Fundamental Algorithms, Home work - 5

Tulsi Jain

1. First Problem

Algorithm 1 Modified Hash Insert Element

```
function HASH INSERT(T, k)                         ▷ where T Hash, k key
    i =0;
    while i < m do
        j =h(k, i);
        if T[j] == null||T[j] == DELETED then
            T[j] = k;
            return
        end if
        i = i + 1;
    end while
    "Error - Overflow"
end function
```

Algorithm 2 Delete Hash Element

```
function DELETE HASH(T, k)                         ▷ where T Hash, k key
    i = 0; j;
    while i < m   ||   T[j]! = null do
        j =h(k, i);
        if T[j] == k then
            T[j] = DELETED;
            return
        end if
        i = i + 1;
    end while
end function
```

2. Second Problem

Do this question by Indicator random variable.
Let's ith element is just inserted now. Then random variable $X_{i,j}$ denoted the number of collision to be occurred when j is inserted.

$I[X_{i,j}]$ is 1 if there is a collision otherwise 0.
$E[I[X_{i,j}]] =$ Probability of event $= \frac{1}{m}$

Expected number of collision is, $E[X] = \sum_1^n \sum_{i+1}^n X_{i,j}$
$E[X] = \sum_1^n \sum_{i+1}^n \frac{1}{m}$
$= \sum_1^n X_i \frac{(n-i)}{m} = \frac{n^2}{m} - \frac{n(n+1)}{2m}$
Expected number of collision is $= \frac{n^2}{2m} - \frac{n}{2m}$

3. Third Problem

3.1

Probability that exactly k, elements goes to a particular slot is.

$Q_k = \binom{n}{k}(\frac{1}{n})^k(\frac{n-1}{n})^{(n-k)}$

$\binom{n}{k}$ for choosing key
$(\frac{1}{n})^k$ for k keys go to a particular slot
$(\frac{n-1}{n})^{n-k}$ for n - k keys goes to other slots

3.2

P_k is probability of maximum number of keys in any slot is exactly k, M = k, and there is only one such slot exist.
$P_k <=$ Probability that some slot has k keys. (This implies k need not to be an maximum number of keys in a hash and it can be any slot. K is free of anything condition). There are n slots to select from probability is nQ_k
Hence $P_k <= nQ_k$

4. Fourth Problem

Let's view this as a 2-D array of dimension m(slots) X L (longest-chain).
Let's select a random slot, k with prob ($\frac{1}{m}$) and say it's length is L_k. Now select a random number, x from 1 to L. if $x > L_k$, it means selected location is empty hence we need to redo the step otherwise return the x element.

Above describe procedure return any key with uniform Probability. Now, let's calculate its running time. Running time is equal to the time until we able to return a key.

Probability of finding it in first attempt is, p = $\frac{n}{mL}$. Hence, total number of trial needed until find the slot is whose length, L_k is greater than randomly selected element, x is $\frac{1}{p} = \frac{mL}{n} = \frac{L}{\alpha}$.

After this, we need to traverse the selected slot-list to find the element x. It would be $O(L)$. Hence expected time is $\frac{L}{\alpha} + O(L) = O(L + \frac{L}{\alpha}) = O(L.(1 + \frac{1}{\alpha}))$