CMSC 858L: Quantum Complexity

Instructor: Daniel Gottesman

Spring 2023

4 QMA, PSPACE, and universal quantum circuits

4.1 QMA

To define a quantum analog of NP, we need to both handle the random aspect of quantum measurement and to convert everything to quantum states and circuits. The solution to add randomness is essentially the same as for BPP and BQP: Namely, use a checking circuit with a probability 2/3 of giving the correct answer. When we do this for a classical complexity class, we get a class known as MA, for "Merlin-Arthur." This class comes with a story attached to it: Arthur is a king, intelligent but merely mortal. He can do any polynomial-time computation. However, he has an advisor Merlin, a powerful wizard who can perform any computation. Unfortunately, Merlin can be a bit tricky and Arthur does not trust him, so he insists that whenever Merlin gives him an answer to any question that Merlin be able to prove that the answer is true. For instance, Arthur could ask Merlin "Is this Boolean formula satisfiable?" If Merlin answers "yes," then he can follow that up with a proof by giving a satisfying assignment for the formula and Arthur can then check that using his own computational abilities. This certainly allows Merlin to convince Arthur of the "yes" answer for any language in NP. However, Arthur is also satisfied with a statistical proof, so we allow the checking algorithm to be randomized, giving the potentially slightly larger class MA.

Making MA quantum (giving QMA, "quantum Merlin-Arthur") is actually a bit more challenging because there is a choice we have to make: We have two classical things in the definition of NP: A classical checking circuit and a classical witness. Which of these should we make quantum? Making just the witness quantum doesn't make sense since there is no way for the classical circuit to interact with it in a quantum way, so we should at least make the checking circuit quantum.

But should we allow the witness to be a quantum state or leave it as a classical bit string? These give what apparently two different complexity classes, QMA and QCMA.

Definition 1. *Fix a particular efficiently computable approximately universal gate set $\mathcal{G}$. Let QCMA be the set of languages L such that there exists a uniform family of polynomial-size quantum circuits $Q_{n,m}$ (with gates drawn from $\mathcal{G}$) that takes two inputs (x, w) with the following properties: For any instance x,*

1. *If $x \in L$, then exists w_x with $|w_x| = O(\text{poly}(|x|))$ and $\text{Prob}(M_{Q_{|x|,|w_x|}}(|x\rangle \otimes |w_x\rangle) = 1) \geq 2/3$.*
2. *If $x \notin L$, then for any w_x with $|w_x| = O(\text{poly}(|x|))$, $\text{Prob}(M_{Q_{|x|,|w_x|}}(|x\rangle \otimes |w_x\rangle) = 0) \geq 2/3$.*

In the first case, w_x is the witness *for $x \in L$.*

Definition 2. *Fix a particular efficiently computable approximately universal gate set $\mathcal{G}$. Let QMA be the set of languages L such that there exists a uniform family of polynomial-size quantum circuits $Q_{n,m}$ (with gates drawn from $\mathcal{G}$) that takes two inputs $(x, |\psi\rangle)$ with the following properties: For any instance x,*

1. *If $x \in L$, then exists an n-qubit state $|\psi\rangle_x$ with $n = O(\text{poly}(|x|))$ and $\text{Prob}(M_{Q_{|x|,n}}(|x\rangle \otimes |\psi\rangle_x) = 1) \geq 2/3$.*
2. *If $x \notin L$, then for any n-qubit state $|\psi\rangle_x$ with $n = O(\text{poly}(|x|))$, $\text{Prob}(M_{Q_{|x|,n}}(|x\rangle \otimes |\psi\rangle_x) = 0) \geq 2/3$.*

In the first case, w_x is the witness *for $x \in L$.*

We will discuss these two classes again later. QMA is usually considered the important one, the most natural quantum analog of NP. One thing that is worth noting right now is that as far as we know, all the QMA-complete problems must be promise problems because of the requirement that the probability is at least 2/3 of either accepting or rejecting the witness. Certainly, the most natural complete problems we know don't have this probability separation without a restriction on the possible instances. Without the promise, they are generally *QMA-hard* instead.

4.2 PSPACE

Going to still larger complexity classes, we get PSPACE. PSPACE is the class of decision problems that can solved in polynomial *space* on a Turing machine. PSPACE is a large complexity class because we can run very long algorithms which use only a modest amount of space and then erase it and use it again.

Definition 3. *Let PSPACE be the set of languages that can be decided by a Turing machine using $O(\text{poly}(|x|))$ space for instance x.*

We could also define it as languages decided by circuit families which use a polynomial number of bits; however, we would have to use a broader definition of "uniform," to allow circuits generated by a Turing machine in polynomial time. This is because a Turing machine using only logarithmic space can't generate exponential-size circuits. Similarly, any PSPACE algorithm will halt in at most exponential time.

Theorem 1. *Let A be an algorithm that uses $f(|x|)$ space on input $|x|$. Then A on input $|x|$ either halts or repeats itself after a time at most $2^{O(f(|x|))}$.*

Proof. Consider the state of the Turing machine tape (the *whole* tape that is used, not just one cell) at time i. If there are A different symbols in Σ, then there are total of $A^{f(|x|)}$ different possible states of the tape. There are also $|K|$ head states and $O(f(|x|) + |x|)$ head locations (conceivably there could be some blank spots on the tape between the input and the part that is used, but there have to be a limited amount of them or the algorithm would never start writing again), for a total of $O((f(|x|) + |x|)|K|A^{f(|x|)} = 2^{O(f(|x|))}$ different possible states of tape plus head.

The action of the Turing machine is completely determined by the state of the tape and the head. In particular, if it repeats the same tape state and the same head state, it will repeat the next time step and so on, looping. Since there are only $2^{O(f(|x|))}$ possible different states given the amount of space used by the algorithm, it must either halt or repeat itself after this time. □

PSPACE contains all of the complexity classes we have talked about so far (except for P/poly which is a bit of a special case). Here is a quick proof sketch that BQP $\subseteq$ PSPACE. The proof that QMA $\subseteq$ PSPACE is a bit more involved and we will get back to it later.

Proof that BQP $\subseteq$ PSPACE. A quantum circuit can be simulated on a classical circuit by multiplying out the $2^n \times 2^n$ matrices given by the quantum gates. This takes exponential time and naively takes exponential space. However, we can reduce this to exponential time and polynomial space.

Claim 1. *We can calculate any single matrix element using only polynomial space:*

Proof of claim.

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

Each term in this sum is a product of polynomially many terms so can be computed using polynomial time and space. We can step through the value of the n-tuple $(c_0, \ldots, c_{n-1})$ and keep a running total for the sum while doing so, discarding any previous partial sum. At any given time, then, we only need to keep track of the n-tuple (which requires linear space) and the current value of the sum (which requires polynomial space to keep track of with adequate precision). Thus, the whole sum can be computed in polynomial space. □

A single matrix element doesn't tell us the probability of getting outcome 1 when the first qubit is measured, but we can compute

$$\text{Prob(outcome 1)} = \sum_{a=1a_1a_2\ldots a_m} |\langle a| \prod_{i=0}^{n} U_i |b\rangle|^2 \tag{2}$$

(where $|b\rangle$ is the initial state of the computation), the total probability over all outcomes with 1 in the first qubit. Each term in the sum is computable with polynomial space by the claim, and again we can keep a running total of the sum with polynomial space. □

You might expect that at this point, we should be introducing another complexity class of problems that can be solved by quanutm algorithms that use only polynomial space, but it turns out that quantum algorithms with polynomial space give the same complexity class as classical algorithms with polynomial space. We will (probably) return to this when we talk about PSPACE at more length.

4.3 Quantum Mechanics and Quantum Gates

This is intended to be a quick review of the quantum formalism and the quantum gates we will be using. (Pure) quantum states are vectors in a complex Hilbert space. We write them with angle brackets called *kets*. General pure are usually labelled with Greek letters, particularly ψ and ϕ, and can be expanded in a standard basis labelled by bit strings: e.g.,

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle \tag{3}$$

$$|\phi\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle \tag{4}$$

The inner product of two quantum states is written using *bras* and kets: e.g.,

$$\langle\phi|\psi\rangle = a^*\alpha + b^*\beta + c^*\gamma + d^*\delta. \tag{5}$$

Here, the star indicates complex conjugation. In principle, quantum states need to be normalized $\langle\psi|\psi\rangle = 1$, but we often don't bother to do this explicitly. One place where it is important to do so is when measuring; a measurement in the standard basis has a probabilistic result, giving outcome a with probability equal to the absolute value squared of the component of a, e.g., the probability of getting outcome 00 if we measure $|\psi\rangle$ above in the standard basis for both qubits is $|\alpha|^2$. The normalization then ensures that the total probability sums to 1.

Quantum gates are unitary operations $UU^\dagger = I$ (which ensures they preserve the inner product and therefore total probability). Here † represents the Hermitian adjoint, which is the complex conjugate transpose of the matrix representation. Standard example quantum gates are

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad \text{Hadamard} \tag{6}$$

$$R_\theta = \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix} \quad \text{Phase gate} \tag{7}$$

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{Controlled-NOT} \tag{8}$$

Another example gate is the Toffoli gate:

$$Tof|a, b, c\rangle = |a, b, c \oplus ab\rangle. \tag{9}$$

Technically, quantum states are vectors in a *projective Hilbert space*, because they are only defined up to a global phase (i.e., one that applies equally to all states). Note that changing by a global phase does not affect the probability of a measurement outcome.

We will be using some additional more advanced properties of quantum states, but this should get you started. And if this material is unfamiliar, you should review it.

4.4 Universal Quantum Circuits

Recall:

Definition 4. *Let $\mathcal{G}$ be a set of quantum gates acting on a bounded number of qubits (usually either 2 or 3).*

- *$\mathcal{G}$ is a* universal *set of gates if for any unitary U, there exists a circuit (of any size) containing gates from $\mathcal{G}$ that realizes the transformation U.*
- *$\mathcal{G}$ is* approximately universal *if, for any unitary U and any error ϵ, there exists a circuit composed of gates from $\mathcal{G}$ that realizes a unitary U_ϵ such that $\|U - U_\epsilon\|_{sup} < \epsilon$.*

The standard example of a universal set of gates is $\mathcal{G} = \{\text{single-qubit gates}, CNOT\}$. Two standard examples of approximately universal sets of gates are $\mathcal{G} = \{H, R_{\pi/8}, CNOT\}$ and $\{H, R_{\pi/4}, Tof\}$. Here $CNOT$ is the controlled-NOT, H is the Hadamard transform, R_θ is the diagonal phase gate $R_\theta|0\rangle = e^{-i\theta}|0\rangle$, $R_\theta|1\rangle = e^{i\theta}|1\rangle$, and Tof is the 3-qubit Toffoli gate (also called the controlled-controlled-NOT). (Often, $R_{\pi/4}$ is called S and $R_{\pi/8}$ is called T.)

One advantage of dealing with approximately universal gate sets is that we can now use a finite gate set. Note that a finite gate set can only ever be approximately universal and not exactly universal, since the number of circuits with a finite gate set is countable whereas the set of all unitaries is uncountable.

We said earlier that the definition of BQP doesn't depend on which universal gate set we pick. This is a consequence of the Solovay-Kitaev theorem, which says that any approximately universal gate set can not just approximate any unitary to any desired accuracy $\epsilon > 0$, but can do so with a number of gates that is only polylogarithmic in $1/\epsilon$. Moreover, it is constructive.

Theorem 2 (Solovay-Kitaev)**.** *Let $\mathcal{G}$ be a universal set of gates. Then for any unitary V in a fixed Hilbert-space dimension D, there exists a classical algorithm to find, for any $\epsilon > 0$, a quantum circuit of size $O(\text{poly}(\log(1/\epsilon)))$ which realizes U_ϵ such that $\|V - U_\epsilon\| < \epsilon$. The classical algorithm runs in time $O(\text{poly}(\log(1/\epsilon)))$ as well.*

The original version of the Solovay-Kitaev theorem requires that $\mathcal{G}$ is closed under inverses, but there is a recent improvement that removes that requirement. It is also worth noting that certain gate sets (such as $\{H, R_{\pi/8}\}$) can achieve this approximation more efficiently (i.e., with a lower exponent of the logarithm) than the general case.