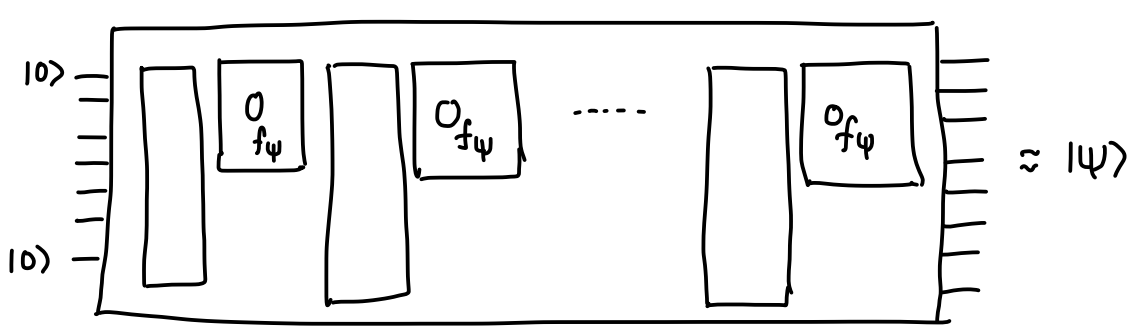
LECTURE 24 (April 17th)

TODAY State and Unitary Synthesis

RECAP

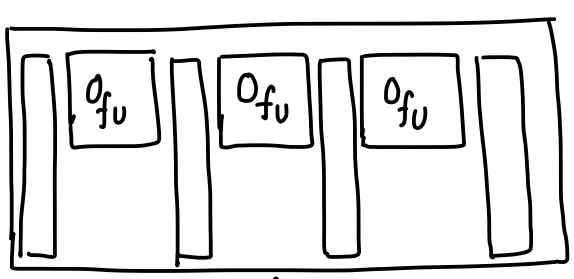
State synthesis problem Is there a quantum query algorithm, a polynomial $p(n)$ and an encoding schme that maps n-qubit states $|\psi\rangle$ to a function $f_\psi : \{0,1\}^{p(n)} \to \{0,1\}$ s.t. A makes poly(n) queries to f_ψ and outputs a good approximation to $|\psi\rangle$?



where $O_{f_\psi}|x\rangle|b\rangle \to |x\rangle|b \oplus f_\psi(x)\rangle$ for $x \in \{0,1\}^{p(n)}$

If the answer is yes, then in this sense state synthesis is no harder than computing an appropriate boolean function

Unitary Synthesis Problem Is there a quantum algorithm A, a polynomial $p(n)$ and a encoding scheme that maps n-qubit unitaries U to a boolean function $f_U : \{0,1\}^{p(n)} \to \{0,1\}$ such that A makes poly(n) queries to f_U, uses poly(n) qubits of space and approximately implements U?



Here, for every input state $|\psi\rangle$, $A|\psi\rangle \approx U|\psi\rangle$

If we can synthesize unitaries, we can also synthesize states [Why?]

First, if we allow exponentially many ancillas, then unitary synthesis is possible with one-query (also state synthesis)

The algorithm will be left to the exercises, but roughly it reads the description of the entire unitary in one query and then builds U. This is not efficient in terms of time & space

What if we restrict to space efficient algorithms? Is state or unitary synthesis possible?

Theorem (Aaronson) There is an $(n+1)$-query algorithm A and an encoding of states $|\psi\rangle$ into boolean functions $f_\psi : \{0,1\}^{poly(n)} \to \{0,1\}$ that solves the state synthesis problem.

Moreover, the algorithm is space efficient (only uses poly(n) qubits) and non-oracle gates can be implemented by poly(n) · size circuits Oracle gates f_ψ will in general require exponential sized circuits however

Proof The key idea is that a state can be decomposed in terms of conditional amplitudes which can be encoded into f

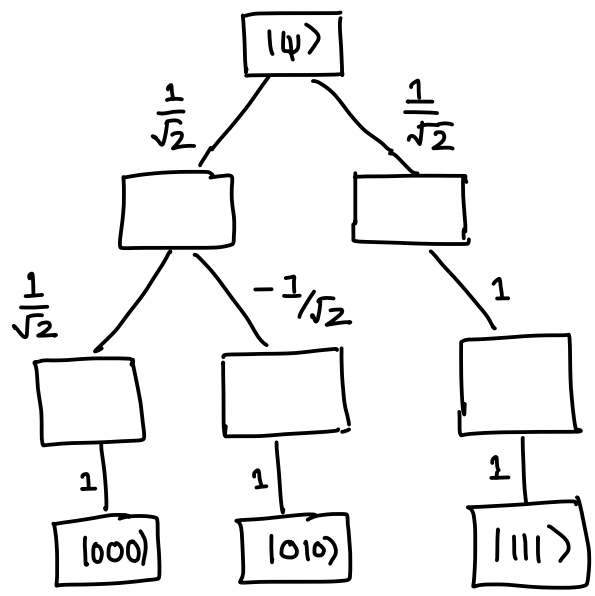
For example, $|\psi\rangle = \frac{1}{\sqrt{4}}|000\rangle - \frac{1}{\sqrt{4}}|010\rangle + \frac{1}{\sqrt{2}}|111\rangle$

$$= \frac{1}{\sqrt{2}}|0\rangle \otimes \left(\frac{1}{\sqrt{2}}|00\rangle - \frac{1}{\sqrt{2}}|10\rangle\right) + \frac{1}{\sqrt{2}}|1\rangle \otimes (|11\rangle)$$

conditional amplitudes for the first qubit

$$= \frac{1}{\sqrt{2}}|0\rangle \otimes \left(\frac{1}{\sqrt{2}}|0\rangle \otimes (|0\rangle) - \frac{1}{\sqrt{2}}|1\rangle (\otimes|0\rangle)\right) + \frac{1}{\sqrt{2}}|1\rangle \otimes \left(|1\rangle \otimes (|1\rangle)\right)$$

We can build a binary tree to represent all conditional amplitudes



For a general state $|\psi\rangle = \sum_{x \in \{0,1\}^n} \beta_x |x\rangle$

$$= \alpha_0 |0\rangle|\psi_0\rangle + \alpha_1 |1\rangle|\psi_1\rangle$$

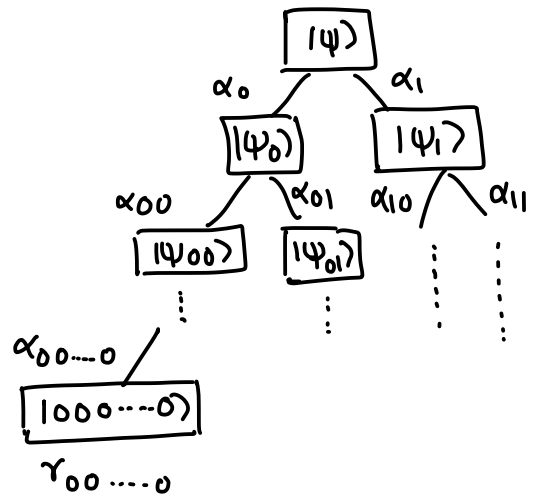
$$= \alpha_{00}|0\rangle|\psi_{00}\rangle + \alpha_{01}|1\rangle|\psi_{01}\rangle$$

$$= \sum_{x \in \{0,1\}^n} \gamma_x \cdot \alpha_{x_1} \cdot \alpha_{x_1 x_2} \cdot \alpha_{x_1 x_2 x_3} \cdots |x\rangle$$

reals

complex phase

Pictorially,



We get the amplitude of $|x\rangle$ by taking product of all α's and r on the path from $|x\rangle$ to root

The oracle will encode values of the numbers

$\alpha_{x_1 \cdots x_j}$ for all $x_1 \in \{0,1\}, \cdots x_j \in \{0,1\}, \forall j$

and $r_{x_1 \cdots x_n}$ for all $x_1 \cdots x_n \in \{0,1\}^n$

The state synthesis algorithm is the following:

1. Query oracle to get "α_0" and "α_1" ← These are real numbers upto some precision and prepare $\alpha_0|0\rangle + \alpha_1|1\rangle$

2. Controlled on first qubit, ask oracle for conditional amplitudes corresponding to the left or the right child

$$\alpha_0|0\rangle + |\text{"}\alpha_{00}, \alpha_{01}\text{"}\rangle + \alpha_1|1\rangle \otimes |\text{"}\alpha_{10}, \alpha_{11}\text{"}\rangle$$

Use an extra ancilla to prepare

$$\alpha_0|0\rangle \otimes \boxed{|\text{"}\alpha_{00}, \alpha_{01}\text{"}\rangle} \otimes (\alpha_{00}|0\rangle + \alpha_{01}|1\rangle)$$

$$+ \alpha_1|1\rangle \otimes \boxed{|\text{"}\alpha_{10}, \alpha_{11}\text{"}\rangle} \otimes (\alpha_{10}|0\rangle + \alpha_{11}|1\rangle)$$

↳ uncompute to reset them to all zeros

3. Repeat until we have $\sum_{x \in \{0,1\}^n} \alpha_{x_1} \cdot \alpha_{x_1 x_2} \cdot \alpha_{x_1 x_2 x_3} \cdots |x\rangle$

4. Query the oracle in superposition to obtain "r_x" to prepare the state

$$\sum_{x \in \{0,1\}^n} \beta_x |x\rangle \otimes |\text{"}r_x\text{"}\rangle$$ ↳ uncompute to reset to 0

The algorithm makes $(n+1)$ queries and with poly(n) bits of precision one gets a state that is exponentially close to $|\psi\rangle$

Remark A different algorithm by Irani, Natarajan, Nirkhe, Rao and Yuen gave gave a 2-query algorithm with exponentially small error but the non-oracle unitaries are not time efficient

Update A recent work by Rosenthal gave a one query algorithm for state synthesis that only uses poly(n) ancillas and all non-oracle unitaries in the circuit are time efficient as well

What about unitary synthesis?

Rosenthal showed that with $2^{n/2}$ queries one can synthesize any unitary with poly(n) ancillas

Lombardi, Ma and Wright showed that

Theorem No algorithm can synthesize a unitary with one-query and poly(n) ancillas.

What's a unitary that might be difficult to synthesize?

A Haar random unitary is one option and in fact it works
but we will choose a different candidate that is easier to analyze
In fact, we will choose a distribution over unitaries and show that
with high probability a random unitary from this distribution cannot
be synthesized with one query

Candidate Distribution that is hard to synthesize

Pick $L = 2^{n-1}$ random states with $\pm \frac{1}{\sqrt{2^n}}$ coefficients:

$$|\psi_k\rangle = \sum_{x \in \{0,1\}^n} \frac{f_k(x)}{\sqrt{2^n}} |x\rangle$$ where $f_k : \{0,1\}^n \to \{\pm 1\}$ is a uniformly random function

One can show w.h.p. $|\psi_1\rangle, \ldots |\psi_L\rangle$ are linearly independent

Our candidate unitary distribution: choose any unitary U that
maps span $\{|\psi_1\rangle, \ldots |\psi_L\rangle\}$
to span $\{|\text{bin}(1)\rangle, \ldots, |\text{bin}(L)\rangle\}$
(bin(1), ..., bin(L): binary representation)

If one can synthesize U, one can distinguish the following two cases:

Case 1 the algorithm is given as input $|\psi_k\rangle$ for k chosen uniformly at random in $[L]$

If the algorithm implements U, it can measure to check if output
is in span $\{|1\rangle, \ldots |L\rangle\}$ and accept. With probability 1, it always accepts

Case 2 the algorithm is given as input $|\psi_h\rangle = \sum_{x \in \{0,1\}^n} \frac{h(x)}{\sqrt{2^n}} |x\rangle$ where $h : \{0,1\}^n \to \{\pm 1\}$ is uniformly random

Note that projection of $|\psi_h\rangle$ on span $\{|1\rangle, \ldots |L\rangle\}$ has norm $\frac{1}{2}$ w.h.p.
so, the algorithm will accept with probability $1/2$.

NEXT TIME We will show that no algorithm can distinguish these two cases with one query
no matter which oracle f_U is chosen