习题二

Hachey

2.1

某应用需要在 10 人中以加密方式共享一个 100bit 的信息 s，使得其中任意两人根据自己收到的信息能够恢复原始信息，但任意一人无法根据自身收到的信息了解 s 的任何情况。为此，10 位相关人员依次编号为 $0,1,2,\cdots,9$。一种共享信息的方法如下：选择一个长度为 101 比特的素数 q,并将其剩余域记为 $\mathrm{GF}(q)$。在 $\mathrm{GF}(q)$ 中均匀一致地选定元素 f，并利用拉格朗日插值法获得一个系数取自 $\mathrm{GF}(q)$ 的一次多项式 $p(x)=(x-10)s-(x-11)f$，使得 $p(10)=f, p(11)=s$。第 i 个人收到的信息定义为 $p(i)$。

(a) 请你说明如何根据计算从任意两人收到的信息中恢复 s。

(b) 请你利用概率知识说明任何人仅凭自己收到的信息无法获知 s 的任意有价值信息。

解：

(a) 设第 i 个人收到的信息为 $p(i)$，第 j 个人收到的信息为 $p(j)$，其中 $i\neq j$。由题，我们有

$$p(i)=(i-10)s-(i-11)f$$
$$p(j)=(j-10)s-(j-11)f$$

解上述方程组可得

$$s=\frac{11-j}{i-j}p(i)-\frac{11-i}{i-j}p(j)$$

因此，我们可以根据计算从任意两人收到的信息中恢复 s。

(b) 任何一个人收到的信息为 $p(i)$，即多项式 $p(x)$ 在 $x=i$ 处的值。由于 $p(x)$ 是一个一次多项式，而且 f 是在 $\mathrm{GF}(q)$ 中均匀一致地选定的，所以 $p(i)$ 的值是随机的，不包含关于 s 的任何信息。换句话说，$p(i)$ 的值是均匀分布的，因此任何一个人无法从 $p(i)$ 推断出 s 的任何信息。这是因为在 $\mathrm{GF}(q)$ 中，$p(i)$ 的每一个可能的值都是等概率的，与 s 的取值无关。因此，没有人可以单独通过自己收到的信息了解 s 的任何情况。

2.2

投掷一枚均匀硬币 n 次，如果第 i 次投掷和第 j 次投掷出现同一面，则令 $X_{ij}=1$, 否则，令 $X_{ij}=0$。证明：$X_{ij}(i<j)$ 两两独立但不相互独立。

证明：

首先证明 X_{ij} 两两独立。 对于任意的 i,j,k,l 满足 $i<j$ 且 $k<l$，需证明 X_{ij} 和 X_{kl} 独立，即证

$$P(X_{ij}=n, X_{kl}=m) = P(X_{ij}=n)\cdot P(X_{kl}=m)$$

其中 $n,m\in\{0,1\}$。设第 i 次投掷正面朝上时，令 $\xi_i=1$；否则，令 $\xi_i=0$。以 $n=1,m=1$ 为例，有

$$P(X_{ij}=1, X_{kl}=1) = P(\xi_i=\xi_j, \xi_k=\xi_l)$$

$$P(X_{ij}=1) = P(\xi_i=\xi_j), P(X_{kl}=1) = P(\xi_k=\xi_l)$$

由于硬币是均匀的，所以 ξ_i,ξ_j,ξ_k,ξ_l 是独立的，因此

$$P(\xi_i=\xi_j, \xi_k=\xi_l) = P(\xi_i=\xi_j)\cdot P(\xi_k=\xi_l)$$

n,m 的值为其他情况的证明类似。综上所述，X_{ij} 和 X_{kl} 独立。

接下来证明 X_{ij} 不相互独立。 考虑三个随机变量 X_{ij},X_{ik},X_{jk}，其中 $i<j<k$。如果我们知道 $X_{ij}=1$ 和 $X_{ik}=1$，那么一定有 $\xi_i=\xi_j=\xi_k$，从而有 $X_{jk}=1$。设事件 A 表示 $X_{ij}=1$，事件 B 表示 $X_{ik}=1$，事件 C 表示 $X_{jk}=1$，则有

$$P(C|A,B)=1$$

那么

$$P(C,A,B) = P(C|A,B)\cdot P(A,B) = P(A,B)$$

由 (a) 知 $P(A,B)=P(A)\cdot P(B)=1/4$，所以 $P(C,A,B)=1/4$。而 $P(A)=P(B)=P(C)=1/2$，所以

$$P(C,A,B)\neq P(A)\cdot P(B)\cdot P(C)$$

因此，X_{ij} 不相互独立。

综上所述，X_{ij} 两两独立但不相互独立。

2.3

假设 $x_1,x_2,\cdots,x_n$ 是从 $\left[0,2^k\right)$ 中两两独立地均匀抽取的 n 个数。

证明：用桶排序 $x_1,x_2,\cdots,x_n$ 时，时间复杂度的数学期望仍然是线性的。

证明：

设桶的个数为 m，第 j 个桶有 X_j 个数。由于 $x_1,x_2,\cdots,x_n$ 为均匀抽取，则 X_j 服从二项分布，即 $X_j\sim B(n,1/m)$。因此，X_j 的数学期望为 $E(X_j)=n/m$，方差为 $Var(X_j)=$

$n/m \cdot (1-1/m)$，从而 $E(X_j^2) = Var(X_j) + E(X_j)^2 = n/m \cdot (1-1/m) + (n/m)^2$。当 $m \approx n$ 时，$E(X_j^2) \approx 2 - 1/m$。

设桶排序的时间复杂度为 T，则

$$T = O\left(n + \sum_{j=1}^{m} X_j^2 + m\right)$$

T 的数学期望为

$$\begin{aligned} E(T) &= O(n+m) + O\left(\sum_{j=1}^{m} E\left(X_j^2\right)\right) \\ &= O(n+m) + O(m(2-1/m)) \\ &= O(n) \end{aligned}$$

因此，桶排序的时间复杂度的数学期望仍然是线性的。

2.4

身份证号码的前 6 位表示地区编码，中间 8 位是生日，最后 k 位是 k 个 $[0,9]$ 之间的随机数字。为了确保身份证号码以 99% 的概率具有唯一性，试建立模型确定 k 的取值。

解：

根据题意，只有同一地区、同一生日的人的身份证号码才可能会重复。设同一地区、同一生日的人数为 m，则问题转化为：将 m 个球放入 10^k 个箱子中，设事件 ε 表示不存在含有两个球的箱子，求使 $P(\varepsilon) \geq 0.99$ 成立的最小的 k。由生日悖论可知，$P(\varepsilon) \approx e^{-m^2/2n}$，其中 $n = 10^k$。因此，我们有

$$e^{-m^2/2n} \geq 0.99$$

化简得 $n \geq -m^2/2\ln(0.99)$，即

$$k \geq \log_{10}\left(-\frac{m^2}{2\ln(0.99)}\right)$$

假设 $m = 100$，则 $k \geq 5.70$，因此 $k = 6$ 时可以满足题意。

2.5

在开放寻址散列法中，散列表是一个数组 A 且每个桶均无拉链。数组中每个位置要么包含一个散列项要么是空的。对每个待散列对象 x，散列函数 h 定义了数组中的探测位置序列 $h(x,0), h(x,1), \cdots$。 Insert(x)如下操作：按照 h 定义的探测位置序列 $h(x,0), h(x,1), \cdots$ 在数

组中寻找空位置 k，并将 x 存入 $A[k]$。 Find(x)如下操作：依次探查 h 定义的探测位置序列 $h(x,0),h(x,1),\cdots$ 中的每个位置；如果 $A[h(x,i)] = x$，则返回 $h(x,i)$；否则，返回 -1 表明 x 未出现在散列表中。

假设用具有 $2n$ 个存储位置的数组作为开放寻址散列表存储 n 个数据项，并且 $h(x,i)$ 服从 $0,1,\cdots,2n-1$ 上的均匀分布，$h(x,1),h(x,2),\cdots$ 相互独立。用 $X_i(1 \le i \le n)$ 表示第 i 次执行 Insert 操作时探查的位置个数， $X = \max_i X_i$ 表示 n 次插入操作中各次操作的最大探查次数。

(1) 证明：Insert(x)需要探查 a 个存储位置的概率至多为 2^{-a} ；

(2) 证明：$\Pr[X_i > 2\log n] \le 1/n^2$;

(3) 证明：$\Pr[X > 2\log n] \le 1/n$;

(4) 证明：$E[X] = O(\log n)$。

证明：

(1) 由题知 $h(x,i)$ 服从 $0,1,\cdots,2n-1$ 上的均匀分布，且相互独立。由于 $2n$ 个存储位置中存放了 n 个数据项，则每个数据项占据每个存储位置的概率为 $n/(2n) = 1/2$。

那么，Insert(x)需要探查 1 个存储位置的概率为 $\frac{n}{2n}$，需要探查 2 个存储位置的概率为 $\frac{n}{2n} \cdot \frac{n-1}{2n}$。以此类推，需要探查 a 个存储位置的概率为

$$\frac{n}{2n} \cdot \frac{n-1}{2n} \cdots \frac{n-a+1}{2n} \le \left(\frac{1}{2}\right)^a = 2^{-a}$$

所以，Insert(x)需要探查 a 个存储位置的概率至多为 2^{-a}。

(2) 由 (1) 知 $\Pr[X_i \ge a] \le 2^{-a}$，那么

$$\Pr[X_i > 2\log n] \le 2^{-2\log n} = 1/n^2$$

(3) 由题，有

$$\begin{aligned}
\Pr[X > 2\log n] &= \Pr[\max X_i > 2\log n] \\
&= 1 - \Pr[\max X_i \le 2\log n] \\
&= 1 - \prod_{i=1}^{n} \Pr[X_i \le 2\log n] \\
&= 1 - \prod_{i=1}^{n} (1 - \Pr[X_i > 2\log n])
\end{aligned}$$

由 (2) 知 $\Pr[X_i > 2\log n] \leq 1/n^2$，所以

$$\begin{aligned}\Pr[X > 2\log n] &\leq 1 - \left(1 - \frac{1}{n^2}\right)^n \\ &\leq 1 - \left(\mathrm{e}^{-\frac{1}{n^2}}\right)^n \\ &= 1 - \mathrm{e}^{-\frac{1}{n}} \\ &\leq 1 - (1 - 1/n) \\ &= 1/n\end{aligned}$$

(4) 首先，定义指示器随机变量 I_a，

$$I_a = \begin{cases} 1, & \text{if exists } X_i \geq a \\ 0, & \text{if exists } X_i < a \end{cases}$$

那么有

$$X = \max_{1\leq i\leq n} X_i = \sum_{a=1}^{\infty} I_a$$

由于 X 是最大探查次数，我们可以将 X 的期望值表示为所有 I_a 的期望值之和：

$$E[X] = E\left[\sum_{a=1}^{\infty} I_a\right] = \sum_{a=1}^{\infty} E[I_a]$$

由 (2)，I_a 的期望

$$\begin{aligned}E[I_a] &= 1 \cdot P(X_i \geq a) + 0 \cdot P(X_i < a) \\ &= P(X_i \geq a) \\ &\leq \begin{cases} 1/n^2, & \text{if } a > 2\log n \\ 1, & \text{otherwise} \end{cases}\end{aligned}$$

因此，$E[X]$ 可以被限制为：

$$\begin{aligned}E[X] &\leq \sum_{a=1}^{\lfloor 2\log n\rfloor} 1 + \sum_{a=\lfloor 2\log n\rfloor+1}^{\infty} 1/n^2 \\ &= \lfloor 2\log n\rfloor + \sum_{a=\lfloor 2\log n\rfloor+1}^{\infty} 1/n^2 \\ &\leq \lfloor 2\log n\rfloor + \frac{\pi^2}{6}\end{aligned}$$

从而我们得到：

$$E[X] = O(\log n)$$

2.6

假设我们有 m 首歌曲和 n 名听众，每名听众有一个自己喜欢的歌曲列表。如果两名听众共同喜欢的歌曲越多，则说明听众的音乐口味越趋于相同。试利用本章所学内容，设计一个高效的算法找出口味相同的听众对。允许大家在完成作业时对题目中未明确的部分进行自由发挥。

解：

考虑使用 MinHash 算法来解决这个问题。取哈希函数的个数为 30，并定义两听众喜欢的歌曲列表的相似度不小于 0.8 时，认为他们的口味相同。算法伪代码如下：

Algorithm 1: FindSimilarListeners(S, L)

Input: 歌曲列表 S，听众列表 L，每个听众 L_i 喜欢的歌曲列表 S_i

Output: 口味相似的听众对集合 R

```
begin
    R ← ∅ ;
    𝓗 ← RandomHash() for i = 1 to 30 ;
    M ← Matrix(n, 30) ;
    foreach S_i do
        foreach h_j ∈ 𝓗 do
            M_ij = min{h_j(s) | s ∈ S_i} ;
        end
    end
    for i = 1 to n − 1 do
        for j = i + 1 to n do
            s ← 0 ;
            for k = 1 to 30 do
                if M_ik = M_jk then
                    s ← s + 1 ;
                end
            end
            if s/30 ≥ 0.8 then
                R ← R ∪ {(L_i, L_j)} ;
            end
        end
    end
    return R;
end
```