1. Searching in \( O(1) \) time (M. Ajtai, J. Komlôs & E. Szemerédi, 1985).

We continue our discussion on devising a data structure for a static input set such that searching for any element can be done in \( O(1) \) time. Two levels of hashing will be employed to store the input set.

Input: \( S = \{ k_1, k_2, ..., k_s \} \subseteq M \).

Let \( M = \{ 0, 1, ..., m-1 \} \) and \( N = \{ 0, 1, ..., n-1 \} \).

W.L.O.G., Let \( p = (m + 1) \) be a prime number. \( \mathbb{Z}_p = \{ 0, 1, ..., p-1 \} \).

For any \( 1 \leq k \leq m \), let \( h_k(x) = kx \mod p \mod n \).

Let \( V \subseteq M \) be any set where \( |V| = v \).

Let \( B_i(k, n, V) \) be the set of elements of \( V \) that are hashed into \( i \) by \( h_k \), for \( i \in N \).

\( B_i(k, n, V) = \{ x \in V: h_k(x) = i, \; i = 0, 1, ..., n-1 \} \).

Let \( |B_i(k, n, V)| = b_i(k, n, V) \).

**Lemma.** \( \sum_{k=1}^{m} \sum_{i=0}^{n-1} \binom{b_i(k, n, V)}{2} \leq \frac{mn^2}{n} \) for all \( V \subseteq M \) and \( n > v \).

**Corollary.** \( \exists k, \; s.t. \; \sum_{i=0}^{n-1} \binom{b_i(k, n, V)}{2} < \frac{v^2}{n} \).

**How do we store \( S \)?**

There will be two levels of hashing. In the first level use: \( n = s, V = S \)

Let \( h_k \) be a hash function that satisfies the following equation (1). The existence of such a function is ensured by the above Corollary. Assume that \( \binom{a}{b} = 0 \) when \( a < b \).

\[
\sum_{i=0}^{s-1} \binom{b_i(k, s, S)}{2} < \frac{s^2}{s} = s
\] (1)
In the second level do the following:

- For the bucket \( i \) \((0 \leq i \leq s - 1)\)
  
  Use a hash function \( h_{ki} \) with “n” = \( b_i(k, s, S)^2 \). In this case the hashing will be perfect (for an appropriate choice of \( h_{ki} \)).

Space for the hash functions = \((s + 1)\).
Space for the first level = \( s \).
Space for the second level = \( \sum_{i=0}^{s-1} b_i(k, s, S)^2 \).

From equation (1)

\[ \sum_{i=0}^{s-1} [(b_i(k, s, S))^2] < 2s + \sum_{i=0}^{s-1} (b_i(k, s, S)) = 3s. \]

\[ \rightarrow \text{Total memory used} = O(s). \]

Note.

Searching time = \( O(1) \).

We only have to do two hash function evaluations.

**How do we find good \( k \) values?**

We have to find \((s+1)\) hash functions such that for each function the above Corollary holds. If the set that is hashed is \( V \) with \(|V| = v\), then we can try each value of \( k \) and this trivial algorithm takes \( O(mn) \) time. This can be done in \( O(mv \log v) \) time as well.
Fact.

For at least $\frac{1}{2}$ of the $k$-values $\sum_{i=0}^{n-1} \binom{\binom{k}{i} n V}{2} < \frac{2v^2}{n}$.

- If we pick a random $k$, then $\text{Prob. } \left[ \sum_{i=0}^{n-1} \binom{\binom{k}{i} n V}{2} < \frac{2v^2}{n} \right] \geq \frac{1}{2}$.
- As a result, the time needed to find a good $k$ is $\tilde{O}(n \log v)$.

Therefore, the time needed to find all the $(s+1)$ hash functions is $\tilde{O}(s + \sum_{i=0}^{s-1} b_i (k, s, S)^2 \log s) = \tilde{O}(s \log s)$.

The probabilistic method:

is used to show the existence of objects that possess a given set of properties.

We use two basic facts:

1. If $X$ is a random variable with a mean $\mu$ then
   
   $X$ takes on a value that is $\geq \mu$ and $X$ takes on a value that is $\leq \mu$.

2. Let $U$ be a set of objects and let $P$ be a property.
   
   If $\text{Prob. } \left[ \text{a random object of } U \text{ has property } P \right] > 0$ then
   
   it implies that $U$ has at least one object with property $P$.

Example 1.

Let $G(V, E)$ be an undirected graph. Then

$\exists$ a partition of $V$ into $V_1$ and $V_2$, s.t. the number of edges from $V_1$ to $V_2$ is $\geq \frac{|E|}{2}$.

Proof.

For every node $u \in V$

Put it in $V_1$ with probability $= \frac{1}{2}$; 

Put it in $V_2$ with probability $= \frac{1}{2}$; 

For any edge $e \in E$

Probability that it goes from $V_1$ to $V_2 = \frac{1}{2}$.

$\Rightarrow$ The expected number of edges from $V_1$ to $V_2$ is $\geq \frac{|E|}{2}$.

Using (1), $\exists$ a partition for which the number of edges from $V_1$ to $V_2$ is $\geq \frac{|E|}{2}$. 

Example 2.
Input: \( F = C_1 \land C_2 \land C_3 \land \ldots \land C_m \), which is a CNF Boolean formula on \( n \) variables.

**Fact:** There exists an assignment that satisfies \( \geq \frac{m}{2} \) clauses.

**Proof.**
Let \( C_i \) be any clause with \( k \) literals. Give a random assignment to the variables.

\[
\text{Prob.} \ [C_i \text{ is not satisfied}] = 2^{-k}.
\]

\[\Rightarrow \text{Prob.} \ [C_i \text{ is satisfied}] = 1 - 2^{-k} \geq \frac{1}{2}.\]

\[\Rightarrow \text{Expected number of satisfied clauses} = \frac{m}{2}.\]

Using (1), \( \exists \) an assignment that satisfies \( \geq \frac{m}{2} \) clauses.

**Example 3.**
Let \( C_n \) be a complete graph on \( n \) nodes.

Let \( k \) and \( t \) be integers.

\( R(k, t) \) is the minimum value of \( n \) s.t. if the edges of \( C_n \) are colored with red and blue, then for each such coloring \( \exists \) either a red clique of size \( k \) or a blue clique of size \( t \). \( R(k, t) \) is known as the Ramsey number.

**Fact.**
If \( \binom{n}{k} 2^{1 - \left(\frac{k}{2}\right)} < 1 \) then \( R(k, k) > n. \)

**Proof.**
Color the edges randomly.

Let \( X \) be a subset of nodes with \( |X| = k \).

\[\text{Prob.} \ [X \text{ is unicolored}] = 2^{1 - \left(\frac{k}{2}\right)}.\]

\[\Rightarrow \text{Prob.} \ [\exists \text{ a subset } X \text{ of size } k \text{ that is unicolored}] \leq \binom{n}{k} 2^{1 - \left(\frac{k}{2}\right)}.\]

If \( \binom{n}{k} 2^{1 - \left(\frac{k}{2}\right)} < 1 \) then

\[\text{Prob.} \ [\text{No subset } X \text{ of size } k \text{ is unicolored}] > 0.\]

\[\Rightarrow \exists \ \text{A coloring under which no subset } X \text{ of size } k \text{ is unicolored.}\]

\[\Rightarrow R(k, k) > n.\]
What is the maximum value of \( n \) for which \( \binom{n}{k} 2^{1-\left(\frac{k}{2}\right)} < 1 \)?

\[
\binom{a}{b} \approx (ae/b)^b
\]

\[
\left(\frac{ne}{k}\right)^k 2^{1-\left(\frac{k}{2}\right)} < 1
\]

\[
n^k = \frac{2^{k(k-1)z}}{z} \left(\frac{k}{e}\right)^k
\]

\[
n^k \approx \left[ 2^{\frac{(k-1)z}{2}} \frac{k}{e}\right]^k
\]

\[
n = 2^{\frac{(k-1)z}{2}} \frac{k}{e} \text{ is a lower bound on } R(k,k).
\]