Hashing (1)
The general setup:

- $\mathcal{U}$ – a universe of possible “data items”
- $T[0 \ldots m - 1]$ – a table for storing data, *indices are called slots, buckets, or bins*
- $h: \mathcal{U} \rightarrow \{0, \ldots, m - 1\}$ – a “hash function” *maps data items to slots*
- A **collision** is a pair $(a, b)$ such that $a \neq b$ but $h(a) = h(b)$
Resolving collisions by chaining
Dictionary Operations:

- \textit{insert}(a): insert \(a\) in the linked list \(T[h(a)]\)
- \textit{search}(a): search for \(a\) in \(T[h(a)]\)
- \textit{delete}(a): search for and delete \(a\) in \(T[h(a)]\)

Running times:

- \textit{insert} – \(O(1)\)
- \textit{search}, \textit{delete} – \(O(n)\) (worst case)

Worst case occurs when all items hash to the same slot

Better: choose a \textit{random} hash function hopefully — no “pile ups”
Universal Hashing [Carter & Wegman, 1975]

- $\mathcal{K}$ – a finite, non-empty set of hash keys
- $\mathcal{H} = \{h_k\}_{k \in \mathcal{K}}$ – a family of hash functions $h_k : \mathcal{U} \to \{0, \ldots, m - 1\}$, indexed by $k \in \mathcal{K}$

**Def’n:** $\mathcal{H}$ is called universal if for all $a, b \in \mathcal{U}$ with $a \neq b$,

$$\left| \{k \in \mathcal{K} : h_k(a) = h_k(b)\} \right| \leq \frac{|\mathcal{K}|}{m}.$$

**Probabilistic interpretation:** if $R$ is a random variable, uniformly distributed over $\mathcal{K}$, then

$$\Pr[h_R(a) = h_R(b)] \leq \frac{1}{m}.$$
Using Universal Hash Functions

Assume distinct items $a_1, \ldots, a_n$ are stored in table

Let $\alpha := n/m =$ “load factor”

Assume $R$ is uniformly distributed over $\mathcal{K}$

For $i = 1, \ldots, n$, define

$$S_i := \# \text{ of items in slot } h_R(a_i)$$

That is, $S_i$ is the number of items in the slot occupied by $a_i$

The values $R, S_1, \ldots, S_n$ are random variables.

For each $i = 1, \ldots, n$, we wish to bound $\mathbb{E}[S_i]$. 
Claim: $E[S_i] \leq \alpha + 1$ for each $i = 1, \ldots, n$.

Proof: for $i, j = 1, \ldots, n$, define indicator variables

$$C_{ij} := \begin{cases} 1 & \text{if } h_R(a_i) = h_R(a_j) \\ 0 & \text{otherwise} \end{cases}$$

For all $i, j$:

$$E[C_{ij}] = \Pr[h_R(a_i) = h_R(a_j)] \leq \frac{1}{m} \quad \text{if } i \neq j$$

$$= 1 \quad \text{if } i = j$$

Write $S_i$ as sum of indicator variables: $S_i = \sum_{j=1}^{n} C_{ij}$

By linearity of expectation:

$$E[S_i] = \sum_{j=1}^{n} E[C_{ij}] = E[C_{ii}] + \sum_{j \neq i} E[C_{ij}]$$

$$\leq 1 + \frac{(n - 1)}{m}$$

$$\leq \alpha + 1 \quad \text{QED}$$
interpretation:

- for each $i$, the expected # of items in $a_i$’s slot (including $a_i$ itself) is $\leq \alpha + 1$
- the expected time to perform a single dictionary operation is $O(\alpha + 1)$
- by linearity of expectation, expected time to perform $k$ dictionary operations is $O(k(\alpha + 1))$

special case: $\alpha = O(1)$ (i.e., $n = O(m)$)

- expected time per operation is $O(1)$
Suppose hash table contains items $a_1, \ldots, a_n$, and that $R$ is uniform over $\mathcal{K}$

For $s = 0, \ldots, m - 1$, define

$$L_s := \# \text{ of } a_i \text{'s that hash to slot } s \text{ under } h_R$$

Set $M := \max\{L_s : s = 0, \ldots, m - 1\}$

We want to bound $E[M]$, assuming universal hashing

**Jensen says:** $E[M]^2 \leq E[M^2]$

**Observe:** $M^2 \leq V := \sum_{s=0}^{m-1} L_s^2$

**Claim:** $E[V] \leq n^2/m + n$
Proof of claim: Define indicator variables

\[ I_{i,s} := \begin{cases} 1 & \text{if } h_R(a_i) = s \\ 0 & \text{otherwise} \end{cases} \]

We have

\[
V = \sum_{s=0}^{m-1} L_s^2 = \sum_{s=0}^{m-1} \left( \sum_{i=1}^{n} I_{i,s} \right)^2
\]

\[
= \sum_s \left( \sum_i I_{i,s} \right) \left( \sum_j I_{j,s} \right)
\]

\[
= \sum_{i,j} \sum_s I_{i,s}I_{j,s} = \sum_{i,j} C_{ij}
\]
So we have

\[ V = \sum_{i,j} C_{ij} \]

and by linearity of expectation, we have

\[ E[V] = \sum_{i,j} E[C_{ij}] = \sum_i E[C_{ii}] + \sum_{i \neq j} E[C_{ij}] \leq n + n(n-1)/m \leq n^2/m + n \]

QED
Corollary: \( E[M] \leq \sqrt{n^2/m} + n \)

Special case: \( \alpha = O(1) \) (i.e., \( n = O(m) \))

\[
E[M] = O(\sqrt{m})
\]

- This bound is tight
- Counter-intuitive: it may be the case that \( E[L_s] = O(1) \) for each slot \( s \)

Again: expected value of max may be much larger than max of expected values