

Chapter 4

Universal Gate Sets, Reversible Computation

Quantum algorithms look nothing like classical algorithms, and we will need to start from scratch in understanding how to program a quantum computer.

A quantum circuit on n qubits consists of n wires, each carrying a qubit of information. The qubits are subjected to a sequence of gates (one, two or even three qubit gates). After application of all the gates (say m of them), some of the qubits are measured in the standard basis. We want $m = mO(\text{poly}(n))$ i.e. m is bounded by some polynomial in the number of qubits.

The quantum circuit carries out some unitary transform on the input state, and the final measurement is in the standard basis. Alternatively, we can think of the entire quantum circuit + measurement as measuring the input state in some rotated basis. In physics speak this alternate way of thinking corresponds to the Heisenberg picture.

A basic question we can ask about quantum circuits is whether they can compute functions that classical circuits can compute. Consider a classical circuit C_f that computes a function $f : \{0, 1\}^n \rightarrow \{0, 1\}^k$. On input an n -bit string x , it outputs a k bit string y . We would like to design a quantum circuit U_f that on input the basis state $|x\rangle$ outputs the basis state $|y\rangle = |f(x)\rangle$? If we had such a quantum circuit U_f , then we could also feed in a superposition as input, $|\phi\rangle = \sum_x \alpha_x |x\rangle$, and by linearity the output state would be $U_f |\phi\rangle = \sum_x \alpha_x |f(x)\rangle$.

Note that U_f is unitary, so $U_f^\dagger U_f = U_f U_f^\dagger = I$. The circuit U_f^\dagger is just the circuit U_f applied backwards (last gate first). This means that we can apply U_f and then U_f^\dagger and it is the same as doing nothing. i.e. we get back the input. In particular it means that the function f must be a $1-1$ and onto. It is a bijection or a permutation. i.e. we cannot have $x \neq x'$ such that $f(x) = f(x')$. This appears to be a major restriction on the classical functions that we can compute using quantum circuits. There is a trick to getting around this. We will simply include x in the output, so that U_f that on input the basis state $|x\rangle$ outputs the basis state $|x\rangle |f(x)\rangle$. This prevents the outputs from colliding on inputs x and x' . As before, if we feed in a superposition as input, $|\phi\rangle = \sum_x \alpha_x |x\rangle$, the output state is also a superposition: $U_f |\phi\rangle = \sum_x \alpha_x |x\rangle |f(x)\rangle$.

Looking more closely at U_f and U_f^\dagger , we note that every gate of the circuit has to satisfy the same

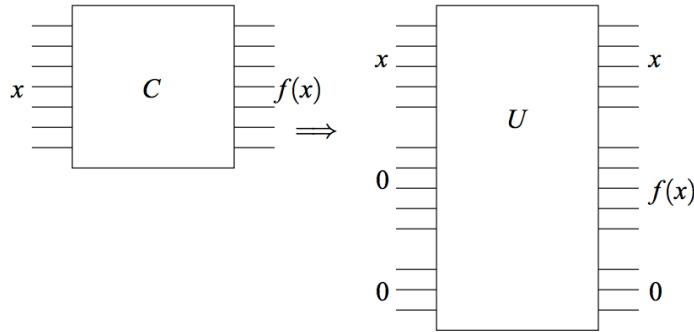
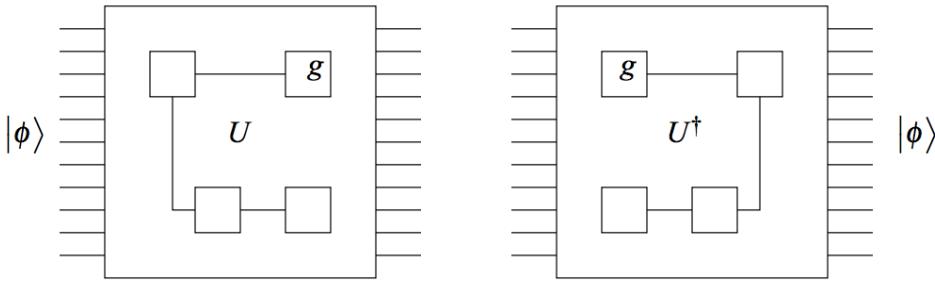


Figure 4.1: Note that the input and output have the same number of qubits in the reversible quantum circuit.

property of implementing a permutation. This makes it more challenging to convert a classical circuit C_f to the corresponding quantum circuit U_f . We have to make each step of the computation reversible.



The circuits for U and U^\dagger are the same size and have mirror image gates. Examples:

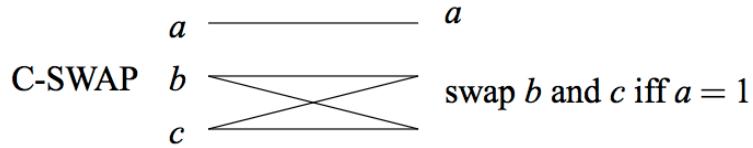
$$H = H^\dagger$$

$$\text{CNOT} = \text{CNOT}^\dagger$$

$$R_\theta = R_{-\theta}^\dagger$$

Recall that any classical circuit can be synthesized from AND and NOT gates. The NOT gate is invertible, and is realized by the quantum X gate. But clearly the AND gate is not reversible. We could either define a reversible NOT gate by retaining and outputting the input bits. A more elegant way to do this is the c-SWAP gate (Figure 4). On classical inputs (basis states) a, b, c , the bit a is the control bit, and if $a = 1$ then b and c are swapped.

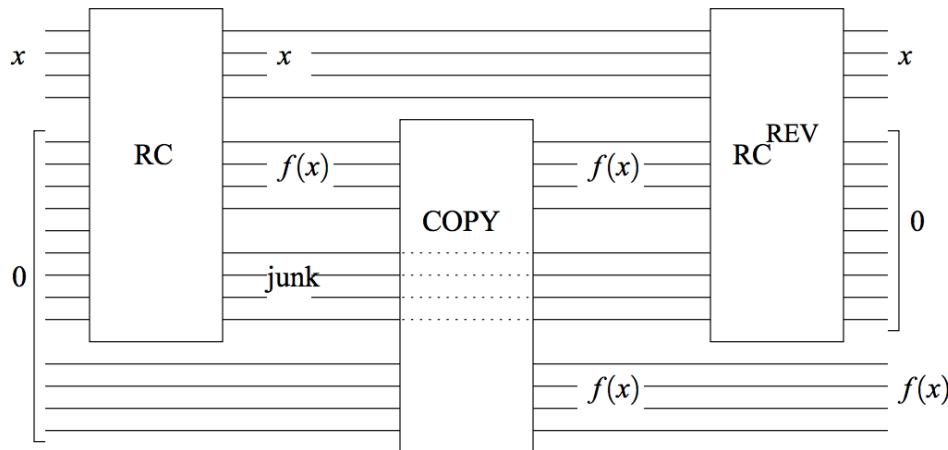
To realize an AND gate on inputs a, b using a c-SWAP gate, we use a c-SWAP gate with $c = 0$. Now the output bits are a, b', c' with $b' = \bar{a}b$ and $c' = ab$. So c' is the desired output bit. (Note that we have lost the property that the input bits a, b are retained. But it is easily remedied by applying a 2-qubit gate that maps $a, \bar{a}b$ to a, b).



To design a quantum circuit corresponding to a general classical circuit, there is one other functionality we must realize: fanout. This is easily done with the c-SWAP gate by setting $b = 1$ and $c = 0$. Now the three output bits are a, \bar{a}, a . It should now be straightforward to take any classical circuit C_f consisting of AND and NOT gates and translate it into a reversible circuit that outputs both x and $f(x)$ on input x . Unfortunately it also outputs a number of other bits that were left over from all the scratch bits introduced to make the computation reversible. So U_f on input $|x\rangle|0^m\rangle$ outputs $|x\rangle|f(x)\rangle|g(x)\rangle$, where we think of the leftover scratch bits $g(x)$ as "garbage bits." As we shall see, these garbage bits are a serious problem while designing quantum algorithms.

To see the problem with garbage bits, recall that we wish to invoke U_f on a superposition $|\phi\rangle = \sum_x \alpha_x |x\rangle$. By linearity the output state will be $\sum_x \alpha_x |x\rangle|f(x)\rangle|g(x)\rangle$. The extra quantum register $|g(x)\rangle$ is entangled with the first two registers, and this changes the effect of any further quantum circuit applied to the first two registers: $\sum_x \alpha_x |x\rangle|f(x)\rangle$. By the principle of deferred measurement, discarding the third register has the same effect as measuring it, and this disturbs the quantum state of the first two registers.

To deal with this issue we must erase the third register $|g(x)\rangle$. But how can we accomplish this if we are not allowed to discard the third register (or its contents)? The answer lies in the unitarity of U_f . We can apply U_f^\dagger , and it will undo the action of U_f , thereby erasing $|g(x)\rangle$. Unfortunately this also erases the second register $|f(x)\rangle$, which was the point of the quantum circuit in the first place! But since $|fx\rangle$ is a computational basis state (an n -bit string), we can clone it using CNOT gates. So that is what we do before applying U_f^\dagger , and that leaves us with a copy of $|f(x)\rangle$, giving us the desired quantum circuit.



4.1 Phase State

Suppose we are given a classical circuit C_f for computing a boolean function (a function whose output is always 0 or 1). i.e. $f : \{0, 1\}^n \rightarrow \{0, 1\}$. We wish to transform the superposition $\sum_{x \in \{0,1\}^n} \frac{1}{2^{n/2}} |x\rangle$ into $\sum_{x \in \{0,1\}^n} \frac{-1^{f(x)}}{2^{n/2}} |x\rangle$. Here is a quantum circuit to create the superposition: Start with n qubits each in the state $\sum_{x \in \{0,1\}^n} \frac{1}{2^{n/2}} |x\rangle$

We now wish to apply a phase depending upon $f(x)$. Let us input this superposition (together with a number of scratch qubits initialized to $|0\rangle$) to the quantum circuit U_f to get the superposition $\sum_{x \in \{0,1\}^n} \frac{1}{2^{n/2}} |x\rangle |f(x)\rangle$

Apply the phase flip gate Z to the $f(x)$ -qubit to get the superposition $\sum_{x \in \{0,1\}^n} \frac{-1^{f(x)}}{2^{n/2}} |x\rangle |f(x)\rangle$

Uncompute $|f(x)\rangle$ by applying U_f^\dagger to get the desired superposition $\sum_{x \in \{0,1\}^n} \frac{-1^{f(x)}}{2^{n/2}} |x\rangle$. Apply U_f^\dagger to get the superposition $\sum_{x \in \{0,1\}^n} \frac{-1^{f(x)}}{2^{n/2}} |x\rangle$.