1. Suppose you are given the following keys: 112, 2542, 9992, 5502 and the following hash function h(x) = x mod 10.

Hash the keys using the hash function. How many keys collide?

All of the keys end in 2. All of the keys will be equivalent to 2 mod 10 and as such, they will all collide.

Choose a random1 hash function, ha,b from H10007,10. Include your random choice of ha,b with your answer to this problem. Before you rehash the numbers with the new hash function, determine the probability that the keys 112 and 2542 collide when hashed with ha,b? Hash each of the keys 112, 2542, 992, 502 with ha,b. How many keys collided?

Probability of keys 112 and 2542 colliding:

|  |  |  |
| --- | --- | --- |
| 112 |  | 2 |
| 2542 |  | 2 |
| 992 |  | 2 |
| 502 |  | 2 |

Still has all collisions.

If you had 1000 keys (all keys were positive integers less than 10000) inserted into a hash table of size 2000 using a hash function, ha,b, randomly chosen from H10007,2000 (the family of universal hash functions we defined in class). What is the expected number of collisions you would have if you inserted a new key, x, into the hash table?

Probability of Conflict: n/m = 1000/2000 = 0.5 chances of a collision.

2. Using the technique for perfect hashing that we discussed in class, store the set of keys: {10, 22, 37, 40, 52, 60, 70, 72, 75}.

For your first hash function (“outer hash” function) use ha,b(k) = ((ak + b) mod p) mod m, where a = 3, b = 42, p = 101, m = 9.

Page 278 CLRS

|  |  |  |
| --- | --- | --- |
| H(k) | ((ak + b) mod p) mod m | H(k) |
| H(10) |  | 3 |
| H(22) |  | 6 |
| H(37) |  | 0 |
| H(40) |  | 0 |
| H(52) |  | 3 |
| H(60) |  | 3 |
| H(70) |  | 0 |
| H(72) |  | 6 |
| H(75) |  | 6 |

3. What is the probability that you use at most 16n space for the secondary hash tables? What is the probability (in terms of k) that you use at most kn space (for some constant k)?

With m as the space being used, if we’re using 16n space, then m = 16n

For k spaces:

4. (10 points) How many times must you attempt to construct one of a perfect hash table’s sub-tables so you have one with probability greater than 99.99999%? This is a failure rate of 1 out of one ten million.

∗Many of these questions came from outside sources.

1Use a random number generator.

5. You keep thinking back to the question for homework assignment 1. You are sure you could have solved this faster on average. Here was the question and the new running time you think you can achieve:

Trying to finance your education, you decide to invest your D dollars in the stock market. The problem is you don’t know which stocks to invest in. You decide ask your friends for stock tips and invest in every stock any of your friends suggests. If you receive n unique stock names you will invest D/n in each of the n stocks. Since some of your friends have given you the same stock name, you need to find a way to remove the duplicates.

Design an efficient algorithm to perform this task. You may call any algorithm as a subroutine that was presented in lectures 1-3. Your algorithm must run in O(n) average case time where n is the number of names given to you by your friends.

Demonstrate that your algorithm runs in O(n) average case time.

State your improved average and worst case running time is of your algorithm in big-Oh notation. Justify your reasoning.2

6. Your strategy worked and you have earned 1/2 of your fall tuition on the stocks you invested in. NYU is full of smart people (or you just got lucky...) Most of your friends gave you multiple stock tips. You decide to take out to lunch any friend whose stock tips earned you more than 20% return (i.e. if friend 1 suggested AMD, TSLA, and GME - by following your friends advice you made 22%)3

Let n be the number of stock tips you received, and f is the number of friends, and m ≤ 5 is the maximum number of stock suggestions any one of your friends gave you. In O(n + f) average case time, determine who you will be taking to lunch. You have an array A, where A[i] holds the stock name and the people who suggested you buy the stock and how much money you have made from owning this stock.

Design an efficient algorithm to perform this task. You may call any algorithm as a subroutine that was presented in lectures 1-3. Your algorithm must run in O(n + f) average case time where m is the number of names given to you by your friends.

Justify that your algorithm runs in O(n + f) average case time.

if you wrote a clever algorithm, see if you can justify that your algorithm runs in O(n+f ∗m) average case time where now m ≤ n. If you can do this, you will receive a small amount of extra credit,

7. People started to hear about your talent in the stock market (perhaps you shouldn’t have posted your gains on r/wallstreetbets) and they have started asking you for advice. You are never one to not spot an opportunity, and you decided to write a weekly newsletter and charge people for access. This took off (perhaps it helped that your cousin posted about your success and your for pay newsletter on reddit...)

2

at this point, I don’t think I need to remind you to not take any stock advice from this class!

3

If a friend gave you 3 stock tips, you take the average return for those stock tips and test if the average was more than a 20% return.

2

However your mass email provider has been having problems and sometimes the newsletters are not sent to your subscribers. Not wanting to get a bad reputation, you want to quickly determine if a users complaint is legitimate.

The problem is you haven’t gotten a new computer and you don’t have that much memory to store all your subscribers, so you need to come up with an alternative method to know (within a reasonable probability) if the request is legitimate. If it is legitimate, you quickly forward them the newsletter (you have their email address from the customer support request.)

Here is your plan:

• You create an array S and initialize all the values to ‘n’.

• You then take all your subscribers and hash their email address, and set

S[hash(email address)] = ‘s’.

If a new person subscribes, you hash their email address, and set

S[hash(email address)] = ‘s’.

• If someone emails you saying they didn’t get your newsletter, you quickly hash their

name. If S(hash(email address)] = ‘s’, you send them your newsletter, otherwise you

send them your sign up page on your website.

For this question, assume the simple uniform hashing assumption.

(a) What is the probability a subscriber doesn’t get their newsletter and when you checked

to see if they were a subscriber you found that S[hash(email address)] = ‘s’? Assume

they used the correct email address.

(b) What is the probability a non-subscriber doesn’t get their newsletter and when you

checked to see if they were a subscriber you found that S[hash(email address)] = ‘s’?

8. You started to realize your solution in question 7 was letting a few people have your newsletter without subscribing. You don’t want to buy a new laptop (you are making too much money off the market) so instead you use two hash functions as follows. • You create an array S and initialize all the values to ‘n’.

• You then take all your subscribers and hash their email address, and set

S[hash1(email address)] = ‘s’, and S[hash2(email address)] = ‘s’.

If a new person subscribes, you hash their email address, and set S[hash1(email address)] = ‘s’, and S[hash2(email address)] = ‘s’.

• If someone emails you saying they didn’t get your newsletter, you quickly hash their name. If both S[hash1(email address)] = ‘s’ and S[hash2(email address)] = ‘s’, you send them your newsletter, otherwise you send them your sign up page on your website.

For this question, assume the simple uniform hashing assumption.

(a) What is the probability a subscriber doesn’t get their newsletter and when you checked to see if they were a subscriber you found that S[hash1(email address)] = ‘s’ and S[hash2(email address)] = ‘s’,? Assume they used the correct email address.

(b) What is the probability a non-subscriber doesn’t get their newsletter and when you checked to see if they were a subscriber S[hash1(email address)] = ‘s’ and S[hash2(email address)] = ‘s’,?

3

9. (3 bonus points) Think of a good 4 exam question for the material covered in Lecture 3.