Classical Complexity Review

Definition (language): consider an alphabet Σ (e.g. $\Sigma = \{0,1\}$) and the set Σ^* of all strings of any length formed by letters of Σ , e.g.

$$\Sigma^* = \bigcup_{n \in \mathbb{N}} \{0,1\}^n$$

A subset L of Σ^* is called a formal language. Languages are very general, for example they could describe the set of all 3-SAT instances with a satisfying assignment, or the set of all local Hamiltonians with ground state energy 0.

Complexity classes are defined as sets of languages ("the languages recognized by the class"). We can think of each *L* as describing a type of problem to solve. Each element $x \in L$ is a specific problem (of fixed size |x| since x is finite length string), e.g. a specific 3-SAT formula or Hamiltonian whose ground state we want to find.

Classical Complexity Review

Definition (Circuit Verifier): a circuit verifier is a family of Boolean circuits $\{V_r\}_{r\in\mathbb{N}}$ satisfying:

1. Each circuit V_r has size r (size = number of gates in the circuit)

2. The output of each V_r is a single bit, representing YES/NO or accept/reject.

Circuit verifiers provide an alternative way to define complexity classes without referring to Turing machines, which generalizes more naturally to the quantum setting.

There is a subtlety when we do this: the mapping $r \to V_r$ itself should be efficiently computable. This stipulation is called uniformity. This is satisfied if we have a simple rule to go from the pattern of gates in V_r to those in V_{r+1} .

We also don't notate the input size of the verifiers at this stage, because the relation of r to the input size is crucial to defining different complexity classes. This relationship is inferred from the definition of each class.

Classical Complexity Review

Definition (polynomial time). $L \in P$ if there exists a polynomial p and a verifier $\{V_r\}_{r\in\mathbb{N}}$ such that

$$x \in L \implies V_{p(|x|)}(x) = 1$$

$$x \notin L \implies V_{p(|x|)}(x) = 0$$

P is regarded as a the set of problems that can be efficiently solved on a classical computer.

Although P includes problems that run in time n^{100} on inputs of size n, which is not truly feasible, we include these problems for two reasons

1) History shows that once a problem is known to be in P, subsequent optimizations can lower its run time.

2) Freeing ourselves from petty details opens our mind to greater things. ☺

Classical Complexity Review

Definition (nondeterministic polynomial time). $L \in NP$ if there exists polynomials p and q, and a verifier $\{V_r\}_{r\in\mathbb{N}}$ such that

$$x \in L \implies (\exists y)\left[|y| = q(|x|) \wedge V_{p(|x|)}(x,y) = 1\right]$$

$$x \notin L \implies (\forall y)\left[|y| = q(|x|) \implies V_{p(|x|)}(x,y) = 0\right]$$

NP is the set of problems with efficiently verifiable proofs. The string y in the definition is often called a witness. We call the case $x \in L$ a yes instance, and say that for all yes instances there is witness that allows the verifier to read the input and the witness, and decide in poly time that $x \in L$.

The case $x \notin L$ is a no instance, and for a no instance any witness y will be rejected by the verifier. This captures the fact that the verifier cannot be fooled or cheated by a false proof.

Classical Complexity Review

Definition (poly-time reduction): A reduction from language A to language B is an efficiently computable function $f : \Sigma^* \to \Sigma^*$ with the property

$$x \in A \Leftrightarrow f(x) \in B$$

Why can't we take $B = \{0\}$, let f be the zero function, and reduce every problem to recognizing 0?

The answer is that the reduction needs to be efficiently computable. We can't map every $x \in A$ to the zero bit unless we already know how to recognize the language A in polynomial time.

If L is a language and every language in NP is poly-time reducible to L, then L is NP-hard.

If L is an NP-hard language and $L \in NP$ then L is called NP-complete.

Classical Complexity Review

Cook-Levin theorem: 3-SAT is NP-complete.

The proof is based on mapping the Boolean circuit verifier to a Boolean proposition that enforces the valid operation of the gates in the circuit at each time step.

If the computation were deterministic, then all the variables in the proposition would be literals. But the fact that the computation is nondeterministic leaves us with an existentially quantified formula,

$$(\exists y) R(x, y)$$

And evaluating the truth value of such existential formulas (where R(x,y) is efficiently computable for fixed x,y) is NP-complete. (Fagin's theorem)

A fancy name for a table that records the history of each time step of a classical computation is the "Cook-Levin tableau."

Classical Complexity Review

Randomized complexity: classical randomized computation can be thought of as Boolean circuits with an unlimited supply of fair coins.

Alternatively, randomized classes can be thought of in terms of counting paths over unconstrained inputs in a deterministic circuit.

Definition (Bounded-Error Probabilistic Polynomial-time). $L \in BPP$ if there exists polynomials p and q, and a verifier $\{V_r\}_{r\in\mathbb{N}}$ such that

$$x \in L \implies \left(2^{-q(|x|)} \left|\{y : |y| = q(|x|) \wedge V_{p(|x|)}(x,y) = 1\}\right|\right) \geq 2/3$$

$$x \notin L \implies \left(2^{-q(|x|)} \left|\{y : |y| = q(|x|) \wedge V_{p(|x|)}(x,y) = 1\}\right|\right) \leq 1/3$$

Classical Complexity Review

For randomized classes, the probability to output 1 when $x \in L$ is called the completeness, and the probability to output 1 when $x \notin L$ is called soundness. (these terms come from logic, an axiomatic system is complete if you can prove everything true, and sound if you cannot prove anything false).

The values of 2/3 and 1/3 are just chosen for convenience, any constants bounded away from ½ would be equivalent because we used parallel repetition to show that

$$BPP(c,s) = BPP(1-(1-c)^L, s^L) \quad , \quad L = \text{poly}(n)$$

For technical reasons BPP does not have any complete problems. This subtlety also occurs for BQP and QMA, the most important quantum complexity classes, so we may discuss it more later. But the solution is to define promise languages:

$$L = L_{\text{YES}} \cup L_{\text{NO}} \quad , \quad L_{\text{YES}} \cap L_{\text{NO}} = \emptyset$$

And only require the BPP machine to decide whether $x \in L_{\text{YES}}$ or $x \in L_{\text{NO}}$ promised that one of these two possibilities is the case. We don't care what happens when x is outside of $L_{YES} \cup L_{NO}$.

Classical Complexity Review

MA (Merlin Arthur) is the generalization of NP to allow for a BPP verifier. The wizard Merlin hands Arthur a witness that he can verify on a BPP machine.

More powerful classical complexity classes: PH, PP, #P, PSPACE. All of these relate to the story of quantum complexity in interesting and vital ways.

The polynomial hierarchy PH is a generalization of NP and co-NP, with a natural complete problem of evaluating Boolean formulas with *k* alternating quantifiers at level *k* of the hierarchy.

PP is a generalization of BPP that removes the bounded-error stipulation, so the acceptance probability for yes instances is anything larger than ½, and for no instances anything small than ½. We will later learn that PP = PostBQP ("quantum computation with post selection") and it will be important to know that PP contains the polynomial hierarchy, $PH \subseteq PP$.

Quantum Complexity Theory

BQP: Bounded-Error Quantum Polynomial-Time. The quantum analog of P (or BPP), this is the set of problems which we regard as efficiently solvable with a quantum computer.

QMA: Quantum Merlin Arthur. The quantum analog of NP (or MA), the hard problems in this class represent problems we are unlikely to be able to solve, even with a quantum computer.

$P \subseteq BQP$, and Shor's proof that $\mathrm{FACTORING} \in \mathrm{BQP}$ provided informal evidence that $BQP \neq P$. Though this is clearly a central question in quantum computing, many of the formal results came later. A major research result last year was an "oracle problem" that is in BQP but not in PH.

Instead, the first major development that kickstarted quantum complexity theory was Kitaev's generalization of the Cook-Levin theorem to the quantum setting in 1999.

This cornerstone result states that the local Hamiltonian problem is QMA-complete. Therefore it is very unlikely it can be solved efficiently in general, even with a quantum computer.

Quantum Complexity Theory

To formally define BQP, we only need to replace our classical circuit verifiers with quantum circuit verifiers (uniform families of quantum circuits) $\{V_r\}_{r\in\mathbb{N}}$.

The classical inputs are replaced with qubit inputs, the Boolean gates are replaced with (local) unitary gates, and the output is the measurement of a single qubit in the computational basis.

Definition (Bounded-Error Quantum Polynomial Time). $L \in BQP$ if there exists a polynomial p and a quantum circuit verifier $\{V_r\}_{r\in\mathbb{N}}$ such that

$$x \in L \implies \Pr\left[V_{p(|x|)}(|x\rangle) = 1\right] \geq 2/3$$

$$x \notin L \implies \Pr\left[V_{p(|x|)}(|x\rangle) = 1\right] \leq 1/3$$

For what range of completeness and soundness to we have $BQP(c,s) = BQP(2/3, 1/3)$?

Quantum Complexity Theory

To go from NP to MA, we allowed the verifier Arthur to be a BPP circuit instead of a P circuit. But QMA does more than just giving Arthur a quantum computer.

In QMA, Merlin is not restricted to sending a classical witness, but rather he can send an arbitrarily complex quantum state, which Arthur plugs into his quantum computer to verify.

Definition (Quantum Merlin Arthur). $L \in QMA$ if there exists polynomials p and q, and a quantum circuit verifier $\{V_r\}_{r\in\mathbb{N}}$ such that

$$x \in L \implies \left(\exists |\xi\rangle \in \mathbb{C}^{2^{q(|x|)}}\right) \Pr\left[V_{p(|x|)}(|x\rangle|\xi\rangle) = 1\right] \geq 2/3$$

$$x \notin L \implies \left(\forall |\xi\rangle \in \mathbb{C}^{2^{q(|x|)}}\right) \Pr\left[V_{p(|x|)}(|x\rangle|\xi\rangle) = 1\right] \leq 1/3$$

3 QMA (Marriott and Watrous , "Quantum Arthur-Merlin Games", 2005)

A QMA verification procedure A is a family of quantum circuits $\{A_x : x \in \Sigma^*\}$ that is generated in polynomial time, together with a function $m \in poly$. The function m specifies the length of Merlin's message to Arthur, and it is assumed that each circuit A_x acts on $m(|x|) + k(|x|)$ qubits for some function k specifying the number of work qubits used by the circuit. As we have done in the previous section, when the input x has been fixed or is implicit we will generally write m to mean $m(|x|)$, k to mean $k(|x|)$, and so forth, in order to simplify our notation. When we want to emphasize the length of Merlin's message, we will refer to A as an m-qubit QMA verification procedure.

Consider the following process for a string $x \in \Sigma^*$ and a quantum state $|\psi\rangle$ on m qubits:

1. Run the circuit A_x on the input state $|\psi\rangle|0^k\rangle$.
2. Measure the first qubit of the resulting state in the standard basis, interpreting the outcome 1 as *accept* and the outcome 0 as *reject*.

The probability associated with the two possible outcomes will be referred to as $\Pr[A_x \text{ accepts } |\psi\rangle]$ and $\Pr[A_x \text{ rejects } |\psi\rangle]$ accordingly.

Definition 3.1. The class $\mathrm{QMA}(a, b)$ consists of all languages $L \subseteq \Sigma^*$ for which there exists a QMA verification procedure A for which the following holds:

1. For all $x \in L$ there exists an m qubit quantum state $|\psi\rangle$ such that $\Pr[A_x \text{ accepts } |\psi\rangle] \geq a$.
2. For all $x \notin L$ and all m qubit quantum states $|\psi\rangle$, $\Pr[A_x \text{ accepts } |\psi\rangle] \leq b$.

For any $m \in poly$, the class $\mathrm{QMA}_m(a, b)$ consists of all languages $L \subseteq \Sigma^*$ for which there exists an m-qubit QMA verification procedure that satisfies the above properties.

Quantum Complexity Theory

For what range of completeness and soundness to we have $QMA(c,s) = QMA(2/3, 1/3)$?

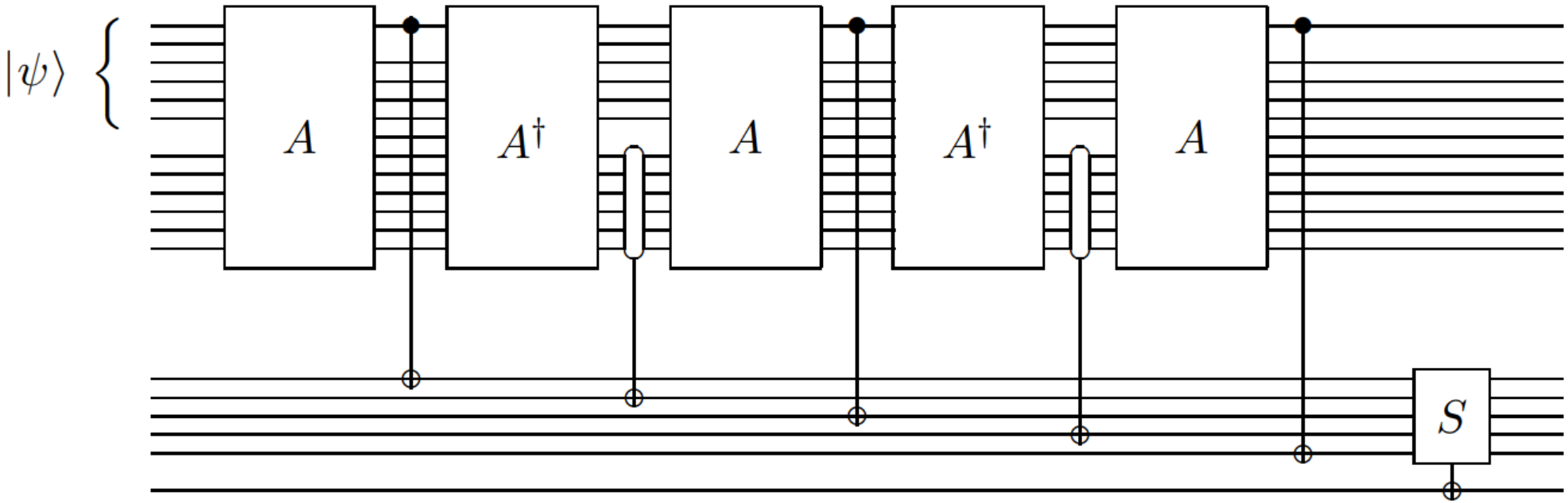
It is possible to amplify the completeness and soundness for QMA as we have done for BPP, MA, and BQP, but the proof for QMA is more complicated because we only get one copy of Merlin's state.

If we are in a YES instance (there exists some state Merlin would like to send to convince us), then Merlin could send us many copies of the state to use for parallel repetition.

But if it is a NO instance, and we allow Merlin to send us a state that he claims to be many copies of the witness, what is to stop him from somehow entangling the copies and gaining an advantage?

Quantum Complexity Theory

The early QMA amplification schemes required slightly increasing the length of the message and upper bounded the extent to which Merlin could cheat. The first "in-place" amplification using one copy of $|\xi\rangle$ was given by Marriott and Watrous.



The intermediate operations acting on the upper ancilla register are measurements resetting them to |00…0>. If we are amplifying QMA(2/3, 1/3), then S is a majority vote.

Quantum Complexity Theory

Definition (the k-local Hamiltonian problem)

Input: a set of m positive semi-definite Hermitian matrices $H_1, ..., H_m$, each acting on $\mathbb{C}^{2^k}$. Each matrix entry has poly(n) bits. The operators norms are bounded by $\|H_i\| \leq 1$ for all i. Each matrix comes with a specification of the k qubits on which it acts (out of the total of n qubits). Also given as input are two numbers a, b (each described by poly(n) bits) with $b - a \geq 1/\mathrm{poly}(n)$.

Output: Is the smallest eigenvalue of $H = H_1 + H_2 + \ldots + H_m$ smaller than *a*, or larger than *b*?

Note that we are promised that the ground energy is below a or above b (and $b - a$ is the promise gap). As with the other bounded-error classes, all the complete problems for QMA are promise problems.

Whether the ground state energy is non-zero or exactly zero, without a promise, is undecidable. If the promise gap is exponentially small (the precise local Hamiltonian problem) then it is complete for QMA with exponentially small bounded error, which turns out to be equal to PSPACE (2016).

Quantum Complexity Theory

Proof Strategy for the Quantum Cook-Levin Theorem

LOCAL HAMILTONIAN $\in$ QMA

To put LH in QMA, we challenge Merlin to send us the ground state, and then we check it using phase estimation. In the NO instance he can't cheat because of the variational principle.

LOCAL HAMILTONIAN is QMA-hard

Here we need to show that finding ground state energies can be as difficult as doing nondeterministic quantum computation. To do this we will map an arbitrary quantum circuit with a constrained output and an unconstrained input register (i.e. a QMA verifier) into the ground state of a local Hamiltonian in such a way that the ground state energy will be sensitive to the acceptance prob of the QMA verifier.

Like the Cook-Levin tableau, these ground states will record the history of a quantum computation in a way that allows us to check validity with local constraints. These are called Feynman-Kitaev history states.

Quantum Complexity Theory

Suppose the QMA verifier runs the sequence of local unitary gates $U_1, ..., U_T$ on the input and the witness. The history of the computational steps looks like this:

$$|x\rangle|\xi\rangle \quad , \quad U_1|x\rangle|\xi\rangle \quad , \quad U_2U_1|x\rangle|\xi\rangle \quad , \quad \ldots \quad , \quad U_T...U_1|x\rangle|\xi\rangle$$

In the classical Cook-Levin proof, we would put each time step on its own set of bits. If we did this in the quantum case, the time steps might look like this:

$$|\psi_{t=0}\rangle|\psi_{t=1}\rangle...|\psi_{t=T}\rangle$$

Again in the classical proof, each gate acts on a few input bits and a few output bits. We can check that the inputs match the correct outputs using a local constraint that only acts on those bits.

We want local constraints that distinguish the state $|\psi_t\rangle|\psi_{t+1}\rangle$ from some other state $|\psi_t\rangle|\psi'\rangle$. What is the problem with this if the $|\psi\rangle$'s are *n* qubit states and we check them with a *k*-local operator?

Quantum Complexity Theory

The problem occurs if $|\psi_t\rangle|\psi_{t+1}\rangle$ and $|\psi_t\rangle|\psi'\rangle$ are both highly entangled states (which is the generic case).

Local observables are only sensitive to local reduced density matrices. If both states are highly entangled, then in both cases the RDMs will be very close to maximally mixed, and so local observables tell us nothing.

Entanglement fundamentally breaks the proof strategy of the classical Cook-Levin theorem.

However, paraphrasing the comments of Mike and Ike on quantum cloning and QKD, whatever quantum takes away with one hand, it gives us something new and beautiful with the other.

In particular, the solution to locally checking the time steps of a quantum computation is to entangle them, instead of recording them on separate registers in a tensor product.

Quantum Complexity Theory

Distilling the previous example, even locally checking the identity gate is impossible to do across a tensor product. This would require distinguishing $|\alpha\rangle|\alpha\rangle$ from $|\alpha\rangle|\beta\rangle$ using a k-local operator, when $|\alpha\rangle, |\beta\rangle$ are arbitrary n-qubit quantum states.

But notice that if we have the state $\frac{1}{\sqrt{2}}(|0\rangle|\alpha\rangle + |1\rangle|\beta\rangle)$, then the RDM of the first qubit (a local RDM) tells us a lot about the relation of $|\alpha\rangle, |\beta\rangle$.

This gives us hope for local constraints if we entangle the time steps of the computation,

$$|\psi\rangle = \frac{1}{\sqrt{T+1}} \sum_{t=0}^{T} U_t \dots U_1 |x, \xi\rangle |t\rangle$$

Which is the (baseline form) of what is called a **Feynman-Kitaev history state**.