LECTURE 6 (February 5)

TODAY Oracle Separations
BQP vs NP

RECAP Simon's Problem

Given a black-box $f : \{0,1\}^n \to \{0,1\}^n$ promised that either

- f is 1-to-1
- OR $\exists$ an unknown string $s \neq 0$ s.t. $\forall\, x \neq y$, $f(x) = f(y)$ iff $y = x \oplus s$

Figure out which case we are in

Theorem (Simon)
(a) $\exists$ a quantum algorithm solving the problem with $O(n)$ queries with constant error
(b) any classical algorithm requires $\Theta(2^{n/2})$ queries for constant error

From query complexity to oracle separations $\exists\, O$ and a language L^O s.t. $L^O \in BQP^O$, $L^O \notin BPP^O$

How do we define input to the TM? The Oracle? Handle all input lengths

This is achieved via a standard argument called diagonalization.

For every n, let f_n be truth table of a function $f : \{0,1\}^n \to \{0,1\}^n$ $2^n \cdot n$ bits

$$f_n = \begin{cases} \text{uniform 1-1} & \text{w.p. } 1/2 \\ \text{uniform Simon's fn.} & \text{w.p. } 1/2 \end{cases}$$

Oracle O on length n string x outputs $O(x) = f_n(x)$

$L^O = \{ 1^n \mid f_n \text{ is a Simon function} \}$

Exercise

$\boxed{\text{Claim 1}}$ $\mathbb{P}_O\left[\, \mathbb{P}_A\left[\text{fixed } BPP^O \text{ machine } A^O \text{ decides } L^O \text{ correctly on } 1^n\ \forall n \geq 1\right] \geq \frac{2}{3}\, \right] = 0$

$\Rightarrow \mathbb{P}_O\left[\exists\, BPP^O \text{ machine } A^O \text{ s.t. } \mathbb{P}\left[A^O \text{ decides } L^O \text{ correctly on } 1^n\ \forall n \geq 1\right] \geq \frac{2}{3}\right] = 0$

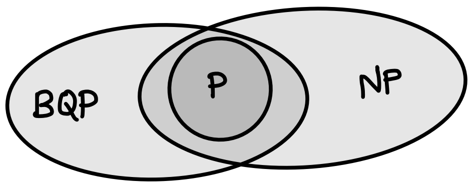
$\boxed{\text{Claim 2}}$ $\mathbb{P}_O\left[\, \mathbb{P}_M\left[M^O \text{ decides } L^O \text{ correctly on } 1^n\ \forall n \geq 1\right] \geq 0.8\right] \geq \frac{1}{2}$
↳ measurement

$\Rightarrow \exists\, O$ s.t. $L^O \in BQP^O$
$L^O \notin BPP^O$ $\square$

BQP vs NP

Can quantum computers solve NP-complete problems?
i.e. is $NP \subseteq BQP$?

Generally, it is believed that the answer is NO and these classes are incomparable and the picture looks like the following



Now we will see some heuristic evidence of this in the form of oracle separations

$BQP^O \not\subseteq NP^O$

$\exists$ an oracle O and a language L^O s.t. $L^O \in BQP^O$ yet $L^O \notin NP^O$

This is based on the complement of Simon's problem

Oracle O on length n string x outputs $O(x) = f_n(x)$

$coSimon^O = \{ 1^n \mid f_n \text{ is a one-to-one function} \}$

First of all, $coSimon^O \in BQP^O$ since Simon's algorithm works in both cases with probability $\geq 2/3$.

Why complement? Because we want to show $coSimon^O \notin NP^O$ and the secret string s in Simon's problem can serve as a certificate for an NP^O-machine. But it is not clear that there is any short certificate for the fact that f_n is one-to-one

This time we will construct the oracle adversarially (instead of probabilistically)

Let $M_1, M_2, \ldots$ be an enumeration of NP^O-machines and let $p_i(n)$ be the time that M_i takes which is some poly(n)

M_i can only query inputs of length $p_i(n)$ on input 1^n

We will choose f_n on larger and larger input lengths so that NP^O-machine will fail.

Let n_i be the next input length n on which we haven't defined the oracle and that satisfies $\frac{2^n}{2} \geq p_i(n)^2$

- We will choose an arbitrary one-to-one fn. $f:\{0,1\}^{n_i} \to \{0,1\}^{n_i}$ and "try" $f_{n_i} = f$

- Run M_i on the current oracle on input 1^{n_i}

[If it queries a different input length that is undefined, set arbitrarily]

- If M_i outputs "f_n is Simon's function" $\to$ We set $f_{n_i} = f$ & all the other input lengths that were undefined & queried arbitrarily

- If M_i outputs "f is one-to-one fn" $\to$ We choose another Simon's fn that is consistent with the NP-certificate (which only depends on the input 1^{n_i} and the queries).
  This is possible since M_i only makes $p_i(n)$ queries and if $p_i(n)^2 \le \frac{2^n}{2}$ there is a Simon's function consistent with the NP-certificate
  Now the same certificate causes M_i to accept 1^{n_i} which is not in the language
  By definition, any string not in the language should not have any certificate

  Overall, our oracle O now implies that M_i fails on input length n_i
  Thus, $\text{coSimon}^O \notin NP^O$ and this shows that $BQP^O \not\subseteq NP^O$ $\square$

Can quantum computers solve NP-hard problems ? Is $NP^O \subseteq BQP^O$?

Let us take the SAT problem which is NP-complete

Given a boolean formula $f(x_1, \ldots x_n)$ in variables $x_1, \ldots x_n \in \{0,1\}$ each check if it is satisfiable i.e. if $\exists\, x \in \{0,1\}^n$ such that $f(x) = 1$

E.g. $(x_1 \vee x_2 \vee x_3) \wedge (\overline{x_3} \vee x_5)$

Given an assignment $x \in \{0,1\}^n$, one can efficiently check if f is satisfiable

Suppose one has access to an oracle that on input x, outputs $f(x)$
Deterministically it would take $O(2^n)$ queries to check if f is satisfiable by brute-force

Can a quantum algorithm do better ?

Theorem (1) Grover's algorithm solves the above problem with $O(2^{n/2})$ quantum queries given a unitary that implements $|x, b\rangle \longmapsto |x, b \oplus f(x)\rangle$ OR $|x\rangle \to (-1)^{f(x)}|x\rangle$

(2) No quantum algorithm can solve this in $o(2^{n/2})$ queries
i.e. $\nexists$ no polynomial query quantum algorithm that solves SAT

Diagonalization $\implies$ $NP^O \not\subseteq BQP^O$

From now on, we will only study these questions in the query model
These imply oracle separations via standard diagonalization arguments as we have seen, so we will not repeat them

Grover's Search Algorithm

We will consider a simpler version of the problem

Suppose we have an oracle O that on input n implements $f:\{0,1\}^n \to \{0,1\}$
such that either $f \equiv 0$ i.e. f is the all zero function
OR there is exactly one x^* such that $f(x^*) = 1$
↳ We call this the marked element

The problem is to determine which type of function O was given

Quantum Algorithm has access to the phase oracle $U_f : |x\rangle \to (-1)^{f(x)}|x\rangle$

This either does nothing (if $f \equiv 0$) or adds a phase to the marked element
Let us focus on this case

Idea Start with the uniform superposition over all inputs $x \in \{0,1\}^n$
i.e.

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle = |+\rangle^{\otimes n} = |\psi_0\rangle \leftarrow \text{Initial State}$$

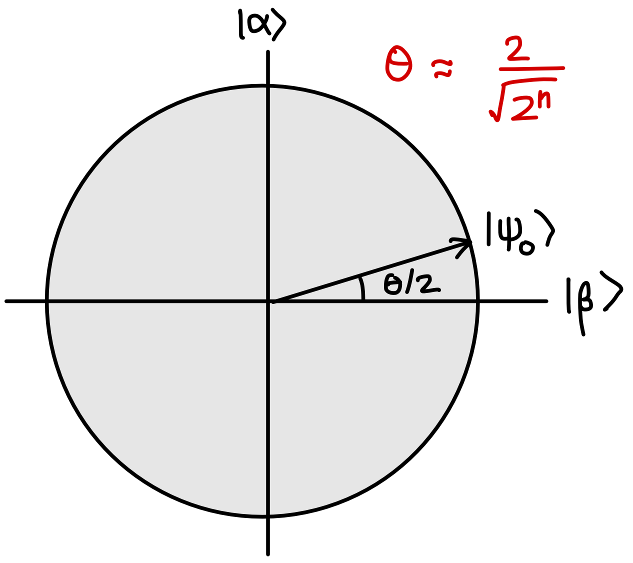
$$= \frac{1}{\sqrt{2^n}}|x^*\rangle + \sqrt{\frac{2^n-1}{2^n}}\underbrace{\left(\frac{1}{\sqrt{2^n-1}}\sum_{x\neq x^*}|x\rangle\right)}_{|\beta\rangle}$$

$$= \frac{1}{\sqrt{2^n}}\underset{\substack{\downarrow\\ \text{Marked state}}}{|\alpha\rangle} + \sqrt{\frac{2^n-1}{2^n}}\underset{\substack{\downarrow\\ \text{superposition over all unmarked state}}}{|\beta\rangle}$$

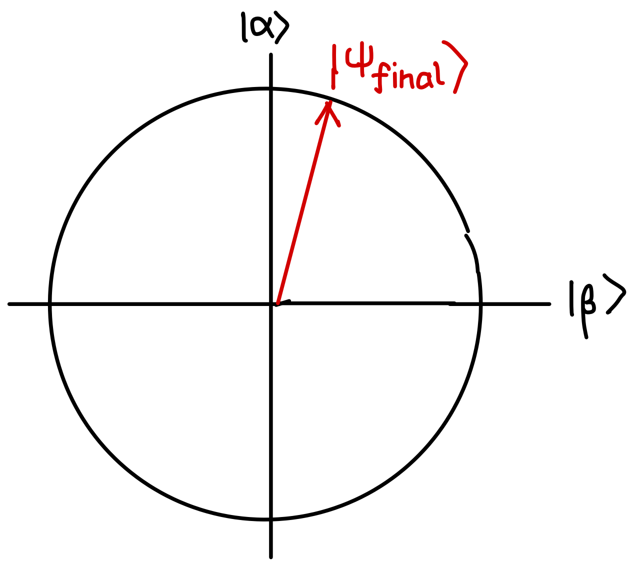
Move the amplitude to the marked element slowly

Let us write $|\psi_0\rangle = \sin\left(\frac{\theta}{2}\right)|\alpha\rangle + \cos\left(\frac{\theta}{2}\right)|\beta\rangle$ where $\sin\left(\frac{\theta}{2}\right) = \frac{1}{\sqrt{2^n}}$

Initially, our state looks like this What we want



What can we do?

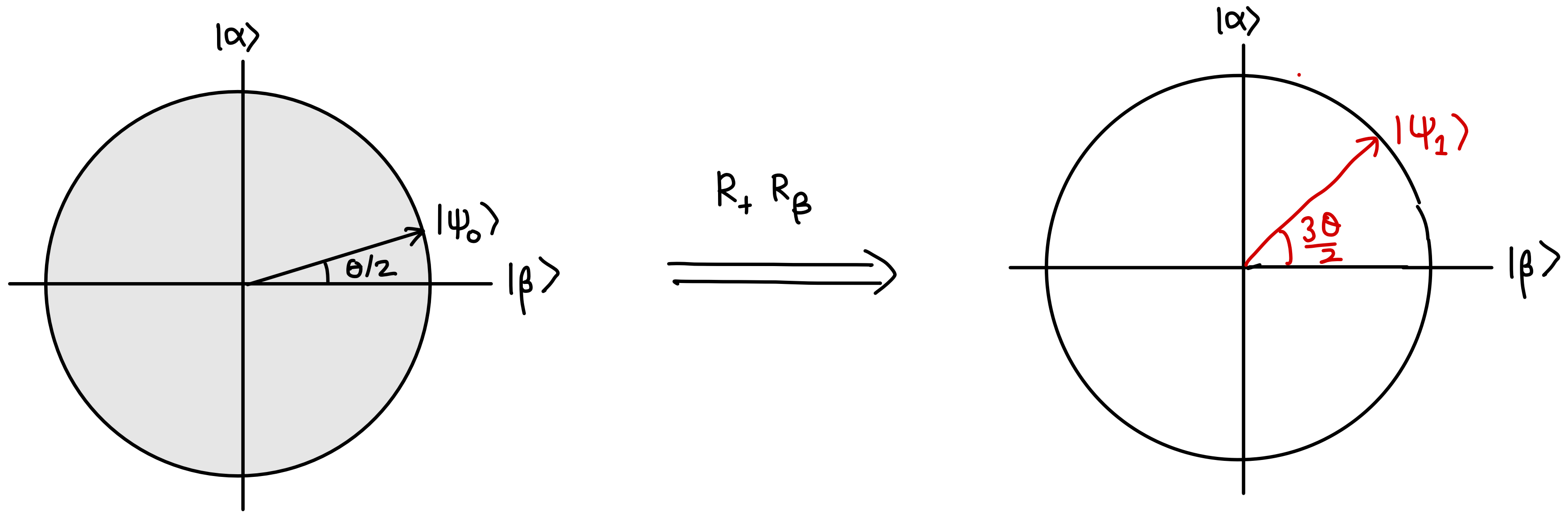


Suppose we had access to the following two unitaries

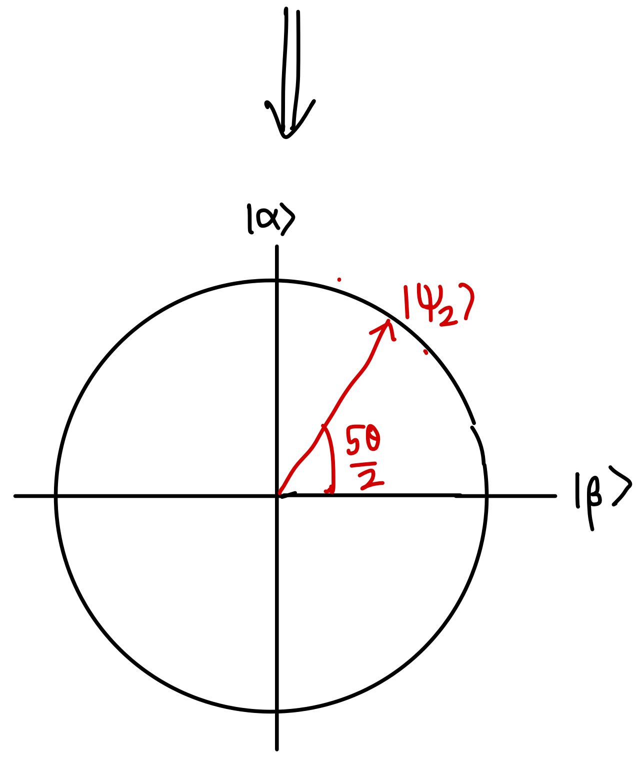
$\boxed{1}$ Reflect about $|\beta\rangle$ in the span $\{|\alpha\rangle, |\beta\rangle\}$-plane $\quad R_\beta$

$\boxed{2}$ Reflect about $|\psi_0\rangle = |+\rangle^{\otimes n}$ in span $\{|\alpha\rangle, |\beta\rangle\}$-plane $\quad R_+$

If we apply R_β and then R_+, what happens ?



Suppose we apply it again



After k-iterations, angle becomes $(2k+1)\frac{\theta}{2}$

If $(2k+1)\frac{\theta}{2} \approx \frac{\pi}{2}$ we will get a final state that has large amplitude with the marked state

Measuring the final state in the computational basis gives us x^* $\Rightarrow$ check if $f(x^*) = 1$ to distinguish

$\Rightarrow$ $k \approx \sqrt{2^n}$ iterations suffice

How do we implement the reflections ?

R_+ = reflection about $|+\rangle^{\otimes n}$ state
No queries needed, can be efficiently implemented by a circuit as well $\quad$ (see supplementary material)

R_β = reflection about $|\beta\rangle$
can be implemented by one query to U_f

NEXT TIME This is the best one can do for the search problem

Any quantum algorithm needs $\Omega(2^{n/2})$ queries $\Rightarrow NP^{O} \not\subseteq BQP^{O}$

We will introduce a general technique to prove lower bound on quantum algorithms