1998-04-02

Determining Acceptance Possibility for a Quantum Computation is Hard for PH

Determining Acceptance Possibility for a Quantum Computation is Hard for PH

Stephen Fenner [*]
University of Southern Maine

Frederic Green [†]
Clark University

Steven Homer [‡]
Boston University

Randall Pruim [§]
Boston University
Calvin College

November 14, 1997

Abstract

It is shown that determining whether a quantum computation has a nonzero probability of accepting is at least as hard as the polynomial time hierarchy. This hardness result also applies to determining in general 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 seem to be classically infeasible are feasible using quantum computers. One central result is Shor's bounded-error polynomial-time algorithm for

[*]Computer Science Department, University of Southern Maine, Portland, ME 04104. E-mail: fenner@cs.usm.maine.edu. Supported in part by the NSF under grant and CCR 95-01794.

[†]Department of Mathematics and Computer Science, Clark University, Worcester, MA 01610. E-mail: fgreen@black.clarku.edu.

[‡]Computer Science Department, Boston University, Boston, MA 02215. E-mail: homer@cs.bu.edu. Supported in part by the NSF under grant NSF-CCR-9400229.

[§]Computer Science Department, Boston University, Boston, MA 02215. E-mail: rpruim@calvin.edu. On leave from Department of Mathematics and Statistics, Calvin College, Grand Rapids, MI 49546.

discrete logarithm and integer factoring on both a quantum Turing machine [Sho94] and (equivalently) quantum circuits [Sho97]. 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 [BB92]. 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") is hard for the polynomial-time hierarchy, by showing that QAP is equivalent to the problem of exact counting [Wag86a]. Exact counting, in turn, is hard for PH under randomized reductions [Tod91, TO92], and may still be hard even if $\mathsf{P} = \mathsf{NP}$. Our main result is

Theorem 1.1 *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.*

We prove Theorem 1.1 in Section 3. 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.

Determining non-zero acceptance probability of a *classical* machine is NP-complete, whereas 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 not the case.

Work of Bennet *et al.* [BBBV] and recently of Fortnow and Rogers [FR97] 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, as mentioned before, corresponds to NP.

The relationship between quantum computing and counting problems has been previously observed ([Sim94, FR97, BBBV]). 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. The method employed here is to prove the hardness of QAP by giving an exact characterization of QAP in terms of counting problems.

2 Probabilistic and Quantum Computation

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

$$QAP = \{\langle M, x\rangle \mid M \text{ encodes a quantum machine which has non-zero probability of accepting on input } x\}.$$

We review here briefly the models of classical probabilistic computation and quantum computation which we will employ in this paper. Those who are already familiar with quantum models of computation can skip the rest of this section. Our development is based on Turing machines, but can just as easily be based on quantum circuits [Deu85], which are polynomially equivalent to quantum Turing machines [Yao93]. See the references for more details regarding the models used here (e.g., [Sim94]) as well as equivalent formulations (e.g., [Ber97]).

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 obtained by adding the probabilites assigned to each node at level t which 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 transition from one node to another (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) the probabilities depend only on the configurations and not on their location in the tree, thus a configuration may label more than one node in the tree, but in each case the subtree below is identical. 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.

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 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 iff it is orthogonal.

3 Main Result

Theorem 1.1 follows immediately from Corollary 3.5 below, which precisely characterizes the difficulty of testing a quantum computer for non-zero accepting probability in terms of a counting class known to be hard for PH. This corollary follows from Theorem 3.2, which 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 must 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*

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

See [FFK94] for more information 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 $f \in$ 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 [Sim94] and Deutsch and Jozsa [DJ92]. Let $k \in \mathbf{N}$ and let $L \subseteq \Sigma^*$ be a set in P such that for all x of length n,

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

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

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

where x is the contents of the input tape and $\vec{y}, b$ are the contents of t ($\vec{y}$ 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, $\vec{y} = \vec{0}$ and $b = 0$. Q first scans over all the bits of $\vec{y}$ 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 $|x, \vec{y}, b\rangle$ in this way yields

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

where $\vec{y} \cdot \vec{y}'$ is the dot product of the bit vectors $\vec{y}$ and $\vec{y}'$ [DJ92, Sim94]. Thus Q scanning the first m bits of the tape t corresponds to the global transition

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

Q then simulates the deterministic computation of $L(x, \vec{y})$ in a reversible manner, using other work tapes [Deu85, Ben82].[1] Let $b_{\vec{y}}$ be the one-bit result of the

[1] 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 used quantum circuits for the proof, this technicality goes away.

computation of $L(x, \vec{y})$. Q sets $b = b_{\vec{y}}$. The superposition is now

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

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

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

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

$$\begin{aligned} \langle x, \vec{0}, 1 \mid \psi \rangle &= \frac{1}{\sqrt{2}} \frac{1}{2^m} \sum_{\vec{y}} (-1)^{\vec{y} \cdot \vec{0} + b_{\vec{y}} 1} \\ &= \frac{1}{\sqrt{2}} \frac{1}{2^m} \sum_{\vec{y}} (-1)^{b_{\vec{y}}} \\ &= -\frac{1}{\sqrt{2}} \frac{1}{2^{m-1}} f(x). \end{aligned}$$

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

Thus $|x, \vec{0}, 1\rangle$ 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 [FR97].

Theorem 3.3 (Fortnow, Rogers) *For any ptime quantum machine M (whose transition amplitudes 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 $f(x)/d^{p(|x|)}$.*

QAP provides a complete characterization of the following known complexity classes. The characterization is immediate from the definition of QAP and from Theorems 3.2 and 3.3.

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, $x \in L$ iff $f(x) = 0$. The class* co-$\mathsf{C_{=}P}$ *is the set of all languages with complements in* $\mathsf{C_{=}P}$.

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

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

Thus, QAP is complete for $\mathsf{co\text{-}C_=P}$.

Graph Nonisomorphism is an example of a problem in $\mathsf{co\text{-}C_=P}$ that is not known to be in NP. Corollary 3.5 shows that there is a quantum machine that takes two graphs as input and accepts with probability zero iff the two graphs are isomorphic.

It is known that $\mathsf{C_=P}$ is hard for the polynomial hierarchy under randomized reductions [TO92, Tar93].

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[2] also collapses. Combining our results with those of Fortnow and Rogers [FR97], we find that $QAP \in \mathsf{BQP}$ also implies the collapse of the counting hierarchy.

4 Conclusion

One may ask if a polynomial-time nondeterministic Turing machine has a non-zero acceptance probability. This problem exactly characterizes the class NP. QAP is the analogous problem in the quantum setting and, as we have seen in this paper, exactly characterizes the class $\mathsf{co\text{-}C_=P}$. This 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., $\mathsf{co\text{-}C_=P} \subseteq \mathsf{BQP}$), and even more unlikely that it can be amplified classically (i.e., $\mathsf{co\text{-}C_=P} \subseteq \mathsf{BPP}$). 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.

There are a number of problems left open. For the purposes of BQP computation it is sufficient to use a restricted set of *rational* amplitudes for quantum computation, namely amplitudes in the set $\{-1, -\frac{4}{5}, -\frac{3}{5}, 0, \frac{3}{5}, \frac{4}{5}, 1\}$ (see [ADH97, SY96]). However, it is not clear if our method would work using only such amplitudes.

[2] This is a hierarchy built over the class PP instead of NP. See [Wag86b] for a definition.

Also, we found here that if $QAP \in \mathsf{BQP}$, then the counting hierarchy collapses to PP. It would be interesting to see if it collapses even further (say, to BQP). This would give us a better understanding of how much harder QAP is than BQP.

References

[ADH97] L. Adelman, J. DeMarrais, and M. Huang. Quantum computability. *sicom*, 26:1524–1540, 1997.

[BB92] Andrè Berthiaume and Gilles Brassard. The quantum challenge to structural complexity theory. In *Proceedings of the 7th IEEE Structure in Complexity Theory Conference*, pages 132–137. IEEE, 1992.

[BBBV] C.H. Bennet, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computation. Manuscript.

[Ben82] P.A. Benioff. Quantum mechanical hamiltonian models of turing machines. *Journal of Statistical Physics*, 29:515–546, 1982.

[Ber97] André Berthiaume. Quantum computation. In L. Hemaspaandra and A. L. Selman, editors, *Complexity Theory Retrospective II*, chapter 2, pages 23–50. Springer-Verlag, 1997.

[Deu85] D. Deutsch. Quantum theory. In *Proceedings of the Royal Society of London*, pages 97–117, 1985.

[DJ92] D. Deutsch and R. Jozsa. Rapid solutions of problems by quantum computation. In *Proceedings of the Royal Society of London*, pages 553–558, 1992.

[FFK94] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48(1):116–148, 1994. An earlier version appeared in *Proceedings of the 6th Annual IEEE Structure in Complexity Theory Conference*, 1991, pp. 30–42.

[FR97] Lance Fortnow and John Rogers. Complexity limitations on quantum computation. Unpublished manuscript, 1997.

[Sho94] Peter 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.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime number factorization and discrete logarithms on a quantum computer. *SIAM J. Comp.*, 26:1484–1509, 1997.

[Sim94] D. Simon. On the power of quantum computation. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 124–134. IEEE, 1994.

[SY96] R. Solovay and A. Yao. Manuscript., 1996.

[Tar93] J. Tarui. Probabilistic polynomials, $AC^{(0)}$ functions and the polynomial-time hierarchy. *Theoretical Computer Science*, 113:167–183, 1993.

[TO92] 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.

[Tod91] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.

[Wag86a] K. Wagner. Compact descriptions and the counting polynomial time hierarchy. *Acta Informatica*, 23:325–356, 1986.

[Wag86b] K. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23:325–356, 1986.

[Yao93] A.C.-C. Yao. Quantum circuit complexity. In *Proceedings of the 34th IEEE Symposium on Foundations of Computer Science*, pages 352–361, 1993.