



中国计算机学会
China Computer Federation

Power of Randomization

高嘉煊

IIIS, TSINGHUA UNIVERSITY

SAMJIA2000@126.COM



前言

在算法中使用了随机函数，且随机函数的返回值直接或者间接的影响了算法的执行流程或执行结果，这样的算法被称为随机化算法

(randomized algorithm)。就是将算法的某一步或某几步置于运气的控制之下，即该算法在运行的过程中的某一步或某几步涉及一个随机决策，或者说其中的一个决策依赖于某种随机事件。

在一些问题中，在实践或是理论的层面，算法中的随机化都可以给出不错的结果。

本文将介绍一些概率论的基本概念以及工具，比如Markov bound、Chernoff bound、Markov chain等等。

同时会介绍一些随机化思想的应用以及一些有代表性的问题，比如近似计算#DNF等等。



概率论

- 样本空间/sample space : 一个试验的所有可能结果的集合
 - 每个样本有一定的概率出现，所有样本出现的概率之和为1，记样本 ω 发生的概率为 $p(\omega)$
- 事件/event : 样本空间的一个子集
 - 事件发生的概率为事件中所有样本发生的概率之和，即 $\Pr[A] = \sum_{\omega \in A} p(\omega)$
- eg.
 - 投掷两枚硬币，考虑哪一面朝上，那么样本空间一共有四种结果：
 - $S=\{(H,H), (H,T), (T,H), (T,T)\}$
 - 令事件E表示“两枚硬币投掷的结果一样”，则 $E=\{(H,H), (T,T)\}$



概率论

- 独立事件A,B，满足 $\Pr[A \cap B] = \Pr[A] \Pr[B]$
- 条件概率 $\Pr[A|B]$ 表示B发生的条件下A发生的概率，即 $\Pr[A|B] = \frac{\Pr[A \cap B]}{\Pr[B]}$
- 全概率公式： $\Pr[A] = \Pr[A|B] \Pr[B] + \Pr[A|\bar{B}] \Pr[\bar{B}]$

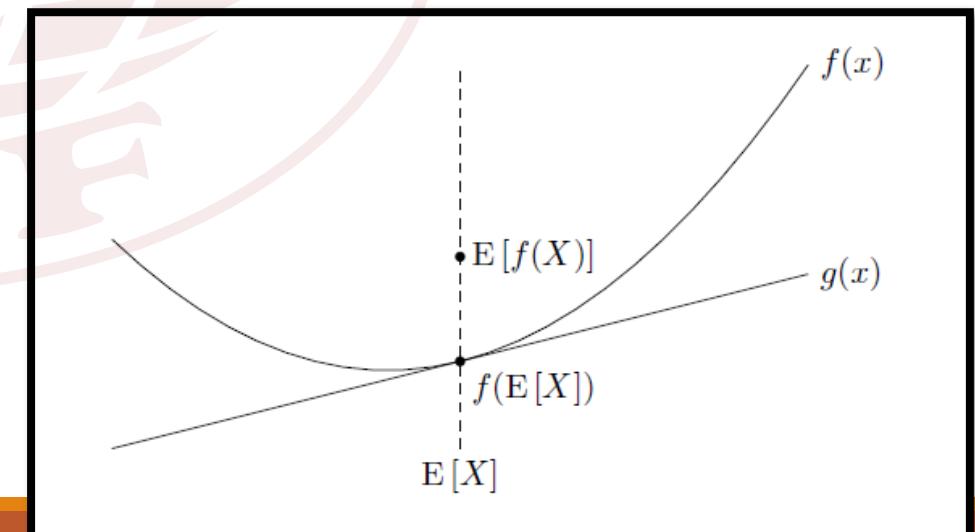


概率论

- 随机变量：定义在样本空间上面的实值函数。通常用大写字母表示，如 $X, Y, S\dots$
- 两个随机变量是独立的，当且仅当 $\Pr[X = x \cap Y = y] = \Pr[X = x] \cdot \Pr[Y = y]$
- 期望：对于一个随机变量 X , $E[X] = \sum_{\omega \in S} p(\omega)X(\omega)$
- 期望的线性性： $E[aX + bY] = aE[X] + bE[Y]$

概率论

- Union bound(Boole's inequality) : $\Pr[\cup A_i] \leq \sum \Pr[A_i]$
- Markov's inequality : $\Pr[X \geq \alpha] \leq \frac{E[X]}{\alpha}$
- Jensen's inequality : 如果 f 是一个下凸函数, 则 $f(E[X]) \leq E[f(X)]$





随机算法

- Las Vegas algorithm vs. Monto Carlo algorithm
- Las Vegas algorithm
 - 有一定概率失败，但是可以区分正确结果和错误结果
 - E.g. 找一个 $[l, r]$ 范围内的素数，可以随机选择数字，判断是否为素数
- Monto Carlo algorithm
 - 有一定概率失败，无法区分，重复多次运行程序以提高概率



Min-cut

给出一个无向图，定义图的一个割为，将点集分成两个集合之后，这两个集合之间的边的个数，求图的最小割。

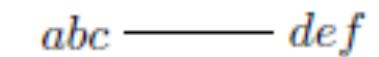
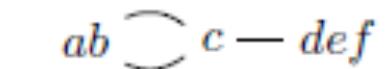
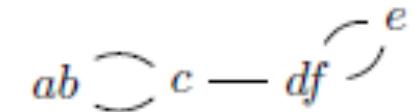
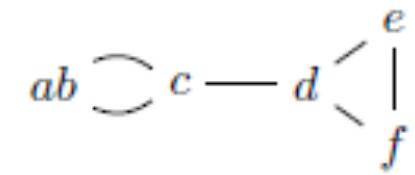
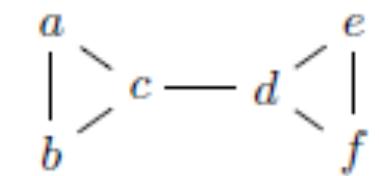


Karger's Min-cut Algorithm

等概率选择图中的一条边，将这条边两端的节点收缩成一个点

在得到的新图上继续执行以上过程，直到只剩下两个节点

将这两个节点之间的边的数量作为答案





Karger's Min-cut Algorithm

Karger's min-cut algorithm 有 $\frac{1}{C(n,2)}$ 的概率得到最小割。

证明？



Karger's Min-cut Algorithm

- 假设图G中存在一个大小为k的最小割，分成的两个点集分别为S和T，那么每个节点的度数都至少为k，G的边数至少为 $kn/2$ ，
- 随机选择一条边，选中最小割中的边的概率最多是 $k/(kn/2)=2/n$ ，没有选中最小割中的边的概率至少为 $1-2/n=(n-2)/n$
- 于是： $\prod_{i=3}^n \frac{i-2}{i} = \frac{2}{n(n-1)}$
- 证毕。



Karger's Min-cut Algorithm

故只需要将这个算法重复执行超过 n^2 次，就有很大概率得到最小割

这是一个Monte Carlo算法（无法验证结果正确性但是有一定概率正确）

当然，也可以使用最小割树来做这个题



中国计算机学会
China Computer Federation

最小圆覆盖

给出平面上n个点，要求求出一个最小的圆，使得所有点都在这个圆里面





最小圆覆盖

三点确定一个圆

枚举两个点作为圆上的两个点，然后枚举其他所有点，如果不在当前的圆内，那么将第三个点设置为这个点，最后判断得到的圆是否包含了所有点。

时间复杂 $O(n^3)$



最小圆覆盖

经典问题，随机增量法

先将所有点random shuffle

三点确定一个圆

记 O_i 为包含前*i*个点的最小的圆

对于*i*

- 如果点*i*在 O_{i-1} 里面，那么 $O_i = O_{i-1}$
- 否则，点*i*必定在 O_i 的边界上，在1..i-1中枚举另外两个点作为另外两个端点

时间复杂度 $\sum_i O(i^2) \times \frac{3}{i} = O(n^2)$

其中 $\frac{3}{i}$ 为第*i*个点在圆外的概率，即它是圆周上三个点之一的概率



最小圆覆盖

继续改进算法

对于i

- 如果点i在 O_{i-1} 里面，那么 $O_i = O_{i-1}$
- 否则，点i必定在 O_i 的边界上，在1..i-1中枚举一个点j，如果点j不在圆中，那么它一定要作为{1..j,i}的端点，在1..j-1中再枚举另外一点作为端点

如果点i不在 O_{i-1} 中，那么需要花费的时间为： $\sum_j O(j^2) \times \frac{2}{j} = O(i)$

于是总的时间复杂度为： $\sum_i O(i) \times \frac{3}{i} = O(n)$



最小圆覆盖

$$O(n^3) \Rightarrow O(n^2) \Rightarrow O(n)$$

随机增量法

- V图

避免输入的数据较差，避免worst cases



中国计算机学会
China Computer Federation

随机搜索树

Randomized Search Trees

Treap

每个节点有一个键值，它的键值小于等于右子树中所有节点的键值，大于等于左子树中所有节点的键值

每个节点有一个优先级，每个节点的优先级高于它的儿子的优先级。



Treap

插入节点的时候，首先定位节点，新建一个节点，给他一个随机的优先级

根据优先级进行旋转，调整至正确的结构。



Treap

假设所有键值为 $a[1..n]$

考虑键值为 $a[i]$ 的节点 x 以及键值为 $a[j]$ 的节点 y , 其中 $i < j$

如果 y 是 x 的祖先, 那么键值为 $a[i+1]..a[j-1]$ 的节点中对应的优先级都不可
以高于 y , 否则 y 不会成为 x 的祖先

即 $a[j]$ 对应节点的优先级是 $a[i+1..j]$ 中最高的

于是这样的概率是 $\frac{1}{j-i}$, 所以 x 的深度的期望为 $O(\log n)$

故而插入删除的期望复杂度都是 $O(\log n)$



【集训队作业2018】世界是个动物园

有一个竞赛图，对于节点*i*有 k_i 个不相交的区间

$[l_{i,1}, r_{i,1}], [l_{i,2}, r_{i,2}], \dots, [l_{i,k}, r_{i,k}]$

对于在区间中的点*j*，竞赛图中存在边(i,j)

对于不在区间中的点*j*，竞赛图中存在边(j,i)

对于*i=1..n*，求1..i的点组成的图中的强连通分量个数。

$$n \leq 2 \times 10^5$$



【集训队作业2018】世界是个动物园

竞赛图的性质：强连通分量形成一条链的结构

加入节点*i*时，假设有*k*个强连通分量，按顺序编号为1..*k*

$\forall (i, j) \in E$ ，找到*j*所在强连通分量编号最小的，记为*L*

$\forall (j, i) \in E$ ，找到*j*所在强连通分量编号最大的，记为*R*

如果 $L \leq R$ ，那么*i*以及*L..R*这些强连通分量将会合并成一个新的强连通分量

如果 $L > R$ ，那么 $L=R+1$ ，则*i*将在*R*和*L*之间插入，独自成为一个新的强连通分量

如何维护这个过程？

- 找到*L*以及*R*



【集训队作业2018】世界是个动物园

在加入新节点的过程中，对于所有节点维护一个权值 p ，使得，如果节点 x, y 分别属于强连通分量 $u, v (u < v)$ ，那么 $p[x] < p[y]$

查询 L 和 R 即变成了求区间最大值最小值

如何维护？

重量平衡树！



重量平衡树

给每个节点的权值 p ，类似于一般的treap中每个节点的键值，可以用于比较两个节点的排名

在treap中，给每个节点一个权值区间 $[l,r]$ ，将这个节点的权值定为 $p = \frac{l+r}{2}$ ，右子树的区间为 $[p,r]$ ，左子树的区间为 $[l,p]$

使用实数或大范围整数以保证精度。



重量平衡树

插入的时候，跟treap一样，给新建的节点一个随机的优先级，按照优先级进行旋转。

在重量平衡树中，旋转完之后，花费 $O(\text{子树大小})$ 的时间对整棵子树的权值进行重构

时间复杂度？



重量平衡树

一次旋转之后，这个新增节点能够成为子树根节点的概率为 $\frac{1}{size}$ ，其中 size 为这棵子树的大小

于是期望花费的时间为 $\frac{1}{size} \times O(size) = O(1)$

一次插入的均摊复杂度仍然为 $O(\log n)$



【集训队作业2018】世界是个动物园

使用重量平衡树以及线段树等数据结构，可以维护节点的权值，以及求区间最大最小值，时间复杂度 $O(n \log^2 n)$

注意到线段树中维护的是区间中的最大最小值，但是在加入节点*i*的时候， $p[1..i-1]$ 的相对大小是不会改变的，所以线段树修改的时候只需要修改包含节点*i*的区间

时间复杂度变为 $O(n \log n)$



中国计算机学会
China Computer Federation

MAX CUT

给出一个图，求图的最大割





MAX CUT

一个很naive的做法

对于每个节点，在{0,1}中随机，0表示它在集合S中，1表示它在集合T中

那么这个算法得到的割的期望大小为 $\frac{m}{2}$

得到一个大小为 $\frac{m}{2}$ 的割的概率至少为 $\frac{1}{m+1}$

证明？



MAX CUT

得到一个大小为 $\frac{m}{2}$ 的割的概率至少为 $\frac{1}{m+1}$

证明？

$$m/2 = E[X] = (1-p)E[X|X < m/2] + pE[X|X \geq m/2] \leq (1-p)\frac{m-1}{2} + pm$$

$$\text{解出 } p \geq \frac{1}{m+1}$$

所以只需要运行程序 $O(m)$ 次即可找到一个大小至少为 $\frac{m}{2}$ 的割

Eg. 所有图都存在一个大小至少为 $\frac{m}{2}$ 的割



MAX CUT

近似比 (approximation ratio): 一个算法的近似比为，其得到的解与最优解的比值

在MAX CUT中，我们得到了一个比值超过 $\frac{1}{2}$ 的算法，上述算法成为MAX CUT的一个 $\frac{1}{2}$ – *approximation* 算法

MAX CUT是NP-hard的



MAX SAT

有n个布尔变量以及m个式子

每个式子为一些变量或一些变量的否进行或运算的结果，如 $x_1 \text{ or } \neg x_3$

给每个变量赋值，使得尽量多的式子的值为真



中国计算机学会
China Computer Federation

MAX SAT

MAX SAT是NP-hard问题。





MAX SAT

类似MAX CUT中的算法，给每个变量一个{True , False}中的随机值

那么对于每条式子，如果式子长度为k，那么被满足的概率为 $1 - 2^{-k} \geq \frac{1}{2}$

直接给每个变量随机取值，得到一个 $1/2$ -approximation算法



MAX SAT

首先将原问题放宽成一个线性规划问题

用 C_1, C_2, \dots, C_m 表示这m条式子， z_i 表示第i条式子是否被满足， y_i 表示变量i的取值
则问题变成

最大化 $\sum z_i$

条件：

- $\forall j = 1..m, \sum_{i \in C_j^+} y_i + \sum_{i \in C_j^-} (1 - y_i) \geq z_j$
- $0 \leq y_i \leq 1, 0 \leq z_i \leq 1$

线性规划问题可以在多项式时间内解决，得到解 \hat{y}, \hat{z} ，记 $Z = \sum \hat{z}_i$

显然原问题的解不大于Z



MAX SAT

在MAX SAT的线性规划问题中，存在多项式算法求出解 \hat{y}, \hat{z}

对于每个变量 x_i ，令其有 \hat{y}_i 的概率值为真，有 \hat{y}_i 的概率值为假

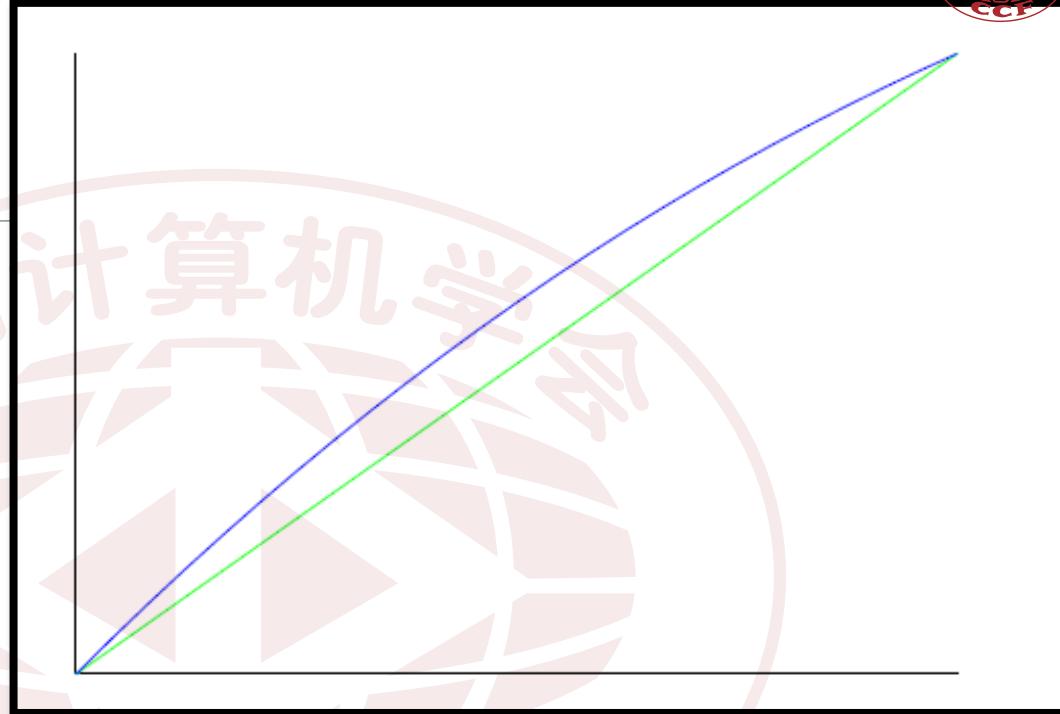
不失一般性的，假设式子 C_j 有 k 个变量， C_j 为 $x_1 \vee x_2 \vee \dots \vee x_k$

那么 C_j 为真的概率为： $1 - \prod_{i=1}^k (1 - \hat{y}_i)$

令 $z = \hat{z}_j$.

MAX SAT

$$\begin{aligned}
 \Pr [C_j \text{ is satisfied}] &= 1 - \prod_{i=1}^k (1 - \hat{y}_i) \\
 &\geq 1 - \prod_{i=1}^k (1 - z/k) \\
 &= 1 - (1 - z/k)^k \\
 &\geq z(1 - (1 - 1/k)^k). \\
 &\geq z(1 - 1/e).
 \end{aligned}$$



其中第一次放缩是由于 $\sum \hat{y}_i \geq z$, 故而取 $\hat{y}_i = \frac{z}{k}$, 可以得到

$$\prod_{i=1}^k (1 - \hat{y}_i)$$
 的一个上界是 $\left(1 - \frac{z}{k}\right)^k$

第二次放缩: 函数 $f(z) = 1 - \left(1 - \frac{z}{k}\right)^k$ 是上凸函数



MAX SAT

式子 C_j 为真的概率至少为 $\hat{z}_j \left(1 - \frac{1}{e}\right)$

那么 $E[\text{值为真的式子的数量}] = Z \left(1 - \frac{1}{e}\right) \approx 0.632$

于是该算法的近似比为0.632



MAX SAT

在MAX SAT问题中，前面谈到了两种算法
可以看到，前面那个较为弱的算法，适合用于处理式子长度较长的情况
而后者，使用线性规划的算法，适合用于式子长度较短的情况
那么有没有办法合并两个算法，得到一个更好的算法，达到更高的近似比呢？



MAX SAT

显然，这两个算法是不可以直接合并到一起的

但是，如果对于同样的输入，使用两个算法，都对这个输入跑一遍，将得到的两个结果取较大值，会得到一个近似比为 $3/4$ 的算法

证明？



MAX SAT

定义随机变量 X_j, Y_j 分别表示在两个算法中，式子 C_j 值为真的概率，那么：

$$\begin{aligned} E[X_j] + E[Y_j] &\geq (1 - 2^{-k}) + (1 - (1 - 1/k)^k) \hat{z}_j \\ &\geq ((1 - 2^{-k}) + (1 - (1 - 1/k)^k)) \hat{z}_j \\ &= (2 - 2^{-k} - (1 - 1/k)^k) \hat{z}_j. \end{aligned}$$

当 $k=1, k=2$ 时， $2 - 2^{-k} - \left(1 - \frac{1}{k}\right)^k = \frac{3}{2}$ ，而随着 k 增大，这个式子的值也跟着增大

于是： $E[\sum X_j] + E[\sum Y_j] \geq \frac{3}{2} Z$

那么两者中必有一个至少为 $\frac{3}{4} Z$

得到一个近似比为 $3/4$ 的算法



MAX SAT

算法1：随机得到每个变量取值，得到近似比为0.5

算法2：将integer programming转化成linear programming， 将LP中得到的解作为每个变量取值， 得到近似比为0.632

算法3：上诉两个算法取较大值， 得到近似比为0.75

Problem? 0.5和0.632如何得到0.75?

近似比的定义！



近似计算

有一类对象，我们希望知道他们有多少

1. 一个无向图的生成树个数
2. (01背包) 有 n 个物品和一个容量为 b 的背包，每个物品只有一个，问将物品放到背包里的方案数
3.



近似计算

Fully polynomial-time randomized approximation scheme(FPRAS)

对于一个数数问题，FPRAS会以不小于 $1 - \delta$ 的概率输出一个与答案 $answer$ 的相对误差不超过 ϵ 的数字 x ，即 $\left| \frac{x}{answer} - 1 \right| \leq \epsilon$ ，对应的时间复杂度为关于数据规模 n , $\frac{1}{\epsilon}$ 和 $\ln(\frac{2}{\delta})$ 的多项式



Monto Carlo Simulation

在集合U中有一个子集G，现在要求估计G的大小。

Monto Carlo Simulation

- 随机的从U中选择一个元素，判断是否在G中
- 随机n次，其中有m次，令 $|\hat{G}| = \frac{m}{n} \cdot |U|$ 为估计的结果。

误差？



Approximating #DNF

DNF计数

一条DNF公式是若干条布尔式进行或运算的结果，其中每条布尔式都是若干个布尔变量或它们的否进行并运算的结果

$$\text{Eg. } (x_1 \wedge \neg x_2) \vee (x_2 \wedge x_3)$$

问使得DNF公式为真的变量赋值的方案数



Approximating #DNF

Monto Carlo Simulation

给每个变量随机一个值，可以在多项式时间内判断是否合法

为了保证以不小于 $1 - \delta$ 的概率输出一个与答案相对误差不超过 ϵ 的答案
需要随机多少次(sampling)？



Approximating #DNF

记集合 U 为所有变量赋值的可能，即 $|U| = 2^n$

记集合 G 为使得DNF为真的变量赋值的集合，设 $\rho = \frac{|G|}{|U|}$

假设抽样了 N 次，第 i 次的结果为 Y_i （1表示在 G 中，否则不在），记 $Y = \sum_i Y_i$

那么 $E[Y] = \sum_i E[Y_i] = N \cdot \rho$

令 $|\check{G}| = \frac{Y}{N} \cdot |U|$



Approximating #DNF

$$\rho = \frac{|G|}{|U|}, \quad |\check{G}| = \frac{Y}{N} \cdot |U|$$

$$\Pr[(1 - \epsilon)|G| \leq |\check{G}| \leq (1 + \epsilon)|G|]$$

$$= \Pr[(1 - \epsilon)\rho|U| \leq \frac{|U|}{N} \cdot Y \leq (1 + \epsilon)\rho|U|]$$

$$= \Pr[(1 - \epsilon)\rho N \leq Y \leq (1 + \epsilon)\rho N]$$

$$= \Pr[|Y - \rho N| \leq \epsilon\rho N]$$



Approximating #DNF

根据Chernoff bound, $\Pr[|x - \mu| \geq \delta\mu] \leq 2e^{-\frac{\delta^2}{3}\cdot\mu}$

则 $\Pr[|Y - \rho N| \geq \epsilon\rho N] \leq 2e^{-\frac{\epsilon^2}{3}\rho N}$

则 $\delta \leq 2e^{-\frac{\epsilon^2}{3}\rho N}$, 令 $\delta = 2e^{-\frac{\epsilon^2}{3}\rho N}$

得到: $N = \frac{3}{\epsilon^2} \cdot \frac{\ln(\frac{2}{\delta})}{\rho}$

观察式子, 在前面的Monto Carlo Simulation中 $\frac{1}{\rho}$ 是指数级的



Approximating #DNF

令 $H_i = \{(v, i) | v \text{ satisfies } C_i\}$

令 $U = \bigcup_i H_i$, $G = \{(v, i) | (v, i) \in H_i \text{ and } \forall j < i, (v, j) \notin H_j\}$

那么 G 的大小即为前面的问题中，我们要求的集合大小



Approximating #DNF

这样选择的U和G, $\rho = \frac{|G|}{|U|}$ 如何?

对于所有G中的元素 (v, i) , v只可能在至多m个式子中出现

则 $\rho \geq \frac{1}{m}$, 即 $\frac{1}{\rho} \leq m$

所以需要收集的样本数量至少为: $N = \frac{3}{\epsilon^2} \cdot \frac{\ln\left(\frac{2}{\delta}\right)}{\rho} \leq \frac{3}{\epsilon^2} \cdot \ln\left(\frac{2}{\delta}\right) \cdot m$

为关于 $m, \ln\left(\frac{2}{\delta}\right), \frac{1}{\epsilon}$ 的多项式, 可以使用Monto Carlo Simulation



Approximating #KNAPSACK

有n个物品，他们的大小为 a_1, a_2, \dots, a_n 和一个大小为 b 的背包，满足 $0 \leq a_1 \leq a_2 \leq \dots \leq a_n \leq b$

有n个整数 x_1, x_2, \dots, x_n 表示是否选择第i个物品，取值为0或1

问有多少种给x赋值的方式使得 $\sum_i a_i x_i \leq b$

即求 $S = \{\vec{x} \mid \sum_{i=1}^n a_i x_i \leq b\}$ 的大小



Approximating #KNAPSACK

放缩：令 $a'_i = \lfloor \frac{n^2 a_i}{b} \rfloor$, 那么 $0 \leq a'_i \leq n^2$, $\left(\frac{b}{n^2}\right) a'_i \leq a_i < \left(\frac{b}{n^2}\right) a'_i + \left(\frac{b}{n^2}\right)$

令 $S' = \{\vec{x} \mid \sum_{i=1}^n a'_i x_i \leq n^2\}$

那么， $S \subseteq S'$

证明？



Approximating #KNAPSACK

$S \subseteq S'$

证明？

如果 $\sum_i a_i x_i \leq b$

$$\begin{aligned} \sum_{i=1}^n a'_i x_i &= \sum_{i=1}^n \lfloor n^2 a_i / b \rfloor x_i \\ &\leq \sum_{i=1}^n (n^2 / b) a_i x_i \\ &= (n^2 / b) \sum_{i=1}^n a_i x_i \\ &\leq n^2, \end{aligned}$$



Approximating #KNAPSACK

$S \subseteq S'$ 成立，而 $S' \subseteq S$ 不一定成立

但是，可以证明，对于 S' 中的元素 \vec{x} ，可以选择至多一个 $x_i = 1$ ，将其变为0之后就得到一个 S 中的元素 \vec{x}'

证明？



Approximating #KNAPSACK

证明？

对于 $\vec{x} \in S'$, 如果所有 $x_i = 1$ 的位置都满足 $a_i \leq \frac{b}{n}$, 那么 $\sum_i a_i x_i \leq b$, 即 $\vec{x} \in S$

否则, 找到一个位置 i 使得 $x_i = 1$ 且 $a_i > \frac{b}{n}$, 将 x_i 变成 0 得到 \vec{y} , 那么

$$\begin{aligned}\sum_{j=1}^n a_j y_j &= \sum_{j=1}^n a_j x_j - a_i \\&< \sum_{j=1}^n ((b/n^2)a'_j + b/n^2)x_j - b/n \\&\leq (b/n^2) \sum_{j=1}^n a'_j x_j + b/n - b/n \\&\leq (b/n^2)n^2 \\&= b.\end{aligned}$$



Approximating #KNAPSACK

对于 S' 中的元素 \vec{x} ，可以选择至多一个 $x_i = 1$ ，将其变为0之后就得到一个 S 中的元素 \vec{x}'

$$\text{于是, } \frac{|S|}{|S'|} \geq \frac{1}{n+1}$$



Approximating #KNAPSACK

计算 $|S'|$ 可以使用DP，记 $C(k, m) = |\{\vec{x} | \sum_{i=1}^k a'_i x_i \leq m\}|$

可以在多项式时间内求出C，那么 $|S'| = C(n, n^2)$

使用C来从 S' 中取样

由于 $\frac{|S|}{|S'|} \geq \frac{1}{n+1}$ ，那么以至少 $1 - \delta$ 的概率得到相对误差不超过 ϵ 的答案，
需要 $\frac{3(n+1)}{\epsilon^2} \ln \frac{2}{\delta}$ 个样本



Markov Chain

马尔科夫链和马尔科夫过程是一个随机过程，其中 X_{t+1} 的分布只与 X_t 的值有关，即：

$$\Pr [X_{t+1} = j \mid X_t = i_t, X_{t-1} = i_{t-1}, \dots, X_0 = i_0] = \Pr [X_{t+1} = j \mid X_t = i_t].$$

Eg. 随机游走

马尔科夫链的状态空间(state space)：随机变量 X_t 的所有取值

马尔科夫链是有限的：状态空间的大小是有限的

马尔科夫链是均匀的： $\Pr[X_{t+1} = j | X_t = i]$ 只与 i, j 有关而与 t 无关

马尔科夫链的转移矩阵： $P_{i,j} = \Pr[X_{t+1} = j | X_t = i]$ （每行的和都为1）



Markov Chain

平稳分布(Stationary distribution) : $\pi = \pi P$ (其中 $\sum_i \pi_i = 1$)

- 必定存在，因为 $P1 = 1$, P 的一个特征值为 1, 故而存在 $\pi = \pi P$
- 不唯一



Markov Chain

两个随机变量之间的距离(total variation distance): $d_{TV}(X, Y) = \sup_A (\Pr[X \in A] - \Pr[Y \in A])$

对于有限马尔科夫链，定义两个分布之间的距离为: $d_{TV}(x, y) = \max_A \sum_{i \in A} (x_i - y_i) = \max_A |\sum_{i \in A} (x_i - y_i)|$

实际上, $\|x - y\|_1 = \sum_i |x_i - y_i| = 2d_{TV}(x, y)$



Markov Chain

Mixing time

在一些拥有较好的性质的有限马尔科夫链上，对于初始的分布 x ，最终会收敛到某个平稳分布 π 上。

$$\forall \epsilon > 0, \exists t_{mix}(\epsilon) \text{ s.t. } \forall t \geq t_{mix}(\epsilon), d_{TV}(xP^t, \pi) \leq \epsilon$$



Markov Chain

Coupling Lemma

对于任意两个随机变量 X, Y , 有 $d_{TV}(X, Y) \leq \Pr[X \neq Y]$

不可约(irreducible): $\forall i, j, \exists t \text{ s.t. } p_{i,j}^t \neq 0$

- Strongly connected



Markov Chain

周期(period): 对于一个马尔科夫链，一个状态*i*的周期定义为 $m = \text{gcd}(\{t > 0 | p_{ii}^t \neq 0\})$ ；如果 $m=1$, 那么状态*i*是非周期性的(aperiodic)

一个马尔科夫链是非周期性的，当且仅当所有状态都是非周期性的

Lazy Markov chain

- $p'_{ij} = \frac{1}{2} p_{ij}, i \neq j$
- $p'_{ii} = \frac{1}{2} + \frac{1}{2} p_{ii}$



Markov Chain

Lemma 如果一个马尔科夫链的状态*i*是非周期性的，那么存在 t_0 使得 $\forall t \geq t_0, p_{ii}^t \neq 0$

Lemma 任意有限不可约的，非周期性的马尔科夫链都会收敛到一个唯一的平稳分布



中国计算机学会
China Computer Federation

Approximating #PERFECT-MATCHINGS

给出一个稠密二分图，求它的完美匹配的个数





Approximating #PERFECT-MATCHINGS

令 M_k 表示边数的k的匹配的集合，令 $m_k = |M_k|$

令 $r_i = \frac{m_i}{m_{i-1}}$, $r_1 = m_1$, 那么答案为 $r_1 \times r_2 \times r_3 \times \dots \times r_n$

考虑估计 r_i 的值 \hat{r}_i , 最后估计 $\hat{m}_n = \hat{r}_1 \times \hat{r}_2 \times \dots \times \hat{r}_n$

如何从 $M_i \cup M_{i-1}$ 中均匀取样?



Approximating #PERFECT-MATCHINGS

构造马尔科夫链

在 $M_i \cup M_{i-1}$ 中，任取一个状态 M ，有 $\frac{1}{2}$ 的概率留在 M ，有 $\frac{1}{2}$ 的概率随机选择一条边 $e = uv \in E$ ，然后

- 如果 $M \in M_i$ 而且 $e \in M$ ，那么转移到 $M' = M - e$
- 如果 $M \in M_{i-1}$ 并且 u, v 在 M 中都是没有被匹配的，那么转移到 $M' = M + e$
- 如果 $M \in M_{i-1}$ ， u 是与 w 匹配的而 v 尚未匹配，那么转移到 $M' = M + e - uw$
- 否则留在 M



Approximating #PERFECT-MATCHINGS

前面构造的马尔科夫链是非周期性的，不可约的

- 自环保证了非周期性
- 不可约可以构造

收敛到唯一的平稳分布 π ，其中 $\pi_i = \frac{1}{|M_i \cup M_{i-1}|}$ (why?)

Markov Chain Monto Carlo(MCMC) method

Approximating #PERFECT-MATCHINGS

Definition 5. Let π denote the stationary distribution of a Markov chain \mathcal{MC} . For a subset A of states of \mathcal{MC} , the *conductance* of A is defined as

$$\Phi(A) = \frac{\sum_{ij \in \delta(A)} \pi_i p(i, j)}{\sum_{i \in A} \pi_i},$$

where $\delta(A) = \{ij : i \in A, j \notin A\}$. Note that this is just the conditional probability that the stationary distribution *escapes* from A in a single step, given that it is initially in A . The conductance of \mathcal{MC} , $\Phi(\mathcal{MC})$, is defined as the minimum of conductances of subsets A with $0 < \sum_{i \in A} \pi_i \leq 1/2$.

Lemma 4.4.[10] *Let \mathcal{MC} be an ergodic symmetric Markov chain with s states and such that $p(i, i) \geq 1/2$ for all states i , and let $\epsilon \in (0, 1]$. Then the minimum t such that the distribution at time t is a near-uniform distribution with tolerance ϵ is at most $\frac{2}{\Phi(\mathcal{MC})^2} (\ln s + \ln \epsilon^{-1})$.*

Theorem 4.1. *If G is dense then $\Phi(\mathcal{MC}(G)) \geq 1/12n^6$.*



中国计算机学会
China Computer Federation

Approximating #PERFECT-MATCHINGS

实际上，这个算法可以在一些 $\frac{m_n}{m_{n-1}}$ 是 $O(\text{poly}(n))$ 的图上有较好的效果。





中国计算机学会
China Computer Federation

参考资料

1. Abbas Mehrabian , Approximation algorithms for counting the number of perfect matchings in bipartite graphs , university of Waterloo , 2010
2. James Aspnes , randomized algorithm , yale , 2019



中国计算机学会
China Computer Federation

END

