LA-UR-01-6417
Approved for public release;
distribution is unlimited.

Title: **Quantum Decision Trees and Semidefinite Programming**

Author(s): Howard Barnum, Michael Saks, and Mario Szegedy

Submitted to:

http://lib-www.lanl.gov/cgi-bin/getfile?00818934.pdf

Quantum Decision Trees and Semidefinite Programming

Howard Barnum,[1,2] Michael Saks[2], Mario Szegedy[3]

[1] *CCS-3, MS B256, Los Alamos National Laboratory*
Los Alamos, NM 87545
[2] *DIMACS Center and Department of Mathematics, Rutgers University*
96 Frelinghuysen Road, Piscataway NJ 08854-8018 USA
[3] *Department of Computer Science, Rutgers University, Piscataway NJ 08854-8018 USA*
email: barnum@lanl.gov, saks@math.rutgers.edu, szegedy@cs.rutgers.edu

May 23, 2002

Abstract We reformulate the notion of quantum query complexity in terms of inequalities and equations for a set of positive matrices, which we view as a quantum analogue of a decision tree. Using the new formulation we show that: 1. Every quantum query algorithm needs to use at most n quantum bits in addition to the query register. 2. For any function f there is an algorithm that runs in polynomial time in terms the truth table of f and (for $\epsilon > 0$) computes the ϵ-error quantum decision tree complexity of f. 3. Using the dual of our system we can treat lower bound methods on a uniform platform, which paves the way to their future comparison. In particular we describe Ambainis's bound in our framework. 4. The output condition on quantum algorithms used by Ambainis and others is not sufficient for an algorithm to compute a function with ϵ-bounded error: we show the existence of algorithms whose final entanglement matrix satisfies the the condition, but for which the value of f cannot be determined from a quantum measurement on the accessible part of the computer.)

1 Introduction and summary of results

The bounded-error quantum query model is both relevant to understanding powerful explicit non-query quantum algorithms such as Shor's factoring algorithm [1],[2], and theoretically important as the quantum analogue of the classical decision tree model. Quadratic speedups over the best probabilistic decision trees have been shown for some functions (notably OR of n variables with $O(\sqrt{n})$ queries [3], [4], for which there is a matching lower bound [5]) and examples also exist where only a linear speedup is possible. For partial functions, examples of exponential speedups exist, but for total functions, the gap is known to be at most a degree-6 polynomial [6], while the best gap that has been established for an explicit function is quadratic. It is suspected that the 6th-degree bound on the gap can be considerably strengthened. Many other interesting questions about quantum query complexity, involving both lower and upper bounds, remain open as well.

A computation in a deterministic classical query complexity model may be viewed as a *decision tree*, in which the k-th level of the tree represents the result of the k-th query, and the two branches from each node on the $k-1$st level to its daughters on the k-th level are labeled by the value of the bit queried in step k of the computation. A computation computes a function if at each leaf of the tree, the known bits of the input determine the function's value. In a probabilistic query model, each level has, preceding the binary branching on bit-value, a branching on the index of the bit queried, with each branch labeled by the probability of querying that bit.

In this paper, we provide a view of quantum query computation that we believe is as close as one can come to the decision tree view of classical query complexity, and use it to establish several general results about quantum query complexity, including a bound on the space required to implement a computation in the model, and an algorithm for computing the query complexity of functions. We believe that approaching query complexity from this point of view can also help in establishing new upper and lower bounds. The analogy to classical decision trees is very imperfect, roughly because in a quantum computation the computer state deriving from a particular sequence of queries may interfere

with that derived from another sequence of queries. Nevertheless, the structure we find on this approach yields several interesting results. In particular, we:

1. express the condition that a query algorithm computes f with bounded error in terms of the matrix $M[x,y] = \langle \Psi_x | \Psi_y \rangle$ of inner products ("Gram matrix") of final states of the computer after running the algorithm on inputs x (y);

2. show that the workspace of the computer can be limited to at most n qubits without losing the computational power of the model (as part of Theorem 3).

3. give an algorithm that for $\epsilon > 0$ computes $DQ_\epsilon(f)$ of any $f : \{0,1\}^n \to \{0,1\}$ in DTIME(C^n) for some fixed C (in Section 7).

4. formulate the quantum query algorithm as an evolution of Gram matrices ("quantum decision tree") which gives us a nice and compact definition of the quantum decision tree depth (query complexity). This allows us to formulate the existence of a depth-d decision tree as the feasibility of a semidefinite program. We derive the corresponding dual program, which naturally encompasses possible lower bounds for the primal.

5. find an analogue of what is called "branching according to the value of a variable" in the classical case (see the discussion following Theorem 3).

2 Mathematical preliminaries and notation

Pure quantum states, such as those in a mathematical model of a quantum computer, are vectors in a complex linear space. We use Dirac notation $|i\rangle$ for orthonormal vectors of a "standard" basis of a register, labeled by strings i. Expressions like $|i\rangle|z\rangle$ are the tensor products of these vectors, $|i\rangle \otimes |z\rangle$. For general vectors we omit the bra or ket notation, although we write the inner product of vectors Ψ and Ξ as $\langle \Psi | \Xi \rangle$ (this is is anti-linear in Ψ, linear in Ξ, and is defined by $\langle \Psi | \Xi \rangle := \sum_k \overline{(\Psi)_k} (\Psi)_k$, the overbar denoting complex conjugation. (We generally use * to denote the hermitian conjugate ("adjoint") of a matrix; M^*'s elements are the complex conjugates of those of M's transpose.)

A complex $n \times n$ matrix M is called positive semidefinite if $xMx^* \geq 0$ for every complex row vector x. Any positive semidefinite matrix can be written as a *Gram matrix*: a matrix whose i,j entry is the inner product of the i-th and j-th vectors of a list of n vectors. And any such matrix of inner products is positive semidefinite. If the vectors are normalized, their Gram matrix satisfies $M_{xx} = 1, |M_{xy}| \leq 1$. If we write the components of the vectors Ψ_j as the columns of a matrix Ψ (whose elements are thus $\Psi_{ij} := (\Psi_j)_i$, then we may express their Gram matrix as $\Psi^* \Psi$. Two such systems of n vectors, arrayed in $r \times n$ and $s \times n$ matrices Ψ and Φ respectively (note this means the vectors in Ψ are of length r, while those in Φ are of length s), have the same Gram matrix precisely when there is an $s \times r$ partial isometry Γ taking the Ψ vectors to the Φ vectors: Γ

$$\Gamma \Psi = \Phi. \tag{1}$$

A maximal partial isometry is an $s \times r$ matrix Γ which satisifes that $\Gamma^* \Gamma = \Pi$ is a projector. This implies that $\Gamma \Gamma^*$ is a projector of the same rank. If this rank is maximal, i.e. equal to $\min s, r$ then the partial isometry is called maximal. Thus a maximal $s \times r$ partial isometry satisfies

$$\Gamma^* \Gamma = I_r \tag{2}$$

if $r \leq s$, and if $s \leq r$ satisfies

$$\Gamma \Gamma^* = I_s \ , \tag{3}$$

where I_r is the $r \times r$ identity matrix. If $r \leq s$, Γ may be viewed as the matrix of an inner-product-preserving (isometric) linear mapping from the row space of Ψ (the space in which the vectors of the system Ψ live) to an r-dimensional subspace of the s-dimensional row space of Φ, while if $s \leq r$, it projects onto an s-dimensional subspace of the row space, and then isometrically maps this onto the column space. The point is, this is precisely the sort of map that takes a set of vectors from one space to another of possibly different dimension, without changing their inner products (and hence their Gram matrix) if they are in its domain.

An important application of the notion of partial isometry is:

Proposition 1 *Let x_i be n (unnormalized) complex r-vectors such that*

$$\sum_i x_i x_i^* = I_r \ .$$

Then there exist an $r \times s$ partial isometry Γ with $s \leq n$, and orthonormal s-vectors e_i such that

$$\Gamma e_i = x_i \ .$$

A corollary is a finite-dimensional case of the Naimark extension theorem: that any resolution of unity into positive matrices: $X_i \geq 0 : \sum_i X_i = I_r$ may be "lifted" to a resolution of unity into orthonormal projectors P_i, whose ranks are those of the X_i, in a space of dimension s no greater than the sum of the ranks:

$$X_i = \Gamma P_i \Gamma, \tag{4}$$

with Γ an $r \times s$ partial isometry as before. (Apply Proposition 1 to the eigenvectors of X_i normalized to have norm equal to the square root of the associated eigenvalue.) Such resolutions of unity are called "generalized measurements" in quantum mechanics; for such a measurement the probability of outcome X_i in state w is given by tr $ww^* X_i$, and the conditions on the X_i enforce that these are normalized, positive probabilities. The corollary ensures that such measurements correspond to ordinary quantum-mechanical "von Neumann"measurements of complete sets of projectors, with the usual probabilities $||P_i e||^2$ when the state vector is e, on some larger space into which the space containing the state w is embedded as a subspace, with w embedded as e.

3 Quantum query complexity: definitions

Consider an arbitrary $f : \{0,1\}^n \to \{0,1\}^k$. A d-query computation for f in the quantum query model consists of a sequence of d unitary operators and a particular complete set of orthogonal projection operators P_w acting on the query register and workspace of a quantum query computer. The query register is a complex vector space spanned by orthonormal "standard" basis states $|i\rangle, i \in \{0, ..., n\}$ (meant to represent "which bit of the input $x \in \{0,1\}^n$ is to be queried at the next query step"), while the workspace is a complex vector space spanned by 2^m standard basis states $|z\rangle, z \in \{0,1\}^m$ labelled by the m-bit strings z, where m may be any nonnegative integer. Each U in the sequence acts on the $n \cdot 2^m$-dimensional tensor product of the query register and workspace, i.e. the space spanned by the pairs $|i\rangle|z\rangle$. We will call this the "accessible space" of the quantum computer. Completeness of the projectors P_w means that $\sum_k P_k = I$, I being the identity operator on the accessible space. Thus, defining H_x as the space onto which P_w projects, H as the accessible space, that $H = \bigoplus_{w \in \{0,1\}^k} H_w$.

While this specifies, mathematically, what we mean by a quantum query computation for f, we need to interpret it further, specifying how the algorithm is run to get a result, and how this result is interpreted as a value for f. This will also enable us to quantify the success of the algorithm in computing f.

Besides the accessible space, the computer in our formulation of quantum query computation includes an "input register" spanned by a standard orthonormal basis $|x\rangle, x \in \{0,1\}^n$. The full computer state space is the tensor product of this input register with the accessible space, and thus is spanned by the $n2^{mn}$ orthonormal vectors $|x\rangle|i\rangle|z\rangle$. The unitaries and measurements specifying the algorithm do not act on the input register; it is accessible only by a special type of unitary, the query or oracle operator, described below.

A quantum query computation proceeds as follows: the initial state is $|x\rangle|0\rangle|0\rangle$. It is acted upon sequentially by the unitaries $U_0, O_x, U_1, O_x, U_2, O_x \ldots U_d$. O_x is a phase change operator associated with the "queried" input $x \in \{0,1\}^n$, whose action on the basis vectors is described by:

$$O_x|i\rangle|z\rangle = (-1)^{x_i}|i\rangle|z\rangle. \tag{5}$$

Thus, as mentioned above, the the standard basis vector $|i\rangle$ of the query register specifies which bit of x is queried if the query unitary O_x is applied to the state. In the subspace of the computer space where the query register is set to zero, the query acts as the identity (i.e. the query is "turned off"). For every input $x \in \{0,1\}^n$ the computation leads to the result vector

$$\Psi_x = U_d O_x \ldots U_2 O_x U_1 O_x U_0 |x\rangle|0\rangle|0\rangle. \tag{6}$$

The output is a random variable W with a distribution $\pi_x : \{0,1\}^k \to \mathbf{R}$ given by:

$$\pi_x(w) = ||P_w \Psi_x||^2.$$

The set of projectors onto these subspaces constitutes a standard "von Neumann" quantum measurement on the accessible space of the computer, and the probabilities are the standard quantum-mechanical probabilities for the outcomes of this measurement.

We note a few equivalent (with respect to query complexity defined as number of queries) variants of our definition of a quantum query algorithm. First, we could have specified that the orthonormal projectors P_w be onto subspaces specified by a partition Λ of the standard basis. However, the final pre-measurement unitary U_d in our definition of the algorithm can convert any desired complete set of projectors into a measurement in the standard basis, so this is not a restriction. In a circuit complexity setting, it is critical that the final measurement giving the computer output be in the standard basis—indeed, more is required, namely that the output be encoded in a standard way in a register of the computer; we have equivalence here only because we do not care about the complexity (in terms of number of local gates) of the unitaries such as U_{d+1}, or indeed of the processing necessary to associate the elements of the partition of the standard basis with a pointer we can directly interpret as the value of W. We could also have allowed any "positive operator valued measurement" on the accessible space. However, since we are going to bound the space (in qubits) required to implement query algorithms, we do not make this simplification, because we want to make explicit, and count, the extra space required to implement a POVM as a standard measurement of projectors.

We are interested in computations that compute a function f of the input x, with bounded error. For such computations, the output distribution π_x satisfies:

$$\pi_x(f(x)) \geq 1 - \epsilon \tag{7}$$

for all x and some fixed error parameter $0 \leq \epsilon < 1/2$. That is, the probability of getting the right answer $f(x)$ is no less than $1 - \epsilon$, for all inputs x. The complexity of the quantum query algorithm defined above is the number of queries, d. The minimal d for query algorithms satisfying (7) for f is defined to be the quantum query complexity of f, denoted $DQ_\epsilon(f)$.

4 The geometry of the output vectors

Consider the $2^n \times 2^n$ matrix M whose elements are

$$M[x,y] = \langle \Psi_x | \Psi_y \rangle \tag{8}$$

a.) M is positive semidefinite and Hermitian.

b.) For all $x, y \in \{0,1\}^n$, $|M[x,y]| \leq 1$.

c.) For all x $M_{x,x} = 1$.

Let H be the accessible space associated with a quantum query algorithm. Let H_w be the mutually orthogonal subspaces $H = \bigoplus_{w \in \{0,1\}^k} H_w$ corresponding to the final measurement. We express condition (7) as:

$$\langle P_w \Psi_x | P_w \Psi_x \rangle \geq 1 - \epsilon \tag{9}$$

for every x with $f(x) = w$. Conversely, if the output vectors (depending only on the sequence of unitaries U_i) Ψ_x of a quantum algorithm are such that there exists a complete set of P_w satisfying Equation (9), we can modify our algorithm (by using these projectors for the final measurement) so that for the new algorithm (7) is satisfied. Therefore, if we wished, we could define a quantum query algorithm as a sequence of unitaries, and ϵ-computation of f by the algorithmn as the *existence* of a set of projectors for which (9) is satisified—a step in reducing the our description of a quantum algorithm to the one in terms of positive matrices. In other words, as part of our move towards a semidefinite programming formulation, we have changed the role the projectors P_w from variables describing an algorithm, to objects whose existence is part of a constraint on the remaining variables (the U_i) specifying that they constitute a successful ϵ-bounded errror computation of f.

In order to further simplify expression (9) let Ψ denote the (rectangular) matrix whose columns are the elements of the system $\{\Psi_x | x \in \{0,1\}^n\}$. Since we have $\sum_{w \in \{0,1\}^k} P_w = I$, where I is the identity of H,

$$M = \Psi^* \Psi = \sum_{w \in \{0,1\}^k} \Psi^* P_w^* P_w \Psi =: \sum_{w \in \{0,1\}^k} M_w, \tag{10}$$

where the last equality defines the matrices M_w (which are thus the Gram matrices of the P_w-projected output vectors). Condition (9) is equivalent to the requirement that if $f(x) = w$ then the (x,x) entry of M_w is at least $1 - \epsilon$. All M_w are positive. Therefore M satisfies

Condition O (Output condition): *M can be decomposed into positive matrices: $M = \sum_{w \in \{0,1\}^k} M_w$, such that the (x,x) entry of M_w is at least $1 - \epsilon$ whenever $f(x) = w$.*

Having shown that for any quantum query algorithm computing f with ϵ-bounded error, the matrix M of inner products of output vectors satisfies **O**, we now show that for any $d \times d$ matrix M satisfying Condition **O**, there exists a system of d vectors $\{\Phi_x\}$ and projectors P_w such that M is the inner product matrix of the system $\{\Phi_x\}$ and for all w, M_w is the inner product matrix of the vectors $\{P_w \Phi_x\}$.

This follows from a general proposition.

Proposition 2: Let $M = \sum_{w=1}^{l} M_w$, $M_w \geq 0$ be $n \times n$ matrices. Let M be the Gram matrix of n d-vectors, so that $M = \Psi^* \Psi$ for Ψ a $d \times n$ matrix. Then there are an $nd \times n$ matrix Φ and $nd \times nd$ projectors P_w such that $M_w = \Phi^* P_w \Phi$ (and so $M = \Phi^* \Phi$). Thus, there are also an $nd \times n$ partial isometry Γ, and projectors P_w such that $M_w = \sum_w \Phi^* \Gamma^* P_w \Gamma \Phi$.

In other words, any decomposition of a positive semidefinite matrix representable as a Gram matrix of n length-d vectors as a sum of n positive matrices, can be obtained from n vectors of greater length (nd), projected via the P_w onto *orthogonal* subspaces, and then projected onto a d-dimensional subspace; the inner products of the resulting vectors will give M_w.

Proof: Define $X_w := M^{-1/2} M_w M^{-1/2}$. Then $X_w \geq 0$, $\sum_w X_w = I$, and by Naimark's theorem there are projectors P_w and an $s \times n$ partial isometry Γ such that $X_w = \Gamma P_w \Gamma^*$. $s \leq ld$, since there are l

M_w's each with rank no greater than d. Hence $M_w = \Psi^*\Gamma^* P_w P_w \Gamma^* \Psi^*$, so the matrices $\Phi := \Gamma^*\Psi$ and projectors P_w are those required by the Proposition.

The use of this proposition is that if condition **0** tells us there is a POVM measurement for which outcome X_w has probability greater than $1-\epsilon$ whenever $f(x) = w$, the proposition supplies us with an isometry and a set of final computer state vectors in a larger state space, such that the same M and M_w correspond to measurement outcomes of a set of orthogonal projectors on a larger space. If we have an algorithm A that satisfies **0**, we can supply enough auxiliary qubits of workspace to implement the isometry Γ from A's accessible space to a larger space (extended to be a unitary whose domain includes the auxiliary qubits along with A's accessible space) as a non-query step. In particular, if the workspace and query register contain $d = 2^m$ qubits, and the function f has range $\{0,1\}^k$, so $l = 2^k$, then $s \leq ld = 2^{mk}$ is the dimension of the space required for the projectors P_w and vectors (row space) of Φ, so we need no more than k auxiliary bits of workspace to realize the final measurement by standard-basis projectors instead of POVMs.

We have proven:

Theorem 1 *For every quantum query algorithm that computes a function f and satisfies the output condition (7) of the first section with result vectors $\{\Psi_x\}$ there is an algorithm of the same length, whose system of result vectors $\{\Phi_x\}$ satisfies* **O**. *Also, $\Psi^*\Psi = \Phi^*\Phi$. Similarly, for any quantum query algorithm whose system of result vectors satisfies* **O**, *we can find one of the same length, using no more than k qubits more accessible space, satisfying* (7) *with orthogonal projectors measured on the output.*

In the section after next, we wil lproceed to find conditions on a matrix M that, along with M **a.)-c.)** and **Condition O**, imply the existence of a d-query quantum algorithm or f with ϵ-bounded error. The relationship between this algorithm and M will be the obvious one: M will be the matrix of inner products of output vectors of the algorithm. And we will be able to bound the amount of workspace needed by this algorithm. However, in the next section, we digress to consider a different output condition on the inner-product matrix, which has been much used in deriving lower bounds on quantum query complexity.

5 Ambainis' output condition

If the computation producing M computes f with bounded error (satisfies (7)), then M necessarily satisfies the following condition connecting it with f:

Condition A: For all $x, y \in \{0,1\}^n$ with $f(x) \neq f(y)$ we have

$$|M[x,y]| \leq 2\sqrt{\epsilon(1-\epsilon)} \quad \text{(Ambainis [7])} \tag{11}$$

This necessary condition has been useful in obtaining lower bounds on quantum query complexity. Here we show that (considered with properties **a.)-c.)**) condition **A** is not sufficient to ensure the existence of orthogonal projectors P_w such that (7) is satisfied: Condition **O** on the output matrix is strictly stronger than Condition **A**.

Theorem 2 *There is a function $f : \{0,1\}^n \to \{0,1\}$ and a matrix M that satisfy* **a.)-c.)** *and* **A** *but do not satisfy* **O**.

We leave the proof of this theorem to the appendix.

Remark 1 *Since* **A** *and* **O** *are different output conditions, we may define a new notion of quantum decision complexity based on a.)-c.) and* **A**. *This we call $DQA_\epsilon(f)$. Clearly,*

$$DQA_\epsilon(f) \leq DQ_\epsilon(f),$$

and equality probably does not hold. On the other hand, Ambainis's bound is also a lower bound on $DQA_\epsilon(f)$. It would be interesting to exhibit an f that distinguishes between the two complexity measures. (Theorem 3 does not establish a strict separation between them, because while it does show that there exist computations satisfying the Ambainis output condition for f but not computing f, one could imagine that for every such computation there existed another of the same length computing f.)

6 Space efficient computations and a positive matrix representation of query algorithms

In this section we use the idea of representing algorithms in terms of positive matrices to bound the number of qubits of workspace required so for a d-query algorithm attempting to compute a function f. The bound is a function of the number of queries and the number of bits of f's arguments and values. The space bound is important not only because we are interested in space resources used by query algorithms, but because, by describing query algorithms in terms of a bounded number of positive matrices of bounded size, it allows us, in the next two sections, to formulate the calculation of the query complexity of a given function as the feasibility of a well-defined semidefinite program.

Specifically, this section proves the following theorem.

Theorem 3 *For every r-step quantum query algorithm for $f : \{0,1\}^n \to \{0,1\}^k$ there exists an array of real matrices $M^{(j,i)}$, $j = 0, ..., r, i = 1, ...n$, with the following properties:*
1. $\sum_i M^{(0,i)} = E$, where E is the all-ones matrix.
2. $\sum_i M^{(j+1,i)} = \sum_{i=0}^n D_i M^{(j,i)} D_i$, where D_0 is the identity matrix and for $1 \leq i \leq n$ D_i is the diagonal matrix whose (x,x) entry is $(-1)^{x_i}$.

If in addition the algorithm computes f with ϵ-bounded error, then
3. $M^{(r)} := \sum_i M^{(r,i)}$
satisfies Condition **O**.
4. Conversely, for any sequence of matrices $M^{(j,i)}$ satsfying 1, 2, there exists a quantum query algorithm for f such that $M^{(j,i)}$ are the inner-product matrices of the components $\Psi_{x,i}^{(j)}$ having i in the query register, of the states at time j of a computer whose quantum workspace contains at most n qubits and whose query register contains $\lceil \log n + 1 \rceil$ qubits. If also 3. holds, the algorithm ϵ-computes f.

Proof:

For every $x \in \{0,1\}^n$ define:

$$\Psi_x^{(0)} = U_0|0\rangle|0\rangle \tag{12}$$

$$\Psi_x^{(j)} = U_j O_x \dots U_2 O_x U_1 O_x U_0 \Psi_x^0 =: \sum_i |i\rangle \Psi_{x,i}^{(j)} \tag{13}$$

Note that the state $|0\rangle|0\rangle$ in (12) refers to computer starting with the standard initial state $|0...0\rangle$ of the workspace, and the standard initial state $|0\rangle$ of the query register. Also note that (13) defines the (sub-normalized) states $\Psi_{x,i}^{(j)}$. These are the components of the computer state at time j, having i in the query register.

We define the system of vectors $\Xi_x^{(j)}$ via

$$\Xi_x^{(j)} = O_x \Psi_x^{(j)} = \sum_{i: i=0 \text{ OR } x(i)=0} |i\rangle \Psi_{x,i}^{(j)} - \sum_{i, x(i)=1} |i\rangle \Psi_{x,i}^{(j)}. \tag{14}$$

These are the computer states, on inputs x, just after the $j+1$st query.

Let $M^{(j)}$ denote the matrix whose elements are $\langle \Psi_x^{(j)} \mid \Psi_y^{(j)} \rangle$. Here and in the sequel the elements of all matrices will be indexed by $\{(x,y) \mid x, y \in \{0,1\}^n\}$. Notice that for $1 \leq j \leq r$ the matrix

$M^{(j)}$ equals the matrix whose elements are $\langle \Xi_x^{(j-1)} \mid \Xi_y^{(j-1)} \rangle$, since in the non-query step we perform a unitary transformation of the state space, which leaves the inner product of any two states invariant. For $0 \leq j \leq d$, $0 \leq i \leq n$ we define:

$$M^{(j,i)} \quad \text{to be the matrix whose elements are} \quad \langle \Psi_{x,i}^{(j)} \mid \Psi_{y,i}^{(j)} \rangle.$$

It is obvious that all entries of $M^{(0)}$ are 1, since in the beginning all vectors $\Psi_x^{(0)}$ are the same. That $M^{(j)} = \sum_{i=0}^{n} M^{(j,i)}$, and $M^{(j,i)}$ are all positive matrices follows from the orthogonality of the components of the sum in Equation (13). 2. of the theorem is essentially a restatement of Equation (14). If complex matrices M satisfy $1-3$ of the theorem, so do the real matrices $(M + M^T/2)$. With Theorem 1, this implies that for every quantum query algorithm (and for every quantum query algorithm computing f) there is a sequence of matrices of the form described in the theorem.

We now prove the converse, namely that for any set of matrices described by the theorem, we can find vectors $\Psi_x^{(j)}$, $\Psi_{x,i}^{(j)}$, $\Xi_x^{(j)}$ and unitary transformations $U_0, \ldots, U_d$, that produce these matrices via (12), (13) and (14). We prove this by induction on j. In the beginning, for every input, the state is set to $|000\rangle$, and the matrix of scalar products is clearly all 1's. Also, by Equation (14) the query step, as defined by the sign change operation, automatically satisfies the required transformation of the decomposition (item 2 of the theorem), and the only claim that remains to be proven is that no matter what decomposition $M^{(j)} = \sum_{i=0}^{n} M^{(j,i)}$ we give, there are always vectors $\Psi_{x,i}^{(j)}$, and a unitary transformation U_j such that

1. $M^{(j,i)}[x,y] = \langle \Psi_{x,i}^{(j)} | \Psi_{x,i}^{(j)} \rangle$;
2. $\Psi_x^{(j)} = U \Xi_x^{(j-1)}$ for every x.

Here we keep Definitions (13) and (14), where we interpret the leftmost and rightmost sides of Equation (13) as the definition for $\Psi_x^{(j)}$. The first item of the claim follows from the positivity of the matrices $M^{(j,i)}$, and the fact that the dimension of the workspace is large enough (2^n) that any 2^n by 2^n positive matrix can be represented as the Gram matrix of some system of vectors in it. By Proposition 2, we can represent these vectors indexed by $x \in \{0,1\}^n, i \in \{0, \ldots n\}$ as the projections, from a $\lceil \log n + 1 \rceil 2^n$-dimensional into a 2^n-dimensional space, of the i-th component of some vectors indexed only by x. These i-th components are obtained by projecting with an orthogonal set of projectors P_i, $i \in \{1, \ldots, n\}$. Each P_i has rank no greater than 2^n. The total space may thus be realized as the space of an n-qubit workspace register and a $\lceil \log n + 1 \rceil$-qubit query register. It immediately follows that the Gram matrix of the $\{\Psi_x^{(j)}\}$ system is $\sum_{i=0}^{n} M^{(j,i)} = M^{(j)}$, i.e. the same as that of the $\{\Xi_x^{(j-1)}\}$ system. The latter has Gram matrix $M^{(j)}$, because, as we noted, the query step changes the Gram matrix according to our rules. Then we use the fact that if in a Hilbert space two systems of vectors have the same pairwise scalar products, there is a unitary transformation of the space that takes one system into the other. The role of this unitary is to ensure that the new positive matrices $M^{(j)}$ and $M^{(i,i)}$ are realized, as required in a a quantum algorithm, as inner products of vectors in the workspace register $|\Psi(j)_{x,i}\rangle$ that are projections of the (acessible) computer statevector onto the subspaces S_i of all vectors with a definite value i in the query register. In other words, the projectors P_i corresponding to the Naimark lifting of the new decomposition are transformed from arbitrary projectors (with respect to the vectors $\{\Xi_x^{(j-1)}\}$) to projectors onto subspaces defined by definite values of the query index. This is the U_j required as part of the algorithm whose existence is part 4. of the thoerem. ■

7 Computing the quantum decision tree complexity

The conditions 1.-3. and **O** on the entries of the matrices $M^{(j,i)}$ are all linear, the only non-linear conditions being that these matrices are positive. Therefore, these conditions, which are equivalent

to the existence of an r-query quantum algorithm that ϵ-computes f, are feasibility conditions for a semidefinite program in the variables $M^{(j,i)}$. The matrices have dimension 2^n by 2^n, which gives a $2^{O(n)}$ theoretical running time for computing the ϵ-bounded quantum decision tree complexity by semidefinite programming.

To be more explicit, there is a semidefinite program Π_q whose feasibility is equivalent the existence of a q-query algorithm that ϵ-computes f. We may compute the $D_\epsilon(f)$ by checking the feasibility of Π_q for $1 \leq q \leq n$. Weak feasibility of such a program may be checked in time polynomial in the number of entries of the matrices involved. Each matrix has $(2^n)^2$ entries, ignoring inconsequential duplications due to symmetry), for a total of $q2^{2n}\lceil \log n + 1 \rceil = 2^{O(n)}$, so the time to check all n of these programs is also no more than $2^{O(n)}$. Weak feasibility means that, for our choice of fixed δ, our algorithm returns 0 if the program is farther than δ from a feasible program, and 1 if the program is farther than δ from an infeasible one; it may return either answer if the program is within δ of the feasible/infeasible boundary.

I think we should do a very careful analysis here. The question should be, whether we can genuinely compute the $DQ_\epsilon(f)$ for any fixed ϵ, or rather, whether we can bound it tightly enough so that we can exhibit an algorithm that will get within a (hopefully, known!) linear factor of it asymptotically (we need to formulate this precisely). The latter is more or less what we are interested in for applications to determining asymptotic bounds on the ϵ-query complexity of classes of functions, for we are mainly interested in the asymptotic complexity up to a linear factor, which is ϵ-independent; the ϵ-dependent multiplicative constant is usually not of interest, and I suspect our method will not give it efficiently.

For zero-error query complexity D_0, the complexity can still be effectively computed but the efficiency of the computation is less clear, because only in special cases are polynomial algorithms known for exact feasibility of semidefinite programs.

8 The Dual Problem

Let $M_{i,j,k,l}$, $d_{i,j,l}$ $(1 \leq i,j \leq N; 1 \leq k \leq t; 1 \leq l \leq s)$ be fixed positive real numbers with the property that $M_{i,j,k,l} = M_{j,i,k,l}$ and $d_{i,j,l} = d_{j,i,l}$ for evey setting of l. Try to find positive symmetric $N \times N$ real matrices $X_1, X_2, \ldots, X_t$ such that for every $i \leq i,j \leq N$, $1 \leq l \leq s$:

$$\sum_{k=1}^{t} X_k[i,j] M_{i,j,k,l} \quad \mathbf{rel_{i,j,l}} \quad d_{i,j,l}, \tag{15}$$

where $\mathbf{rel_{i,j,l}} \in \{=, \geq\}$, and $\mathbf{rel_{i,j,l}} = \mathbf{rel_{j,i,l}}$, for evey setting of k and l.

Lemma 1 *The the above system is infeasible if and only if there is a system of $N \times N$ real symmetric matrices $Y_1, \ldots, Y_s$ such that*

1. *For $1 \leq k \leq t$ the $N \times N$ matrices $\left(\sum_{l=1}^{s} Y_l[i,j] M_{i,j,k,l}\right)_{i,j}$ are all positive;*
2. *Whenever $\mathbf{rel_{i,i,l}}$ is $\geq$, we have $Y_l[i,i] \leq 0$.*
3. *$\sum_{i,j,l} d_{i,j,l} Y_l[i,j] < 0$*

Proof: We use the following form of the duality principle: Let $\mathcal{C} \subseteq R^k$ and $\mathcal{D} \subseteq R^m$ be closed convex cones, with dual cones $\mathcal{C}^*$ and $\mathcal{D}^*$. Let M be a real valued $k \times m$ matrix and let $c \in R^k$ and $d \in R^m$. Then by a variant of [7]:

$$\{x \mid x \in \mathcal{C}, d - Mx \in \mathcal{D}\} = \emptyset \iff \{y \mid yM \in \mathcal{C}^*, y \in \mathcal{D}^*, yd < 0\} \neq \emptyset \tag{16}$$

In our case the primal variables are $\{X_k[i,j] \mid 1 \leq k \leq t;\ 1 \leq i \leq j \leq N\}$, and the dual variables are $\{Y_l[i,j] \mid 1 \leq l \leq s;\ 1 \leq i \leq j \leq N\}$. Cone $\mathcal{C}$ is the set of all evaluations of the primal variables, that correspond to positive $X_1, X_2, \ldots, X_t$. Cone $\mathcal{D}$ is the set of those evaluations of the dual variables for

which $Y_l[i,j] = 0$ if $\mathbf{rel_{i,i,l}}$ is the equality, and $Y_l[i,j] \leq 0$ if $\mathbf{rel_{i,i,l}}$ is $\geq$. To the proof of the lemma one needs to observe that $\mathcal{C}^* = \mathcal{C}$, and that $\mathcal{D}^*$ is the set of those evaluations of the dual variables for which $Y_l[i,j] \leq 0$ if $\mathbf{rel_{i,i,l}}$ is $\geq$, and the other variables are unrestricted. By dualizing the original conditions and applying the above remarks we get the lemma. We omit any further detais.

We use this lemma to write down the dual of the semidefinite programming description (criterions 1-3 in Theorem , and output condition **O**, or alternatively **A**) of the quantum query complexity. In this section we assume that f (the function the decision tree computes) is a function from $\{0,1\}^n$ to k different output values (k is an arbitrary fixed number). The inverse images of every output value of f we call a group or an *input group*. (Thus if f is Boolean we have exactly two groups.)

First we get rid of simply indexed $M^{(i)}$s by observing that it is sufficient to write down that:

$$M^{i+1,0} + M^{i+1,1} + \ldots + M^{i+1,n} = M^{i,0} + D_1 M^{i,1} D_1 + \ldots + D_n M^{i,n} D_n \tag{17}$$

for $0 \leq i \leq d-1$. Output condition **O** is written as:

$$M^{d,0} + D_1 M^{d,1} D_1 + \ldots + D_n M^{d,n} D_n = \Sigma_1 + \Sigma_2 + \ldots + \Sigma_k, \tag{18}$$

where $\Delta_i * \Sigma_i \geq (1-\epsilon)\Delta_i$. Here $*$ means the element-wise multiplication of two identical-shaped matrices, the result of which is a matrix with the same shape. In order to express the coefficient matrices of our equations we introduce:

1. E is the all 1 matrix of size 2^n by 2^n;
2. $E^{x_i} = D_i E D_i$.
3. Λ_i are (variable) partial diagonal matrices, with non-zeros only in the diagonal entries corresponding to the i^{th} input group.
4. Δ_i are partial diagonal matrices, with ones in the diagonal entries corresponding to the i^{th} input group.

We have that $A * E = A$ for every matrix A. Also, $A * E^{x_i} = D_i A D_i$. $A * \Delta_i$ is a partial diagonal matrix with non-zero entries in the diagonal entries corresponding to the i^{th} input group and equal to the corresponding elements of A. The next table summarizes these equations, and indicates the names of the primal and the dual variables. Above the horisontal separating line the left hand side equals to the right hand side, and below it the left hand side is greater or equal than the right hand side.

	$M^{(0,0)}$	$\ldots$	$M^{(0,n)}$	$M^{(1,0)}$	$\ldots$	$M^{(1,n)}$		$M^{(d-1,0)}$	$\ldots$	$M^{(d-1,n)}$	Σ_1	$\ldots$	Σ_n	RHS
B_0	E	$\ldots$	E											E
B_1	$-E$	$\ldots$	$-E^{x_n}$	E	$\ldots$	E								$\mathbf{0}$
B_2				$-E$	$\ldots$	$-E^{x_n}$								$\mathbf{0}$
$\vdots$							$\ddots$							$\vdots$
B_{d-1}								E	$\ldots$	E				$\mathbf{0}$
B_d								$-E$	$\ldots$	$-E^{x_n}$	E	$\ldots$	E	$\mathbf{0}$
Λ_1											Δ_1			$(1-\epsilon)\Delta_1$
$\vdots$												$\ddots$	$\vdots$	
Λ_k													Δ_k	$(1-\epsilon)\Delta_k$

The primal variable matrices $M^{i,j}$ $(0 \leq i \leq d; 0 \leq j \leq n)$ and Σ_i are constrained to be non-negative.

Lemma 2 *The primal system is* infeasible *if and only if there are* $B_0, \ldots, B_d$ *and* $\Lambda_1, \ldots, \Lambda_k$ *such that the positivity conditions*

$$\begin{array}{llll} B_0 - B_1, & B_0 - D_1 B_1 D_1, & \ldots, & B_0 - D_n B_1 D_n \in \mathcal{S}; \\ B_1 - B_2, & B_1 - D_1 B_2 D_1, & \ldots, & B_1 - D_n B_2 D_n \in \mathcal{S}; \\ & \vdots & & \\ B_{d-1} - B_d, & B_{d-1} - D_1 B_d D_1, & \ldots, & B_{d-1} - D_1 B_d D_1 \in \mathcal{S}; \end{array} \tag{19}$$

$$B_d + \Lambda_1, \ldots, B_d + \Lambda_k \in \mathcal{S}; \tag{20}$$

as well as the input-output condition

$$\mathbf{IO}_\epsilon : \quad \sum_{x,y} B_0[x,y] + (1-\epsilon) \sum_i tr(\Lambda_i) < 0. \tag{21}$$

It is also required that each Λ_i *has negative entries.*

The proof of the lemma easily comes from Lemma 1. Next we are going to work on rephrasing the constrains of Lemma 2. We first start with some minor changes. We denote: $Y_0 = -B_d, Y_1 = -B_{d-1}, \ldots, Y_d = -B_0$. Then by working on the output conditions the following dual formulations can be obtained:

Dual System (zero error): The primal system is *infeasible* if and only if there are $Y_0, \ldots, Y_d \in \mathcal{S}$ such that:

$$\begin{array}{ll} D_i Y_0 D_i - Y_1 \in \mathcal{P} & \text{for every } 0 \leq i \leq n; \\ D_i Y_1 D_i - Y_2 \in \mathcal{P} & \text{for every } 0 \leq i \leq n; \\ \vdots & \\ D_i Y_{d-1} D_i - Y_d \in \mathcal{P} & \text{for every } 0 \leq i \leq n; \end{array} \tag{22}$$

and the input-output conditions:

$\mathbf{IO}_0$/a: The diagonal blocks of Y_0 that correspond to each input group are identically zero;

$\mathbf{IO}_0$/b: The sum of the entries of Y_d is positive.

Example: Consider $f = x_1 \wedge x_2$. We how that the (zero-error) quantum decision tree complexity of this function is greater than 1 (i.e. it is 2) by giving a Y_0 and Y_1 that satisfies the conditions in (22) as well as **$\mathbf{IO_0}$/a** and **$\mathbf{IO_0}$/b**:

$$Y_0 = \begin{pmatrix} 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 2 \\ -2 & 2 & 2 & 0 \end{pmatrix}; \quad Y_1 = \begin{pmatrix} -6 & 0 & 0 & 5 \\ 0 & -6 & 0 & 5 \\ 0 & 0 & -6 & 5 \\ 5 & 5 & 5 & -11.5 \end{pmatrix}$$

Here the rows and columns of the matrices correspond to inputs 00, 01, 10, 11 in this order.

Dual System (ϵ-error with output condition A): The primal system is *infeasible* if there are $Y_0, \ldots, Y_d \in \mathcal{S}$ that satisfy System (22) and the conditions

$\mathbf{IO}_A$/a: The diagonal blocks of Y_0 that correspond to each input group are identically zero;

$\mathbf{IO}_A$/b: $\sum_{x,y} Y_d[x,y] - 2\sqrt{\epsilon(1-\epsilon)} \sum_{x,y} Y_0[x,y] > 0$

Remark: Notice that all we claim is that the feasibility of the latter system is sufficient for the infeasibility of the primal system. This is because the new system comes from the dualization of a system that is slightly stronger than the original primal, namely we changed the output condition for that of Ambainis. We did so to simplify $\mathbf{IO}_\epsilon$, which is the true condition in the ϵ-error case. We have seen in Section 3 that, at least isolated from the entire algorithm, the Ambainis output condition is strictly weaker than that of the original model. One can easily verify using Lemma 1 that $\mathbf{IO_A}$ indeed corresponds to output condition **A**.

8.1 Ambainis's bound revisited

In order to see how restrictive the lower bound method of Ambainis, we reformulate it into our dual setup, where it gains an elegant form. We will see that the restriction we are looking for is that there is a fixed matrix A and reals $\lambda_1, \ldots \lambda_d$ (the later turns out to be an arithmetic sequence) such that $Y_i = A + \lambda_i I$ for every $0 \leq i \leq d$. In the sequel we denote the smallest eigenvalue of a matrix M by $\mathrm{sev}(M)$.

Theorem 4 *Let f be a function on $\{0,1\}^n$ A be an 2^n by 2^n matrix whose diagonal blocks corresponding to the input groups of f are all zeros. Define*

$$\begin{aligned} \sigma &= \left(1 - 2\sqrt{\epsilon(1-\epsilon)}\right) \sum_{x,y} A[x,y]; \\ \nu &= \max_i -\mathrm{sev}(D_i A D_i - A). \end{aligned}$$

Then $QD_\epsilon(f) \geq \frac{\sigma}{2^n \nu}$.

Proof: We show that $Y_i = A + i\nu I$ satisfy (22) and $\mathbf{IO_A}$ as long as $d < \frac{\sigma}{2^n \nu}$. First, $\mathbf{IO_A}$ obviously holds, since the sum of the entries of $Y_d = A + d\nu I$ is $\sum_{x,y} A_{x,y} - d\nu 2^n > \sum_{x,y} A_{x,y} - \sigma$, and so

$$\sum_{x,y} Y_d[x,y] - 2\sqrt{\epsilon(1-\epsilon)} \sum_{x,y} Y_0[x,y] > \sum_{x,y} A[x,y] - \sigma + 2\sqrt{\epsilon(1-\epsilon)} \sum_{x,y} A[x,y] = 0.$$

We also need to show (22). Since every positivity constraint is of the form

$$D_i \left(A - j\nu I\right) D_i - A + (j+1)\nu I \in \mathcal{S},$$

what we need to show is that the matrix $D_i A D_i - A + \nu I$ is positive for $1 \leq i \leq n$, which follows from the fact that the smallest eigenvalue of the matrix is greater than or equal to 0.■

Example: In the example below we use Theorem 4 for a promise problem related to Grover's algorithm. We show that if we are promised that input x has a single 1 in it, i.e. if it is of the form 0..010..0, then we need $\Omega(\sqrt{n})$ quantum queries to determine the position of the 1 in the ϵ-error setup. After obvious adjustments in Theorem 4 to promise problems, we realize that we can use the n by n matrix A, which is the all 1 matrix minus the unit matrix. The entry-sum of this matrix is $n^2 - n$. The matrix $D_i A D_i - A$ is of the form

$$\begin{pmatrix} 0 & & -2 & & 0 \\ & \ddots & -2 & & \\ -2 & \ldots -2 & 0 & -2 \ldots & -2 \\ & & -2 & 0 & \\ 0 & & -2 & & \ddots \end{pmatrix},$$

and its smallest eigenvalue is $-\theta(\sqrt{n})$. This gives us an $\Omega\left(\frac{n^2}{n\sqrt{n}}\right) = \Omega(\sqrt{n})$ lower bound.

9 Trees and Branching

The notion of classical (deterministic) decision tree is inseparable from the notion of branching. Is there an analogue of branching for the quantum case? The best answer may be that $M^{(j)} = \sum_{i=0}^{n} M^{(j,i)}$ of Theorem 3 can be viewed as an expression for quantum branching, and it is unlikely that we get a substantially better notion.

In order to understand how classical and quantum decision trees relate to each other, we describe classical (deterministic) decision trees in terms of the matrices in Theorem 3. Without loss of generality we can assume that a decision tree of depth d is a complete binary tree of depth d. With every node of the tree we associate the set of inputs that arrive at that node. For a fixed level j these subsets are disjoint, and their union is the entire input set, $\{0,1\}^n$. We can define $M^{(j)}$ such that

1. The (x,y) entry of $M^{(j)}$ is 1 if input x and input input y arrive at the same node on level j.
2. The (x,y) entry of $M^{(j)}$ is 0 if input x and input input y arrive at different nodes on level j.

Clearly, $M^{(j)}$ has a diagonal block structure, where each block is the all 1 matrix corresponding to some node at level j. The decomposition $M^{(j)} = \sum_{i=0}^{n} M^{(j,i)}$ is this:

1. $M^{(j,0)} = \frac{1}{2} M^{(j)}$,
2. $M^{(j,i)}$ is $\frac{1}{2}$ times the sum of those blocks that are associated with those nodes, where we branch on variable x_i.

One can easily see, that $M^{(j)}$ and $M^{(j,i)}$ satisfy the conditions of Theorem 3. Conversely,

Lemma 3 *If a quantum algorithm satisfying the conditions of Theorem 3 also satisfies that each $M^{(j)}$ has a diagonal block structure, where the blocks are all one matrices, and the blocks of $M^{(j+1)}$ are contained in the blocks of $M^{(j)}$, then the quantum decision tree is really classical.*

References

[1] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Proc. 37th ann. symp. on the foundations of computer science*, pp. 56–65, 1994.

[2] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comp.*, pp. 1484–1509, 1997.

[3] Lov Grover, "A fast quantum mechanical algorithm for database search," *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC)*, pp. 212–219, May 1998.

[4] L. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical Review Letters*, pp. 325–328, July 1997.

[5] C. H. Bennett, G. Brassard, E. Bernstein, and U. Vazirani, "Strengths and weaknesses of quantum computing," *SIAM Journal on Computing*, vol. 26, pp. 1510–1523, 1997.

[6] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf, "Quantum lower bounds by polynomials," *FOCS '98*, pp. 352–361, 1998.

[7] A. Ambainis, "Quantum lower bounds by quantum arguments," *Proceedings of the 32nd Annual ACM Symposium on the Theory of Computing (STOC)*, pp. 636–643, 2000.

10 APPENDIX

Proof of Theorem 2: We will set $\epsilon = 1/3$. Let $m = n^\alpha$, where $\alpha > 1$ is a sufficiently large constant to be determined later. Select 2^n points randomly on the surface of the unit sphere of the m dimensional Hilbert space, and identify the system of Ψ_xs with those vectors. **a.)–c.)** are of course satisfied. It is perhaps a surprising aspect of the multi-dimensional geometry that if α is a sufficiently large constant, then with high probability all angles in between the above unit vectors are very close to 90 degree. Fix such a choice of α and vectors. When the angles are close enough to 90 degrees, the matrix $M = \Psi^*\Psi$ satisfies Condition **A** for *any* $f : \{0,1\}^n \to \{0,1\}$.

We will show by a counting argument that there is an f for which **O** does not hold. The argument hinges on fact that for the decomposition $M = M_0 + M_1$ we can assume that

$$M_0 = \Psi^* U^* \Lambda_0^* \Lambda_0 U \Psi \quad (23)$$

$$M_1 = \Psi^* U^* \Lambda_1^* \Lambda_1 U \Psi. \quad (24)$$

Here U, Λ_0 and Λ_1 are m dimensional square matrices, U is unitary, Λ_0 and Λ_1 are real diagonal, and $\Lambda_0^*\Lambda_0 + \Lambda_1^*\Lambda_1 = I$. The positive operators with matrices $U^*\Lambda_0^*\Lambda_0 U$ and $U^*\Lambda_1^*\Lambda_1 U$ form what is called in quantum theory a *generalized measurement*, that is, a set of positive operators that sum to the identity.

Remark 2 *We defined our decompositions of M into positive operators by*

$$M_0 = \Psi^* Z^* P_0^* P_0 Z \Psi \quad (25)$$

$$M_1 = \Psi^* Z^* P_1^* P_1 Z \Psi, \quad (26)$$

where Z is a maximal partial isometry into some (large) space, and P_0 and P_1 are orthogonal projectors of that space. This decomposition would not be sufficient for our purposes, because there is no a priori limit on the dimension of the image space of Z. But by introducing "generalized" or "positive operator valued" measurements instead of the projectors of (2), we are able to work in the m-dimensional space where all Ψ_xs lie. The Naimark extension theorem assures us that the decomposition (23-24) using a generalized measurement may also be expressed in the form (25-26). In (23-24) we also used the fact that, since the measurement has only two outcomes, the corresponding two matrices are diagonalizable with the same U.

Let $N > 12m^6$ be an integer, called the precision parameter. We will approximate the entries of U and Λ_0 with Gauss-rationals, whose denominator is N. We call the matrices with the approximated entries $U^\sim$ and $\Lambda_0^\sim$. Recall that for each function f, if the computation with result-vectors Ψ_x computes the function, with error bounded by ϵ, then there exists a decomposition $M = M_0 + M_1$ that satisfies Condition **O**. Given M, any such decomposition is determined by U and Λ_0, If f and g are different Boolean functions, and a computation with result vectors Ψ_x computes both functions, then Condition **O** requires that two decompositions exist, $M = M_0^f + M_1^f$ and $M = M_0^g + M_1^g$, with properties specified by **O**. Our argument proceeds by showing that:

1. If f and g are different Boolean functions of n Boolean variables, then the best rational approximation $(U^\sim(f), \Lambda_0^\sim(f))$ to the U, Λ_0 pair giving the decomposition for f, is different from the pair $(U^\sim(g), \Lambda_0^\sim(g))$ best approximating the decomposition for g.

2. The number of distinct $(U^\sim, \Lambda_0^\sim)$ pairs are less than 2^{2^n}, the number of Boolean functions with n input bits.

To see 1. notice that for two different Boolean functions, f and g, there is an $x \in \{0,1\}^n$ such that $f(x) \neq g(x)$. Say, $f(x) = 0$ and $g(x) = 1$. For the $(U_f, \Lambda_{0,f})$ pair of f we have $\|\Lambda_{0,f} U_f \Psi_x\|^2 \geq 1 - \epsilon$,

while for the same pair of g we have $\|\Lambda_{0,g} U_g \Psi_x\|^2 \leq \epsilon$. If, nevertheless, the corresponding approximating pairs were the same, at least one of the formulae in which we replace U with $U^\sim$ and Λ_0 with $\Lambda_0^\sim$ would deviate from the original formula by at least $1/6$. We show, however, that we have chosen the precision parameter N high enough that the approximation error of the above formulae is always less than $1/6$. Since all entries of the elements of Ψ_x, U, Λ_0 are less than 1, if we compute the difference

$$||\Lambda_0 U \Psi_x||^2 - ||\Lambda_0^\sim U^\sim \Psi_x||^2$$

the sum we obtain will have the property that each term is at most $2/N$ (we are dealing with complex numbers, and only $2/N$ approximation is guaranteed). The number of summands is at most m^6, and our claim follows.

To verify 2., notice that since the elements of U and Λ_0 have modulus at most one, $U^\sim$ and $\Lambda_0^\sim$ can be described with $2m^2 + m$ integers in the range $[-N, N]$ (two integers for each of the m^2 complex entries of $U^\sim$, and one for each of the m real diagonal elements of Λ_0). Thus the number of different (U, Λ_0) pairs is at most

$$(2N)^{2m^2+m} \ll 2^{2^n}.$$