E0 224 Computational Complexity Theory **Indian Institute of Science, Bangalore**

Fall 2014 **Department of Computer Science and Automation**

Lecture 21: Oct 27, 2014

Lecturer: Chandan Saha <chandan@csa.iisc.ernet.in> *Scribe: Jaiprakash*

1. Randomized Reduction

We have already defined randomized algorithm. Now, we are going to define the notion of randomized reduction between two languages.

Definition 1.
A language $A \subseteq \{0,1\}^*$ reduces to language $B \subseteq \{0,1\}^*$ under a randomized polynomial time reduction, denoted by $A \leq_r B$, if there exist a probabilistic TM M such that for every $x \in \{0,1\}^*$, $Pr[B(M(x)) = A(x)] \geq 2/3$.

It says that, if $B \in$ **BPP** and $A \leq_r B$ then $A \in$ **BPP**.

Definition 2.
Class BP.NP:
BP.NP $= \{ L : L \leq_r$ **3SAT** $\}$
e.g. **Unique-SAT** $\in$ **BP.NP** .

Assignment3 1.
If $\overline{3SAT} \in$ **BP.NP** then **PH** $= \sum_3^P$.

2. Randomized Space-Bounded Computation:

We have seen the definition of space-bounded computation. Here we are extending it to the probabilistic setting.

Definition 3.
A PTM M has space complexity $s(n)$ if for every input of size n and every path of computation on input x of length n, M halts using no more than $s(n)$ cells in the work tapes.

Definition 4.
Class BPL A language $L \subseteq \{0,1\}^*$ is in **BPL** if there is a probabilistic TM M with space complexity $O(\log n)$ such that,
$Pr[M(x) = L(x)] \geq 2/3$.

Definition 5.
Class RL A language $L \subseteq \{0,1\}^*$ is in **RL** if there is a probabilistic TM M with space complexity $O(\log n)$ such that,
$x \in L$ then $Pr[M(x) = 1] \geq 2/3$, and
$x \notin L$ then $Pr[M(x) = 1] = 0$.

Assignment3 2.
Prove that **BPL** $\subseteq$ **P** .

UPATH := $\{< G, s, t >: G$ is a undirected graph and there is a path from s to t $\}$.

i.e. given a n vertex directed graph G and two vertices s and t, determine whether s is connected to t in G.

Theorem 1.
UPATH $\in$ **RL** [1]

Proof idea : Initialize the variable v to the start vertex s and in each step choose a random neighbor u of v, and set $v \leftarrow u$ (i.e. take a random walk starting from s). If the walk reaches to t within $100n^4$ steps then accept, otherwise reject. This procedure takes log-space (a counter to store the number of steps and to store the current node pointer).

Theorem 2.
Any language in **BPL** can be decided by a deterministic $O(\log^{3/2} n)$ space TM. [1]

3. Interactive Proof(IP) Systems

If the verifier wants to check that a statement (given by prover) is true or not, one way is, the prover provides a certificate(proof) and the verifier checks the validity of the certificate, and another way is, the verifier and the prover interact with each other. The prover is all powerful and have unlimited computational resources but cannot be trusted, while the verifier has limited computational power and is honest. The verifier asks series of explanation to the prover, before he is "convinced" (either the prover is trusted or not).
Does Interaction between prover and verifier give additional power?

3.1. Some Remarks:

IP provides a general framework to convince the verifier when the prover cannot be trusted.

1. **Color blind problem:** Suppose there are two persons in the system named as P1 and P2. Suppose P1 is color blind and he has two balls, P2 is claiming to P1 that these balls have different colors, one red the other yellow. But, P1 does not trust P2. How can P1 convince himself that the balls indeed have different colors ?

   IP can help P1 in this scenario. P1 asks P2 the color of balls, and holds red ball (as said by P2) in right hand, and yellow ball (as said by P2) in left hand. Then P1 turns his back to P2 and tosses a coin. If the coin comes up "head" then P1 keeps the ball as it is, otherwise he switches the balls. Now again P1 asks P2 the color of balls. If P2 is honest then he will say correct colors of given balls. But if P2 is dishonest

then P2 is not able to guess the answer with probability better than 1/2. P1 repeats this procedure 100 times and finds out if P2 is lying with very high probability.

2. Cryptographic Protocols : **IP** underlies a huge research effort in cryptography, particularly in the study of zero-knowledge protocols.

3. **I**P captures whole **PSPACE** and hence gives an alternate characterization of the class **PSPACE**. Study of the class **IP** formed the background for another important result in complexity theory - the **PCP** theorem.

3.2. Interactive Proof with deterministic verifier and prover:

In this section we consider deterministic verifier and prover.

Let us consider the well known **3SAT** problem. Suppose the verifier has a 3CNF boolean formula, the aim is to check the satisfiability of formula using interaction with prover. The verifier proceeds clause by clause in the 3CNF formula and asks the prover the values of each literal in that clause. If there is no conflict in the literals' values given by prover in every step and these values satisfy the given claues, then verifier is convinced that given clause is satisfiable.

Definition 6.

Interaction between two deterministic functions: Let $f, g : \{0,1\}^* \to \{0,1\}^*$ be functions and $k \geq 0$ be an integer (allowed to depend upon the input size). A k-round interaction between f and g on input $x \in \{0,1\}^*$, denoted by $< f, g > (x)$ is the sequence of strings $a_1, a_2, ..., a_k \in \{0,1\}^*$ defined as follows:
$a_1 = f(x)$
$a_2 = g(x, a_1)$
$a_3 = f(x, a_1, a_2)$
$a_4 = g(x, a_1, a_2, a_3)$
$\vdots \quad \vdots$
$a_k = \cdots$
The output of f, denoted by $OUT_f(< f, g > (x))$ is $f(x, a_1, a_2, ..., a_k)$.

Definition 7.

Class DIP: A language $L \subseteq \{0,1\}^*$ is in **DIP**, if there is a deterministic Turing machine V(verifier) which on input $x, a_1, ..., a_i$ runs in time $poly(|x|)$, and interacts with a prover function $P : \{0,1\}^* \to \{0,1\}^*$, for $poly(|x|)$ many rounds such that,
$x \in L \Rightarrow \exists P$ s.t $OUT_V(< V, P > (x)) = 1$ and, (Completeness)
$x \notin L \Rightarrow \forall P$ s.t $OUT_V(< V, P > (x)) = 0$ (Soundness)

Theorem 3.

DIP=NP

Proof. 1)Let $L \in$ **NP**, then L has one round deterministic proof(certificate). Therefore $L \in$ **DIP**.

2)Let $L \in$ **DIP**, then there exists a transcript $(a_1, a_2, .., a_k)$ for $k =$ polynomial in size of input.

Let V be the verifier, checks indeed,

$V(x) = a_1$,

$V(x, a_1, a_2) = a_3$,

$V(x, a_1, a_2, a_3, a_4) = a_5$,

$\vdots \qquad\qquad \vdots$

$V(x, a_1, ..., a_k) = 1$, [By definition of **DIP**]

Here the transcript serves as the certificate for $x \in L$. Therefore L is in **NP**. □

3.3. Probabilistic Interactive Proofs

In the previous section we have seen that **DIP=NP**, i.e by assuming the verifier is deterministic we are not getting anything new. In order for interaction to provide any extra power (over the class NP), we make the verifier probabilistic (i.e verifier questions will be computed using a probabilistic algorithm). The verifier can accept the proof for wrong statement with small probability($\leq \frac{1}{3}$) and can reject the proof for a wrong statement with high probability($\geq \frac{2}{3}$) regardless of the strategy the prover uses.

Definition 8.

Probabilistic verifier function: In order to model an interaction between f(probabilistic verifier function) and g(deterministic prover function), f works on an additional random string $r \in_r \{0,1\}^m$.
An interaction between f and g on input x is a sequence of strings,
$a_1 = f(x, r)$
$a_2 = g(x, a_1)$
$a_3 = f(x, r, a_1, a_2)$
$a_4 = g(x, a_1, a_2, a_3)$
$\vdots \quad \vdots$
$a_k = :.....$
Since the verifier can see the random string r, and the prover doesn't have access to r, so we call this *private coin model*.

Definition 9.

Class IP[k]: A language $L \subseteq \{0,1\}^*$ is in **IP** if there is a probabilistic Turing machine V that on input $x, r, a_1, a_2, ..., a_i$ runs in time $poly(|x|)$ and interacts with a prover function $P : \{0,1\}^* \to \{0,1\}^*$ for an integer $k \geq 1$(may depend on the input length) rounds such that,
$x \in L \Rightarrow \exists P$ s.t $Pr_r[OUT_V(< V, P > (x)) = 1] \geq 2/3$, and (Completeness)
$x \notin L \Rightarrow \forall P$ s.t $Pr_r[OUT_V(< V, P > (x)) = 1] \leq 1/3$ (Soundness)

Definition 10.

Class IP: IP=$\bigcup_{c>0}$ **IP**$[n^c]$

3.4. Remarks:

1. **IP=PSPACE** (we will see a proof of this later).

2. Class **IP** definition is unchanged if we replace the completeness parameter $2/3$ by $1-2^{-n^s}$ and the soundness parameter $1/3$ by 2^{-n^s} for any fixed constant $s > 0$

3. In fact, completeness probability can be assumed to be 1 without loss of generality. (we will see this when we prove **IP=PSPACE**).

4. Irrespective of completeness probability, soundness probability =0 implies the class is **NP**.

5. Allowing the prover to use random bits doesn't give us any additional power(simple averaging argument).

References

[1] S.ARORA and B.BARAK Computational Complexity: A Modern Approach, Cambridge University Press, 2009

[2] Michael Sipser "Introduction to Theory of Computation", Cengage Learning