Quantum NP is Hard for PH

S. Fenner
Computer Science Department, University of Southern Maine, Portland, ME 04104, USA
E-mail: fenner@cs.usm.maine.edu

F. Green
Department of Mathematics and Computer Science, Clark University, Worcester, MA 01610, USA
E-mail: fgreen@black.clarku.edu

S. Homer
Computer Science Department, Boston University, Boston, MA 02215, USA
E-mail: homer@cs.bu.edu

R. Pruim
Department of Mathematics and Statistics, Calvin College, Grand Rapids, MI 49546, USA
E-mail: rpruim@calvin.edu

In analogy with NP, a language L is in the class NQP if the strings in L are exactly those accepted by a polynomial-time quantum machine with non-zero probability. Previously, Adleman, Demarrais and Huang showed that $\mathsf{NQP} \subseteq \mathsf{PP}$. The sharper upper bound $\mathsf{NQP} \subseteq \mathsf{coC_{=}P}$ is implicit in Adleman et al. and results of Fortnow and Rogers. Here we prove that in fact NQP and $\mathsf{coC_{=}P}$ are the same class, by showing that determining whether a quantum computation has a non-zero probability of accepting is hard for $\mathsf{coC_{=}P}$. This implies that NQP is hard for the polynomial-time hierarchy. The hardness result also applies to determining whether a given quantum basis state appears with nonzero amplitude in a superposition, or whether a given quantum bit has positive expectation value at the end of a quantum computation.

1 Introduction

This decade has seen renewed interest and great activity in quantum computing. This interest has been spurred by the clear formal definition of the quantum computing model and by the surprising discovery that some important computational problems which may be classically infeasible are feasible using quantum computers. One central result is Shor's bounded-error polynomial-time algorithms for discrete logarithm and for integer factoring on both a quantum Turing machine [1] and (equivalently) quantum circuits [2]. This opens the possibility that if such machines can be constructed, or effectively simulated, then one can rapidly factor large integers and compromise a good deal of modern cryptography.

While the main research focus has been on finding efficient quantum algorithms for hard problems, attention has also been paid to determining the strength of quantum computation *vis-à-vis* its classical (probabilistic) counterpart[3]. In this paper we take a further step in this direction by proving that testing for non-zero acceptance probability of a quantum machine is classically an extremely hard problem. In fact, we prove that this problem – which we call QAP ("quantum acceptance possibility") and which is complete for NQP (a quantum analog of NP) – is hard for the polynomial-time hierarchy. This is done by showing that NQP is precisely the exact counting class[4] $\mathsf{coC_{=}P}$:

Theorem 1.1 $\mathsf{NQP} = \mathsf{coC_{=}P}$.

$\mathsf{coC_{=}P}$, in turn, is hard for PH under randomized reductions[5,6], and may still be hard even if $\mathsf{P} = \mathsf{NP}$. Thus

Corollary 1.2 *The problem of determining if the acceptance probability of a quantum computation is non-zero (QAP) is hard for the polynomial time hierarchy under polynomial-time randomized reductions.*

Graph Nonisomorphism is an example of a problem in $\mathsf{coC_{=}P}$ that is not known to be in NP. Theorem 1.1 (more precisely, Corollary 3.5 below) shows that there is a quantum machine that takes two graphs as input and accepts with probability zero exactly when the two graphs are isomorphic.

Adleman, Demarrais, and Huang[7] previously showed that $\mathsf{NQP} \subseteq \mathsf{PP}$. The sharper upper bound $\mathsf{NQP} \subseteq \mathsf{coC_{=}P}$ is implicit in their proof and a recent result of Fortnow and Rogers[8]. The main contribution of this paper is to obtain the *lower bound* $\mathsf{coC_{=}P} \subseteq \mathsf{NQP}$.

We prove Theorem 1.1 in Section 3, where it is restated as Corollary 3.5 and Corollary 1.2 is restated as Corollary 3.6. The proof can be easily adapted to show hardness of determining whether any given quantum bit must be zero (or one) with certainty in a quantum computation, or more generally, whether some given quantum state shows up in a superposition with nonzero amplitude. Both of these questions are equivalent to QAP, and therefore also NQP-complete.

Determining non-zero acceptance probability of a *classical* machine is complete for NP, but determining exact accepting probability is much harder: it is hard for $\#\mathsf{P}$. By analogy, one might have hoped QAP would be significantly easier than the problem of determining the exact accepting probability of a *quantum* computation, and possibly even to locate QAP within the polynomial hierarchy. Our work shows that this is probably not the case.

Work of Bennet *et al.*[9] and recently of Fortnow and Rogers[8] has suggested that quantum computation with bounded error probability (BQP) is most likely unable to solve NP-hard problems. Combined with our result, this implies that BQP is even less likely than PH to contain QAP. We take this as evidence that quantum computers, even if implemented, will be unable to amplify exponentially small probabilities to such an extent that they become reliably detectable by means of repeated experiments and observations. This difference between bounded error computation and determining non-zero acceptence probability exists classically as well; in the classical case, bounded error computation corresponds to BPP and determining non-zero acceptence probability corresponds to NP.

The relationship between quantum computing and counting problems has been previously observed [10,8,9]. Our result further strengthens the connections between quantum computation and counting complexity and strengthens previous results in this area by providing the first example of a quantum computation problem whose complexity can be precisely characterized in terms of a counting class.

The essential distinction between classical probabilistic models and quantum machines, and the true source of power in the latter, rests in the fact that the states in a quantum superposition can cancel each other, a phenomenon known as destructive interference. Since many states can be involved in such a cancellation, certain measurable properties of the quantum state can be very sensitive to the *number* of classically accepting paths. Our result, while using and extending the resulting connection between quantum computation and counting problems, also serves to clarify it.

2 Probabilistic and Quantum Computation

We let $\Sigma = \{0, 1\}$. We are interested in decision problems (languages) over Σ^*. Of particular interest are the language

$$\begin{aligned} QAP = \{\langle M, x, 0^t \rangle \mid M &\text{ encodes a quantum machine which has} \\ &\text{non-zero probability of accepting } x \text{ in } t \text{ steps}\}, \end{aligned}$$

and the class NQP, which will be defined at the end of this section.

We review here briefly the models of classical probabilistic computation and quantum computation which we will employ in this paper. Our development is based on Turing machines, but can just as easily be based on quantum circuits [11], which are polynomially equivalent to quantum Turing machines [12]. See the references for more details regarding the models used here [10] as well as equivalent formulations [13]. Those who are already familiar with Turing machine models for quantum computation can skip to the definition of NQP at the end of this section.

A classical probabilistic computation can be viewed as a tree. Each node in the tree is labeled with a configuration (instantaneous description of tape contents, head location and internal state) of the Turing Machine. Edges in the tree are labeled with real numbers in the interval $[0, 1]$, which correspond to the probability of a transition from the parent configuration to the child configuration. Each level of the tree represents one time step (hereafter referred to as a *step*). Throughout this paper we will consider only computations (both classical and quantum) for which the depth of the tree (time) is polynomial in the length of the input. Probabilities can be assigned to a node by multiplying the probabilities along the path from the root to that node. The probability of the computation being in configuration c at time t is the sum of the probabilites assigned to each node at level t that has been assigned configuration c.

In order for such a tree to represent a probabilistic computation, it must be constrained by *locality*, and *classical probability*. Locality constraints require that the probability assigned to the edge from one node to another correspond to the action of one step of a probabilistic Turing machine, so in particular, the probability (1) is non-zero only if a Turing machine could actually make such a transition (thus for example, the only tape cells which can change are the ones which were under a head in the parent configuration), and (2) depends only on that part of the configuration which determines the action of the machine, and not on the rest of the configuration or the location in the tree. Probability constraints require that the sum of all probabilities on any level is always 1. It is equivalent to require that the sum of the probabilities on the edges leaving any node equal 1. For the purposes of complexity considerations, it is usually sufficient to consider probabilities from the set $\{0, \frac{1}{2}, 1\}$. If one considers the probabilistic machine to be a Markov chain, the entire computation can be represented by a matrix which transforms vectors of configurations into vectors of configurations, with the coefficients corresponding to probabilities.

The probability that a machine accepts on input x after t steps is

$$\sum_{c \in \Gamma_{acc}} \Pr[\text{configuration } c \text{ at step } t \mid \text{configuration } c_0 \text{ at step } 0]$$

where Γ_{acc} is the set of all accepting configurations and c_0 is the initial configuration corresponding to an input x. Note that the class NP can be defined in terms of probabilistic machines: A language, L, is in NP if and only if there is a probabilistic machine M and a polynomial p such that

$$x \in L \iff Pr[M \text{ accepts } x \text{ in } p(|x|) \text{ steps}] \neq 0$$

A quantum computation can be similarly represented by a tree, only now the constraints are locality and *quantum probability.* In the quantum computation, the edges are assigned algebraic (see section 4) complex-valued *probability* *amplitudes.* The amplitude of a node is again the product along the path to that node. The amplitude associated with being in configuration c at step t is the sum of the amplitudes of all nodes at level t labeled with c. The probability is the squared absolute value of the amplitude. A configuration c uniquely corresponds to a quantum state, denoted by $|c\rangle$. The states $|c\rangle$, for all configurations c, form an orthonormal basis in a Hilbert space. At each step we consider a quantum computation to be in a superposition $|\varphi\rangle$ of basis states, and write this as

$$\sum_{c \in \Gamma} \alpha_c |c\rangle$$

where α_c is the amplitude of $|c\rangle$. Since the basis states $|c\rangle$ are mutually orthonormal, the amplitude α_c of $|c\rangle$ in a superposition $|\varphi\rangle$ is the inner product of $|c\rangle$ with $|\varphi\rangle$, denoted by $\langle c \mid \varphi\rangle$. The probability of accepting is defined as for the probabilistic computation.

Once again the sum of the probabilities on any level must be 1 ($\sum |\alpha_c|^2 = 1$). As before, a restricted set of amplitudes for local transitions is sufficient, namely rational numbers or square roots of rational numbers. In fact, the machine we construct will only use amplitudes in $\{0, \pm\frac{1}{\sqrt{2}}, \pm 1\}$.It is *not*, however, sufficient to require that the sum of the squares of the amplitudes leaving any node be 1. This is due to the effects of *interference* among the configurations. A quantum computation can also be represented by a matrix which transforms quantum states into quantum states (represented as vectors in a Hilbert space with basis states $|c\rangle$, i.e., states of form $|\varphi\rangle$ as above). To satisfy the constraints of quantum probability, this matrix must be *unitary* (its inverse is its conjugate transpose). In the case where all amplitudes are real numbers, a matrix is unitary if and only if it is orthogonal.

The class NQP can be defined analogously to the class NP by replacing the probabilistic machine with a quantum machine:

Definition 2.1 *A language L is in* NQP *if and only if there is a quantum Turing machine Q and a polynomial p such that*

$$x \in L \iff Pr[Q \text{ accepts } x \text{ in } p(|x|) \text{ steps}] \neq 0$$

Clearly QAP is complete for NQP.

3 Main Result

Theorem 3.2 shows how to design quantum machines for which the resulting amplitude of the unique accepting state is closely related to some given function in the class GapP. Before giving the proof, we define this class of functions.

Definition 3.1 *Given any $L \subseteq \Sigma^*$, let $L_x = \{y \in \Sigma^* \mid \langle x, y\rangle \in L\}$. A function $f : \{0,1\}^* \to \mathbf{Z}$ is in* GapP *if there is a language L in* P *and an integer k such that,*

$$f(x) = \frac{|\Sigma^{n^k} \cap L_x| - |\Sigma^{n^k} - L_x|}{2},$$

where $n = |x|$.

This is equivalent to saying that a GapP function is the difference (gap) between the number of accepting paths and the number of rejecting paths in some nondeterministic polynomial time computation. More information can be found in the references [14] about the intuition behind this definition and the basic properties of the class GapP.

Now we are ready to prove the technical theorem on which Theorem 1.1 rests.

Theorem 3.2 *For any $f \in \mathsf{GapP}$, there is a ptime quantum Turing machine Q and a polynomial p such that, for all x of length n,*

$$\Pr[Q(x) \text{ accepts}] = \frac{f(x)^2}{2^{p(n)}}.$$

In fact, for all x, $Q(x)$ has a unique accepting configuration which it reaches with probability amplitude exactly $-f(x)/2^{p(n)/2}$.

Proof Sketch: Our proof directly uses techniques of Simon [10] and Deutsch and Jozsa [15]. Let $k \in \mathbf{N}$ and let $L \subseteq \Sigma^*$ be a set in P such that for all x of length n,

$$f(x) = \frac{|\Sigma^{n^k} \cap L_x| - |\Sigma^{n^k} - L_x|}{2}.$$

Let M be a polynomial time machine recognizing L, so that for all $\langle x, y\rangle$, $\langle x, y\rangle \in L$ iff M accepts on input $\langle x, y\rangle$. Fix an input x of length n and let $m = n^k$. When our quantum machine Q takes x on its read-only input tape, it will use $m+1$ bits of a special work tape t. It will use other work tapes only for deterministic, reversible computation. We denote a possible configuration of $Q(x)$ as a basic state

$$|x, \mathbf{y}, b\rangle$$

where x is the contents of the input tape and $\mathbf{y}, b$ are the contents of t ($\mathbf{y}$ is a vector of m bits, and b is a single bit). We suppress the other configuration information, i.e., the state of Q, the positions of the heads, and the contents of the other work tapes. This other information is irrelevant because at all important steps of the computation, the same state and head positions of Q will appear in all configurations in the superposition, and all other work tapes besides t will be empty.

Initially, $\mathbf{y} = \mathbf{0}$ and $b = 0$. Q first scans over all the bits of $\mathbf{y}$ and applies to each bit what has become a useful and popular local transition rule

$$\begin{aligned} |0\rangle &\mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |1\rangle &\mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned}$$

In general, scanning an arbitrary state $|x, \mathbf{y}, b\rangle$ in this way yields

$$|x, \mathbf{y}, b\rangle \mapsto \frac{1}{2^{m/2}} \sum_{\mathbf{y}'} (-1)^{\mathbf{y}\cdot\mathbf{y}'} |x, \mathbf{y}', b\rangle,$$

where $\mathbf{y}\cdot\mathbf{y}'$ is the dot product $\sum_{i=1}^{m} y_i y_i'$ of the bit vectors $\mathbf{y}$ and $\mathbf{y}'$. The above transformation [15,10] is called the Fourier transform of the basis $|x, \mathbf{y}, b\rangle$. Thus Q scanning the first m bits of the tape t corresponds to the global transition

$$|x, \mathbf{0}, 0\rangle \mapsto \frac{1}{2^{m/2}} \sum_{\mathbf{y}} |x, \mathbf{y}, 0\rangle.$$

Q then simulates the deterministic computation of M on input $\langle x, \mathbf{y}\rangle$ in a reversible manner [11,16], using other work tapes[a]. Let $b_{\mathbf{y}}$ be the one-bit result of the computation of $M(x, \mathbf{y})$. Q sets $b = b_{\mathbf{y}}$. The superposition is now

$$\frac{1}{2^{m/2}} \sum_{\mathbf{y}} |x, \mathbf{y}, b_{\mathbf{y}}\rangle.$$

Afterwards, Q repeats the scan it performed at the beginning, using the same local transformation rule, except that it now includes all $m+1$ bits, including b, in the scan. This leads Q into a new superposition

$$|\psi\rangle = \frac{1}{\sqrt{2}} \frac{1}{2^m} \sum_{\mathbf{y}} \sum_{\mathbf{y}', b'} (-1)^{\mathbf{y}\cdot\mathbf{y}' + b_{\mathbf{y}} b'} |x, \mathbf{y}', b'\rangle.$$

We now consider the coefficient of $|x, \mathbf{0}, 1\rangle$ in $|\psi\rangle$:

$$\langle x, \mathbf{0}, 1 \mid \psi\rangle = \frac{1}{\sqrt{2}} \frac{1}{2^m} \sum_{\mathbf{y}} (-1)^{\mathbf{y}\cdot\mathbf{0} + b_{\mathbf{y}} 1}$$

[a]This computation is also done obliviously so that the internal state and tape head position of the machine is the same for all components of the superposition at any given time. If we had used quantum circuits for the proof, this technicality would have been unnecessary.

$$= \frac{1}{\sqrt{2}}\frac{1}{2^m}\sum_{\mathbf{y}}(-1)^{b_{\mathbf{y}}}$$
$$= -\frac{1}{\sqrt{2}}\frac{1}{2^{m-1}}f(x).$$

Finally, Q deterministically looks at the $m+1$ bits of the tape t. If it sees $\mathbf{0}, 1$ it accepts; otherwise, it rejects.

Thus $|x, \mathbf{0}, 1\rangle$ is the unique accepting configuration of Q, and it has probability amplitude

$$-\frac{1}{\sqrt{2}}\frac{1}{2^{m-1}}f(x)$$

which implies the theorem by setting $p(n) = 2m - 1 = 2n^k - 1$. □

A converse to Theorem 3.2 directly follows from work of Fortnow and Rogers [8].

Theorem 3.3 (Fortnow, Rogers) *For any ptime quantum machine M (with transition amplitudes that are positive or negative square roots of rational numbers), there is a* GapP *function f, a natural number d, and a polynomial p such that M accepts any input x with probability exactly $f(x)/d^{p(|x|)}$.*

Combining Theorems 3.2 and 3.3 provides an exact characterization of NQP in terms of a counting class known to be hard for PH.

Definition 3.4 *A language L is said to be in the class* $\mathsf{C_{=}P}$ *if there is a* GapP *function f such that for any x, $x \in L$ if and only if $f(x) = 0$. The class* $\mathsf{coC_{=}P}$ *is the set of all languages with complements in* $\mathsf{C_{=}P}$.

Corollary 3.5 *A language L is in* $\mathsf{C_{=}P}$ *(resp.,* $\mathsf{coC_{=}P}$*) if and only if there is a polynomial-time quantum Turing machine Q such that for any x,*

$$x \in L \iff \Pr[Q(x) \textit{ accepts}] = 0 \ \textit{(resp., } \Pr[Q(x) \textit{ accepts}] \neq 0\textit{).}$$

Thus $\mathsf{NQP} = \mathsf{coC_{=}P}$.

It is known that $\mathsf{C_{=}P}$ is hard for the polynomial hierarchy under randomized reductions [6,17]. Thus

Corollary 3.6 *QAP is hard for* PH *under randomized reductions.*

Hence if QAP is anywhere in PH, then PH collapses; in fact, the counting hierarchy also collapses.[b] Combining our results with those of Fortnow and Rogers [8], we find that $QAP \in \mathsf{BQP}$ also implies the collapse of the counting hierarchy.

[b] This is a hierarchy built over the class PP instead of NP. The counting hierarchy was originally defined in terms of counting quantifiers [4]. The assertion follows from an alternative characterization in terms of oracles [18].

4 Robustness of NQP

In our definition of NQP we assumed that the probability amplitudes were algebraic. In this section we want to explore briefly the extent to which this is significant. Let NQP_S be the class defined like NQP, but with amplitudes taken from the set S. So $\mathsf{NQP} = \mathsf{NQP}_{\overline{\mathbf{Q}}}$, where $\overline{\mathbf{Q}}$ is the algebraic complex numbers. Similar notation applies to other quantum classes.

Adleman et al.[7] show that $\mathsf{BQP} = \mathsf{BQP}_{\overline{\mathbf{Q}}} = \mathsf{BQP}_{\mathbf{Q}} = \mathsf{BQP}_{\{0,\pm 3/5,\pm 4/5,\pm 1\}}$. The proof of Theorem 3.2 shows that $\mathsf{coC_{=}P} \subseteq \mathsf{NQP}_{\{0,\pm\frac{1}{\sqrt{2}},\pm 1\}}$, and it can be modified to show that $\mathsf{coC_{=}P} \subseteq \mathsf{NQP}_{\{0,\pm 3/5,\pm 4/5,\pm 1\}}$ as well. These inclusions together with Corollary 4.2 below show that $\mathsf{NQP} = \mathsf{NQP}_{\overline{\mathbf{Q}}} = \mathsf{NQP}_{\{0,\pm 3/5,\pm 4/5,\pm 1\}}$, generalizing a theorem of Fortnow-Rogers (Theorem 3.3).

We begin by recalling some basic facts from algebra. Let $\alpha_1,\ldots,\alpha_n$ be complex numbers. Let $\mathbf{Q}(\alpha_1,\ldots,\alpha_n)$ be the smallest subfield of $\mathbf{C}$ containing $\alpha_1,\ldots,\alpha_n$. A basic fact of abstract algebra is that $\alpha_1,\ldots,\alpha_n$ are all algebraic (over $\mathbf{Q}$) iff $\mathbf{Q}(\alpha_1,\ldots,\alpha_n)$ (as a vector space over $\mathbf{Q}$) is finite dimensional.

The main theorem of this section follows.

Theorem 4.1 *Let M be any quantum accept/reject TM that has algrebraic transition amplitudes and runs in polynomial time. Then there are real algebraic numbers $\alpha_1,\ldots,\alpha_n$ linearly independent over $\mathbf{Q}$, and* GapP *functions $f_1,\ldots,f_n$, such that for any input x,*

$$\Pr[M(x) \text{ accepts}] = \frac{1}{D}\sum_{j=1}^{n} f_j(x)\alpha_j$$

for some integer D depending only on $|x|$.

Proof Sketch: The transition amplitudes mentioned in M (not necessarily real), together with their complex conjugates, generate a field F that has finite dimension over $\mathbf{Q}$ and that is closed under complex conjugate. Let $\beta_1,\ldots,\beta_m$ be a basis for F. Every element of F can be expressed uniquely as a linear combination of the β_i. Furthermore, there are unique rationals $\{q_{i,j,k}\}_{1\le i,j,k\le m}$ such that $\beta_i\beta_j = \sum_k q_{i,j,k}\beta_k$. Hence for any two elements $a = \sum a_i\beta_i$ and $b = \sum b_i\beta_i$ of F, the coefficient of β_k in ab is $\sum_i\sum_j a_i b_j q_{i,j,k}$. Now let $\alpha_1,\ldots,\alpha_n$ be any basis of $F\cap\mathbf{R}$ over $\mathbf{Q}$.

We may assume WLOG that the $q_{i,j,k}$ are all integers. If not, we redifine the basis to clear all the denominators.

Fix any input x, and let U be the (exponentially large) global unitary 1-step transition matrix for $M(x)$. U only depends on $|x|$. It is clear that each entry of U is in F, and moreover there is an integer d and an integer-valued

FP function u such that the (i,j)th entry of U is

$$(U)_{i,j} = \frac{1}{d}\sum_{k=1}^{m} u(x,i,j,k)\beta_k.$$

The proof now proceeds as in the proof of lemma 3.2 of Fortnow and Rogers [8], except that here we add and multiply elements of F. Multiplying U times itself then reduces to obtaining exponential sums of polynomial products of the $u(x,i,j,k)$'s and $q_{i,j,k}$. But GapP is closed under these operations. So there are GapP functions $g_1,\ldots,g_m$ such that the (i,j)th entry of U^t is

$$(U^t)_{i,j} = \frac{1}{d^t}\sum_{k=1}^{m} g_k(x,0^t,i,j,k)\beta_k.$$

Now take t to be the running time of $M(x)$. The acceptance probability is the sum of squared absolute values of all entries in "accepting" rows of U^t. Again using the closure properties of GapP, there are GapP functions $h_1,\ldots,h_m$ such that

$$\Pr[M(x) \text{ accepts}] = \frac{1}{d^t}\sum_{i=1}^{m} h_i(x)\beta_i.$$

Since this quantity is real, there is a fixed integer e and integer-valued functions $f_1,\ldots,f_n$ such that

$$\Pr[M(x) \text{ accepts}] = \frac{1}{d^t e}\sum_{j=1}^{n} f_j(x)\alpha_j.$$

Further, the f_j are obtained from the h_i by a fixed set of field operations, so the f_j are all in GapP. Setting $D = d^t e$ proves the theorem. $\square$

Corollary 4.2 *For M as above, the set $\{x \mid \Pr[M(x)\ accepts] = 0\}$ is in $\mathsf{C_=P}$.*

Proof. Since the α_j are all linearly independent over $\mathbf{Q}$, the probability is zero iff all the $f_j(x)$ are zero, iff $f(x) = 0$ where

$$f(x) = \sum_{j=1}^{n}[f_j(x)]^2.$$

The function f is clearly in GapP. $\square$

5 Conclusion

One may ask if a polynomial-time probabilistic Turing machine has a non-zero acceptance probability. This problem is NP-complete. QAP is the analogous problem in the quantum setting and it is NQP-complete. As we have seen in this paper, $\mathsf{NQP} = \mathsf{coC_{=}P}$, which is a much harder class than NP, and our characterization shows that QAP is nowhere in the polynomial hierarchy unless the polynomial hierarchy and the counting hierarchy collapse and are equal.

We interpret this as a lower bound on the capabilities of quantum computers. Just as it is unlikely that an NP machine's acceptance probability can be amplified (i.e., that $\mathsf{NP} \subseteq \mathsf{BPP}$), so is it unlikely that a quantum machine's acceptance probability can be amplified (i.e., $\mathsf{coC_{=}P} \subseteq \mathsf{BQP}$), and even more unlikely that it can be amplified classically (i.e., $\mathsf{coC_{=}P} \subseteq \mathsf{BPP}$). To our knowledge, this is the first hardness result of this nature regarding quantum computation. The result also shows how destructive interference can lead to vastly different behaviors for acceptance probabilities in classical and quantum machines.

Note that the results here show that if $\mathsf{NQP} \subseteq \mathsf{BQP}$, then the counting hierarchy collapses to PP. It would be interesting to see if it collapses even farther (say, to BQP). This would give us a better understanding of how much harder NQP is than BQP.

Acknowledgements

The work of S. Fenner was supported in part by the NSF under grant NSF-CCR-95-01794. The work of S. Homer was supported in part by the NSF under grant NSF-CCR-94-00229. The work of R. Pruim was done while visiting the Computer Science Department at Boston University. We thank C. Pollett, J. Watrous and the referees for helpful comments.

References

1. P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 124–134. IEEE, 1994.
2. P. W. Shor. Polynomial-time algorithms for prime number factorization and discrete logarithms on a quantum computer. *SIAM J. Comp.*, 26:1484–1509, 1997.
3. A. Berthiaume and G. Brassard. The quantum challenge to structural complexity theory. In *Proceedings of the 7th IEEE Structure in Com-*

plexity Theory Conference, pages 132–137. IEEE, 1992.
4. K. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23:325–356, 1986.
5. S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.
6. S. Toda and M. Ogiwara. Counting classes are at least as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 21(2):316–328, 1992.
7. L. Adleman, J. DeMarrais, and M. Huang. Quantum computability. *SIAM Journal on Computing*, 26:1524–1540, 1997.
8. L. Fortnow and J. Rogers. Complexity limitations on quantum computation. In *Proceedings of the 13th IEEE Conference on Computational Complexity*, pages 202–209. IEEE, 1998.
9. C. H. Bennet, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computation. *SIAM Journal on Computing*, 26:1510–1523, 1997.
10. D. Simon. On the power of quantum computation. *SIAM Journal on Computing*, 26:1474–1483, 1997.
11. D. Deutsch. Quantum theory, the church-turing principal, and the universal quantum computer. In *Proceedings of the Royal Society of London*, pages 97–117, 1985.
12. A. C.-C. Yao. Quantum circuit complexity. In *Proceedings of the 34th IEEE Symposium on Foundations of Computer Science*, pages 352–361, 1993.
13. A. Berthiaume. Quantum computation. In L. Hemaspaandra and A. L. Selman, editors, *Complexity Theory Retrospective II*, chapter 2, pages 23–50. Springer-Verlag, 1997.
14. S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48(1):116–148, 1994.
15. D. Deutsch and R. Jozsa. Rapid solutions of problems by quantum computation. In *Proceedings of the Royal Society of London*, pages 553–558, 1992.
16. P. A. Benioff. Quantum mechanical hamiltonian models of turing machines. *Journal of Statistical Physics*, 29:515–546, 1982.
17. J. Tarui. Probabilistic polynomials, $\mathrm{AC}^{(0)}$ functions and the polynomial-time hierarchy. *Theoretical Computer Science*, 113:167–183, 1993.
18. J. Torán. Complexity classes defined by counting quantifiers. *J. Assoc. Comput. Mach.*, 38(3):753–774, 1991.