Ve 281 Homework 2 Gengchen Yang 518370910088. 楊庚辰

Question 1.

Linear Probing:

	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
0		22	22	22	22	22	22	22	22
1								88	88
2									
3									
4				4	4	4	4	4	4
5					15	15	15	15	15
6						28	28	28	28
7							17	17	17
8									59
9			31	31	31	31	31	31	31
10	10	10	10	10	10	10	10	10	10

Quadric Probing

	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
0		22	22	22	22	22	22	22	22
1									
2								88	88
3							27	27	27
4				4	4	4	4	4	4
5									
6						28	28	28	28
7									59
8					15	15	15	15	15
9			31	31	31	31	31	31	31
10	10	10	10	10	10	10	10	10	10

扫描全能王 创建

Double hashing:

	0	1	2	3	4	5	6	7	8
0		22	22	22	22	22	22	22	22
1									59
2						28	28	28	28
3							17	17	17
4				4	4	4	4	4	4
5									
6					15	15	15	15	15
7								88	88
8									
9			31	31	31	31	31	31	31
10	10	10	10	10	10	10	10	10	10

Question 2.

a. Since each key is equally likely to be hashed into each slot, the the possibility of a key to be inserted to this slot is $(\frac{1}{n})$,

k keys inserted: $(\frac{1}{n})^k$;

$(n-k)$ keys not inserted: $(1-\frac{1}{n})^{n-k}$

k out of n keys (select): $C_n^k = \binom{n}{k}$

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

b. P[(one slot contains k keys) && (no ~~other~~ slot contains more than k keys)]

$\leq P$[one slot contains k keys]

$\leq P$[first slot contains k keys] + ... + P[n^{th} slot contains k keys] $= nQ_k$

扫描全能王 创建

c.

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

$$= \left(\frac{1}{n}\right)^k \left(1-\frac{1}{n}\right)^{n-k} \left[\frac{\sqrt{2\pi n}\cdot n^n}{\sqrt{2\pi k}\cdot k^k \cdot \sqrt{2\pi(n-k)}\cdot (n-k)^{n-k}} + O\left(\frac{1}{n}\right) \right]$$

$$\le \frac{\sqrt{n}}{\sqrt{2\pi k(n-k)}} \cdot \frac{n^n}{k^k \cdot (n-k)^{n-k}} \cdot \left(\frac{1}{n}\right)^k \cdot 1.$$

$$\le \frac{n^{n-k}}{(n-k)^{n-k}} \cdot \left(\frac{1}{k}\right)^k \cdot 1$$

$$= \left(1-\frac{k}{k-n}\right)^{n-k} \cdot \frac{1}{k^k}$$

$$\le e^{-\frac{n-k}{(k-n)/k}} \cdot \frac{1}{k^k}$$

$$= \left(\frac{e}{k}\right)^k$$

d. $Q_k < \left(\frac{e}{k}\right)^k$, hence $\ln Q_k < -k \ln k$, when $k = c\log n / \log(\log n)$

we have: $\ln Q_{k_0} < -c\frac{\ln(n)}{\ln(\ln n)} \cdot \ln\left(\frac{\ln n}{\ln\ln n}\right)$

We obviously have $\exists N \; \forall n>N$ $\ln n > 2 \ln\ln n$

Hence $\ln Q_{k_0} < -c\frac{\ln n}{\ln\ln n}\cdot\frac{1}{2}\cdot\ln\ln n = -\frac{1}{2}c\ln n$, when $c=6$,

we have $Q_{k_0} < \frac{1}{n^3}$

Hence $P_k < n Q_k < n\cdot\frac{1}{n^3} = \frac{1}{n^2}$

扫描全能王 创建

e.

$$E[M] = \sum_{k=1}^{n} k \cdot P_k$$

$$= \sum_{k=1}^{k_0} k P_k + \sum_{k=k_0+1}^{n} k P_k$$

$$\le k_0 \cdot \sum_{k=1}^{k_0} P_k + n \sum_{k=k_0+1}^{n} P_k$$

$$\le \frac{c \ln n}{\ln \ln n} \cdot \sum_{k=1}^{k_0} + n \sum_{k=k_0+1}^{n} \frac{1}{n^3}$$

$$= O\left(\frac{\log n}{\log \log n}\right)$$

Question 3.

Groups of 7 will lead to the median of medians ranking from $\left[\frac{2}{7}, \frac{5}{7}\right]$. Hence the corresponding formula would be:

$$T(n) \le cn + T\left(\frac{n}{7}\right) + T\left(\frac{5}{7}n\right), \quad T(1) \le c$$

Taking c to be 7, we have $T(n) \le cn + cn + 5cn = 7cn$

Hence we can still conclude it is still $O(n)$

扫描全能王 创建

Question 4.

a. $\boxed{NO}$, "step", "pest", ... will have the same sum, hence the same hash value.

b. The second is true.

Resizing can't take care of the problem mentioned in a, but ~~resolution~~ truely can place items with same hash value into certain patterned slots.

c. $O(n)$.

The rehashing step needs to insert n elements again to the m' table, which is $n \cdot O(1) = O(n)$.

(Also, to make it ammortized $O(1)$, all steps are: 1. Insert n elements into m table, $O(n)$; rehash, $O(n)$; Insert $(\frac{m'}{m}-1)\cdot n$ elements. $O[(\frac{m'}{m}-1)n]$

Ammortized cost is $O(\frac{m'}{m}+1)$).

d. No, normally it does not.

For example, with $L=0.5$, suppose k is small, then every $\frac{1}{2}k$ new element inserted after the first "boundary" is reached, it needs to rehash again.

then the amortized cost would become $\frac{n_{previous} + \frac{1}{2}k}{\frac{1}{2}k} = O(n)$ in the worst case, instead of $O(1)$.

扫描全能王 创建

Question 5.

	T_0	T_1	T_2	T_3	T_4	T_5	T_6 (rehashing)
0							
1	4371	4371	4371	4371	4371	4371	4371
2							
3		1323	1323	1323	1323	1323	1323
4			6173	6173	6173	6173	6173
5						9679	9679.
6							
7					4344	4344	4344.
8							
9				4199	4199	4199	4199

When 1989 needs to be inserted, we have $(7-1989) \equiv 6 \pmod 7$

where $9+6 \equiv 5$, $5+6 \equiv 1$, $1+6 \equiv 7$, $7+6 \equiv 3$, $3+6 \equiv 9 \pmod{10}$,

but all spaces are occupied.

Hence it needs rehashing.

The first prime number after 20 is 23.

Hence after rehashing,

1° 4371 → slot 1,

2° 1323 → slot 12.

3° 6173 → slot 9.

4° 9679 → slot 19.

5° 4344 → slot 20

6° 4199 → slot 13

7° 1989 → slot 11.

扫描全能王 创建

Question 6.

$$S(L) = \frac{1}{L} \cdot \ln \frac{1}{1-L}, \quad U(L) = \frac{1}{1-L}.$$

Since $S(L) < 2$, $U(L) < 4$, $L < \frac{3}{4}$.

$$\text{Size} \geq \frac{|S|}{L} = \frac{1162}{0.75} \approx 1549.3$$

Hence the first prime number after 1550 is $\boxed{1553}$

Question 7.

a. With open addressing, "insert() calls find()" and if the key is not found, it inserts $(O(1))$ into the blank slot.

The time it takes to find the empty slot is $U(L)$, with a L_{max}, we have $U(L) = \frac{1}{2}[1 + (\frac{1}{1-L})^2]$ for linear probing and $U(L) = \frac{1}{1-L}$ for quadratic probing.

Hence $U(L) \leq \frac{1}{2}[1 + (\frac{1}{1-L_{max}})^2]$ and is $O(1)$.

When a rehashing is needed, assume currently we have M objects, $L=0.5$.

Then inserting these M objects takes $O(M)$,

The rehashing itself is inserting the M objects into a $4M$ table, and also takes $O(M)$,

Inserting $M+1$ to $2M$ objects also takes $O(M)$

Hence the ammortized time for insertion is still $O(M)$

扫描全能王 创建

Yes, it is.

Similarly, first M objects $\rightarrow O(M)$

rehashing into $\alpha \cdot 2M$ table $\rightarrow O(M)$

inserting $(\alpha-1)M$ objects $\rightarrow O((\alpha-1)M)$, $((\alpha-1)M > M$ since $\alpha > 2)$

adding together, it's still ~~O(M)~~ $O(1)$ ~~can~~ time complexity

for the αM objects.

扫描全能王 创建