Pairwise Independent and Hashing
## Pairwise Independence

### Definition

1. A set of events $E_1, E_2, \ldots, E_n$ is $k$-wise independent if for any subset $I \subseteq [1, n]$ with $|I| \leq k$,

   $$\Pr\left(\bigcap_{i \in I} E_i\right) = \prod_{i \in I} \Pr(E_i).$$

2. A set of random variables $X_1, X_2, \ldots, X_n$ is $k$-wise independent if for any subset $I \subseteq [1, n]$ with $|I| \leq k$, and any values $x_i, i \in I$,

   $$\Pr\left(\bigcap_{i \in I} X_i = x_i\right) = \prod_{i \in I} \Pr(X_i = x_i).$$

If true for $k = n$ the random variables are *mutually independent*. 
Pairwise Independent

Definition

The random variables $X_1, X_2, \ldots, X_n$ are said to be pairwise independent if they are 2-wise independent. That is, for any pair $i, j$ and any values $a, b$,

$$\Pr((X_i = a) \cap (X_j = b)) = \Pr(X_i = a) \cdot \Pr(X_j = b).$$

Application: We can construct $m = 2^b - 1$ uniform pairwise independent 0-1 random variable from $b$ independent, uniform random bits, $X_1, \ldots, X_b$.

$m = 2^b - 1$ uniform pairwise independent 0-1 random variable in a sample space with $2 \cdot 2^b$ simple events.
Construction of Pairwise Independent Bits

We are given \( b \) independent, uniform random bits, \( X_1, \ldots, X_b \).

Let \( S_1, \ldots, S_{2^b-1} \) be an arbitrary order of all the non-empty subsets of \( \{1, 2, \ldots, b\} \).

Let \( \oplus \) be the exclusive-or operation. Define \( m = 2^b - 1 \) random variables

\[
Y_j = \bigoplus_{i \in S_j} X_i = \sum_{i \in S_j} X_i \mod 2
\]

- \( \Pr(Y_i = 1) = \Pr(Y_i = 0) = 1/2 \). Let \( z \in S_i \). Fix the bits in \( S_i - \{z\} \). The value of \( Y_i \) is determined by the value of \( z \).

- Pairwise independence: For any \( c, d \in \{0, 1\} \)

\[
\Pr((Y_k = c) \cap (Y_\ell = d)) = \Pr(Y_\ell = d \mid Y_k = c) \cdot \Pr(Y_k = c) = 1/4.
\]

Since the value of \( Y_\ell \) is determined by \( z \in S_\ell \setminus S_k \)

Thus, \( Y_1, \ldots Y_{2^b-1} \) are pairwise independent, uniform \( \{0, 1\} \) random variables.
The Expectation Argument: Large Cut-Set in a Graph.

**Theorem**

Given any graph $G = (V, E)$ with $n$ vertices and $m$ edges, there is a partition of $V$ into two disjoint sets $A$ and $B$ such that at least $m/2$ edges connect a vertex in $A$ to a vertex in $B$.

Let $Y_1, \ldots, Y_n$ pairwise independent uniform $\{0, 1\}$ random variables, generated from $\log_2 n + 1$ independent random bits.

Place such that vertex $i$ is in set $A$ if $Y_i = 0$ else vertex $i$ is placed in set $B$.

Let $Z_e = 1$ if edge $e$ crosses the cut, and $Z_e = 0$ otherwise.

Let $e = \{i, j\}$, then $\Pr(Z_e = 1) = \Pr(Y_i \neq Y_j) = \frac{1}{2}$,

$$E[Z] = E \left[ \sum_{i=1}^{m} Z_i \right] = \sum_{i=1}^{m} E[Z_i],$$

the sample space has an assignment with a cut $\geq m/2$.

The sample space has only $2^n$ simple event, algorithm can try all simple events to find a good assignment.
Deviation Bound

You cannot use Chernoff bound but you can use Chebyshev bound.

**Theorem**

Let \( X = \sum_{i=1}^{n} X_i \), where the \( X_i \) are pairwise independent random variables. Then

\[
\text{Var}[X] = \sum_{i=1}^{n} \text{Var}[X_i].
\]

**Proof:** \( \text{Var}[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} \text{Var}[X_i] + 2 \sum_{i<j} \text{Cov}(X_i, X_j) \).

For Pairwise independent \( X_i, X_2, \ldots, X_n \),

\[
\text{Cov}(X_i, X_j) = \mathbb{E}[(X_i - \mathbb{E}[X_i])(X_j - \mathbb{E}[X_j])] = \mathbb{E}[X_i X_j] - \mathbb{E}[X_i] \mathbb{E}[X_j] = 0.
\]

**Corollary**

Let \( X = \sum_{i=1}^{n} X_i \), where the \( X_i \) are pairwise independent random variables. Then

\[
\Pr(|X - \mathbb{E}[X]| \geq a) \leq \frac{\text{Var}[X]}{a^2} = \frac{\sum_{i=1}^{n} \text{Var}[X_i]}{a^2}.
\]
Perfect Hashing

We want to store $n$ records using minimus space and minimum retrieval (search) time.

We can store the $n$ records in a sorted order. Space $= O(n)$, retrieval time $= O(\log n)$

We can hash the $n$ keys to a table of size $O(n)$, with $O(1)$ expected retrieval time, and $O(\log n)$ expected maximum retrieval time. (We need a table of size $\Omega(n^{1+\epsilon})$ for expected maximum $1/\epsilon$.)

**Goal:** Store a static dictionary of $n$ items in a table of $O(n)$ space such that any search takes $O(1)$ time.

Static dictionary - any insert or delete operation requires rearranging the entire table.
Universal hash functions

Definition

Let $U$ be a universe with $|U| \geq n$ and $V = \{0, 1, \ldots, n - 1\}$. A family of hash functions $\mathcal{H}$ from $U$ to $V$ is said to be $k$-universal if, for any elements $x_1, x_2, \ldots, x_k$, when a hash function $h$ is chosen uniformly at random from $\mathcal{H}$,

$$\Pr(h(x_1) = h(x_2) = \ldots = h(x_k)) \leq \frac{1}{n^{k-1}}.$$ 

If $\Pr(h(x_1) = h(x_2) = \ldots = h(x_k)) = \frac{1}{n^{k-1}}$, then for any $x_1, x_2, \ldots, x_k$ the random variables $h(x_1), \ldots, h(x_k)$ are $k$-pairwise independent.
Example of 2-Universal Hash Functions

Universe $U = \{0, 1, 2, \ldots, m - 1\}$
Table keys $V = \{0, 1, 2, \ldots, n - 1\}$, with $m \geq n$.
A family of hash functions obtained by choosing a prime $p \geq m$,

$$h_{a,b}(x) = ((ax + b) \mod p) \mod n,$$

and taking the family

$$\mathcal{H} = \{h_{a,b} \mid 1 \leq a \leq p - 1, 0 \leq b \leq p\}.$$

Lemma

$\mathcal{H}$ is 2-universal.
Lemma

\( \mathcal{H} \) is 2-universal.

**Proof:** We first observe that for \( x_1, x_2 \in \{0, \ldots, p - 1\} \), \( x_1 \neq x_2 \),

\[
ax_1 + b \neq ax_2 + b \mod p.
\]

Thus, if \( h_{a,b}(x_1) = h_{a,b}(x_2) \) there is a pair \((s, r)\) such that,

1. \((ax_1 + b) \mod p = r\)
2. \((ax_2 + b) \mod p = s\)
3. \(s \neq r, s = (r \mod n)\)

For each \( r \) there are \( \leq \lceil \frac{p}{n} \rceil - 1 \) values \( s \neq r \) such that \( s = (r \mod n) \), and for each pair \((r, s)\) there is only one pair \((a, b)\) that satisfies the relation.

Over all the \( p(p - 1) \) choice of \((a, b)\), \( r \) gets \( p \) different values.

Thus, the probability of a collision is \( \leq \frac{p(\lceil \frac{p}{n} \rceil - 1)}{p(p-1)} \leq \frac{1}{n} \).
**Lemma**

If \( h \in \mathcal{H} \) is chosen uniformly at random from a 2-universal family of hash functions mapping the universe \( U \) to \([0, n - 1]\), then for any set \( S \subset U \) of size \( m \), with probability \( \geq 1/2 \) the number of collisions is bounded by \( m^2/n \).

**proof:**

Let \( s_1, s_2, \ldots, s_m \) be the \( m \) items of \( S \). Let \( X_{ij} \) be 1 if the \( h(s_i) = h(s_j) \) and 0 otherwise. Let \( X = \sum_{1 \leq i < j \leq n} X_{ij} \).

\[
E[X] = E \left[ \sum_{1 \leq i < j \leq n} X_{ij} \right] = \sum_{1 \leq i < j \leq m} E[X_{ij}] \leq \binom{m}{2} \frac{1}{n} < \frac{m^2}{2n},
\]

Markov's inequality yields

\[
\Pr(X \geq m^2/n) \leq \Pr(X \geq 2E[X]) \leq \frac{1}{2}.
\]
**Definition**

A hash function is perfect for a set $S$ if it maps $S$ with no collisions.

**Lemma**

If $h \in \mathcal{H}$ is chosen uniformly at random from a 2-universal family of hash functions mapping the universe $U$ to $[0, n - 1]$, then for any set $S \subset U$ of size $m$, such that $m^2 \leq n$ with probability $\geq 1/2$ the hash function is perfect.

\[ \Pr(X \geq 1) \leq \Pr(X \geq m^2/n) \leq \Pr(X \geq 2E[X]) \leq \frac{1}{2}. \]
Theorem

The two-level approach gives a perfect hashing scheme for $m$ items using $O(m)$ bins.

Level I: use a hash table with $n = m$. Let $X$ be the number of collisions,

$$\Pr(X \geq m^2/n) \leq \Pr(X \geq 2\mathbb{E}[X]) \leq \frac{1}{2}. $$

When $n = m$, there exists a choice of hash function from the 2-universal family that gives at most $m$ collisions.
Level II: Let \( c_i \) be the number of items in the \( i \)-th bin. There are \( \binom{c_i}{2} \) collisions between items in the \( i \)-th bin, thus

\[
\sum_{i=1}^{m} \binom{c_i}{2} \leq m.
\]

For each bin with \( c_i > 1 \) items, we find a second hash function that gives no collisions using space \( c_i^2 \). The total number of bins used is bounded above by

\[
m + \sum_{i=1}^{m} c_i^2 \leq m + 2 \sum_{i=1}^{m} \binom{c_i}{2} + \sum_{i=1}^{m} c_i \leq m + 2m + m = 4m.
\]

Hence the total number of bins used is only \( O(m) \).
Perfect Hashing

**Theorem**

*There is a storage method that can store $m$ keys in a table of size $O(m)$ with $O(1)$ search time.*