LECTURE 3 (January 24)

TODAY BQP & its properties

We will introduce our first complexity class BQP (Bounded Quantum Polynomial Time)

This class corresponds to decision problems that can be solved efficiently with a quantum computer and is the quantum analog of the complexity class P

First, let us recall that a problem is in P if there is a deterministic Turing Machine (TM) that solves it in poly(n) time where n is the input length

One can define BQP in terms of a quantum TM but this is not easy to define because a quantum TM takes as input an infinite superposition over all strings $\{0,1\}^*$

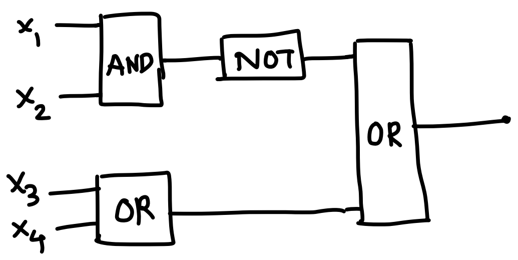
We will define BQP in terms of <u>uniform circuit families</u>

Quantum Circuit

A classical circuit on n bits applies elementary boolean gates (such as AND, OR, NOT) and computes a function

$$f : \{0,1\}^n \longrightarrow \{0,1\}$$

E.g.



The gates only act on ≤ 2 bits & any function $f: \{0,1\}^n \to \{0,1\}$ can be computed by some circuit on n bits

A quantum circuit is similar and applies a sequence of quantum gates that act on O(1)-many qubits & outputs a bit at the end by measuring one designated output qubit

To define it formally, let us first define quantum gates

<u>k-local quantum gate</u> This is a unitary U that acts on k-qubits Usually $k = O(1)$

Often the gate U is applied to a subset of k qubits in an n-qubit system

Formally, this means that the unitary $I \otimes I \otimes \cdots \otimes I \otimes U \otimes I \otimes \cdots I$ is applied on the n qubits where all qubits that are not in the desired subset are acted on by identity

Some examples of quantum gates

$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ — bitflip or NOT gate

$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ — phaseflip gate

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

$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ — Controlled NOT

$X|0\rangle = |1\rangle$
$X|1\rangle = |0\rangle$

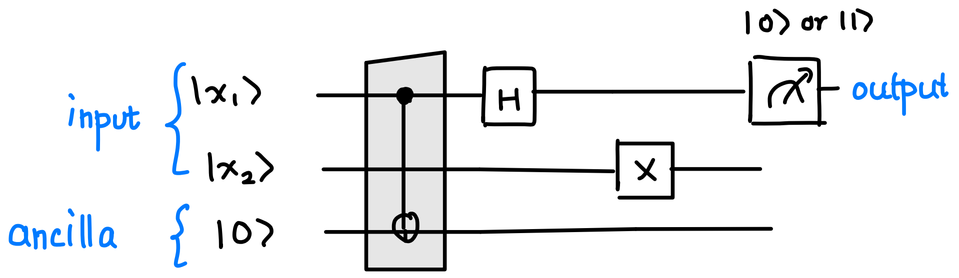
$Z|0\rangle = |1\rangle$
$Z|1\rangle = -|1\rangle$

$H|0\rangle = |+\rangle$
$H|1\rangle = |-\rangle$

$\text{CNOT}\,|0\rangle|x\rangle = |0\rangle|x\rangle$
$|1\rangle|x\rangle = |1\rangle|x \oplus 1\rangle$

Control ↲ ↳ Target

A quantum circuit on n-qubits applies these gates in sequence and may also use some extra m-qubits as workspace. These extra qubits are initialized to $|0\rangle$ typically and are called ancillas



Control

Target

= CNOT gate

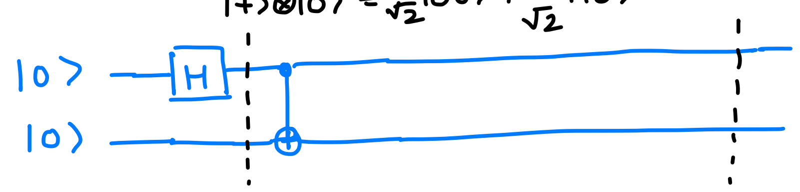
One designated qubit (say the first one) is measured in the $\{|0\rangle, |1\rangle\}$ basis denoted [measurement symbol] and that is the output of the circuit

In general, the circuit can take any superposition over all n qubits as input

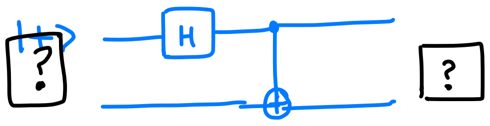
Size of the circuit = Number of gates in the circuit

Some Examples

1. $|+\rangle \otimes |0\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle$



$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle = |EPR\rangle$

2. 

(IN - CLASS EXERCISE)

A first attempt at defining efficient quantum computation :

Problem is efficiently solvable by a quantum computer if ∃ a family of poly(n)-sized circuits that solves it on length n- inputs

The problem with this is that it may be hard to find such circuits!

<u>Uniform Quantum Circuit Family</u> A quantum circuit family $\{C_n\}_{n\in\mathbb{N}}$ is uniform if there is a poly-time classical algorithm that outputs the description of C_n on input 1^n

Can a classical algorithm even compute the description, which could contain arbitrary complex numbers ?

A fundamental result called the Solovay-Kitaev theorem says that the gates H and CCNOT (controlled-controlled-NOT) is a universal gate set for quantum computation meaning any arbitrary 2-qubit unitary may be approximated by short sequences of H & CCNOT.

These gates are analogous to {AND, OR, NOT} gates for classical circuit & we can restrict ourselves to circuits with these two gates

<u>Randomness in Quantum Circuits</u>

Quantum Circuits are inherently probabilistic, so how do we say it solves a problem ?

Let's look at randomized computation first

A randomized algorithm solves a decision problem with bounded error if $\mathbb{P}[\text{answer is correct}] \geq \frac{2}{3}$

$\frac{2}{3}$ is arbitrary, we can choose it be $\frac{1}{2}+\varepsilon$ for any constant ε & we can even make it $1-2^{-n}$ by repeating the algorithm n times independently & taking the majority outcome among these n outcomes

This is called the amplification trick and runtime only increases by a factor of n

BPP = class of problems that can be solved in poly-time by a bounded error randomized algorithm

$\boxed{\text{BQP}}$ BQP = class of decision problems for which $\exists$ a uniform quantum circuit family that outputs the correct answer on all inputs with probability $\geq \frac{2}{3}$

One can amplify it to $1-2^{-n}$

Are quantum classes closed under subroutines ?

P^P = class of problems that can be solved by a P-machine that can call a subroutine that solves any problem in P in a single step

Clearly, $P \subseteq P^P$ but it is not hard to see that $P^P = P$

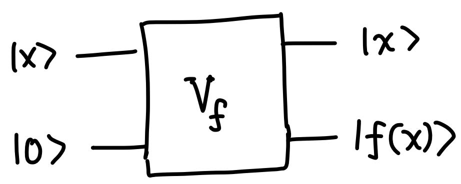
$\boxed{\text{Question}}$ Does $BQP^{BQP} = BQP$?

This question is a bit more subtle to answer

What do we mean by invoking a quantum subroutine that solves a BQP problem ?

Let $f(x)$ be the answer to the BQP problem on input x

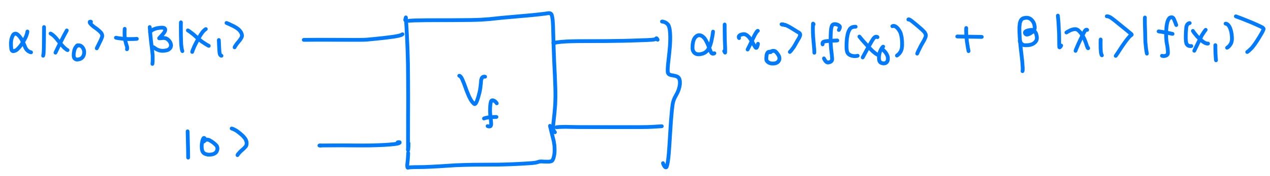
A quantum circuit that has access to a subroutine that solves f can use a special unitary gate in the circuit



(Why this is the right model will be discussed in the next couple of lectures)

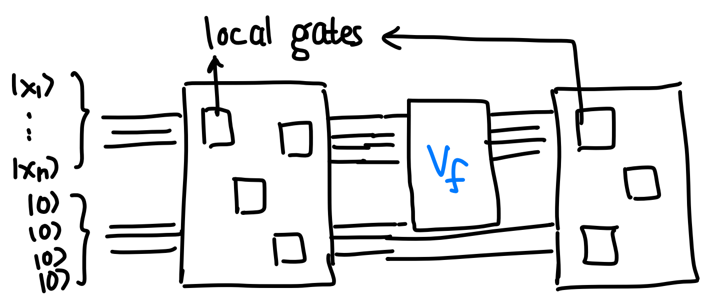
This gate is not local and can act on many qubits simultaneously

Example



Quantum circuit can use this gate anywhere in the circuit

Example



≡ Circuit C

Note that V_f "computes" the true value of the function so the quantum circuit can use these true values in intermediate superpositions

Now, in order to say $BQP^{BQP} \subseteq BQP$ we need to come with a quantum circuit that does not use any V_f gates and outputs the answer computed by a quantum circuit C which uses V_f gates

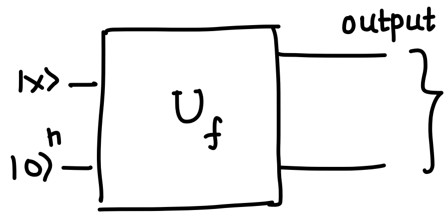
The obvious idea is to use the BQP circuit that computes f instead of V_f

Let us call this circuit U_f

Now there are two issues that need to be handled

① <u>Error</u> : U_f only computes f with probability $\geq \frac{2}{3}$ as opposed to V_f

All we know about U_f is that



Entangled Junk

$$= \alpha|f(x)\rangle|\psi_{f(x)}\rangle + \beta|\neg f(x)\rangle|\psi_{\neg f(x)}\rangle$$

$\hookrightarrow |\alpha|^2 \geq 2/3$

② <u>Entangled Junk</u> is also a problem