Hash Sampling(TC 11.2-6)

Rivers

July 9, 2018

Problem:

Suppose we have stored n keys in a hash table of size m, with collisions resolved by chaining, and that we know the length of each chain, including the length L of the longest chain. Describe a procedure that selects a key uniformly at random from among the keys in the hash table and returns it in expected time O(L * (1 + m/n)).

Solution:

There is a correct solution on this site: stackoverflow-efficiently-picking-a-random-element-from-a-chained-hash-table And it says:

> Repeat the following steps until an element is returned:
>
> - Randomly select a bucket. Let k be the number of elements in the bucket.
> - Select p uniformly at random from $1...L$. If $p <= k$ then return the pth element in the bucket.
>
> It should be clear that the procedure returns an element uniformly at random. We are sort of applying rejection sampling to the elements placed in a 2D array.
>
> The expected number of elements per bucket is n/m. The probability that the sampling attempt succeeds is $(n/m)/L$. The expected number of attempts needed to find a bucket is therefore $L * m/n$. Together with the $O(L)$ cost of retrieving the element from the bucket, the expected running time is $O(L*(1+m/n)) = O(L \cdot (1 + \alpha))$.
>
> [**?**]

Perhaps the most confusing thing will be the first paragraph. Why does this algorithm succeed in sampling randomly? Well, the reasoning behind

this is pretty simple. The probability of any element being selected is equal to the probability of being selected as the first legal element. And we know that the probability of selecting an illegal element is $1 - \frac{n}{mL}$. So the probability of being selected as the first legal element is

$$\sum_{i=0}^{+\infty} \frac{1}{mL}(1 - \frac{n}{mL})^i = \frac{1}{mL}\frac{(1 - \frac{n}{nL})^{+\infty} - 1}{1 - \frac{n}{mL} - 1} = \frac{1}{n}$$