论题 2-9 作业

姓名：陈劭源　　　　学号：161240004

1 [CS] Problem 5.8

a. When $n \le k$, the probability that all n items hash to different locations is

$$P_1 = \frac{k^{\underline{n}}}{k^n} = \frac{k!}{(k-n)!k^n}$$

and when $n > k$, the probability is 0.

b. When $i \le k+1$, the probability that the ith item is the first collision is

$$P_2(i) = \frac{k^{\underline{i-1}}}{k^{(}i-1)} \times \frac{i-1}{k} = \frac{k!(i-1)}{(k-i+1)!k^i}$$

and when $i > k+1$, the probability is 0.

c. Let X denote the number of items you hash until the first collision, then

$$E(X) = \sum_{i=2}^{k+1}(i-1)P_2(i) = \left(\sum_{i=2}^{k+1}\frac{k!(i-1)i}{(k-i+1)!k^i}\right) - 1$$

d. When $k = 20$, the expected number is 5.29358.

When $k = 100$, the expected number is 12.20996.

2 [CS] Problem 5.11

a. Since we must go through the whole list to ensure that the item is indeed not in the hash table, the running time for an unsuccessful search is $\Theta(1 + length)$, where the expectation of $length$ is n/k, thus the expected running time is $\Theta(1 + n/k)$.

b. Assume there are i elements inserted after the item you are searching for, then the expected running time is

$$t_i = \Theta(1 + \sum_{j=0}^{i} j\binom{i}{j}(1/k)^j(1-1/k)^{i-j}) = \Theta(1 + i/k)$$

And the expected running time for a successful search is

$$\Theta(\frac{1}{n}\sum_{i=0}^{n-1} t_i) = \Theta(\frac{1}{n}\sum_{i=0}^{n-1}(1 + i/k)) = \Theta(1 + (n-1)/2k)$$

Since there is no need to go through the whole list and we can stop searching immediately we've found the item, the expected running time of a successful search is about half of the unsuccessful one.

3 [CS] Problem 5.14

By Theorem 5.15, the expected number of empty slots when hashing $2k$ items into a hash table with k slots is

$$E(X) = k(1 - 1/k)^{2k}$$

and when k grows large, the expected fraction of empty slots is

$$\lim_{k\to\infty} E(X)/k = \lim_{k\to\infty} k(1-1/k)^{2k}/k = \lim_{k\to\infty}(1-1/k)^{2k} = e^{-2}$$

4 [TC] Problem 11.2-3

Since the linked list is not randomly accessible, binary search does not apply and we can only use sequential search, so the modification does not affect the running time for successful searches and unsuccessful searches.

We have to insert the item to the correct position in the list, so the average running time of insertions becomes $\Theta(1+\alpha/2)$ where α is the load factor.

This modification does not affect the running time of deletions, which is still $O(1)$ if the lists are doubly linked.

5 [TC] Problem 11.2-5

There are $|U| > nm$ elements (pigeons), but there are only m slots (holes). By the pigeonhole principle, there exists a subset of U, consisting of n keys that all hash to the same slot. If we want to search for the last element in the slot, the running time is $\Theta(n)$. Therefore, the worst-case searching time for hashing with chaining is $\Theta(n)$.

6 [TC] Problem 11.2-6

```
RANDOM-SELECT($T, m, L$)
1  repeat
2      $x$ = RANDOM($1, m$)
3      $y$ = RANDOM($1, L$)
4  until $y \le T[x].length$
5  $t = T[x].head$
6  for $i = 2$ to $y$
7      $t = t.next$
8  return $t$
```

In line 1-4, we repeat trying to choose a position in the hash table randomly, so the procedure chooses the key uniformly at random.

For each iteration in line 1-4, the probability of success is $n/mL = \alpha/L$. So the total number of iterations follows the geometric distribution with parameter $p = \alpha/L$, and its expectation is $1/p = L/\alpha$. In line 5-7, we search the element in a list, whose length is at most L, so the running time is $O(L)$. Therefore, the total expected running time is $O(L+L/\alpha) = O(L(1+1/\alpha))$.

7 [TC] Problem 11.3-3

The string can be represented as

$$S = \sum_{i=0}^{n-1} S_i (2^p)^i$$

Consider $(2^p)^i \mod (2^p-1)$, we have

$$(2^p)^i \bmod (2^p-1) = (2^p \bmod (2^p-1))^i = 1^i = 1$$

therefore

$$S \bmod (2^p-1) = \sum_{i=0}^{n-1} S_i (2^p)^i \bmod (2^p-1) = \sum_{i=0}^{n-1} S_i \bmod (2^p-1)$$

The result is only in terms of the sum of the characters in the string and has nothing to do with the position of each character. That means, if string y is a permutation of string x, then they hash to the same value.

If there are many string pairs that one of them is a permutation of the other, such as ‘POT’ and ‘TOP’, in our application, then the hash function will lead to a lot of collisions, which is undesirable.

8 [TC] Problem 11.3-4

k	kA	$kA \bmod 1$	$m(kA \bmod 1)$	$h(k)$
61	37.70007	.70007	700.07	700
62	38.31811	.31811	318.11	318
63	38.93614	.93614	936.14	936
64	39.55418	.55418	554.18	554
65	40.17221	.17221	172.21	172

9 [TC] Problem 11.4-2

```
HASH-DELETE($T,k$)
$T[$HASH-SEARCH$(T,k)] =$ DELETED
```

```
HASH-INSERT($T,k$)
$i = 0$
repeat
    $j = h(k,i)$
    if $T[j] ==$ NIL or $T[j] ==$ DELETED
        $T[j] = k$
        return $j$
    else
        $i = i+1$
until $i == m$
error “hash table overflow”
```

10 [TC] Problem 11.4-3

By Theorem 11.6 and Theorem 11.8, when the load factor is $3/4$, the expected number of probes in an unsuccessful search is at most $1/(1-\alpha) = 4$, the expected number in a successful search is at most $(1/\alpha)\ln 1/(1-\alpha) = 1.848$; when the load factor is $7/8$, the expected number of probes in an unsuccessful search is at most $1/(1-\alpha) = 8$, the expected number in a successful search is at most $(1/\alpha)\ln 1/(1-\alpha) = 2.377$.

11 [TC] Problem 11-1

a. If an insertion requires strictly more than k probes, the first k probes are unsuccessful. For the $(i+1)$th probe, the probability of unsuccessful probe is $(n-i)/(m-i) \le 1/2$, since the probe sequence is a random permutation of $0, 1, \cdots, m-1$ according to uniform hashing assumption, and the probability that all the k probes are unsuccessful is less than 2^{-k}. So the probability is at most 2^{-k}.

b. Substituting $k = 2\lg n$ into ***a.*** yields that the probability is at most $2^{-2\lg n} = n^{-2}$, i.e. the probability is $O(1/n^2)$.

c. Consider the complementary event, that $X \le 2\lg n$, which is equivalent to that for all i, $X_i \le 2\lg n$. Because $X_1, X_2, \cdots, X_n$ are independent, we have

$$\begin{aligned} P(X \le 2\lg n) &= \prod_{i=1}^{n}(1 - P(X_i > 2\lg n)) \\ &\ge (1-n^{-2})^n \\ &\ge 1 - 1/n & \text{(By Bernoulli's inequality)} \end{aligned}$$

Hence,

$$P(X > 2\lg n) = 1 - P(X \le 2\lg n) \le 1/n$$

Therefore, $P(X > 2\lg n) = O(1/n)$.

d. Since the probe sequence is a permutation of $1, 2, \cdots, m-1$, the maximum length of the probe sequence is the items in the hash table plus one, which is at most n. Hence, the expected length of the longest probe is

$$\begin{aligned} E[X] &= \sum_{i=1}^{n} iP(X=i) \\ &= \sum_{i=1}^{\lfloor 2\lg n\rfloor} iP(X=i) + \sum_{i=\lfloor 2\lg n\rfloor+1}^{n} iP(X=i) \\ &\le \lfloor 2\lg n\rfloor \sum_{i=1}^{\lfloor 2\lg n\rfloor} P(X=i) + n \sum_{i=\lfloor 2\lg n\rfloor+1}^{n} P(X=i) \\ &= \lfloor 2\lg n\rfloor \sum_{i=1}^{\lfloor 2\lg n\rfloor} P(X=i) + nP(X > 2\lg n) \\ &\le 2\lg n + 1 \end{aligned}$$

Therefore, $E[X] = O(\lg n)$.

12 [TC] Problem 11-2

(a) Let random variable X_i denote the number of keys hashing to the ith slot. X_i follows the binomial distribution with parameter n and $p = 1/n$. Therefore

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

(b) Let $E_{i,k}$ denote the event that the ith slot contains the most keys and it contains k keys. Since $E_{i,k} \subset (X_i = k)$, we have $P(E_{i,k}) \leq Q_k$. Therefore,

$$P_k = P(M = k) = P\left(\bigcup_{i=1}^{n} E_{i,k}\right) \leq \sum_{i=1}^{n} P(E_{i,k}) \leq nQ_k$$

(c) First, we have to bound the binomial coefficient

$$\begin{aligned}
\binom{n}{k} &= \frac{n!}{(n-k)!k!} \\
&\leq \frac{n^k}{k!} \\
&\leq n^k \left(\frac{e}{k}\right)^k && \text{(By equation (3.18), Stirling's approximation)}
\end{aligned}$$

Then we have

$$\begin{aligned}
Q_k &= \left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{n-k} \binom{n}{k} \\
&< \left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{n-k} n^k \left(\frac{e}{k}\right)^k \\
&\leq \left(\frac{1}{n}\right)^k n^k \left(\frac{e}{k}\right)^k \\
&= e^k / k^k
\end{aligned}$$