Derandomization, part 4

Jonathan Kelner
4/7/2011

$$G_r(\sigma) = L(\sigma|S_1)L(\sigma|S_2)\ldots L(\sigma|S_r)$$

- **Claim:** If $H_1(m) \geq r/\epsilon$ and $\epsilon \leq 1/r$, gives an $\epsilon$-PRG for circuits of size $r$

  - **Proof:**
    - Suppose for sake of contradiction that have circuit $C$ of size $r$ s.t.
      $$\left| \Pr_{\sigma \in (0,1)^{r \cdot (m)}} [C(G_r(\sigma)) = 1] - \Pr_{\rho \in (0,1)^{r \cdot (m)}} [C(\rho) = 1] \right| \geq \epsilon$$
    - By indistinguishability vs. predictability result, enough to show
      $$\Pr[G_r(\sigma_1),\ldots,G_r(\sigma_{r-1}) = G_r(\sigma_r)] \geq \frac{1}{2} + \frac{\epsilon}{r}$$
    - Let $y_i = G_r(\sigma)_i = L(\sigma|S_i)$ for all $i$, and let $x = \sigma|S_i$, $q = \sigma|S_i^c$.
    - Can predict $y_i = L(x)$ from $y_{i-1},\ldots,y_{i-1}$. Get contradiction if can predict it from $x$.
    - By averaging argument, can set $q$ to fixed value s.t. prob over vals of $x$ s.t.
      $$|S_i \cap S_i^c| = \log r$$
    - $y_i$ only depends on $|S_i \cap S_i^c| = \log r$ bits of $x$, so can compute with circuit of size $r$.
    - Get total circuit of size $\leq r^2 \leq \epsilon r$, computing $L(x)$ with prob $\geq 1/2 + \epsilon/2r$, which gives contradiction.

**Last Time**

- **Definition:** Average-case hardness $H_1(m)$ of lang $L$ on inputs of length $m$ is largest $s$ s.t. no circuit of size $\leq s$ can match $L$ on at least $1/2 + 1/s$ fraction of inputs of length $m$
- **Definition:** Let $\ell > m > d$. A family of subsets $S_1,\ldots,S_r \subseteq \{1,\ldots,\ell\}$ is a $(\ell,m,d)$-design of size $r$ if $|S_j| = m$ for every $j$ and $|S_j \cap S_k| \leq d$ for every $j \neq k$.
- For given $r,m$, let $d(m) = O(m^2)$. We showed we can get a $(\ell,m,\log r)$-design $S_1,\ldots,S_r$.
- Nisan-Wigderson PRG $G_r$ will be given by
  $$G_r(\sigma) = L(\sigma|S_1)L(\sigma|S_2)\ldots L(\sigma|S_r)$$
- **Claim:** If $H_1(m) \geq r/\epsilon$ and $\epsilon \leq 1/r$, this is an $\epsilon$-PRG for circuits of size $r$.

**Design Parameters and Resulting Derandomizations**

- Need $r$ sets ($\approx$ random bits used by algorithm)
  - Can be as big as running time of machine-poly(n)
- Random seed is of size $\ell$-size of union of all the $S_i$ (i.e., universe they live in)
- Simulation tries all random seeds, runs poly time algorithm for each
  - So running time grows like poly(n)*$2^\ell$.
- Need intersections to be of size $O(\log(n))\cdot\Omega(\log(r))$
- We gave construction with:
  - Number of sets $\approx r$
    - Size of sets $m$ has to be $\geq r\log(r)$.
    - Intersections of size $O(\log(r))$ (think of as being at most small const * m).
    - Random seed of size $\ell \cdot r \cdot (\log(n))$.
    - So running time $= \text{poly}(n)^*2^{\text{poly}(\log(\log(n)))}$
  - So running time grows like poly(n)*$2^{\text{poly}(\log(\log(n)))}$.
- Basic reason for not getting polynomial running time:
  - Number of sets exp in square root of seed size, so need seed to be $\Omega(\log(n))$, not $O(\log(n))$
  - When set size $= O(\log(n))$, can give a better construction where number of sets is exponential in seed size $\ell$
    - Just greedily choose the sets, give counting argument
    - Everything is of size $\log(n)$, so can do brute force alg in poly time
    - See original paper (posted on course website).
- Let's seed be of size $O(\log(n))$, gives running time of poly(n)

4/12/11
Derandomizing Space-Bounded Computation

- Will look at small space bounds (e.g., log), where need to be a little careful with defs
  - **Time bounds**: will require randomized computations to terminate in time exponential in space bound
  - **Access to random tape**: rand tape unidirectional
  - Can fix input $x$, think of distinguisher only as function of random bits
- Will see if we can make PRGs against these
  - Will not require unproven complexity assumptions

Reduction to Automata

- Will do something stronger: fool poly-sized automata
- Definition: $(m,k)$-automaton $Q$ is finite state machine with:
  - States $1,\ldots,m$
  - $2^k$ transitions per state, correspond to strings in $\{0,1\}^k$
  - Will identify with its transition function
  - $Q(i;x)=j$ means goes from $i$ to $j$ when fed input $x$
- Given $\text{BSPACE}(S)$ machine with $n$ rand bits, get $(m,k)$-automaton with $m=2^{O(S)}$, $k=O(S)$
- Get random bits in blocks of size $k$
- Assume WLOG that $n$=power of 2, unique, absorbing accept state
- Want $G:\{0,1\}^\ell \to \{0,1\}^n$ that fools these
  - Will find one with $\ell=O(S \log n)=O(\log m \log n)$
  - If start with log space, will use $O(\log^2)$ input bits
- What will this show?