Name : Seetharaman Krishnamoorthy

NetID : SK6599

UNIV ID : N13274420

CS6033 - Assignment 3

1) * Let the list of SSN obtained from Bank be 'N'.

* Let the list of SSN in company database be 'n'

* To find the list of customers affected, we should first store all the SSN in list 'N' into a hash table.

* Then we can search for each SSN from company database to see if it is present in the hash table.

Algorithm

```
INSERT-SSN-AND-SEARCH(T, N, n)
    for entry in N
        HASH-INSERT(T, entry)

    for item in n
      r = HASH-SEARCH(T, item)
      if r == NIL
          print item not affected
      else
          print item affected
```

* Here the HASH-INSERT and HASH-SEARCH takes $\Theta(1)$ time.

* So the Total running time is

$$\text{Running time} = \Theta(n) + \Theta(n)$$
$$= 2\,\Theta(n)$$

$$\Rightarrow \text{Total Running time} = \Theta(n)$$

2) * Let 'n' be the total words in Book B.

* we need to calculate the number of unique words in Book B.

* To find unique words, we will search for a word in hash table. If the slot is empty, we Insert the word and increase the count.

* If the slot is not empty, we assume that the word is duplicate and ignore it.

Algorithm

```
COUNT-UNIQUE-WORDS (B)
    S = 0
    for word in B
        slot = HASH-~~INSERT~~SEARCH (T, word)
        if slot == NIL
            HASH-INSERT (T, word)
            S = S+1
        else
            continue
    return S        // S contains total no. of unique
                    // words.
```

Running time = $\Theta(n)$

3) * we have 4 hash functions to analyze.

* The square functions are a bad choice because it would probably not distribute the keys and map keys to restricted locations.

* This is the same case with 11 i^2 mod 100 function too.

* So we are left with two choices. The 12 i mod 100 distributes the key but it maps the key only to even locations which might cause collisions.

* Ans : i^3 mod 100 since it spreads out and minimises the collision.

4) - To construct a perfect hash table, we choose a hash function from universal family of hash functions and repeat the step until we get a failure rate of 1 out of 1 million.

- $E(\text{collision}) < 1/2$ for each time

So we keep on choosing it until we get the perfect sub-table.

So, $E(\text{collision}) < 1/2 \cdot 1/2 \cdot 1/2 \cdots 1/2$ (19 times)

which gives $E(\text{collision}) < \frac{1}{1000000}$

⇒ Ans: we should attempt 19 times to construct a perfect sub-table with probability greater than 99.9999 %.

5) a) Total size is 'm'

No. of items in A is 'n'

The probability of CHECK(K) returns wrong answer is no. of items in A divided by total no. of elements in A.

$$Pr[h(K) = h(K')] \leq \frac{1}{m}$$

$\Rightarrow$ Probability of CHECK(K) returns wrong answer is n/m

b) - we have a second hash function $h'(x)$

- we cant do perfect hashing because there should not be extra space.

- The solution here would be to use double - hashing.

INSERT(K):

```
x = h(K)
y = h'(x)
set A[y] = 1
```

CHECK(K):

```
x = h(K)
y = h'(x)
if A[y] == 1
    return true
else
    return false
```

- Now as we apply double-hashing the probability is reduced by half.

- So the probability that CHECK(k) returns wrong answer is.

$$P_r = \frac{n/2}{m}$$

$$\Rightarrow \quad E(\text{collision}) \leq \frac{n}{2m}$$

6) $H = \{ h_{a,b}(x) = (ax + b \bmod n) \bmod m$ where

$a \in \{1, \dots n-1\}$ and

$b \in \{0, \dots n-1\}$

where $m = 2n$

- This function is not universal because $n > m$.

Example

$m = 100$, $n = 50$

$a = 20$ $\quad k_1 = 51$

$b = 30$ $\quad k_2 = 61$

$(ak_1 + b \bmod n) \bmod m = (ak_2 + b \bmod n) \bmod m$

$(1050 \bmod 50) \bmod 100 = (1250 \bmod 50) \bmod 100$

$0 = 0$

-> Both Hash function collide.

7) Consider the case of perfect hashing. Design the level 2 table efficiently so that the table size is less than $(n_j)^2$.