

Probabilistic Algorithms and Proofs

Lecture by Russell Impagliazzo
Notes by William Matthews

Lecture May 26, 2010

1 Probabilistic Algorithms

Recall that $L \in BPP$ if exists TM $A \in P$ such that $x \in L \implies \Pr_r[A(x, r) = 1] \geq .9$ and $x \notin L \implies \Pr_r[A(x, r) = 0] \geq .9$.

Lemma 1. *If $L \in BPP$, for all polynomials $q(n)$ there is an algorithm $A \in P$ such that $x \in L \implies \Pr_r[A(x, r) = 1] \geq 1 - 2^{-q(|x|)}$ and $x \notin L \implies \Pr_r[A(x, r) = 0] \geq 1 - 2^{-q(|x|)}$.*

Proof. Let A_1 be an algorithm with error probability $< .1$ (as in the definition of BPP .) For $k = 16q(|x|) + 1$, let $A_2(x, (r_1, \dots, r_k))$: run $A_1(x, r_i)$ for each i and output the majority answer.

A_2 is wrong only if there exists a set $S \subseteq \{1, \dots, k\}$ such that $|S| > k/2$ and $A_1(x, r_i)$ is wrong for all $i \in S$.

$\Pr[A_2 \text{ is wrong}] \leq \binom{k}{k/2} \max \Pr[A_1(x, r_i) \text{ is wrong for all } i \in S] \leq \binom{k}{k/2} (.1)^{k/2} \leq 2^k / 10^{k/2} = (4/10)^{k/2} \leq 2^{-k/2} \leq 2^{-q(|x|)}$. \square

Recall $P/poly$ is the class of problems solvable with a series of polynomial size (non-uniform) circuits.

Claim 1. $BPP \subseteq P/poly$

Proof. Apply Lemma 1 with $q(n) = 2n$ to get A such that for all x , $\Pr[A(x, r) \text{ is wrong on } x] \leq 2^{-2n}$. The probability that there exists an x such that the algorithm makes a mistake is at most $2^n 2^{-2n} = 2^{-n}$. Thus, with positive probability, the algorithm makes no mistakes on strings of length n . Thus, there exists a sequence of random bits r such that the algorithm makes no mistakes on strings of length n . Let $C(x)$ be a circuit which simulates $A(x, r)$. Thus, $C(x)$ never makes a mistake on x . \square

2 Probabilistic Proofs

NP = statements with short “proofs” (certificates, solutions) that they are true that are easily verifiable. We can think of NP as a prover (in exponential time, or more) giving a proof to a verifier in P .

What if instead, the verifier is a randomized polynomial time algorithm?

Example: Limited Diophantine Equations (LDE). Given a formula ϕ and number m , does there exists x_1, \dots, x_n such that $\phi(x_1, \dots, x_n) = 0$ and $x_i \in \{-m, \dots, m\}$.

By construction, if $\langle \phi, m \rangle \in LDE$, then there is a poly-size witness x_1, \dots, x_n . However, computing $\phi(x_1, \dots, x_n)$ might involve exponentially large numbers.

Prover: Choose x_1, \dots, x_n

Verifier: Choose a moderate size prime modulus q , and then verify that $\phi(x_1, \dots, x_n) \pmod q = 0$.

If $\langle \phi, m \rangle \in LDE$ then the verifier will always accept.

If $\langle \phi, m \rangle \notin LDE$ then the verifier rejects with high probability.

LDE is in the class MA “Merlin Arthur” (Think of Merlin “magically” coming up with the proof, and Arthur using some randomness to check the answer.)

In MA , Merlin talks and Arthur just listens and then verifies what Merlin says.

What if Arthur talks too? In this case, it requires that Arthur is randomized (otherwise Merlin would already know what Arthur was going to say.)

Example: Graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there exists a bijection $\pi : V_1 \rightarrow V_2$ such that $(u, v) \in E_1 \iff (\pi(u), \pi(v)) \in E_2$.

Consider the following probabilistic interactive proof that two graphs are *not* isomorphic.

Arthur (Verifier): Flip a coin $b \in \{1, 2\}$ choose H , a random permutation of G_b , and send H to Merlin.

Merlin (Prover): (Compute as needed), reply with b' such that H is isomorphic to G'_b

Arthur: Accept if $b = b'$

If the graphs are *not* isomorphic, then Merlin can always cause Arthur to accept. If the graphs are isomorphic, then Arthur accepts with probability $1/2$. (This can be repeated as necessary to decrease the error. The repetitions may be done in parallel: Choose b_1, \dots, b_k , construct and send H_1, \dots, H_k ; Merlin then sends back b'_1, \dots, b'_k .)

This problem is in the class $AM[2]$ (“Arthur Merlin 2,” since Arthur speaks first and there are 2 rounds of interaction.)

At the end of the protocol, note that Arthur learns *nothing* more than whether G_1 and G_2 are isomorphic. In particular, he doesn’t learn the isomorphism. Such a proof system is said to be a zero knowledge interactive proof system.

Some claims without proofs:

Theorem 1. $AM[O(1)] = AM[2]$

Theorem 2. $AM[2] \subseteq NP/poly$

Theorem 3. If $coNP \subseteq AM[2]$, then $PH = \Sigma_3$.

Corollary 1. If Graph Isomorphism is NP -complete then $PH = \Sigma_3$.

What about $AM[poly]$?

Theorem 4. $AM[poly] = PSPACE$