Lecture 11

k-wise independence and its applications

Course: Algorithms for Big Data

Instructor: Hossein Jowhari

Department of Computer Science and Statistics
Faculty of Mathematics
K. N. Toosi University of Technology

Spring 2021

Independence

Let $X_1, \ldots, X_n$ be discrete random variables. We say $X_1, \ldots, X_n$ are (mutually) independent if for all values $\alpha_1, \ldots, \alpha_n$ we have

$$Pr(X_1 = \alpha_1, \ldots, X_n = \alpha_n) = \prod_{i=1}^{n} Pr(X_i = \alpha_i)$$

Limited Independence

k-wise independence: Let $X_1, \ldots, X_n$ be discrete random variables. We say $X_1, \ldots, X_n$ are k-wise independent if for every subset $S = \{s_1, \ldots, s_\ell\} \subseteq \{1, \ldots, n\}$ of cardinality at most k and all values $\alpha_1, \ldots, \alpha_\ell$ we have

$$Pr(X_{s_1} = \alpha_1, \ldots, X_{s_\ell} = \alpha_\ell) = \prod_{i=1}^{\ell} Pr(X_{s_i} = \alpha_i)$$

Special Case: Let $X_1, \ldots, X_n$ be $\{0,1\}$-valued random variables where for each i we have $Pr(X_i = 0) = Pr(X_i = 1)$. We say $X_1, \ldots, X_n$ are k-wise independent if for every subset $S = \{s_1, \ldots, s_\ell\} \subseteq \{1, \ldots, n\}$ of cardinality at most k and all values $\alpha_1, \ldots, \alpha_\ell$ we have

$$Pr(X_{s_1} = \alpha_1, \ldots, X_{s_\ell} = \alpha_\ell) = (\frac{1}{2})^\ell$$

Constructing k-wise independent $\{0,1\}$-valued random variables using little randomness

First Idea: Assume n is even. We let $X_1, \ldots, X_n$ be random (cyclic) shift of $\underbrace{1,0,1,0,1,0,\ldots,1,0}_{n}$. We have

$Pr(X_i = 0) = Pr(X_i = 1) = \frac{1}{2}$

Question: How much randomness is used in this construction?

Answer: $\log n$ bits

But X_i and X_j are not independent. ☹

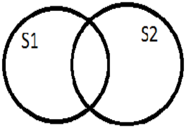
$$Pr(X_1 = 0, X_2 = 0) = 0 \neq \frac{1}{4}$$

Second Idea: [Pair-wise Independent Random Bits] Let $Y_1, \ldots, Y_m$ be mutually independent random bits. We construct $n = 2^m - 1$ random bits from $Y_1, \ldots, Y_m$. For each non-empty subset $S \subseteq [m] = \{1, \ldots, m\}$ we let

$$X_S = \sum_{r \in S} Y_r \mod 2$$

Claim: The random bits $\{X_S\}_{S \subseteq [m]}$ are pair-wise independent.

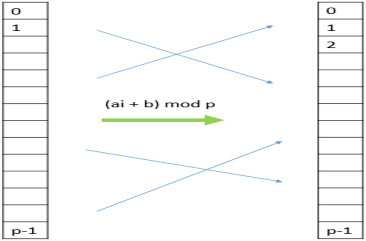
Proof: Exercise.



Conclusion: We can generate n pairwise random bits from $\log n + 1$ mutually independent random bits.

Third Idea: [Pair-wise Independent Random Numbers] Let p be a prime where $p \geq n$. We choose the random numbers a and b from $\mathbb{Z}_p$ independently. We let

$$X_i = (ai + b) \mod p$$



- $\forall i \in [n], \alpha \in \mathbb{Z}_p, \quad Pr(X_i = \alpha) = \frac{1}{p}$
- $\forall i, j \in [n], \alpha_1, \alpha_2 \in \mathbb{Z}_p, \quad Pr(X_i = \alpha_1, X_j = \alpha_2) = (\frac{1}{p})^2$

Fourth Idea: [Third Idea Generalized] Consider the field $\mathbb{F}_p$ where p is large enough. Let $Y_0, Y_1, \ldots, Y_\ell$ be mutually independent samples from $\mathbb{F}_p$. For $a \in \mathbb{F}_p$, we define

$$X_a = \sum_{i=0}^{\ell} Y_i a^i = Y_0 + Y_1 a + Y_2 a^2 + \ldots + Y_\ell a^\ell$$

Note all computations are done in the field $\mathbb{F}_p$

Lemma: The random variables $\{X_a\}_{a \in \mathbb{F}_p}$ are $(\ell + 1)$-wise independent.

Claim 1: For $b \in \mathbb{F}_p$, we have $Pr(X_a = b) = \frac{1}{p}$.

Proof: Proof by induction on ℓ. The case $\ell = 0$ is trivial.

$Pr(X_a = b) = Pr(\sum_{i=0}^{\ell} Y_i a^i = b)$

$= \sum_{c \in \mathbb{F}_p} Pr(\sum_{i=0}^{\ell} Y_i a^i = b \mid \sum_{i=0}^{\ell-1} Y_i a^i = c) Pr(\sum_{i=0}^{\ell-1} Y_i a^i = c)$

$= \sum_{c \in \mathbb{F}_p} Pr(Y_\ell a^\ell = b - c)\frac{1}{p}$ (By induction hypothesis)

$= \frac{1}{p} \sum_{c \in \mathbb{F}_p} Pr(Y_\ell a^\ell = b - c)$

$= \frac{1}{p} \sum_{c \in \mathbb{F}_p} \frac{1}{p} = \frac{1}{p}$

Claim 2: For all $a_0, \ldots, a_\ell \in \mathbb{F}_p$, we have

$$Pr(X_{a_0} = b_0, \ldots, X_{a_\ell} = b_\ell) = \frac{1}{p^{\ell+1}}$$

Proof Sketch: Note that when $Y_0, \ldots, Y_\ell$ are fixed, $\sum_{i=0}^{\ell} Y_i a^i$ is polynomial of degree at most ℓ and X_a is the evaluation of this polynomial at point $a \in \mathbb{F}_p$.

Fact A: A polynomial of degree d is (uniquely) determined by its evaluation at $d+1$ points.

Fact B: There are $p^{\ell+1}$ polynomial of degree at most ℓ over $\mathbb{F}_p$.

$$\text{Facts A and B} \;\Rightarrow\; Pr(X_{a_0} = b_0, \ldots, X_{a_\ell} = b_\ell) = \frac{1}{p^{\ell+1}}$$

Vandermonde Matrix

$$\begin{pmatrix} 1 & a_1 & a_1^2 & \cdots & a_1^\ell \\ 1 & a_2 & a_2^2 & \cdots & a_2^\ell \\ 1 & a_3 & a_3^2 & \cdots & a_3^\ell \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_p & a_p^2 & \cdots & a_p^\ell \end{pmatrix} \begin{pmatrix} Y_0 \\ Y_1 \\ Y_2 \\ \vdots \\ Y_l \end{pmatrix} = \begin{pmatrix} X_{a_1} \\ X_{a_2} \\ X_{a_3} \\ \vdots \\ X_{a_p} \end{pmatrix}$$

- Vandermonde matrix is full rank (has rank $\ell + 1$)
- (Lemma) let $M \in \mathbb{F}_p^{n\times\ell}$ be a full rank matrix and $Y = Y_0, \ldots, Y_\ell$ be independent samples from $\mathbb{F}_p$ then MY is $(\ell + 1)$-wise independent.

We can generate k-wise independent $(0, 1)$-valued random variables $X_1, \ldots, X_n$ from $O(k)$ random numbers from $\mathbb{F}_p$ where $p \geq n$. Therefore to able to generate every X_i we need to keep at most $O(k \log n)$ bits.

In particular, we can generate 4-wise independent $(-1, +1)$-valued random variables $\sigma_1, \ldots, \sigma_n$ by keeping $O(\log n)$ random bits.

Applications: Pairwise Independence

Problem: Estimating the number of distinct elements in the stream $A = a_1, \ldots, a_m$

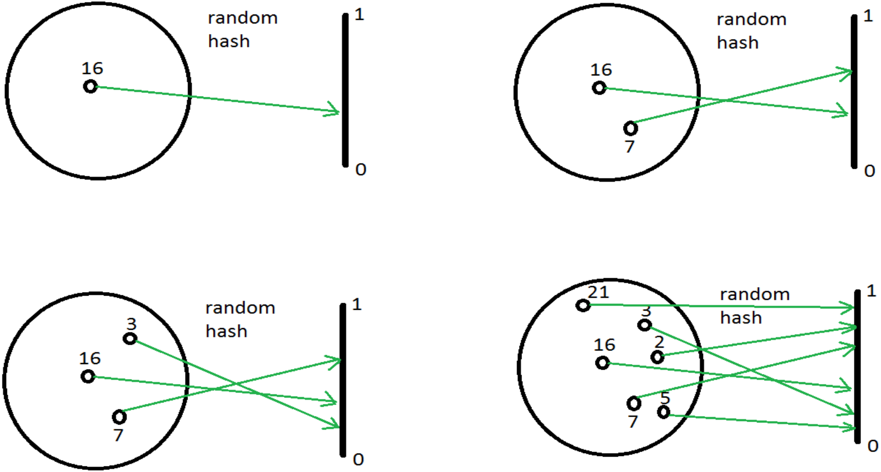
$$a_i \in \{1, \ldots, n\}$$

Example: $A = 2, 3, 2, 1, 2, 9, 8, 2, 5, 2, 2, 4, 6, 2, 2, 5, 2$

$F_0 = \text{number of distinct elements } = 8$

Idea: Hash the elements $[n] = \{1, \ldots, n\}$ (randomly and independently) to the continuous interval $(0, 1)$. Let $h(x)$ be the hash of $x \in [n]$. Output $\frac{1}{\min h(x)}$

Justification: If F_0 is high then there will an element x where its hash $h(x)$ is close to zero.


random
hash
16
1
0
random
hash
16
7
1
0
random
hash
3
16
7
1
0
random
hash
21
3
16
2
5
7
1
0

AMS Idea:

- Choose a pairwise independent hash function h from $[n] = \{1, \ldots, n\}$ to $\{0, 1, \ldots, p - 1\}$ where p is a prime greater than or equal to n. Let $h(x)$ be the hash of $x \in [n]$.
- Let $zeros(x)$ be the number of trailing zeros in the bit representation of number x. For example $zeros(000011) = 4$ and $zeros(100001) = 0$.
- Given the stream A, compute $z = \max_{a_i \in A} zeros(h(a_i))$
- Output 2^z

Space Complexity of the AMS Idea: To store the hash function h, we only need to choose and keep two random numbers a and p from $\mathbb{F}_p$. Since $p = O(n)$, this takes only $O(\log n)$ bits. Also the value z is easily computable by having access to the function h and storing at most $O(\log n)$ bits. Therefore the algorithm requires only $O(\log n)$ bits.

Proposition 2.3 *For every $c > 2$ there exists an algorithm that, given a sequence A of members of N, computes a number Y using $O(\log n)$ memory bits, such that the probability that the ratio between Y and F_0 is not between $1/c$ and c is at most $2/c$.*

$$Pr(\frac{1}{c} \leq \frac{Y}{F_0} \leq c) \geq 1 - \frac{2}{c}$$