CMSC 858L: Quantum Complexity

Instructor: Daniel Gottesman

Spring 2023

19 The Complexity Class PP

19.1 References

PP is a standard classical complexity class, and you can find out more about in any standard classical complexity textbook. For the proof that QMA $\subseteq$ PP, see Marriott and Watrous, "Quantum Arthur-Merlin Games," cs/0506068 and Vidick and Watrous, "Quantum Proofs," arXiv:1610.01664. PostBQP is defined and its relation to PP given in Aaronson, "Quantum Computing, Postselection, and Probabilistic Polynomial-Time," quant-ph/0412187.

19.2 Unbounded error polynomial time

If we remove the "B" ("Bounded") from BPP, we get PP, which is defined as follows:

Definition 1. *Let PP be the set of languages L such that there exists a polynomial-time randomized algorithm $A(x)$ (i.e., a uniform family of polynomial-size circuits including the ability to generate random bits) with the following properties: For any instance x,*

1. *If $x \in L$, then* $\mathrm{Prob}(A(x) = 1) > 1/2$.
2. *If $x \notin L$, then* $\mathrm{Prob}(A(x) = 0) > 1/2$.

In other words, it is like BPP, but the difference between the acceptance probabilities of the "yes" and "no" instances can be arbitrarily small. Note that PP $\subseteq$ PSPACE: This is what we showed in problem set 1, prob. 1c (up to some slight differences in definitions).

What can we do in PP? It turns out to be much more powerful than BPP. For instance, NP $\subseteq$ PP: Here is a PP algorithm to decide any NP problem. Suppose the witness for a specific instance would be of size n. Choose a random witness candidate w of length n and check if it is a valid witness. If it is, return 1 ("yes"). Otherwise, return 1 with probability $1/2 - 2^{-(n+1)}$ and 0 with probability $1/2 + 2^{-(n+1)}$. If we have a "no" instance, the witness will never pass the check and so we get 0 with probability greater than $1/2$, as required. If it is a "yes" instance, the chance of randomly picking a valid witness is at least 2^{-n} (more if there is more than one valid witness), so the probability of returning 1 is at least

$$2^{-n} + (1 - 2^{-n})(\frac{1}{2} + 2^{-(n+1)}) = \frac{1}{2} + 2^{-n} - 2^{-(2n+1)} > \frac{1}{2}. \tag{1}$$

PP also lets us add up exponentially large sums:

Theorem 1. *Let L be the language consisting of instances (f, S, δ), where $f : \mathbb{Z}_2^n \to [0, 1]$ is a polynomial-time computable function (specified, for instance, as a circuit), $\delta = \Omega(\exp(-\mathrm{poly}(n)))$, $S \geq \delta$, and we are promised that either*

1. *$\sum_x f(x) > S$ ("yes" instances), or*

2. $\sum_x f(x) < S - \delta$ *("no" instances).*

Then $L \in PP$.

Note that since δ is allowed to be exponentially small, the promise is a very mild one and can also be handled by having the output of f be rational numbers with reasonable denominators (no more than exponential in n).

Proof. Here is a randomized algorithm whose probability of acceptance equals the value of the sum divided by 2^n, which is equal to the average value of $f(x)$: Choose a uniform random x and return 1 with probability $f(x)$. For instances where $S = 2^{n-1}(1+\delta)$, this algorithm fits the condition for L to be in PP.

For other values of S, we can slightly modify the algorithm as follows: If $S > 2^n$, return $A(f, S, \delta) = 1$, since in this case, the answer is automatic since $0 \leq f(x) \leq 1$ for all x. Otherwise, choose random x. Round $f(x)$ off to $m > n + \log(2/\delta)$ bits of precision to get $\tilde{f}$ and let $S' = S - \delta/2$. Return $A(f, S, \delta) = 1$ with probability

$$\frac{1}{2}(1 + \tilde{f}(x) - S'/2^n) \tag{2}$$

and otherwise return $A(f, S, \delta) = 0$. (This probability is between 0 and 1 because of the constraints on $f(x)$ and S.) What is the overall acceptance probability?

$$\text{Prob}(A(f, S, \delta) = 1) = 2^{-n} \sum_x \frac{1}{2}(1 + \tilde{f}(x) - S'/2^n) = \frac{1}{2} + 2^{-(n+1)} \left(\sum_x \tilde{f}(x) - S' \right). \tag{3}$$

Thus, this is accepted with probability greater than 1/2 if $\sum \tilde{f} > S'$ and less than 1/2 if $\sum \tilde{f} < S'$.

Now $|\sum \tilde{f} - \sum f| < 2^n 2^{-m} < \delta/2$. Thus, in the "yes" instance, $\sum \tilde{f} > S - \delta/2 = S'$ and the algorithm accepts with probability greater than 1/2 as desired. In the "no" instance, $\sum \tilde{f} < S - \delta/2 = S'$, so the algorithm accepts with probability less than 1/2, again as desired. □

Note that this immediately applies to sums of functions which take on values in other bounded ranges by simply shifting and rescaling those ranges into $[0, 1]$.

19.3 PP and Quantum Classes

You may recall that the proof that BQP $\subseteq$ PSPACE relied primarily on the fact that you can add up exponential sums in PSPACE. Thus, this same argument proves that in fact BQP $\subseteq$ PP.

Let's review it: If we have a quantum circuit $U_n, \ldots, U_0$, then the ampltiude to start at b and transition to a through the circuit is

$$\langle a| \prod_{i=0}^{n} U_i |b\rangle = \sum_{c_0, \ldots, c_{n-1}} [U_0]_{a,c_0} \prod_{i=1}^{n-1} [U_i]_{c_{i-1},c_i} [U_n]_{c_{n-1},b}. \tag{4}$$

Each term in the sum is computable in polynomial time, so thm. 1 applies and it is possible to compute this sum in PP up to exponential accuracy.

However, this tells us the total probability of transitioning from initial state b to a, where a and b specify the whole state. If we are interested in the marginal probability of outcome a_0 where the other output qubits could have any value a_1, we must compute

$$\text{Prob}(a_0) = \sum_{a_1} |\langle a_0 a_1| \prod_{i=n}^{0} U_i |b\rangle|^2 \tag{5}$$

$$= \sum_{a_1} \sum_{c_0, \ldots, c_{n-1}} \sum_{c'_0, \ldots, c'_{n-1}} [U_0]_{a_0 a_1, c_0} \prod_{i=1}^{n-1} [U_i]_{c_{i-1},c_i} [U_n]_{c_{n-1},b} [U_0]^*_{a_0 a_1, c'_0} \prod_{i=1}^{n-1} [U_i]^*_{c'_{i-1},c'_i} [U_n]^*_{c'_{n-1},b}. \tag{6}$$

This is again an exponential sum and can be computed in PP.

Note that we don't yet know that PP is closed under subroutines (although it is), so it wasn't sufficient to just say it is a sum of a sum, but we need to note that the whole thing can be rewritten as a single big sum.

Also, for simplicity, let us assume that all the gates have only real matrix elements (which we have seen is equivalent to standard quantum computation), and that way we don't have to deal with complex sums, just real sums which can be rescaled so that each term is in the range $[0, 1]$, as discussed above.

The upshot is that we can compute marginal probability distributions in PP up to high accuracy:

Theorem 2. *Given a polynomial-size quantum circuit Q with input in the standard basis, using a PP algorithm, it is possible to determine if the marginal probability of output b on some qubits is greater than S or less than $S - \delta$ (for $\delta = \Omega(\exp(-\text{poly}(n)))$).*

This is a decision version of a strong simulation (with high accuracy) of the quantum circuit.

Given the strength of this result, we can prove even more:

Theorem 3. *QMA $\subseteq$ PP.*

Proof. Given a language L in QMA, let us show it is in PP. Using witness amplification, we can take a verification circuit using an m-qubit witness and amplify its success probability, so that if the instance is a "no" instance, all witnesses are rejected with probability at least $1 - 2^{-(m+1)}$ and in a "yes" instance, there is a witness that is accepted with probability at least $1 - 2^{-(m+1)}$.

Then consider the following quantum algorithm: Generate a maximally mixed state on m qubits as a witness and run the (amplified) verification circuit on it. Then accept if the circuit accepts and reject if it rejects.

In a "no" instance, the circuit accepts with probability at most $2^{-(m+1)}$ since it accepts with at most that probability for all witnesses. In a "yes" instance, the maximally mixed state can be decomposed as a mixture of a valid witness (with weight 2^{-m}) and other states which may or may not be valid witnesses. In this case, the valid witness is accepted with probability $1 - 2^{-(m+1)}$. The other witnesses might be rejected as much as 100% of the time. The total acceptance probability for the "yes" instance is thus at least

$$2^{-m}(1 - 2^{-(m+1)}) = 2^{-m} - 2^{-(2m+1)} = 2^{-(m+1)} + (2^{-(m+1)} - 2^{-(2m+1)}). \tag{7}$$

Now, by thm. 2, using a PP algorithm, we can determine if the quantum circuit is accepted with probability at least $S = 2^{-m} - 2^{-(2m+1)}$ or at most $S - \delta$, $\delta = 2^{-(m+1)} - 2^{-(2m+1)}$. This decides L.

□

19.4 PostBQP

Another consequence of thm. 2 is that we can simulate another quantum class, known as PostBQP. PostBQPis the class of decision problems solvable by polynomial-size quantum circuits with the additional capability of post-selection, where we can force measurements to have certain outcomes.

Definition 2. *PostBQP is the class of languages L for which there exists a quantum algorithm Q with the following properties:*

1. *Q has polynomial size,*
2. *Q measures two qubits A (with outcome a) and B (with outcome b),*
3. *For all instances,* $\text{Prob}(a = 1) > 0$,
4. *If $x \in L$, then* $\text{Prob}(b = 1|a = 1) \geq 2/3$,
5. *If $x \notin L$, then* $\text{Prob}(b = 0|a = 1) \geq 2/3$.

(Note that there are certain subtleties in the choices of universal gate sets allowed in the definition of PostBQP, but that any standard finite approximately universal set such as $\{H, R_{\pi/8}, CNOT\}$ is fine.)

The measurement of a single qubit is sufficient to allow post-selection on very complicated or very improbable conditions. For instance, we can easily solve any NP problem using post-selection: Create a superposition of all possible witnesses and run the verification circuit on the witness. Then post-select on the measurement of the output qubit of the verification circuit and measure (without post-selection) the qubits containing the prospective witness. The only portion of the state that survives the post-selection is the portion containing valid witnesses, and therefore conditioned on the post-selected qubit, the other measurement gives us an actual valid witness for the NP problem.

Post-selection is very powerful but is not intended to be physically realistic. While you can do it in a real system, it requires repeated trials to succeed, and post-selecting on an exponentially unlikely outcome would require exponentially many trials to have a good chance of succeeding once. However, one model of how quantum mechanics would work in the presence of time travel allows post-selection using time machines. More relevantly, though, studying PostBQP allows us to prove many interesting things about other complexity classes (although again some of them like PP are not exactly physically realistic either).

The first observation about PostBQPis that it is also inside PP:

Theorem 4. *PostBQP $\subseteq$ PP.*

Proof. Consider a quantum circuit Q satisfying the conditions for PostBQP. Then it has outputs abc, where a and b are the two qubits measured by Q and c is the state of the remaining qubits. Now,

$$\text{Prob}(b = i | a = 1) = \text{Prob}(a = 1, b = i)/\text{Prob}(a = 1), \tag{8}$$

and in particular, in a "yes" instance, $\text{Prob}(a = 1, b = 1) > \text{Prob}(a = 1, b = 0)$, whereas in a "no" instance, $\text{Prob}(a = 1, b = 0) > \text{Prob}(a = 1, b = 1)$. The difference in either case is at least $\delta = 1/3\text{Prob}(a = 1)$.

If we use a standard set of gates in our circuits (and this is the subtlety noted above), then any non-zero value of $\text{Prob}(a = 1)$ that can be achieved is $\Omega(\exp(-\text{poly}(n)))$. Since each of $\text{Prob}(a = 1, b = 1)$ and $\text{Prob}(a = 1, b = 0)$ is an exponential sum (as discussed above in the proof of thm. 2), the difference $\text{Prob}(a = 1, b = 1) - \text{Prob}(a = 1, b = 0)$ is also an exponential sum, and thus a PP algorithm can decide if it is larger than $\delta/2$ or smaller than $-\delta/2$, deciding the PostBQP language under consideration. □

Remarkably, the converse is true as well:

Theorem 5. *PP $\subseteq$ PostBQP, and thus PostBQP = PP.*

One consequence of this theorem is that we can easily prove some standard properties of PP using PostBQP. Suppose you have two language $L_1, L_2 \in$ PP. It is not straightforward to prove directly that $L_1 \cap L_2 \in$ PP, but if you have PostBQP algorithms for both L_1 and L_2, it is straightforward to run both algorithms and accept only if both accept, showing that $L_1 \cap L_2 \in$ PostBQP.