数理解析研究所講究録
1166 巻 2000 年 207-215

Quantum Oracles and Computational Complexity

HARUMICHI NISHIMURA (西村 治道), MASANAO OZAWA (小澤正直)
Graduate School of Human Informatics and School of Informatics and Sciences
Nagoya University, Nagoya 464-8601, Japan

Abstract

In this paper, we consider the following problem (the subroutine problem). If the operation performed by an oracle is efficiently computable, can a non-oracle quantum computer simulate a quantum computer with this oracle efficiently ? Bennett et al. and Aharonov et al. respectively solved the subroutine problem when their oracles evaluate deterministic and probabilistic functions. However, both of them considered only the case where machines enter the query mode deterministically. We extend the notion of an oracle of a quantum computer to a device for performing a unitary transformation, and solve the subroutine problem in the general setting where a query state and a nonquery state may superpose. Using this result, we show that **EQP** and **BQP** are robust in the general setting, which extends the robustness of **BQP** shown by Bennett et al. in the case where machines enter the query mode deterministically. Moreover, we prove the robustness of **ZQP** by improving the method of Bennett et al. employed for the above **BQP** result.

1 Introduction

In computational complexity theory, an oracle may be described informally as a device for evaluating some Boolean function f at unit cost per evaluation. This allows us to formulate questions such as, "If we add the power of computing f to a Turing machine, which functions could be efficiently computed by that Turing machine ?". Computer theorists have used relativizations by oracles as a powerful tool in studying complexity theory. In quantum computations, many researchers have investigated the power of quantum computations using a quantum Turing machine (QTM) with an oracle which computes a Boolean function. Berthiaume and Brassard [5] recast the promise problem of Deutsch and Jozsa [6] in complexity theoretic terms, constructing an oracle relative to which the QTM is exponentially more efficient than any deterministic Turing machine. Bernstein and Vazirani [4] subsequently constructed an oracle which produces a superpolynomial gap between the quantum and probabilistic Turing machines. This result was improved by Simon [11], who constructed an oracle which produces an exponential gap between the quantum and probabilistic Turing machines. Extending Simon's idea and using some new techniques, Shor [10] gave quantum polynomial time algorithms for factoring problems and discrete logarithms. On the other hand, Bennett et al. [2] showed that relative to an oracle chosen uniformly at random, with probability 1, **NP**-complete problems cannot solved by a QTM in polynomial time.

Lately, Aharonov et al. [1] considered a quantum circuit with an oracle that computes a probabilistic function instead of a Boolean function which is computed deterministically. We generalize these notions of oracles in terms of unitary operators. Our quantum oracle is defined as a device for implementing a unitary operator U at unit cost per evaluation.

This allows us to formulate questions such as, "If we add the power of computing U to a QTM, which functions could be efficiently computed by that QTM ?".

If a unitary transformation U is efficiently computable, an oracle QTM with U seems to have no more power than a non-oracle QTM. In fact, in the classical case, if a language L is efficiently computable, a non-oracle Turing machine can simulate an oracle Turing machine with L efficiently by substituting a machine computing L for a query to L. However, in the case of quantum computing, we need to consider a superposition in which some configurations are in the query mode but others are not. Moreover, quantum states with query strings of different lengths may superpose, even if each element of the superposition is in the query mode. In these case, if we merely substitute a QTM computing U for a query to U, quantum coherence will collapse. In this paper, we discuss this problem. The problem is stated as follows. "If a unitary transformation U is efficiently computable by a QTM, is there a QTM simulating a oracle QTM with U efficiently ?". This problem is called the *subroutine problem.* We will solve the subroutine problem positively. Bennett et al. [2] solved the subroutine problem when an oracle evaluates a deterministic function. Aharonov et al. [1] solved the subroutine problem for quantum circuits instead of QTMs when an oracle evaluates a probabilistic function. However, deterministic and probabilistic functions can be represented by unitary transformations and moreover both of them have considered only the case where machines enter the query mode deterministically. Thus our result can be considered to be a generalization of their results. We can also solve the subroutine problem by using the simulation of QTMs by quantum circuits, generalized quantum controls, and the simulation of quantum circuits by QTMs [7, 8, 12]. However, our method using a quantum analogue of a time constructible function is simple and it can reduce the polynomial slowdown as much as possible, comparing with the method using quantum circuits.

EQP, **BQP**, and **ZQP** are respectively the classes of languages which are recognized by polynomial time QTMs carrying out error-free, Monte Carlo, and Las Vegas algorithms. We investigate the robustness of these quantum complexity classes. Bennett et al. [2] showed the robustness of **BQP** in the case where machines enter the query mode deterministically. Using a solution for the subroutine problem and the method of the proof of Bennett et al., we can show easily that **EQP** and **BQP** are robust in the general case where a query state and a nonquery state may superpose. The method of Bennett et al. is as follows. Instead of querying an oracle, they runs a QTM M that outputs the same answer as the oracle's with high probability, copies the output, and reverses the computation of M. This procedure allows us to carry out a computation that given a query string y as input, outputs only y and the oracle's answer with high probability. Thus a query step of an oracle QTM can be substituted by a Monte Carlo non-oracle QTM. However, in the case of Las Vegas algorithms, if we use the same procedure, we cannot distinguish correct answers from wrong ones. Then the algorithm will not be Las Vegas. We prove the robustness of **ZQP** by keeping a witness to distinguish the case where a QTM queries an oracle correctly from the other case in addition to the method of [2].

This paper is organized as follows. In Section 2 we give definitions and basic theorems on QTMs. In Section 3 we introduce a stationary time constructible function, and solve the subroutine problem by using this function. This section also contains the rigorous formulation of oracle QTMs. In Section 4 we show that **EQP**, **BQP**, and **ZQP** are robust in general form, improving the method of Bennett et al. and using a solution of the

subroutine problem.

2 Preliminaries

A quantum Turing machine (QTM) M is a quantum system consisting of a processor, a bilateral infinite tape, and a head to read and write a symbol on the tape. The formal definition of a QTM as a mathematical structure is given as follows. An *processor configuration set* is a finite set with two specific elements denoted by q_0 and q_f, where q_0 represents the *initial processor configuration* and q_f represents the *final processor configuration.* A *symbol set* is a finite set of the cardinality at least two with a specific element denoted by B and called the *blank.* A *Turing frame* is a pair (Q, Σ) of a processor configuration set Q and a symbol set Σ. In what follows, let (Q, Σ) be a Turing frame. A *tape configuration* from a symbol set Σ is a function T from the set $\mathbf{Z}$ of integers to Σ such that $T(m) = B$ except for finitely many $m \in \mathbf{Z}$. The set of all the possible tape configurations is denoted by $\Sigma^{\#}$. The set $\Sigma^{\#}$ is a countable set. The *configuration space* of (Q, Σ) is the product set $\mathcal{C}(Q, \Sigma) = Q \times \Sigma^{\#} \times \mathbf{Z}$. A *configuration* of (Q, Σ) is an element $C = (q, T, \xi)$ of $\mathcal{C}(Q, \Sigma)$. Specifically, if $q = q_0$ and $\xi = 0$ then C is called an *initial configuration* of (Q, Σ), and if $q = q_f$ then C is called a *final configuration* of (Q, Σ). The *quantum state space* of (Q, Σ) is the Hilbert space $\mathcal{H}(Q, \Sigma)$ spanned by $\mathcal{C}(Q, \Sigma)$ with the canonical basis $\{|C\rangle | C \in \mathcal{C}(Q, \Sigma)\}$ called the *computational basis.* A *quantum transition function* for (Q, Σ) is a function from $Q \times \Sigma \times Q \times \Sigma \times \{-1, 0, 1\}$ into the complex number field $\mathbf{C}$. A *(single tape) prequantum Turing machine* is defined to be a triple $M = (Q, \Sigma, \delta)$ consisting of a Turing frame (Q, Σ) and a quantum transition function δ for (Q, Σ).

Let $M = (Q, \Sigma, \delta)$ be a prequantum Turing machine. An element of Q is called a *processor configuration* of M, the set Σ is called the *alphabet* of M, the function δ is called the *quantum transition function* of M, and a (initial or final) configuration of (Q, Σ) is called the *(initial or final) configuration* of M. A unit vector in $\mathcal{H}(Q, \Sigma)$ is called a *state* of M. The *evolution operator* of M is a linear operator M_δ on $\mathcal{H}(Q, \Sigma)$ such that

$$M_\delta |q, T, \xi\rangle = \sum_{p \in Q, \tau \in \Sigma, d \in \{-1,0,1\}} \delta(q, T(\xi), p, \tau, d) |p, T_\xi^\tau, \xi + d\rangle$$

for all $(q, T, \xi) \in \mathcal{C}(Q, \Sigma)$, where T_ξ^τ is a tape configuration defined by

$$T_\xi^\tau(m) = \tau \quad (m = \xi), \quad T(m) \quad (m \neq \xi).$$

The domain of M_δ is defined to be the set of all $|\psi\rangle$ such that

$$\sum_{C \in \mathcal{C}(Q,\Sigma)} |\langle C|\psi\rangle|^2 ||M_\delta |C\rangle||^2 < \infty.$$

A (single tape) prequantum Turing machine is said to be a *(single tape) quantum Turing machine (QTM)* if the evolution operator is unitary. We can also define multi-tape QTMs and multi-track QTMs similarly.

The following theorem proved in [9] characterizes the quantum transition functions that give rize to QTMs. If it is assumed that the head must move either to the right or to the left at each step, condition (c) of Theorem 2.1 is automatically satisfied. In this case, Theorem 2.1 is reduced to the result to Bernstein and Vazirani [4].

Theorem 2.1 *A prequantum Turing machine $M = (Q, \Sigma, \delta)$ is a QTM if and only if δ satisfies the following condition.*

(a) *For any $(q, \sigma) \in Q \times \Sigma$,*

$$\sum_{p \in Q, \tau \in \Sigma, d \in \{-1,0,1\}} |\delta(q, \sigma, p, \tau, d)|^2 = 1.$$

(b) *For any $(q, \sigma), (q', \sigma') \in Q \times \Sigma$ with $(q, \sigma) \neq (q', \sigma')$,*

$$\sum_{p \in Q, \tau \in \Sigma, d \in \{-1,0,1\}} \delta(q', \sigma', p, \tau, d)^* \delta(q, \sigma, p, \tau, d) = 0.$$

(c) *For any $(q, \sigma, \tau), (q', \sigma', \tau') \in Q \times \Sigma^2$,*

$$\sum_{p \in Q, d=0,1} \delta(q', \sigma', p, \tau', d-1)^* \delta(q, \sigma, p, \tau, d) = 0.$$

(d) *For any $(q, \sigma, \tau), (q', \sigma', \tau') \in Q \times \Sigma^2$,*

$$\sum_{p \in Q} \delta(q', \sigma', p, \tau', -1)^* \delta(q, \sigma, p, \tau, 1) = 0.$$

Let $M = (Q, \Sigma, \delta)$ be an m-track QTM. Then Σ can be factorized as $\Sigma = \Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_m$ and $T \in \Sigma^{\#}$ can be written in the form $(T^1, T^2, \ldots, T^m)$, where $T^i \in \Sigma_i^{\#}$ for $i = 1, \ldots, m$. T^i is called an i-th track string. In what follows, for a string $x = x_0 x_1 \cdots x_{k-1}$ of length k, we denote by $\mathrm{t}[x]$ a tape configuration such that $\mathrm{t}[x](i) = x_i \ \ (0 \leq i \leq k-1)$, B (otherwise). For any tape configuration T, we will write $T = (T^1, \ldots, T^j)$ if $T = (T^1, \ldots, T^j, \mathrm{t}[\varepsilon], \ldots, \mathrm{t}[\varepsilon])$ where ε denotes the empty string. We will abbreviate a configuration $(q, (T^1, \ldots, T^k, \mathrm{t}[\varepsilon], \ldots,), (0, \ldots, 0))$ as $(q, (T^1, \ldots, T^k), 0)$. Let $E(\hat{\xi} = j)$, $E(\hat{q} = p)$, $E(\hat{T} = T_0)$ and $E(\hat{T^i} = T_0)$ be respectively projections on $\mathrm{span}\{|q, T, j\rangle | q \in Q, \ T \in \Sigma^{\#}\}$, $\mathrm{span}\{|p, T, \xi\rangle | T \in \Sigma^{\#}, \ \xi \in \mathbf{Z}\}$, $\mathrm{span}\{|q, T_0, \xi\rangle | q \in Q, \ \xi \in \mathbf{Z}\}$ and $\mathrm{span}\{|q, T, \xi\rangle | q \in Q, \ T = (T^1, \cdots, T_0, \cdots, T^m) \in \Sigma^{\#}, \ \xi \in \mathbf{Z}\}$. A QTM $M = (Q, \Sigma, \delta)$ is said to be *stationary*, if given an initial configuration C, there exists some $t \in \mathbf{N}$ satisfying $||E(\hat{\xi} = 0) E(\hat{q} = q_f) M_\delta^t |C\rangle||^2 = 1$ and for all $s < t$ $||E(\hat{q} = q_f) M_\delta^s |C\rangle||^2 = 0$. The positive integer t is called the *computation time* of M for input state $|C\rangle$, and $M_\delta^t |C\rangle$ is called the *final state* of M for $|C\rangle$. Specifically, if $|C\rangle = |q_0, \mathrm{t}[x], 0\rangle$, the integer t is called the computation time on input x. A *polynomial time* QTM is a stationary QTM such that the computation time on every input is a polynomial in the length of the input.

3 Oracle Quantum Turing Machines

A *stationary time constructible (ST-constructible)* QTM of a function $f : \mathbf{N} \to \mathbf{N}$ is defined to be a stationary QTM such that if the initial state is $|q_0, \mathrm{t}[x], 0\rangle$, then the final state is $|q_f, \mathrm{t}[x], 0\rangle$ and that the computation time is $f(|x|)$, where $|x|$ denotes the length of x. A function $f : \mathbf{N} \to \mathbf{N}$ is said to be *stationary time constructible (ST-constructible)* if there exists a stationary time constructible QTM of f. An ST-constructible function has the following nice property.

Lemma 3.1 *For any k-dimensional polynomial p, there is an ST-constructible function f such that $p+f$ is an ST-constructible (and monotone increasing) k-dimensional polynomial.*

Bennett et al. [2] defined an oracle quantum Turing machine as the following special QTM. An oracle quantum Turing machine has a special tape called an *oracle tape* and its processor configuration set includes special elements q_q and q_a, which are respectively called the *prequery* processor configuration and the *postquery* processor configuration. For convenience, the processor enters q_q only when the head position of the oracle tape is zero. Given a language L called an *oracle language*, this machine evolves as follows.

(1) If the processor configuration is q_q and the string written on the oracle tape is $(x, b) \in \{0,1\}^* \times \{0,1\}$, the processor enters q_a while the content of the oracle tape changes to $(x, b \oplus L(x))$ deterministically in a single step, where " $\oplus$ " denotes the exclusive-or.

(2) If the processor configuration is not q_q, then M evolves according to the quantum transition function.

In this paper, we give a formal definition of more general oracle quantum Turing machines, which has an *oracle unitary transformation* instead of an oracle language.

Let Q be a processor configuration set including q_q and q_a, let Σ be a symbol set, let δ be a function from $(Q\backslash\{q_q\}) \times \Sigma \times (Q\backslash\{q_a\}) \times \Sigma \times \{-1,0,1\}$ to $\mathbf{C}$, and let U be a unitary transformation such that $U|x\rangle \in \mathrm{span}\{|z\rangle | z \in \{0,1\}^n\}$ for any $x \in \{0,1\}^n$. Then $M = (Q, \Sigma, \delta, U)$ is said to be an *oracle prequantum Turing machine (with U)*. The *evolution operator* of M is defined to be a linear operator U_M on $\mathcal{H}(Q, \Sigma)$ such that

$$U_M|q, T, \xi\rangle = \begin{cases} \sum_{p \in Q\backslash\{q_a\}, \tau \in \Sigma, d \in \{-1,0,1\}} \delta(q, T(\xi), p, \tau, d)|p, T^\tau_\xi, \xi + d\rangle & (q \neq q_q) \\ \sum_{y \in \{0,1\}^{|x|}} \langle y|U|x\rangle |q_a, \mathrm{t}[y], 0\rangle & (q = q_q, T = \mathrm{t}[x], \xi = 0) \\ |q_a, T, \xi\rangle \quad \text{(otherwise)}. \end{cases}$$

M is said to be an *oracle quantum Turing machine (oracle QTM)* if U_M is unitary. Then we can obtain the following necessary and sufficient conditions by the similar way to the proof of Theorem 2.1 [9].

Theorem 3.2 *An oracle prequantum Turing machine $M = (Q, \Sigma, \delta, U)$ is an oracle QTM if and only if the following quantum transition function δ' for $((Q \cup \{r\})\backslash\{q_q, q_a\}, \Sigma)$ satisfies conditions* (a)–(d) *of Theorem 2.1. Here r is an element that does not belong to Q.*

$$\delta'(q, \sigma, p, \tau, d) = \begin{cases} \delta(q_a, \sigma, q_q, \tau, d) & (q = p = r) \\ \delta(q_a, \sigma, q, \tau, d) & (q = r,\ p \neq r) \\ \delta(q, \sigma, q_q, \tau, d) & (p = r,\ q \neq r) \\ \delta(q, \sigma, p, \tau, d) & (q \neq r,\ p \neq r). \end{cases}$$

Similarly we can define a multi-tape oracle QTM. For example, if M is a k-tape oracle QTM and a state $|\psi\rangle$ of M is $|q_q, (T^1, \ldots, T^{k-1}, \mathrm{t}[x]), (d_1, \ldots, d_{k-1}, 0)\rangle$, the state $U_M|\psi\rangle$ is defined to be $\sum_{y \in \{0,1\}^{|x|}} \langle y|U|x\rangle |q_a, (T^1, \ldots, T^{k-1}, \mathrm{t}[y]), (d_1, \ldots, d_{k-1}, 0)\rangle$. Then the k-th tape is called an oracle tape. We can consider an oracle QTM with a language L, defined by Bennett et al., to be a multi-tape oracle QTM with the unitary transformation U_L such that $U_L|x, b\rangle = |x, b \oplus L(x)\rangle$ for all $(x, b) \in \{0,1\}^* \times \{0,1\}$. In what follows, we denote by M^U (and M^L) an arbitrary oracle QTM with a unitary transformation U (and a language L).

We introduce a notion of supersimulation necessary for a solution of the subroutine problem. For a stationary QTM, we can consider supersimulation as a special case of simulation defined in [8]. We denote by $\mathcal{D}(M,x)$ the set $\{C \in \mathcal{C}(Q,\Sigma) | \exists s \leq t\ [\langle C|M_\delta^s|q_0, \mathrm{t}[x], 0\rangle \neq 0]\}$, where t is the computation time of M on input x. Let $M = (Q,\Sigma,\delta)$ be a stationary QTM and $M' = (Q', \Sigma_1 \times \Sigma_2, \delta')$ be a single tape QTM such that $Q \times \Sigma \subseteq Q' \times \Sigma_1$. We say that *$M'$ supersimulates M with slowdown f*, if there exists a function $f : \mathbf{N} \to \mathbf{N}$ such that for any input x of M and $C \in \mathcal{D}(M,x)$ there exists some $T' \in \Sigma_2^{\#}$ (depending on x), and that

$$M_{\delta'}^{f(|x|)}|C\rangle|T'\rangle = \sum_{C' \in \mathcal{C}(Q,\Sigma)} \langle C'|M_\delta|C\rangle|C'\rangle|T'\rangle,$$

where $|C\rangle|T'\rangle$ denotes $|q,(T,T'),\xi\rangle$ for $C = (q,T,\xi)$. We can extend a similar notion to multi-tape QTMs and oracle QTMs. For any QTM $M = (Q,\Sigma,\delta)$ and any $r \in Q$, we can obtain the following oracle QTM $M^U = (Q',\Sigma,\delta',U)$ with $Q' = ((Q\backslash\{r\})\times\{0,1\})\cup\{q_q, q_a\}$.

$$\begin{aligned}
&\delta'((q,0),\sigma,(p,1),\tau,d) = \delta(q,\sigma,p,\tau,d),\\
&\delta'((q,1),\sigma,(q,0),\sigma,0) = 1,\\
&\delta'((q,0),\sigma,q_q,\tau,d) = \delta(q,\sigma,r,\tau,d),\\
&\delta'(q_a,\sigma,(p,1),\tau,d) = \delta(r,\sigma,p,\tau,d),\\
&\delta'(q_a,\sigma,q_q,\tau,d) = \delta(r,\sigma,r,\tau,d).
\end{aligned}$$

In particular, if U is the identity operator, then M^U supersimulates M. Thus, we can consider a QTM to be a special case of an oracle QTM.

We say that a unitary transformation U is *polynomial time computable* by a QTM M if the final state of M for the initial state $|q_0, \mathrm{t}[x], 0\rangle$ with $|x| = n$ is $\sum_{y\in\{0,1\}^n}\langle y|U|x\rangle|q_f, \mathrm{t}[y], 0\rangle$, and the computation time of M is a polynomial in n.

Using ST-constructible functions, we can solve the subroutine problem of QTMs.

Theorem 3.3 *If a unitary transformation U is polynomial time computable by a QTM M, there are a polynomial p and a polynomial time QTM M' such that M' supersimulates a polynomial time oracle QTM M^U with slowdown p.*

The degree of the polynomial slowdown in Theorem 3.3 is reduced as much as possible. It is same as the case of deterministic or probablistic Turing machines.

Corollary 3.4 *If a unitary transformation U is computable by a QTM M in linear time, there is a polynomial time QTM M' such that M' supersimulates a linear time oracle QTM M^U in quadratic time.*

We say that a unitary transformation U is *approximately polynomial time computable* by a QTM M if (1) $U|x\rangle \in \mathrm{span}\{|z\rangle | z \in \{0,1\}^n\}$ for any $x \in \{0,1\}^n$, (2) there is a family of unitary transformations $\{U_l'\}$ such that the final state of M for the initial state $|q_0, \mathrm{t}[x, 1^l], 0\rangle$ with $|x| = n$ is

$$\sum_{y\in\{0,1\}^*} \langle y|U_l'|x\rangle|q_f, \mathrm{t}[y, 1^l], 0\rangle$$

and $||U|x\rangle - U_l'|x\rangle|| \leq 1/2^l$ for any $x \in \{0,1\}^*$, and (3) the computation time of M is a polynomial in n and l.

Let $M = (Q, \Sigma, \delta)$ be a stationary QTM and $M' = (Q', \Sigma_1 \times \Sigma_2, \delta')$ be a single tape QTM such that $Q \times \Sigma \subseteq Q' \times \Sigma_1$. We say that M' *supersimulates* M *with bounded error and slowdown* f, if there exists a function $f : \mathbf{N}^2 \to \mathbf{N}$ such that for any input $y = (x, 1^l)$ of M, where $l \in \mathbf{N}$, and any $C \in \mathcal{D}(M, y)$ there exists some $T' \in \Sigma_2^{\#}$ (depending on y), and that $||M_{\delta'}^{f(|x|,l)}|C\rangle|T'\rangle - (M_\delta|C\rangle) \otimes |T'\rangle|| \leq 1/2^l$. The following theorem holds by the similar way to the proof of Theorem 3.3.

Theorem 3.5 *If a unitary transformation U is approximately polynomial time computable by a QTM M, for any $l \in \mathbf{N}$ there are a polynomial $p(n, l)$ and a polynomial time QTM M that supersimulates a polynomial time oracle QTM M^U with bounded error and slowdown p.*

4 The Robustness of Quantum Complexity Classes

We can apply Theorems 3.3 and 3.5 to the robustness of the quantum complexity classes **EQP**, **BQP**, and **ZQP**. We shall now define complexity classes for oracle QTMs. These definitions are natural extensions of complexity classes for QTMs [4, 8]. In what follows, the ranges of quantum transition functions are the polynomial time computable numbers.

We say that an oracle QTM M *accepts (rejects)* $x \in \{0,1\}^*$ *with probability* p if the final state $|\psi\rangle$ of M for the initial state $|q_0, \mathrm{t}[x], 0\rangle$ satisfies

$$||E(\hat{T}^1 = \mathrm{t}[x])E(\hat{T}^2 = \mathrm{t}[1])|\psi\rangle||^2 = p \quad (||E(\hat{T}^1 = \mathrm{t}[x])E(\hat{T}^2 = \mathrm{t}[0])|\psi\rangle||^2 = p).$$

We say that M *recognizes* a language L with probability p if M accepts x with probability of at least p for any $x \in L$ and rejects x with probability of at least p for any $x \notin L$. Moreover, we say that M recognizes L *with probability uniformly larger than* p, if there is a constant $0 < \eta \leq 1 - p$ such that M recognizes L with probability $p + \eta$. A language L' is in $\mathbf{BQP}^L$ ($\mathbf{EQP}^L$) if there is a polynomial time oracle QTM $M^L = (Q, \Sigma, \delta, U_L)$ that recognizes L' with probability uniformly larger than $\frac{1}{2}$ (with probability 1). Then the QTM M^L is called **BQP** type. A language L' is in $\mathbf{ZQP}^L$ if there is a polynomial time QTM $M^L = (Q, \Sigma, \delta, U_L)$ satisfying the following conditions: (1) M^L recognizes L with probability uniformly larger than $\frac{1}{2}$, (2) If M accepts (rejects) x with a positive probability, M rejects (accepts) x with probability 0. Such a QTM M^L is called **ZQP** type. For classes $\mathcal{C}$ and $\mathcal{D}$ of languages, let $\mathcal{C}^{\mathcal{D}} = \bigcup_{L \in \mathcal{D}} \mathcal{C}^L$. If $\mathcal{C}^{\mathcal{C}} = \mathcal{C}$, the class $\mathcal{C}$ is said to be *robust for subroutines.*

If L is in **EQP**, then we can construct a polynomial time oracle QTM such that only the input x and the answer $L(x)$ are written on the tape of the final state with probability 1 by the method of [3] (concretely, by using the reversal lemma of [4]). Thus, **EQP** is robust for subroutines by Theorem 3.3.

Corollary 4.1 $\mathbf{EQP}^{\mathbf{EQP}} = \mathbf{EQP}$.

Bennett et al. [2] showed the following theorem to use a Monte Carlo quantum algorithm as a subroutine of another quantum algorithm.

Theorem 4.2 *If a language L is in* **BQP**, *for any $\varepsilon > 0$ there is a QTM M which recognizes L with probability $1 - \varepsilon$ and has the following property. The computation time of M for $|q_0, \mathrm{t}[x], 0\rangle$ is a polynomial in $|x|$ and $\log 1/\varepsilon$, and the final state is $\alpha|q_f, T, 0\rangle + |\psi\rangle$, where $|\alpha|^2 \geq 1 - \varepsilon$ and $T = (\mathrm{t}[x], \mathrm{t}[L(x)])$.*

Theorem 4.2 guarantees that without loss of generality a **BQP** type QTM recognizing L has a clean tape with only the input x and the answer $L(x)$ with arbitrary large probability after the computation. In other words we can assume that a **BQP** type QTM has only one accepting configuration. Bennett et al. [2] insisted that **BQP** is robust for subroutines as the corollary of Theorem 4.2, since this theorem allows us to use a QTM recognizing an oracle language instead of the oracle itself. However, they considered the case where machines enter the query mode deterministically, and thus they did not discuss the possibility that the coherence of different computation paths collapses by the insertion of a QTM recognizing an oracle language. We have already solved this problem by Theorem 3.5, so that we can show that **BQP** is robust for subroutines in the general setting where a query state and a nonquery state may superpose.

Corollary 4.3 $\mathbf{BQP}^{\mathbf{BQP}} = \mathbf{BQP}$.

Now we consider the robustness of **ZQP**. If we apply Theorem 4.2 to a language in **ZQP**, the obtained algorithm will not be Las Vegas. Thus we need the following theorem, which means that we can also assume that a **ZQP** type QTM has only one accepting configuration.

Theorem 4.4 *If a language L is in* **ZQP**, *for any $\varepsilon > 0$ there is a QTM M which recognizes L with probability $1-\varepsilon$ and has the following property. The computation time of M for $|q_0, \mathrm{t}[x], 0\rangle$ is a polynomial in $|x|$ and $\log 1/\varepsilon$, and the final state is $\alpha|q_f, T, 0\rangle + |\psi\rangle$, where $|\alpha|^2 \geq 1-\varepsilon$, $T = (\mathrm{t}[x], \mathrm{t}[L(x)])$, and $E(\hat{q} = q_f)E(\hat{T}^2 = \mathrm{t}[\star])|\psi\rangle = |\psi\rangle$. Here, we denote by $\star$ a special symbol of the second track of M.*

Let $M = (Q, \Sigma, \delta)$ be a stationary QTM and $M' = (Q', \Sigma_1 \times \Sigma_2, \delta')$ be a single tape QTM such that $Q \times \Sigma \subseteq Q' \times \Sigma_1$. We say that M' *supersimulates M with zero error and slowdown f*, if there exists a function $f : \mathbf{N}^2 \to \mathbf{N}$ such that for any input $y = (x, 1^l)$ of M, where $l \in \mathbf{N}$, and any $C \in \mathcal{D}(M, y)$ there exists some $T' \in \Sigma_2^{\#}$ (depending on y) satisfying the following condition: If $M_{\delta'}^{f(|x|,l)}|C\rangle|T'\rangle = |\phi\rangle + |\psi_1\rangle$ and $E(\hat{T}^2 = \mathrm{t}[\star])(|\phi\rangle + |\psi_1\rangle) = |\psi_1\rangle$, then $(M_\delta|C\rangle) \otimes |T'\rangle = |\phi\rangle + |\psi_2\rangle$, $|||\psi_1\rangle - |\psi_2\rangle|| \leq 1/2^l$, and $E(\hat{T}^2 = \mathrm{t}[\star])|\psi_2\rangle = 0$. Using Theorem 4.4 we can show the following lemma by the similar way to the proof of Theorem 3.3.

Lemma 4.5 *If L is in* **ZQP**, *there are a polynomial $p(n, l)$ and a polynomial time QTM M that supersimulates a polynomial time oracle QTM M^L with U_L with zero error and slowdown p.*

We can show that **ZQP** is robust for subroutines by using Lemma 4.5. Then we need to construct our algorithm so that we cannot erase the symbol $\star$ written as a witness of an error in the subsequent steps.

Theorem 4.6 $\mathbf{ZQP}^{\mathbf{ZQP}} = \mathbf{ZQP}$.

References

[1] D. Aharonov, A. Kitaev, and N. Nisan, *Quantum circuits with mixed states*, in: Proceedings of the 31th Annual ACM Symposium on Theory of Computing (1998) 20-30.

[2] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, *Strengths and weaknesses of quantum computing*, SIAM J. Comput. **26** (1997) 1510-1523.

[3] C. H. Bennett, *Logical reversibility of computation*, IBM J. Res. Develop. **17** (1973) 525-532.

[4] E. Bernstein and U. Vazirani, *Quantum complexity theory*, in: Proceedings of the 25th Annual ACM Symposium on Theory of Computing (1993) 11-20. Journal version appeared in SIAM J. Comput. **26** (1997) 1411-1473.

[5] A. Berthiaume and G. Brassard, *The quantum challenge to structural complexity theory*, in: Proceeding of the 7th Annual Structure in Complexity Theory Conference (1992) 132-137.

[6] D. Deutsch and R. Jozsa, *Rapid solution of problems by quantum computation*, Proc. Roy. Soc. London Ser. A **439** (1992) 553-558.

[7] A. Kitaev, *Quantum computations: algorithms and error correction*, Russian Math. Surveys **52** (1997) 1191-1249.

[8] H. Nishimura and M. Ozawa, *Computational complexity of uniform quantum circuit families and quantum Turing machines*, preprint available from the LANL quant-ph archive 9906095.

[9] M. Ozawa and H. Nishimura, *Local transition functions of quantum Turing machines*, preprint available from the LANL quant-ph archive 9811069.

[10] P. W. Shor, *Algorithms for quantum computations: Discrete log and factoring*, in: Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science (1994) 124-134. Journal version, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, appeared in SIAM J. Comput. **26** (1997) 1484-1509.

[11] D. Simon, *On the power of quantum computation*, in: Proceeding of the 35th Annual IEEE Symposium on Foundations of Computer Science (1994) 116-123. Journal version appeared in SIAM J. Comput. **26**, 1474-1483 (1997).

[12] A. Yao, *Quantum circuit complexity*, in: Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science (1993) 352-361.