6.841 Advanced Complexity Theory May 2, 2007

Lecture 22

Lecturer: Madhu Sudan *Scribe: Jelani Nelson*

1 Overview

This lecture describes a natural DNP-complete problem first proposed by Impagliazzo and Levin [1].

2 Universal Problems for DNP

We begin by recalling the definition of the complexity class DNP (Distributional NP) from last lecture.

Definition 1 DNP *is the class of languages* (π, D) *where* π *is an efficiently computable function specifying an NP language (given an input* x *find a poly-size witness* y *such that* $\pi(x, y) = 1$*), and* D *is a poly-time sampleable distribution. That is, there is a uniform poly-time algorithm* G *such that* $G(\{0,1\}^n) \subseteq \{0,1\}^n$*, and an input* x *of length* n *is drawn according to the distribution* $G(U_n)$*, i.e. if* D_n *is the distribution on length-*n *inputs specified by* D *then* $D_n = G(U_n)$*. Here* U_n *is the uniform distribution on* $\{0,1\}^n$*.*

A reduction from a DNP language A to a DNP language B using randomized reductions R, T is said to succeed on an input x if the random variable $R(x)$ is distributed according to D_2, and for the y found serving as a witness to $R(x)$ the string $T(x, R(x), y)$ serves as a witness to x. If we have an R, T that succeed with probability at least $1/\text{poly}(n)$ for each x then we have a valid reduction from A to B.

Now consider a universal language π_{univ} such that $\pi_{univ}((M, x'), y) = 1$ iff $M(x', y) = 1$ where M is the Turing-machine description of a poly-time computable function. Then we can reduce a DNP language (π, D) to (π_{univ}, D') where D' has the same distribution over x as D and chooses the machine M according to some probability distribution such that each machine gets picked with constant probability (e.g. the ith machine is chosen with probability 2^{-i}). We do not get a single DNP-complete language since D' differs based on the language we are reducing from, but we do get a single language that all DNP-problems with distribution D reduce to. Impagliazzo and Levin showed in [1] that for every language (π, D) there is a language $(\tilde{\pi}, U)$ that (π, D) reduces to, where U is the uniform distribution. By composing the universal language reduction with their reduction, we get that (π_{univ}, U) is DNP-complete. The rest of this lecture describes the proof of their result.

3 A Special Case of [Impagliazzo, Levin]

Recall that for a language (π, D) there is a uniform poly-time algorithm G for sampling from D. If G were 1-to-1 then $D = U$, so (π, D) reduces to (π_{univ}, U). Now we discuss the case where G is 2^ℓ-to-1 and we know ℓ. The idea for dealing with this case is to reduce it to the 1-to-1 case.

We use pairwise independent hashing. If G is 2^ℓ-to-1 then its image size is $2^{n-\ell}$. We pick a random pairwise independent hash function h from $\{0,1\}^n$ to $\{0,1\}^{n-\ell+2}$ (the reason for the "2" will become clear later) and hope that we have no collisions. Our input to the new problem $\tilde{\pi}$ will be $(h, h(x))$ where $\tilde{\pi}((h, z), (s, y)) = 1$ iff $h(G(s)) = z$ and $\pi(G(s), y) = 1$. To map the witness (s, y) back to our original problem π, we set $T(x, s, y) = y$ if $x = G(s)$ (i.e. a witness for $\tilde{\pi}$ was not found for some $x' \neq x$ that collided with x under h); otherwise we label our attempt at a reduction as a failure.

To bound the success of our reduction, we first need the following claim.

Claim 2 *Pr*$[\forall x' \in G(\{0,1\}^n) - \{x\},\ h(x') \neq h(x)] \geq 3/4$.

Proof Fix $x' \neq x$. Since h is pairwise independent we have $\Pr[h(x') = h(x)] \leq 1/|\text{range}(h)|$, so by the union bound we have $\Pr[\exists x' \in G(\{0,1\}^n) - \{x\},\ h(x') = h(x)] \leq |G(\{0,1\}^n)|/|\text{range}(h)| = 2^{n-\ell}/2^{n-\ell+2} = 1/4$. ■

Now to analyze the probability of success of our reduction, let E be the event that $(h, h(x))$ uniquely specifies x. By Claim 2 $\Pr[E] \geq 3/4$. If $\tilde{\pi}$ is easy, then there is some AvgBPP algorithm A that fails to find a witness on a negligible fraction of (h, z) pairs (where (h, z) is drawn under the uniform distribution). Let B be this set of (h, z) pairs where A fails so that $\Pr_{h,z}[(h, z) \in B] = \delta$ is negligible. Then we have

$$\begin{aligned}
\Pr_{x \in G(\{0,1\}^n),h}\left[(h, h(x)) \in B\right] &\leq \Pr_{x,h}\left[(h, h(x)) \in B | E\right] \cdot \Pr[E] + \Pr\left[\neg E\right] & (1)\\
&= \Pr_{h,z}[(h, z) \in B | E] \cdot \Pr[E] + \Pr[\neg E] & (2)\\
&\leq \frac{\Pr_{h,z}[(h, z) \in B]}{\Pr[E]} \cdot \Pr[E] + \Pr[\neg E] & (3)\\
&\leq \delta + 1/4 & (4)
\end{aligned}$$

Line (2) follows from (1) since the event E occurring implies z uniquely specifies x.

Now the probability that our reductions fails to work is at most $\Pr_h[h(x) \text{ does not uniquely specify } x] + \Pr_{x,h}[(h, h(x)) \in B]$ by the union bound. By Claim 2 and the above analysis, this quantity is at most $1/4 + (\delta + 1/4) = 1/2 + \delta$, which is at least $1/\text{poly}(n)$, and thus (π, D) reduces to $(\tilde{\pi}, U)$.

4 The General Case

In the previous section we assumed that G was 2^ℓ-to-1 and that we knew ℓ. In reality G can be any function from $\{0,1\}^n$ to $\{0,1\}^n$, and there might not even be an "ℓ" to know! The idea of [1] to overcome this obstacle might be reminiscent of the protocol of Goldwasser and Sipser [2] for approximating the size of a set.

We do the following upon being given an input x for (π, D):

1. Guess $\ell \in [0, n]$ at random. In the analysis, you should think about "the right ℓ" to guess being the ℓ such that there are approximately $2^{n-\ell}$ other y's such that $|G^{-1}(x)| \approx |G^{-1}(y)|$ (within a factor of 2).
2. Guess $k \in [0, n]$ at random such that $|\{s | G(s) = x\}| \approx 2^k$ (again, within a factor of 2).
3. Pick a random pairwise independent hash function $h : \{0,1\}^n \to \{0,1\}^{n-\ell+O(1)}$.
4. Pick a random pairwise independent hash function $\tilde{h} : \{0,1\}^n \to \{0,1\}^{k+O(1)}$.
5. Pick $w \in \{0,1\}^k$ uniformly at random.

Now the input of our reduction to $\tilde{\pi}$ is $(k, \ell, h, h(x), \tilde{h}, w)$. Our new language $(\tilde{\pi}, U)$ will be such that $\tilde{\pi}((h, z, \tilde{h}, w), (s, y)) = 1$ iff $h(G(s)) = z$, $\pi(G(s), y) = 1$, and $\tilde{h}(s) = w$. We then transform a witness (s, y) for $\tilde{\pi}$ to a witness for π by outputting y if $G(s) = x$ and labeling our reduction attempt a failure otherwise.

The analysis of the general case conditions on $h, z, \tilde{h}$, and w uniquely specifying s then proceeds as in Section 3. We omit the details, but it can be shown that s being specified uniquely happens with non-negligible probability (with probability at least $\Omega(1/n^2)$ — essentially guessing k, ℓ is the limiting factor).

This completes the proof that (π_{univ}, U) is DNP-complete. We glossed over one detail and that is how to represent the (M, x), the inputs to π_{univ}, as single strings. This can be done by writing $x' = (\langle M \rangle, x)$ where $\langle M \rangle$ is a prefix-free encoding of the description of the machine M. A prefix-free encoding is a mapping from integers to $\{0,1\}^*$ such that no encoding of one integer is a prefix of the encoding of another integer. A simple such encoding is to map the integer represented in binary as $b_1 b_2 \ldots b_k$ to the string $b_1 b_1 b_2 b_2 \ldots b_{k-1} b_{k-1} b_k \overline{b_k}$.

References

[1] Russell Impagliazzo, Leonid A. Levin. No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random. In *Proc. 31st Annual Symp. on Foundations of Computer Science (FOCS)*, pages 812–821, 1990.

[2] Shafi Goldwasser, Michael Sipser. Private Coins versus Public Coins in Interactive Proof Systems. In *Proc. 18th Annual ACM Symp. on Theory of Computing (STOC)*, pages 59–68, 1986.