CMSC 858L: Quantum Complexity

Instructor: Daniel Gottesman

Spring 2023

20 More About PP and PostBQP

20.1 References

PostBQP is defined and its relation to PP given in Aaronson, "Quantum Computing, Postselection, and Probabilistic Polynomial-Time," quant-ph/0412187. PH is a standard classical complexity class and can be found in standard textbooks, along with Toda's theorem.

The definition of PostBPP and the relationship to BPP_{path} comes from Bravyi, DiVincenzo, Oliveira, and Terhal, "The Complexity of Stoquastic Local Hamiltonian Problems," quant-ph/0606140. BPP_{path} and its inclusion in BPP^{NP} is from Han, Hemaspaandra, and Thierauf, "Threshold Computation and Cryptographic Security," SIAM J. Computing, vol. 26, p. 59 (1997). However, the proof here is simplified from those papers since it avoids the detour into BPP_{path}.

20.2 PP= PostBQP

Recall that last time we defined PostBQP and PP and showed that PostBQP $\subseteq$ PP. Now let us prove that they are actually equal.

Since the definition of PP involves a classical polynomial-time randomized algorithm, we can assume that the random bits used in the algorithm come from some predetermined string. Given a particular string x of n random bits, the algorithm outputs $f(x)$. For a "yes" instance, the number of values of x for which $f(x) = 1$ must be $> 2^{n-1}$ (so the acceptance probability for random x is greater than 1/2), and for a "no" instance, the number of values x for which $f(x) = 0$ is $> 2^{n-1}$. We need to find a polynomial-time quantum algorithm with post-selection to decide this same language.

Let $s = |\{x|f(x) = 1\}|$. Then as a first step in the quantum algorithm, prepare the state $\sum_x |x\rangle|f(x)\rangle$. Then perform the Hadamard on all qubits of the first register, which gives us

$$\sum_x \sum_y (-1)^{x\cdot y}|y\rangle|f(x)\rangle. \tag{1}$$

Now we post-select on the first register being $y = 0$. We now have for the last qubit the state

$$|\psi\rangle = \frac{(2^n - s)|0\rangle + s|1\rangle}{\sqrt{(2^n - s)^2 + s^2}}. \tag{2}$$

If s is very different from 2^{n-1}, we can easily determine whether it is larger or smaller than 2^{n-1} by making a number of copies of $|\psi\rangle$ and measuring them to see whether 0 or 1 is the most common. In the hard case where $s \approx 2^{n-1}$, the state is very close to $|0\rangle + |1\rangle$ and we need to determine whether the amplitude of $|1\rangle$ is slightly larger or smaller than the amplitude of $|0\rangle$.

Essentially we need to distinguish the two states $|\psi_+\rangle = \sqrt{(1-\epsilon)/2}|0\rangle + \sqrt{(1+\epsilon)/2}|1\rangle$ and $|\psi_-\rangle = \sqrt{(1+\epsilon)/2}|0\rangle + \sqrt{(1-\epsilon)/2}|1\rangle$ (although we don't know the exact value of ϵ). Now, these are non-orthogonal states and are in fact very close together, so normal quantum mechanics cannot distinguish them except with very small probability. But one thing you *can* do in standard quantum mechanics is something called *unambiguous state discrimination*, in which you do a POVM (generalized measurement) which has three outcomes: the two states you want to distinguish and a third "don't know" outcome. When the unambiguous state discrimination succeeds, you definitely know which state you have, but when it fails, you learn nothing. If the two states are very close together, as they are here, the probability of the "don't know" outcome is very high, but when we have post-selection available, we can post-select on the cases where we don't get that outcome!

To be more specific, we can imbed the two-dimensional Hilbert space into a 3-dimensional Hilbert space and measure by projecting on the outcomes

$$|A_+\rangle = \alpha\sqrt{(1-\epsilon)/2}|0\rangle - \alpha\sqrt{(1+\epsilon)/2}|1\rangle + i\alpha\sqrt[4]{1-\epsilon^2}|2\rangle, \tag{3}$$

$$|A_+\rangle = \alpha\sqrt{(1+\epsilon)/2}|0\rangle - \alpha\sqrt{(1-\epsilon)/2}|1\rangle + i\alpha\sqrt[4]{1-\epsilon^2}|2\rangle, \tag{4}$$

$$|?\rangle = \frac{1}{\sqrt{2}}\gamma(|0\rangle + |1\rangle) + i\delta|2\rangle \tag{5}$$

where $\alpha^2 = 1/(1+\sqrt{1-\epsilon^2})$ and γ and δ are chosen so that $|?\rangle$ is chosen to be normalized and orthogonal to $|A_\pm\rangle$.

Now, we don't know ϵ in the actual system, so suppose we perform the above POVM designed for ϵ but instead we have the state $|\psi'_\pm\rangle = \sqrt{(1\mp\epsilon')/2}|0\rangle + \sqrt{(1\pm\epsilon')/2}|1\rangle$. Then the probabilities of the outcomes $\pm$ given input $|\psi'_+\rangle$ are:

$$\begin{aligned}
\text{Prob}(+) &= |\langle\psi'_+|A_+\rangle|^2 && (6)\\
&= \alpha^2|\sqrt{(1-\epsilon)(1-\epsilon')}/2 - \sqrt{(1+\epsilon)(1+\epsilon')}/2|^2 && (7)\\
&= \frac{\alpha^2}{2}\left(1+\epsilon\epsilon' - \sqrt{(1-\epsilon^2)(1-\epsilon'^2)}\right) && (8)\\
&= \frac{\alpha^2}{2}\left[1+\epsilon\epsilon' - 1 + (\epsilon^2+\epsilon'2)/2 + O(\epsilon^4+\epsilon'^4+\epsilon^2\epsilon'^2)\right] && (9)\\
&= \frac{\alpha^2}{4}\left[(\epsilon+\epsilon')^2 + O(\epsilon^4+\epsilon'^4+\epsilon^2\epsilon'^2)\right] && (10)\\
\text{Prob}(-) &= |\langle\psi'_+|A_-\rangle|^2 && (11)\\
&= \alpha^2|\sqrt{(1+\epsilon)(1-\epsilon')}/2 + \sqrt{(1-\epsilon)(1+\epsilon')}/2|^2 && (12)\\
&= \frac{\alpha^2}{2}\left(1-\epsilon\epsilon' - \sqrt{(1-\epsilon^2)(1-\epsilon'^2)}\right) && (13)\\
&= \frac{\alpha^2}{4}\left[(\epsilon-\epsilon')^2 + O(\epsilon^4+\epsilon'^4+\epsilon^2\epsilon'^2)\right] && (14)\\
& && (15)
\end{aligned}$$

For the initial state $|\psi'_-\rangle$, the probabilities are switched. When ϵ and ϵ' are small, the ratio of the probability of the $+$ outcome to the $-$ outcome is about $(\epsilon\pm\epsilon')^2/(\epsilon\mp\epsilon')^2$ for the $|\psi'_\pm\rangle$ state.

In particular, if we post-select on getting one of the outcomes $+$ or $-$ and $\epsilon' = c\epsilon$, then the probability of the two outcomes are about

$$\frac{(1\pm c)^2}{2(1+c^2)}, \tag{16}$$

which is $1/2 + \Theta(1)$ if $c = \Theta(1)$. Thus, if we perform the above POVM n times for $\epsilon = 2^{-i}$ for i running from 1 to n, we will always get a $\Theta(1)$ chance of distinguishing the two states for one of the values of i. We can repeat each measurement a number of times to get the desired level of accuracy for the overall algorithm. This proves that PP $\subseteq$ PostBQP.

20.3 Polynomial hierarchy

Now, the next thread is bring in classical knowledge about PP in the form of Toda's theorem. To explain what Toda's theorem is, I first have to define a number of new complexity classes. (Infinitely many, actually. But they are closely related.)

Recall that NP can be defined as the set of languages for which there exists a witness such that the checking circuit accepts it. The checking circuit $C(x, w)$ is a Boolean function of the instance x and the prospective witness w. So phrasing the definition of NP in a symbolic way, we get that $L \in$ NP when there is some polynomial time computable C such that

$$x \in L \quad \Longleftrightarrow \quad \exists w \text{ s.t. } C(x, w) = 1. \tag{17}$$

We can also define co-NP similarly: $L \in$ co-NP when there is some polynomial time computable C such that

$$x \in L \quad \Longleftrightarrow \quad !\exists w \text{ s.t. } C(x, w) = 1; \tag{18}$$

or equivalently, $L \in$ co-NP when there is some polynomial time computable $\overline{C} = 1 \oplus C$ such that

$$x \in L \quad \Longleftrightarrow \quad \forall w \ \overline{C}(x, w) = 1. \tag{19}$$

That is, NP is associated with statements of the form "there exists" and co-NP is associated with statements of the form "for all".

What if we have both quantifiers? For instance, we can define a complexity class Σ_2^p as the set of languages L for which there exists some polynomial time computable C such that

$$x \in L \quad \Longleftrightarrow \quad \exists u \text{ s.t. } \forall w \ C(x, u, w) = 1. \tag{20}$$

Then Σ_2^p contains both NP and co-NP. We could also put the quantifiers the other way to get a class Π_2^p containing languages L for which there is C such that

$$x \in L \quad \Longleftrightarrow \quad \forall u \exists w \text{ s.t. } C(x, u, w) = 1. \tag{21}$$

This also contains both NP and co-NP. Note that Π_2^p contains a language L iff Σ_2^p contains the complement of L. That is, $\Pi_2^p = \text{co} - \Sigma_2^p$. This is because NOT switches the quantifiers $\forall$ and $\exists$: $!\forall x, B(x) = \exists y ! B(y)$.

The classes Σ_2^p and Π_2^p are not as interesting as NP and co-NP, they still contain a number of interesting problems. Whereas NP problems tend to be relatively straightforward optimizations, the "best" of something, these two involve more complicated optimzations where you have two different kinds of optimizations working against each other. For instance, the traveling salesman is NP-complete: Does there exist a path of length less than L? But suppose you ask, is the length of the shortest path exactly L? That involves 2 quantifiers: Does there *exist* a path of length L? And, *for all* other paths, are they at least as long as L? This is an Σ_2^p question.

We can continue adding quantifiers to get new classes, although they become less and less practically relevant:

Definition 1. *Let a Σ_n^p statement $S(\cdot)$ be one of the form $\exists u \ P(u, \cdot)$, where $P(u, \cdot)$ is a Π_{n-1}^p statement. Let a Π_n^p statement $P(\cdot)$ be one of the form $\forall u \ S(u, \cdot)$, where $S(u, \cdot)$ is a Σ_{n-1}^p statement. Σ_0^p and Π_0^p statements are of the form $C(\cdot) = 1$ for some polynomial-time computable C.*

Then Σ_n^p is the class of languages L for which $x \in L$ iiff x satisfies a Σ_n^p statement. Π_n^p is the class of languages L for which $x \in L$ iiff x satisfies a Π_n^p statement. The polynomial hierarchy

$$PH = \cup_{n=1}^{\infty} \Sigma_n^p. \tag{22}$$

Σ_n^p and Π_n^p statements are defined using alternating quantifiers. We usually define them for $n \geq 1$, but you can think of $\Sigma_0^p = \Pi_0^p =$ P. Repeating the same quantifier twice in a row just allows you to combine them into a single quantifier with a larger variables (e.g., $\forall x \ \forall y = \forall (x, y)$). Note that $\Sigma_{n-1}^p, \Pi_{n-1}^p \subseteq \Sigma_n^p, \Pi_n^p$ (which is why we don't need to take the union over both Σ and Π to define PH.

One critical fact about the polynomial hierarchy is that either all of the Σ and Π are distinct or at some point they become all the same:

Theorem 1. *If $\Sigma_n^p = \Pi_n^p$ for some $n \geq 1$, then $\Sigma_n^p = \Sigma_m^p$ for any $m > n$.*

If the hypothesis of the theorem is true, this is referred to as "the polynomial hierarchy collapses to the nth level."

Proof. Suppose $\Sigma_n^p = \Pi_n^p$. Then given some language L in Σ_{n+1}^p, $x \in L$ iff x satisfies $\forall u P(u, \cdot)$. But $P(u, \cdot) = \exists v S(u, v, \cdot)$ is a Π_n^p statement, so by the hypothesis, there is a Σ_n^p statement that is true whenever $P(u, \cdot)$ is true, and only then (since it must define the same language). Assume the Σ_n^p statement is $\forall v P'(u, v, \cdot)$, where $P'(u, v, \cdot)$ is a Π_{n-1}^p statement. Then the original Σ_{n+1}^p statement is equivalent to $\forall u \forall v P'(u, v, \cdot)$. We can combine the two $\forall$ quantifiers, giving a Σ_n^p statement $\forall (u, v) P'(u, v, \cdot)$. Thus $\Sigma_{n+1}^p = \Sigma_n^p$.

Essentially the same argument shows that $\Pi_{n+1}^p = \Pi_n^p$, and therefore $\Pi_{n+1}^p = \Pi_n^p = \Sigma_n^p = \Sigma_{n+1}^p$. By induction, it follows that all Σ_m^p and Π_m^p for $m \geq n$ are equal. $\square$

Theorem 2 (Toda's theorem)**.** *$PH \subseteq P^{PP}$.*

The proof of this is somewhat involved, so I will not go through it.