6.895 Randomness and Computation April 5, 2006

Lecture 15

Lecturer: Ronitt Rubinfeld *Scribe: Jinwoo Shin*

Today, we will review some linear algebra facts which need for us, and show how a rapidly mixing graph helps randomness.

1 Linear Algebra Review

Definition 1 *ν is an eigenvector of A with the corresponding eigenvalue λ if $\nu A = \lambda\nu$.*

Definition 2 *The L_2-norm of a vector $v = (v_1, v_2, \ldots, v_n)$ is defined as $\sqrt{\sum_{i=1}^{n} {v_i}^2}$.*

Definition 3 *The vectors $v^{(1)}, \ldots, v^{(n)}$ are orthonormal if*

$$v^{(i)} \cdot v^{(j)} = \begin{cases} 1 & \text{if } i{=}j \\ 0 & \text{otherwise} \end{cases}$$

where the inner product $v \cdot w$ is defined as $\sum v_i \cdot w_i$.

For example, the transition matrix P of the d-regular undirected graph G has an uniform stationary distribution, because P is doubly stochastic. Therefore, P has an uniform eigenvector $(\frac{1}{n}, \ldots, \frac{1}{n})$ with the corresponding eigenvalue 1. Also, because the scalar producted vector to the eigenvector is still an eigenvector, $(\frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}})$ is an eigenvector which has 1 as the L_2-norm. The following theorem is important in analyzing our algorithm.

Theorem 4 *If the transition matrix P is a real and symmetric matrix, there exist eigenvectors $v^{(1)}, \ldots, v^{(n)}$ which form orthonormal basis with corresponding eigenvalues $1 = \lambda_1 \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$.*

Let's assume P has eigenvectors $v^{(1)}, \ldots, v^{(n)}$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$. Then we can observe the following facts.

Fact 5

- *αP has eigenvectors $v^{(1)}, \ldots, v^{(n)}$ with corresponding eigenvalues $\alpha\lambda_1, \ldots, \alpha\lambda_n$.*
- *$P + I$ has eigenvectors $v^{(1)}, \ldots, v^{(n)}$ with corresponding eigenvalues $\lambda_1 + 1, \ldots, \lambda_n + 1$.*
- *P^k has eigenvectors $v^{(1)}, \ldots, v^{(n)}$ with corresponding eigenvalues ${\lambda_1}^k, \ldots, {\lambda_n}^k$.*

Also, if $v^{(1)}, \ldots, v^{(n)}$ form orthonormal basis, any vector w can be expressed the linear combination of $v^{(i)}$'s ($w = \sum \alpha_i v^{(i)}$) and the L_2-norm of w is $\sqrt{\sum {\alpha_i}^2}$.

2 Mixing Time and Eigenvalues

Under the observation of the previous section, we can prove the following main theorem which tells about the mixing time of random walks.

Theorem 6 *If P is a transition matrix of an undirected, non-bipartitie, d-regular and connected graph, and π_0,$\bar{\pi}$ are the starting distribution and the stationary distribution respectively, then*

$$\| \pi_0 P^t - \bar{\pi} \| \leq |\lambda_2|^t$$

Proof From the theorem of the previous section, P has eigenvectors $v^{(1)}, \ldots, v^{(n)}$ which form orthonormal basis with corresponding eigenvalues $1 = \lambda_1 \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$. Hence, π_0 can be expressed as $\sum_{i=1}^{n} \alpha_i v^{(i)}$, and $\pi_0 P^t = \sum_{i=1}^{n} \alpha_i v^{(i)} P^t = \sum_{i=1}^{n} \alpha_i {\lambda_i}^t v^{(i)}$. Also,

$$\begin{aligned}
\| \pi_0 P^t - \alpha_1 v^{(1)} \| &= \| \sum_{i=2}^{n} \alpha_i {\lambda_i}^t v^{(i)} \| \\
&= \sqrt{\sum_{i=2}^{n} {\alpha_i}^2 {\lambda_i}^{2t}} \\
&\leq |\lambda_2|^t \cdot \sqrt{\sum_{i=2}^{n} {\alpha_i}^2} \\
&\leq |\lambda_2|^t \cdot \| \pi_0 \| \\
&\leq |\lambda_2|^t
\end{aligned}$$

The last inequality holds because the L_1-norm of π_0 is 1 and its L_2-norm is less than its L_1-norm. We can check $\alpha_1 v^{(1)} = \bar{\pi}$ by setting $\pi_0 = \bar{\pi}$ in the above result and letting t go to ∞. Therefore, our result follows. ■

3 Reducing Randomness Requirements

Suppose probabilistic polynomial algorithm A outputs a correct value of a function f with high probability. Let f be a binary function from $\{0,1\}^n$ to $\{0,1\}$ and assume A tosses r coins such that $\forall x, \Pr[A(x) \neq f(x)] \leq \frac{1}{100}$. For getting a better error-ratio (up to 2^{-k}), our first algorithm goes like this.

1. Repeat $O(k)$ times
   1.1. Pick a random $s = (s_1, \ldots, s_r) \in \{0,1\}^r$.
   1.2. Run $A(x)$ with coins $s_1, \ldots, s_r$.
2. Output the majority answer of the step 1.

This algorithm is just running $O(k)$ copies of A and getting the majority answer. We can analyze using the Chernoff bounds why this gives 2^{-k} as an error-ratio as we did in the first problem of the first homework. As you check easily, our first algorithm needs $O(kr)$ random bits. Our goal is construction an new algorithm which needs less random bits. This is possible if we use a random walk in a rapidly mixing graph. For our purpose, let's assume there exists an undirected, non-bipartitie, d-regular and connected graph of 2^r nodes which has a transition matrix P such that the absolute value of its second eigenvalue(λ_2) is less than $\frac{1}{10}$. From the theorem in the previous section, we can see that λ_2 guarantees the mixing time of the random walk in the graph. Our second algorithm goes like this.

1. Pick a random node $s = (s_1, \ldots, s_r) \in \{0,1\}^r$.
2. Repeat $7k$ times
   2.1. Let a new s be a random neighbor of an old s.
   2.2. Run $A(x)$ with coins $s_1, \ldots, s_r$.
3. Output the majority answer of the step 2.

We can easily check that our second algorithm uses $r + 7k \cdot \lceil \log d \rceil = r + O(k)$ random bits.(Assume d is constant.) Obviously, it is better than $O(kr)$ random bits in our first algorithm. Now define some notions for analyzing our second algorithm.(Our analysis is for knowing why our second algorithm gives 2^{-k} as an error-ratio.)

Definition 1
- *Let B be $\{s | A(x) \neq f(x)$ if A runs with coins $s\}$.*
- *Let N be a diagonal matrix such that $N_{ii} = 1$ if $i \in B$.*
- *Let M be a diagonal matrix such that $M_{ii} = 1$ if $i \notin B$.*

We can see that $|B| \leq \frac{1}{100}$. Call s 'bad' if $s \in B$, and 'good' otherwise. Then, if ρ is a probability distribution, $|\rho N| = \Pr[s \text{ is bad}]$ and $|\rho M| = \Pr[s \text{ is good}]$. Let S be the sequence of "good/bad"("correct/incorrect") of length $7k$, and define Q_i as follows,

$$Q_i = \begin{cases} M & \text{if } S_i \text{ is 'correct'.} \\ N & \text{if } S_i \text{ is 'incorrect'.} \end{cases}$$

Then, $\Pr[S] = |\rho P Q_1 \dots P Q_{7k}|$ holds. (This is not a trivial fact.) Using these notations, we will show why our second algorithm gives a lower error-ratio (2^{-k}) in the next lecture.