Universal Hash Functions:

Theorem:

For key $k$, $E[n_{h(k)}] \leq \begin{cases} \alpha + 1 & \text{if } k \in T \\ \alpha & \text{otherwise} \end{cases}$

Notations: $n_i = \text{number of items in bucket } i \text{ of hash table } T$
$n = \text{total number of items in } T$
$m = \text{size of } T$

$$\frac{n}{m} = \text{load factor of } T = \alpha$$

Proof:

Let $X_{kl} = \begin{cases} 1 & \text{if } h(k) = h(l) \\ 0 & \text{otherwise} \end{cases}$ where $h$ is the hash function used

Let $Y_k = \text{number of keys that hash to same slot as } k \text{ (excluding } k)$

Then $Y_k = \sum_{l \neq k} X_{kl}$

$$E[Y_k] = E[\sum_{l \neq k} X_{kl}] = \sum_{l \neq k} E[X_{kl}] = \sum_{l \neq k} \frac{1}{m}$$

If $k \notin T$ then $n_{h(k)} = Y_k$ and $|\{l \in T | l \neq k\}| = n \rightarrow E[Y_k] = \frac{n}{m}$

On the other hand, if $k \in T$ then $n_{h(k)} = Y_k + 1$ and $|\{l \in T | l \neq k\}| = n - 1 \rightarrow E[Y_k] = \frac{n-1}{m}$

See example on the next page.
Example of universal set of hash functions:

\[ h_{a,b}(k) = ((ak + b) \mod p) \mod m \]

“p” is a prime number greater than any key, “p” is fixed

Choose \( \{a \in \{1,2, \ldots, p - 1\} \} \) and \( \{b \in \{0,1, \ldots, p - 1\} \} \) to select hash function of set

Cuckoo Hashing:

More detailed info at: https://en.wikipedia.org/wiki/Cuckoo_hashing

Notes:

Runtime of cuckoo hashing, find takes O(1), delete takes O(1), insert takes O(1) but amortized expected.

How it works: Use 2 hash functions \( h_1 \) and \( h_2 \), key \( k \) will either be stored at \( h_1(k) \) or \( h_2(k) \), each slot in table \( T \) contains one key.

Pseudo Code for cuckoo hashing:

```
Insert(k)
    pos = h_1(k)
    repeat n times
        { 
            b = T[pos]
            T[pos] = k
            If b == NULL, return ← cuckoo k has kicked out b
            If pos = h(b) then pos = h_2(b)
            Else pos = h_1(b)
            k = b ← b becomes new cuckoo
        }
    Rehash() and Insert(k)
```
Cuckoo Graph for n items:
Vertices = \{0,1,\ldots,m-1\}
Edges = \{(h_1(k), h_2(k))| k \in T\}

Assume \( h_1(k) \) and \( h_2(k) \) are uniform and independent random edges, an insertion will succeed if there is no cycle in the cuckoo graph.

Lemma: For \( c > 1 \) and \( m \geq 2cn \)

\[
\Pr[\text{cuckoo graph has path of length } l \text{ from } i \text{ to } j] \leq \frac{1}{mc^l}
\]

“c” is a constant less or equal to 3

Proof by induction on \( l \):

Base: \( l = 1 \) Edge(i,j) exists in graph with \( \Pr \leq n \frac{2}{m^2} = n \frac{2}{2cnm} = \frac{1}{c^1m} \)

Note: \( \Pr[\text{a single random edge being}(i,j)] = \frac{2}{m^2} \)

When \( l > 1 \) shortest path from \( i \) to \( j \) has length \( l \) only if there exists \( p \) and
\[1, \text{there exists a shortest path from } i \text{ to } p \text{ of length } l - 1 \text{ with } pr \leq \frac{1}{mc^l} \]
\[2, \text{there exists the edge } (p, j) \text{ with } pr \leq \frac{1}{mc} \]

Together, \( \leq \frac{1}{m^2c^l} \) sum over possible \( p \rightarrow \leq \frac{1}{mc^l} \)

Probability that \( k \) and \( k' \) hash to the same path/bucket is probability of a path from \( h_1(k) \) or \( h_2(k) \) to \( h_1(k') \) or \( h_2(k') \) \( \leq 4 \sum_{l=1}^{\infty} \frac{1}{mc^l} = \frac{4}{m} \frac{1}{c - 1} = O\left(\frac{1}{m}\right) \)

Rehash means choosing new hash functions and rehashing all keys, \( \Pr[\text{rehash}] \)
\( \leq \text{Prob hashing creates cuckoo graph with a cycle} \)
\( \leq \sum_{i=1}^{m} \Pr[\text{cycle involving } i] \leq \sum_{i=1}^{m} \sum_{l=1}^{\infty} \frac{1}{mc^l} \leq \frac{1}{2} \text{ for } c \geq 3 \)