Lecture 24

Lecturer: Scott Aaronson

1 Quantum circuits that can be efficiently simulated

Last time, we saw the Gottesman-Knill Theorem, which says that any circuit composed of CNOT, Hadamard, and phase gates can be similated in classical polynomial time. We also began our discussion of Valiant's Matchgates. In this lecture, we finish our discussion of Valiant's Matchgates, and also describe Vidal's efficient classical simulation of quantum computation with limited entanglement.

1.1 Valiant's Matchgates

1.1.1 Bosons vs. Fermions

Recall that last time, we saw that there were two fundamentally different kinds of particles: "bosons," which were force particles like photons; and "fermions," which were matter particles like quarks. In a system of identical non-interacting particles, we calculate the amplitudes for states of the system in future configurations in fundamentally different ways for these two kinds of particles. Observe, in Figure 1, the two particles could enter the same configuration by either taking paths a and b or by taking the paths labeled c and d:

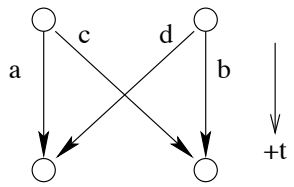

Figure 1: Two identical particles can enter identical configurations by either taking paths a and b, or by "switching places" and taking paths c and d.

The difference between bosons and fermions is that for bosons, the amplitude for this final configuration is given by $ab + cd$, i.e., the amplitudes for these two pairs of paths add, whereas for fermions, the amplitude is given by $ab - cd$, i.e., the paths interfere with one another. (Generally, one looks at the sign of the permutation of the particles among the various positions in the configuration corresponding to each term.)

One might wonder how we know whether the amplitude should be $ab - cd$ or $cd - ab$. The simple answer is that we *don't* know, and moreover it *doesn't matter*—a global phase shift is undetectable by an observer and either one of these will assign the same probabilities to observations. We only know that the universe does it in some *consistent* way.

It's worth remarking that, if the two final positions are the same position, then the amplitudes ab and cd are the same, where we find that bosons still add, giving an amplitude of $2ab$ (they end

up on top of each other—for example, a laser consists of many photons "on top of each other"), whereas the two terms for fermions *cancel each other out*, i.e., they have amplitude $ab - cd = 0$. This is interpreted as saying that fermions, in contrast to bosons, "take up space" and is known to physicists as the "Pauli exclusion principle" or "Fermi pressure." The amplitudes for a fermion are spread out, somewhat (like a Gaussian distribution), so the cancellation of the amplitudes has the effect of keeping fermions from getting too close to each other. It turns out that this is what prevents neturon stars from collapsing.

The n-particle generalization of this is as follows: suppose we have n identical, non-interacting particles, and for $i, j \in \{1, \ldots, n\}$ a_{ij} denotes the amplitude for a single particle going from position i in some configuration to position j in some other configuration of the n particles (i.e., a_{ij} is calculated imagining that no other particles are present). Now, for the matrix A such that a_{ij} is the (i, j) entry, the amplitude for the prescribed final configuration is given by $\mathrm{per}(A)$ if the particles are bosons, whereas it is given by $\det(A)$ if the particles are fermions. To repeat, it was crucial that we assumed that the particles were non-interacting and identical (no interference occurs for distinct particles—two states that differ in any way do not interfere in quantum mechanics). We know that, although $\det(A)$ and $\mathrm{per}(A)$ look superficially similar, they are extremely different computationally: the former can be computed in classical polynomial time, whereas the latter is #P-complete.

Another remark is in order: although calculating the amplitudes for a configuration of bosons reduces to computing the Permanent – a #P-complete problem – this doesn't necessarily imply that we can use a system of bosons to solve #P-complete problems, only that the configuration can be computed with a #P-oracle—something we already knew since we saw $\mathrm{BQP} \subseteq \mathrm{P}^{\#\mathrm{P}}$, and we believe that BQP faithfully models what can be computed using quantum mechanics. We don't expect for this containment to be an equality. We expect that the instances of permanents arising from bosons are of a special form (that can be simulated in BQP), and thus we don't expect that such systems of non-interacting identical particles can be set up for arbitrary, hard instances.

1.1.2 Matchgates

The link between all of this talk of particle physics and quantum circuits is roughly that a system of identical non-interacting fermions can be simulated in classical polynomial time: suppose we have a quantum circuit in which every gate is a "matchgate," i.e., has a two-qubit unitary of the form

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & a & c & 0 \\ 0 & d & b & 0 \\ 0 & 0 & 0 & ab - cd \end{bmatrix}$$

for some a, b, c, and d, and suppose our initial state is, for example $|01100010\rangle$, or more generally, any n-bit string. Suppose we want to compute the amplitude for some outcome after applying our circuit to this initial state, e.g., $|10010100\rangle$.

Our first observation is that this is trivial if the strings have different Hamming weights—the amplitude will be zero in this case, since our gates preserve the Hamming weight. The second observation is that if the strings have the same Hamming weight, we can reinterpret the $|1\rangle$ bits as fermionic particles and the $|0\rangle$ bits as "vacuum," and then calculate the final amplitude as follows: for each individual "fermion" in the input string, and each individual "fermion" in the output string,

we calculate the amplitude for the input fermion ending up in that position in the output string, i.e., the amplitude for a transition between two strings of Hamming weight 1, such as $|01000000\rangle$ and $|00000010\rangle$ in our example. This computation can be done in classical polynomial time since it only involves keeping track of n amplitudes at each time step, and if there are k particles in each configuration, we repeat this k^2 times in total. We then form the $k \times k$ matrix of these amplitudes, and calculate its determinant to obtain the amplitude for this final configuration, which we know can also be done in classical polynomial time. This is correct since we see that in mapping $|11\rangle$ to itself, the gate introduces a factor of $ab - cd$ to the amplitude, exactly as it should for a fermion, and the preservation of the Hamming weight corresponds to particle being neither created nor destroyed. (This matrix of transition amplitudes is apparently also known as the "Jordan-Wigner transformation.")

More generally (though we won't get into too much detail here) we can use "Pfaffians" to calculate the probability of measuring some qubit to be a $|1\rangle$ after the circuit is applied. Using an algorithm that is closely related to the algorithm for counting the number of perfect matchings in a planar graph, it turns out that we can also calculate this probability in classical polynomial time. In fact, still more general kinds of circuits can be simulated in classical polynomial time in this way. For example, we can include terms that correspond to fermions being created and annihilating each other in a vacuum, and use any gate of the form

$$\begin{bmatrix} x & 0 & 0 & y \\ 0 & a & c & 0 \\ 0 & d & b & 0 \\ z & 0 & 0 & w \end{bmatrix}$$

such that $ab - cd = xw - yz$, provided that gates act on adjacent qubits on a line only. Strictly speaking, we don't even need to require that these matrices are unitary for the simulation to be efficient, only for the correspondence with particle physics to hold.

1.2 Quantum computation with limited entanglement

There is one other class of quantum circuits that can be efficiently simulated on a classical computer: circuits with limited entanglement. For example, pure states which are always unentangled can easily be simulated classically, since we only need to write $2n$ amplitudes (rather than 2^n) on each time step. More generally, we consider the following way of measuring the entanlgement of a quantum state:

Definition 1 (Schmidt rank) *The* Schmidt rank χ *of a bipartite state (on two subsystems)* $|\psi\rangle$ *is given by the minimum number such that we can write*

$$|\psi\rangle = \sum_{i=1}^{\chi} \lambda_i |a_i\rangle \otimes |b_i\rangle$$

It turns out that the Schmidt rank be computed efficiently by diagonalizing the density matrix on one of the two sides, and finding how many different eigenvalues it has. In general, the Schmidt rank may be as high as $2^{n/2}$; it turns out that in some cases, when it is small (polynomially bounded) we can efficiently simulate the circuit.

Given an n-qubit state, let $\chi_{\max}$ denote the maximum of χ over all bipartitions of the n qubits. Then we have

Theorem 1 (Vidal) *Suppose a quantum computation involves nearest-neighbor interactions only among qubits on a line, and that $\chi_{\max}$ is polynomially bounded at every step. Then the computation can be efficiently simulated on a classical computer.*

The simulation is essentially carried out via dynamic programming – for each qubit, we store a $\chi \times \chi$ matrix encoding how each qubit interacts with the other qubits (in terms that may be familiar from general relativity, we use contraction of tensors to obtain a more compact representation). We then show that this compact representation can be locally updated for nearest-neighbor operations, and that we can obtain the probabilities from them. Thus, we only need to store $O(n\chi^2)$ amplitudes on each time step to simulate these circuits.

This algorithm is not meant for simulating physical systms—this is for simulating quantum circuits, i.e., applying nearest-neighbor unitaries. Vidal also later developed a related algorithm for simulating physical systems with nearest-neighbor Hamiltonians or for estimating the ground states where there is limited entanglement, and then was able to use this algorithm to solve some problems in condensed-matter physics where the states of interest had dimensions too high to deal with using existing techniques. It turns out that cases with low entanglement encompass most cases that physicists care about, since creating entanglement is hard—this is, after all, what makes building quantum computers so hard in the first place.

2 Grab bag

2.1 Counterfactual Computing [Josza-Mitchison]

This is best described as a cute observation about quantum algorithms. It turns out that it's easiest to explain using the example of Grover's algorithm.

$$\overbrace{[\,|\cdots|\,|i|\,|\cdots|\,]}^{N}$$
$$\text{marked}$$

Suppose we're searching a list of N items using Grover's algorithm. If the ith item is marked, then we find it with high probability in $O(\sqrt{N})$ queries. Notice, on the other hand, that if *no* item is marked, then we also learn *that* with high probability in $O(\sqrt{N})$ queries by the algorithm's failure.

At this point, we ask the slippery question, "what is the probability that the ith location was queried?" Strictly speaking, this question does *not* make sense, but we could imagine that we measured the query register at each time step, and use the probability of outcome $|i\rangle$ as this probability. Then, since in the absence of any marked item, we query using the uniform superposition at every step, the probability of querying location i at each step is $1/N$. Since there are $O(\sqrt{N})$ steps, the "total probability" with which we query location i is $\sim 1/\sqrt{N}$. Thus, one could say that we learn that item i is not marked, but almost certainly without querying it.

2.1.1 Vaidman's Bomb

We could cast the whole problem more dramatically as follows: suppose the ith position being unmarked corresponds to there being a bomb in the ith location, where the bomb explodes if that position is queried. Imagining ourselves in the role of the Quantum Bomb Squad, we would like to know whether or not there is a bomb in the ith location without querying it and exploding the

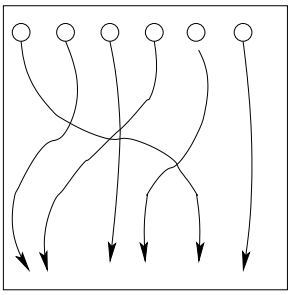

Figure 2: Nonabelian anyons being "passed" across each other in a simulated 2-dimensional surface

bomb. Provided that one can enter a coherent superposition of querying the bomb's location and not querying it, the problem is solved: for some large N, we query each location with amplitude $1/\sqrt{N}$. For most of the locations we do nothing. If we query the ith location and there is a bomb there, then that branch of the wave function is killed. If there isn't, we can switch the phase of the amplitude of the ith position (from $1/\sqrt{N}$ to $-1/\sqrt{N}$) and use Grover amplification to increase the amplitude of querying that location in the future, and make another query. After repeating this process $O(\sqrt{N})$ times, if there is no bomb, we learn that with constant probability (since in this case we actually simulate Grover's algorithm finding the marked ith item), but if there is a bomb, again, the "probability that we query it" is only $\sim 1/\sqrt{N}$ since we die before we would perform the Grover amplification step (and hence, this corresponds to running Grover's algorithm with no marked item). By taking N sufficiently large, we can make the "probability" of dying arbitrarily low. (Needless to say, this is unlikely to actually work in practice due to the difficulty of entering a coherent superposition, which is made particularly difficult by the possibility of an explosion, which would tend to be highly decoherent.)

Thus, counterfactual computing is the study of in what settings you can learn about something without having to interact with it. (In the case of the bomb, though, the reason that we learn anything is that *if it's safe*, then we look in the ith position.) Simple classical analogues of this also exist: suppose we have two processes, P_1 and P_2, where P_1 is computing some boolean function f and P_2 is idle. Suppose that if $f(x) = 1$, then P_1 kills P_2. Then, if we come back and check that P_2 is still running after P_1 would have completed, then we could learn that P_1 computed $f(x) \neq 1$ from P_2, even though P_2 never computed f and (in this case) P_1 never interacted with P_2. In any case, Roger Penrose (apparently) applied counterfactual computation to propose a scheme where, if one was Orthodox Jewish, one could switch on a light on the Sabbath without having to flip the switch.

2.2 Topological Quantum Computation

This is an architectural proposal for implementing a quantum computer that is distinguished by involving "a huge amount of interesting math." (This is what attracted the attention of Alexei Kitaev and Mike Freedman.) A *nonabelian anyon* is a kind of particle (cf. bosons and fermions) that provably can't exist in three dimensions, but can exist in two dimensions (although it isn't clear that they have ever been observed). By analogy, we saw that when bosons switch places, nothing happens, whereas when fermions switch places, "the universe picks up a minus sign;" we could also imagine particles that, when they switch places, some arbitrary group action or some

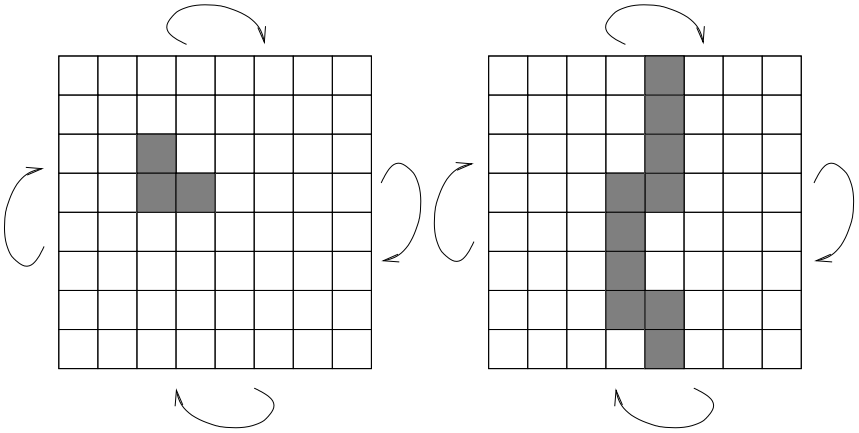

Figure 3: Shaded regions represent sets of errors that are topologically trivial (left) and nontrivial (right) on a torus. Computation might fail in the second case, but not in the first.

arbitrary unitary is applied to the universe—some more general kind of symmetry. No fundamental particles that we have observed in nature are anything like these nonabelian anyons, but supposing we confine things to some two-dimensional lattice, we could hope to build composite particles ("quasiparticles") that simulate the behavior of these nonabelian anyons. If we could do this, then just by creating n nonabelian anyonic quasiparticles and swapping them past each other in various ways, we could perform a universal quantum computation (see Figure 2). There are lots of interesting connections to knot theory here.

Of course, all of our various proposals which can perform universal quantum computations are theoretically equivalent to one another. The upshot of this proposal is that fault-tolerance comes for free. This is in contrast to our quantum circuit architecture, where we relied on the Threshold Theorem—a complicated hierarchical error-correction scheme operating on all of the qubits in parallel (constantly measuring them and correcting errors) that guaranteed that a constant threshold on the error rate was sufficient to allow arbitrarily long quantum computations. Actually implementing the scheme described in the Threshold Theorem has been extremely difficult. In the topological architecture, fault-tolerance arises from the intrinsic physics of the system: we don't even measure the actual paths—we only measure topological invariants, and thus we care about the global topology. For example, in a related scheme for topological error-correction, The only way for our computation to be incorrect would be for qubits along a topologically nontrivial path to be corrupted (see Figure 3). As an aside, it would be much easier to make these schemes work if space had four dimensions, rather than three.

2.3 Proposals for more powerful "realistic" models of computation

We've been talking for the whole course about what quantum mechanics is, and how it changes the structure of computational complexity theory. A natural thing to wonder is, if we were surprised once by the laws of physics, what further surprises could there be? Could there be another model of computation beyond quantum computing?

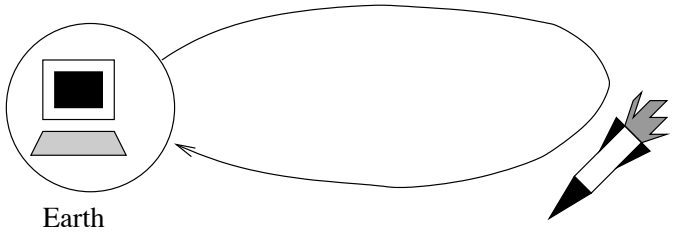

Figure 4: A "relativistic computer" – an observer sets a computation running and travels at relativistic speeds away from Earth and back, to read off the result of the computation.

2.3.1 Quantum Field Theory Computation

We've already discussed some such speculative ideas in this course, such as quantum computers with closed timelike curves and quantum computers with post-selection. In terms of physics, people have wondered (for example) if quantum field theories could yield another, more powerful model of computation. Quantum field theories combine quantum mechanics with special relativity (and hopefully, one day, will include general relativity as well). We could ask, would building a computer with these principles in mind yield a larger complexity class than BQP?

This question has been very hard to address since these quantum field theories are not even mathematically well-defined. No one can prove that they even exist! (Proving that they make any kind of mathematical sense is related to one of the Clay problems.) These theories are, to some extent, "hacks" that work up to a certain energy scale, but beyond it they give nonsensical answers. We can't study computational complexity in this kind of a world.

There is one special class, called *topological quantum field theories*, which we can work with concretely (and for which results exist). In these theories, we have two space dimensions, one time dimension, and no matter or energy—just points and cuts that can move around on a $(2+1)$-dimensional manifold (so the only degrees of freedom are topological). This is the simplest interesting class of quantum field theories—they are studied because they *can* be given rigorous mathematical definitions. These were rigorously studied by Witten in the '80s.

These topological quantum field theories are directly related to a knot invariant called the *Jones polynomial*, a function of knots which was invented for independent reasons. In 2000, Freedman, Kitaev, Larsen, and Wang defined a corresponding computational problem of simulating topological quantum field theories, and showed that it is BQP-complete, so these topological quantum field theories give computational power equivalent to that granted by quantum mechanics. Since simulating these topological quantum field theories was equivalent to estimating the Jones polynomial for a knot, their simulation also implied an efficient quantum algorithm for estimating the Jones polynomial. Their paper was *extremely* difficult to understand, though. Recently, Aharonov, Jones, and Landau also gave a direct BQP algorithm for estimating the Jones polynomial (which is also BQP-complete, essentially by Witten's work).

2.3.2 Relativistic Computation

We don't think we can use a quantum computer to solve NP-complete problems; what about a classical relativistic computer? One proposal is immediate: we set a classical computer to work on some very hard problem, board a space ship, and travel away from the Earth and back at close to

the speed of light (see Figure 4). Then, as in the twins paradox, billions of years have elapsed in the Earth's reference frame, (your friends are long dead, the sun is a red giant, etc.) but you can read off the answer from your computer. Alternatively, you could set your computer working, and move *extremely* close to the event horizon of a black hole to slow down the rate of time in your reference frame (from the perspective of an external observer) to the same effect.

Assuming you're willing to pay the price for this, there are additional problems with these proposals: in contrast to classical computer science, where energy is not an issue (we know, e.g., by Landauer's work that the dissipation of energy is not required by classical computation), approaching relativistic speeds consumes a lot of energy. In particular, to obtain an *exponential* speed-up, since the our time dialation at velocity v is given by the $\frac{1}{\sqrt{1-v^2}}$ factor, an exponential speed-up requires that v is exponentially close to the speed of light—but then, the amount of energy we consume similarly involves a $\frac{1}{\sqrt{1-v^2}}$ factor, so it requires an exponential amount of energy, and thus an exponential amount of fuel is necessary.

Thus, to study relativistic computation, we'd need to consider the energy requirements. But, too much energy in a bounded volume creates a black hole, so the amount of energy and physical space required are *also* related. Thus, in this case, and exponential time speed-up requires an exponential amount of physical space—and thus, since the speed at which the fuel in the far part of a fuel tank is limited by the speed of light, we'd still incur an exponential time overhead.

Now, on the other hand, people have studied circuits with extremely limited computational models, for example threshold circuits in which only a limited number of the gates can be on, which corresponds in a reasonable way to a energy limitation (this was studied in a recent Complexity paper where they prove some interesting things) but this was all done within polynomial factors.

It's fair to say that we would like a quantum theory of gravity to tell us whether or not it is possible, for example, to pack an large number of bits into a small region of space. We won't be able to do any kind of experiment in the forseeable future involving both tiny particles and entire solar systems and galaxies to test any proposed theory of quantum gravity, e.g., string theory, but these questions about the limits of computation – can you build a closed timelike curve? can you build a stable wormhole? or, can you spawn off a "baby universe" to do an unlimited amount of computation for you? – are things that physicists cannot answer because they do not have a quantum theory of space and time.

2.4 Open problem: Public-key cryptography secure against quantum polynomial time adversaries

One topic of particular contemporary interest is the construction of new schemes for public-key cryptography that could be secure against quantum adversaries. (By contrast, *private-key* cryptosystems are not known to be broken by quantum polynomial time adversaries, in general.) One of the reasons for all of the excitement about Shor's algorithm was that the security of cryptosystems based on the RSA function depend crucially on the presumed intractibility of factoring. In fact, most of the currently known constructions for public-key cryptosystems are based on problems in abelian groups – RSA, elliptic curves, Diffie-Hellman, Buchman-Williams, and El Gamal all are – and are therefore not secure against a quantum polynomial time adversary. The one family of known exceptions to this are based on a proposal by Ajtai and Dwork (and later improved by Regev) based on the problem of finding the shortest vector in an integer lattice—essentially, based on instances of the hidden subgroup problem for the dihedral group: a nonabelian group. Thus,

they are not *known* to be secure against a quantum adversary; they are merely not known to be broken. There are also proposals based on braid groups which are not known to be broken by a quantum adversary, but at the same time are not even known to be secure against even a classical adversary under any standard hardness assumptions. An alternative but related subject is quantum key distribution based on the uncertainty principle—by sending polarized photons over a fiber-optic cable, you can obtain a cryptosystem from this quantum mechanical assumption. This is much easier to do than quantum computing since it doesn't involve entanglement, and it has been demonstrated experimentally, but it requires a quantum communication network for coherently sending photons over long distances, which is a hard engineering problem and does not yet exist (and yet, devices for quantum key distribution are already on the market).

The approximate shortest vector problems used by Ajtai-Dwork are *not* NP-complete, but a variant of these problems are, which naturally leads to the long-standing open question of whether or not cryptography can be based on the assumption that P $\neq$ NP—whether or not we can base a cryptosystem on a NP-complete problem. This question is as old as the modern, complexity-theoretic approach to cryptography, and if we believe that NP-complete problems cannot be solved efficiently on a quantum computer, would solve our problem, but the current sentiment is that it seems unlikely. In the first place, the kind of hardness that we need for cryptographic constructions is not just that infinitely many hard instances exist (as suffices for P $\neq$ NP—this would correspond to there *existing* hard-to-decode messages rather than *most* messages being hard to decode) but that there is some NP-complete problem such that the average case is *provably* as hard as the worst case, then we need a way to efficiently sample from this hard distribution to construct a one-way function (which is essential for any kind of cryptography), and then beyond even *that* we actually need a trapdoor one-way function (one that can be efficiently inverted given an additional secret key) for public-key cryptography. In the first place, the worst-case to average-case reduction seems unlikely to exist, and it has been ruled out for nonadaptive reductions by Bogdanov and Trevisan (unless the polynomial-time hierarchy collapses to the third level), building on a similar classic result by Feigenbaum and Fortnow ruling out nonadaptive reductions to instances chosen uniformly at random, itself extending a beautiful argument by Brassard that one-way permutations cannot be based on such reductions unless NP=co-NP. Stronger negative results were recently obtained for the specific problem of basing the constructions of one-way functions on NP-hardness by Akavia, Goldreich, Goldwasser, and Moshkovitz, and also by Pass.

2.5 Open problems: a partial list

We didn't get to discuss any more topics in lecture. The following is a list of suggested open problems and topics for research in quantum complexity theory:

- What is the power of quantum computing with separable mixed states? Closely related: the one-clean-qubit model (Ambainis-Schulman-Vazirani: no gate-by-gate simulation).
- Are there gate sets that yield intermediate power between quantum and classical?
- What is the interplay between noise rates and computational complexity? Related: does BQP = $\mathrm{BPP}^{\mathrm{BQNC}}$?
- Can the nonabelian hidden subgroup problem be solved in quantum polynomial time? Is there an efficient quantum algorithm for graph isomorphism?

- BQP vs. the polynomial hierarchy
- What is the need for structure in quantum speed-ups: P vs. BQP relative to a random oracle. Is there a function with permutation symmetry yielding a superpolynomial speed-up?
- Quantum algorithms for approximating #P-complete problems and its dual: #P-complete problems arising from quantum computation.

6.845 Quantum Complexity Theory
Fall 2010