ELSEVIER

Theoretical Computer Science 276 (2002) 147–181

Theoretical Computer Science

www.elsevier.com/locate/tcs

Computational complexity of uniform quantum circuit families and quantum Turing machines

Harumichi Nishimura, Masanao Ozawa *

Graduate School of Human Informatics, Nagoya University, Chikusa-ku, Nagoya 464-8601, Japan

Received February 1999; received in revised form November 2000; accepted January 2001
Communicated by O. Watanabe

Abstract

Deutsch proposed two sorts of models of quantum computers, quantum Turing machines (QTMs) and quantum circuit families (QCFs). In this paper we explore the computational powers of these models and re-examine the claim of the computational equivalence of these models often made in the literature without detailed investigations. For this purpose, we formulate the notion of the codes of QCFs and the uniformity of QCFs by the computability of the codes. Various complexity classes are introduced for QTMs and QCFs according to constraints on the error probability of algorithms or transition amplitudes. Their interrelations are examined in detail. For Monte Carlo algorithms, it is proved that the complexity classes based on uniform QCFs are identical with the corresponding classes based on QTMs. However, for Las Vegas algorithms, it is still open whether the two models are equivalent. We indicate the possibility that they are not equivalent. In addition, we give a complete proof of the existence of a universal QTM efficiently simulating multi-tape QTMs. We also examine the simulation of various types of QTMs such as multi-tape QTMs, single tape QTMs, stationary, normal form QTMs (SNQTMs), and QTMs with the binary tapes. As a result, we show that these QTMs are computationally equivalent to one another as computing models implementing not only Monte Carlo algorithms but also exact (or error-free) ones.

Keywords: Quantum computation; Complexity theory; Quantum Turing machines; Uniform quantum circuit families; Universal quantum Turing machines

1. Introduction

In the early 1980s, Feynman [14] suggested that computers based on quantum mechanics would carry out computations more efficiently than classical ones, and Benioff

* Corresponding author.
E-mail address: ozawa@mailaps.org (M. Ozawa).

0304-3975/02/$ - see front matter
PII: S0304-3975(01)00111-6

[4] started the study of quantum mechanical Hamiltonian models of Turing machines. In the late 1980s, Deutsch introduced quantum Turing machines (QTMs) [10] and quantum circuits [11] as models of quantum computers. Using Deutsch's models, several results were obtained to suggest that quantum computers are more powerful than classical ones [9, 7, 12, 21]. Eventually, Shor [19] proposed efficient quantum algorithms for the factoring problem and the discrete logarithm problem, which are considered to have no efficient algorithms in computational complexity theory and applied to public-key cryptosystems. Since then, many experiments have been attempted to realize a quantum computer.

Up to now, the above two models appear to have been studied under different objectives. A QTM models a programmable computing machine and has been used as a mathematical model for studying the efficiency of quantum computation. On the other hand, a quantum circuit has been studied mainly as a physical model for realization. Thus, in order to make a bridge between these two approaches, it is important to give a detailed comparison of their computational powers from a complexity theoretical point of view.

The existence of a universal QTM was shown first by Deutsch [10]. However, his universal QTM needs exponential slowdown for simulating QTMs. In 1993, Bernstein and Vazirani [7] claimed that there is an efficient universal QTM, and gave a detailed proof in [8]. But their universal QTM is applicable only to QTMs such that the head must move either to the right or to the left at each step (two-way QTMs), and their method cannot afford an efficient simulation of a general QTM the head of which is not required to move. Shortly after, Yao [24] claimed the existence of a universal QTM simulating general QTMs, with the following sketch of the proof: He first shows that there is a quantum circuit simulating a given QTM for arbitrary steps, and his universal QTM is designed to carry out each step of the computing of the quantum circuit. This sketch also implicitly contains the existence of a QTM that simulates any quantum circuit. From the above argument, it is often claimed in the literature that quantum circuits and QTMs are computationally equivalent. However, from the computational complexity theoretical point of view, the following points are left for further investigations.

In the first place, Yao did not define the uniformity of quantum circuit families (QCFs). Since a single quantum circuit has a constant input length, we need to consider families of quantum circuits for comparing the computational power of quantum circuits with QTMs. From the viewpoint of polynomial complexity, it is well known that Boolean circuit families with arbitrary input length should satisfy a uniformity condition, as long as they are computationally no more powerful than Turing machines. The uniformity of QCFs was mentioned briefly by Ekert-Jozsa [13] and Shor [20]. As pointed out by Shor, we need to introduce a definition of uniformity quite different from Boolean circuit families, because each wire has continuously many different states rather than only two in Boolean circuits. Secondly, the complexity classes of QCFs have not been defined explicitly. Shor [20] claimed that QTMs and QCFs are equivalent as probabilistic computing models implementing Monte Carlo algorithms, but the

proof has not been given. Moreover, it has not been discussed yet whether two models are equivalent as probabilistic machines implementing Las Vegas algorithms or exact algorithms (algorithms which always produce correct answers). In order to study these problems, we should set up various complexity classes for QTMs and QCFs according to constraints on the algorithms.

In this paper, we shall introduce the rigorous formulation of uniformity of QCFs and investigate the detailed relationship among complexity classes of QTMs and uniform QCFs. We introduce the class **BUPQC** of languages that are efficiently recognized by Monte Carlo type uniform QCFs, and show that **BUPQC** coincides with the class **BQP** of languages that are efficiently recognized by Monte Carlo type QTMs. On the other hand, we show that the class **ZQP** of languages that are efficiently recognized by Las Vegas type QTMs is included in the class **ZUPQC** that are efficiently recognized by Las Vegas type uniform QCFs. However, it still remains open whether these models are equivalent as computing models implementing Las Vegas algorithms. Moreover, we indicate the possibility that the inclusion is proper.

In addition, we discuss the relationship among various types of QTMs, in particular, single tape QTMs and multi-tape QTMs. In the classical case, it is possible to simulate a multi-tape Turing machine by a single tape Turing machine with quadratic polynomial slowdown. Multi-tape QTMs are indispensable to examine the $o(n)$-space bounded complexity or count the number of steps of a QTM. Thus, it is important to investigate the level of the computational equivalence of single tape QTMs and multi-tape QTMs.

We generalize Yao's construction of quantum circuits simulating single tape QTMs to multi-tape QTMs and give a complete proof of the existence of a single tape universal QTM efficiently simulating multi-tape QTMs. This shows that a multi-tape QTM can be simulated with arbitrary accuracy by a single tape QTM with polynomial slowdown. We also examine the simulation of various types of QTMs such as multi-tape QTMs, single tape QTMs, stationary, normal form QTMs (SNQTMs), and QTMs with the binary tapes. As a result, we show that these QTMs are computationally equivalent to one another as computing models implementing not only Monte Carlo algorithms but also exact ones.

This paper is organized as follows. In Section 2 we give definitions on QTMs and explain related notions. In Section 3, we adapt some of the basic lemmas on QTMs given by Bernstein and Vazirani [8] to the present approach. Moreover, we show that QTMs with the binary tapes can simulate two-way QTMs without error. In Section 4 we show that there is a universal QTM simulating multi-tape QTMs. This section also contains the rigorous formulation of quantum circuits. In Section 5 we formulate the uniformity of QCFs and introduce various classes of languages recognized by QTMs and uniform QCFs. We also show that QTMs and uniform QCFs are equivalent as probabilistic computing models implementing Monte Carlo algorithms, we indicate the possibility that these two models are not computationally equivalent as computing models implementing Las Vegas algorithms, and we show that SNQTMs are equivalent to multi-tape QTMs as computing models implementing exact algorithms.

2. Quantum Turing machines

In what follows, for any integers $n<m$ the interval $\{n,n+1,\ldots,m-1,m\}$ is denoted by $[n,m]_{\mathbf{Z}}$. 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. We refer to Deutsch [10] for the physical formulation of a QTM. The formal definition of a QTM as a mathematical structure is given as follows. A *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 cardinality at least 2 with a specific element denoted by B and called the *blank*. 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. For any $T\in\Sigma^{\#}$, $\tau\in\Sigma$, and $\xi\in\mathbf{Z}$, the tape configuration T_{ξ}^{τ} is defined by

$$T_{\xi}^{\tau}(m)=\begin{cases}\tau & \text{if } m=\xi,\\ T(m) & \text{if } m\neq\xi.\end{cases}$$

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. The *configuration space* of (Q,Σ) is the product set $\mathscr{C}(Q,\Sigma)=Q\times\Sigma^{\#}\times\mathbf{Z}$. A *configuration* of (Q,Σ) is an element $C=(q,T,\xi)$ of $\mathscr{C}(Q,\Sigma)$. 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 $\mathscr{H}(Q,\Sigma)$ spanned by $\mathscr{C}(Q,\Sigma)$ with the canonical basis $\{|C\rangle|\mathbf{C}\in\mathscr{C}(Q,\Sigma)\}$ called the *computational basis*. A *quantum transition function* for (Q,Σ) is a function from $Q\times\Sigma\times Q\times\Sigma\times[-1,1]_{\mathbf{Z}}$ 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 an (initial or final) configuration of (Q,Σ) is called an (*initial or final*) *configuration* of M. A unit vector in $\mathscr{H}(Q,\Sigma)$ is called a *state* of M. The *evolution operator* of M is a linear operator M_{δ} on $\mathscr{H}(Q,\Sigma)$ such that

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

for all $(q,T,\xi)\in\mathscr{C}(Q,\Sigma)$. The above equation uniquely defines the bounded operator M_{δ} on the space $\mathscr{H}(Q,\Sigma)$ [17].

A (single tape) prequantum Turing machine is said to be a (*single tape*) *quantum Turing machine* (*QTM*) if the evolution operator is unitary.

A quantum transition function δ for (Q,Σ) is said to be *two-way* if $\delta(p,\sigma,q,\tau,0)=0$ for any $(p,\sigma,q,\tau)\in(Q\times\Sigma)^2$. A prequantum Turing machine (or QTM) $M=(Q,\Sigma,\delta)$ is said to be two-way if δ is two-way (In [8], two-way QTMs are merely called QTMs, and QTMs in this paper are called general QTMs.)

The following theorem proved in [17] characterizes the quantum transition functions that give rise to QTMs. The quantum transition function of a two-way QTM satisfies condition (c) of Theorem 2.1 automatically. In this case, Theorem 2.1 is reduced to the result due to Bernstein and Vazirani [7, 8].

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

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

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

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

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

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

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

Ko and Friedman [16] introduced the notion of efficiently computable numbers. A real number x is *polynomial-time computable* if there is a polynomial-time computable function ϕ such that $|\phi(1^n)-x|\leqslant 2^{-n}$ and $\phi(1^n)\in\{m/2^n\mid m\in\mathbf{Z}\}$ for any $n\in\mathbf{N}$. We denote by P$\mathbf{R}$ the set of polynomial-time computable real numbers and let P$\mathbf{C}=\{x+y\sqrt{-1}\mid x,y\in\mathrm{P}\mathbf{R}\}$. We say that a QTM $M=(Q,\Sigma,\delta)$ is *in* P$\mathbf{C}$ if the range of δ is included in P$\mathbf{C}$. In this paper, we define a QTM to be with amplitudes in $\mathbf{C}$, since in Section 5 we investigate QTMs with amplitudes in $\mathbf{C}$ as a mathematical object. However, from the complexity theoretical point of view, we need to require that QTMs are in P$\mathbf{C}$ as defined by Bernstein and Vazirani [8]. When we consider a universal QTM in Section 4, we also restrict the QTMs given as the input of the universal QTM to QTMs in P$\mathbf{C}$, since not every QTM can be (efficiently) encoded with absolute accuracy by classical means. We now define the *code* $c(x)$ *of an element* x *in* P$\mathbf{R}$ by the code of a polynomial-time bounded deterministic Turing machine computing one of its rational approximations, and define the code of an element $z=x+y\sqrt{-1}$ in P$\mathbf{C}$ by $c(z)=\langle c(x),c(y)\rangle$. Then the QTMs in P$\mathbf{C}$ can be easily encoded: we define the *code*

of a QTM $M=(Q,\Sigma,\delta)$ in P**C** to be the list of the codes of elements $\delta(q,\sigma,p,\tau,d)$ in P**C**, where $(q,\sigma,p,\tau,d)\in Q\times\Sigma\times Q\times\Sigma\times[-1,1]_{\mathbf{Z}}$.

A finite string from a symbol set Σ is called a Σ-string. The length of a Σ-string x is denoted by $|x|$ and the set of all the possible Σ-strings is denoted by Σ^*. A tape configuration T from Σ is said to *represent* a Σ-string $x=\sigma_0\cdots\sigma_{k-1}$ of length k, if T satisfies

$$T(m)=\begin{cases}\sigma_m & \text{if } m\in[0,k-1]_{\mathbf{Z}},\\ B & \text{otherwise.}\end{cases}$$

In what follows, we denote by tape$[x]$ the tape configuration representing x.

For symbol sets $\Sigma_1,\ldots,\Sigma_k$ with the blanks $B_1,\ldots,B_k$, the product set $\Sigma=\Sigma_1\times\cdots\times\Sigma_k$ can be considered as a symbol set with the blank $B=(B_1,\ldots,B_k)$. The projection from Σ to Σ_i is denoted by π_i. If $s_i=\sigma_{1i}\cdots\sigma_{ni}$ is a Σ_i-string of length n for $i=1,\ldots,k$, the Σ-string $(\sigma_{11},\ldots,\sigma_{1k})\cdots(\sigma_{n1},\ldots,\sigma_{nk})$ is also denoted by $(s_1,\ldots,s_k)$. A *k-track* QTM is such that the alphabet Σ is factorized as $\Sigma=\Sigma_1\times\cdots\times\Sigma_k$ with symbol sets $\Sigma_1,\ldots,\Sigma_k$. The symbol set Σ_i is called the ith track alphabet of this QTM. If the tape configuration is T, the ith track configuration is defined as the function $T^i=\pi_iT\in\Sigma_i^{\#}$, so that we have $T(m)=(T^1(m),\ldots,T^k(m))$ for any $m\in\mathbf{Z}$. For $i=1,\ldots,j$, let s_i be a Σ_i-string of length at most n and $s_iB^{n_i}$ be the Σ_i-string $s_iBB\cdots B$ of length n. Then, tape$[s_1,\ldots,s_j]$ abbreviates tape$[(s_1B^{n_1},\ldots,s_jB^{n_j},\underbrace{B^n,\ldots,B^n}_{k-j})]$.

Let a symbol set Σ be decomposed as $\Sigma=\Sigma_1\times\cdots\times\Sigma_k$. The quantum state space $\mathscr{H}(Q,\Sigma)$ can be factorized as $\mathscr{H}(Q,\Sigma)=\mathscr{H}(Q)\otimes\mathscr{H}(\Sigma^{\#})\otimes\mathscr{H}(\mathbf{Z})$ or $\mathscr{H}(Q,\Sigma)=\mathscr{H}(Q)\otimes\mathscr{H}(\Sigma_1^{\#})\otimes\cdots\otimes\mathscr{H}(\Sigma_k^{\#})\otimes\mathscr{H}(\mathbf{Z})$, where $\mathscr{H}(Q)$, $\mathscr{H}(\Sigma^{\#})$, $\mathscr{H}(\Sigma_i^{\#})$, and $\mathscr{H}(\mathbf{Z})$ are the Hilbert spaces generated by Q, $\Sigma^{\#}$, $\Sigma_i^{\#}$ and $\mathbf{Z}$, respectively. Then, the computational basis state $|q,T,\xi\rangle$ can be represented as $|q,T,\xi\rangle=|q\rangle|T\rangle|\xi\rangle$ or $|q,T,\xi\rangle=|q\rangle|T^1\rangle\cdots|T^k\rangle|\xi\rangle$ by the canonical bases $\{|q\rangle\mid q\in Q\}$ of $\mathscr{H}(Q)$, $\{|T\rangle\mid T\in\Sigma^{\#}\}$ of $\mathscr{H}(\Sigma^{\#})$, $\{|T^i\rangle\mid T^i\in\Sigma_i^{\#}\}$ of $\mathscr{H}(\Sigma_i^{\#})$, and $\{|\xi\rangle\mid\xi\in\mathbf{Z}\}$ of $\mathscr{H}(\mathbf{Z})$.

Let $M=(Q,\Sigma,\delta)$ be a QTM, and we assume the numbering of Q and Σ such that $Q=\{q_0,\ldots,q_{|Q|-1}\}$ and $\Sigma=\{\sigma_0,\ldots,\sigma_{|\Sigma|-1}\}$, where we denote by $|X|$ the cardinality of a set X. We define projections $E^{\hat{q}}(q_j)$, $E^{\hat{T}(m)}(\sigma_j)$ for $m\in\mathbf{Z}$, and $E^{\hat{\xi}}(\xi)$ for $\xi\in\mathbf{Z}$ by

$$\begin{aligned}E^{\hat{q}}(q_j)&=|q_j\rangle\langle q_j|\otimes I_2\otimes I_3,\\ E^{\hat{T}(m)}(\sigma_j)&=\sum_{T(m)=\sigma_j}I_1\otimes|T\rangle\langle T|\otimes I_3,\\ E^{\hat{\xi}}(\xi)&=I_1\otimes I_2\otimes|\xi\rangle\langle\xi|,\end{aligned}$$

where I_1,I_2, and I_3 are the identity operators on $\mathscr{H}(Q)$, $\mathscr{H}(\Sigma^{\#})$, and $\mathscr{H}(\mathbf{Z})$, respectively. Moreover, if M is a k-track QTM with alphabet $\Sigma=\Sigma_1\times\cdots\times\Sigma_k$, we define a projection $E^{\hat{T}^i}(T^i)$ for $T^i\in\Sigma_i^{\#}$ where $i=1,\ldots,k$ by

$$E^{\hat{T}^i}(T^i)=I_1\otimes I_{2,1}\otimes\cdots\otimes I_{2,i-1}\otimes|T^i\rangle\langle T^i|\otimes I_{2,i+1}\otimes\cdots\otimes I_{2,k}\otimes I_3,$$

where $I_{2,j}$ is the identity operator on $\mathscr{H}(\Sigma_j^{\#})$.

A QTM $M=(Q,\Sigma,\delta)$ is said to be *stationary* [8, Definition 3.12], if for every initial configuration C, there exists some $t\in\mathbf{N}$ such that $\|E^{\hat{\xi}}(0)E^{\hat{q}}(q_f)M_\delta^t|C\rangle\|^2=1$ and for all $s<t$ we have $\|E^{\hat{q}}(q_f)M_\delta^s|C\rangle\|^2=0$. The positive integer t is called the *computation time* of M for input state $|C\rangle$. Specifically, if $|C\rangle=|q_0,\mathrm{tape}[x],0\rangle$, it is called the computation time of M on input x. A *polynomial-time bounded* QTM is a stationary QTM such that on every input x the computation time is bounded by a polynomial in the length of x. Moreover, let $|\phi\rangle=\sum_{x\in\Sigma^n}\alpha_x|q_0,\mathrm{tape}[x],0\rangle$ for some $n\in\mathbf{N}$. Then, if the computation time of M on every input x satisfying $\alpha_x\neq 0$ is t, the state $M_\delta^t|\phi\rangle$ is called the *output state* of M for *input state* $|\phi\rangle$. A QTM $M=(Q,\Sigma,\delta)$ is said to be in *normal form* [8, Definition 3.13], if $\delta(q_f,\sigma,q_0,\sigma,1)=1$ for any $\sigma\in\Sigma$. In what follows "SNQTM" abbreviates "stationary, normal form QTM". We may consider only SNQTMs, without loss of generality, to develop quantum complexity theory as shown later (Theorem 5.8).

Finally, we shall give a formal definition of simulation. Let $M=(Q,\Sigma,\delta)$ and $M'=(Q',\Sigma',\delta')$ be QTMs. Let t be a positive integer and $\varepsilon>0$. Let $e:\mathscr{C}(Q,\Sigma)\to\mathscr{C}(Q',\Sigma')$ be an injection computable in polynomial time, $d:\mathscr{C}(Q',\Sigma')\to\mathscr{C}(Q,\Sigma)$ a function computable in polynomial time satisfying $d\cdot e=\mathrm{id}$, and f a function from $\mathbf{N}^2$ to $\mathbf{N}$. We say that M' *simulates* M for t steps with *accuracy* ε and *slowdown* f (under the *encoding* e and the *decoding* d), if for any $C_0\in\mathscr{C}(Q,\Sigma)$, we have

$$\sum_{C'\in\mathscr{C}(Q,\Sigma)}\left|\,|\langle C'|M_\delta^t|C_0\rangle|^2-\sum_{C\in d^{-1}(C')}|\langle C|M_{\delta'}^{f(t,\lceil\frac{1}{\varepsilon}\rceil)}|e(C_0)\rangle|^2\right|\leqslant\varepsilon. \tag{1}$$

If f depends only on t and Eq. (1) is satisfied for $\varepsilon=0$, we merely say that M' simulates M for t steps with slowdown f. In particular, we say that M' simulates M for t steps *by a factor of* s if $f(t)=st$.

We have discussed solely single tape QTMs, but our arguments can be adapted easily to multi-tape QTMs. We refer to [17] for the formulation of multi-tape QTMs.

3. Basic lemmas for QTMs

In this section, we present several definitions, lemmas and theorems necessary to prove theorems in Sections 4 and 5. Except for Lemma 3.2, they are given by Bernstein and Vazirani [8] and we adapt them to the present approach. We refer to [8] for these proofs. In [8], the dovetailing lemma and the branching lemma are given for two-way QTMs, but we extend them to general QTMs including multi-tape QTMs.

Let $S\subseteq Q\times\Sigma$. A complex-valued function δ on $S\times Q\times\Sigma\times[-1,1]_{\mathbf{Z}}$ is *unidirectional*, if we have $d=d'$ whenever $\delta(p,\sigma,q,\tau,d)$ and $\delta(p',\sigma',q,\tau',d')$ are both non-zero, where $q\in Q$, $(p,\sigma),(p',\sigma')\in S$, $\tau,\tau'\in\Sigma$, and $d,d'\in[-1,1]_{\mathbf{Z}}$. A prequantum Turing machine (or QTM) is said to be *unidirectional* if the quantum transition function is unidirectional. This definition is a natural extension of the definition of [8] to the case where the head is not required to move. It is easy to see that a unidirectional prequantum Turing machine is a unidirectional QTM if it satisfies conditions (a) and

(b) of Theorem 2.1. We can show the following lemma for a unidirectional QTM in a similar way to [8]. This lemma allows us to extend a partially defined unidirectional quantum transition function to characterize a QTM.

Lemma 3.1 (Completion lemma). *Let δ' be a unidirectional function on $S \times Q \times \Sigma \times [-1,1]_{\mathbf{Z}}$, where $S \subseteq Q \times \Sigma$. Assume that δ' satisfies the following conditions* (a) *and* (b),

(a) *For any $(q,\sigma) \in S$,*

$$\sum_{p \in Q, \tau \in \Sigma, d \in [-1,1]_{\mathbf{Z}}} |\delta'(q,\sigma,p,\tau,d)|^2 = 1.$$

(b) *For any $(q,\sigma),(q',\sigma') \in S$ with $(q,\sigma) \neq (q',\sigma')$,*

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

Then there is a unidirectional QTM $M = (Q,\Sigma,\delta)$ such that $\delta(p,\sigma,q,\tau,d) = \delta'(p,\sigma,q,\tau,d)$ whenever $\delta'(p,\sigma,q,\tau,d)$ is defined.

As is well-known, any deterministic Turing machine (DTM) $M = (Q,\Sigma,\delta)$ can be simulated by a DTM $M' = (Q',\{B,1\},\delta')$ with slowdown by a factor of $\lceil \log |\Sigma| \rceil$. Using the completion lemma, we can prove a similar statement for unidirectional QTMs.

Lemma 3.2. *Any unidirectional QTM $M = (Q,\Sigma,\delta)$ can be simulated by a unidirectional QTM $M' = (Q',\{B,1\},\delta')$ with slowdown by a factor of* $3k$, *where $k = \lceil \log |\Sigma| \rceil$.*

Proof. Throughout this proof, we denote by $\sigma_0 \cdots \sigma_{k-1}$ the binary representation of $\sigma \in \Sigma$. Let $Q' = (Q \times \{1\}) \cup (\bigcup_{j=1}^{k} (Q \times \{B,1\}^j \times \{1,2\})) \cup (Q \times [1,k-1]_{\mathbf{Z}} \times \{3\})$. We define the function $e : \mathscr{C}(Q,\Sigma) \to \mathscr{C}(Q',\{B,1\})$ such that $e(p,T,\xi) = (p,\tilde{T},k\xi)$, where $\tilde{T}$ is the tape configuration from $\{B,1\}$ such that $\tilde{T}(kj) \cdots \tilde{T}(kj+k-1) = \sigma_0 \cdots \sigma_{k-1}$ if $T(j) = \sigma$ for any $j \in \mathbf{Z}$, that is, the function e determines the configuration of M' corresponding to a configuration of M. If a state $|p,T,\xi\rangle$ of M such that $T(\xi) = \sigma$ evolves to $|q,T_\xi^\tau,\xi+d\rangle$ with amplitude $\delta(p,\sigma,q,\tau,d)$, the corresponding state $|p,\tilde{T},k\xi\rangle$ of M' evolves to $|q,\tilde{T}_\xi^{(\tau)},k(\xi+d)\rangle$ with the same amplitude in $3k$ steps by the following function δ' on $S = (Q_1 \times \{B,1\}) \cup (Q_2 \times \{B\})$.

$$\delta'((p,\sigma_0,\ldots,\sigma_{i-1},1),\sigma_i,(p,\sigma_0,\ldots,\sigma_i,1),B,1) = 1 \quad (0 \leqslant i \leqslant k-1), \tag{2}$$

$$\begin{aligned}&\delta'((p,\sigma_0,\ldots,\sigma_{k-1},1),b,(q,\tau_0,\ldots,\tau_{k-1},2),b,-1)\\ &\quad = \delta(p,\sigma,q,\tau,d) \ (b \in \{B,1\}),\end{aligned} \tag{3}$$

$$\delta'((q,\tau_0,\ldots,\tau_i,2),B,(q,\tau_0,\ldots,\tau_{i-1},2),\tau_i,-1) = 1 \quad (1 \leqslant i \leqslant k-1), \tag{4}$$

$$\delta'((q,\tau_0,2),B,(q,1,3),\tau_0,d)=1, \tag{5}$$

$$\delta'((q,i,3),\tau_i,(q,i+1,3),\tau_i,d)=1 \quad (1\leqslant i\leqslant k-1,\ (q,k,3)=(q,1)). \tag{6}$$

Here, let $b\in\{B,1\}$, we put $Q_1=(\bigcup_{j=0}^{k-1}(Q\times\{B,1\}^j\times\{1\}))\cup(Q\times\Sigma'\times\{1\})\cup(Q\times[1,k-1]_{\mathbf{Z}}\times\{3\})$, where Σ' is the subset of $\{0,1\}^k$ corresponding to Σ, we put $Q_2=\bigcup_{j=1}^{k}(Q\times\{B,1\}^j\times\{2\})$, and $\tilde{T}_{\xi}^{(\tau)}$ is the tape configuration from $\{B,1\}$ defined by

$$\tilde{T}_{\xi}^{(\tau)}(m)=\begin{cases}\tau_{m \bmod k} & \text{if } k\xi\leqslant m\leqslant k\xi+k-1,\\ \tilde{T}(m) & \text{otherwise.}\end{cases}$$

For any element (p,σ,q,τ,d) except the elements defined by the above equations, we define $\delta'(p,\sigma,q,\tau,d)=0$. Eq. (2) represents the operation of recording the current symbol σ scanned by the head of M in the processor of M' in k steps. Eq. (3) represents the operation of transforming the processor configuration p and the symbol σ of M recorded in the processor of M' to a new processor configuration q and symbol τ with amplitude $\delta(p,\sigma,q,\tau,d)$. Since M is unidirectional, the direction d in which the head of M moves is uniquely determined by q. Eqs. (4) and (5) represent the operation of writing the symbol string corresponding to the new symbol τ of M in turn on k cells of M' in k steps. Eq. (6) represents the operation of moving the head of M' to the direction d in $k-1$ steps. By the above operations, M' carries out the operation corresponding to one step of M.

We can see that the function δ' is unidirectional and satisfies conditions (a) and (b) of the completion lemma, so that there exists a quantum transition function that carries out the above steps by the completion lemma. It is easy to see that M' simulates M with slowdown by a factor of $3k$. □

Since every two-way QTM is simulated by a unidirectional QTM with slowdown by a factor of 5 [8, Lemma 5.5], Lemma 3.2 implies that any two-way QTM is simulated by a unidirectional QTM with the binary tape with slowdown by a constant factor independent of the input.

A reversible Turing machine (RTM) $M=(Q,\Sigma,\delta)$ with classical transition function δ can be canonically identified with the QTM $M'=(Q,\Sigma,\delta')$ such that the range of δ' is $\{0,1\}$ and that $\delta'(p,\sigma,q,\tau,d)=1$ if and only if $\delta(p,\sigma)=(q,\tau,d)$ for any $(p,\sigma,q,\tau,d)\in Q\times\Sigma\times Q\times\Sigma\times[-1,1]_{\mathbf{Z}}$. We consider that the class of RTMs is a subclass of the class of QTMs under this identification. Then, the RTM identified with an SNQTM is called a stationary, normal form RTM and we abbreviate it as an "SNRTM".

Theorem 3.3 (Synchronization theorem). *If f is a function mapping symbol strings to symbol strings which can be computed by a DTM in polynomial time and if $|f(x)|$ depends only on $|x|$, then there is a two-way SNRTM such that the output state for input state $|q_0,\mathrm{tape}[x],0\rangle$ is $|q_f,\mathrm{tape}[x,f(x)],0\rangle$ and whose computation time is a polynomial in $|x|$. Moreover, if f and f^{-1} can be computed by DTMs in polynomial time and if $|f(x)|$ depends only on $|x|$, then there is a two-way SNRTM such*

that the output state for input state $|q_0, \mathrm{tape}[x], 0\rangle$ *is* $|q_f, \mathrm{tape}[f(x)], 0\rangle$ *and that the computation time is a polynomial in* $|x|$.

Given any QTM $M=(Q,\Sigma,\delta)$ and any symbol set Σ', the QTM $M(\Sigma')=(Q,\Sigma\times\Sigma',\delta')$ is called the QTM constructed by the *addition* of the track (with alphabet Σ' to M) if for any $(p,(\sigma,\sigma'),q,(\tau,\tau'),d)\in(Q\times(\Sigma\times\Sigma'))^2\times[-1,1]_{\mathbf{Z}}$, we have

$$\delta'(p,(\sigma,\sigma'),q,(\tau,\tau'),d)=\Delta_{\sigma'}^{\tau'}\delta(p,\sigma,q,\tau,d),$$

where Δ denotes the Kronecker delta. Given any k-track QTM $M=(Q,\Sigma_1\times\cdots\times\Sigma_k,\delta)$ and any permutation $\pi:[1,k]_{\mathbf{Z}}\to[1,k]_{\mathbf{Z}}$, the k-track QTM $M'=(Q,\Sigma_{\pi(1)}\times\cdots\times\Sigma_{\pi(k)},\delta')$ is called the QTM constructed by the *permutation* π of the tracks (of M) if for any $(p,(\sigma_{\pi(1)},\ldots,\sigma_{\pi(k)}),q,(\tau_{\pi(1)},\ldots,\tau_{\pi(k)}),d)\in(Q\times(\Sigma_{\pi(1)}\times\cdots\times\Sigma_{\pi(k)}))^2\times[-1,1]_{\mathbf{Z}}$, we have

$$\delta'(p,(\sigma_{\pi(1)},\ldots,\sigma_{\pi(k)}),q,(\tau_{\pi(1)},\ldots,\tau_{\pi(k)}),d)=\delta(p,(\sigma_1,\ldots,\sigma_k),q,(\tau_1,\ldots,\tau_k),d).$$

Lemma 3.4 (Dovetailing lemma). *For* $i=1,2$, *let* $M_i=(Q_i,\Sigma,\delta_i)$ *be an SNQTM with initial and final processor configurations* $q_{i,0}$ *and* $q_{i,f}$. *Then there is a normal form QTM* $M=(Q,\Sigma,\delta)$ *with initial and final processor configurations* $q_{1,0}$ *and* $q_{2,f}$ *satisfying the following condition*: *If* C_0 *is an initial configuration of* M_1, *the computation time for the input state* $|C_0\rangle$ *of* M_1 *is* s, *and* $M_{\delta_1}^s|C_0\rangle=\sum_{T\in\Sigma^\#}\alpha_T|q_{1,f},T,0\rangle$, *then we have*

$$M_\delta^t|C_0\rangle=M_{\delta_1}^t|C_0\rangle \quad for\ t<s,$$

$$M_\delta^{s+t}|C_0\rangle=\sum_{T\in\Sigma^\#}\alpha_T M_{\delta_2}^t|q_{2,0},T,0\rangle \quad for\ t\geqslant 0.$$

Such an M is called the QTM constructed by dovetailing M_1 and M_2.

Even if M is the normal form QTM constructed by dovetailing SNQTMs M_1 and M_2, it is not always stationary. What conditions ensure that the QTM M is stationary? It is easy to see that one of the answers is to satisfy the following conditions (i) and (ii).

(i) The output state of M_1 for input state $|q_0,\mathrm{tape}[x],0\rangle$ is represented by

$$\sum_{y\in\Sigma^n}\alpha_y|q_f,\mathrm{tape}[y],0\rangle$$

for some integer n, where n depends on $|x|$.

(ii) M_2 is a stationary QTM such that if the input state is $|q_0,\mathrm{tape}[x],0\rangle$, the computation time for the input state depends only on $|x|$.

Condition (i) ensures that all computational basis vectors in the final superposition of M_1 represent the output strings of the same length, and condition (ii) ensures that if the final superposition of M_1 satisfying condition (i) is given as the initial state of M_2, every computational path of M_2 reaches a final configuration simultaneously. These conditions are called the *dovetailing conditions*.

Lemma 3.5 (Branching lemma). *Let $M_i=(Q_i,\Sigma,\delta_i)$ be an SNQTM for $i=1,2$. Then there is an SNQTM $M=(Q,\Sigma\times\{B,1\},\delta)$ satisfying the following condition with initial and final processor configurations q_0 and q_f. If the initial configuration of M_i is $C_i=(q_{i,0},T_0,0)$ such that the computation time of M_i for $|C_i\rangle$ is s_i and that $M_{\delta_i}^{s_i}|C_i\rangle=\sum_{T\in\Sigma^\#}\alpha_{i,T}|q_{i,f},T,0\rangle$, then we have*

$$M_\delta^{s_i+4}|q_0,(T_0,T_i),0\rangle=\sum_{T\in\Sigma^\#}\alpha_{i,T}|q_f,(T,T_i),0\rangle,$$

where $T_1=\mathrm{tape}[B]$ and $T_2=\mathrm{tape}[1]$.

Lemma 3.6 (Looping lemma). *There are an SNRTM $M=(Q,\Sigma,\delta)$ and a constant c with the following properties. On any positive input k written in binary, the computation time of M is $t=\mathrm{O}(k\log^c k)$ and the output state of M for the input state $|q_0,T,0\rangle$ is $|q_f,T,0\rangle$. Moreover, M on input k visits a special processor configuration q^* exactly k times, each time with its head back in cell 0. That is, there exist some q^* in Q and k positive integers $t_i<t$, where $i=1,\dots,k$, such that*

$$\|E^{\hat{q}}(q^*)E^{\hat{\xi}}(0)M_\delta^{t_i}|q_0,T,0\rangle\|^2=1\quad\text{and}\quad\|E^{\hat{q}}(q^*)M_\delta^{s}|q_0,T,0\rangle\|^2=0 \tag{7}$$

for all $s\neq t_1,\dots,t_k$.

An RTM M satisfying the above condition is called a looping machine.

For any real number $\varepsilon>0$, we denote by $\mathrm{Acc}(\varepsilon)$ the least number m satisfying $1/2^m\leqslant\varepsilon$. For convenience, we define $\mathrm{Acc}(0)=B$. Let $\tilde{\mathbf{C}}=\{a+ib\mid a,b\in\mathbf{Q}\}$. The *code* of an $m\times n$ matrix $M=(m_{ij})$ with the components in $\tilde{\mathbf{C}}$ is defined to be the list of finite sequences of numbers $\langle\langle x_{11},y_{11}\rangle,\langle x_{12},y_{12}\rangle,\dots,\langle x_{mn},y_{mn}\rangle\rangle$, where $x_{ij}=\mathrm{Re}(m_{ij})$ and $y_{ij}=\mathrm{Im}(m_{ij})$.

Let $\mathscr{H}$ be the Hilbert space spanned by the orthonormal system $\mathscr{B}=\{|1\rangle,\dots,|n\rangle\}$ and $\mathscr{L}(\mathscr{H})$ be the set of all linear transformations on $\mathscr{H}$. Let e be a function mapping any $(U,\varepsilon)\in\mathscr{L}(\mathscr{H})\times\mathbf{R}_{\geqslant 0}$ to the following finite string $e(U,\varepsilon)$: if U has the matrix $A=(a_{ij})$ with $a_{ij}=\langle i|U|j\rangle$, then $e(U,\varepsilon)$ is the code of $A'=(a'_{ij})$, where A' is the element of the set $\mathscr{X}=\{B=(b_{ij})\mid b_{ij}\in\tilde{\mathbf{C}},\ \|A-B\|\leqslant\varepsilon\}$ chosen uniquely by appropriate means. We call $e(U,\varepsilon)$ the *ε-approximate code of U*. Let M be a multi-track QTM such that the alphabet of each track contains 0 and 1. For some U in $\mathscr{L}(\mathscr{H})$, we say that *given the ε'-approximate code, a QTM M carries out U with accuracy ε (in t steps on the first track)*, if there is a unitary transformation U' such that $\|U'-U\|\leqslant\varepsilon$ and for any $|j\rangle\in\mathscr{B}$ we have

$$M_\delta^t|q_0,\mathrm{tape}[j,e(U,\varepsilon'),\mathrm{Acc}(\varepsilon)],0\rangle=\sum_{i=1}^{n}|q_f,\mathrm{tape}[i,e(U,\varepsilon'),\mathrm{Acc}(\varepsilon)],0\rangle\langle i|\,U'|j\rangle.$$

In particular, if $\varepsilon=\varepsilon'=0$ in the above condition, we merely say that *M carries out U (in t steps)*. Analogously, we say that M carries out U with accuracy ε in t steps on the *ith track* under appropriate modification of the above definition.

The following theorem is a restricted version of the unitary theorem found by Bernstein and Vazirani [8], but it serves our purpose.

Theorem 3.7 (Unitary theorem). *Let $\mathscr{H}$ be the Hilbert space spanned by the orthonormal system $\mathscr{B}=\{|1\rangle,\ldots,|n\rangle\}$. Then there is a two-way SNQTM M that for any unitary transformation U on $\mathscr{H}$, given the $\varepsilon/4(10\sqrt{n})^n$-approximate code, carries out U with accuracy ε in time polynomial in $1/\varepsilon$ and the length of the input on its first track.*

4. Quantum circuits

An element of $\{0,1\}^m$ is called a *bit string of length m* or an *m-bit string*. For any m-bit string $x=x_1\ldots x_m$, the bit x_i is called the *ith bit* of x. An m-input n-output *Boolean gate* is a function mapping m-bit strings to n-bit strings. An n-input n-output Boolean gate is called an n-bit Boolean gate. Suppose that G is an m-input n-output Boolean gate. An n-bit string $y_1\ldots y_n$ is called the *output* of G for *input* $x_1\ldots x_m$ if $G(x_1\ldots x_m)=y_1\ldots y_n$. A Boolean gate G is said to be *reversible* if G is a bijection. For example, the Boolean gate $M_2(N)$ that for input $xy\in\{0,1\}^2$ produces output $x(x+y \bmod 2)\in\{0,1\}^2$ is a 2-bit reversible Boolean gate called the controlled not gate (Fig. 1). The first bit is called the control bit, and the second bit is called the target bit.

To define quantum gates, we shall first introduce the notion of a wire. A *wire* is an element of a countable set of 2-state systems. The set of wires is in one-to-one correspondence with the set of natural numbers called *bit numbers*. Formally, the wire of bit number j is represented by the Hilbert space $\mathscr{H}_j\cong\mathbf{C}^2$ spanned by a basis $\{|0\rangle_j,|1\rangle_j\}$, an orthonormal system in one-to-one correspondence with $\{0,1\}$. An observable $\hat{n}_j=|1\rangle_j\langle 1|_j$ in the Hilbert space $\mathscr{H}_j$ is called a jth *bit observable*. Let $\Lambda=\{j_1,\ldots,j_n\}\subseteq\mathbf{N}$, where $j_1<\cdots<j_n$. A composite system of n wires with different bit numbers in Λ is represented by the Hilbert space $\mathscr{H}_\Lambda=\bigotimes_{j\in\Lambda}\mathscr{H}_j$. In the Hilbert space $\mathscr{H}_\Lambda$, the orthonormal system

$$\{|x_1\rangle_{j_1}\ldots|x_n\rangle_{j_n}|\ x_1\ldots x_n\in\{0,1\}^n\}$$

in one-to-one correspondence with $\{0,1\}^n$ is called the *computational basis* on Λ. Henceforth, we shall also write $|x_1,\ldots,x_n\rangle=|x_1\rangle_{j_1}\ldots|x_n\rangle_{j_n}$. Thus, we obtain

$$1\otimes\cdots 1\otimes\hat{n}_{j_k}\otimes 1\cdots\otimes 1|x_1,\ldots,x_k,\ldots,x_n\rangle=x_k|x_1,\ldots,x_k,\ldots,x_n\rangle.$$

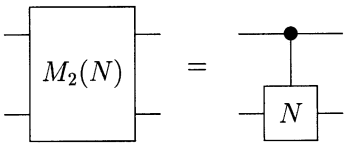

Fig. 1. The controlled not gate $M_2(N)$.

An n-bit quantum gate is physically to be interacting with n wires such that the state transition from the input state to the output state is represented by the time evolution of the composite system of the n wires. Formally, for any set $\Lambda \subseteq \mathbf{N}$, a *$\Lambda$-quantum gate* is defined to be a unitary operator on the corresponding Hilbert space $\mathscr{H}_\Lambda$. In particular, a $[1,n]_{\mathbf{Z}}$-quantum gate is called an *n-bit quantum gate*. The *S-matrix* of a Λ-quantum gate is the matrix representing its gate in the computational basis on Λ. For any Λ-quantum gate G and any unit vectors $|\psi\rangle$ and $|\phi\rangle$ in $\mathscr{H}_\Lambda$, if $G|\psi\rangle = |\phi\rangle$, the vector $|\phi\rangle$ is called the *output state* of G for the *input state* $|\psi\rangle$. In particular, if the input state is $|\psi\rangle = |x_1 \ldots x_n\rangle$, the bit string $x_1 \ldots x_n$ is called the *input* of G. Henceforth when no confusion may arise, we usually identify the S-matrix of a quantum gate with the quantum gate itself.

We can represent an n-bit reversible Boolean gate by a $2^n \times 2^n$ orthogonal matrix whose entries are equal to zero or one. Thus, we may consider an n-bit reversible Boolean gate to be a sort of n-bit quantum gate, and consider that the class of reversible Boolean gates is a subclass of the class of quantum gates.

Let π be a permutation on $[1,n]_{\mathbf{Z}}$. The *permutation operator* of π is the operator V_π on $\mathscr{H}_{[1,n]_{\mathbf{Z}}}$ that transforms $|x_1 \ldots x_n\rangle$ to $|x_{\pi(1)} \ldots x_{\pi(n)}\rangle$ for any n-bit string $x_1 \ldots x_n$. For any finite set Λ, we denote by I_Λ the identity operator on $\mathscr{H}_\Lambda = \bigotimes_{\lambda \in \Lambda} \mathscr{H}_\lambda$. For any m-bit quantum gate G, the *n-bit extension* of G is the n-bit quantum gate $G \otimes I_{[m+1,n]_{\mathbf{Z}}}$ denoted by $G[n]$, where $m \leqslant n$. For any set $\mathscr{G}$ of quantum gates, an n-bit quantum gate G is said to be *decomposable by $\mathscr{G}$* if there are n_i-bit quantum gates G_i in $\mathscr{G}$ with $n_i \leqslant n$ and permutations π_i on $[1,n]_{\mathbf{Z}}$ satisfying

$$G = U_1 \ldots U_m \quad \text{where } U_i = V_{\pi_i}^\dagger G_i[n] V_{\pi_i} \tag{8}$$

for $i = 1, 2, \ldots, m$. In this case, G is also said to be *decomposable by m gates in $\mathscr{G}$*. The least number of such m is called the *size of G for $\mathscr{G}$*. For any $\varepsilon > 0$, we say that G is *decomposable by $\mathscr{G}$ with accuracy ε*, if $\|G - U_1 \ldots U_m\| \leqslant \varepsilon$ is satisfied instead of Eq. (8).

A *universal set* is a set of quantum gates by which any quantum gate is decomposable with any accuracy. An *elementary gate* is an element of a given universal set. Henceforth, $R_{1,\theta}, R_{2,\theta}$, and $R_{3,\theta}$ denote the 1-bit quantum gates whose S-matrices are given as follows.

$$R_{1,\theta} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \quad R_{2,\theta} = \begin{pmatrix} e^{i\theta} & 0 \\ 0 & 1 \end{pmatrix}, \quad R_{3,\theta} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}.$$

Barenco et al. [3] proved that any quantum gate is decomposable by the infinite set

$$\mathscr{G}_u = \{R_{1,\theta}, R_{2,\theta}, R_{3,\theta}, M_2(N) \mid \theta \in [0, 2\pi]\}.$$

as follows.

Theorem 4.1. *Any n-bit quantum gate G is decomposable by at most* $O(n^3 2^{2n})$ *quantum gates in* $\mathscr{G}_u$.

Thus, $\mathscr{G}_u$ is a universal set. In what follows, the size of a quantum gate for $\mathscr{G}_u$ is merely called the *size* of the quantum gate.

We shall now consider a finite universal set. Henceforth, $\mathscr{R}$ denotes a polynomial time computable real $2\pi\sum_{i=1}^{\infty} 2^{-2^i}$. The following lemma was obtained essentially by Bernstein and Vazirani [8].

Lemma 4.2. *For any* $\theta \in [0, 2\pi]$ *and* $\varepsilon > 0$, *there is a non-negative integer* $k \leqslant O(1/\varepsilon^4)$ *such that* $|k\mathscr{R} - \theta| \pmod{2\pi} \leqslant \varepsilon$. *Moreover, there is a DTM which produces on input* $\theta \in P\mathbf{R}$ *and* $\mathrm{Acc}(\varepsilon)$ *a non-negative integer* $k \leqslant O(1/\varepsilon^4)$ *satisfying the above inequality in time polynomial in the length of the input.*

Henceforth, $\mathscr{G}_{\mathscr{R}}$ denotes the finite set of quantum gates defined by

$$\mathscr{G}_{\mathscr{R}} = \left\{ R_{1,\mathscr{R}}, R_{2,\mathscr{R}}, R_{3,\mathscr{R}}, M_2(N) \mid \mathscr{R} = 2\pi \sum_{i=1}^{\infty} 2^{-2^i} \right\}.$$

Since any 1-bit quantum gate in $\mathscr{G}_u$ is decomposable by $\mathscr{G}_{\mathscr{R}}$ with any accuracy by Lemma 4.2, the set $\mathscr{G}_{\mathscr{R}}$ is a universal set. In what follows, the size of a quantum gate for $\mathscr{G}_{\mathscr{R}}$ is called the $\mathscr{G}_{\mathscr{R}}$-*size* of the quantum gate.

An n-bit quantum circuit consists of quantum gates and wires, and represents how those gates are connected with some of those wires. Formally, it is defined as follows. Let $\mathscr{G}$ be a set of quantum gates. An n-bit *quantum circuit K based on* $\mathscr{G}$ is a finite sequence $(G_m, \pi_m), \ldots, (G_1, \pi_1)$ such that each pair (G_i, π_i) satisfies the following conditions.

(1) G_i is an n_i-bit quantum gate in $\mathscr{G}$ with $n_i \leqslant n$.

(2) π_i is a permutation on $[1, n]_{\mathbf{Z}}$.

In this case, we say that the wire of bit number $\pi_i(j)$, where $j \leqslant n_i$, *is connected with* the *jth pin* of G_i. The positive integer m is called the *size of K for* $\mathscr{G}$. In particular, the size of K for $\mathscr{G}_u$ is merely called the *size* of K and the size of K for $\mathscr{G}_{\mathscr{R}}$ is called the $\mathscr{G}_{\mathscr{R}}$-*size* of K. The unitary operator $U_m \cdots U_1$, where $U_i = V_{\pi_i}^{\dagger} G_i[n] V_{\pi_i}$ for $i \in [1, m]_{\mathbf{Z}}$, is called the n-bit *quantum gate determined by K* and denoted by $G(K)$. From the definition, the size of $G(K)$ for $\mathscr{G}$ is at most the size of K for $\mathscr{G}$. Suppose that $K_1 = (G_m, \pi_m), \ldots, (G_1, \pi_1)$ and $K_2 = (G'_l, \pi'_l), \ldots, (G'_1, \pi'_1)$ are n-bit quantum circuits based on $\mathscr{G}$. Then $K_2 \circ K_1 = (G'_l, \pi'_l), \ldots, (G'_1, \pi'_1), (G_m, \pi_m), \ldots, (G_1, \pi_1)$ is called the concatenation of K_1 and K_2, and $K_1^n = \underbrace{K_1 \circ \cdots \circ K_1}_{n}$ is called the concatenation of n K_1's.

Next, we define k-input m-output quantum circuits. A k-input m-output n-bit quantum circuit is physically to be an n-bit quantum circuit based on a set of quantum gates; its input is a k-bit string and a constant $(n-k)$-bit string, and its output is the m-bit string obtained by measuring the bit observables of specified m wires after the unitary transformation determined by the circuit.

Formally, a *k-input m-output n*-bit *quantum circuit* $\mathbf{K}$ is a 4-tuple $(K, \Lambda_1, \Lambda_2, S)$ satisfying the following conditions.

(1) K is an n-bit quantum circuit.

(2) Λ_1 and Λ_2 are two subsets of $[1,n]_{\mathbf{Z}}$ satisfying $|\Lambda_1| = k$ and $|\Lambda_2| = m$, respectively.

(3) S is a function from $[1,n]_{\mathbf{Z}} \setminus \Lambda_1$ to $\{0,1\}$.

Henceforth, we write $b_j = S(j)$ for any $j \in [1,n]_{\mathbf{Z}} \setminus \Lambda_1$.

Let $\mathbf{K} = (K, \Lambda_1, \Lambda_2, S)$ be a k-input m-output n-bit quantum circuit, where $\Lambda_1 = \{j_1, \ldots, j_k\}$ and $\Lambda_2 = \{i_1, \ldots, i_m\}$, and let $u = u_1 \cdots u_n$ be the n-bit string satisfying $u_{j_1} = x_1, \ldots, u_{j_k} = x_k$ for a k-bit string $x = x_1 \cdots x_k$ and $u_j = b_j$ for all $j \in [1,n]_{\mathbf{Z}} \setminus \Lambda_1$. In what follows, the n-bit string u obtained by such construction is denoted by $u(x, \mathbf{K})$. Let $|\phi\rangle$ be the output state of $G(K)$ for input $u(x, \mathbf{K})$. If the bit observables $\hat{n}_{i_1}, \ldots, \hat{n}_{i_m}$ are measured simultaneously in the output state $|\phi\rangle$, and the outcomes of these measurements are $y_1, \ldots, y_m$, then the bit string $y = y_1 \ldots y_m$ is considered as the *output* of $\mathbf{K}$ for *input* x. From the statistical formula of quantum physics, the probability $\rho^K(y|x)$ such that y is the output of $\mathbf{K}$ for input x is represented by

$$\rho^K(y|x) = \langle u(x, \mathbf{K}) | G(K)^\dagger E_{i_1}(y_1) \cdots E_{i_m}(y_m) G(K) | u(x, \mathbf{K}) \rangle,$$

where $E_{i_p}(y_p)$ is the spectral projection of $1 \otimes \cdots 1 \otimes \hat{n}_{i_p} \otimes 1 \cdots \otimes 1$ pertaining to its eigenvalue y_p. We can consider that $\mathbf{K}$ associates each k-bit string x with the probability distribution $\rho^K(\cdot\,|x)$ on $\{0,1\}^m$. The distribution $\rho^K(\cdot\,|x)$ is called the *output distribution for x determined by* $\mathbf{K}$. Henceforth, when no confusion may arise, we shall identify $\mathbf{K}$ with K.

Now, we shall give the notion of a simulation of a QTM by a quantum circuit. The total variation distance between two distributions $\mathscr{D}$ and $\mathscr{D}'$ over the same domain I is $\sum_{i \in I} |\mathscr{D}(i) - \mathscr{D}'(i)|$. A quantum circuit K will be said to t-simulate a QTM $M = (Q, \Sigma, \delta)$ with accuracy ε, if the following holds for any Σ-string x. Let $\mathscr{D}$ be the probability distribution of the outcomes of the simultaneous measurement of the tape cells from cell $-t$ to cell t after t steps of M for input state $|q_0, \mathrm{tape}[x], 0\rangle$. Let $\mathscr{D}'$ be the probability distribution of the Σ-string obtained by decoding the output of K for the input of the bit string obtained by encoding x. Then the total variation distance between $\mathscr{D}$ and $\mathscr{D}'$ is at most ε. Formally, it is defined as follows.

Let $e : \Sigma \to \{0,1\}^\lambda$, where $\lambda = \lceil \log |\Sigma| \rceil$, be an injection computable in polynomial time, and let $d : \{0,1\}^\lambda \to \Sigma$ be a function computable in polynomial time such that $d \cdot e = \mathrm{id}$. For any Σ-string $x = x_1 \cdots x_k$, positive integer t and bit string $z = z_1 \cdots z_{2t+1}$, where $z_i \in \{0,1\}^\lambda$, we define the encoding function $e_t : \Sigma^* \to \{0,1\}^{(2t+1)\lambda}$ by

$$e_t(x_1 \cdots x_k) = \begin{cases} \underbrace{e(B) \cdots e(B)}_{t}\, e(x_1) \cdots e(x_k)\, \underbrace{e(B) \cdots e(B)}_{t+1-k} & \text{if } t+1 \geqslant k, \\ \underbrace{e(B) \cdots e(B)}_{t}\, e(x_1) \cdots e(x_{t+1}) & \text{if } t+1 < k, \end{cases}$$

and define the decoding function $d_t : \{0,1\}^{(2t+1)\lambda} \to \Sigma^{2t+1}$ by

$$d_t(z_1 \cdots z_{2t+1}) = d(z_1) \cdots d(z_{2t+1}).$$

Then a $((2t+1)\lambda$-input $(2t+1)\lambda$-output) quantum circuit K is said to *t-simulate* a QTM $M = (Q, \Sigma, \delta)$ with *accuracy* ε (under the *encoding* e_t and the *decoding* d_t), if for any Σ-string x, we have

$$\sum_{y \in \Sigma^{2t+1}} |\rho_t^M(y|x) - \tilde{\rho}^K(y|x)| \leqslant \varepsilon,$$

where

$$\tilde{\rho}^K(y|x) = \sum_{z \in d_t^{-1}(y)} \rho^K(z|e_t(x)),$$
$$\rho_t^M(y|x) = \langle q_0, \mathrm{tape}[x], 0|(M_\delta^t)^\dagger E_{M,-t}(y_1) \cdots E_{M,t}(y_{2t+1}) M_\delta^t |q_0, \ \mathrm{tape}[x], 0\rangle.$$

When $\varepsilon = 0$, the quantum circuit K is merely said to *t-simulate* the QTM M.

Yao [24] discussed the simulation of a QTM by a quantum circuit under a similar but different formulation. He showed that given a QTM $M = (Q, \Sigma, \delta)$ and positive integers t and n, there is an n-input quantum circuit that simulates M for t steps on any input of M with length $\lceil n/\lceil \log |\Sigma| \rceil \rceil$ and that its "size" (the "size" is the number of Deutsch gates [11] constructing the circuit) is at most some fixed polynomial in t and n. Our formulation requires that a quantum circuit simulate a QTM M on *every* input of M and we shall extend quantum circuits used by Yao [24] to those which can simulate multi-tape QTMs. In addition, we shall construct a quantum circuit based on $\mathscr{G}_u$ instead of Deutsch gates in order to take advantage of this simulation later.

Theorem 4.3. *Let $M = (Q, \Sigma, \delta)$ be a k-tape QTM, and let $t \in \mathbf{N}$. Then, there is a quantum circuit of size $\mathrm{O}(t^{k+1})$ that t-simulates M.*

Proof. We consider the case where M is a single tape QTM. See Appendix A for the generalization to multi-tape QTMs. We shall construct a quantum circuit $K_{\mathscr{G}}$ which t-simulates M. The quantum gate determined by $K_{\mathscr{G}}$ is connected with $l_0 + (2t+1)l$ wires, where $l_0 = \lceil \log |Q| \rceil$ and $l = 2 + \lceil \log |\Sigma| \rceil$. We divide their wires into a part consisting of the first l_0 wires and $2t+1$ parts which are, respectively, consisting of l wires. The part consisting of the first l_0 wires represents the processor configuration of M. This set of wires is called cell 'P' of $K_{\mathscr{G}}$. The state of cell P of $K_{\mathscr{G}}$ is represented by a unit vector in the Hilbert space spanned by the computational basis $\{|q\rangle\}$, where $q \in \{0,1\}^{l_0}$. For $j \in [0, 2t]_{\mathbf{Z}}$, the wires of bit numbers $l_0 + jl + 1, \ldots, l_0 + jl + l$ represent the symbol in the $(j-t)$th cell of M and whether the head scans this cell or not. This set of wires is called cell $j - t$ of $K_{\mathscr{G}}$. For $i \in [-t, t]_{\mathbf{Z}}$, the state of cell i of $K_{\mathscr{G}}$ is represented by a unit vector in the Hilbert space spanned by the computational basis $\{|\sigma_i s_i\rangle\}$, where $\sigma_i \in \{0,1\}^{\lceil \log |\Sigma| \rceil}$ and $s_i \in \{0,1\}^2$.

Next, we define quantum gates G_1 and G_2, two types of components of $K_{\mathscr{G}}$. In what follows, $p, q, \ldots$ denote binary strings representing elements of Q, the symbols

$\sigma, \tau, \ldots$ denote binary strings representing elements of Σ, and $s = \bar{0}, \bar{1}, \bar{2}$ denote 00,01,10, respectively. Then, we denote the computational basis state $|q\sigma_1 s_1 \sigma_2 s_2 \cdots \sigma_k s_k\rangle$ on the set $[1, l_0 + kl]_{\mathbf{Z}}$ of bit numbers by $|q; \sigma_1 s_1; \sigma_2 s_2; \cdots; \sigma_k s_k\rangle$. Now G_1 is an $(l_0 + 3l)$-bit quantum gate satisfying the following conditions (i) and (ii).

(i) $G_1|w_{p,\sigma_1,\sigma,\sigma_3}\rangle = |v_{p,\sigma_1,\sigma,\sigma_3}\rangle$, where

$$
\begin{aligned}
|w_{p,\sigma_1,\sigma,\sigma_3}\rangle &= |p; \sigma_1\bar{0}; \sigma\bar{1}; \sigma_3\bar{0}\rangle,\\
|v_{p,\sigma_1,\sigma,\sigma_3}\rangle &= \sum_{q,\tau} \delta(p,\sigma,q,\tau,-1)|q; \sigma_1\bar{2}; \tau\bar{0}; \sigma_3\bar{0}\rangle\\
&\quad + \sum_{q,\tau} \delta(p,\sigma,q,\tau,0)|q; \sigma_1\bar{0}; \tau\bar{2}; \sigma_3\bar{0}\rangle\\
&\quad + \sum_{q,\tau} \delta(p,\sigma,q,\tau,1)|q; \sigma_1\bar{0}; \tau\bar{0}; \sigma_3\bar{2}\rangle
\end{aligned}
$$

for any $(p, \sigma_1, \sigma, \sigma_3) \in Q \times \Sigma^3$; the summation $\sum_{q,\tau}$ is taken over all $(q, \tau) \in Q \times \Sigma$.

(ii) $G_1|h\rangle = |h\rangle$ for each vector $|h\rangle$ in the subspace H of $\mathbf{C}^{2^{l_0+3l}}$ spanned by three types of vectors:

(1) $|q; \sigma_1 s_1; \sigma_2 s_2; \sigma_3 s_3\rangle$, where $s_2 \neq \bar{1}$ and none of s_1, s_2, s_3 are equal to $\bar{2}$;

(2) $|u^1_{p,\sigma,\sigma_2,\sigma_3}\rangle = \sum_{q,\tau} \delta(p,\sigma,q,\tau,0)|q; \tau\bar{2}; \sigma_2\bar{0}; \sigma_3\bar{0}\rangle + \sum_{q,\tau} \delta(p,\sigma,q,\tau,1)|q; \tau\bar{0}; \sigma_2\bar{2}; \sigma_3\bar{0}\rangle$;

(3) $|u^2_{p,\sigma,\tau,\sigma_1,\sigma_2,\sigma_3}\rangle = \sum_{q\in Q} \delta(p,\sigma,q,\tau,1)|q; \sigma_1\bar{2}; \sigma_2\bar{0}; \sigma_3\bar{0}\rangle$.

Let $W = \{|w_{p,\sigma,\sigma_1,\sigma_3}\rangle |\ (p,\sigma,\sigma_1,\sigma_3) \in Q \times \Sigma^3\}^{\perp\perp}$ and $V = \{|v_{p,\sigma,\sigma_1,\sigma_3}\rangle |\ (p,\sigma,\sigma_1,\sigma_3) \in Q \times \Sigma^3\}^{\perp\perp}$, where $S^{\perp}$ denotes the orthogonal complement of a set S so that $S^{\perp\perp}$ denotes the subspace generated by S. By Theorem 2.1 the subspaces W, V and H are all orthogonal to one another and it is verified that $\{|v_{p,\sigma,\sigma_1,\sigma_3}\rangle\}$ is an orthonormal system of V. Thus, there exists a quantum gate G_1 satisfying the above condition. Let G_2 be an $(l_0 + (2t+1)l)$-bit reversible Boolean gate which changes all $s_i = \bar{2}$ to $s_i = \bar{1}$.

Henceforth, given any $m \in [1, 2t+1]_{\mathbf{Z}}$, we say that an $(l_0 + ml)$-bit quantum gate G is connected with cells $i_1, \ldots, i_m$, where $i_1 < \cdots < i_m$, if each j_0th pin of G, for $j_0 \in [1, l_0]_{\mathbf{Z}}$, and each $(l_0 + jl - l + k)$th pin of G, for $j \in [1, m]_{\mathbf{Z}}$, $k \in [1, l]_{\mathbf{Z}}$, are, respectively, connected with the wires of bit numbers j_0 and $l_0 + (i_j + t)l + k$. Now let $K_{\mathscr{G}}$ be the quantum circuit based on $\mathscr{G} = \{G_1, G_2\}$ constructed as follows. First, $2t-1$ G_1's are connected in such a way that for $j \in [1, 2t-1]_{\mathbf{Z}}$ the jth G_1 is connected with cells $j-t-1, j-t$ and $j-t+1$. The $(l_0 + (2t+1)l)$-bit quantum circuit constructed from these G_1's is called K_1. Lastly, G_2 is connected with cells $-t, -t+1, \ldots, t$. The $(l_0 + (2t+1)l)$-bit quantum circuit constructed from this G_2 is called K_2. Let $K_{\mathscr{G}} = (K_2 \circ K_1)^t$. The quantum circuit $K_2 \circ K_1$ is illustrated in Fig. 2. From the definitions of G_1 and G_2, it can be verified that $K_{\mathscr{G}}$ carries out the operation corresponding to one step of M as follows.

If the state of M after t' steps with $t' < t$ is $|p, T, i\rangle$ with $T(i) = \sigma$, the input state of the $(t'+1)$th $K_2 \circ K_1$ is

$$|p; T(-t)\bar{0}; \cdots; T(i-1)\bar{0}; T(i)\bar{1}; T(i+1)\bar{0}; \cdots; T(t)\bar{0}\rangle.$$

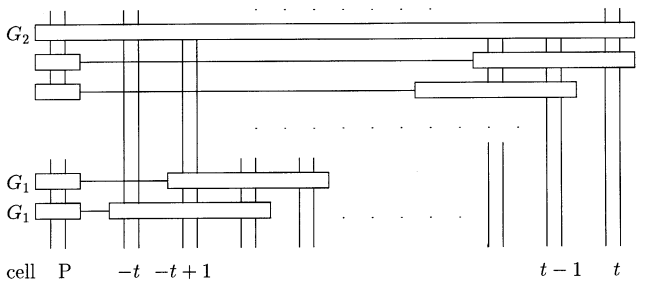

Fig. 2. The quantum circuit $K_2 \circ K_1$ based on $\mathscr{G}$.

From condition (ii—1) of G_1, this state does not change until ith G_1 is carried out. When ith G_1 is carried out, from condition (i) of G_1 this state is transformed into

$$\begin{aligned}
&\sum_{q,\tau} \delta(p,\sigma,q,\tau,-1)|q; T(-t)\bar{0}; \cdots; T(i-1)\bar{2}; \tau\bar{0}; T(i+1)\bar{0}; \cdots; T(t)\bar{0}\rangle \\
&\quad + \sum_{q,\tau} \delta(p,\sigma,q,\tau,0)|q; T(-t)\bar{0}; \cdots; T(i-1)\bar{0}; \tau\bar{2}; T(i+1)\bar{0}; \cdots; T(t)\bar{0}\rangle \\
&\quad + \sum_{q,\tau} \delta(p,\sigma,q,\tau,1)|q; T(-t)\bar{0}; \cdots; T(i-1)\bar{0}; \tau\bar{0}; T(i+1)\bar{2}; \cdots; T(t)\bar{0}\rangle.
\end{aligned}$$

By condition (ii) of G_1 this state does not change until K_2 is carried out. Finally, from the definition of G_2, the state after passing K_2 in the $(t'+1)$th $K_2 \circ K_1$ is transformed into

$$\begin{aligned}
&\sum_{q,\tau} \delta(p,\sigma,q,\tau,-1)|q; T(-t)\bar{0}; \cdots; T(i-1)\bar{1}; \tau\bar{0}; T(i+1)\bar{0}; \cdots; T(t)\bar{0}\rangle \\
&\quad + \sum_{q,\tau} \delta(p,\sigma,q,\tau,0)|q; T(-t)\bar{0}; \cdots; T(i-1)\bar{0}; \tau\bar{1}; T(i+1)\bar{0}; \cdots; T(t)\bar{0}\rangle \\
&\quad + \sum_{q,\tau} \delta(p,\sigma,q,\tau,1)|q; T(-t)\bar{0}; \cdots; T(i-1)\bar{0}; \tau\bar{0}; T(i+1)\bar{1}; \cdots; T(t)\bar{0}\rangle.
\end{aligned}$$

By the above transformation, it can be verified that $K_2 \circ K_1$ simulates the operation of M such that "if the processor configuration is p and the head reads the symbol σ of cell i after t' steps, then the head writes the symbol τ, the processor configuration turns to q, and the head moves to d with amplitude $\delta(p,\sigma,q,\tau,d)$".

From Theorem 4.1 the quantum gate G_1 is decomposable by O(1) gates in $\mathscr{G}_u$. It is easy to see that the quantum gate G_2 is decomposable by $O(2t+1)$ gates in $\mathscr{G}_u$. Thus there are an (l_0+3l)-bit quantum circuit $K_{u,1}$ of constant size and an $(l_0+(2t+1)l)$-bit quantum circuit $K_{u,2}$ of size $O(2t+1)$ based on $\mathscr{G}_u$ such that the quantum gates determined by them are G_1 and G_2, respectively. Now let K_a be an $(l_0+(2t+1)l)$-bit quantum circuit obtained by decomposing each G_1 in K_1 into O(1) gates in $\mathscr{G}_u$. Then the size of K_a is $O(2t+1)$. Similarly, from K_2 we can obtain an $(l_0+(2t+1)l)$-bit quantum circuit K_b of size $O(2t+1)$. Thus, $K=(K_b \circ K_a)^t$ is a quantum circuit of size $O(t^2)$ that t-simulates M. □

Let V be a 2^n-dimensional transformation and $A=\{j_1,\ldots,j_n\}$ a set of integers with $j_1<\cdots<j_n$. Then we say that a multi-track QTM M *carries out* V (*with accuracy* ε) *on the cell-set* A of the ith track, if M carries out the following algorithm.

1. For $m=1,\ldots,n$, the QTM M transfers the symbol written on each cell j_m of the ith track to cell m of an empty extra track. Henceforth, let this extra track be the kth track.
2. M carries out V (with accuracy ε) on the kth track.
3. M transfers the symbol written on each cell m of the kth track to cell j_m of the ith track.

Now, we give a proof of the existence of a universal QTM that efficiently simulates every multi-tape QTM in P**C** with arbitrarily given accuracy.

Theorem 4.4. *There is a two-way SNQTM M_u such that for any positive integer t, positive number ε, multi-tape QTM M in* P**C**, *and input string x of M, the QTM M_u on input (t,ε,c_M,x) simulates M for t steps with accuracy ε and slowdown of at most a polynomial in t and $1/\varepsilon$, where c_M is the code of M.*

Proof. For simplicity, we consider the case where M is a single tape QTM. When M is a multi-tape QTM, we can prove this theorem similar to the proof shown in the following by using a quantum circuit given in Appendix A instead of a quantum circuit given in the proof of Theorem 4.3.

In what follows, we shall construct a multi-track QTM $M_u=(Q_u,\Sigma_u,\delta_u)$ that simulates M for t steps with accuracy ε for any given t,ε, and M. The input of M_u consists of the input x of M, the desired number of steps t, the desired accuracy ε, and the code of M. Henceforth, we fix t, ε, and M. In this proof, we shall use the same notations as in the proof of Theorem 4.3. By the proof of Theorem 4.3, there is a quantum circuit $K_{\mathscr{G}}=(K_2\circ K_1)^t$ based on $\mathscr{G}=\{G_1,G_2\}$ that t-simulates M. The QTM M_u has six tracks and the alphabet of each track contains 0 and 1. The first track of M_u will be used to represent the computation of $K_{\mathscr{G}}$ approximately. The second and the third track of M_u will be, respectively, used to record an approximate code of G_1 and $\mathrm{Acc}(\varepsilon)$. The fourth track of M_u will contain counters C_0 and C_1. The values of C_0 and C_1 count the numbers of subcircuits of the form $K_2\circ K_1$ and G_1 in $K_{\mathscr{G}}$ which have been carried out so far, respectively. The fifth track of M_u is used to record the input of M_u. The sixth track is used as a working track.

Let $k=2^{l_0+3l}$ and $\varepsilon'\leqslant\varepsilon/16t(2t-1)(10\sqrt{k})^k$. The QTM M_u carries out $K_{\mathscr{G}}$ with accuracy ε after a preparation. The preparation is to compute the ε'-approximate code $c(G_1)$ of G_1 from c_M and write $c(G_1)$ on the second track of M_u, to write $\mathrm{Acc}(\varepsilon)$ on the third track of M_u, and to write the $(l_0+(2t+1)l)$-bit string $x'=q_0T(-t)\bar{0}\cdots T(0)\bar{1}\cdots T(t)\bar{0}$ corresponding to the initial configuration $|q_0,\mathrm{tape}[x],0\rangle$ of M on the first track of M_u, where the string x' represents the input of $K_{\mathscr{G}}$. Given c_M and a sufficiently small positive number $c\varepsilon'$, where c depends only on k, i.e., $|Q|$ and $|\Sigma|$, but is independent of t and ε, we can compute the ε'-approximate code of the (k-dimensional) S-matrix of G_1 in polynomial time in $\log t$ and $\log 1/\varepsilon$ by using the definition of G_1 given in the

proof of Theorem 4.3 and the orthonormalization of Schmidt. According to the synchronization theorem there is an SNQTM that carries out the preparation given above in time polynomial in the length of the input of M_u, i.e., a polynomial in t and $\log 1/\varepsilon$.

The algorithm for carrying out $K_{\mathscr{G}}$ with accuracy ε is as follows. At first, the values of counters C_0 and C_1 are zero.

Step 1: Carry out steps 2–4 until the value of counter C_0 comes to t.

Step 2: Carry out steps 2.1 and 2.2 until the value of counter C_1 comes to a multiple of $2t-1$.

Step 2.1: When the value of C_1 is $i \pmod{2t-1}$, carry out the 2^{l_0+3l}-dimensional transformation G_1 with accuracy $\varepsilon/4t(2t-1)$ on the cell-set $[0, l_0-1]_{\mathbf{Z}} \cup [l_0+il, l_0+il+l-1]_{\mathbf{Z}}$ of the first track.

Step 2.2: Increase the value of counter C_1 by one.

Step 3: Carry out the $2^{l_0+(2t+1)l}$-dimensional transformation G_2 on the cell-set $[0, l_0+(2t+1)l-1]_{\mathbf{Z}}$ of the first track.

Step 4: Increase the value of counter C_0 by one.

Since G_2 is a reversible Boolean gate which transforms all $s_i = \bar{2}$ to $s_i = \bar{1}$, we can construct an SNRTM M_2 that carries out step 3 in time polynomial in t by the synchronization theorem. We can construct an SNQTM M_1 that carries out G_1 with accuracy $\varepsilon/4t(2t-1)$ in time polynomial in $4t(2t-1)/\varepsilon$ and $|c(G_1)| = \mathrm{O}(k^2 \log 1/\varepsilon')$ by the unitary theorem. Moreover, we can construct SNQTMs to run counters C_0 and C_1 by the looping lemma. The QTM M_u can be constructed by applying the addition of tracks, the permutation of tracks, and the dovetailing lemma to the above SNQTMs and the SNQTM that carries out the preparation.

It is clearly verified that the operation of steps 2–4 corresponds to carrying out $K_2 \circ K_1$ with accuracy $\varepsilon/4t$ by the proof of Theorem 4.3. Thus if the value of counter C_0 comes to t, then M_u carries out the quantum gate determined by $K_{\mathscr{G}}$ with accuracy ε (It is known that if $\||\phi\rangle - |\psi\rangle\| \leqslant \varepsilon$ for two state vectors $|\phi\rangle, |\psi\rangle$, the total variation distance between the probability distributions determined by them is at most 4ε [8]). Now let $q_{0,u}$ and $q_{f,u}$ be the initial and final processor configurations of M_u. Let the encoding $e : \mathscr{C}(Q, \Sigma) \to \mathscr{C}(Q_u, \Sigma_u)$ be a function satisfying $e(q_0, \mathrm{tape}[x], 0) = (q_{0,u}, \mathrm{tape}[B, B, B, B, \langle t, \varepsilon, c_M, x\rangle], 0)$. Let $x = x_0 x_1 \cdots x_{|x|-1}$. Let the decoding $d : \mathscr{C}(Q_u, \Sigma_u) \to \mathscr{C}(Q, \Sigma)$ be a function satisfying the following condition. For any $(T^1, \ldots, T^6) \in \Sigma_u^{\#}$ satisfying

$$T^1 = \mathrm{tape}[qT(-t)\bar{0} \cdots T(\xi)\bar{1} \cdots T(t)\bar{0}],$$
$$T^2 = \mathrm{tape}[c(G_1)], \quad T^3 = \mathrm{tape}[\mathrm{Acc}(\varepsilon)], \quad T^5 = \mathrm{tape}[\langle t, \varepsilon, c_M, x\rangle],$$

the equation $d(q_{f,u}, (T^1, \ldots, T^6), 0) = (q, T', \xi)$ holds; the tape configuration T' of M satisfies

$$T'(i) = \begin{cases} T(i) & \text{if } i \in [-t, t]_{\mathbf{Z}}, \\ x_i & \text{if } t < |x| - 1 \text{ and } i \in [t+1, |x|-1]_{\mathbf{Z}}, \\ B & \text{otherwise.} \end{cases}$$

It is easy to see that M_u simulates M for t steps with accuracy ε and the computation time is bounded by a polynomial in t and $1/\varepsilon$. □

Remark 1. Any pair of QTMs dovetailed in the proof of Theorem 4.4 can be constructed so that it can satisfy the dovetailing conditions (cf. Lemma 3.3). Thus, stationarity is preserved by dovetailing them. Indeed, the fact that all QTMs constructed in the proof of Theorem 4.4 satisfy conditions (i) and (ii) of the dovetailing conditions can be verified from the statements of the synchronization theorem, the looping lemma, and the unitary theorem.

Remark 2. Recently, Kitaev [15] and Solovay [22] independently proved that there is a quantum algorithm which decomposes a given n-bit quantum gate into $\mathrm{poly}(2^n, \log 1/\varepsilon)$ elementary gates with accuracy ε. Applying this result to the proof of Theorem 4.4, we can replace a polynomial in n and $1/\varepsilon$ in the statement of Theorem 4.4 by a polynomial in n and $\log 1/\varepsilon$.

5. Computational complexity of uniform QCFs and QTMs

A *quantum circuit family* (*QCF*) is an infinite sequence $\mathscr{K} = \{\mathbf{K}_n\}_{n\geqslant 1}$ such that $\mathbf{K}_n$ is an n-input ($f(n)$-output $g(n)$-bit) quantum circuit. A QCF $\mathscr{K}$ is said to be *based on* a set $\mathscr{G}$ of quantum gates if every $\mathbf{K}_n$ in $\mathscr{K}$ is based on $\mathscr{G}$. A QCF $\mathscr{K}$ is said to be of *size* s based on $\mathscr{G}$ if the size of $\mathbf{K}_n$ for $\mathscr{G}$ is $s(n)$ for a function s from $\mathbf{N}$ to $\mathbf{N}$. If s is a polynomial, it is called a *polynomial size* QCF *based on* $\mathscr{G}$. Moreover, if $\mathscr{G} = \mathscr{G}_u$, then $\mathscr{K}$ is merely called a *polynomial size* QCF. For any quantum circuit K, the quantum gate $G(K)$ determined by K is decomposable by $\mathscr{G}_u$ from Theorem 4.1. Thus, in what follows, we consider only quantum circuits based on subsets of $\mathscr{G}_u$.

First we define the code of a quantum circuit based on $\mathscr{G}_{\mathscr{R}}$. Let $K = (G_m, \pi_m), \ldots, (G_1, \pi_1)$ be a quantum circuit based on $\mathscr{G}_{\mathscr{R}}$. Then the $\mathscr{G}_{\mathscr{R}}$-*code* of K, denoted by $c_r(K)$, is defined to be the list of finite sequences of natural numbers $\langle e_r(G_1), \ldots, e_r(G_m)\rangle$, where for $j \in [1,m]_{\mathbf{Z}}$ we have

$$e_r(G_j) = \begin{cases} \langle i, \pi_j(1)\rangle & \text{if } G_j = R_{i,\mathscr{R}}, \\ \langle 4, \pi_j(1), \pi_j(2)\rangle & \text{if } G_j = M_2(N). \end{cases}$$

Let $\mathbf{K}$ be a k-input m-output n-bit quantum circuit $\mathbf{K} = (K, \Lambda_1, \Lambda_2, S)$ based on $\mathscr{G}_{\mathscr{R}}$, where $[1,n]_{\mathbf{Z}} \backslash \Lambda_1 = \{i_1, \ldots, i_{n-k}\}$ and $\Lambda_2 = \{j_1, \ldots, j_m\}$. Then the $\mathscr{G}_{\mathscr{R}}$-code of $\mathbf{K}$, denoted by $c_r(\mathbf{K})$, is defined to be the list of finite sequences of natural numbers,

$$c_r(\mathbf{K}) = \langle\langle\langle i_1, S(i_1)\rangle, \ldots, \langle i_{n-k}, S(i_{n-k})\rangle\rangle, c_r(K), \langle j_1, \ldots, j_m\rangle\rangle.$$

Given a QCF $\mathscr{K} = \{\mathbf{K}_n\}_{n\geqslant 1}$ of size s based on $\mathscr{G}_{\mathscr{R}}$, the QCF $\mathscr{K}$ is said to be $\mathscr{G}_{\mathscr{R}}$-*uniform* if the function $1^n \mapsto c_r(\mathbf{K}_n)$ is computable by a DTM in time $p(s(n))$ for some polynomial p.

The $\mathscr{G}_{\mathscr{R}}$-uniform QCFs are a subclass of the general uniform QCFs defined below. As Shor pointed out in [20], the entries of the S-matrices of quantum gates in a uniform QCF must be polynomial-time computable numbers. Actually, Shor required that the entries should be computable in the sense that the first $\log n$ bits are computable in time polynomial in n, while we require that the first n bits of a computable number are computable in time polynomial in n. This requirement is consistent with the restriction of transition amplitudes of QTMs given by Bernstein-Vazirani [8]. From the above, we require that the entries of elementary gates be restricted to be polynomially computable ones. Thus it is natural to assume that any uniform QCF can be decomposed into the elementary gates in

$$\mathscr{G}_{\mathrm{PC}} = \{R_{1,\theta},\ R_{2,\theta},\ R_{3,\theta},\ M_2(N) \mid \theta \in \mathrm{PC} \cap [0, 2\pi]\}.$$

According to the above, we shall give the formal definition of uniform QCFs for QCFs based on the set $\mathscr{G}_{\mathrm{PC}}$ instead of the universal set $\mathscr{G}_u$. For any $\theta \in \mathrm{PC}$, let $c(\theta)$ be the code of θ. Let $K = (G_m, \pi_m), \ldots, (G_1, \pi_1)$ be a quantum circuit based on $\mathscr{G}_{\mathrm{PC}}$. Then the *code* of K, denoted by $c(K)$, is defined to be the list of finite sequences of natural numbers $\langle e(G_1), \ldots, e(G_m)\rangle$, where for $j \in [1, m]_{\mathbf{Z}}$ we have

$$e(G_j) = \begin{cases} \langle\langle i, c(\theta)\rangle, \pi_j(1)\rangle & \text{if } G_j = R_{i,\theta}, \\ \langle 4, \pi_j(1), \pi_j(2)\rangle & \text{if } G_j = M_2(N). \end{cases}$$

Similar to the case of the code of a quantum circuit on $\mathscr{G}_{\mathscr{R}}$, we can define the code of a k-input m-output n-bit quantum circuit $\mathbf{K}$ based on $\mathscr{G}_{\mathrm{PC}}$. Given a QCF $\mathscr{K} = \{\mathbf{K}_n\}_{n \geqslant 1}$ of size s based on $\mathscr{G}_{\mathrm{PC}}$, the QCF $\mathscr{K}$ is said to be *uniform* if the function $1^n \mapsto c(\mathbf{K}_n)$ is computable by a DTM in time $p(s(n))$ for some polynomial p. It is easy to see that a $\mathscr{G}_{\mathscr{R}}$-uniform QCF is uniform.

As is well-known, the discrete Fourier transform

$$|a\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{c=0}^{2^n-1} \exp\left(\frac{2\pi \mathrm{i} ac}{2^n}\right) |c\rangle,$$

where $a = 0, \ldots, 2^n - 1$, plays an important role in Shor's algorithm [13, 20]. It is easy to see that the polynomial size QCF $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ that performs the discrete Fourier transform is such that on input 1^n the code of K_n

$$c(K_n) = \langle c^1(A), c^{12}(B_1), c^2(A), \ldots, c^{1n}(B_{n-1}), \ldots, c^{(n-1)n}(B_1), c^n(A)\rangle$$

can be computed by a polynomial-time bounded DTM,
where

$$c^j(A) = \langle\langle 3, c(\pi)\rangle, j\rangle, \langle\langle 1, c(\pi/4)\rangle, j\rangle$$

and

$$\begin{aligned} c^{ij}(B_k) = {} & \langle\langle 3, c(\pi/2^{k+1})\rangle, i\rangle, \langle\langle 2, c(-\pi/2^{k+2})\rangle, j\rangle, \langle\langle 3, c(\pi/2^{k+2})\rangle, j\rangle, \langle 4, i, j\rangle, \\ & \langle\langle 2, c(\pi/2^{k+2})\rangle, j\rangle, \langle\langle 3, c(-\pi/2^{k+2})\rangle, j\rangle, \langle 4, i, j\rangle. \end{aligned}$$

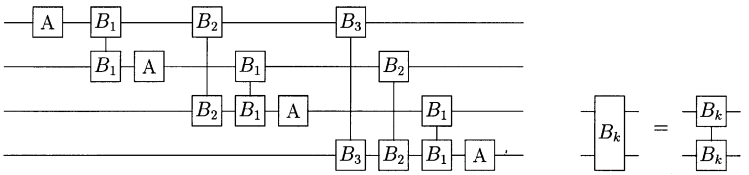

Fig. 3. The quantum circuit K_4. In this figure, $A = R_{1,\pi/4} \cdot R_{3,\pi}$, and B_k is the 2-bit quantum gate determined by the quantum circuit $K_{B,k}$ (Fig. 4) based on $\mathscr{G}_u$. The S-matrix of B_k is $\mathrm{diag}(1,1,1,\exp(\mathrm{i}\pi/2^k))$, where $\mathrm{diag}(a_1,\ldots,a_n)$ is an n-dimensional diagonal matrix whose diagonal components are $a_1,\ldots,a_n$ in this order.

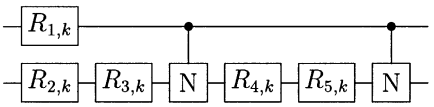

Fig. 4. The quantum circuit $K_{B,k}$. In this figure, $R_{1,k} = R_{3,\pi/2^{k+1}}$, $R_{2,k} = R_{2,-\pi/2^{k+2}}$, $R_{3,k} = R_{3,\pi/2^{k+2}}$, $R_{4,k} = R_{2,\pi/2^{k+2}}$, and $R_{5,k} = R_{3,-\pi/2^{k+2}}$.

Thus, $\mathscr{K}$ is uniform. For example, K_4 is the quantum circuit illustrated in Figs. 3 and 4.

A formal definition of a simulation of a QCF by a QTM is given as follows. Let M be a multi-track QTM such that the alphabet of each track contains 0 and 1. We say that M *carries out an n-input k-bit quantum circuit* $\mathbf{K} = (K, \Lambda_1, \Lambda_2, S)$ if for every n-bit string x, the output state of M for input state $|q_0, \mathrm{tape}[x], 0\rangle$ is

$$\sum_{y \in \{0,1\}^k} |q_f, \mathrm{tape}[x, y], 0\rangle \langle y|G(K)|u(x, \mathbf{K})\rangle.$$

For any function $f : \mathbf{N} \to \mathbf{N}$, we say that M *simulates* a QCF $\mathscr{K} = \{\mathbf{K}_n\}_{n \geqslant 1}$ in time $f(n)$, if on every n-bit input string, M carries out $\mathbf{K}_n$ and the computation time is $f(n)$. Then the following lemma holds.

Lemma 5.1. *For any $\mathscr{G}_{\mathscr{R}}$-uniform QCF $\mathscr{K}$, there exist a polynomial p and an SNQTM M which simulates $\mathscr{K}$ in time $p(s(n))$, where s is the size of $\mathscr{K}$.*

Proof. Let $\mathscr{K} = \{\mathbf{K}_n\}_{n \geqslant 1}$ be a $\mathscr{G}_{\mathscr{R}}$-uniform QCF of size s, and let $\mathbf{K}_n = (K_n, \Lambda_{1,n}, \Lambda_{2,n}, S_n)$. First, we show that there exists a multi-track QTM M that carries out the following steps for input state $|q_0, \mathrm{tape}[x], 0\rangle$. Throughout this proof, we assume that the length of x is n.

Step 1: Write 1^n on the third track, $c_r(\mathbf{K}_n)$ on the fourth track, and $u = u(x, \mathbf{K}_n)$ on the second track.

Step 2: Iterate the following steps 3 and 4 for $l = 1$ to $s(n)$, where step 4 refers to step 4.1 or 4.2.

Step 3: On the fourth track, scan the lth component $\langle h, i\rangle$ or $\langle h, i, j\rangle$ of $c_r(K_n)$, where $h \in [1,4]_{\mathbf{Z}}$ and $i, j \in [1, |\Lambda_{1,n}| + |\mathrm{domain}(S_n)|]_{\mathbf{Z}}$. That is, h is the index in $\mathscr{G}_{\mathscr{R}}$ of the lth

quantum gate constructing K_n, and i (and j) is the bit number of the wire connected to the lth gate.

Step 4.1. When $h=4$, if $i<j$, then carry out the unitary transformation $|x,y\rangle \mapsto |x,x+y \bmod 2\rangle$ on the cell-set $\{i,j\}$ of the second track. If $i>j$, then carry out the unitary transformation $|x,y\rangle \mapsto |x+y \bmod 2, y\rangle$ on the cell-set $\{i,j\}$ of the second track.

Step 4.2. When $h \neq 4$, carry out the transformation $R_{h,\mathscr{R}}$ on the cell-set $\{i\}$ of the second track.

Step 5. Empty the fourth and the third tracks.

Since $\mathscr{K}$ is a $\mathscr{G}_{\mathscr{R}}$-uniform QCF of size s, there is a DTM that computes the function $1^n \mapsto c_r(\mathbf{K}_n)$ in time polynomial of $s(n)$. Thus, we construct an SNQTM that carries out step 1 in time polynomial of $s(n)$ by using the synchronization theorem, the addition and the permutation of tracks, and the dovetailing lemma. Moreover, using the synchronization theorem we can construct SNQTMs for steps 3 and 5 that run in time polynomial of $s(n)$. For each unitary transformation in step 4, we can construct an SNQTM that carries it out using the completion lemma. For example, an SNQTM that carries out the unitary transformation $R_{1,\mathscr{R}}$ is such that the quantum transition function δ satisfies

$$\delta(q_0,0,q_1,0,-1) = \delta(q_0,1,q_1,1,-1) = \cos\mathscr{R},$$
$$-\delta(q_0,0,q_1,1,-1) = \delta(q_0,1,q_1,0,-1) = \sin\mathscr{R}, \quad \delta(q_1,B,q_f,B,1) = 1,$$
$$\delta(q_f,a,q_0,a,1) = 1 \quad (a \in \{B,0,1\}).$$

Similarly, we can also construct SNQTMs that carry out the other unitary transformations. Now we can construct an SNQTM that accomplishes step 4 by applying the addition and the permutation of tracks, the branching lemma, and the synchronization theorem to SNQTMs that carry out their unitary transformations. An SNQTM which carries out step 4.1 or 4.2 according to h in step 3 can be constructed by the branching lemma, the addition and the permutation of tracks, and the dovetailing lemma. We can construct an SNQTM that carries out step 2 by the looping lemma. Finally, we can construct the desired QTM M by applying the addition and the permutation of tracks, and the dovetailing lemma to SNQTMs that carries out steps 1, 2, and 5. Each dovetailed SNQTM can be constructed so that the dovetailing conditions can be satisfied.

It is easy to see that M carries out K_n and the computation time of M is a polynomial of $s(n)$. From the above, M simulates $\mathscr{K}$ in time polynomial of $s(n)$. □

Using Theorem 4.3 and Lemma 5.1, we investigate the detailed relationships among complexity classes between QTMs and QCFs. We shall now define classes of language efficiently recognized by QTMs or QCFs implementing Monte Carlo, Las Vegas, and exact algorithms, that is, quantum analogues of the probabilistic complexity classes **BPP**, **ZPP**, and **P**.

We say that a QTM M *accepts* (or *rejects*) $x \in \{0,1\}^*$ *with probability* p if the output state $|\psi\rangle$ of M for input state $|q_0, \mathrm{tape}[x], 0\rangle$ satisfies

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

We say that M *recognizes* a language L with probability at least p if M accepts x with probability at least p for any $x \in L$ and rejects x with probability 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 $0 < \eta \leqslant 1 - p$ such that M recognizes L with probability at least $p + \eta$. Let A be a subset of $\mathbf{C}$. A language L is in $\mathbf{BQP}_A$ (or $\mathbf{EQP}_A$) if there is a polynomial-time bounded QTM $M = (Q, \Sigma, \delta)$ that recognizes L with probability uniformly larger than $\frac{1}{2}$ (or with probability 1) and $\mathrm{range}(\delta) \subseteq A$. A language L is in $\mathbf{ZQP}_A$ if there is a polynomial-time bounded QTM $M = (Q, \Sigma, \delta)$ satisfying the following conditions.

(1) M recognizes L with probability uniformly larger than $\frac{1}{2}$.

(2) $\mathrm{range}(\delta) \subseteq A$.

(3) If M accepts (rejects) input x with a positive probability, M rejects (accepts) x with probability 0.

From these definitions, we obviously have $\mathbf{EQP}_A \subseteq \mathbf{ZQP}_A \subseteq \mathbf{BQP}_A$. In what follows, when $A = \mathrm{P}\mathbf{C}$, we denote $\mathbf{BQP}_A$, $\mathbf{EQP}_A$, and $\mathbf{ZQP}_A$ by $\mathbf{BQP}$, $\mathbf{EQP}$, and $\mathbf{ZQP}$, respectively.

Let M be an SNQTM that recognizes a language L with probability uniformly larger than $\frac{1}{2}$ in time $t(n)$, where n is the length of the input of M. Then we can recognize L with probability uniformly larger than $1 - \varepsilon$ by iterating the computation of M on the input $k = \mathrm{O}(\log 1/\varepsilon)$ times (ε is a positive number independent of the input) and calculating the majority of the k answers. Moreover, Bennett et al. [6] showed that an SNQTM that recognizes L with probability uniformly larger than $1 - \varepsilon$ in time $ct(n)$ (here, c is a polynomial in $\log 1/\varepsilon$ and independent of n) can be constructed. This fact means that the classes **BQP** and **ZQP** we have now defined are identical with **BQP** and **ZQP** defined in [8, 9].

A definition of recognition of languages by quantum circuits is given as follows. Let K be an n-input 2-output quantum circuit and $x \in \{0,1\}^n$. When $\rho^K(01|x) = p$ (or $\rho^K(00|x) = p$), we say that K *accepts* (or *rejects*) x *with probability* p. For any language $L_n \subseteq \{0,1\}^n$, we say that K *recognizes* L_n with probability at least p if K accepts x with probability at least p for any $x \in L_n$ and K rejects x with probability at least p for any $x \notin L_n$.

We need to consider circuit families in order to recognize languages including strings with different lengths. In what follows, we write $L_n = L \cap \{0,1\}^n$ for any language L. We say that a *QCF* $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ *recognizes* a language L *with probability* at least p if K_n recognizes L_n with probability at least p for any $n \in \mathbf{N}$. We say that $\mathscr{K}$ recognizes a language L *with probability uniformly larger than* p if there is a constant $0 < \eta \leqslant 1 - p$ such that K_n recognizes L_n with probability at least $p + \eta$ for any n. We say that a language L has *bounded-error* (*or exact*) *uniform polynomial size quantum*

circuits, in symbols $L \in \mathbf{BUPQC}$ (or $L \in \mathbf{EUPQC}$), if there is a uniform polynomial size QCF $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ that recognizes L with probability uniformly larger than $\frac{1}{2}$ (with probability 1). Moreover, we say that a language L *has zero-error uniform polynomial size quantum circuits*, in symbols $L \in \mathbf{ZUPQC}$, if there is a uniform polynomial size QCF $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ recognizing with probability uniformly larger than $\frac{1}{2}$ and satisfying $\rho^{K_{|x|}}(00|x) = 0$ or $\rho^{K_{|x|}}(01|x) = 0$ for any $x \in \{0,1\}^*$. From these definitions we obviously have $\mathbf{EUPQC} \subseteq \mathbf{ZUPQC} \subseteq \mathbf{BUPQC}$.

As is well-known, $\mathbf{P}$ is identical with the class of languages that have uniform polynomial size Boolean circuits [18]. In this paper, uniform Boolean circuit families mean polynomial time uniform ones, while in computational complexity theory, more restricted families have been investigated and some of them are also equivalent to polynomial time bounded DTMs. The following identical relation holds between complexity classes of QTMs and QCFs. This relation means that QTMs and uniform QCFs are equivalent as probabilistic machines implementing Monte Carlo algorithms as suggested by Shor [20].

Theorem 5.2. $\mathbf{BQP} = \mathbf{BUPQC}$.

Proof. Let $L \in \mathbf{BQP}$. Then without loss of generality, we can assume that there is a QTM $M = (Q, \Sigma, \delta)$ that recognizes L with probability uniformly larger than $\frac{1}{2}$ in time $p(n)$, where p denotes a polynomial (See Remark 2). This QTM M can be $p(n)$-simulated by a quantum circuit K_n of size $\mathrm{O}(p^2(n))$ constructed as the proof of Theorem 4.3. The quantum gate $G(K_n)$ can be decomposed into two sorts of quantum gates G_1 and G_2 as given in the proof of Theorem 4.3, and the array of G_1 and G_2 in K_n can be computed in time polynomial in n. Moreover, $\mathrm{range}(\delta) \subseteq \mathrm{P}\mathbf{C}$ by the definition of $\mathbf{BQP}$, so that from the S-matrix of G_1 we can compute the array of elementary gates in $\mathscr{G}_{\mathrm{P}\mathbf{C}}$ decomposing G_1 in time independent of n. Obviously, G_2 can be decomposed into elementary gates in $\mathscr{G}_{\mathrm{P}\mathbf{C}}$ in time polynomial in n. Therefore, there is a DTM which on input 1^n produces the code $c(K_n)$ in time polynomial in n. Thus, $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ is uniform. From the above, $L \in \mathbf{BUPQC}$.

Conversely, suppose $L \in \mathbf{BUPQC}$. Then, for all $n \in \mathbf{N}$ there is a quantum circuit K_n of size $p(n)$ based on $\mathscr{G}_{\mathrm{P}\mathbf{C}}$ which recognizes L_n with probability $\frac{1}{2} + \eta$, where p is a polynomial and $0 < \eta \leqslant \frac{1}{2}$ is a constant independent of n. Moreover, there is a DTM M_0 that computes the function $1^n \mapsto c(K_n)$ in time polynomial in n. Assume that the length of a bit string x is n. Let $c(K_n) = \langle e(G_1), \ldots, e(G_k), \ldots, e(G_{p(n)}) \rangle$, where $e(G_k) = \langle \langle i, c(\theta) \rangle, \pi_k(1) \rangle$ if $G_k = R_{i,\theta}$ and $e(G_k) = \langle 4, \pi_k(1), \pi_k(2) \rangle$ if $G_k = M_2(N)$. Now we compute the $\mathscr{G}_{\mathscr{R}}$-code $c_r(K_{n,\varepsilon})$ of a quantum circuit $K_{n,\varepsilon}$ based on $\mathscr{G}_{\mathscr{R}}$ such that $\|G(K_n) - G(K_{n,\varepsilon})\| \leqslant \varepsilon$ from $c(K_n)$ as follows. For each $k = 1, \ldots, p(n)$, from the component $\langle i, c(\theta) \rangle$ of $c(K_n)$ representing $G_k = R_{i,\theta}$ in K_n, we compute an integer m such that $\|R_{i,\theta} - R_{i,\mathscr{R}}^m\| \leqslant \varepsilon / p(n)$ by Lemma 4.2, and replace the component $\langle \langle i, c(\theta) \rangle, \pi_k(1) \rangle$ in $c(K_n)$ by $\underbrace{\langle i, \pi_k(1) \rangle, \ldots, \langle i, \pi_k(1) \rangle}_{m}$. It is easy to see that the computation time of this algorithm is at most a polynomial in n and $\log 1/\varepsilon$. Now let $\varepsilon \leqslant \eta/2$. Then the QCF

$\mathscr{K}_\varepsilon = \{K_{n,\varepsilon}\}_{n\geqslant 1}$ based on $\mathscr{G}_\mathscr{R}$ recognizes L with probability at least $\frac{1}{2}+\eta/2$. Next, we consider the $\mathscr{G}_\mathscr{R}$-size of $K_{n,\varepsilon}$. For each 1-bit quantum gate $R_{j,\theta}$ ($j=1,2,3,\ \theta\in[0,2\pi]$) constructing K_n, the positive integer m determined by Lemma 4.2 such that $\|R_{j,\theta} - R^m_{j,\mathscr{R}}\| \leqslant \varepsilon/p(n)$ is at most $\mathrm{O}(p^4(n)/\varepsilon^4)$. Thus, the $\mathscr{G}_\mathscr{R}$-size $s(n)$ of $K_{n,\varepsilon}$ is at most $s(n)=\mathrm{O}(p^4(n)/(\frac{\eta}{2})^4)\times p(n)=\mathrm{O}(p^5(n))$. Therefore, $\mathscr{K}_\varepsilon$ is a $\mathscr{G}_\mathscr{R}$-uniform QCF of size $s(n)$. Applying Lemma 5.1 to $\mathscr{K}_\varepsilon$, given as input an n-bit string there is a QTM $M=(Q,\Sigma,\delta)$ that carries out $K_{n,\varepsilon}$ in time $\mathrm{O}(q(s(n)))$, where q is a polynomial. From the proof of Lemma 5.1, it is easy to see that range$(\delta)\subseteq \mathbf{PC}$. Therefore we conclude $L\in\mathbf{BQP}$. □

Remark 1. Using the proof of $\mathbf{BUPQC}\subseteq\mathbf{BQP}$ in Theorem 5.2, we can show the existence of a polynomial-time bounded universal QTM which simulates any given uniform QCF with any accuracy.

Remark 2. Any polynomial-time bounded QTM M can be simulated by a two-tape QTM M' whose computation time is exactly a polynomial in the length of the input, using time constructible functions to count the number of steps, as follows: (1) M' writes $1^{p(n)}$ on the second tape, where n is the length of the input, $p(n)$ is a time constructible polynomial, and the computation time of M is bounded by $p(n)$; (2) Every time when M' carries out one step of M on the first tape, M' changes 1 to B on the second tape; (3) When M' completes the computation of M, the first tape of M does not change the contents of the first tape any more, while M' changes 1 to B on the second tape; (4) if the second tape scans B, then M' halts.

The following theorem can be verified by a proof similar to that of $\mathbf{BQP}\subseteq\mathbf{BUPQC}$ in Theorem 5.2, and means that QTMs are not more powerful than uniform QCFs as probabilistic machines implementing exact or Las Vegas algorithms.

Theorem 5.3. (1) $\mathbf{EQP}\subseteq\mathbf{EUPQC}$.
(2) $\mathbf{ZQP}\subseteq\mathbf{ZUPQC}$.

It is open whether the inclusion relations in Theorem 5.3 are proper or not. In the proof of $\mathbf{BUPQC}\subseteq\mathbf{BQP}$ in Theorem 5.2, we are allowed to replace quantum gates with some additional errors, while an analogous argument does not work in Theorem 5.3.

It has been considered that Shor's factoring algorithm is a Las Vegas quantum algorithm. We shall show this fact by proving that a certain language corresponding to the factoring problem is not only in **ZUPQC** but also in **ZQP**. The factoring problem is polynomial-time Turing reducible to the language FACTOR $=\{\langle N,k\rangle \mid N$ has a non-trivial prime factor larger than $k\}$ and the class of problems solved by Las Vegas algorithms is closed under polynomial-time Turing reductions. On the other hand, as suggested by Theorem 5.3, any language in **ZQP** can be recognized most typically by a Las Vegas quantum algorithm. Thus, in order to verify that Shor's factoring

algorithm is a Las Vegas quantum algorithm, it is sufficient to show that FACTOR is in **ZQP**.

Theorem 5.4. FACTOR $\in$ **ZQP**.

Proof. Let $\langle N,k\rangle$ be an input of the algorithm to be constructed. In the following algorithm that recognizes FACTOR, we use a Las Vegas primality testing algorithm (for example, such an algorithm can be constructed by the algorithm of Solovay and Strassen [23] and the algorithm of Adleman and Huang [2]) and Shor's factoring algorithm [20]. At first, let LIST $=\{N\}$.

Step 1: Carry out steps 2–4 while the greatest number in LIST is larger than 1.

Step 2: For the greatest number N' in LIST, check whether N' is prime or not by the Las Vegas primality testing algorithm. If N' is judged to be prime, then go to step 3. If N' is judged to be composite, go to step 4. Otherwise, output a special mark '?' and end.

Step 3: If $N'>k$ then output 1 and end. Otherwise, output 0 and end.

Step 4: On input N', carry out Shor's factoring algorithm. If a factor p is found, then replace N' in LIST by p and N'/p, and go to step 2. If no factor is found, output '?' and end.

Step 2 can be implemented by a polynomial-time bounded SNQTM, because **ZPP** is included in **ZQP**. Step 3 can also be implemented by a polynomial-time bounded SNQTM using the synchronization theorem. In step 4 we can divide Shor's factoring algorithm into three processes: (1) a process that produces a factor candidate of N'; (2) a process that iterates process (1) $j=\mathrm{O}((\log N)^2)$ times in order to obtain j factor candidates; (3) a process that produces a true factor if the factor exists in the j candidates, and otherwise produces '?'. Note that process (1) also includes a deterministic algorithm performed efficiently for the case where N' is an even number or a prime power. We have shown that the discrete Fourier transform can be implemented by a uniform polynomial size QCF in this section. Using a similar way, we can make sure that process (1) can be carried out by a uniform polynomial size QCF $\mathscr{K}$. Let $\varepsilon>0$ be a small constant independent of N. Similar to the proof of Theorem 5.2, for any K_n in $\mathscr{K}$, the $\mathscr{G}_{\mathscr{R}}$-code of a quantum circuit $K_{n,\varepsilon}$ based on $\mathscr{G}_{\mathscr{R}}$ such that $\|G(K_n)-G(K_{n,\varepsilon})\|\leqslant\varepsilon$ can be computed in time polynomial in n. Thus $\mathscr{K}_\varepsilon=\{K_{n,\varepsilon}\}_{n\geqslant 1}$ is $\mathscr{G}_{\mathscr{R}}$-uniform. We can construct an SNQTM M_1 that carries out $\mathscr{K}_\varepsilon$ by Lemma 5.1. An SNQTM M_2 which carries out process (2) can be constructed by inserting M_1 into a looping machine j times. We can construct an SNQTM M_3 that carries out process (3) by the synchronization theorem, and construct an SNQTM M implementing Shor's factoring algorithm by applying the addition and the permutation of tracks and the dovetailing lemma to M_2 and M_3.

In step 4 the probability that produces '?' is less than $1/N$, since by one round of process (1) we get a true factor with probability at least $\Omega(1/\log N)$ and we repeat process (1) $\mathrm{O}((\log N)^2)$ rounds to reduce the probability that produces '?' up to less than $1/N$. In step 2, by iterating the Las Vegas primality testing a polynomial number

of times we can make the probability that produces '?' less than $1/N$. Moreover, steps 2–4 will be carried out at most $\log N$ times. Thus, the above algorithm produces '?' with probability at most $\eta < \frac{1}{2}$, where η is independent of the input. Now it is easy to conclude that FACTOR $\in$ **ZQP**. □

Remark. From Theorems 5.3 and 5.4 it follows that FACTOR $\in$ **ZUPQC**. However, this fact can be verified in a more straightforward argument. In fact, we have verified that Shor's factoring algorithm (step 4) in the algorithm of the proof of Theorem 5.4 can be implemented by a uniform polynomial size QCF. On the other hand, the other part of the algorithm can be written as a classical probabilistic algorithm. Coin flips can be implemented by Hadamard gates, and the classical deterministic part can be implemented by Toffoli gates. These two sorts of gates can be decomposed into O(1) elementary gates in $\mathscr{G}_{\mathbf{PC}}$. Thus, the other part of the algorithm can be also implemented by a uniform polynomial size QCF.

By analogous arguments, we can also show that Shor's algorithm for the discrete logarithm problem defined in [20] is a Las Vegas quantum algorithm.

Considering the proof of Theorem 5.4, it might be expected that **ZQP** is equal to **ZUPQC**. However, we should notice that the above algorithm uses a Las Vegas type primality testing to produce a correct answer. This primality testing prevents us from producing incorrect answers. But this check-algorithm is a classical Las Vegas one. A classical Las Vegas algorithm can be exactly carried out by a Las Vegas type QTM, since a polynomial-time bounded probabilistic Turing machine can be exactly simulated by a polynomial-time bounded QTM. Now, in the case where such a check-algorithm is carried out by a uniform QCF, it is not known whether we can implement this algorithm by a QTM. Thus, even if a quantum algorithm is carried out efficiently by a Las Vegas type uniform QCF, we cannot say that the algorithm is efficiently carried out by a Las Vegas type QTM.

The state transition of a QTM is determined by the quantum transition function, finite numbers of complex numbers, while in order to characterize that of a QCF, we can use infinite numbers of complex numbers even under the uniformity condition. This suggests that some QCF cannot be simulated exactly by a QTM. In fact, we can show that a QCF carrying out the discrete Fourier transform cannot be *exactly* simulated by any QTM as follows.

Proposition 5.5. *A QCF $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ carrying out the discrete Fourier transform*

$$|a\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{c=0}^{2^n-1} \exp\left(\frac{2\pi \mathrm{i} ac}{2^n}\right) |c\rangle,$$

where $a = 0, \ldots, 2^n - 1$, cannot be exactly simulated by any QTM.

Proof. Let $\bar{\mathbf{Q}}$ be the algebraic closure of **Q**. Let $F(\alpha_1, \ldots, \alpha_m)$ be the field generated by $\alpha_1, \ldots, \alpha_m$ over a field F. The range of the quantum transition function of a QTM $M = (Q, \Sigma, \delta)$ consists of finite numbers of complex numbers $\{\alpha_1, \ldots, \alpha_m\}$. Thus, the set

$\{\langle C' \mid M_\delta^t | C\rangle \mid C', C \in \mathscr{C}(Q,\Sigma),\ t \in \mathbf{Z}_{\geqslant 0}\}$ is included in an extended field $\mathbf{Q}(\alpha_1,\ldots,\alpha_m)$ of $\mathbf{Q}$. On the other hand, the 2^n-dimensional unitary matrix representing the quantum gate $G(K_n)$ determined by K_n contains the complex number $e^{2\pi i/2^n}$ as the components. Therefore, it is sufficient to show the relation $\{e^{2\pi i/2^n} \mid n \in \mathbf{N}\} \not\subseteq \mathbf{Q}(\alpha_1,\ldots,\alpha_m)$. The dimension of the vector space $\mathbf{Q}(e^{2\pi i/2},\ldots,e^{2\pi i/2^n}) = \mathbf{Q}(e^{2\pi i/2^n})$ over $\mathbf{Q}$ is 2^{n-1}. Moreover, $\mathbf{Q}(e^{2\pi i/2^n}) \subseteq \bar{\mathbf{Q}}$. Henceforth, let $F_k = \mathbf{Q}(\alpha_1,\ldots,\alpha_k) \cap \bar{\mathbf{Q}}$. Now, we shall show that F_k is a finite extension of $\mathbf{Q}$ by induction on k. When $k=0$, it is trivial. Suppose that F_k is a finite extension of $\mathbf{Q}$. If $F_{k+1} = F_k$, then it is easy to see that F_{k+1} is a finite extension of $\mathbf{Q}$. Now, suppose that $F_{k+1} \neq F_k$ and let $\gamma \in F_{k+1} \backslash F_k$. Then there is a non-constant rational expression $f(x)$ over $\mathbf{Q}(\alpha_1,\ldots,\alpha_k)$ such that $\gamma = f(\alpha_{k+1})$. Since γ is in $\bar{\mathbf{Q}} \backslash \mathbf{Q}$, there is a minimal polynomial g over $\mathbf{Q}$ of γ, so that we have $g \circ f(\alpha_{k+1}) = g(\gamma) = 0$. It follows that α_{k+1} is algebraic over $\mathbf{Q}(\alpha_1,\ldots,\alpha_k)$. Supposing that l is the dimension of the vector space $\mathbf{Q}(\alpha_1,\ldots,\alpha_{k+1})$ over $\mathbf{Q}(\alpha_1,\ldots,\alpha_k)$, the degree of γ over $\mathbf{Q}(\alpha_1,\ldots,\alpha_k)$ is at most l. Let h_1 be the minimal polynomial over $\mathbf{Q}(\alpha_1,\ldots,\alpha_k)$ of γ. Since γ is also algebraic over $\mathbf{Q}$ and $\mathbf{Q} \subseteq \mathbf{Q}(\alpha_1,\ldots,\alpha_k)$, the polynomial h_1 divides the minimal polynomial h_2 over $\mathbf{Q}$ of γ. The coefficients of h_1 are in $\bar{\mathbf{Q}}$, since h_2 is uniquely decomposable over $\bar{\mathbf{Q}}$. Thus, the coefficients of h_1 are in F_k, so that the degree of γ over F_k is at most l. Therefore, F_{k+1} is a finite extension of F_k. By inductive hypothesis, F_{k+1} is a finite extension of $\mathbf{Q}$. Therefore, $\mathbf{Q}(\alpha_1,\ldots,\alpha_m) \cap \bar{\mathbf{Q}}$ is a finite extension of $\mathbf{Q}$, and hence we have $\{e^{2\pi i/2^n} \mid n \in \mathbf{N}\} \not\subseteq \mathbf{Q}(\alpha_1,\ldots,\alpha_m)$. □

Thus, there is a fair chance that **EQP** $\neq$ **EUPQC** or that **ZQP** $\neq$ **ZUPQC**.

Next we introduce the notion of the uniformity of QCFs based on finite subsets of $\mathscr{G}_u$ and consider classes of languages recognized by such QCFs.

Assume that a finite set $\mathscr{G}$ of quantum gates is indexed as $\mathscr{G} = \{G_1,\ldots,G_l\}$, where G_i is an n_i-bit quantum gate for $i = 1,\ldots,l$. Let $K = (G_{i_m},\pi_m),\ldots,(G_{i_1},\pi_1)$ be a quantum circuit based on $\mathscr{G}$. Then the *$\mathscr{G}$-code* $c_{\mathscr{G}}(K)$ is defined to be the list of finite sequences of natural numbers, $\langle\langle i_1,\pi_1(1),\pi_1(2),\ldots,\pi_1(n_{i_1})\rangle,\ldots,\langle i_m,\pi_m(1),\pi_m(2),\ldots,\pi_m(n_{i_m})\rangle\rangle$. Moreover, let $\mathbf{K}$ be a k-input m-output n-bit quantum circuit $\mathbf{K} = (K,\Lambda_1,\Lambda_2,S)$ based on $\mathscr{G}$, where $[1,n]_{\mathbf{Z}} \backslash \Lambda_1 = \{i_1,\ldots,i_{n-k}\}$ and $\Lambda_2 = \{j_1,\ldots,j_m\}$. Then the $\mathscr{G}$-code of $\mathbf{K}$, denoted by $c_{\mathscr{G}}(\mathbf{K})$, is defined by the list of finite sequences of natural numbers,

$$c_{\mathscr{G}}(\mathbf{K}) = \langle\langle\langle i_1, S(i_1)\rangle,\ldots,\langle i_{n-k}, S(i_{n-k})\rangle\rangle, c_{\mathscr{G}}(K), \langle j_1,\ldots,j_m\rangle\rangle.$$

A QCF $\mathscr{K} = \{\mathbf{K}_n\}_{n\geqslant 1}$ of size s based on $\mathscr{G}$ is said to be *$\mathscr{G}$-uniform* if the function $1^n \mapsto c_{\mathscr{G}}(\mathbf{K}_n)$ is computable by a DTM in time $p(s(n))$ for some polynomial p. Furthermore, a QCF $\mathscr{K}$ is said to be *semi-uniform* if there is a finite set $\mathscr{G} \subseteq \mathscr{G}_u$ such that $\mathscr{K}$ is $\mathscr{G}$-uniform. Now the following lemma holds similar to Lemma 5.1.

Lemma 5.6. *For any semi-uniform QCF $\mathscr{K}$, there exists a polynomial p and a QTM M which simulates $\mathscr{K}$ in time $p(s(n))$, where s is the size of $\mathscr{K}$.*

We say that a language L has *bounded-error* (*or exact*) *semi-uniform polynomial size quantum circuits*, if there is a semi-uniform polynomial size QCF $\mathscr{K} = \{K_n\}_{n\geqslant 1}$ that recognizes L with probability uniformly larger than $\frac{1}{2}$ (with probability 1). In this case,

we write $L \in \mathbf{BSPQC}$ (or $L \in \mathbf{ESPQC}$). We say that L has *zero-error semi-uniform polynomial size quantum circuits*, if there is a semi-uniform polynomial size QCF $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ recognizing L with probability uniformly larger than $\frac{1}{2}$ and satisfying $\rho^{K_{|x|}}(00|x) = 0$ or $\rho^{K_{|x|}}(01|x) = 0$ for any $x \in \{0,1\}^*$. In this case, we write $L \in \mathbf{ZSPQC}$. From these definitions we obviously have $\mathbf{ESPQC} \subseteq \mathbf{ZSPQC} \subseteq \mathbf{BSPQC}$.

The following theorem shows that semi-uniform polynomial size QCFs are equivalent to polynomial-time bounded QTMs whose transition amplitudes are arbitrary complex numbers.

Theorem 5.7. (1) $\mathbf{BQP_C} = \mathbf{BSPQC}$.
(2) $\mathbf{EQP_C} = \mathbf{ESPQC}$.
(3) $\mathbf{ZQP_C} = \mathbf{ZSPQC}$.

Proof. We shall show only statement (1). Statements (2) and (3) can be proved similarly.

Let $L \in \mathbf{BQP_C}$. Then, there is a QTM $M = (Q, \Sigma, \delta)$ that recognizes L with probability uniformly larger than $\frac{1}{2}$ in time $p(n)$, where p denotes a polynomial. For any $n \in \mathbf{N}$ there is a quantum circuit K_n of size $\mathrm{O}(p^2(n))$ that $p(n)$-simulates M by Theorem 4.3. We use the same notations as the proof of Theorem 4.3 by identifying K_n with K in this proof. Then the quantum gates G_1 and G_2 constructing K_n are decomposable by at most $q(n)$ gates in a finite subset $\mathscr{G}$ of $\mathscr{G}_u$, where $q(n)$ is a polynomial. If $\mathscr{G}$ is indexed, there is a DTM that computes the function $1^n \mapsto c_{\mathscr{G}}(K_n)$ in time polynomial in n by the construction of the quantum circuit in Theorem 4.3. Thus, $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ is a semi-uniform polynomial size QCF that recognizes L with probability uniformly larger than $\frac{1}{2}$.

Conversely, suppose $L \in \mathbf{BSPQC}$. Then, there is a semi-uniform polynomial size QCF $\mathscr{K} = \{K_n\}_{n \geqslant 1}$ that recognizes L with probability uniformly larger than $\frac{1}{2}$. By Lemma 5.6, given as input an n-bit string, there is a QTM M that carries out K_n in time $\mathrm{O}(p(n))$, where p is a polynomial. Thus, M recognizes L with probability uniformly larger than $\frac{1}{2}$. □

Remark. Unlike Theorem 5.2, the proof of the existence of a quantum circuit that recognizes $L \in \mathbf{BQP_C}$ in Theorem 5.7 is non-constructive. For example, if a language L can be recognized with probability uniformly larger than $\frac{1}{2}$ by a polynomial-time bounded QTM M, there is a semi-uniform polynomial size QCF $\mathscr{K}$ that recognizes L with probability uniformly larger than $\frac{1}{2}$, but we do not know how to find out $\mathscr{K}$ from M efficiently.

Similar to the proof of Theorem 5.7, by modifying the proof of Theorem 4.4 non-constructively, we can show that SNQTMs (and QTMs with the binary tapes by Lemma 3.2) are equivalent to multi-tape QTMs as machines implementing not only Monte Carlo algorithms but exact ones from the viewpoint of the polynomial-time complexity.

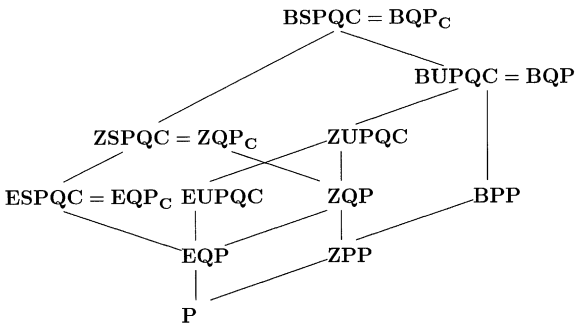

Fig. 5. The inclusions among the classes of languages discussed in this section.

Theorem 5.8. *For any multi-tape QTM M, there is an SNQTM M' (depending on M) that given any positive integer t, simulates M for t steps. Moreover, if M is in* P**C**, *then so is M'.*

Adleman et al. [1] have shown that if all complex numbers are allowed as transition amplitudes of QTMs, for any language L, there exists a language $L' \in \mathbf{BQP_C}$ which is Turing equivalent to L. As a result, **BSPQC** is also a set with uncountable cardinality.

Fig. 5 summarizes the inclusions among the classes of languages which we have discussed in this section.

Appendix A. The generalization of the construction of Theorem 4.3 to multi-tape QTMs

We can extend the construction of Theorem 4.3 to multi-tape QTMs. In what follows, let $\boldsymbol{a} = (a_1, \ldots, a_k)$, $\boldsymbol{a}_j = (a_{j1}, \ldots, a_{jk})$, and $\Sigma = \Sigma_1 \times \cdots \times \Sigma_k$. Let $M = (Q, \Sigma, \delta)$ be a k-tape QTM. This time we use $l_0 + \sum_{j=1}^{k}(2t+1)(2+\lceil \log|\Sigma_j| \rceil)$ wires for the simulation. Conditions (i) and (ii) in the proof of Theorem 4.3 are modified as follows; we denote

$$|q; \sigma_{11}\bar{2}; \sigma_{21}\bar{0}; \sigma_{31}\bar{0}; \cdots; \sigma_{1k}\bar{2}; \sigma_{2k}\bar{0}; \sigma_{3k}\bar{0}\rangle, \ldots,$$

$$|q; \sigma_{11}\bar{0}; \sigma_{21}\bar{0}; \sigma_{31}\bar{2}; \cdots; \sigma_{1k}\bar{0}; \sigma_{2k}\bar{0}; \sigma_{3k}\bar{2}\rangle$$

by

$$|q; \sigma_{11}\sigma_{21}\sigma_{31}; \cdots; \sigma_{1k}\sigma_{2k}\sigma_{3k}; -1, \ldots, -1\rangle, \ldots,$$

$$|q; \sigma_{11}\sigma_{21}\sigma_{31}; \cdots; \sigma_{1k}\sigma_{2k}\sigma_{3k}; 1, \ldots, 1\rangle,$$

respectively.

(i′) $G_1|w_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle = |v_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle$, where

$$|w_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle = |p;\sigma_{11}\bar{0};\sigma_1\bar{1};\sigma_{31}\bar{0};\cdots;\sigma_{1k}\bar{0};\sigma_2\bar{1};\sigma_{3k}\bar{0}\rangle,$$
$$|v_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle = \textstyle\sum_{q,\tau,\boldsymbol{d}} \delta(p,\boldsymbol{\sigma},q,\boldsymbol{\tau},\boldsymbol{d})|q;\sigma_{11}\tau_1\sigma_{31};\cdots;\sigma_{1k}\tau_k\sigma_{3k};\boldsymbol{d}\rangle$$

for any $(p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3)\in Q\times\Sigma^3$; the summation $\sum_{q,\boldsymbol{\tau},\boldsymbol{d}}$ is taken over all $(q,\boldsymbol{\tau},\boldsymbol{d})\in Q\times\Sigma\times[-1,1]_{\mathbf{Z}}^k$.

(ii′) $G_1|h\rangle = |h\rangle$ for each vector $|h\rangle$ in the subspace H of $\mathbf{C}^{2^{l_0+3l}}$ spanned by

$$1+2\times\sum_{j=0}^{k-1}5^j = \frac{1}{2}(2^k+1)$$

types of vectors, where $l=\sum_{j=1}^{k}(2t+1)(2+\lceil\log|\Sigma_k|\rceil)$.

(1) $|q;\sigma_{11}s_{11};\sigma_{21}s_{21};\sigma_{31}s_{31};\cdots;\sigma_{1k}s_{1k};\sigma_{2k}s_{2k};\sigma_{3k}s_{3k}\rangle$,
where $\boldsymbol{s}_2\neq(\bar{1},\ldots,\bar{1})$ and none of s_{1i},s_{2i},s_{3i} are equal to $\bar{2}$ for some $i\in[1,k]_{\mathbf{Z}}$.

(2) For each $j\in[1,k]_{\mathbf{Z}}$ and $(D_{k-j+1},\ldots,D_k)\in[1,2]_{\mathbf{Z}}\times[-2,2]_{\mathbf{Z}}^{j-1}$, we have

$$|u^{j,D_{k-j+1},\ldots,D_k}_{p,\sigma_{11},\sigma_1,\sigma_{31},\ldots,\sigma_{1(k-j)},\sigma_{k-j},\sigma_{3(k-j)},h(D_{k-j+1}),\ldots,h(D_k)}\rangle$$
$$= \textstyle\sum[\delta(p,\boldsymbol{\sigma},q,\boldsymbol{\tau},\boldsymbol{d})|q;\sigma_{11}\tau_1\sigma_{31};\cdots;\sigma_{1(k-j)}\tau_{k-j}\sigma_{3(k-j)}\rangle$$
$$\otimes|f(D_{k-j+1});\cdots;f(D_k);d_1,\ldots,d_{k-j},\ g(D_{k-j+1}),\ldots,g(D_k)\rangle],$$

where the summation is taken over $q\in Q$, $\tau_m\in\Sigma_m$, $d_m\in[-1,1]_{\mathbf{Z}}$ for $m\in[1,k-j]_{\mathbf{Z}}$, and $\tau_n\in S(D_n)$, $d_n\in S'(D_n)$ for $n\in[k-j+1,k]$. Here, for $i\in[k-j+1,k]_{\mathbf{Z}}$, we have

$$h(D_i)=\begin{cases}\sigma_i,\tau_i,\sigma_{1i},\sigma_{2i},\sigma_{3i} & \text{if } D_i=\pm2,\\ \sigma_{1i},\sigma_{2i},\sigma_i & \text{if } D_i=-1,\\ \sigma_{1i},\sigma_i,\sigma_{3i} & \text{if } D_i=0,\\ \sigma_i,\sigma_{2i},\sigma_{3i} & \text{if } D_i=1,\end{cases}\qquad f(D_i)=\begin{cases}\sigma_{1i}\sigma_{2i}\sigma_{3i} & \text{if } D_i=\pm2,\\ \sigma_{1i}\sigma_{2i}\tau_i & \text{if } D_i=-1,\\ \sigma_{1i}\tau_i\sigma_{3i} & \text{if } D_i=0,\\ \tau_i\sigma_{2i}\sigma_{3i} & \text{if } D_i=1,\end{cases}$$

$$g(D_i)=\begin{cases}\mp1 & \text{if } D_i=\pm2,\\ d_i-D_i & \text{if } D_i\in[-1,1]_{\mathbf{Z}},\end{cases}\qquad S(D_i)=\begin{cases}\emptyset & \text{if } D_i=\pm2,\\ \Sigma_i & \text{if } D_i\in[-1,1]_{\mathbf{Z}},\end{cases}$$

and

$$S'(D_i)=\begin{cases}\emptyset & \text{if } D_i=\pm2,\\ \{-1,0\} & \text{if } D_i=-1,\\ \{-1,0,1\} & \text{if } D_i=0,\\ \{0,1\} & \text{if } D_i=1.\end{cases}$$

Let $W=\{|w_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle\,|\,(p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3)\in Q\times\Sigma^3\}^{\perp\perp}$ and $V=\{|v_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle\,|\,(p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3)\in Q\times\Sigma^3\}^{\perp\perp}$. By the unitarity conditions of the quantum transition functions of multi-tape QTM [17], the subspaces W,V and H are all orthogonal to one another and it is verified that $\{|v_{p,\boldsymbol{\sigma}_1,\boldsymbol{\sigma},\boldsymbol{\sigma}_3}\rangle\}$ is an orthonormal system of V. Thus, there exists

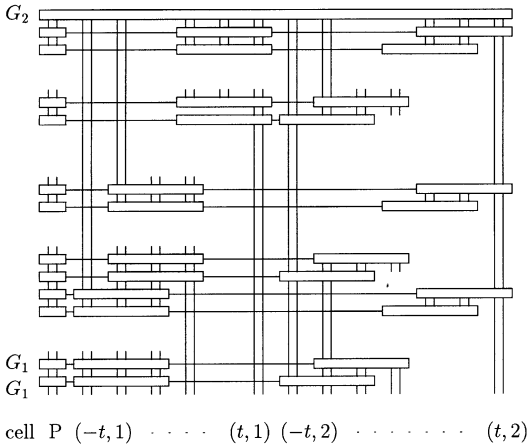

Fig. 6. The quantum circuit K that simulates one step of a two-tape QTM M.

a quantum gate G_1 satisfying the above condition. The subcircuit K simulating one step of M consists of $(2t-1)^k$ quantum gates G_1 and a reversible Boolean gate G_2, which works as in the case of single tape QTMs, and for $i_1,\ldots,i_k=0,1,\ldots,2t-2$ the $(\sum_{j=1}^{k} i_j(2t-1)^{k-j})$th G_1 is connected with first l_0 wires and the wires of bit numbers $l_0+i_1l_1+1,\ldots,l_0+i_1l_1+l_1-1, l_0+(2t-1)l_1+i_2l_2+1,\ldots,l_0+(2t-1)l_1+i_2l_2+l_2-1,\ldots,l_0+(2t-1)(\sum_{j=1}^{k-1} l_j)+i_kl_k+1,\ldots,l_0+(2t-1)(\sum_{j=1}^{k-1} l_j)+i_kl_k+l_k-1$. Here, $l_j=2+\lceil\log|\Sigma_j|\rceil$. In the case of $k=2$, the subcircuit K is illustrated in Fig. 6. Similar to the case of single tape QTMs, we can see that t consecutive subcircuits t-simulates M. Therefore, Theorem 4.3 holds for arbitrary k-tape QTMs.

Acknowledgements

We thank John Watrous for helpful comments. H.N. thanks Tatsuie Tsukiji and Yasuo Yoshinobu for helpful discussions.

References

[1] L.M. Adleman, J. DeMarrais, M.A. Huang, Quantum computability, SIAM J. Comput. 26 (1997) 1524–1540.
[2] L.M. Adleman, M.A. Huang, Primality testing and two dimensional Abelian varieties over finite fields, Lecture Notes in Math., Vol. 1512, Springer, New York, 1992.
[3] A. Barenco, C.H. Bennett, R. Cleve, D. DiVicenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A 52 (1995) 3457–3467.
[4] P. Benioff, The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines, J. Statist. Phys. 22 (1980) 563–591.
[5] C.H. Bennett, Logical reversibility of computation, IBM J. Res. Develop. 17 (1973) 525–532.
[6] C.H. Bennett, E. Bernstein, G. Brassard, U. Vazirani, Strengths and weaknesses of quantum computing, SIAM J. Comput. 26 (1997) 1510–1523.

[7] E. Bernstein, U. Vazirani, Quantum complexity theory (preliminary abstract), Proc. 25th Annual ACM Symp. on Theory of Computing, ACM Press, New York, 1993, pp. 11–20.
[8] E. Bernstein, U. Vazirani, Quantum complexity theory, SIAM J. Comput. 26 (1997) 1411–1473.
[9] A. Berthiaume, G. Brassard, Oracle quantum computing, J. Modern Opt. 41 (1994) 2521–2535.
[10] D. Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer, Proc. Roy. Soc. London Ser. A 400 (1985) 96–117.
[11] D. Deutsch, Quantum computational networks, Proc. Roy. Soc. London Ser. A 425 (1989) 73–90.
[12] D. Deutsch, R. Jozsa, Rapid solution of problems by quantum computation, Proc. Roy. Soc. London Ser. A 439 (1992) 553–558.
[13] A. Ekert, R. Jozsa, Shor's quantum algorithm for factoring numbers, Rev. Modern Phys. 68 (1996) 733–753.
[14] R. Feynman, Simulating physics with computers, Internat. J. Theoret. Phys. 21 (1982) 467–488.
[15] A. Kitaev, Quantum computations: algorithms and error correction, Russian Math. Surveys 52 (1997) 1191–1249.
[16] Ker-I. Ko, H. Friedman, Computational complexity of real functions, Theoret. Comput. Sci. 20 (1982) 323–352.
[17] M. Ozawa, H. Nishimura, Local transition functions of quantum Turing machines, RAIRO Theor. Inform. Appl. to appear. Eprint available from http://xxx.lanl.gov/archive/quant-ph/9811069.
[18] C.H. Papadimitriou, Computational Complexity, Addison-Wesley, Reading, MA, 1994.
[19] P.W. Shor, Algorithms for quantum computations: discrete log and factoring, Proc. 35th Annual IEEE Symp. on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 1994, pp. 124–134.
[20] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Comput. 26 (1997) 1484–1509.
[21] D. Simon, On the power of quantum computation, Proc. 35th Annual IEEE Symp. on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 1994, pp. 116–123.
[22] R. Solovay, Private communication.
[23] R. Solovay, V. Strassen, A fast Monte-Carlo test for primality, SIAM J. Comput. 6 (1977) 84–85.
[24] A. Yao, Quantum circuit complexity, Proc. 34th Annual IEEE Symp. on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 1993, pp. 352–361.