MAT 4930: Quantum Algorithms and Complexity **Spring 2021**

Lecture 6: Quantum Gates and Complexity

Lecturer: Jean-François Biasse *TA: Robert Hart*

6.1 Evolution of a Closed System

In this section, we present the postulate of the behavior of a closed system. The evolution of the state of a qubit is crucial in quantum computing. Indeed, our goal is precisely to design ways to get an input state to evolve over time to an output state whose measurement gives us the answer to a computational problem.

Quantum theory postulates that the evolution of a state is linear. This means that if the vectors $|\psi_i\rangle$ are a basis for the states of a system of qubits, and if a fixed transformation U is known to map each $|\psi_i\rangle$ to $U|\psi_i\rangle$ then this uniquely determines the transformation of any state via

$$U : |\psi\rangle = \sum_i \alpha_i |\psi_i\rangle \longmapsto \sum_i \alpha_i U|\psi_i\rangle.$$

Example 1 *If a fixed transformation maps $|0\rangle$ to $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ and $|1\rangle$ to $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$, then*

$$\alpha_0|0\rangle + \alpha_1|1\rangle \longmapsto \frac{\alpha_0+\alpha_1}{\sqrt{2}}|0\rangle + \frac{\alpha_0-\alpha_1}{\sqrt{2}}|1\rangle$$

This means that the transformation can be modeled by the linear operator corresponding to the matrix

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Definition 6.1 (Evolution Postulate) *The time-evolution of the state of a closed quantum system is described by a unitary operator. For any evolution $|\psi_1\rangle \to |\psi_2\rangle$ of the closed system, there is a unitary operator U such that*

$$|\psi_2\rangle = U|\psi_1\rangle.$$

Example 2 *The NOT gate is a transformation on a qubit that maps the basis state in the following way:*

$$|0\rangle \longmapsto |1\rangle, \quad |1\rangle \longmapsto |0\rangle.$$

This evolution can be modeled by the unitary matrix

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

We can easily check that $XX^\dagger = X^\dagger X = I$ the identity.

6.2 The Clifford+T gate set

Classical circuits are made of gates from a small gate set that can generate any arbitrary circuit: a *universal* gate set. For example, it is known that the NAND gate that maps $(a, b) \in \{0, 1\}$ to $\overline{a.b} \in \{0, 1\}$ is sufficient to generate all possible logical circuit. Depending on engineering trade-offs, other gates can be used in classical circuits, including more involved ones if the designers know they will be used repeatedly.

The situation in quantum computation is similar in the sense that we only seek to implement a limited set of linear transformation on input states, which we call *quantum gates*. The only difference is that it is impossible to generate all unitary matrices from a finite set of matrices. Instead, we seek to approximate any unitary transformation from a product of matrices from a chosen quantum get set. Not all set of unitary matrices have the property to approximate any input unitary. We focus on the most famous one: the Clifford + T gate set.

Definition 6.2 (Pauli matrices) *The Pauli matrices are*

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

The Pauli matrices generate a finite group of 24 elements called the Pauli group G_1 given by

$$G_1 = \langle X, Y, Z \rangle = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}.$$

Definition 6.3 (Clifford group) *The Clifford group on* 1 *qubit is the group of unitary operators U such that $UPU^\dagger \in G_1$ for all Pauli matrix P:*

$$Cl_1 := \{U \in \mathbb{C}^{2\times 2} \textit{ unitary such that } \forall P \in G_1,\ UPU^\dagger \in G_1\}$$

Definition 6.4 (Hadamard gate) *We define the Hadamard gate over* 1 *qubit by*

$$H := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Definition 6.5 (Phase shift gate) *We define the phase shift over* 1 *qubit by*

$$S := \sqrt{Z} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

Proposition 6.6 *The Clifford group over* 1 *qubit is generated by H and S.*

To finalize our universal gate set over 1 qubit, we need a matrix that does not belong to the Clifford group, namely the T-gate.

Definition 6.7 (T-gate) *We define the T-gate by*

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

The Clifford+T gate set allows us to approximate any 2×2 unitary matrix. This means that any operation on 1 qubit can be approximated by a circuit made of gates from the Clifford + T gate set. This fact is formalized by the Solovay-Kitaev theorem.

Theorem 6.8 (Solovay-Kitaev) *Let $\mathcal{G}$ be the set $\{H, S, T\}$, and let $\varepsilon > 0$ be a constant. There exists a constant $c > 0$ such that for all unitary $U \in \mathbb{C}^{2\times 2}$, there exists a sequence $(U_i)_{i\leq l}$ such that*

$$l \in O(\log^c(1/\varepsilon)) \text{ and } \left\| \prod_i U_i - U \right\| \leq \varepsilon.$$

There are efficient algorithms to find the sequence of $U_i \in \mathcal{G}$ that approximate a given U. This problem is known as *approximate synthesis*, and this can be viewed as compiling the code of a quantum algorithm. To approximate all possible circuits over n qubits, we need an additional gate that can realize entanglement between 2 qubits. The typical choice is a quantum analogue of the controlled-not gate which is called the CNOT gate.

Definition 6.9 (CNOT gate) *The CNOT gate is described by the $C^{4\times 4}$ unitary matrix*

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

The CNOT gate is named this way because it acts on the qubits as a CNOT gate controlled by the first register. Indeed, it maps $|0\rangle|\mathbf{x}\rangle$ to $|0\rangle|\mathbf{x}\rangle$ and $|1\rangle|\mathbf{x}\rangle$ to $|1\rangle|\overline{\mathbf{x}}\rangle$ where $|\overline{\mathbf{x}}\rangle$ is the state where the coefficients of $|\mathbf{x}\rangle$ are flipped.

Example 3 *Assume the input state $|\psi\rangle$ is the product state of $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|0\rangle$, then the CNOT gate acts as*

$$\text{CNOT}|\psi\rangle = \frac{1}{\sqrt{2}}(\text{CNOT}|0\rangle|0\rangle + \text{CNOT}|1\rangle|0\rangle) = \frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|1\rangle),$$

which is an entangled states as we saw previously. Thus, the CNOT was able to turn a product state into an entangled one.

Clifford + T + CNOT gates are enough to approximate any quantum circuit.

6.3 Quantum Complexity

Given a choice of a gate set $\mathcal{G}$ that can approximate any circuit (typically, Clifford + T + CNOT), we want to decide how to measure the cost of a given circuit. Evidently, multiple circuits can achieve the same transformation on a system of n qubits, and not all circuits are equal. In this section, we present the cost function that are most widely used to capture the cost of a circuit. Then, complexity theory consists in observing the asymptotic behavior of these costs functions as the size of the input of the computational problem we're solving grows. For example, Shor's algorithm is a factoring algorithm with polynomial cost, which means that the cost function of the circuit that realizes Shor's algorithm is bounded by a polynomial in the size of the integers given as input. The larger the integer we are attempting to factor, the larger the cost. However, this growth happens at a much slower pace than all known *classical* algorithms (i.e. non-quantum). This is one of the main motivation for the study of quantum algorithms in the scope of cryptography (the security of some of the most widely used cryptographic functions rely on the hardness of factoring large numbers).

G **cost** The most obvious way to measure the cost of a circuit is to simply count the number of gates it contains. We call this the G-cost.

D**-cost** Another relevant metric to measure the performance of a circuit is its depth, i.e. the length of the longest path from the input to the output. This metric essentially captures the execution time. The shorter the depth, the faster the circuit will be executed. It is possible for a circuit to be shallow while having a large G-cost. For example if many operations happen independently on different qubits.

DW**-cost** The Depth-Width metric (DW-cost) is a more recent criterion to determine the performance of a quantum circuit that captures similar information as the Area-Time metric for classical circuits. The idea here is that an idling qubit (i.e. a quantum memory register that is being left alone) has a cost. Indeed, most realistic models for quantum circuits anticipate that a certain level of quantum error correction (a topic mostly outside of the scope of this course) will be necessary to protect the information from the interference of the environment. Therefore, each unit of time we need to keep the quantum memory up has a cost. The DW-cost is by definition Depth $\times$ Width of the circuit. With this metric, a shallow circuit with many qubits will be impacted, even if it has a short execution time. In general, this metric tends to impact memory-entensive algorithms.