41076: Methods in Quantum Computing

'Quantum Circuit' Module

Min-Hsiu Hsieh
Centre for Quantum Software & Information, Faculty of Engineering and Information Technology, University of Technology Sydney

Abstract

Contents to be covered in this lecture are

1. Universal Gate Sets;
2. Exact Quantum Circuits Simulation;
3. Embedding classical computation into quantum circuits

In an abstract manner, a *circuit* is a model of computation that aims to compute a function, in which the output of the function is produced through a sequence of basic elements, called *gates*. Take for example the classical circuit model. Denote $\mathbb{B}^n := \mathbb{Z}_2^n$. Let $f : \mathbb{B}^n \to \mathbb{B}^m$ be a Boolean function that takes an n-bit string as input and outputs an m-bit string. Let $\mathcal{G}$ be a collection of basic logic gates. A *Boolean circuit* for f is a sequence of gates $\{g_1, \cdots, g_L\} \in \mathcal{G}$ which converts an input $\boldsymbol{x} \in \mathbb{B}^n$ to the output $\boldsymbol{y} \in \mathbb{B}^m$ with a fixed size of K auxiliary bits (called memory cell). Here L is called the size of the circuit. A list of elementary Boolean logic gates is given in Table 1.

Important questions in the circuits model include

- whether there exists an *universal* set of elementary gates so that any function computable in this model can be computed by a sequence of basic gates from this set.
- how to efficiently simulate a larger circuit with gates from the basic gate sets.
- how big is the circuit size L required with the input size to the function f?

In this module, we will answer the first two questions in the setting of quantum circuit model.

1 Quantum Gates and Universal Gate Sets

Denote $\mathbb{H}^n = \mathcal{H}_2^{\otimes n}$ the space of n copies of 2-dimensional Hilbert space $\mathcal{H}_2$. Let $\mathbb{U}^n$ be the collection of unitary operations on $\mathbb{H}^n$. Following from the quantum postulates, a quantum gate is a unitary operation. Collections of basic single-qubit gates and multiple-qubit gates are summarized in Table 2 and 3, respectively.

Let us start with the formal definition of *universal gate sets*.

Definition 1 *A set of gates $\mathcal{G}$ is universal for a computational model if any function computable in this model can be computed by a circuit that uses only gates in $\mathcal{G}$.*

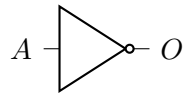
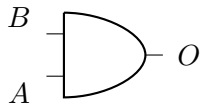
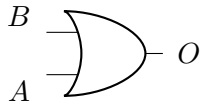
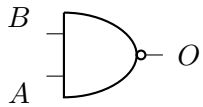
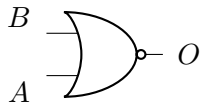
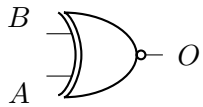
Name	Gates	Truth Table
NOT		A | O 0 | 1 1 | 0
AND		A | B | O 0 | 0 | 0 1 | 0 | 0 0 | 1 | 0 1 | 1 | 1
OR		A | B | O 0 | 0 | 0 1 | 0 | 1 0 | 1 | 1 1 | 1 | 1
NAND		A | B | O 0 | 0 | 1 1 | 0 | 1 0 | 1 | 1 1 | 1 | 0
NOR		A | B | O 0 | 0 | 1 1 | 0 | 0 0 | 1 | 0 1 | 1 | 0
XNOR		A | B | O 0 | 0 | 1 1 | 0 | 0 0 | 1 | 0 1 | 1 | 1

Table 1: Boolean logic gates

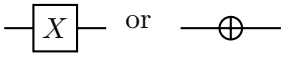
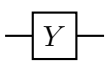
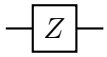
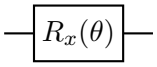
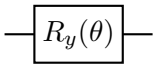
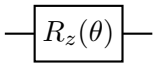
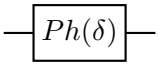
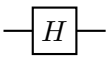
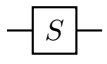
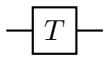
Name	Gates	Matrix
Pauli-X		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
$R_x(\theta)$		$\begin{bmatrix} \cos\frac{\theta}{2} & i\sin\frac{\theta}{2} \\ i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$
$R_y(\theta)$		$\begin{bmatrix} \cos\frac{\theta}{2} & \sin\frac{\theta}{2} \\ -\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$
$R_z(\theta)$		$\begin{bmatrix} e^{i\theta/2} & 0 \\ 0 & e^{-i\theta/2} \end{bmatrix}$
$Ph(\delta)$		$\begin{bmatrix} e^{i\delta} & 0 \\ 0 & e^{i\delta} \end{bmatrix}$
Hadamard		$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Phase		$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
T		$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix}$

Table 2: Single-qubit gates

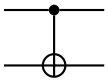
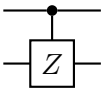
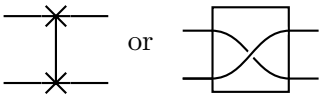
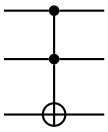
Name	Gates	Matrix
CNOT		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
CZ		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
SWAP		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Toffoli		$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$

Table 3: Multiple-qubit gates

For the classical circuit model, the {NAND}, {NOR} and {AND, NOT} are universal.

Denote by $SU(d)$ the special unitary group of degree d, i.e., collection of $d \times d$ unitary matrices with determinant 1. For the quantum circuit model, the universality in Definition 1 is equivalent to the following.

Definition 2 *A set of quantum gates $\mathcal{G}$ is universal if the group generated by $\mathcal{G}$ is dense*[1] *in $SU(d)$.*

Examples of quantum universal gate sets include

- $\{\text{Toffoli}, H\}$.
- $\{H, T, \text{CNOT}\}$
- $\{\text{CNOT}\} \cup S_1$, where S_1 is the set of single-qubit gates.

Note that the existence of universal gate sets only implies that an arbitrary circuit can be approximated by a finite circuit from the gate set, with no bound on its length. In other words, universality did not promise the simulation of an arbitrary circuit can be done efficiently.

Theorem 3 (Solovay-Kitaev) *Given any universal set of gates $\mathcal{G}$ that is closed under inverse, any unitary operation $U \in SU(d)$ can be ε-approximated using $O(\log^c(\frac{1}{\varepsilon}))$ gates from $\mathcal{G}$ for some constant c.*

With the seminal result of Theorem 3, we know that a quantum circuit of m constant-qubit gates can be approximated to ε error by a quantum circuit of $O(m \log^c(\frac{m}{\varepsilon}))$ gates from a desired finite universal gate set. This guarantees the efficiency of the simulation. A proof of this theorem can be found in Ref. [3].

2 Exact Quantum Circuits Simulation

Let $|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle$. The bloch sphere representation of $|\psi\rangle$ is given in Figure 1.

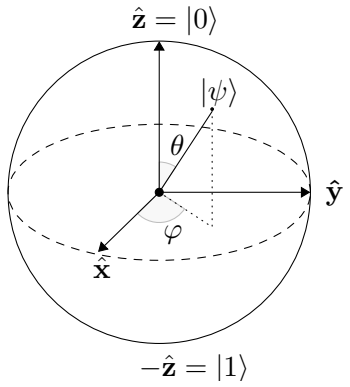


Figure 1: Bloch sphere of $|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle$.

[1] In topology and related areas of mathematics, a subset A of a topological space X is called dense if every point $x \in X$ either belongs to A or is a limit point of A.

Applying the ration gate $R_y(\alpha)$ to $|\psi\rangle$ yields

$$\begin{aligned} |\phi\rangle &= R_y(\alpha)|\psi, \varphi = 0\rangle \\ &= \cos\frac{\theta'}{2}|0\rangle + \sin\frac{\theta'}{2}|1\rangle, \end{aligned} \tag{1}$$

where $\theta' = \theta - \alpha$. We can see that the angle θ (along the $\hat{\mathbf{y}}$ axis) is rotated by an angle α.

Exercise 4 *Show that any $U \in SU(2)$ can be expressed as $U = R_z(\alpha)R_y(\theta)R_z(\beta)$.*

An arbitrary 2×2 unitary V can be related to some $U \in SU(2)$ by a phase shift gate $Ph(\delta)$, defined in Table 2. There are a few identity relations that are useful [1].

Lemma 5 *1. $R_y(\theta_1)R_y(\theta_2) = R_y(\theta_1 + \theta_2)$*

2. $R_z(\theta_1)R_z(\theta_2) = R_z(\theta_1 + \theta_2)$

3. $XR_y(\theta)X = R_y(-\theta)$

4. $XR_z(\theta)X = R_z(-\theta)$

For a 2×2 unitary U, define the $(m+1)$-qubit control operator $\Lambda_m(U)$

$$\Lambda_m(U)|\boldsymbol{a}\rangle \otimes |\xi\rangle = \begin{cases} |\boldsymbol{a}\rangle \otimes U|\xi\rangle & \text{if } |\boldsymbol{a}\rangle{=}|\mathbf{1}\rangle \\ |\boldsymbol{a}\rangle \otimes |\xi\rangle & otherwise \end{cases} \tag{2}$$

where $|\boldsymbol{a}\rangle = |a_1\rangle \otimes \cdots \otimes |a_m\rangle$, for all i $|a_i\rangle$ and $|\xi\rangle \in \mathcal{H}_2$. Choose $m = 2$ and

$$U = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

the $\Lambda_2(U)$ recovers the Toffoli gate in Table 3.

Lemma 6 *A controlled unitary W gate, $\Lambda_1(W)$, can be simulated by a quantum network composed of single qubit gates and CNOT gate, as shown in Figure 2, where $ABC = I$ and $AXBXC = W$.*

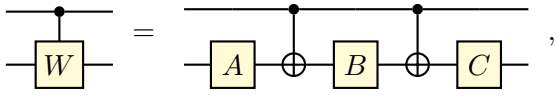


Figure 2: Simulation of controlled unitary gates.

Proof.

Follows from Exercise 4, $W = R_Z(\theta)R_Y(\alpha)R_Z(\beta)$, then, we can choose quantum circuits $A = R_Z(\theta)R_Y(\alpha/2)$, $B = R_Y(-\alpha/2)R_Z(-\theta/2 - \beta/2)$, and $C = R_Z(\beta/2 - \theta/2)$. ■

Lemma 7 *The gate $\Lambda_1(Ph(\delta))$ can be simulated by a single-qubit gate $E = R_z(-\delta)Ph(\frac{\delta}{2})$, as shown in Figure 3.*

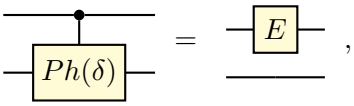


Figure 3: Simulation of controlled phase shift.

Theorem 8 *For any 2×2 unitary U, the $\Lambda_1(U)$ can be simulated by at most four single-qubit gates and two CNOT gates.*

Proof. The proof directly follows from Lemmas 6 and 7. ■

Lemma 9 *Given a $\Lambda_2(U)$, it can be decomposed into five two-qubit gates as follows, where $U = V^2$.*

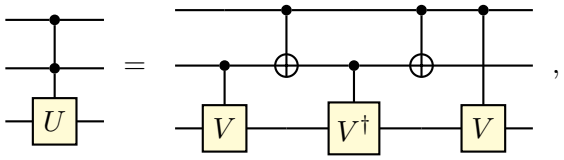


Figure 4: Simulation of controlled-controled unitary gates.

Proof. It is easy to verify that if both the first two qubits are $|1\rangle$, then the operation $V \cdot V = U$ is applied. Otherwise, either I or $VV^\dagger$ is applied. ■

Exercise 10 *How many single-qubit gates and CNOT gates are required to simulate a control-control-U, where $U \in SU(2)$?*

Exercise 11 *Show that a quantum circuit can be transformed to circuits that use only real matrices.*

Theorem 12 *A $\Lambda_m(U)$ gate can be simulated with $\Theta(n^2)$ basic gates.*

Proof. The first step is to note that $\Lambda_n(U)$ can be simulated by two $\Lambda_{n-1}(X)$ gates, one $\Lambda_{n-1}(V)$ gate and two $\Lambda_1(V)$ gates, where $V^2 = U$ (Figure 5). Then we can recursively decompose the three $\Lambda_{n-1}(\cdot)$ gates, each $\Lambda_{n-1}(\cdot)$ requiring three $\Lambda_{n-2}(\cdot)$ gates and two two-qubit gates.

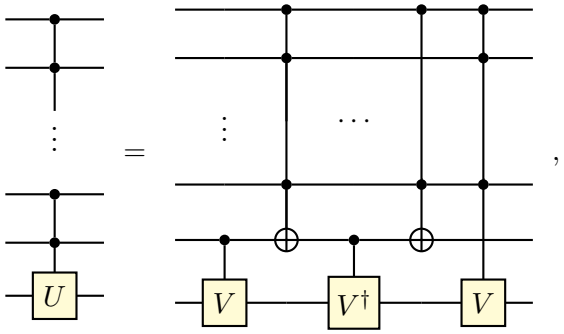


Figure 5: Simulation of $\Lambda_n(U)$ gate.

■

3 Quantum Computation of Classical Boolean Functions

Here, we will discuss how to embed classical computation into quantum circuits with constant overhead, a result derived in Ref. [4]. There are two major steps to arrive at the final goal; namely, conversion of Boolean circuits to reversible circuits and embedding the reversible circuits into quantum circuits.

Firstly, it is easy to show that every Boolean function, even an irreversible one, can be converted to a circuit that only uses bijective gates. For any $f : \mathbb{B}^n \to \mathbb{B}^m$, we define $f_\oplus : \mathbb{B}^n \times \mathbb{B}^m \to \mathbb{B}^n \times \mathbb{B}^m$ by

$$f_\oplus : (\boldsymbol{u}, \boldsymbol{v}) \to (\boldsymbol{u}, \boldsymbol{v} \oplus f(\boldsymbol{u})). \tag{3}$$

In other words, $f_\oplus$ copies the output of the original Boolean function f to the second register. It is not difficult to verify that $f_\oplus$ is a bijection. We will denote $\mathcal{G}_\oplus := \{f_\oplus : \forall f \in \mathcal{G}\}$.

While the function $f_\oplus$ is bijection, the computation might produce "garbage", i.e., extra information which has to be forgotten after the computation is done. Specifically, in classical computers, extra auxiliary bits are allowed in the computation process, and these auxiliary bits might contain information related to the input and output bits. Let $U \in \mathbb{B}^n$ and $W \in \mathbb{B}^K$ be the input and memory registers, where $U \cap W = \emptyset$. Let $Y \subset U \cup W$ be the output register of size m. A sequence of gates, represented by G, that computes $\boldsymbol{y} = G(\boldsymbol{x})$ for some input $\boldsymbol{x}$ in the register U and $\mathbf{0}$ bit string in the memory register W initially:

$$G : (\boldsymbol{x}, \mathbf{0}) \to (\boldsymbol{y}, \boldsymbol{g}), \tag{4}$$

where $\boldsymbol{g} \in U \cup W \backslash Y$ is some garbage bits which lives outside the register Y.

We formally define the *reversible computation* so that these auxiliary bits have to be returned to its initial state.

Definition 13 (Reversible computation) *Let $G : \mathbb{B}^n \to \mathbb{B}^n$ be any bijective function. A sequence of bijective operations $\{r_1, \cdots, r_L\}$ is said to compute G reversibly if their composition $r_L \circ \cdots \circ r_1$ maps $(\boldsymbol{x}, \mathbf{0})$ to $(G(\boldsymbol{x}), \mathbf{0})$ for every $\boldsymbol{x} \in \mathbb{B}^n$, where $\mathbf{0}$ is the ground state of some auxiliary register.*

Lemma 14 [4] *Assume that a function $f : \mathbb{B}^n \to \mathbb{B}^m$ is computable by a Boolean circuit of size L from the set $\mathcal{G}$. Then $f_\oplus$ can be implemented by a sequence of gates of size $2L + m$.*

Proof. Let the sequence of gates $\{g_1, \cdots, g_L\} \in \mathcal{G}$ compute f. Let $\mathcal{G}_\oplus := \{g_\oplus : \forall g \in \mathcal{G}\}$. Then the sequence of bijection gates $\{g_{1,\oplus}, \cdots, g_{L,\oplus}\}$ also computes f.

Let U, W and Y be the input, memory and output registers, where $U \cap W = \emptyset$ and $Y \subset U \cup W$. Let V be a new register of size m. Denote G the composition of $\{g_{1,\oplus}, \cdots, g_{L,\oplus}\}$ that computes f with the assistance of auxiliary memory bits $\mathbf{0}$ in the register W:

$$G : (\boldsymbol{u}, \mathbf{0}, \boldsymbol{v}) \to (f(\boldsymbol{u}), \boldsymbol{g}, \boldsymbol{v}), \tag{5}$$

where $\boldsymbol{g} \in U \cup W \backslash Y$. Then performing the copy operation $\tau : Y \to V$:

$$\tau : (f(\boldsymbol{u}), \boldsymbol{g}, \boldsymbol{v}) \to (f(\boldsymbol{u}), \boldsymbol{g}, \boldsymbol{v} \oplus f(\boldsymbol{u})) \tag{6}$$

Finally, we can uncompute G without affecting the solution in the register V:

$$G^{-1} : (f(\boldsymbol{u}), \boldsymbol{g}, \boldsymbol{v} \oplus f(\boldsymbol{u})) \to (\boldsymbol{u}, \mathbf{0}, \boldsymbol{v} \oplus f_(\boldsymbol{u})). \tag{7}$$

Composition of Eqs. (5)-(7) yields

$$G^{-1} \circ \tau \circ G : (\boldsymbol{u}, \mathbf{0}, \boldsymbol{v}) \to (\boldsymbol{u}, \mathbf{0}, \boldsymbol{v} \oplus f(\boldsymbol{u})), \tag{8}$$

so that the function $f_\oplus$ in Eq. (3) is implemented (after we discard the middle register W). The overall number of gates used is $2L + m$.

Furthermore, the computation can be done by the gate set $\mathcal{G}_\tau = \mathcal{G}_\oplus \cup \tau$. ■

Corollary 15 *Let $G : \mathbb{B}^n \to \mathbb{B}^n$ be a bijection. Suppose that G and G^{-1} are computable in $\mathcal{G}$ by Boolean circuits of size L and L'. Then G can be represented by $\mathcal{G}_\tau$ by a sequence of gates of size $2L + 2L' + 4n$.*

Proof. Let X, Y be classical register of size n, and denote $(\boldsymbol{x}, \boldsymbol{y})$ the bit string in the X, Y registers. The first step is to perform $G_\oplus : X \to Y$

$$G_\oplus : (\boldsymbol{x}, \mathbf{0}) \to (\boldsymbol{x}, G(\boldsymbol{x})), \tag{9}$$

where the assistance of some memory register that we chose to ignore here. By Theorem 14, the step requires circuits of size $2L + n$. The second step is to perform $G_\oplus^{-1} : Y \to X$

$$G_\oplus^{-1} : (\boldsymbol{x}, G(\boldsymbol{x})) \to (\mathbf{0}, G(\boldsymbol{x})), \tag{10}$$

which resets the X register to $\mathbf{0}$. This step requires circuits of size $2L' + n$. The third step is to perform a copy operation $\tau : Y \to X$, defined in Eq. (6)

$$\tau : (\mathbf{0}, G(\boldsymbol{x}) \to (G(\boldsymbol{x}), G(\boldsymbol{x})). \tag{11}$$

This step requires n bit-wise operations. Finally, we can perform another copy operation $\tau : X \to Y$

$$\tau : (G(\boldsymbol{x}, G(\boldsymbol{x}) \to (G(\boldsymbol{x}), \mathbf{0}), \tag{12}$$

followed by remove the register Y. ■

Denote $\mathbb{H}^n = \mathcal{H}_2^{\otimes n}$ the space of n copies of 2-dimensional Hilbert space $\mathcal{H}_2$. Let $\mathbb{U}^n$ be the collection of unitary operations on $\mathbb{H}^n$. Fix the quantum universal gate set to be $\mathcal{Q} = \{\text{CNOT}\} \cup S_1$. For an $U \in \mathbb{U}^n$, define a new operator $\Lambda(U)$ on $\mathbb{U}^{n+1}$ by

$$\Lambda(U)|a\rangle \otimes |\xi\rangle = \begin{cases} |0\rangle \otimes |\xi\rangle & \text{a=0} \\ |1\rangle \otimes U|\xi\rangle & \text{a=1} \end{cases} \tag{13}$$

where $|\xi\rangle \in \mathbb{H}^n$.

Theorem 16 *Let $U \in \mathbb{U}^n$ satisfy $U|\mathbf{0}\rangle = |\mathbf{0}\rangle$. Then $\Lambda(U)$ can be represented by a sequence of quantum gates in $\mathcal{Q} \cup U$ of length $4n + 1$, and the gate U is used only once.*

Proof. Let A and B be quantum registers of size n qubits. Let the input register be $X = \{1\} \cup A$, where 1 denotes the control qubit. The line of proof is similar to that of Corollary 15. We first perform a controlled copy operation (a CNOT) $CNOT : A \to B$

$$CNOT^{A \to B}|1\rangle \otimes |\boldsymbol{a}\rangle_A \otimes |\mathbf{0}\rangle_B = |1\rangle \otimes |\boldsymbol{a}\rangle_A \otimes |\boldsymbol{a}\rangle_B. \tag{14}$$

This copy operation requires n pair-wise CNOT operations. The second step is another controlled copy operation $CNOT : B \to A$

$$CNOT^{B\to A}|1\rangle \otimes |\boldsymbol{a}\rangle_A \otimes |\boldsymbol{a}\rangle_B = |1\rangle \otimes |\boldsymbol{0}\rangle_A \otimes |\boldsymbol{a}\rangle_B. \tag{15}$$

The third step is to perform the unitary U on register B.

$$U^B|1\rangle \otimes |\boldsymbol{0}\rangle_A \otimes |\boldsymbol{a}\rangle_B = |1\rangle \otimes |\boldsymbol{0}\rangle_A \otimes U|\boldsymbol{a}\rangle_B. \tag{16}$$

The fourth step is to perform a controlled copy operation $CNOT : B \to A$

$$CNOT^{B\to A}|1\rangle \otimes |\boldsymbol{0}\rangle_A \otimes U|\boldsymbol{a}\rangle_B = |1\rangle \otimes U|\boldsymbol{a}\rangle_A \otimes U|\boldsymbol{a}\rangle_B. \tag{17}$$

Finally, another controlled copy operation is performed $CNOT : A \to B$

$$CNOT^{A\to B}|1\rangle \otimes U|\boldsymbol{a}\rangle_A \otimes U|\boldsymbol{a}\rangle_B = |1\rangle \otimes U|\boldsymbol{a}\rangle_A \otimes |\boldsymbol{0}\rangle_B, \tag{18}$$

followed by removeing the register B. ■

We can extend $\Lambda(U)$ in Eq. (13) to have ℓ bits of control register. For any $U : \mathbb{B}^\ell \to \mathbb{U}^n$, we can define

$$\Lambda(U)(|\boldsymbol{a}\rangle \otimes |\xi\rangle) = |\boldsymbol{a}\rangle \otimes U(\boldsymbol{a})|\xi\rangle, \tag{19}$$

where $|\xi\rangle \in \mathbb{H}^n$.

Finally, we can reach the main conclusion.

Theorem 17 *Let $f : \mathbb{B}^k \to \mathbb{B}^\ell$ and $U : \mathbb{B}^\ell \to \mathbb{U}^n$. Let $T = \Lambda(U) \in \mathbb{U}^{\ell+n}$ and $\mathcal{F} = \Lambda(U \circ f) \in \mathbb{U}^{k+n}$. If the function f can be computed by a Boolean function of size L from $\mathcal{G}$, then $\mathcal{F}$ can be computed by a sequence of $2L + 1$ gates in the bases $\mathcal{G}_\tau \cup T$ with T being used only once.*

One of the applications of this theorem is to prepare an arbitrary state $|\eta\rangle = \cos\theta|0\rangle + \sin\theta|1\rangle$. Express θ as a bit string of s bits, followed by

$$\begin{aligned} \Lambda(U)|\theta\rangle \otimes |0\rangle &= |\theta\rangle \otimes U(\theta)|0\rangle &(20)\\ &= |\theta\rangle \otimes |\eta\rangle, &(21)\end{aligned}$$

where

$$U(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{22}$$

Further reading

The paper [2] established equivalence between the quantum Turing machines and quantum circuit models. Specifically, Ref. [2] showed that quantum Turing machines can simulate, and be simulated by, uniform families of polynomial size quantum circuits, with at most polynomial slowdown.

References

[1] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter, *Elementary gates for quantum computation*, Phys. Rev. A **52** (1995Nov), 3457–3467.

[2] A. Chi-Chih Yao, *Quantum circuit complexity*, Proceedings of 1993 ieee 34th annual foundations of computer science, 1993, pp. 352–361.

[3] Christopher M. Dawson and Michael A. Nielsen, *The solovay-kitaev algorithm*, 2005.

[4] A. Yu. Kitaev, *Quantum measurements and the abelian stabilizer problem*, 1995.