高级算法作业一

1190200122 袁野

2022 年 3 月 20 日

1.1 切比雪夫不等式为

$$Pr(|X - E[X]| \geq t) \leq \frac{Var(X)}{t^2}$$

设 X_1 为将 n 个球均匀随机地投入 n 个箱子之后第一个箱子的球数，设 $Y_i = 1$ 为第 i 个球落入第一个盒子，$Y_i = 0$ 为第 i 个球未落入第一个盒子，则有

$$Pr(Y_i = 1) = \frac{1}{n}$$

$$E(X_1) = \sum_{i=1}^{n} Pr(Y_i = 1) = 1$$

$$Var(X_1) = np(1-p) = 1 - \frac{1}{n}$$

代入切比雪夫不等式可有

$$Pr(|X_1 - 1| \geq t) \leq \frac{1 - \frac{1}{n}}{t^2} \quad (t > 0)$$

$$Pr(X_1 \geq t + 1) \leq \frac{1 - \frac{1}{n}}{t^2} \quad (t > 0)$$

$$Pr(X_1 > t) \leq \frac{n-1}{nt^2} \quad (t > 0)$$

故第一个箱子中球数大于 t 的概率不超过 $\frac{n-1}{nt^2}$。

1.2

(1) 对于服从标准正态分布的随机变量 X 有

$$E(X) = 0, Var(X) = 1, E(X^2) = 1$$

则有

$$E(X) = E(\sum_{i=1}^{n} r_i \cdot x_i) = \sum_{i=1}^{n} E(r_i) \cdot x_i = \sum_{i=1}^{n} 0 \cdot x_i = 0$$

同理

$$E(Y) = 0$$

由公式

$$Var(X) = E(X^2) - E^2(X)$$

有

$$Var(X) = E((\sum_{i=1}^{n} r_i \cdot x_i)^2) = E(\sum_{i=1}^{n}\sum_{j=1}^{n} r_i r_j x_i x_j) = \sum_{i=1}^{n}\sum_{j=1}^{n} E(r_i r_j) x_i x_j$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n}[i \neq j]E(r_i)E(r_j)x_i x_j + \sum_{i=1}^{n} E(r_i{}^2)x_i{}^2 = \sum_{i=1}^{n} E(r_i{}^2)x_i{}^2 = \sum_{i=1}^{n} x_i{}^2 = 1$$

同理

$$Var(Y) = 1$$

(2)

$$Var(X) = E(X^2) - E(X)^2 = E(X^2) - 0 = E(X^2) = 1$$

同理

$$E(Y^2) = 1$$

由 Cauchy-Schwartz 不等式可知

$$E^2(XY) \leq E(X^2) * E(Y^2) = 1$$

$$-1 \leq E(XY) \leq 1$$

故有

$$E((X-Y)^2) = E(X^2) + E(Y^2) - 2E(XY) = 2 - 2E(XY)$$

$$0 \leq E((X-Y)^2) \leq 4$$

1.2

(1) 属于舍伍德算法。

(2) 每次考虑 S' 是 S 的子集，那么显然 $|S'| < |S| < n$，则 $\exists b < 1 : |S'| = bn$。证毕。

(3) 定义 $S_{(i)}$ 为 S 中阶为 i 的元素，$X_{i,j}$ 为 $S_{(i)}$ 和 $S_{(j)}$ 比较的次数，显然有 $0 \leq X_{ij} \leq 1$。则算法的比较次数为 $\sum_{i=1}^{n}\sum_{j>i} X_{i,j}$。那么易得算法的平均复杂性为

$$E[\sum_{i=1}^{n}\sum_{j>i} X_{i,j}] = \sum_{i=1}^{n}\sum_{j>i} E[X_{i,j}]$$

而

$$E[X_{ij}] = p_{ij} \times 1 + (1 - p_{i,j}) \times 0 = p_{ij}$$

所以我们考虑求得 p_{ij}。分类讨论如下：

1) 当 $i < k < j$ 时，令 A 为随机事件 $X_{ij} = 1$，B 为随机事件算法首次选中 $S_{(i)\ (j)}$ 中元素作为划分元素。B 事件必然发生，因为至少会选择一次 $S_{(k)}$ 作为划分元素。则有

$$p(A|B) = \frac{2}{j-i+1}$$

$$p(A) = \sum_{\alpha} p(A|B_\alpha)p(B_\alpha) = \frac{2}{j-i+1}$$

2) 当 $k \leq i < j$ 时，令 A 为随机事件 $X_{ij} = 1$，B 为随机事件算法首次选中 $S_{(k)\ (j)}$ 中元素作为划分元素。B 事件必然发生，因为至少会选择一次 $S_{(k)}$ 作为划分元素。则有

$$p(A|B) = \frac{2}{j-k+1}$$

$$p(A) = \sum_{\alpha} p(A|B_\alpha)p(B_\alpha) = \frac{2}{j-k+1}$$

3) 当 $i < j \leq k$ 时，令 A 为随机事件 $X_{ij} = 1$，B 为随机事件算法首次选中 $S_{(i)\ (k)}$ 中元素作为划分元素。B 事件必然发生，因为至少会选择一次 $S_{(k)}$ 作为划分元素。则有

$$p(A|B) = \frac{2}{k-i+1}$$

$$p(A) = \sum_{\alpha} p(A|B_\alpha)p(B_\alpha) = \frac{2}{k-i+1}$$

则有

$$\begin{aligned}
E[T(n)] &= \sum_{i=1}^{n}\sum_{j>i} E[X_{ij}] \\
&= \sum_{i=1}^{k-1}\sum_{i<j\leq k}\frac{2}{k-i+1} + \sum_{j=k+1}^{n}\sum_{k\leq i<j}\frac{2}{j-k+1} + \sum_{i=1}^{k-1}\sum_{j=k+1}^{n}\frac{2}{j-i+1} \\
&= 2\sum_{i=1}^{k-1}\frac{k-i}{k-i+1} + 2\sum_{j=k+1}^{n}\frac{j-k}{j-k+1} + \sum_{i=1}^{k-1}\sum_{j=k-i+1}^{n-i}\frac{2}{j+1} \\
&= 2\sum_{i=1}^{k-1}(1-\frac{1}{i+1}) + 2\sum_{j=1}^{n-k}(1-\frac{1}{j+1}) + \sum_{i=1}^{k-1}\sum_{j=k-i+1}^{n-i}\frac{2}{j+1} \\
&= 2n - O(log(k)) - O(log(n-k)) + \sum_{i=1}^{k-1}\sum_{j=k-i+1}^{n-i}\frac{2}{j+1}
\end{aligned}$$

$$\begin{aligned}
\sum_{i=1}^{k-1}\sum_{j=k-i+1}^{n-i}\frac{2}{j+1} &= 2(\sum_{i=1}n\sum_{j=2}^{n-i}\frac{1}{j} - \sum_{i=1}k\sum_{j=2}^{k-i}\frac{1}{j} - \sum_{i=1}n-k\sum_{j=2}^{n-k-i}\frac{1}{j}) \\
&= O(n\,log(n)) - O(k\,log(k)) - O((n-k)\,log(n-k)) \\
&= O(n\,log(\frac{n}{n-k})) - O(k\,log(\frac{k}{n-k})) \\
&\approx O(n)
\end{aligned}$$

综上

$$E(T(n)) = O(n)$$

1.3

算法 1 判断多项式乘积相等

输入: 阶数 m, n, l 和多项式 $p(x), q(x), r(x)$

输出: 如果满足 $p(x)q(x) = r(x)$，输出 True，否则以一定的概率输出 False。

$T \leftarrow 2l$
if $m + n \neq l$ **then**
return False
end if
for $i = 1 \rightarrow$ T **do** 在 $[0, 1000l]$ 均匀随机生成一个整数 x;
    if $p(x)q(x) \neq r(x)$ **then**
return False
    end if
end for
return True

该算法时间复杂度为 $O(l^2)$。

两个不同的 l 阶多项式的交点最多有 l 个，设 $A_i = 1$ 表示第 i 个随机生成的数字为交点，$A_i = 0$ 表示第 i 个随机生成的数字不为交点，因此 $p(A_i = 1) = \frac{l}{1000l} = \frac{1}{1000}$。结果正确的概率极即为 $p = 1 - (\frac{1}{1000})^{2l}$。

该算法属于蒙特卡洛算法。

1.4

算法 2 判断矩阵乘积相等

输入: 阶数 p, q, r 和矩阵 A, B, C

输出: 如果满足 $A \times B = C$，输出 True，否则以一定的概率输出 False。

$N \leftarrow maxp, q, r$
$T \leftarrow 2N$
for $i = 1 \rightarrow$ T **do** 等概率抽取 $v \in 0, 1^r$;
    if $A(Bv) \neq Cv$ **then**
return False
    end if
end for
return True

算法时间复杂度为 $O(max(p, q, r)^3)$。

当结果错误时，只有一种情况，即 $AB \neq C$ 但是 $ABv = Cv$，记作事件 α。

设 $D = AB - C$，当 α 发生时显然有 $D \neq 0, Dv = 0$。

设 D_i 为 D 的一个行向量，且 $D_i \neq 0, D_i v = 0$。

$$D_i \times v = \sum_{j=1}^{r} D_{i,j} v_j = 0$$

由 $D_i \neq 0$ 可知 $\exists j:\ D_{i,j} \neq 0$ 则有

$$v_j = -\frac{\sum_{k=1 \wedge k \neq j}^{r} D_{i,k} v_k}{D_{i,j}}$$

即 v 有 $n-1$ 个自由基，α 这种情况下的 v 只有 2^{r-1} 个，而我们随机生成的 v 可能有 2^n 个，故有

$$p(\alpha) = \frac{2^{r-1}}{2^r} = \frac{1}{2}$$

也就是说当返回 False 时百分百正确，当返回 true 时正确概率为 $\frac{1}{2}$。当我们独立重复 k 次后，得到正确解的概率即为 $1 - \frac{1}{2^k}$。
算法类别为蒙特卡洛算法。

1.5 考虑克鲁斯卡尔算法求最小生成树的过程我们可以想到，先将所有的边按照边权排序，然后从小到大依次枚举，如果两个端点不在同一个集合，那么将这个点的两个端点所在的集合相连为一个集合，显然在做完所有的操作之后整张图的所有点会被练到一个集合里。我们再考虑 contraction 求最小割的算法，其随机挑选边并压缩的过程实际上与克鲁斯卡尔枚举边并将点集相连的操作是等价的。如果该算法正确，那么满足生成树上最大边两端的两个集合为最小割之后的两个集合，那么必然需要保证最小生成树上的先枚举的 $n-2$ 较小边均不为最小割中的边。对比 contraction 算法，上述满足的情况即等价为算法结束时最小割中无边被收缩。2.6 定理 2 内容如下:

设 C 是一个 *min-cut*，其大小为 k，在 *contraction* 求最小割算法结束时，C 中无边被收缩过的概率大于 $\frac{2}{n^2}$

对比该定理的证明过程，我们证明该题目算法的结果中除了最小生成树中最大边外，无边在最小生成树中：
C 为最小边，A_i 表示克鲁斯卡尔生成最小生成树时，第 i 步没有选中 C 边，$1 \le i \le n-2$。
在第一步中选中的边在 C 中的概率最多为 $\frac{k}{\frac{kn}{2}} = \frac{2}{n}$, 即

$$Pr(A_1) \ge 1 - \frac{2}{n}$$

在第二步中，若 A_1 发生，则至少有 $\frac{k(n-1)}{2}$ 条边，选中 C 中边的概率为 $\frac{2}{n-1}$，即

$$Pr(A_2|A_1) \ge 1 - \frac{2}{n-1}$$

在第 i 步中，若 A_1 至 A_i 发生，则有 $n-i+1$ 个节点，即至少有 $\frac{k(n-i+1)}{2}$ 于是

$$Pr(A_i|\bigcap_{1\le j<i} A_j) \ge 1 - \frac{2}{n-i+1}$$

最后我们有

$$Pr(\bigcap_{1\le i\le n-2} A_i) \ge \prod_{1\le i\le n-2} (1 - \frac{2}{n-i+1}) = \frac{2}{n(n-1)} > \frac{2}{n^2}$$

结合该定理，本题算法中最小生成树上的先枚举的 $n-2$ 较小边均不为最小割中的边的概率大于 $\frac{2}{n^2}$。
即该算法正确的概率大于 $\frac{2}{n^2}$。
故输出最小割的概率为 $\Omega(\frac{1}{n^2})$

1.6
(1) 若 I 不是一个独立集，那么有 $\exists u, v \in I, (u,v) \in E$, 我们不妨设 u 的标签小于 v 的标签，那么当 u 被加入 I 之后，由于 $uv \in E$，那么 v 会被删除，则必然满足 $v \notin I$，与假设不符，故 I 是 $G = (V, E)$ 的一个独立集。
(2) 该问题结论并不正确，实际上 $p(B) \ge \frac{1}{d_u+1}$

1.7 设 X_{ij} 表示第 i 个元素和第 j 个元素交换的次数。在冒泡排序中，两个数字如果是倒置元素对，在它们相互间换之后一定不会再一次交换，因此 X_{ij} 最大只能为 1。设 pij 为第 i 个元素和第 j 个元素为倒置元素对的概率。显然 $p_{ij} = \frac{1}{2}$。则 $E[X_{ij}] = 1 \times p_{ij} + 0 \times (1 - p_{ij}) = p_{ij}$

$$E[\sum_{i=1}^{n}\sum_{j=i+1}^{n} X_{ij}] = \sum_{i=1}^{n}\sum_{j=i+1}^{n} E[X_{ij}] = \sum_{i=1}^{n}\sum_{j=i+1}^{n} p_{ij} = \frac{n(n-1)}{4}$$

交换的期望次数为 $\frac{n(n-1)}{4}$

1.8

算法 3 求 $F(z)$

输入: 整数 $z(0 \le z \le n-1)$

输出: 以大于 $\frac{1}{2}$ 正确的概率输出 $F(Z)$

1: 等概率在 $[0, n-1]$ 中生成整数 x

2: $y \leftarrow (z - x + n) mod\ n$ **return** $(F(x) + F(y)) mod\ m$

对于每个数字被篡改的概率为 $\frac{1}{5}$

设 $r[i]$ 表示 $F(i)$ 正确，则

$$p(r[i]) = \frac{4}{5}$$

$$p(r[x] \wedge r[y]) \le \frac{16}{25} \le \frac{1}{2}$$

当运行三次后，返回出现次数较多的结果，此时

$$p(Right) \ge C_3^2(1 - \frac{16}{25})\frac{16}{25}^2 + C_3^3\frac{16}{25}^3 = 0.704512$$