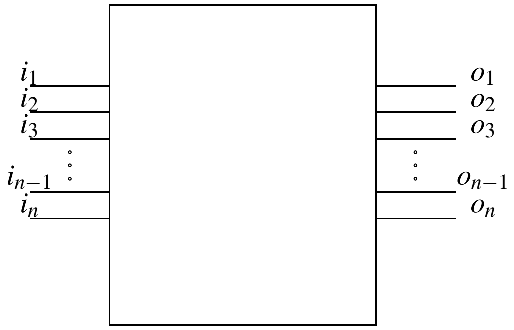
CS 294-2 Quantum Complexity theory 1/31/07
Spring 2007 Lecture 5

1 Quantum Circuits

A quantum circuit implements a unitary operator in a Hilbert space $\mathbb{C}^{2^n}$, given as primitive a (usually finite) collection of gates each of which implements a unitary operator on k qubits for some small k. Unitarity implies that quantum circuits have the same number of inputs and outputs. The picture of a quantum circuit is as follows:



where the quantum gates belong to some universal family of quantum gates.

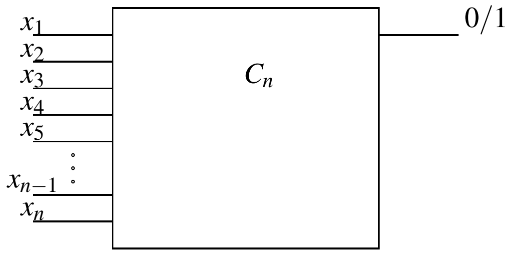
We will define our complexity classes in terms of circuits. Let us start by defining the class P of polynomial time computable decision procedures or languages.

1.1 Class P - Polynomial Time

A definition of the class P in terms of circuits is the following:

$L \in P$ iff there is a family $\mathfrak{F} = \{C_n\}_{n \in \mathbb{N}}$ of circuits such that:

- $|C_n| \leq poly(n), \forall n \in \mathbb{N}$
- *U*niformity The description of the circuit C_n can be computed in time polynomial in n (by a Turing Machine).
- if $|x| = n$ then $C_n(x) = (c \in L?)$



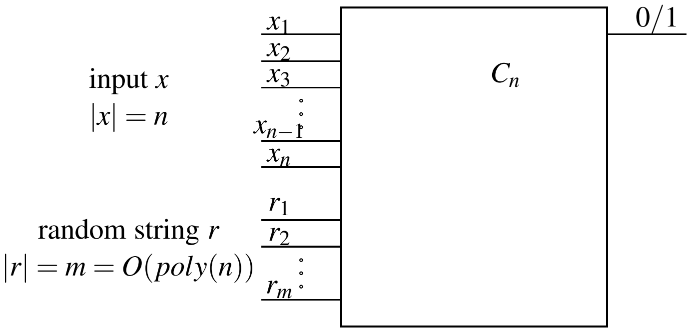
1.2 Class BPP - Bounded Error Probabilistic Polynomial Time

In the seventies it was realized that randomness can sometimes speed up computation. Accordingly the class of efficiently solvable computational problems was expanded to probabilistic polynomial time with small probability of error.

A definition of the class BPP in terms of circuits is the following:

$L \in BPP$ iff there is a family $\mathfrak{F} = \{C_n\}_{n\in\mathbb{N}}$ of circuits such that:

- every circuit C_n has an input x of $|x| = n$ bits and a random input r of $|r| = O(poly(n))$ bits
- $|C_n| \leq poly(n), \forall n \in \mathbb{N}$
- *U*niformity The description of the circuit C_n can be computed in time polynomial in n (by a Turing Machine).
- moreover:
  - if $x \in L$ and $|x| = n$ then $Pr[C_n(x,r) = \text{"}yes\text{"}] \geq 2/3$
  - if $x \notin L$ and $|x| = n$ then $Pr[C_n(x,r) = \text{"}no\text{"}] \geq 2/3$

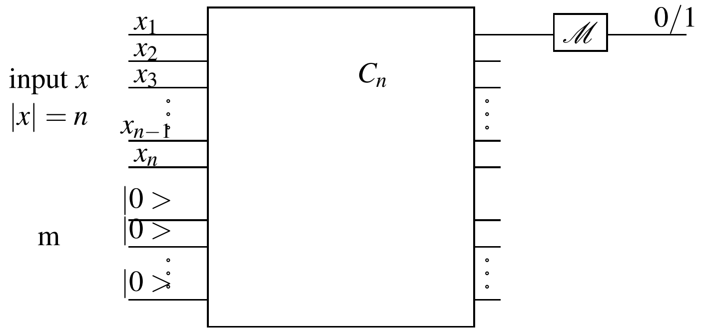


1.3 Class BQP - Bounded Error Quantum Polynomial Time

A definition of the class BQP in terms of circuits is the following:

$L \in BQP$ iff there is a family $\mathfrak{F} = \{C_n \in SU(n)\}_{n\in\mathbb{N}}$ of quantum circuits (unitary operators) such that:

- every circuit C_n has an input x of $|x| = n$ bits and $m = O(poly(n))$ additional inputs of value $|0>$
- the output of the computation is considered to be the outcome of the measurement on the first output of the circuit
- $|C_n| \leq poly(n), \forall n \in \mathbb{N}$
- *U*niformity The description of the circuit C_n can be computed in time polynomial in n (by a Turing Machine).
- moreover:
  - if $x \in L$ and $|x| = n$ then $Pr[measure = 1] \geq 2/3$
  - if $x \notin L$ and $|x| = n$ then $Pr[measure = 0] \geq 2/3$



1.4 Reversibility and P $\subseteq$ BQP

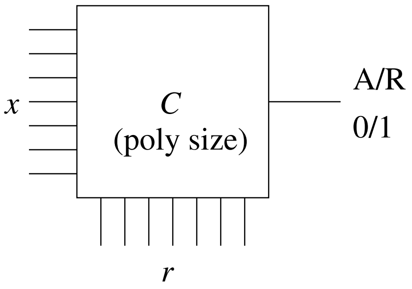
The construction from the last lecture showing how to convert any classical circuit with n inputs and m gates into a reversible circuit with $O(n+m)$ inputs and $O(n+m)$ gates shows that $P \subseteq BQP$. This is because any reversible circuit can be implemented as a quantum circuit which has the same behavior when the input is a computational basis state.

2 BPP $\subseteq$ BQP

We will show that any circuit in BPP can be simulated in BQP by first generating random qubits and then simulating the corresponding polynomial circuit.

2.1 Review: BPP

BPP stands for bounded error probabilistic polynomial time. As an example, consider the language PRIMES consisting of prime numbers. There exists a polynomial size circuit C which takes as input x and some random bits r and outputs 1 for ACCEPT and 0 for REJECT.

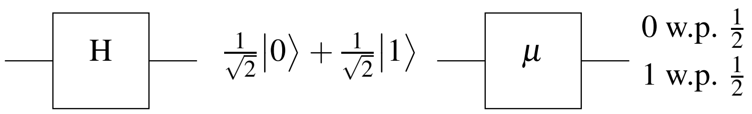


We say PRIMES $\in$ BPP if

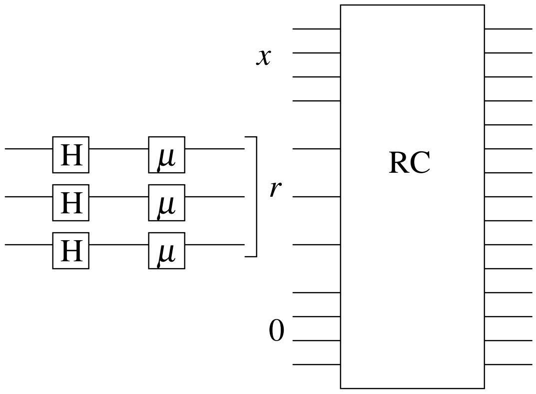
$$\begin{aligned} x \in \text{PRIMES} \quad &\Rightarrow \quad \Pr\{C(x,r) = 1\} \geq 2/3, \\ x \notin \text{PRIMES} \quad &\Rightarrow \quad \Pr\{C(x,r) = 0\} \geq 2/3. \end{aligned}$$

2.2 Simulating BPP

The main difference between a P circuit and a BPP circuit is the additional input of r random bits. We have already shown that any circuit in P can be simulated in BQP. We want to show that it is possible to generate random qubits from $|0\rangle$ inputs. A simple solution is to apply the Hadamard gate to each $|0\rangle$ and then measure. The Hadamard gate converts $|0\rangle$ to $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. Measuring will result is either $|0\rangle$ or $|1\rangle$ with equal probability.



If we generate random bits like this and then run the corresponding quantum circuit to C, we get the straightforward circuit below.



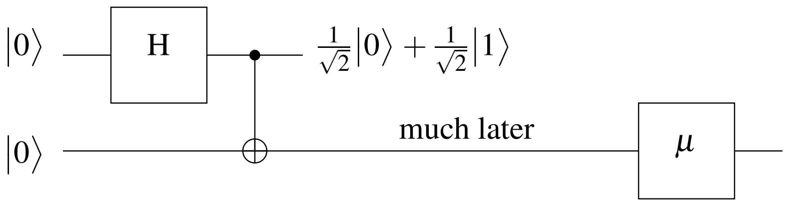
Measurement can be tricky in the intermediate stages of a quantum circuit. Why not skip the measurement and get a superposition of states? Well, if a Hadamard gate occurs in the circuit, we have a problem. The desired outcome is one of these two possibilities with probability $1/2$:

$$|0\rangle \longrightarrow \boxed{\mathrm{H}} \longrightarrow \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$
$$|1\rangle \longrightarrow \boxed{\mathrm{H}} \longrightarrow \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

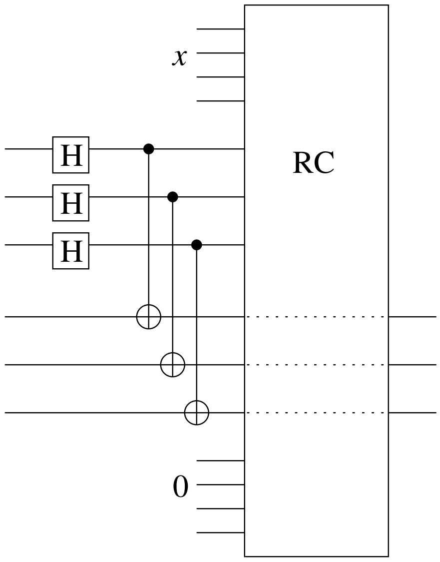
No interference occurs here. Unfortunately, interference can lead to the following undesirable situation in which the randomness disappears:

$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle)\longrightarrow \boxed{\mathrm{H}} \longrightarrow |0\rangle$$

Measurement prevents quantum interference. But, by the principle of deferred measurement, we can postpone the measurement and get the same result. In fact, we can post the measurement indefinitely and not perform it at all.



We now need twice as many qubits as before. Half of them are passed through Hadamard gates and connected by CNOT gates to the other half. This fixes the first half of the qubits to either $|0\rangle$ or $|1\rangle$, even though no measurement was made. It is important to note, however, that since the second half of the qubits are now entangled with the first half, we must be certain not to make any measurements on them either.



3 $BQP \subseteq PSPACE$

Theorem 5.1: $P \subseteq BPP \subseteq BQP \subseteq P^{\#P} \subseteq PSPACE$.

We give a sketch of the proof that $BQP \subseteq P^{\#P}$. We assume without loss of generality that all the transition amplitudes specified in the transition function δ are real (exercise). The action of a quantum circuit may be described by a tree, each node is labelled with a computational basis state, i.e. a bit string. The root of the tree corresponds to the input $|x\rangle$ and applying a gate to any node yields a superposition of basis states represented by the children of that node. We label the edge to each child by the corresponding amplitude. Let us assume that the quantum circuit accepts or rejects depending upon whether the first qubit, when measured in the computational basis is 0 or 1. Thus each leaf of the tree is either an accepting or rejecting node depending on whether the first bit of the string labeling it is 0 or 1. The amplitude of a path p from the root to a leaf of the tree, β_p, is just the product of the branching amplitudes along the path, and is computable to within $1/2^j$ in time polynomial in j. Several paths may lead to the same configuration c. Thus the amplitude of c after application of T gates is the following sum over all T length paths p: $\alpha_c = \sum_{p\ to\ c} \beta_p$. The probability that quantum circuit accepts is $\sum_{accepting\ c} |\alpha_c|^2$. Let $a_p = max(\beta_p, 0)$ and $b_p = max(-\beta_p, 0)$. Then $|\alpha_c|^2$ can be written as $|\alpha_c|^2 = \sum_{p\ to\ c}(a_p - b_p)^2 = \sum_{p\ to\ c} a_p^2 + b_p^2 - \sum_{p,p'\ to\ c} 2a_p b_p$. It follows that the acceptance probability of the quantum circuit can be written as the difference between the two quantities $\sum_{accepting\ c} \sum_{p\ to\ c} a_p^2 + b_p^2$, and $\sum_{accepting\ c} \sum_{p,p'\ to\ c} 2a_p b_{p'}$. Since each of these quantities is easily seen to be in $P^{\#P}$, it follows that $BQP \subseteq P^{\#P}$.

In view of this theorem, we cannot expect to prove that BQP strictly contains BPP without resolving the long standing open question in computational complexity theory, namely, whether or not $P = PSPACE$.