6.896 Quantum Complexity Theory November 11, 2008

Lecture 20

Lecturer: Scott Aaronson

1 Last Time

In the previous lecture, we saw the result of Aaronson and Watrous that showed that in the presence of closed timelike curves (a model due to David Deutsch of 'non-paradox-inducing' time travel), classical and quantum computation are polynomially equivalent, as are polynomial time and space: $P_{CTC} = BQP_{CTC} = PSPACE_{CTC} = BQPSPACE_{CTC}$.

In the process of proving these results we also discussed superoperators, the most general form of operation on quantum systems, and described how fixed-points of these mappings are guaranteed to exist and can be found in polynomial space.

2 The Information Content of Quantum Systems

Complexity Theory does itself a disservice by inventing boring names for its cool ideas. Today's lecture is about $P/poly$ and $BQP/poly$, but it's 'really' about understanding the nature of quantum mechanics (QM).

There are three major 'interpretations' of the theory of QM: Many-Worlds, Bohmian, and Copenhagen/Bayesian. Empirically they make the same predictions, but they make different descriptions of the underlying state of the universe. In particular, they can be seen as having different estimations of the 'information content' of quantum systems.

A quantum state on n qubits is describable by a 2^n-dimensional complex unit vector. Is there 'really' an exponential amount of information in such a system? The Copenhagen interpretation suggests that this is so only to the extent that we consider a probability distribution on n-bit strings to contain exponential information: it takes that much to describe it fully, but we only learn n bits in our observation or measurement. This contrasts with the Many-Worlds perspective, in which all complex amplitudes of the quantum state vector are really 'present' as simultaneously existing alternative states of affairs.

Complexity theory can in effect stage 'battles' between these interpretations, raising questions about the extent to which we can extract and use information in quantum states. One such issue to explore is the extent to which quantum 'advice' helps us solve computational problems. To understand how this works, we need first to understand the classical theory of computing with advice.

2.1 Classical Advice

What does it mean to 'compute with advice'? Suppose there is a trustworthy and all-knowing Advisor, who wants to help a Student solve problems. If the Advisor could see which computational problem the student was working on, s/he could simply state the answer. However, things get more interesting when we suppose that the Advisor is a busy and somewhat hard-to-reach person, who tends to give out the same advice to all students. Let's say the Advisor gives out advice to a student trying to solve a decision problem '$x \in L$?' that only depends on L and the length $|x|$... what kind of complexity class does this notion give rise to?

Definition 1 *($P/k(n)$) Define $P/k(n)$, 'P with $k(n)$ bits of advice', as the class of languages $L \subseteq \{0,1\}^*$ such that there exists a polynomial time Turing machine $M(x,y)$, and a collection $\{a_n\}_{n \in \mathbf{N}}$ of binary 'advice' strings, with $|a_n| = k(n)$, such that for all x, $M(x, a_{|x|}) = 1$ iff $x \in L$.*

Now if $k(n) = 2^n$ and we're allowed random-access to the advice strings, we can encode any decision problem directly into the advice: $P/2^n = ALL$. Thus it's natural to restrict ourselves to polynomial advice:

Definition 2 *($P/poly$) Define $P/poly$, 'P with polynomial advice', as the class of languages $L \subseteq \{0,1\}^*$ such that there exists a polynomial time Turing machine $M(x,y)$, and a collection $\{a_n\}_{n \in \mathbf{N}}$ of binary 'advice' strings, with $|a_n| = O(poly(n))$, such that for all x, $M(x, a_{|x|}) = 1$ iff $x \in L$.*

This class can readily be seen to coincide with the class of languages which have polynomial-sized *circuits*, whose designs may vary in arbitrary ways according to the input length. This arbitrary variation leads us to refer to this class and ones like it as 'nonuniform'. We could also think of these models as exploring the power of 'precomputing', where an exponential amount of computation goes into deriving which algorithm to use for a given length n. (Strictly speaking, though, $P/poly$ contains languages not computable using any uniform precomputing procedure.)

It is easy to see that $P/poly$ is a bigger class than P; indeed, even $P/1$ is uncountable, hence is not even contained in the class of *recursive* languages! Nevertheless, we do know of some limits on the power of $P/poly$. For one thing, a simple counting argument shows that $P/poly$ does not contain the class of all languages, ALL. In fact we know $P/poly$ does not contain $ESPACE$, the class of problems computable in space $2^{O(n)}$ (and we can say things a bit stronger than this).

We also suspect other limits on $P/poly$. For instance, in the early '80s Karp and Lipton proved that if $NP \subset P/poly$ then the Polynomial Hierarchy collapses to its second level, i.e. $PH = NP^{NP}$. This is considered unlikely.

2.2 Quantum Advice

Now it's time to add Q's to everything... by analogy with $P/poly$, we make the following definition:

Definition 3 *($BQP/poly$). Define $BQP/poly$, 'BQP with polynomial advice', as the class of languages $L \subseteq \{0,1\}^*$ such that there exists a uniform family of polynomial sized quantum circuits $C_n(|x\rangle, |\psi\rangle)$, and a collection $\{a_n\}_{n \in \mathbf{N}}$ of binary 'advice' strings, with $|a_n| = O(poly(n))$, such that for every n and all x of length n, $C_n(|x\rangle, |a_n\rangle) = 1$ iff $x \in L$.*

Similarly to before, we have $BQP/poly \neq BQP$. But the fun really starts when we allow our quantum algorithms to receive quantum advice:

Definition 4 *($BQP/qpoly$) Define $BQP/qpoly$, 'BQP with polynomial quantum advice', as the class of languages $L \subseteq \{0,1\}^*$ such that there exists a uniform family of polynomial sized quantum circuits $C_n(|x\rangle, |\psi\rangle)$, and a collection $\{|\psi_n\rangle\}_{n \in \mathbf{N}}$ of quantum 'advice' states on $O(poly(n))$ qubits, such that for every n and all binary strings x of length n, $C_n(|x\rangle, |\psi_n\rangle) = 1$ with probability $\geq 2/3$ if $x \in L$ while $C_n(|x\rangle, |\psi_n\rangle) = 0$ with probability $\geq 2/3$ if $x \notin L$.*

So how big is $BQP/qpoly$? 'High-information content' interpretations of QM should at least suggest that it ought to be quite big, maybe even ALL. Notice that there are way more quantum states on n qubits than strings of length n, so the counting arguments that show obvious limits on $P/poly$ (and $BQP/poly$) no longer work here.

We could indeed try to encode an arbitrary boolean function f_n at each length n, say by preparing advice string $\frac{1}{2^{n/2}} \sum_x |x\rangle|f_n(x)\rangle$. The problem is how to extract this information. Measuring in the standard basis just tells us $(x, f(x))$ for some *random* x, not the one we're actually interested in! (If we 'postselect' on getting the x we were interested in, however, we *can* compute any function: that is, $PostBQP/qpoly = ALL$.)

The *Group Membership* problem is in $BQP/qpoly$: if we consider a group G_n and subgroup $H_n \leq G_n$ as being fixed for each n, and we want to test if an input $x \in G_n$ is in H_n, Watrous' algorithm allows us to do this given advice state $\frac{1}{\sqrt{|H_n|}} \sum_{y \in H_n} |y\rangle$. (Note that this works for any efficiently computable group operation. If we were using permutation representations of group elements, the problem would actually be in P.)

What other evidence do we have that *quantum* advice might be particularly useful for BQP machines? Aaronson has given a *quantum oracle* relative to which $BQP/qpoly$ properly contains $BQP/poly$. It's a start...

Now let's try to get a reasonable *upper*-bound on the power of quantum advice. Surely we can do better than ALL...

Theorem 5 *$BQP/qpoly \subseteq PostBQP/poly$.*

(And we know $PostBQP/poly = PP/poly \neq ALL$ by a counting argument.)

Proof (Sketch): Say $L \in BQP/qpoly$, computed by a family C_n of quantum circuits on advice $\{|\psi_n\rangle\}$.

Our first step is to amplify the success probability of C_n from $2/3$ to $1 - \frac{1}{2^n}$. This can be done by running the algorithm kn times, $k = O(1)$. We need a fresh copy of the advice for each run, so redefine the advice for length n as $|\varphi\rangle = |\psi_n\rangle^{\otimes kn}$.

In a full proof, we would now develop the 'Almost as Good as New' Lemma, which states: If the outcome of a computation's final measurement is some value b with probability $(1 - \epsilon)$ when using advice $|\varphi\rangle$, then using $|\varphi\rangle$ in the computation leaves the advice in a new state $|\varphi'\rangle$ that is $\sqrt{\epsilon}$-close to $|\varphi\rangle$ in trace distance.

Now the proof takes a seemingly crazy turn. Let I, the 'maximally mixed state', be uniform over all $p(n)$ bitstrings, where $p(n)$ is the number of qubits in $|\varphi\rangle$. We ask, does I work as advice in place of the $|\varphi\rangle$ we were previously using? Does it work on all inputs x of length n to help us compute $L(x)$ with success probability $2/3$? Probably not. Then, there exists an x_1 such that $C_n(|x_1\rangle, I)$ succeeds with probability less than $2/3$.

Let's consider the state ρ_1 that is the residual state of the advice register after we ran C_n on (x_1, I), *postselecting* on the event that we succeeded in outputting $L(x_1)$. We ask again: is ρ_1 good advice for C_n to use on every input? If not, some x_2 exists such that $C_n(|x_2\rangle, \rho_1)$ succeeds with probability less than $2/3$. Let ρ_2 be the residual state of the advice register after we ran C_n on (x_2, ρ_1), postselecting on the event that we succeeded in outputting $L(x_2)$. And continue in this fashion for some $t(n)$ stages, t a polynomial. (A technical note: it can be shown that, since we started from the maximally mixed state I for which 'anything is possible', the events we postselect upon at each stage have nonzero probability, so this process can in fact be continued.)

If at any stage we cannot find an x_i to move forward, we must be holding a ρ_{i-1} that works as advice for every input, and we can use it to run the quantum circuit C_n on the input x we're actually interested in, succeeding with high probability. So, what we need to show is that the process *must* halt in polynomially many steps.

The maximally mixed state has a key property we exploit: it is a uniform superposition over basis states, not just over the basis of binary strings $\{|x\rangle : x \in \{0,1\}^{p(n)}\}$, but over *any* orthonormal basis (it is 'rotationally invariant'). In particular, it's uniform with respect to a basis that contains $|\varphi\rangle$, our 'good' advice state. Thus since advice $|\varphi\rangle$ yields the right answer on each x with probability at least $(1 - \frac{1}{2^n})$, I yields the right answer with probability at least $\frac{(1-\frac{1}{2^n})}{2^{p(n)}}$. Similarly, since $|\varphi\rangle$ can be reused on each of $x_1, \ldots x_{t(n)}$ to give the right advice with probability $1 - o(1)$ (by the 'Almost as Good as New' Lemma), the probability that I succeeds on each of these inputs in turn is at least $\frac{(1-o(1))}{2^{p(n)}}$.

But we've designed this sequence of inputs so that the probability that I can be reused on each of them, succeeding at each stage, is less than $(\frac{2}{3})^{t(n)}$. To avoid a contradiction, the process can only continue for $t(n) \leq O(p(n))$ steps. Thus *there exist* $x_1, \ldots x_{O(p(n))}$ such that, if ρ is the residual state after we start with I and postselect on succeeding on each x_i in turn, ρ is a good advice string for *every* $x \in \{0,1\}^n$.

So, we just give this sequence of (classical!) strings to our $PostBQP$ algorithm, along with the correct outcomes $b_1, \ldots b_{O(p(n))}$ for each of them. The algorithm prepares I (easily done), runs it on the advice input-sequence, and postselects on getting outcoms $b_1, \ldots b_{O(p(n))}$. The leftover state ρ is necessarily has success probability $2/3$ when used as advice in C_n to determine if $x \in L$, for any desired $x \in \{0,1\}^n$. This completes the proof sketch.

■