043] 293

[21] LOV K. GROVER: Tradeoffs in the quantum search algorithm. *Phys. Rev. A*, 66:052314, 2002. [doi:10.1103/PhysRevA.66.052314, arXiv:quant-ph/0201152] 293

[22] RAFI HEIMAN AND AVI WIGDERSON: Randomized vs. deterministic decision tree complexity for read-once Boolean functions. *Comput. Complexity*, 1:311–329, 1991. Preliminary version in Structure in Complexity Theory'91. [doi:10.1007/BF01212962] 292

[23] PETER HØYER, TROY LEE, AND ROBERT ŠPALEK: Source codes of semidefinite programs for $\mathrm{ADV}^{\pm}$. [Link], 2006. 299

[24] PETER HØYER, TROY LEE, AND ROBERT ŠPALEK: Negative weights make adversaries stronger. In *Proc. 39th STOC*, pp. 526–535. ACM Press, 2007. [doi:10.1145/1250790.1250867, arXiv:quant-ph/0611054] 294, 295, 299

[25] T. S. JAYRAM, RAVI KUMAR, AND D. SIVAKUMAR: Two applications of information complexity. In *Proc. 35th STOC*, pp. 673–682. ACM Press, 2003. [doi:10.1145/780542.780640] 293

[26] MAURICIO KARCHMER AND AVI WIGDERSON: On span programs. In *Proc. 8th IEEE Conf. on Structure in Complexity Theory*, pp. 102–111. IEEE Comp. Soc. Press, 1993. [doi:10.1109/SCT.1993.336536] 292, 295

[27] SHELBY KIMMEL: Quantum adversary (upper) bound. Technical report, 2011. [arXiv:1101.0797] 293

[28] A. YU. KITAEV: Quantum measurements and the Abelian stabilizer problem. Technical report, 1995. [arXiv:quant-ph/9511026] 313

[29] TROY LEE, FRÉDÉRIC MAGNIEZ, AND MIKLOS SANTHA: A learning graph based quantum query algorithm for finding constant-size subgraphs. Technical report, 2011. [arXiv:1109.5135] 293

[30] TROY LEE, RAJAT MITTAL, BEN W. REICHARDT, ROBERT ŠPALEK, AND MARIO SZEGEDY: Quantum query complexity of state conversion. In *Proc. 52nd FOCS*, pp. 344–353. IEEE Comp. Soc. Press, 2011. [doi:10.1109/FOCS.2011.75, arXiv:1011.3020] 293, 314

[31] FRÉDÉRIC MAGNIEZ, ASHWIN NAYAK, MIKLOS SANTHA, AND DAVID XIAO: Improved bounds for the randomized decision tree complexity of recursive majority. In *Proc. 38th Internat. Colloq. on Automata, Languages and Programming (ICALP'11)*, pp. 317–329. Springer, 2011. [doi:10.1007/978-3-642-22006-7_27] 293

[32] DANIEL NAGAJ, PAWEL WOCJAN, AND YONG ZHANG: Fast amplification of QMA. *Quantum Inf. Comput.*, 9(11):1053–1068, 2011. [ACM:2012106, arXiv:0904.1549] 313

[33] MICHAEL A. NIELSEN AND ISAAC L. CHUANG: *Quantum computation and quantum information*. Cambridge University Press, Cambridge, 2000. 294

[34] VENTZISLAV NIKOV, SVETLA NIKOVA, AND BART PRENEEL: On the size of monotone span programs. In *4th Internat. Conf. on Security in Communication Networks (SCN'04)*, pp. 249–262. Springer, 2004. [doi:10.1007/978-3-540-30598-9_18] 297

[35] BEN W. REICHARDT: Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function (70 pp.). Technical report, 2009. Extended abstract in FOCS'09. [arXiv:0904.2759] 292, 293, 295, 296, 297, 298, 313, 314

[36] BEN REICHARDT: Faster quantum algorithm for evaluating game trees. In *Proc. 22nd Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'11)*, pp. 546–559. ACM Press, 2011. [ACM:2133079, arXiv:0907.1623] 293, 315

[37] BEN REICHARDT: Reflections for quantum query algorithms. In *Proc. 22nd Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'11)*, pp. 560–569. ACM Press, 2011. [ACM:2133080, arXiv:1005.1601] 293, 314

[38] BEN W. REICHARDT: Span-program-based quantum algorithm for evaluating unbalanced formulas. 6th Conf. Theory of Quantum Computation, Communication and Cryptography (TQC), 2011. [arXiv:0907.1622] 293, 315

[39] BEN W. REICHARDT AND ROBERT ŠPALEK: Span-program-based quantum algorithm for evaluating formulas. In *Proc. 40th STOC*, pp. 103–112. ACM Press, 2008. [doi:10.1145/1374376.1374394, arXiv:0710.2630v3] 293, 300, 301, 313

[40] MICHAEL E. SAKS AND AVI WIGDERSON: Probabilistic Boolean decision trees and the complexity of evaluating game trees. In *Proc. 27th FOCS*, pp. 29–38. IEEE Comp. Soc. Press, 1986. [doi:10.1109/SFCS.1986.44] 292, 293

[41] MIKLOS SANTHA: On the Monte Carlo boolean decision tree complexity of read-once formulae. *Random Structures Algorithms*, 6(1):75–88, 1995. Preliminary version in Structure in Complexity Theory'91. [doi:10.1002/rsa.3240060108] 292

[42] MARC SNIR: Lower bounds on probabilistic linear decision trees. *Theoret. Comput. Sci.*, 38:69–82, 1985. [doi:10.1016/0304-3975(85)90210-5] 292

[43] MARIO SZEGEDY: Quantum speed-up of Markov chain based algorithms. In *Proc. 45th FOCS*, pp. 32–41. IEEE Comp. Soc. Press, 2004. [doi:10.1109/FOCS.2004.53, arXiv:quant-ph/0401053] 313

[44] BOHUA ZHAN, SHELBY KIMMEL, AND AVINATAN HASSIDIM: Super-polynomial quantum speed-ups for boolean evaluation trees with hidden structure. In *Innovations in Theoretical Computer Science 2012 (ITCM'12)*, pp. 249–265. ACM Press, 2012. [doi:10.1145/2090236.2090258, arXiv:1101.0796] 293

[45] YECHAO ZHU: Quantum query complexity of subgraph containment with constant-sized certificates. Technical report, 2011. [arXiv:1109.4165] 293

AUTHORS

Ben Reichardt
Department of Electrical Engineering
University of Southern California, Los Angeles, CA
ben.reichardt@usc.edu
http://www-bcf.usc.edu/~breichar

Robert Špalek
Google, Mountain View, CA
spalek@google.com
http://www.ucw.cz/~robert

ABOUT THE AUTHORS

BEN REICHARDT graduated from UC Berkeley in 2006, advised by Umesh Vazirani. He studies algorithms and fault-tolerance schemes for quantum computers, and quantum cryptography.

ROBERT ŠPALEK obtained his Ph. D. in Theoretical Computer Science from CWI, Amsterdam in 2006, under the supervision of Harry Buhrman. His thesis focused on quantum algorithms and quantum query lower bounds, and these are his main research interests till today. He spent one year as a postdoc at UC Berkeley. Since 2007, he has been working as a software engineer in the search quality team at Google. He grew up in the Czech Republic, and now lives in the San Francisco Bay Area with his wife Raina and two little sons who take most of his time. His hobbies include reading, photography, and hiking.