6.895 Randomness and Computation April 19, 2006

Lecture 17

Lecturer: Ronitt Rubinfeld *Scribe: Pouya Kheradpour*

1 Outline

In this lecture we discuss the use of pairwise independent random numbers in algorithms. We will first show an algorithm that only requires pairwise independence of the random bits used. Next, we will show how to generate these pairwise random bits using very few truly random bits allowing us to apply enumeration technique for derandomization. We will also show how to efficiently generate pairwise random *numbers* (as opposed to bits). Finally, we will see how to reduce the error rate of randomized algorithms with one-sided error using pairwise independent numbers.

2 Definitions

We begin with formal definitions of independence. Consider the random variables $x_1 \dots x_n \in T^n$ where $\forall b \in T, \forall i, \Pr[x_i = b] = \frac{1}{|T|}$. We say that $x_1 \dots x_n$ are

- *independent* when $\forall b_1 \dots b_n \in T^n, \Pr[x_1 \dots x_n = b_1 \dots b_n] = \frac{1}{|T|^n}$.
- *pairwise independent* when $\forall b_1 b_2 \in T^2, \forall i \neq j, \Pr[x_i x_j = b_1 b_2] = \frac{1}{|T|^2}$.
- *k-wise independent* when $\forall b_1 \dots b_k \in T^n, \forall$ distinct $i_1 \dots i_k, \Pr[x_{i_1} \dots x_{i_k} = b_1 \dots b_k] = \frac{1}{|T|^k}$.

Notice that pairwise independence is equivalent to 2-wise independence.

3 Large Cut using pairwise independence

Here we present and analyze a randomized algorithm for finding a large cut of a graph requiring only pairwise independent bits.

```
Algorithm LargeCut($G$):
    initialize $V_1, V_2 \leftarrow \{\}, \{\}$
    for every node $i = 1, \dots, n$
        flip coin $r_i$
        if $r_i =$ heads
            $V_1 \leftarrow V_1 \cup \{i\}$
        else
            $V_2 \leftarrow V_2 \cup \{i\}$
```

Now, to show that V_1 and V_2 do define a large cut, consider the following:

$$\begin{aligned}
E[\#\text{ of edges in cut}] &= \sum_{(i,j)\in E} \Pr[v_i \neq v_j] && \text{(linearity of expectation)}\\
&= \sum_{(i,j)\in E} \Pr[v_i = 1 \cap v_j = 0] + \Pr[v_i = 0 \cap v_j = 1]\\
&= \sum_{(i,j)\in E} \Pr[v_i = 1]\Pr[v_j = 0] + \Pr[v_i = 0]\Pr[v_j = 1] && (\textit{pairwise}\text{ independence})\\
&= \sum_{(i,j)\in E} \frac{1}{2} = \frac{|E|}{2}
\end{aligned}$$

Thus, all we need for this algorithm is n coins that are pairwise independent (not truly independent).

4 Efficiently generating pairwise random bits

Now we provide an algorithm for producing N pairwise independent random bits from only $O(\log N)$ truly random bits. Take k truly random bits $b_1 \dots b_k$. $\forall S \subseteq [k]$ such that $S \neq \emptyset$ set $C_S = \oplus_{i \in S} b_i$. Each of the $2^k - 1$ C_S S are pairwise independent (proof left as exercise). Thus, to generate N pairwise random bits we only need $\log N$ truly random bits.

5 Derandomized Large Cut

Because we can generate N pairwise random bits using $\log N$ truly random bits, we can

1. Enumerate all N possibilities for truly random bits of length $\log N$.
2. Use each string of independent bits to generate a string of N pairwise independent bits.
3. Run it through the randomized Large Cut algorithm presented above.
4. Take the cut that is the largest. By the argument above it will have expected size of at least $\frac{|E|}{2}$.

6 Generating pairwise independent random numbers

Pairwise independent random numbers $r_1, \dots, r_q$ between 0 and some prime $q > N$ can be generated using the following algorithm:

```
Algorithm GeneratePI($N$):
    $q \leftarrow$ some prime $> N$
    choose $a, b \in_R \mathbb{Z}_q$
    for $i \leftarrow 1 \dots q$
        $r_i \leftarrow a \cdot i + b \pmod{q}$
```

Notice that this algorithm requires only $2 \log q$ truly random bits to generate a and b.

To see that these number are pairwise independent, consider any $x, w, y, z \in [q]$ where $x \neq w$. It follows that there is only one a, b such that $a \cdot w + b = y$ and $a \cdot x + b = z$ because in

$$\begin{pmatrix} w & 1 \\ x & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} y \\ z \end{pmatrix}$$

the matrix $\begin{pmatrix} w & 1 \\ x & 1 \end{pmatrix}$ has non-zero determinant $w - x$. Thus, $Pr_{a,b}[a \cdot w + b = y \cap a \cdot x + b = z] = \frac{1}{q^2}$ indicating pairwise independence.

7 Generalization of pairwise independence generation

Define $\mathcal{H} = \{h_i : [N] \to [M]\}$ to be a pairwise independent family of functions if

$$\forall x \neq y \in [N], \forall a, b \in [M], Pr_{h \in \mathcal{H}}[h(x) = a \cap h(y) = b] = \frac{1}{M^2}$$

Thus, we notice that our construction for pairwise independent bits was the special case of $M = 2$ and for pairwise independent numbers was $M = q$.

8 “Two-point sampling” with pairwise independent numbers

Consider an algorithm $\mathcal{A}$ with “one-sided error”, i.e.

$$x \in L \Rightarrow Pr_R[\mathcal{A}(x,R)=1] \geq \frac{1}{2}$$

and

$$x \notin L \Rightarrow Pr_R[\mathcal{A}(x,R)=0] = 1$$

It is that that if $\mathcal{A}(x,R)=1$ then R is a “witness to $X \in L$”.

To improve the error probability, we can do one of the following:

1. Repeat k times, use Chernoff bounds to obtain error $\leq 2^{-k}$ with $O(kr)$ random bits.
2. Use a random walk on rapid mixing graph with $r + O(k)$ bits.
3. Use pairwise random bits requiring the generation of only $O(r+k)$ bits.

In previous lectures we saw the first two methods. We will discuss the last method now. We compute $\mathcal{A}(x,r_i)$ for all $1 \leq i \leq t$. If any $\mathcal{A}(x,r_i)=1$, then we output 1. Otherwise, we output 0.

Consider the probability of misclassification. If $x \notin L$ then $\mathcal{A}(x,r_i)$ is always 0 and we do not misclassify.

If $x \in L$ then the probability of misclassification is equal to the probability that we never see a 1. Let $Y \equiv \sum_{i=1}^{t} \mathcal{A}(x,r_i)$. By linearity of expectation, we see that $E[Y] = \sum_{i=1}^{t} E[\mathcal{A}(x,r_i)] \geq \frac{t}{2}$. We cannot use a Chernoff bound to evaluate $\Pr[Y=0]$ because the r_is are not independent and the Markov inequality is too weak. However, we can apply the Chebyshev inequality.

First we must compute the variance of Y. Because the r_is are pairwise independent,

$$\sigma_Y^2 = \sum \sigma^2[\mathcal{A}(x,r_i)] = \sum \left(E[\mathcal{A}(x,r_i)^2] - E^2[\mathcal{A}(x,r_i)]\right) = \sum (p-p^2) \leq \sum \frac{1}{4} = \frac{t}{4}$$

Thus, $\sigma_Y \leq \frac{\sqrt{t}}{2}$. The Chebyshev inequality states that

$$Pr[|x - E[x]| \geq s\sigma] \leq \frac{1}{s^2}$$

Thus, the probability that we encounter no $\mathcal{A}(x,r_i)=1$ when $x \in L$ is

$$Pr[Y=0] \leq Pr[|Y - \frac{t}{2}| > \frac{t}{2}] \leq \frac{1}{\sqrt{t}^2} = \frac{1}{t}$$

Thus, we obtain our desired error rate of 2^{-k} for $t = 2^k$ which requires $O(\log t) = O(k)$ truly random bits.

The name “Two-point sampling” comes from the fact that we only need random numbers a and b. Their size is bounded by q where $q \geq 2^r$ and $q \geq t$ (the numbers have to be big enough and we must be able to generate enough of them). Thus $q = \Omega(r + \log t) = \Omega(r+k)$.