# Approximate Span Programs*

**Tsuyoshi Ito[1] and Stacey Jeffery[2]**

1 **NEC Labs, Princeton, NJ, USA**
2 **Institute for Quantum Information and Matter, California Institute of Technology, Pasadena, CA, USA**
sjeffery@caltech.edu

## Abstract

Span programs are a model of computation that have been used to design quantum algorithms, mainly in the query model. It is known that for any decision problem, there exists a span program that leads to an algorithm with optimal quantum query complexity, however finding such an algorithm is generally challenging. In this work, we consider new ways of designing quantum algorithms using span programs. We show how any span program that decides a problem $f$ can also be used to decide "property testing" versions of the function $f$, or more generally, approximate a quantity called the *span program witness size*, which is some property of the input related to $f$. For example, using our techniques, the span program for OR, which can be used to design an optimal algorithm for the OR function, can also be used to design optimal algorithms for: threshold functions, in which we want to decide if the Hamming weight of a string is above a threshold, or far below, given the promise that one of these is true; and approximate counting, in which we want to estimate the Hamming weight of the input up to some desired accuracy. We achieve these results by relaxing the requirement that 1-inputs hit some *target* exactly in the span program, which could potentially make design of span programs significantly easier. In addition, we give an exposition of span program structure, which increases the general understanding of this important model. One implication of this is alternative algorithms for estimating the witness size when the phase gap of a certain unitary can be lower bounded. We show how to lower bound this phase gap in certain cases.

As an application, we give the first upper bounds in the adjacency query model on the quantum time complexity of estimating the effective resistance between $s$ and $t$, $R_{s,t}(G)$. For this problem we obtain $\widetilde{O}(\frac{1}{\varepsilon^{3/2}} n\sqrt{R_{s,t}(G)})$, using $O(\log n)$ space. In addition, when $\mu$ is a lower bound on $\lambda_2(G)$, by our phase gap lower bound, we can obtain an upper bound of $\widetilde{O}\left(\frac{1}{\varepsilon} n\sqrt{R_{s,t}(G)/\mu}\right)$ for estimating effective resistance, also using $O(\log n)$ space.

**1998 ACM Subject Classification** F.2 Analysis of Algorithms and Problem Complexity

**Keywords and phrases** Quantum algorithms, span programs, quantum query complexity, effective resistance

**Digital Object Identifier** 10.4230/LIPIcs.ICALP.2016.12

## 1 Introduction

Span programs are a model of computation first used to study logspace complexity [13], and more recently, introduced to the study of quantum algorithms in [20]. They are of immense theoretical importance, having been used to show that the general adversary bound gives a

* Full version: https://arxiv.org/abs/1507.00432 [quant-ph]. This work supported by the Institute for Quantum Information and Matter, an NSF Physics Frontiers Center (NFS Grant PHY-1125565) with support of the Gordon and Betty Moore Foundation (GBMF-12500028).



43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016).
Editors: Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi;
Article No. 12; pp. 12:1–12:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

tight lower bound on the quantum query complexity of any decision problem [18, 19]. As a means of designing quantum algorithms, it is known that for any decision problem, there exists a span-program-based algorithm with asymptotically optimal quantum query complexity, but this fact alone gives no indication of how to find such an algorithm. Despite the relative difficulty in designing quantum algorithms this way, there are many applications, including formula evaluation [20, 19], a number of algorithms based on the learning graph framework [3], $st$-connectivity [5] and $k$-distinctness [2]. Although generally quantum algorithms designed via span programs can only be analyzed in terms of their query complexity, in some cases their time complexity can also be analyzed, as is the case with the quantum algorithm for $st$-connectivity. In the case of the quantum algorithm for $k$-distinctness, the ideas used in designing the span program could be turned into a quantum algorithm for 3-distinctness with time complexity matching its query complexity up to logarithmic factors [4].

In this work, we consider new ways of designing quantum algorithms via span programs. Consider Grover's quantum search algorithm, which, on input $x \in \{0,1\}^n$, decides if there is some $i \in [n]$ such that $x_i = 1$ using only $O(\sqrt{n})$ quantum operations [10]. The ideas behind this algorithm have been used in innumerable contexts, but in particular, a careful analysis of the ideas behind Grover's algorithm led to algorithms for similar problems, including a class of *threshold functions*: given $x \in \{0,1\}^n$, decide if $|x| \geq t$ or $|x| < \varepsilon t$, where $|x|$ denotes the Hamming weight; and approximate counting: given $x \in \{0,1\}^n$, output an estimate of $|x|$ to some desired accuracy. The results in this paper offer the possibility of obtaining analogous results for any span program. That is, given a span program for some problem $f$, our results show that one can obtain, not only an algorithm for $f$, but algorithms for a related class of threshold functions, as well as an algorithm for estimating a quantity called the *span program witness size*, which is analogous to $|x|$ in the above example (and is in fact exactly $1/|x|$ in the span program for the OR function — see Section 2.3).

We give several new means of constructing quantum algorithms from span programs. Roughly speaking, a span program can be turned into a quantum algorithm that decides between two types of inputs: those that "hit" a certain "target vector", and those that don't. We show how to turn a span program into an algorithm that decides between inputs that get "close to" the target vector, and those that don't. Whereas traditionally a span program has been associated with some decision problem, this allows us to now associate, with one span program, a whole class of threshold problems.

In addition, for any span program $P$, we can construct a quantum algorithm that estimates the *positive witness size*, $w_+(x)$, to accuracy $\varepsilon$ in $\frac{1}{\varepsilon^{3/2}}\sqrt{w_+(x)\widetilde{W}_-}$ queries, where $\widetilde{W}_-$ is the *approximate negative witness complexity* of $P$. This construction is useful whenever we can construct a span program for which $w_+(x)$ corresponds to some function we care to estimate, as is the case with the span program for OR, in which $w_+(x) = \frac{1}{|x|}$, or the span from for $st$-connectivity, in which $w_+(G) = \frac{1}{2}R_{s,t}(G)$, where $G$ is a graph, and $R_{s,t}(G)$ is the *effective resistance* between $s$ and $t$ in $G$. We show similar results for estimating the negative witness size as well.

Our analysis of the structure of span programs increases the theoretical understanding of this important model. One implication of this is alternative algorithms for estimating the witness size when the phase gap (or spectral gap) of a certain unitary associated with the span program can be lower bounded. This is in contrast to previous span program algorithms, including those mentioned in the previous paragraph, which have all relied on *effective spectral gap* analysis. We show how the phase gap can be lower bounded by $\frac{\sigma_{\max}(A)}{\sigma_{\min}(A(x))}$, where $A$ and $A(x)$ are linear operators associated with the span program and some input $x$, and $\sigma_{\min}$ and $\sigma_{\max}$ are the smallest and largest nonzero singular values.

In addition, our exposition highlights the relationship between span programs and estimating the size of the smallest solution to a linear system, which is a problem solved by Harrow, Hassidim and Lloyd in [11]. It is not yet clear if this relationship can lead to new algorithms, but it is an interesting direction for future work.

An immediate application of our results is a quantum algorithm for estimating the effective resistance between two vertices in a graph, $R_{s,t}(G)$. This example is immediate, because in [5], a span program for $st$-connectivity was presented, in which the positive witness size corresponds to $R_{s,t}(G)$. The results of [5], combined with our new span program algorithms, immediately yield an upper bound of $\widetilde{O}(\frac{1}{\varepsilon^{3/2}} n\sqrt{R_{s,t}(G)})$ for estimating the effective resistance to relative accuracy $\varepsilon$. This upper bound also holds for time complexity, due to the time complexity analysis of [5]. Using our new spectral analysis techniques, we are also able to get an often better upper bound of $\widetilde{O}\left(\frac{1}{\varepsilon} n\sqrt{R_{s,t}(G)/\mu}\right)$, on the time complexity of estimating effective resistance, where $\mu$ is a lower bound on $\lambda_2(G)$, the second smallest eigenvalue of the Laplacian. Both algorithms use $O(\log n)$ space. We also show that a linear dependence on $n$ is necessary, so our results cannot be significantly improved.

These are the first quantum algorithms for this problem in the adjacency query model. Previous quantum algorithms have been in the edge-list model for $d$-regular graphs [22]. These results can be naively extended to the adjacency query model by simulating an edge query with $\sqrt{n/d}$ adjacency queries, using quantum search, which gives an upper bound of $\widetilde{O}\left(\frac{d^{3/2}}{\Phi^2\varepsilon}\sqrt{n/d}\right)$ queries, where $\Phi$ is the *conductance* of the input graph. Our upper bounds improve on this in many cases, including, but not limited to, $d$-regular graphs with $d > \sqrt[4]{n}$, and furthermore, our results do not assume the input graph is regular. Classically, the effective resistance can be computed exactly by inverting the Laplacian, which costs $O(m) = O(n^2)$, where $m$ is the number of edges in the input graph.

## 1.1 Preliminaries

To begin, we fix notation. For vector spaces $V$ and $W$, we let $\mathcal{L}(V, W)$ denote the set of linear operators from $V$ to $W$. For any operator $A \in \mathcal{L}(V, W)$, we denote by $\mathrm{col}A$ the columnspace, $\mathrm{row}A$ the rowspace, and $\ker A$ the kernel of $A$. $\sigma_{\min}(A)$ and $\sigma_{\max}(A)$ denote the smallest and largest non-zero singular values, respectively. $A^+$ denotes the Moore-Penrose pseudo-inverse.

The algorithms in this paper solve either decision problems, or estimation problems. For $f : X \subseteq [q]^n \to \{0, 1\}$, we say that an algorithm *decides $f$ with bounded error* if for any $x \in X$, with probability at least $2/3$, the algorithm outputs $f(x)$ on input $x$. For $f : X \subseteq [q]^n \to \mathbb{R}_{\geq 0}$, we say that an algorithm *estimates $f$ to relative accuracy $\varepsilon$* with bounded error if for any $x \in X$, with probability at least $2/3$, on input $x$ the algorithm outputs $\tilde{f}$ such that $|f(x) - \tilde{f}| \leq \varepsilon f(x)$. In both cases, using $2/3$ is without loss of generality: any algorithm with success probability bounded above $1/2$ by a constant can be amplified to success probability arbitrarily close to 1 by taking the median of the outputs of a constant number of repetitions of the algorithm. We generally omit the description "with bounded error", as all of our algorithms have bounded error.

All algorithms presented in this paper are based on the following structure. We have some initial state $|\phi_0\rangle$, and some unitary operator $U$, and we want to estimate $\|\Pi_0|\phi_0\rangle\|$, where $\Pi_0$ is the orthogonal projector onto the 1-eigenspace of $U$. The first step in this process is a quantum algorithm that estimates, in a new register, the phase of $U$ applied to the input state.

▶ **Theorem 1** (Phase Estimation [14, 9]). *Let $U = \sum_{j=1}^{m} e^{i\theta_j}|\psi_j\rangle\langle\psi_j|$ be the spectral decomposition of a unitary, with $\theta_1, \ldots, \theta_m \in (-\pi, \pi]$. For any $\Theta \in (0, \pi)$ and $\varepsilon \in (0, 1)$, there*

*exists a quantum algorithm that makes* $O\left(\frac{1}{\Theta}\log\frac{1}{\varepsilon}\right)$ *controlled calls to* $U$ *and, on input* $|\psi_j\rangle$*, outputs a state* $|\psi_j\rangle|\omega\rangle$ *such that if* $\theta_j = 0$*, then* $|\omega\rangle = |0\rangle$*, and if* $|\theta_j| \geq \Theta$*,* $|\langle 0|\omega\rangle|^2 \leq \varepsilon$*. If* $U$ *acts on* $s$ *qubits, the algorithm uses* $O(s + \log\frac{1}{\Theta})$ *space.*

The precision needed to isolate $\Pi_0|\phi_0\rangle$ depends on the smallest nonzero phase of $U$, the *phase gap*.

▶ **Definition 2** (Phase Gap). Let $\{e^{i\theta_j}\}_{j\in S}$ be the eigenvalues of a unitary operator $U$, with $\{\theta_j\}_{j\in S} \subset (-\pi, \pi]$. Then the *phase gap* of $U$ is $\Delta(U) := \min\{|\theta_j| : \theta_j \neq 0\}$.

In order to estimate $\|\Pi_0|\phi_0\rangle\|^2$, given a state $|0\rangle\Pi_0|\phi_0\rangle + |1\rangle(I-\Pi_0)|\phi_0\rangle$, we use the following.

▶ **Theorem 3** (Amplitude Estimation [7]). *Let* $\mathcal{A}$ *be a quantum algorithm that outputs* $\sqrt{p(x)}|0\rangle|\Psi_x(0)\rangle + \sqrt{1-p(x)}|1\rangle|\Psi_x(1)\rangle$ *on input* $x$*. Then there exists a quantum algorithm that estimates* $p(x)$ *to precision* $\varepsilon$ *using* $O\left(\frac{1}{\varepsilon}\frac{1}{\sqrt{p(x)}}\right)$ *calls to* $\mathcal{A}$*.*

If we know the amplitude is either $\leq p_0$ or $\geq p_1$ for some $p_0 < p_1$, then we can use amplitude estimation to distinguish between these two cases.

▶ **Corollary 4** (Amplitude Gap). *Let* $\mathcal{A}$ *be a quantum algorithm that, on input* $x$*, outputs* $\sqrt{p(x)}|0\rangle|\Psi_x(0)\rangle + \sqrt{1-p(x)}|1\rangle|\Psi_x(1)\rangle$*. For any* $0 \leq p_0 < p_1 \leq 1$*, we can distinguish between the cases* $p(x) \geq p_1$ *and* $p(x) \leq p_0$ *with bounded error using* $O\left(\frac{\sqrt{p_1}}{p_1-p_0}\right)$ *calls to* $\mathcal{A}$*.*

In order to make use of phase estimation, we will need to analyze the spectrum of a particular unitary, which, in our case, consists of a pair of reflections. The following lemma first appeared in this form in [15]:

▶ **Lemma 5** (Effective Spectral Gap Lemma). *Let* $U = (2\Pi_A - I)(2\Pi_B - I)$ *be the product of two reflections, and let* $\Pi_\Theta$ *be the orthogonal projector onto* $\mathrm{span}\{|u\rangle : U|u\rangle = e^{i\theta}|u\rangle, |\theta| \leq \Theta\}$*. Then if* $\Pi_A|u\rangle = 0$*,* $\|\Pi_\Theta\Pi_B|u\rangle\| \leq \frac{\Theta}{2}\||u\rangle\|$*.*

The following theorem was first used in the context of quantum algorithms by Szegedy [21]:

▶ **Theorem 6** ([21]). *Let* $U = (2\Pi_A - I)(2\Pi_B - I)$ *be a unitary on a space* $H$ *containing* $A = \mathrm{span}\{|\psi_1\rangle, \ldots, |\psi_a\rangle\}$ *and* $B = \mathrm{span}\{|\phi_1\rangle, \ldots, |\phi_b\rangle\}$*. Let* $\Pi_A = \sum_{i=1}^a |\psi_i\rangle\langle\psi_i|$ *and* $\Pi_B = \sum_{i=1}^b |\phi_i\rangle\langle\phi_i|$ *be the orthogonal projectors onto these spaces. Let* $D = \Pi_A\Pi_B$ *be the discriminant of* $U$*, and suppose it has singular value decomposition* $\sum_{j=1}^r \cos\theta_j|\alpha_j\rangle\langle\beta_j|$*, with* $\theta_j \in [0, \frac{\pi}{2}]$*. Then the spectrum of* $U$ *is* $\{e^{\pm 2i\theta_j}\}_j$*. The 1-eigenspace of* $U$ *is* $(A\cap B)\oplus(A^\perp\cap B^\perp)$ *and the* $(-1)$*-eigenspace is* $(A \cap B^\perp) \oplus (A^\perp \cap B)$*.*

Let $\Lambda_A = \sum_{j=1}^a |\psi_j\rangle\langle j|$ and $\Lambda_B = \sum_{j=1}^b |\phi_j\rangle\langle j|$. We note that in the original statement of Theorem 6, the discriminant is defined $D' = \Lambda_A^\dagger\Lambda_B$. However it is easy to see that $D'$ and $D$ have the same singular values: if $D' = \sum_i \sigma_i|v_i\rangle\langle u_i|$ is a singular value decomposition of $D'$, then $D = \sum_i \sigma_i\Lambda_A|v_i\rangle\langle u_i|\Lambda_B^\dagger$ is a singular value decomposition of $D$, since $\Lambda_A$ acts as an isometry on the columns of $D'$, and $\Lambda_B$ acts as an isometry on the rows of $D'$.
The following corollary to Theorem 6 will be useful in the analysis of several algorithms.

▶ **Corollary 7** (Phase Gap and Discriminant). *Let* $D$ *be the discriminant of a unitary* $U = (2\Pi_A - I)(2\Pi_B - I)$*. Then* $\Delta(-U) \geq 2\sigma_{\min}(D)$*.*

# 2 Approximate Span Programs

## 2.1 Span Programs and Decision Problems

In this section, we review the concept of span programs, and their use in quantum algorithms.

▶ **Definition 8** (Span Program). A span program $P = (H, V, \tau, A)$ on $[q]^n$ consists of

1. finite-dimensional inner product spaces $H = H_1 \oplus \cdots \oplus H_n \oplus H_{\text{true}} \oplus H_{\text{false}}$, and $\{H_{j,a} \subseteq H_j\}_{j\in[n],a\in[q]}$ such that $H_{j,1} + \cdots + H_{j,q} = H_j$,
2. a vector space $V$,
3. a *target vector* $\tau \in V$, and
4. a linear operator $A \in \mathcal{L}(H, V)$.

To each string $x \in [q]^n$, we associate a subspace $H(x) := H_{1,x_1} \oplus \cdots \oplus H_{n,x_n} \oplus H_{\text{true}}$.

Although our notation in Definition 8 deviates from previous span program definitions, the only difference in the substance of the definition is that the spaces $H_{j,a}$ and $H_{j,b}$ for $a \neq b$ need not be orthogonal in our definition. This has the effect of removing $\log q$ factors in the equivalence between span programs and the dual adversary bound (for details see [12, Sec. 7.1]). The spaces $H_{\text{true}}$ and $H_{\text{false}}$ can be useful for designing a span program, but are never required, since we can always add an $(n+1)^{\text{th}}$ variable, set $x_{n+1} = 1$, and let $H_{n+1,0} = H_{\text{false}}$ and $H_{n+1,1} = H_{\text{true}}$.

A span program on $[q]^n$ partitions $[q]^n$ into two sets: *positive* inputs, which we call $P_1$, and *negative* inputs, which we call $P_0$. The importance of this partition stems from the fact that a span program may be converted into a quantum algorithm for deciding this partition in the quantum query model [18, 19]. Thus, if one can construct a span program whose partition of $[q]^n$ corresponds to a problem one wants to solve, an algorithm follows. In order to describe how a span program partitions $[q]^n$ and the query complexity of the resulting algorithm, we need the concept of positive and negative witnesses and witness size.

▶ **Definition 9** (Positive and Negative Witness). Fix a span program $P$ on $[q]^n$, and a string $x \in [q]^n$. We say that $|w\rangle$ is a *positive witness for $x$ in $P$* if $|w\rangle \in H(x)$, and $A|w\rangle = \tau$. We define the *positive witness size of $x$* as:

$$w_+(x, P) = w_+(x) = \min\{\||w\rangle\|^2 : |w\rangle \in H(x), A|w\rangle = \tau\},$$

if there exists a positive witness for $x$, and $w_+(x) = \infty$ else. We say that $\omega \in \mathcal{L}(V, \mathbb{R})$ is a *negative witness for $x$ in $P$* if $\omega A \Pi_{H(x)} = 0$ and $\omega\tau = 1$. We define the *negative witness size of $x$* as:

$$w_-(x, P) = w_-(x) = \min\{\|\omega A\|^2 : \omega \in \mathcal{L}(V, \mathbb{R}), \omega A \Pi_{H(x)} = 0, \omega\tau = 1\},$$

if there exists a negative witness, and $w_-(x) = \infty$ otherwise. If $w_+(x)$ is finite, we say that $x$ is *positive* (with respect to $P$), and if $w_-(x)$ is finite, we say that $x$ is *negative*. We let $P_1$ denote the set of positive inputs, and $P_0$ the set of negative inputs for $P$. Note that for every $x \in [q]^n$, exactly one of $w_-(x)$ and $w_+(x)$ is finite; that is, $(P_0, P_1)$ partitions $[q]^n$.

For a decision problem $f : X \subseteq [q]^n \to \{0, 1\}$, we say that $P$ *decides* $f$ if $f^{-1}(0) \subseteq P_0$ and $f^{-1}(1) \subseteq P_1$. In that case, we can use $P$ to construct a quantum algorithm that decides $f$.

▶ **Theorem 10** ([18]). *Fix $f : X \subseteq [q]^n \to \{0, 1\}$, and let $P$ be a span program on $[q]^n$ that decides $f$. Let $W_+(f, P) = \max_{x\in f^{-1}(1)} w_+(x, P)$ and $W_-(f, P) = \max_{x\in f^{-1}(0)} w_-(x, P)$. Then there exists a quantum algorithm that decides $f$ using $O(\sqrt{W_+(f, P)W_-(f, P)})$ queries.*

We call $\sqrt{W_+(f,P)W_-(f,P)}$ the *complexity* of $P$. It is known that for any decision problem, there exists a span program whose complexity is equal, up to constants, to its query complexity [18, 19] ([12, Sec. 7.1] removes log factors in this statement), however, it is generally a difficult task to find such an optimal span program.

## 2.2 Span Programs and Approximate Decision Problems

Consider a span program $P$ and $x \in P_0$. Suppose there is some $|w\rangle \in H(x)$ such that $A|w\rangle$ comes extremely close to $\tau$. We might say that $x$ is very close to being in $P_1$. If all vectors in $H(y)$ for $y \in P_0 \setminus \{x\}$ are very far from $\tau$, it might be slightly more natural to consider the partition $(P_0 \setminus \{x\}, P_1 \cup \{x\})$ rather than $(P_0, P_1)$.

As further motivation, we mention a construction of Reichardt [18, Sec. 3 of full version] that takes any quantum query algorithm with one-sided error, and converts it into a span program whose complexity matches the query complexity of the algorithm. The target of the span program is the vector $|1,\bar{0}\rangle$, which corresponds to a quantum state with a 1 in the answer register and 0s elsewhere. If an algorithm has no error on 1-inputs, it can be modified so that it always ends in exactly this state, by uncomputing all but the answer register. An algorithm with two-sided error cannot be turned into a span program using this construction, because there is error in the final state. This is intuitively in opposition to the evidence that span programs characterize bounded (two-sided) error quantum query complexity. The exactness required by span programs seems to contrast the spirit of non-exact quantum algorithms.

This motivates us to consider the *positive error* of an input, or how close it comes to being positive. Since there is no meaningful notion of distance in $V$, we consider closeness in $H$.

▶ **Definition 11** (Positive Error)**.** For any span program $P$ on $[q]^n$, and $x \in [q]^n$, we define the *positive error of $x$ in $P$* as:

$$e_+(x) = e_+(x,P) := \min\left\{\left\|\Pi_{H(x)^\perp}|w\rangle\right\|^2 : A|w\rangle = \tau\right\}.$$

Note that $e_+(x,P) = 0$ if and only if $x \in P_1$. Any $|w\rangle$ such that $\left\|\Pi_{H(x)^\perp}|w\rangle\right\|^2 = e_+(x)$ is called a *min-error positive witness for $x$ in $P$*. We define

$$\tilde{w}_+(x) = \tilde{w}_+(x,P) := \min\left\{\||w\rangle\|^2 : A|w\rangle = \tau, \left\|\Pi_{H(x)^\perp}|w\rangle\right\|^2 = e_+(x)\right\}.$$

A min-error positive witness that also minimizes $\||w\rangle\|^2$ is called an *optimal min-error positive witness for $x$*.

Note that if $x \in P_1$, then $e_+(x) = 0$. In that case, a min-error positive witness for $x$ is just a positive witness, and $\tilde{w}_+(x) = w_+(x)$.

We can define a similar notion for positive inputs, to measure their closeness to being negative.

▶ **Definition 12** (Negative Error)**.** For any span program $P$ on $[q]^n$ and $x \in [q]^n$, we define the *negative error of $x$ in $P$* as:

$$e_-(x) = e_-(x,P) := \min\left\{\left\|\omega A\Pi_{H(x)}\right\|^2 : \omega(\tau) = 1\right\}.$$

Again, $e_-(x,P) = 0$ if and only if $x \in P_0$. Any $\omega$ such that $\left\|\omega A\Pi_{H(x)}\right\|^2 = e_-(x,P)$ is called a *min-error negative witness for $x$ in $P$*. We define

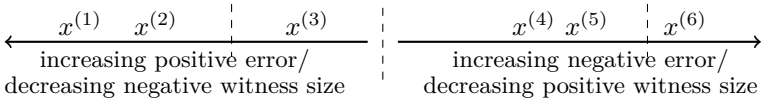
$$\tilde{w}_-(x) = \tilde{w}_-(x,P) := \min\left\{\|\omega A\|^2 : \omega(\tau) = 1, \left\|\omega A\Pi_{H(x)}\right\|^2 = e_-(x,P)\right\}.$$

A min-error negative witness that also minimizes $\|\omega A\|^2$ is called an *optimal min-error negative witness for* $x$.

It turns out that the notion of span program error has a very nice characterization as exactly the reciprocal of the witness size. We prove in the full version (Theorems 2.10 and 2.11):

$$\forall x \in P_0,\ w_-(x) = \frac{1}{e_+(x)}, \qquad \text{and} \qquad \forall x \in P_1,\ w_+(x) = \frac{1}{e_-(x)}.$$

This is a very nice state of affairs, for a number of reasons. It allows us two ways of thinking about approximate span programs: in terms of how small the error is, or how large the witness size is. That is, we can say that an input $x \in P_0$ is *almost positive* either because its positive error is small, or equivalently, because its negative witness size is large. In general, we can think of $P$ as not only partitioning $P$ into $(P_0, P_1)$, but inducing an ordering on $[q]^n$ from most negative — smallest negative witness, or equivalently, largest positive error — to most positive — smallest positive witness, or equivalently, largest negative error. For example, on the domain $\{x^{(1)}, \ldots, x^{(6)}\} \subset [q]^n$, $P$ might induce the following ordering:




The inputs $\{x^{(1)}, x^{(2)}, x^{(3)}\}$ are in $P_0$, and $w_-(x^{(1)}) < w_-(x^{(2)}) < w_-(x^{(3)})$ (although it is generally possible for two inputs to have the same witness size). The inputs $\{x^{(4)}, x^{(5)}, x^{(6)}\}$ are in $P_1$, and $w_+(x^{(4)}) > w_+(x^{(5)}) > w_+(x^{(6)})$. The span program exactly decides the partition $(\{x^{(1)}, x^{(2)}, x^{(3)}\}, \{x^{(4)}, x^{(5)}, x^{(6)}\})$, but we say it *approximates* any partition that respects the ordering. If we obtain a partition by drawing a line somewhere on the left side, for example $(\{x^{(1)}, x^{(2)}\}, \{x^{(3)}, x^{(4)}, x^{(5)}, x^{(6)}\})$, we say $P$ *negatively* approximates the function corresponding to that partition, whereas if we obtain a partition by drawing a line on the right side, for example $(\{x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}, x^{(5)}\}, \{x^{(6)}\})$, we say $P$ *positively* approximates the function.

▶ **Definition 13** (Functions Approximately Associated with $P$)**.** Let $P$ be a span program on $[q]^n$, and $f : X \subseteq [q]^n \to \{0,1\}$ a decision problem. For any $\lambda \in (0,1)$, we say that $P$ *positively* $\lambda$*-approximates* $f$ if $f^{-1}(1) \subseteq P_1$, and for all $x \in f^{-1}(0)$, either $x \in P_0$, or $w_+(x,P) \geq \frac{1}{\lambda} W_+(f,P)$, where $W_+(f,P) := \max_{x \in f^{-1}(1)} w_+(x,P)$. We say that $P$ *negatively* $\lambda$*-approximates* $f$ if $f^{-1}(0) \subseteq P_0$, and for all $x \in f^{-1}(1)$, either $x \in P_1$, or $w_-(x,P) \geq \frac{1}{\lambda} W_-(f,P)$, where $W_-(f,P) := \max_{x \in f^{-1}(0)} w_-(x,P)$. If $P$ decides $f$ exactly, then both conditions hold for any value of $\lambda$, and so we can say that $P$ 0-approximates $f$.

This allows us to consider a much broader class of functions associated with a particular span program. This association is useful, because as with the standard notion of association between a function $f$ and a span program, if a function is approximated by a span program, we can convert the span program into a quantum algorithm that decides $f$ using a number of queries related to the witness sizes. Specifically, we get the following theorem.

▶ **Theorem 14** (Approximate Span Program Decision Algorithms)**.** *Fix* $f : X \subseteq [q]^n \to \{0,1\}$*, and let* $P$ *be a span program that positively* $\lambda$*-approximates* $f$*. Define*

$$W_+ = W_+(f,P) := \max_{x \in f^{-1}(1)} w_+(x,P) \qquad \text{and} \qquad \widetilde{W}_- = \widetilde{W}_-(f,P) := \max_{x \in f^{-1}(0)} \tilde{w}_-(x,P).$$

*There is a quantum algorithm that decides $f$ with bounded error in $O\left(\frac{\sqrt{W_+\widetilde{W}_-}}{(1-\lambda)^{3/2}}\log\frac{1}{1-\lambda}\right)$ queries. Similarly, let $P$ be a span program that negatively $\lambda$-approximates $f$. Define*

$$W_- = W_-(f,P) := \max_{x\in f^{-1}(0)} w_-(x,P) \qquad \text{and} \qquad \widetilde{W}_+ = \widetilde{W}_+(f,P) := \max_{x\in f^{-1}(1)} \tilde{w}_+(x,P).$$

*There is a quantum algorithm that decides $f$ with bounded error in $O\left(\frac{\sqrt{W_-\widetilde{W}_+}}{(1-\lambda)^{3/2}}\log\frac{1}{1-\lambda}\right)$ queries.*

With the ability to distinguish between different witness sizes, we can obtain algorithms for estimating the witness size.

▶ **Theorem 15** (Witness Size Estimation Algorithm). *Fix $f : X \subseteq [q]^n \to \mathbb{R}_{\geq 0}$. Let $P$ be a span program such that for all $x \in X$, $f(x) = w_+(x,P)$ and define $\widetilde{W}_- = \widetilde{W}_-(f,P) = \max_{x\in X} \tilde{w}_-(x,P)$. There exists a quantum algorithm that estimates $f$ to accuracy $\varepsilon$ in $\widetilde{O}\left(\frac{1}{\varepsilon^{3/2}}\sqrt{w_+(x)\widetilde{W}_-}\right)$ queries. Similarly, let $P$ be a span program such that for all $x \in X$, $f(x) = w_-(x,P)$ and define $\widetilde{W}_+ = \widetilde{W}_+(f,P) = \max_{x\in X} \tilde{w}_+(x,P)$. Then there exists a quantum algorithm that estimates $f$ to accuracy $\varepsilon$ in $\widetilde{O}\left(\frac{1}{\varepsilon^{3/2}}\sqrt{w_-(x)\widetilde{W}_+}\right)$ queries.*

Proofs of Theorem 14 and 15 can be found in the full version (Theorems 2.7 and 2.8), but we give a high-level outline here. As in the case of algorithms previously constructed from span programs, our algorithms will consist of phase estimation of a unitary on $H$, applied to some initial state. Unlike previous applications, we will use $|w_0\rangle = A^+\tau$ (discussed more in full version, Section 2.4), as the initial state. This state is independent of the input, and so can be generated with 0 queries. For *negative span program algorithms*, where we want to decide a function negatively approximated by $P$, we will use a unitary $U(P,x)$, defined as follows:

$$U(P,x) := (2\Pi_{\ker A} - I)(2\Pi_{H(x)} - I).$$

This is similar to the unitary used in previous span program algorithms. Note that $(2\Pi_{\ker A} - I)$ is input-independent, and so can be implemented in 0 queries. However, in order to analyze the time complexity of a span program algorithm, this reflection must be implemented (as we are able to do for our applications, following [5]). The reflection $(2\Pi_{H(x)} - I)$ depends on the input, but requires only two queries to implement.

For *positive span program algorithms*, where we want to decide a function positively approximated by $P$, or estimate the positive witness size, we will use a slightly different unitary, $U'(P,x)$.

In order to show how these unitaries can be used to distinguish between inputs with small negative (resp. positive) witnesses, and those that only have large negative (resp. positive) witnesses, we analyze the overlap of the initial state, $|w_0\rangle$ with the 1-eigenspace of $U(P,x)$ (resp. $U'(P,x)$) in terms of the witness size. Specifically, we show that the overlap of $|w_0\rangle$ with the 1-eigenspace of $U(P,x)$ is exactly $\frac{1}{w_-(x,P)}$ (full version, Lemma 3.3), and the overlap of $|w_0\rangle$ with the 1-eigenspace of $U'(P,x)$ is exactly $\frac{1}{w_+(x,P)}$ (full version, Lemma 3.5). We can then use phase estimation, followed by amplitude estimation, to estimate the witness size.

There are then two possibilities for bounding the required precision of phase estimation, which also tells us the number of times we will need to call $U(P,x)$ (resp. $U'(P,x)$), and

therefore, the query complexity of the algorithm. Similar to previous span program algorithms we use the effective spectral gap lemma to show that the overlap of $|w_0\rangle$ with $e^{i\theta}$-eigenspaces of $U(P,x)$ (resp. $U'(P,x)$) is not too large for small $\theta$ (full version, Lemmas 3.2 and 3.4). This leads to Theorem 14 and Theorem 15.

The second way to bound the required precision of phase estimation is to lower bound the phase gap of $U(P,x)$ (resp. $U'(P,x)$), which may be very difficult in general. However, by relating the phase gap of $U(P,x)$ (resp. $U'(P,x)$) to the spectrum of $A$ and $A(x) = A\Pi_{H(x)}$ in a novel way, we show how to lower bound the phase gap in some cases, which may give better results. This leads to the following theorem.

▶ **Theorem 16** (Witness Size Estimation Algorithm Using Real Phase Gap). *We say that a span program is* normalized *if* $\|A^+\tau\| = 1$. *Any span program can be normalized by scaling* $\tau$.

*Fix* $f : X \subseteq [q]^n \to \mathbb{R}_{\geq 0}$ *and let* $P = (H, V, \tau, A)$ *be a normalized span program on* $[q]^n$ *such that for all* $x \in X$, $f(x) = w_+(x,P)$ *(resp.* $f(x) = w_-(x)$*). If* $\kappa \geq \frac{\sigma_{\max}(A)}{\sigma_{\min}(A\Pi_{H(x)})}$ *for all* $x \in X$, *then the quantum query complexity of estimating* $f(x)$ *to relative accuracy* $\varepsilon$ *is at most* $\widetilde{O}\left(\sqrt{f(x)}\kappa/\varepsilon\right)$.

In particular, in our application to effective resistance, it is not difficult to bound the phase gap in this way, which leads to an improved upper bound.

## 2.3 Example

To illustrate how these ideas might be useful, we will give a brief example of how a span program that leads to an algorithm for the OR function can be combined with our results to additionally give algorithms for threshold functions and approximate counting. We define a span program $P$ on $\{0,1\}^n$ as follows:

$$V = \mathbb{R}, \qquad \tau = 1, \qquad H_i = H_{i,1} = \text{span}\{|i\rangle\}, \qquad H_{i,0} = \{0\}, \qquad A = \sum_{i=1}^{n} \langle i|.$$

So we have $H = \text{span}\{|i\rangle : i \in [n]\}$ and $H(x) = \text{span}\{|i\rangle : x_i = 1\}$. It's not difficult to see that $P$ decides OR. In particular, we can see that the optimal positive witness for any $x$ such that $|x| > 0$ is $|w_x\rangle = \sum_{i:x_i=1} \frac{1}{|x|}|i\rangle$. The only linear function $\omega : \mathbb{R} \to \mathbb{R}$ that maps $\tau$ to 1 is the identity, and indeed, this is a negative witness for the string $\bar{0} = 0 \ldots 0$, since $H(\bar{0}) = \{0\}$, and so $\omega A\Pi_{H(\bar{0})} = 0$.

Let $\lambda \in (0,1)$, $t \in [n]$, and let $f$ be a threshold function defined by $f(x) = 1$ if $|x| \geq t$ and $f(x) = 0$ if $|x| \leq \lambda t$, with the promise that one of these conditions holds. Note that if $f(x) = 1$, then $w_+(x) = \||w_x\rangle\|^2 = \frac{1}{|x|} \leq \frac{1}{t}$, so $W_+(f,P) = \frac{1}{t}$. On the other hand, if $f(x) = 0$, then $w_+(x) = \frac{1}{|x|} \geq \frac{1}{\lambda t} = \frac{1}{\lambda} W_+(f,P)$, so $P$ positively $\lambda$-approximates $f$. The only approximate negative witness is $\omega$ the identity, so we have $\widetilde{W}_- = \|\omega A\|^2 = \|A\|^2 = n$. By Theorem 14, there is a quantum algorithm for $f$ with query complexity $\frac{1}{(1-\lambda)^{3/2}}\sqrt{W_+\widetilde{W}_-} = \frac{1}{(1-\lambda)^{3/2}}\sqrt{n/t}$.

Furthermore, since $w_+(x) = \frac{1}{|x|}$, by Theorem 15, we can estimate $\frac{1}{|x|}$ to relative accuracy $\varepsilon$, and therefore we can estimate $|x|$ to relative accuracy $2\varepsilon$, in quantum query complexity $\frac{1}{\varepsilon^{3/2}}\sqrt{n/|x|}$.

These upper bounds do not have optimal scaling in $\varepsilon$, as the actual quantum query complexities of these problems are $\frac{1}{1-\lambda}\sqrt{n/t}$ and $\frac{1}{\varepsilon}\sqrt{n/|x|}$ [6, 7, 1], however, using Theorem 16, the optimal query complexities can be recovered.

## 3 Applications

In this section, we will demonstrate how to apply Theorem 15 and 16 to get new quantum algorithms. Specifically, we will give upper bounds of $\widetilde{O}(n\sqrt{R_{s,t}}/\varepsilon^{3/2})$ and $\widetilde{O}(n\sqrt{R_{s,t}/\lambda_2}/\varepsilon)$ on the time complexity of estimating the effective resistance, $R_{s,t}$, between two vertices, $s$ and $t$, in a graph. Unlike previous upper bounds, we study this problem in the adjacency model.

A *unit flow* from $s$ to $t$ in $G$ is a real-valued function $\theta$ on the directed edges $\vec{E}(G) = \{(u,v) : \{u,v\} \in E(G)\}$ such that:

1. for all $(u,v) \in \vec{E}$, $\theta(u,v) = -\theta(v,u)$;
2. for all $u \in [n] \setminus \{s,t\}$, $\sum_{v\in\Gamma(u)} \theta(u,v) = 0$, where $\Gamma(u) = \{v \in [n] : \{u,v\} \in E\}$; and
3. $\sum_{u\in\Gamma(s)} \theta(s,u) = \sum_{u\in\Gamma(t)} \theta(u,t) = 1$.

Let $\mathcal{F}$ be the set of unit flows from $s$ to $t$ in $G$. The *effective resistance* from $s$ to $t$ in $G$ is defined:

$$R_{s,t}(G) = \min_{\theta\in\mathcal{F}} \sum_{\{u,v\}\in E(G)} \theta(u,v)^2.$$

This quantity gives the resistance of a network of unit resistors described by $G$, but is also an interesting quantity for graph theoretic reasons. For instance, the commute time between $s$ and $t$, which is the expected number of steps in a random walk starting from $s$ to reach $t$, and then return to $s$, is exactly the product of the number of edges in $G$, and $R_{s,t}(G)$ [8].

In the adjacency model, the input is a string $x \in \{0,1\}^{n\times n}$, representing a graph $G_x = ([n], \{\{i,j\} : x_{i,j} = 1\})$ (we assume that $x_{i,i} = 0$ for all $i$, and $x_{i,j} = x_{j,i}$ for all $i,j$). The problem of $st$-connectivity is the following. Given $x \in \{0,1\}^{n\times n}$ and $s,t \in [n]$, decide if there exists a path from $s$ to $t$ in $G_x$. A span-program-based algorithm for this problem was given in [5], with time complexity $\widetilde{O}(n\sqrt{p})$, under the promise that, if $s$ and $t$ are connected in $G_x$, they are connected by a path of length $\leq p$. They use the following span program, defined on $\{0,1\}^{n\times n}$:

$$H_{(u,v),0} = \{0\},\ \ H_{(u,v),1} = \mathrm{span}\{|u,v\rangle\},\ \ V = \mathbb{R}^n,\ \ A = \sum_{u,v\in[n]} (|u\rangle - |v\rangle)\langle u,v|,\ \ |\tau\rangle = |s\rangle - |t\rangle.$$

We have $H = \mathrm{span}\{|u,v\rangle : u,v \in [n]\}$, and $H(x) = \mathrm{span}\{|u,v\rangle : \{u,v\} \in E(G_x)\}$. Throughout this section, $P$ will denote the above span program. We will use this span program to define algorithms for estimating the effective resistance. Ref. [5] are even able to show how to efficiently implement a unitary similar to $U(P,x)$, giving a time efficient algorithm. In the full version, we adapt their proof to our setting, showing that our algorithms are time efficient as well.

The effective resistance between $s$ and $t$ is related to $st$-connectivity by the fact that if $s$ and $t$ are not connected, then $R_{s,t}$ is undefined (there is no flow from $s$ to $t$) and if $s$ and $t$ are connected then $R_{s,t}$ is related to the number and length of paths from $s$ to $t$. In particular, if $s$ and $t$ are connected by a path of length $p$, then $R_{s,t}(G) \leq p$ (take the unit flow that simply travels along this path). In general, if $s$ and $t$ are connected in $G$, then $\frac{2}{n} \leq R_{s,t}(G) \leq n-1$. The span program for $st$-connectivity is amenable to the task of estimating the effective resistance due to the following.

▶ **Lemma 17** ([5]). *For any graph $G_x$ on $[n]$, $x \in P_1$ if and only if $s$ and $t$ are connected, and in that case, $w_+(x,P) = \frac{1}{2}R_{s,t}(G_x)$.*

A near immediate consequence of this, combined with Theorem 15, is the following.

▶ **Theorem 18.** *There exists a quantum algorithm for estimating $R_{s,t}(G_x)$ to accuracy $\varepsilon$ with time complexity $\widetilde{O}\left(\frac{n\sqrt{R_{s,t}(G_x)}}{\varepsilon^{3/2}}\right)$ and space complexity $O(\log n)$.*

By analyzing the spectra of $A$ and $A(x)$, and applying Theorem 16, we can get an often better algorithm (Theorem 19). The *spectral gap* of a graph $G$, denoted $\lambda_2(G)$, is the second largest eigenvalue (including multiplicity) of the Laplacian of $G$, which is defined $L_G = \sum_{u\in[n]} d_u|u\rangle\langle u| - \sum_{u\in[n]}\sum_{v\in\Gamma(u)}|u\rangle\langle v|$, where $d_u$ is the degree of $u$, and $\Gamma(u)$ is the set of neighbours of $u$. The smallest eigenvalue of $L_G$ is 0 for any graph $G$. A graph $G$ is connected if and only if $\lambda_2(G) > 0$. A connected graph $G$ has $\frac{2}{n^2} \leq \lambda_2(G) \leq n$.

The following theorem is an improvement over Theorem 18 when $\lambda_2(G) > \varepsilon$. In particular, it is an improvement for all $\varepsilon$ when we know that $\lambda_2(G) > 1$.

▶ **Theorem 19.** *Let $\mathcal{G}$ be a family of graphs such that for all $x \in \mathcal{G}$, $\lambda_2(G_x) \geq \mu$. Let $f : \mathcal{G}\times[n]\times[n] \to \mathbb{R}_{>0}$ be defined by $f(x,s,t) = R_{s,t}(G_x)$. There exists a quantum algorithm for estimating $f$ to relative accuracy $\varepsilon$ that has time complexity $\widetilde{O}\left(\frac{1}{\varepsilon}n\sqrt{R_{s,t}(G_x)/\mu}\right)$ and space complexity $O(\log n)$.*

**Proof.** We will apply Theorem 16. We first compute $\||w_0\rangle\|^2$, in order to normalize $P$.

▶ **Lemma 20.** $\||w_0\rangle\|^2 = \frac{1}{n}$.

**Proof.** Recall that $|w_0\rangle = A^+\tau$. This is the smallest $|w_0\rangle$ such that $A|w_0\rangle = \tau$. Since $H(x) = H$ when $G_x$ is the complete graph, by Lemma 17, we need only compute $R_{s,t}$ in the complete graph. It's simple to verify that the optimal unit $st$-flow in the complete graph has $\frac{1}{n}$ units of flow on every path of the form $(s,u,t)$ for $u \in [n]\setminus\{s,t\}$, and $\frac{2}{n}$ units of flow on the edge $(s,t)$. Thus, $R_{s,t}(K_n) = \sum_{u\in[n]\setminus\{s,t\}} 2(1/n)^2 + (2/n)^2 = 2/n$. Thus $\||w_0\rangle\|^2 = \frac{1}{2}R_{s,t}(K_n) = \frac{1}{n}$. ◀

Next, we compute the following:

▶ **Lemma 21.** *For any $x \in \mathcal{G}$, $\frac{\sigma_{\max}(A)}{\sigma_{\min}(A(x))} = \sqrt{\frac{n}{\lambda_2(G_x)}} \leq \sqrt{\frac{n}{\mu}}$, so $\kappa(f) \leq \sqrt{\frac{n}{\mu}}$.*

**Proof.** Let $L_x$ denote the Laplacian of $G_x$. We have:

$$A(x)A(x)^T = \sum_{u\in[n]}\sum_{v\in\Gamma(u)}(|u\rangle - |v\rangle)(\langle u| - \langle v|) = 2\sum_{u\in[n]} d_u|u\rangle\langle u| - 2\sum_{u\in[n]}\sum_{v\in\Gamma(u)}|u\rangle\langle v| = 2L_x.$$

Thus, if $L$ denotes the Laplacian of the complete graph, we also have $AA^T = 2L$. Letting $J$ denote the all ones matrix, we have $L = (n-1)I - (J-I) = nI - J$, and since $J = n|u\rangle\langle u|$ where $|u\rangle = \frac{1}{\sqrt{n}}\sum_{i=1}^n |i\rangle$, if $|u_1\rangle,\ldots,|u_{n-1}\rangle,|u\rangle$ is any orthonormal basis of $\mathbb{R}^n$, then $L = n\sum_{i=1}^{n-1}|u_i\rangle\langle u_i| + n|u\rangle\langle u| - n|u\rangle\langle u| = \sum_{i=1}^{n-1} n|u_i\rangle\langle u_i|$, so the spectrum of $L$ is 0, with multiplicity 1, and $n$ with multiplicity $n-1$. Thus, the only nonzero singular value of $A$ is $\sqrt{2n} = \sigma_{\max}(A)$. Furthermore, since $\lambda_2(G_x)$ is the smallest nonzero eigenvalue of $L_x$, and $A(x)A(x)^T = 2L_x$, $\sigma_{\min}(A(x)) = \sqrt{2\lambda_2(G_x)}$. The result follows. ◀

Finally, by scaling $\tau$ to $\frac{\tau}{\|A^+\tau\|} = n\tau$ to get a normalized span program, which has the effect of scaling all positive witnesses by $n$, we can apply Theorem 16 to get an algorithm that makes $\widetilde{O}\left(\frac{\kappa(f)}{\varepsilon}\sqrt{nw_+(x,P)}\right) = \widetilde{O}\left(\frac{1}{\varepsilon}\sqrt{n/\mu}\sqrt{nR_{s,t}}\right)$ calls to $U'(P,x)$. In the full version, we show that this algorithm has time complexity $\widetilde{O}\left(\frac{1}{\varepsilon}n\sqrt{R_{s,t}/\mu}\right)$ and space complexity $O(\log n)$. ◀

Both of our upper bounds have linear dependence on $n$, and this is optimal (see full version). Classically, the best known method of estimating the effective resistance is to compute it, which costs $O(m) = O(n^2)$, where $m$ is the number of edges in the graph. This is accomplished by inverting the Laplacian.

The algorithms from Theorem 18 and 19 are the first quantum algorithms for estimating the effective resistance in the adjacency model, however, the problem has been studied previously in the edge-list model [22], where Wang obtains a quantum algorithm with complexity $\widetilde{O}\left(\frac{d^{3/2}\log n}{\Phi(G)^2\varepsilon}\right)$, where $\Phi(G) \leq 1$ is the conductance (or edge-expansion) of $G$. In the edge-list model, the input $x \in [n]^{[n]\times[d]}$ models a $d$-regular graph (or $d$-bounded degree graph) $G_x$ by $x_{u,i} = v$ for some $i \in [d]$ whenever $\{u, v\} \in E(G_x)$. Wang requires edge-list queries to simulate walking on the graph, which requires constructing a superposition over all neighbours of a given vertex. This type of edge-list query can be simulated by $\sqrt{n/d}$ adjacency queries to a $d$-regular graph, using quantum search, so Wang's algorithm can be converted to an algorithm in the adjacency query model with cost $\widetilde{O}\left(\frac{d^{3/2}}{\Phi(G)^2\varepsilon}\sqrt{\frac{n}{d}}\right)$. We can compare our results to this by noticing that $R_{s,t} \leq \frac{1}{\lambda_2(G)}$ [8], implying that our algorithm always runs in time at most $\widetilde{O}\left(\frac{1}{\varepsilon}\frac{n}{\mu}\right)$. If $G$ is a connected $d$-regular graph, then $\lambda_2(G) = d\delta(G)$, where $\delta(G)$ is the spectral gap of a random walk on $G$. By Cheeger inequalities, we have $\frac{\Phi^2}{2} \leq \delta$ [16], so the complexity of the algorithm from Theorem 19 is at most $\widetilde{O}\left(\frac{1}{\varepsilon}\frac{n}{d\delta}\right) = \widetilde{O}\left(\frac{1}{\varepsilon}\frac{n}{d\Phi^2}\right)$, which is an improvement over the bound of $\widetilde{O}\left(\frac{1}{\varepsilon}\frac{d^{3/2}}{\Phi^2}\sqrt{\frac{n}{d}}\right) = \widetilde{O}\left(\frac{1}{\varepsilon}\frac{d}{\Phi^2}\sqrt{n}\right)$ given by naively adapting Wang's algorithm to the adjacency model whenever $d > \sqrt[4]{n}$. In general our upper bound may be much better than $\frac{1}{\varepsilon}\frac{n}{d\Phi^2}$, since the Cheeger inequality is not tight, and $R_{s,t}$ can be much smaller than $\frac{1}{\lambda_2}$.

## 4 Conclusion and Open Problems

We have presented several new techniques for turning span programs into quantum algorithms, which we hope will have future applications. Specifically, given a span program $P$, in addition to algorithms for deciding any function $f$ such that $f^{-1}(0) \subseteq P_0$ and $f^{-1}(1) \subseteq P_1$, we also show how to get several different algorithms for deciding a number of related threshold problems, as well as estimating the witness size. In addition to algorithms based on the standard effective spectral gap lemma, we also show how to get algorithms by analyzing the real phase gap.

We hope that the importance of this work lies not only in its potential for applications, but in the improved understanding of the structure and power of span programs. A number of very important quantum algorithms rely on a similar structure, using phase estimation of a unitary that depends on the input to distinguish between different types of inputs. Span-program-based algorithms represent a very general class of such algorithms, making them not only important to the study of the quantum query model, but to quantum algorithms in general.

The main avenue for future work is in applications of our techniques to obtain new quantum algorithms. We stress that *any* span program for a decision problem can now be turned into an algorithm for estimating the positive or negative witness size, if these correspond to some meaningful function, or deciding threshold functions related to the witness size. A natural source of potential future applications is in the rich area of property testing problems (for a survey, see [17]).

One final open problem is a possible relationship between estimating the witness size and the HHL algorithm [11]. The HHL algorithm can be used to estimate $\|M^+|u\rangle\|^2$, given the

state $|u\rangle$ and access to a row-computable linear operator $M$. When $M = A(x)$ and $|u\rangle = \tau$, this quantity is exactly $w_+(x)$, so if $A(x)$ is row-computable — that is, there is an efficient procedure for computing the $i^{\text{th}}$ nonzero entry of the $j^{\text{th}}$ row of $A(x)$, then HHL gives us yet another means of estimating the witness size, whose time complexity is known, rather than only its query complexity. We note that the complexity of HHL depends on $\frac{\sigma_{\max}(A(x))}{\sigma_{\min}(A(x))}$, the *condition number* of $A(x)$, which is upper bounded by $\frac{\sigma_{\max}(A)}{\sigma_{\min}(A(x))}$, upon which the complexity of some of our algorithms depends as well. We leave further exploration of this connection for future research.

## References


1. R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48, 2001.
2. A. Belovs. Learning-graph-based quantum algorithm for $k$-distinctness. In *Prooceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2012)*, pages 207–216, 2012.
3. A. Belovs. Span programs for functions with constant-sized 1-certificates. In *Proceedings of the 44th Symposium on Theory of Computing (STOC 2012)*, pages 77–84, 2012.
4. A. Belovs, A. M. Childs, S. Jeffery, R. Kothari, and F. Magniez. Time efficient quantum walks for 3-distinctness. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP 2013)*, pages 105–122, 2013.
5. A. Belovs and B. Reichardt. Span programs and quantum algorithms for $st$-connectivity and claw detection. In *Proceedings of the 20th European Symposium on Algorithms (ESA 2012)*, pages 193–204, 2012.
6. C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing (special issue on quantum computing)*, 26:1510–1523, 1997.
7. G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. In S. J. Lomonaca and H. E. Brandt, editors, *Quantum Computation and Quantum Information: A Millennium Volume*, volume 305 of *AMS Contemporary Mathematics Series Millennium Volume*, pages 53–74. AMS, 2002. `arXiv:quant-ph/0005055v1`.
8. A. K. Chandra, P. Raghavan, W. L. Ruzzo, R. Smolensky, and P. Tiwari. The electrical resistance of a graph captures its commute and cover times. *Computational Complexity*, 6(4):312–340, 1996.
9. R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354, 1998.
10. L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on Theory of Computing (STOC 1996)*, pages 212–219, 1996.
11. A. W. Harrow, A. Hassidim, and S. Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103:150502, Oct 2009.
12. S. Jeffery. *Frameworks for Quantum Algorithms*. PhD thesis, University of Waterloo, 2014. Available at `http://uwspace.uwaterloo.ca/handle/10012/8710`.
13. M. Karchmer and A. Wigderson. On span programs. In *Proceedings of the IEEE 8th Annual Conference on Structure in Complexity Theory*, pages 102–111, 1993.
14. A. Kitaev. Quantum measurements and the Abelian stabilizer problem, 1995. `arXiv:quant-ph/9511026`.
15. T. Lee, R. Mittal, B. Reichardt, R. Špalek, and M. Szegedy. Quantum query complexity of state conversion. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2011)*, pages 344–353, 2011.

**16** D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009.

**17** A. Montanaro and R. de Wolf. A survey of quantum property testing, 2013. arXiv:1310.2035.

**18** B. Reichardt. Span programs and quantum query complexity: The general adversary bound is nearly tight for every Boolean function. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS 2009)*, pages 544–551, 2009.

**19** B. Reichardt. Reflections for quantum query algorithms. In *Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA 2011)*, pages 560–569, 2011.

**20** B. Reichardt and R. Špalek. Span-program-based quantum algorithm for evaluating formulas. *Theory of Computing*, 8(13):291–319, 2012.

**21** M. Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2004)*, pages 32–41, 2004.

**22** G. Wang. Quantum algorithms for approximating the effective resistances in electrical networks, 2013. arXiv:1311.1851.