CSC2401 Introduction to Complexity, Fall 01-02
Lecture 25, Tuesday, 6th December
Lecturer: Allan Borodin
Scribe: Matei David

1 continued from previous lecture

1.1 Proof of Theorem 5

We are trying to show that $BPP \subseteq \Sigma_2^P$. We pick some $L \in BPP$ and we need to show how $L \in \Sigma_2^P$. Once this is done, it follows that $L \in \Pi_2^P$, as $BPP = co - BPP$.

So let M be a PTM deciding L. We defined

$$S_x \stackrel{def}{=} \{y \in \{0,1\}^{q(n)} : M \text{ accepts } x \text{ with random sequence } y\}$$

So we have:

$$\begin{aligned} x \in L \quad &\Rightarrow \quad |S_x| > \frac{2^{q(n)}}{2} \\ x \notin L \quad &\Rightarrow \quad |S_x| < \frac{2^{q(n)}}{q(n)} \end{aligned}$$

Of course, we can get much better error bounds but this is all we need for what follows. Let $m \stackrel{def}{=} q(n)$. Consider the predicate:

$$\mathcal{H}(S) \stackrel{def}{=} \exists u_1 \ldots u_m : \forall v[u_1 \oplus v \in S \vee \ldots \vee u_m \oplus v \in S]$$

Here the quantification is over $\{0,1\}^m$.
Claim: With the notation as above,

$$\begin{aligned} |S| > \frac{2^m}{2} \quad &\Rightarrow \quad \mathcal{H}(S) \text{ holds} \\ |S| < \frac{2^m}{m} \quad &\Rightarrow \quad \mathcal{H}(S) \text{ false} \end{aligned}$$

Proof of Claim

It is an exercise to show that the following is an alternative characterization of $\mathcal{H}(S)$:

$$\mathcal{H}(S) \Leftrightarrow \exists u_1 \ldots u_m : [(u_1 \oplus S) \cup \ldots \cup (u_m \oplus S) = \{0,1\}^m]$$

In the above, $u \oplus S \stackrel{def}{=} \{u \oplus v : v \in S\}$.
Case 1: S is small, that is, $|S| < \frac{2^m}{m}$.

Then, using the alternative characterization of $\mathcal{H}(S)$, we observe how each term of the form $u_i \oplus S$ in the union has at most as many elements as S, $|u_i \oplus S| < \frac{2^m}{m}$. Then m terms will have at most

$$|\bigcup_{i=1}^{m}(u_i \oplus S)| < m \cdot \frac{2^m}{m} = 2^m = |\{0,1\}^m|$$

Thus the union does not contain all m-bit strings, so $\mathcal{H}(S)$ is false.

Case 2: S is large, that is, $|S| > \frac{2^m}{2}$.

For this, we will use the *probabilistic method*. In essence, the latter goes as folows:

> We want to show that there exists some element satisfying some property. We begin by picking a random element. We then compute the probability that this element satisfies the desired property. If this probability is greater than 0, then we know that at least one element of the set will satisfy the property. Equivalently, we could show that the probability the chosen element does not satisfy our property is strictly less than 1, implying that some choice will indeed satisfy it.

For Case 2, we will use the initial definition of $\mathcal{H}(S)$. Let's fix $v \in \{0,1\}^m$, and let's choose $u_1 \ldots u_m$ independently and uniformly at random in $\{0,1\}^m$. It should be easy to see that $u_i \oplus v$ is uniformly distributed in $\{0,1\}^m$. Then, since $|S| > \frac{2^m}{2} = \frac{|\{0,1\}^m|}{2}$,

$$\Pr_{u_i}[u_i \oplus v \notin S] < \frac{1}{2}$$

Since the u_i's are chosen independently, by repeating the process we obtain that:

$$\Pr_{u_1 \ldots u_m}[u_1 \oplus v \notin S \wedge \ldots \wedge u_m \oplus v \notin S] < \frac{1}{2^m}$$

We now consider choosing every possible v after the u_i's. The probability that that the expression inside the definition of $\mathcal{H}(S)$ does not hold is only increased by a factor of the number of choices of v:

$$\Pr_{u_1 \ldots u_m}[\forall v, u_1 \oplus v \notin S \wedge \ldots \wedge u_m \oplus v \notin S] < 2^m \cdot \frac{1}{2^m} = 1$$

Then, as probability theory tells us, there is at least one choice of the u_i's which falsifies the expression inside the probability. This is exactly what we want:

$$\exists u_1 \ldots u_m : \forall v[u_1 \oplus v \in S \vee \ldots \vee u_m \oplus m \in S] \Rightarrow \mathcal{H}(S) \text{ holds}$$

This concludes the proof of our Claim. It is clear that $\mathcal{H}(S)$ is a Σ_2^P characterization of L, therefore $BPP \subseteq \Sigma_2^P$.

1.2 Proof of Theorem 4

$BPP \subseteq POLYSIZE$.

For some language L, let us define a matrix E, in which the rows are indexed by n-bit strings (2^n of them), as possible inputs to the problem, and the columns are indexed by $q(n)$-bit strings, ($2^{q(n)}$ of them) representing sequences of random bits for the PTM.

Then $E_{x,y} = 1$ if and only if the computation of M on input x with random sequence y yields a correct decision (that is, x is in L iff M accepts it).

We can assume that we have an exponentially small 2-sided error:

$$\forall x, \Pr_y[E_{x,y} = 1] \geq \frac{1}{2^{2n}}$$

Then, in each particular row, there could be at most $\frac{2^{q(n)}}{2^{2n}} = 2^{q(n)-2n}$ entries containing a 0. Since there are 2^n rows, there will be a 0 in at most $2^n \cdot 2^{q(n)-2n} = 2^{q(n)-n}$ columns. So there exists a column y with all 1's, meaning that the computation of M with that particular witness y as the sequence of random bits will always yield the correct decision on any input x.

We can *non-uniformly* find such a y for each n, showing that $BPP \subseteq POLYSIZE$. That is, once such a witness y is known, the Turing machine becomes a deterministic machine which can be simulated by a circuit.

2 $\#\mathcal{P}$ and $\mathcal{IP}$ in 15 minutes or less

2.1 $\#\mathcal{P}$

$\#\mathcal{P}$ is a counting class, containing functions $\Sigma^* \to \Sigma^*$.

An example of one such function is $\#3SAT$: given a $3SAT$ formula F, compute the number of satisfying assignments. The latter can be anything between and including 0 and 2^n.

Formally, $\#\mathcal{P}$ consists of functions $f : \{0,1\}^* \to \{0,1\}^*$ such that $\exists$ polytime nondeterministic TM M and $f(x)$ is the encoding of the number of accepting computations of M.

Clearly $\#3SAT \in \#\mathcal{P}$. It shouldn't be too surprising that $\#3SAT$ is $\#\mathcal{P}$-complete with respect to Turing reducibility ($\leq_T$).

In general, although perhaps not provably, any counting problem defined as $\#\{y : R(x,y)\}$ where $L = \{x : \exists y : R(x,y)\}$ is $\mathcal{NP}$-complete, is itself $\#\mathcal{P}$-complete. Surprisingly, this does not work only for $\mathcal{NP}$-complete problems. For example, $2SAT \in \mathcal{P}$, and still $\#2-SAT$ is $\#\mathcal{P}$-complete.

We also know that $\#Perfect\ perfect\ matchings\ in\ a\ bipartite\ graph$ is $\#\mathcal{P}$-complete, although the decision problem $PerfectMatching \in \mathcal{P}$. This is based on a result of Valiant, that the *permanent* of a $\{0,1\}$ matrix is $\#\mathcal{P}$-complete For reference, the permanent is similar to the determinant, except that the sign of every term (for each of the $n!$ terms) is $+1$ rather than $(-1)^{sgn(\pi)}$ for term associated with the permutation π.

Another very significant result (due to Toda) about this complexity class is

$$PH \subseteq P^{\#P} \subseteq PSPACE$$

2.2 $\mathcal{IP}$ in less than 5 minutes

To show that $x \in L$ for some $L \in \mathcal{NP}$, it is enough to exhibit a certificate y for x which can then be verified in polytime. In reality (whatever that may be), a proof can be interactive. The entity (called *the verifier*) which asks whether $x \in L$ for $L \in \mathcal{IP}$ need not be presented with a certificate y right from the start. Rather, the entity (called *the prover*) which is trying to convince the former that $x \in L$ can provide small bits of information at each step, until the verifier becomes convinced whether $x \in L$ or $x \notin L$.

For pleasant emphasis, lets think of the verifier which tries to find out whether $x \in L$ as *the student*, who is nothing more than a randomized polytime machine; and lets think of the prover which tries to convince the student whether $x \in L$ or not, as *the instructor Borodin*, in reality an oracle for L which "knows" everything about L.

We know that $GISO \in \mathcal{NP}$ (graph isomorphism). If there is one isomorphism, we can simply exhibit it. However, we do not know whether $\overline{GISO} \in \mathcal{NP}$, and we do not think so. Here's how an interactive proof would go, in order to show that $(G_1, G_2) \in \overline{GISO}$, that is, that G_1 is *not* isomorphic to G_2:

- The student randomly picks one of the graphs, constructs an isomorphism of that graph, and presents it to Borodin.
- Borodin examines the isomorphic of one of thsee two graphs, and returns to the student the index of the graph which was picked in the first step.

If the graphs are not isomorphic, Borodin, who knows everything, will always be able to tell which graph was picked in step 1. However, if the graphs are isomorphic, at each step Borodin has a fifty fifty chance of getting the wrong answer.

Interestingly,

$$\mathcal{IP} = \mathcal{PSPACE}$$

The theory was originally developped by Goldwasser, Micali and Rackoff, in the context of cryptographic protocols, where one is interestred in convincing some party that one posseses some key, without giving away the key. In the context of graphs, we do not want to give away the isomorphism. It turns out that (subject to complexity assumptions concerning the existence of trapdoor functions) everything in $\mathcal{IP}$ (and hence including everything in NP) has a "zero-knoweldge proof", without giving away anything more than needed. Maybe it shouldn't come as a surprise that an instructor can often convince a student without the student really being told anything useful for the future. But mathematically this is a very interesting and useful concept. Interactive proofs also led the way for PCPs (probabilistically checkable proofs) which has enabled major progress in our understanding of the apparant (subject to reasonable complexity assumptions) difficulty of certain approximation problems.