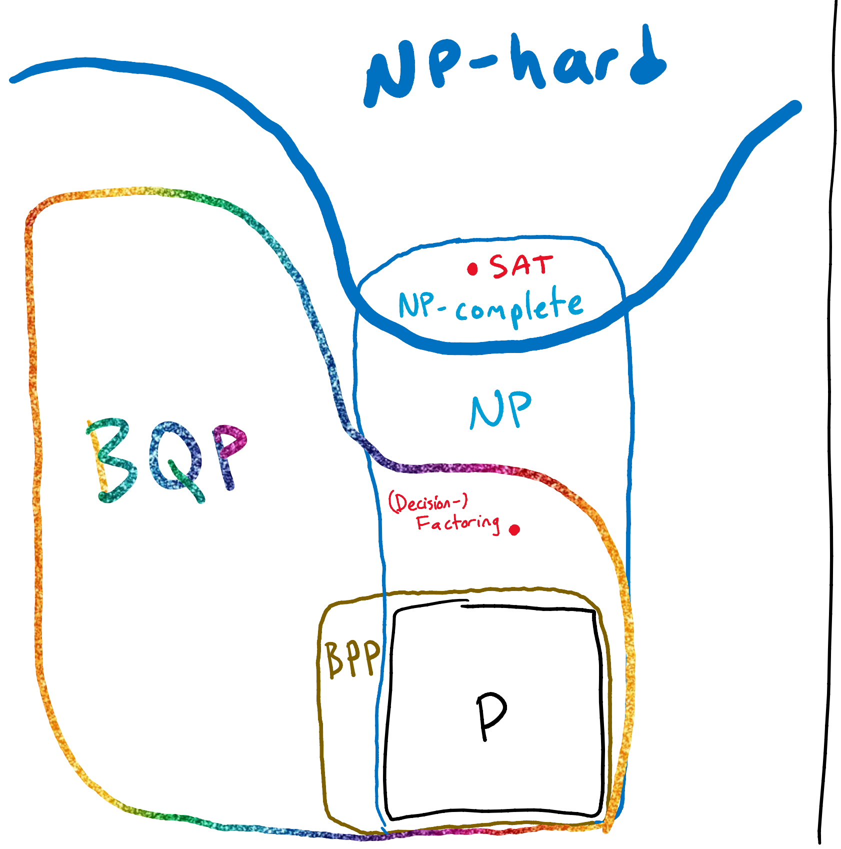
Lecture 22 — Quantum Complexity Theory

Some "complexity classes" of decision (yes/no) problems....

[you don't have to memorize their weird names]

"P" – solvable "efficiently" (poly(n) time) on a classical deterministic computer

"BPP" – solvable efficiently (with low 2-sided error) by classical probabilistic computer



Belief: BPP = P (!)

"BQP" – like BPP but w/ quantum computer

Belief: BQP $\neq$ BPP

because (Decision-) Factoring $\in$ BQP, probably $\notin$ BPP.

"NP": Problems where $\exists$ efficient verification system for yes-inputs.

⟦Also, as we argued last time...⟧

$\Leftrightarrow$ $\exists$ efficient (classical) randomized alg. R s.t. $\forall$ inputs x,

- answer is "yes" $\Rightarrow \Pr[R(x) = \text{"yes"}] > 0$
- answer is "no" $\Rightarrow \Pr[R(x) = \text{"yes"}] = 0$.

"NP-complete": Problems s.t. all problems in NP reduce to them.

<u>Belief</u>: $\exists$ problems in NP but not in BQP; e.g., SAT.

<u>Belief</u>: $\exists$ problems in BQP but not in NP; e.g.... ⟦Hmm. Well, some complexity theorists in the '90s cooked up some weird Rollercoasteresque problems that seem to fit the bill, but honestly the clearest candidate is also the tautological one...⟧ "simulate a quantum computer"

⟦Well, that's not a decision problem, but we can make an appropriate one...⟧

"Quantum-Eval. Problem":

Given quantum code "Q":

Make $X_1, X_2, \ldots, X_n$

} S instructions from {Had, NOT, CNOT, CCNOT (≡ "Add (A&B) To C")}

input

Measure/Print X_1

Output "yes" if $\Pr[Q \text{ prints } 1] \geq 3/4$

Output "no" if $\Pr[Q \text{ prints } 1] \leq 1/4$.

⟦If $\Pr[Q \text{ prints } 1]$ is between $1/4$ & $3/4$, either yes/no output is okay. Complexity theory pedants wouldn't technically call this a "decision problem" but never mind.⟧

"Quantum-Eval Prob." $\in$ BQP ✓

⟦Tautological: given Q, just run it! You get the correct answer (1 = yes, 0 = no) w. prob. $\geq 3/4$⟧

Indeed it's "BQP-complete".

⟦Any task in BQP can be reduced to it.⟧

Doesn't <u>seem</u> to be in NP....

[Why does "Quantum-Eval" not seem to be in NP? Well... say I give you some quantum code Q and claim it prints 1 with high probability. What kind of easy-to-check "witness" could I possibly include that could let you verify this claim, classically...?]

[OTOH, you *should* be able to classically verify this claim in *exponential* time. I mean, you coded a (presumably exp-time) quantum simulator back on HW3 or something...]

Fact: Quantum-Eval, and all of BQP, inside EXPTIME.

Reason: A classical alg. can compute the whole "amplitude tree" for Q in expon. time, hence exactly compute Pr[Q prints x] for any string x.

[In fact, we can do better than this....]

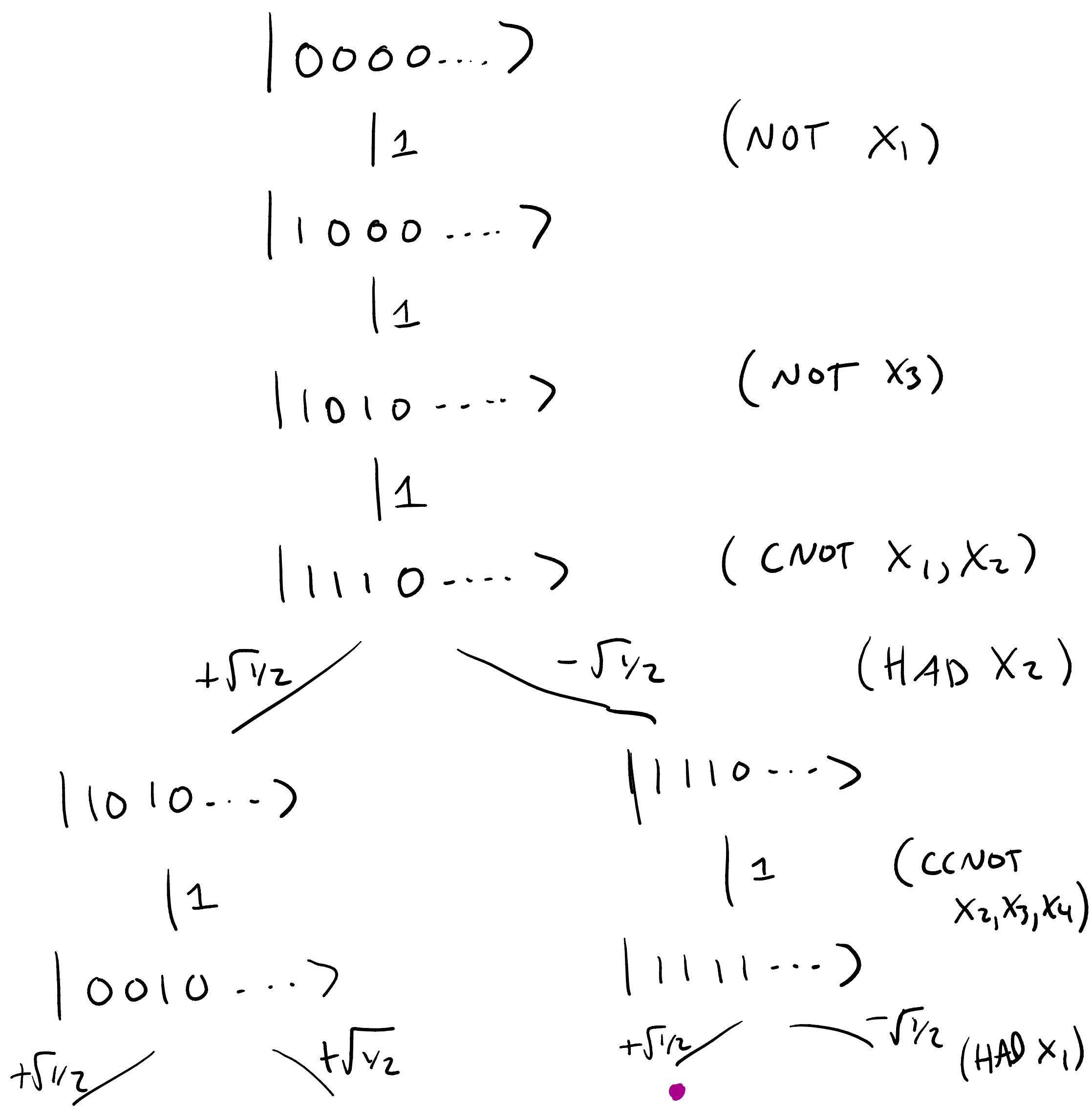
Even better fact: BQP $\subseteq$ "PSPACE"

[[Decision probs solvable using $\leq$ poly(n) memory/space]]

Theorem: Given quantum code Q with n qubits, $S \leq \text{poly}(n)$ instructions from {H, NOT, CNOT, CCNOT}, and $y \in \{0,1\}^n$, can (classically) compute final amplitude on $|y\rangle$ using $\leq \text{poly}(n)$ memory (space) (and $\leq 2^{\text{poly}(n)}$ time).

Proof: [[Don't just compute the whole final state! That could have nonzero amplitude on all 2^n possible $|x\rangle$'s, would take $\geq 2^n$ bits just to store! On the other hand, alg will "imagine" the whole amplitude tree...]]

e.g.: Q: "NOT X_1
NOT X_3
CNOT X_1, X_2 ("Add X_1 to X_2")
HAD X_2
CCNOT X_2, X_3, X_4 ("Add (X_2 AND X_3) to X_4")
. . .
"



Say Q has h many "HAD" instructions.

Tree height: $\underline{s}$. Tree has $\underline{2^h}$ leaves/paths.

[[$h \le s \le \text{poly}(n)$]]

Paths can be encoded by sequences

$p \in \{L, R\}^h$, where $L \equiv$ "left", $R \equiv$ "right"

[[e.g., in pic above, so far $h=2$, path to • encoded by (R, L)]]

<u>Fact</u>: Given $p \in \{L,R\}^h$ and Q, it's easy (poly(n) time) to compute "LeafLabel(p)", the n-bit string at leaf reached by p.

<u>Fact</u>: Product of ampls along p is
$$\left(\sqrt{\tfrac{1}{2}}\right)^h \cdot \mathrm{Sign}(p), \quad \text{where } \mathrm{Sign}(p) = p_1 p_2 \cdots p_h$$
also easy to compute.

<u>Fact</u>: Final ampl. on $|y\rangle$ is
$$\left(\sqrt{\tfrac{1}{2}}\right)^h \cdot \sum_{\substack{p \in \{L,R\}^h : \\ \mathrm{LeafLabel}(p) = y}} \mathrm{Sign}(p).$$

Computable via... "ampl := 0
for $p \in \{L,R\}^h$:
if LeafLabel(p) = y:
ampl += Sign(p)
ampl *= $(\sqrt{1/2})^h$."

efficient to compute

loops for 2^h times,
<u>But</u> <u>reuses</u> space,
Only uses h bits.

$\therefore$ ampl. on $|y\rangle$ computable in poly(n) memory/space.

Can we do better?

[[You might think it's hard too. And that the Hads are the problem...]]

<u>Obvious</u>: If no Had gates, can efficiently simulate classically [[since Q is just classical code!]]

[[Non-obv.]]

"Gottesman-Knill Thm": If no CCNOT gates [[Just H, NOT, CNOT]], can compute any amplitude [[and simulate meas. results]] efficiently classically!

[[So it's not exactly Had gates per se that are difficult... nor CCNOT per se... hm...]]

<u>Rem</u>: All of Q.C. can be done with H, NOT, CNOT, & $\mathrm{Rot}_{\pi/8}$ [[on one qubit!]]

[[But actually, the PSPACE alg. is arguably overkill. To simulate a Q.C. algorithm classically, you don't <u>have</u> to be able to compute the amplitudes/measurement probabilities. Perhaps you could just <u>achieve</u> (approximately) them...]]

<u>Theorem/homework</u>: Given quantum code Q outputting 1 bit, can efficiently convert to classical randomized code R such that...

- If $\Pr[Q \text{ outputs } 1] = \frac{1}{2} + q$ $\quad (-\frac{1}{2} \leq q \leq \frac{1}{2})$
  then $\Pr[R \text{ outputs } 1] = \frac{1}{2} + q/2^h$ $\quad$ (h = # Had ops in Q)

[This is another example of "solving" the simulation task so badly as to be practically worthless... but... it's not <u>nothing</u>.]

E.g. this "solves" the Quantum-Eval prob. as follows:

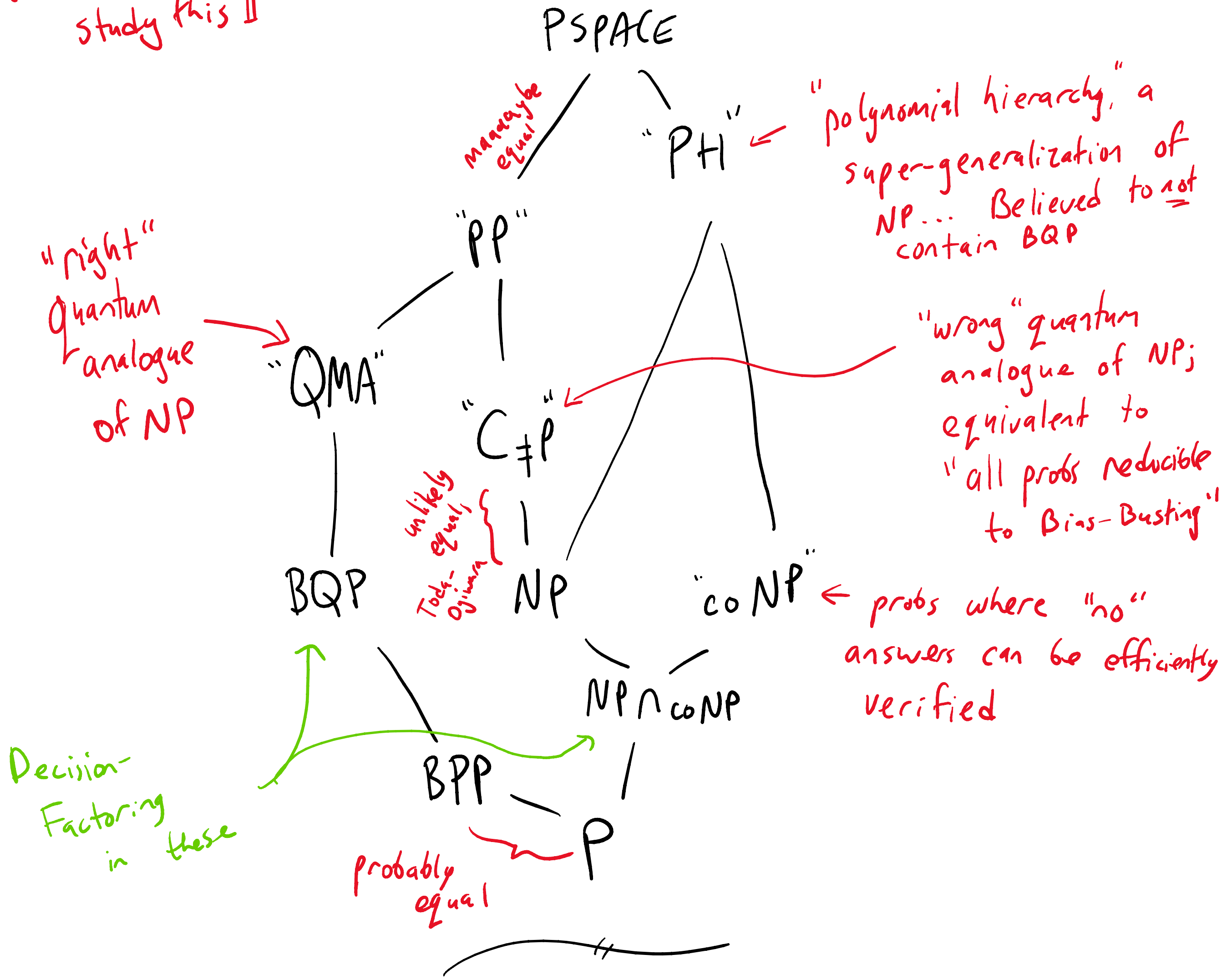
- Q a "yes" input $\Rightarrow \Pr[Q=1] \geq \frac{3}{4}$
  $\Rightarrow \Pr[R=1] \geq \frac{1}{2} + \frac{1}{2^{h+2}} > \frac{1}{2}$.
- Q a "no" input $\Rightarrow \Pr[Q=1] \leq 1/4$
  $\Rightarrow \Pr[R=1] \leq \frac{1}{2} - \frac{1}{2^{h+2}} < \frac{1}{2}$.

$\hookrightarrow$ R gives correct answer w. prob $> \frac{1}{2}$.

[intriguing but practically useless; the "wrong defn" of a successful randomized alg.]

$\Rightarrow$ Quantum-Eval, BQP in complexity class "PP"

[Don't need to study this]



[So... on one hand, can kinda "exponentially badly" efficiently simulate Quantum-Eval. Also know that achieving "$\Pr[Q=1] > 0 \Rightarrow \Pr[R=1] > 0$ & $\Pr[Q=1] = 0 \Rightarrow \Pr[R=1] = 0$" likely impossible, by Bias-Busting, Toda-Ogiwara stuff.

Next time: so... if you believe classical computers must be terrible at sim'ing quantum ones, maybe we can demonstrate "Quantum Advantage"™!]