The Goldwasser-Sipser Lower-Bound Protocol

This material was presented in class on March 5, 2015.

The Goldwasser-Sipser lower-bound protocol uses a pairwise-independent hash-function family $\mathcal{H}_{n,k}$. Pairwise-independent hash-function families were defined in class on March 3, and examples were given.

Suppose that S is a subset of $\{0,1\}^n$ in which membership can be certified (in the NP sense). Both Arthur and Merlin know an integer K. Merlin's goal is to convince Arthur that $|S| \geq K$. We give a protocol with the property that, if $|S| \geq K$, *i.e.*, if Merlin is making a correct claim, then Arthur accepts with high probability, and, if $|S| \leq \frac{K}{2}$, *i.e.*, if Merlin is making a claim that is not just incorrect but *far from correct*, then Arthur rejects with high probability. There is no requirement on what Arthur will do if $\frac{K}{2} < |S| < K$. Let $\mathcal{H}_{n,k}$ be a pairwise-independent hash-function family, where $2^{k-2} < K \leq 2^{k-1}$.

LBP(S, K)

A: Choose $h \in_R \mathcal{H}_{n,k}$ and $y \in_R \{0,1\}^k$.
A $\to$ M: (h, y)
M: Find $x \in S$ such that $h(x) = y$.
M $\to$ A: (x, c), where c is a certificate of $x \in S$
A: Accept if and only if $h(x) = y$ and c is valid.

Let $p^* = \frac{K}{2^k}$ and $p = \frac{|S|}{2^k}$. Assume that $|S| \leq 2^{k-1}$. Note that $K \leq 2^{k-1}$ and that Merlin is trying to convince Arthur that $|S| \geq K$; so, if $|S| > 2^{k-1}$, Merlin can just choose a subset T of S such that $|T| \leq 2^{k-1}$ and convince Arthur that $|T| \geq K$, which implies that $|S| \geq K$; so we lose nothing by assuming that $|S| \leq 2^{k-1}$. We claim that

$$p \geq \mathrm{Prob}_{h,y}\left(\exists x \in S : h(x) = y)\right) \geq \frac{3p}{4}. \tag{1}$$

To see that the upper bound of p in (1) is correct, observe that $|h(S)| \leq |S|$, for any function h. The probability that y chosen uniformly at random from $\{0,1\}^k$ is in $h(S)$ is just $\frac{|h(S)|}{2^k} \leq \frac{|S|}{2^k} = p$.

We can actually prove the lower bound of $\frac{3p}{4}$ in (1) for any y, not just a random y. Let x be an element of S and E_x be the event that $h(x) = y$ for an h chosen uniformly at random from $\mathcal{H}_{n,k}$. Note that the definition of pairwise-independent hash-function families give us $\mathrm{Prob}[E_x] = 2^{-k}$. In (1), we have

$$\mathrm{Prob}_h\left(\exists x \in S : h(x) = y\right) = \mathrm{Prob}_h\left(\bigvee_{x \in S} E_x\right). \tag{2}$$

By the inclusion-exclusion principle (2) is at least

$$\left(\sum_{x \in S} \mathrm{Prob}(E_x)\right) - \frac{1}{2}\left(\sum_{x \neq x' \in S} \mathrm{Prob}(E_x \wedge E_{x'})\right), \tag{3}$$

and the definition of pairwise-independent hash-function families tells us that $\text{Prob}(E_x \wedge E_{x'}) = 2^{-2k}$. So (3) is at least

$$\begin{aligned} &\frac{|S|}{2^k} - \frac{|S|(|S|-1)}{2 \cdot 2^{2k}} \\ &> \frac{|S|}{2^k} - \frac{|S|^2}{2^{2k+1}} \\ &= \frac{|S|}{2^k}\left(1 - \frac{|S|}{2^{k+1}}\right) \\ &\geq p\left(1 - \frac{2^{k-1}}{2^{k+1}}\right) = \frac{3p}{4}. \end{aligned}$$

We can now state precisely what LBP does in the two cases we're interested in: If $|S| \geq K$, then the probability that Arthur accepts in a single execution of LBP is at least

$$\frac{3p}{4} = \frac{3}{4} \cdot \frac{|S|}{2^k} \geq \frac{3}{4} \cdot \frac{K}{2^k} = \frac{3}{4}p^*.$$

On the other hand, if $|S| \leq \frac{K}{2}$, then the probability that Arthur accepts in a single execution of LBP is at most

$$p = \frac{|S|}{2^k} \leq \frac{1}{2} \cdot \frac{K}{2^k} = \frac{1}{2}p^*.$$

To achieve the high-probability result that we want, we just amplify this gap of $\frac{1}{4}p^*$ in the acceptance probabilities of the two cases by running M independent trials of LBP. If Merlin is making a true claim, the expected number of accepts is at least $\frac{3M}{4}p^*$, and, if he is making a far from true claim, the expected number is at most $\frac{M}{2}p^*$; moreover, M can be chosen so that the probability of fewer than $\frac{M}{2}p^*$ accepts in the first case or more than $\frac{3M}{4}p^*$ in the second is negligible.