Oracle Quantum Computing

André Berthiaume * and Gilles Brassard †

Département IRO
Université de Montréal.
C.P. 6128, Succursale "A"
Montréal (Québec) Canada H3C 3J7

Abstract

This paper continues the study of the power of oracles to separate quantum complexity classes from classical (including probabilistic and nondeterministic) complexity classes, which we initiated in an earlier paper [4]. The new results are that, under appropriate oracles, (1) there is a decision problem that can be solved in polynomial time on the quantum computer, which cannot be solved classically even in nondeterministic *polynomial time, and (2) there is a decision problem that can be solved in exponential time on the quantum computer, which cannot be solved on a deterministic computer in less than double exponential time on* all but finitely many *instances.*

1 Review of earlier results

In a bold paper published in the *Proceedings of the Royal Society*, David Deutsch put forth in 1985 the idea that a *quantum computer* could in principle carry out a large amount of computation in parallel on a single piece of hardware by using the principle of quantum superposition [5, 6]. Later, he and Richard Jozsa exhibited a problem that the quantum computer could solve exponentially faster than any deterministic classical computer [7]. However, that problem admits more than one valid solution on most instances, and therefore it does not fit the usual mold in computational complexity theory of considering the difficulty of computing functions or of deciding set membership. (Nevertheless, we pointed out in [4] that their problem can be recast as a decision problem in the context of *promise problems* [9].)

* Supported in part by an NSERC postgraduate fellowship; currently visiting student at Merton College, Oxford University.

† Supported in part by NSERC's E. W. R. Steacie Memorial Fellowship and Québec' FCAR.

The main contribution of [4] was to interpret the result of Deutsch and Jozsa in the light of *oracle computations*, that is computations that can be performed with the help of arbitrarily complex oracles capable of instantly answering a precise set of questions. The direct oracle interpretation of their result is the existence of an oracle under which there is a decision problem that can be solved in linear time on the quantum computer, yet any deterministic classical computer would require exponential time on infinitely many instances to solve the same problem. In particular, this implies that $\mathbf{P}^X \subset \mathbf{QP}^X$ for some oracle X, where $\mathbf{P}$ denotes as usual the class of decision problems that can be solved with certainty in worst-case polynomial time by a classical deterministic computer, whereas $\mathbf{QP}$ denotes the class of decision problems that can be solved with certainty in worst-case polynomial time by the quantum computer, and the superscript X denotes the oracle whose availability we assume.

One might be tempted to think at first that the quantum computer gets its advantage over classical deterministic computers not from the use of quantum superposition, but merely from the much more mundane availability of randomization, which is inherent to quantum computation. After all, it is generally believed that randomized computers enjoy a computational advantage over classical computers, even if the randomized computer is required to have zero probability of yielding an erroneous result. To defuse this uninteresting interpretation, we also claim in [4] an oracle under which there is a decision problem that can be solved in linear time on the quantum computer, yet any probabilistic classical computer that is never allowed to make mistakes would require expected exponential time on infinitely many instances to solve the same problem. In symbols, $\mathbf{QP}^Y \not\subseteq \mathbf{ZPP}^Y$ for another appropriate oracle Y, where $\mathbf{ZPP}$ is the classical class of decision problems that can be solved in expected polynomial time by probabilistic computers that are not allowed to ever make mistakes [8].

2 New results

In this paper, we offer two new results along the lines of [4]. Under the same oracle Y that we had used to separate **QP** from **ZPP**, we show that the quantum computer can solve in linear time some decision problems that cannot be handled in polynomial time even by *nondeterministic* computers.[1] In symbols, $\mathbf{QP}^Y \not\subseteq \mathbf{NP}^Y$. The result is in fact stronger: we have a decision problem that is in $\mathbf{QP}^Y$ which is neither in $\mathbf{NP}^Y$ nor in $\mathbf{co}\text{–}\mathbf{NP}^Y$. (Recall that **NP** denotes the class of decision problems that can be solved in polynomial time by a nondeterministic computer; a problem is in **co–NP** if its complement is in **NP**.) Even better: any classical nondeterministic algorithm for our problem would require exponential time on infinitely many inputs. (We give below the construction of Y because it was omitted from [4].)

One unfortunate aspect of the results mentioned so far is that although classical computers (including probabilistic and nondeterministic) must take much more time than the quantum computer on these problems, this is merely true on infinitely many instances. This is not entirely satisfactory because it turns out that these problems are classically easy on the vast majority of instances. To get a stronger result, we need to consider harder problems (quantum exponential time), which are *much* harder classically. Specifically, we show the existence of yet another oracle under which there is a decision problem that can be solved in exponential time by the quantum computer so that any classical deterministic computer that solves it would have to run for *double* exponential time on *all but finitely many* inputs. It should be noted that simulating this use of the quantum computer with classical parallelism would require the availability of a double exponential number of processors since exponentially many processors running for exponential time can only perform an exponential amount of work.

3 Notation

Let Σ denote the binary alphabet $\{0,1\}$. As always, Σ^n denotes the set of n–bit strings and Σ^* denotes the set of all finite length bit strings. We use a standard bijection σ between Σ^* and the integers, such as $\sigma(x) = bin(1x) - 1$, where "$1x$" denotes string x with a "1" appended in front of it and $bin(y)$ is the integer whose binary representation is y. For instance, $\sigma(010) = bin(1010) - 1 = 9$. Let $\rho : \mathbf{N} \to \Sigma^*$ be the functional inverse of σ.

Consider a set $X \subseteq \Sigma^*$. We say that $B(X)$ holds if for all n either $X \cap \Sigma^n = \emptyset$ or there are exactly 2^{n-1} strings in $X \cap \Sigma^n$. The set S_X is defined as

$$S_X = \{1^n \mid X \cap \Sigma^n = \emptyset\}.$$

In other words, S_X contains only strings of ones, and $1^n \in S_X$ if and only if there are no strings of length n in X. (We had used the "unary complement" of this definition in [4].) When we need the same information in an exponentially more compact form, we use

$$V_X = \{x \in \Sigma^* \mid X \cap \Sigma^{\sigma(x)} = \emptyset\}.$$

The original result of Deutsch and Jozsa [7] implies that S_X can be recognized in linear time on the quantum computer if oracle X is available, provided $B(X)$ holds [4]. Therefore, V_X can be recognized in exponential time on the quantum computer under the same conditions since $x \in V_X$ if and only if $1^{\sigma(x)} \in S_X$.

Let $M_1, M_2, \ldots$ be a standard enumeration of all deterministic Turing machines and let $N_1, N_2, \ldots$ be a standard enumeration of all nondeterministic Turing machines. Finally, let $\alpha : \mathbf{N} \to \mathbf{N}$ be an arbitrary function such that for each integer i there are infinitely many integers n such that $\alpha(n) = i$ (for example, you could take one projection of the standard pairing function). Let $\wp : \mathbf{N} \to \mathbf{N}$ be defined recursively as $\wp(1) = 1$ and $\wp(n) = 2^{\wp(n-1)}$ for $n > 1$.

4 Quantum can beat nondeterminism

Theorem 1. *There exists an oracle relative to which there is a set that can be recognized in worst-case linear time by the quantum computer, yet any nondeterministic Turing machine that accepts it must take exponential time on infinitely many inputs.*

Proof. We shall construct the oracle in stages. At the end of stage n, it will be defined on all strings of length less than $\wp(n+1)$. Initially, set $Y_1 = \emptyset$. At stage $n \geq 1$, simulate every path of nondeterministic machine $N_{\alpha(n)}$ on input $1^{\wp(n)}$ with oracle Y_n for up to $2^{\wp(n)-1}$ steps for each path.

- If there is a path that makes the machine exceed $2^{\wp(n)-1}$ steps, or if the machine rejects, set $Y_{n+1} = Y_n$ and go to the next stage.

[1] It was Charles H. Bennett who pointed out to us that our original construction of oracle Y and set $S \in \mathbf{QP}^Y$ such that $S \notin \mathbf{ZPP}^Y$ also achieved $S \notin \mathbf{co}\text{–}\mathbf{NP}^Y$ unintentionally! Here, we modify slightly our previous construction in order to separate **QP** directly from **NP** rather than going through **co–NP**.

- If the machine accepts within $2^{\wp(n)-1}$ steps, select an arbitrary accepting path and let Q_n be the set of oracle questions asked along that path. Because of the time bound on the computation, note that Q_n contains no more than $2^{\wp(n)-1}$ questions, and none of these questions can be of length greater than $2^{\wp(n)-1}$. But there are $2^{\wp(n)}$ strings of length $\wp(n)$, and therefore we can form a set $R_n \subset \Sigma^{\wp(n)}$ such that $R_n \cap Q_n = \emptyset$ and the number of elements in R_n is exactly $2^{\wp(n)-1}$. Let $Y_{n+1} = Y_n \cup R_n$ and go to the next stage.

Let $Y = \bigcup_{n \geq 1} Y_n$. Note that $B(Y)$ holds because we took care at each step of either putting nothing in Y, or of putting in exactly half the strings of a given length (and we never put in strings of the same length more than once). Therefore, the quantum computer can accept S_Y in linear time given Y as oracle. Consider now an arbitrary nondeterministic Turing machine N_i that purports to accept language S_Y. By definition of α, there are infinitely many integers n such that $i = \alpha(n)$. For any such n we claim that N_i using oracle Y must take more than $2^{\wp(n)-1}$ steps on input $1^{\wp(n)}$. To prove this claim, consider what happens when we run N_i on input $1^{\wp(n)}$, but with oracle Y_n.

- If at least one path of the computation of $N_i^{Y_n}$ on $1^{\wp(n)}$ takes more than $2^{\wp(n)-1}$ steps, by construction $Y_{n+1} = Y_n$ and therefore Y and Y_n agree on all strings of length smaller than $\wp(n+1) = 2^{\wp(n)}$. But then the first $2^{\wp(n)}$ steps of any computation of N_i on input $1^{\wp(n)}$ are identical regardless of whether Y or Y_n is used as oracle since the machine cannot formulate questions long enough to distinguish between these oracles within that time. Therefore, using oracle Y as well, there is at least one path of the computation of N_i on $1^{\wp(n)}$ that takes more than $2^{\wp(n)-1}$ steps.

- If $N_i^{Y_n}$ rejects $1^{\wp(n)}$ within $2^{\wp(n)-1}$ steps, Y and Y_n agree on all strings of length smaller than $2^{\wp(n)}$ just as in the previous case. By the same argument, N_i behaves identically on input $1^{\wp(n)}$ regardless of whether it uses oracle Y or Y_n. Therefore N_i rejects $1^{\wp(n)}$ also with oracle Y. But N_i^Y is assumed to recognize S_Y, and hence $1^{\wp(n)} \notin S_Y$, which means that $Y \cap \Sigma^{\wp(n)} \neq \emptyset$. This is impossible since by construction no strings of length $\wp(n)$ are put into Y in this case.

- If $N_i^{Y_n}$ accepts $1^{\wp(n)}$ within $2^{\wp(n)-1}$ steps, consider the accepting path used in stage n of the construction of Y. The set R_n is designed so that it contains none of the oracle questions asked along this path. Since Y_n and Y differ on strings of length smaller than $2^{\wp(n)}$ only in that the strings in R_n belong to Y but not to Y_n, the same accepting path of N_i on $1^{\wp(n)}$ exists when oracle Y is used. Although N_i^Y and $N_i^{Y_n}$ could behave differently on all other paths, one accepting path is enough for a nondeterministic machine to accept, and therefore N_i^Y accepts $1^{\wp(n)}$. But again N_i^Y is assumed to recognize S_Y, and hence $1^{\wp(n)} \in S_Y$, which means that $Y \cap \Sigma^{\wp(n)} = \emptyset$. This is impossible since by construction $2^{\wp(n)-1}$ strings of length $\wp(n)$ are put into Y in this case.

Only the first case remains possible, and thus by Sherlock Holmes' principle it is the truth. But that case implies that N_i^Y on $1^{\wp(n)}$ takes more than $2^{\wp(n)-1}$ steps on at least one of its computation paths. Since this occurs for infinitely many values of n, any nondeterministic Turing machine that accepts S_Y with oracle Y must spend exponential time on infinitely many inputs. ∎

Corollary. *There exists an oracle Y such that*

$$\mathbf{QP}^Y \not\subseteq \mathbf{NP}^Y \cup \mathbf{co\text{–}NP}^Y.$$

Proof. Consider the oracle Y constructed in the proof of Theorem 1. We already know that $S_Y \in \mathbf{QP}^Y$. Since any nondeterministic Turing machine that accepts S_Y must take exponential time on infinitely many inputs, it follows that $S_Y \notin \mathbf{NP}^Y$. This proves that $\mathbf{QP}^Y \not\subseteq \mathbf{NP}^Y$. Because $\mathbf{QP}$ is closed under Boolean operations, it is clear that the complement of S_Y testifies to the fact that $\mathbf{QP}^Y \not\subseteq \mathbf{co\text{–}NP}^Y$. To prove the theorem, however, we need a single set that is in $\mathbf{QP}^Y$ but in neither $\mathbf{NP}^Y$ nor $\mathbf{co\text{–}NP}^Y$. We leave it for the reader to verify that

$$\tilde{S}_Y = S_Y \cup \{0^n \mid 1^n \notin S_Y\}$$

is such a set. ∎

5 Faster almost everywhere

The set S_Y constructed in the previous section is very sparse indeed: for any n it contains at most one string among all strings of length between n and $2^n - 1$. Moreover, deciding S_Y is easy on most inputs even without using oracle Y: any $x \in \Sigma^*$ that contains at least one 0 is automatically outside of S_Y.

We shall see in this section that, relative to an appropriate oracle, there is a set that is exponentially harder to solve deterministically than quantumly on all but finitely many instances.

Theorem 2. *There exists an oracle relative to which there is a set that can be recognized in worst-case exponential time by the quantum computer, yet any deterministic Turing machine that accepts it must take double exponential time on all but finitely many inputs.*

Proof. We shall construct an oracle Z such that $B(Z)$ holds and any deterministic Turing machine that recognizes V_Z using Z as oracle requires double exponential time on all but finitely many inputs. This construction is inspired by Manuel Blum's beautiful 1967 proof of his compression theorem. We shall construct oracle Z in stages. At the end of stage n, the oracle will be defined on all strings of length up to n. The key ingredient is that we keep track of a list of *cancelled* machines. Intuitively, whenever a machine is cancelled, we have made sure that it does not accept V_Z correctly given Z as oracle. Machines that have not yet been cancelled are called *live.*

Initially, none of the machines are cancelled and $Z_3 = \emptyset$ (for a technical reason we start at stage 3). At stage $n \geq 3$, run each of $M_1, M_2, \ldots, M_n$ on all the inputs $\rho(m)$ for $1 \leq m \leq n$, using oracle Z_n, for a maximum of $2^{n-1}/n^2$ steps on each run. (Recall that ρ is our bijection from the integers onto Σ^*.) Let Q_n stand for the set of all oracle questions asked during those runs. Note that Q_n contains at most $n \times n \times (2^{n-1}/n^2) = 2^{n-1}$ questions.

- If none of the live machines using oracle Z_n stop within $2^{n-1}/n^2$ steps on input $\rho(n)$, set $Z_{n+1} = Z_n$ and go to the next stage.
- Otherwise, let i be the smallest integer such that M_i is live and stops within $2^{n-1}/n^2$ steps on input $\rho(n)$ using oracle Z_n.
  - If $M_i^{Z_n}$ rejects $\rho(n)$, set $Z_{n+1} = Z_n$, cancel machine i, and go to the next stage.
  - Otherwise, let R_n be an arbitrary subset of Σ^n containing exactly 2^{n-1} elements such that $Q_n \cap R_n = \emptyset$ (which is possible since Q_n contains at most 2^{n-1} elements). Set $Z_{n+1} = Z_n \cup R_n$, cancel machine i, and go to the next stage.

Let $Z = \bigcup_{n \geq 3} Z_n$. Note that $B(Z)$ holds because we took care at each step n of either putting nothing in Z, or of putting in exactly half the strings of length n. Therefore, the quantum computer can accept S_Z in linear time, and thus V_Z in exponential time, given Z as oracle.

Consider now an arbitrary deterministic Turing machine M_i that purports to accept language V_Z when given Z as oracle. The first claim is that it is not possible for M_i ever to be cancelled. To show this, assume for a contradiction that it is cancelled at some stage n. In this case, the construction of Z makes sure that M_i using oracle Z_n rejects input $\rho(n)$ if and only if there are no strings of length n in Z, which means that $\rho(n) \in V_Z$. Therefore, M_i does not recognize V_Z when it uses oracle Z_n. To reach the desired contradiction, note that all the questions asked by M_i on input $\rho(n)$ under oracle Z_n find their way into Q_m for all $m \geq n$ and therefore the final oracle Z will not differ from Z_n on any of those questions, which implies that M_i does not recognize V_Z when it uses oracle Z either, contrary to our assumption. (We started at stage 3 to make sure that $2^{m-1}/m^2 \geq 2^{n-1}/n^2$ for all $m \geq n \geq 3$.)

Let n_0 be large enough that all machines of index smaller than i that will eventually be cancelled have been cancelled by stage n_0. Consider any $n > n_0$. We know that machine M_i cannot be cancelled at stage n. Since no machine of smaller index is cancelled at that stage either, it must be that M_i using oracle Z_n spends more than $2^{n-1}/n^2$ steps on input $\rho(n)$. By an argument similar to the one above, this behaviour applies also when the final oracle Z is used. But the length of $\rho(n)$ is logarithmic in n and therefore $2^{n-1}/n^2$ is doubly exponential in the length of $\rho(n)$. Because ρ is one-to-one and onto, we conclude that if M_i^Z accepts V_Z it must take at least double exponential time on all sufficiently large inputs. ∎

6 Leaping out of BPP

One important open question that was mentioned in [4] has not been addressed so far in this paper. Even though we have showed that, in appropriate relativized settings, **QP** includes decision problems lying outside of both **NP** and **co–NP**, and quantum exponential time includes problems that are *really* hard, our results are rather disappointing to anyone who believes that probabilistic algorithms that can make mistakes are essentially as satisfactory in practice as error-free deterministic or probabilistic algorithms, provided the probability of error can be brought down efficiently below any preset threshold. Indeed, all the problems we have considered can be solved just as efficiently by such bounded-error probabilistic algorithms as they can by the quantum computer. For instance, whenever $B(X)$ holds, $S_X \in \mathbf{BPP}$ and V_X can be solved in bounded-error probabilistic exponential time.[2] In other words, we were able to reach outside of **NP** with **QP**, but *not* outside of **BPP**. (Consult [8] for the classical definition of **BPP**.)

This is exactly what Ethan Bernstein and Umesh Vazirani have achieved very recently [10]. For this, they had to tolerate bounded-error probability from the quantum computer as well: they investigated **QBPP**. They have constructed an oracle relative to which $\mathbf{BPP} \subset \mathbf{QBPP}$ (strict inclusion). In fact, relative to this oracle, they have discovered a decision problem that is in **QBPP** but that is not even in **AM**, the powerful Arthur–Merlin class invented by László Babai, which generalizes simultaneously **NP** and **BPP** [1].

7 Conclusion and open questions

Many questions concerning quantum computing remain wide open. What can be proved about the power of the quantum computer in the real world (when oracles are not available)? Even proving that $\mathbf{P} \subset \mathbf{QP}$ (strict inclusion) is currently beyond hope because Bernstein and Vazirani have also proved that $\mathbf{QP} \subseteq \mathbf{PSPACE}$ (recall that $\mathbf{P} \not\subset \mathbf{PSPACE}$ is one of the major outstanding open questions of classical computational complexity theory). Nevertheless, a reasonable goal might be to prove that $\mathbf{P} \subset \mathbf{QP}$ or $\mathbf{BPP} \subset \mathbf{QBPP}$ under the assumption that one-way functions exist. Another approach concerns the $\mathbf{C}_{==}\mathbf{P}[\mathsf{half}]$ class introduced in [4].

An open question concerning oracles is to obtain almost-everywhere hardness for probabilistic or nondeterministic classical computers concerning problems exponentially easier for the quantum computer. (The techniques used in Theorems 1 and 2 are incompatible.) Lacking this, how about almost-everywhere hardness for deterministic computers concerning a problem in **QP**?

It is true that the quantum computer is beyond current technology, but this should not discourage research into quantum computing. Indeed, quantum physics has been used successfully for purposes closely related to computation in a prototype that demonstrates the technological feasibility of quantum cryptography [2, 3]. Ten years ago, quantum cryptography was still pure science-fiction!

[2] In fact, S_Y is even in **co–RP** but note that $\tilde{S}_Y$, even though it is in **BPP**, belongs to neither **RP** nor **co–RP**.

Acknowledgements

We are grateful to Charles H. Bennett for allowing us to use his observation that our previous construction [4] provided a relativized separation between **QP** and **co–NP**, and to Umesh Vazirani for allowing us to mention his latest results in quantum computing. The observation that an exponential number of processors working in parallel for exponential time would not suffice to simulate the quantum computer's calculation of V_Z is due to Lev Levitin.

References

[1] Babai, L. and S. Moran, "Arthur–Merlin games: A randomized proof system, and a hierarchy of complexity classes", *Journal of Computer and System Sciences*, Vol. 36, 1988, pp. 254 – 276.

[2] Bennett, C. H., F. Bessette, G. Brassard, L. Salvail and J. Smolin, "Experimental quantum cryptography", *Journal of Cryptology*, Vol. 5, no. 1, pp. 3 – 28.

[3] Bennett, C. H., G. Brassard and A. K. Ekert, "Quantum Cryptography", *Scientific American*, October 1992, pp. 50 – 57.

[4] Berthiaume, A. and G. Brassard, "The quantum challenge to structural complexity theory", *Proceedings of the 7th Annual IEEE Conference on Structure in Complexity Theory*, Boston, MA, June 1992, pp. 132 – 137.

[5] Deutsch, D., "Quantum theory, the Church–Turing principle and the universal quantum computer", *Proceedings of the Royal Society*, London, Vol. A400, 1985, pp. 97 – 117.

[6] Deutsch, D., in *Quantum Concepts in Space and Time* (R. Penrose and C. Isham, editors), Oxford: Clarendon Press, 1986, pp. 215 – 225.

[7] Deutsch, D. and R. Jozsa, "Rapid solution of problems by quantum computation", *Proceedings of the Royal Society* **A**, London, 1992, to appear.

[8] Gill, J., "Computational complexity of probabilistic Turing machines", *SIAM Journal on Computing*, Vol. 6, 1977, pp. 675 – 695.

[9] Grollmann, J. and A. L. Selman, "Complexity measures for public-key cryptosystems", *SIAM Journal on Computing*, Vol. 17, 1988, pp. 309 – 335.

[10] Vazirani, U., private communication, October 1992.