HIT CS&E

第2章　随机算法及其分类

骆吉洲
计算机科学与技术学院

HIT CS&E

提纲

HIT CS&E

2.1 概率基础

HIT CS&E

概率空间

样本空间Ω：所有基本事件（也称为样本）构成的集合

事件集合Σ：Ω的一个子集称为一个事件

(K1): $\varnothing, \Omega \in \Sigma$　　$\varnothing$-不可能事件,Ω-必然事件

(K2): ∪,∩,\下Σ封闭　　Σ是σ-代数

概率测度Pr: $\Sigma \rightarrow R$ 取非负值

(K3): $\Pr(\Omega)=1$

(K4): $A \cap B=\varnothing \Rightarrow \Pr(A \cup B)=\Pr(A)+\Pr(B)$

(K5*): $A_1 \supset A_2 \supset \ldots$ 且 $\cap_n A_n=\varnothing \Rightarrow \lim_{n\to\infty} \Pr(A_n)=0$

HIT CS&E

事件：

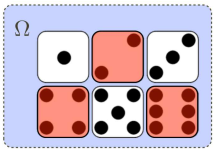
谓词$\mathcal{E}$

集合 $A=\{x \in \Omega | \mathcal{E}(x)\}$

概率：

$\Pr[\mathcal{E}] \quad = \quad \Pr[A]$

∨	∪
∧	∩
¬	Ω \
⇒	⊆



$\mathcal{E}$ = 掷出偶数点

$\Pr[\mathcal{E}]$

= Pr[⚁∨⚃∨⚅]

= Pr[⚁] + Pr[⚃] + Pr[⚅]

HIT CS&E

(K1): $\varnothing, \Omega \in \Sigma$

(K2): ∪,∩,\下Σ封闭

(K3): $\Pr(\Omega)=1$

(K4): $A \cap B=\varnothing \Rightarrow \Pr(A \cup B)=\Pr(A)+\Pr(B)$

$\boxed{\Pr[\neg\mathcal{E}] = 1- \Pr[\mathcal{E}]}$　　$\neg\mathcal{E} \vee \mathcal{E} = \Omega \Rightarrow \Pr[\neg\mathcal{E}] + \Pr[\mathcal{E}]=1$

$\boxed{\mathcal{E}_1 \Rightarrow \mathcal{E}_2,\text{则 } \Pr[\mathcal{E}_1] \le \Pr[\mathcal{E}_2]}$　　$\Pr[\mathcal{E}_2]=\Pr[\mathcal{E}_1]+\Pr[\mathcal{E}_2 \wedge \neg\mathcal{E}_1]$

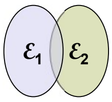
$\boxed{\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] - \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]}$

HIT CS&E

$\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] - \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$

$\Pr[\mathcal{E}_1] = \Pr[\mathcal{E}_1 \wedge \neg(\mathcal{E}_1 \wedge \mathcal{E}_2)] + \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$

$\Pr[\mathcal{E}_2] = \Pr[\mathcal{E}_2 \wedge \neg(\mathcal{E}_1 \wedge \mathcal{E}_2)] + \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$



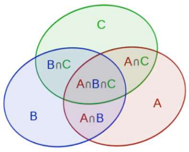
$\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] = \Pr[\mathcal{E}_1 \wedge \neg(\mathcal{E}_1 \wedge \mathcal{E}_2)] + \Pr[\mathcal{E}_2 \wedge \neg(\mathcal{E}_1 \wedge \mathcal{E}_2)] + \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$

将前两式带入第三式，消除无关项，即得结论

HIT CS&E

$\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] - \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$ 容斥原理

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |C \cap A| + |A \cap B \cap C|$$

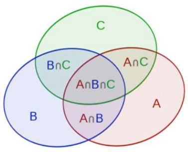


$$\left|\bigcup_{1\le i\le n} S_i\right| = \sum_{i=1}^{n}|S_i| - \sum_{A\in\binom{[n]}{2}}\left|\bigcap_{i\in A} S_i\right| + \sum_{A\in\binom{[n]}{3}}\left|\bigcap_{i\in A} S_i\right| - \sum_{A\in\binom{[n]}{4}}\left|\bigcap_{i\in A} S_i\right| \cdots + (-1)^{k-1}\sum_{A\in\binom{[n]}{k}}\left|\bigcap_{i\in A} S_i\right| + \cdots + (-1)^{n-1}\left|\bigcap_{i=1}^{n} S_i\right|.$$

HIT CS&E

$\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] - \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$ 容斥原理

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |C \cap A| + |A \cap B \cap C|$$



$$\Pr\left[\bigvee_{1\le i\le n}\mathcal{E}_i\right] = \sum_{i=1}^{n}\Pr[\mathcal{E}_i] - \sum_{A\in\binom{[n]}{2}}\Pr\left[\bigwedge_{i\in A}\mathcal{E}_i\right] + \sum_{A\in\binom{[n]}{3}}\Pr\left[\bigwedge_{i\in A}\mathcal{E}_i\right] - \sum_{A\in\binom{[n]}{4}}\Pr\left[\bigwedge_{i\in A}\mathcal{E}_i\right] \cdots + (-1)^{k-1}\sum_{A\in\binom{[n]}{k}}\Pr\left[\bigwedge_{i\in A}\mathcal{E}_i\right] + \cdots + (-1)^{n-1}\Pr\left[\bigwedge_{i=1}^{n}\mathcal{E}_i\right].$$

HIT CS&E

合并界限(The Union Bound)

$\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] - \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]$

$\Downarrow$

$\Pr[\mathcal{E}_1 \vee \mathcal{E}_2] \le \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2]$

$\Downarrow$ 数学归纳

$$\Pr\left[\bigvee_{1\le i\le n}\mathcal{E}_i\right] \le \sum_{i=1}^{n}\Pr[\mathcal{E}_i]$$

适用于任何依赖关系！

HIT CS&E

合并界限(续)

$$\Pr\left[\bigvee_{1\le i\le n}\mathcal{E}_i\right] \le \sum_{i=1}^{n}\Pr[\mathcal{E}_i]$$

$\Pr[GOOD_1] = 1 - \varepsilon_1$

$\Pr[GOOD_2] = 1 - \varepsilon_2$

~~$\Pr[GOOD_1 \wedge GOOD_2] =$~~

~~$\Pr[GOOD_1]\cdot\Pr[GOOD_2]$~~

二者可能不是独立的

$\Pr[GOOD_1 \wedge GOOD_2]$

$= 1 - \Pr[BAD_1 \vee BAD_2]$

$\ge 1 - (\Pr[BAD_1] + \Pr[BAD_2])$ 使用过程中涉及误差累积

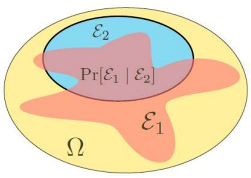
$= 1 - \varepsilon_1 - \varepsilon_2$

HIT CS&E

条件概率

定义. "在随机事件$\mathcal{E}_2$发生的条件下$\mathcal{E}_1$发生"的条件概率定义为

$$\Pr[\mathcal{E}_1 \mid \mathcal{E}_2] = \frac{\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]}{\Pr[\mathcal{E}_2]}$$



如果$\mathcal{E}_1$和$\mathcal{E}_2$是独立的

$$\Pr[\mathcal{E}_1 \mid \mathcal{E}_2] = \frac{\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]}{\Pr[\mathcal{E}_2]} = \frac{\Pr[\mathcal{E}_1]\Pr[\mathcal{E}_2]}{\Pr[\mathcal{E}_2]} = \Pr[\mathcal{E}_1]$$

条件概率链

$\Pr[\mathcal{E}_1 \mid \mathcal{E}_2] = \dfrac{\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2]}{\Pr[\mathcal{E}_2]} \Longrightarrow \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2] = \Pr[\mathcal{E}_1 \mid \mathcal{E}_2] \cdot \Pr[\mathcal{E}_2]$

$\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2 \wedge \ldots \wedge \mathcal{E}_n]$

$= \Pr[(\mathcal{E}_1 \wedge \mathcal{E}_2 \wedge \ldots \wedge \mathcal{E}_{n-1}) \wedge \mathcal{E}_n]$

$= \Pr[\mathcal{E}_n \mid \mathcal{E}_1 \wedge \mathcal{E}_2 \wedge \ldots \wedge \mathcal{E}_{n-1}] \cdot \Pr[\mathcal{E}_1 \wedge \mathcal{E}_2 \wedge \ldots \wedge \mathcal{E}_{n-1}]$

$= \ldots\ldots$ （在$\mathcal{E}_1 \wedge \mathcal{E}_2 \wedge \ldots \wedge \mathcal{E}_{n-1}$重复上述过程）

$= \Pr[\mathcal{E}_1] \cdot \Pr[\mathcal{E}_2 \mid \mathcal{E}_1] \cdot \Pr[\mathcal{E}_3 \mid \mathcal{E}_1 \wedge \mathcal{E}_2] \cdot \ldots \cdot \Pr[\mathcal{E}_n \mid \mathcal{E}_1 \wedge \ldots \wedge \mathcal{E}_{n-1}]$

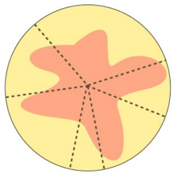
对任意$\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$有：$\Pr\left[\bigwedge_{i=1}^{n} \mathcal{E}_i\right] = \prod_{k=1}^{n} \Pr\left[\mathcal{E}_k \mid \bigwedge_{i<k} \mathcal{E}_i\right]$

全概率公式

若Ω被划分为$\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$（即$\mathcal{E}_i \wedge \mathcal{E}_j = \varnothing$, $\vee_i \mathcal{E}_i = \Omega$），则

$$\Pr[\mathcal{E}] = \sum_{i=1}^{n} \Pr[\mathcal{E} \wedge \mathcal{E}_i] = \sum_{i=1}^{n} \Pr[\mathcal{E} \mid \mathcal{E}_i] \cdot \Pr[\mathcal{E}_i]$$

对任意$\mathcal{E}$成立



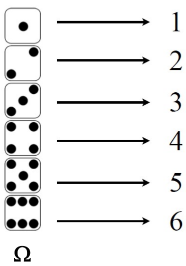
全概率公式常用于
分情况讨论事件的概率

随机变量

概率空间：$(\Omega, \Sigma, \Pr)$

随机变量X：一个函数$X: \Omega \to R$

投掷结果是一个随机变量



随机变量

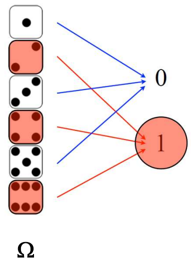
概率空间：$(\Omega, \Sigma, \Pr)$

随机变量X：一个函数$X: \Omega \to R$

投掷结果的奇偶是一个随机变量

事件“$X=x$”即“$\{s \in \Omega \mid X(s)=x\}$”

$\Pr[X=x] = \Pr[\{s \in \Omega \mid X(s)=x\}]$



随机变量的独立性

定义. 随机变量X和Y是独立的，如果

$\Pr[X=x \wedge Y=y] = \Pr[X=x] \cdot \Pr[Y=y]$

对任意x,y成立

随机变量$X_1, X_2, \ldots, X_n$是相互独立的，如果

$\Pr[\wedge_{i \in I} (X_i = x_i)] = \prod_{i \in I} \Pr[X_i = x_i]$

对任意$I \subseteq [n]$和任意x_i $(i \in I)$均成立

随机变量的数学期望

定义. 离散随机变量X的数学期望定义为

$E[X] = \sum_x x \cdot \Pr[X=x]$

其中x取遍X的值域

期望的线性性质

$$\mathrm{E}\left[\sum_{i=1}^{n} a_i X_i\right] = \sum_{i=1}^{n} a_i \cdot \mathrm{E}[X_i]$$

HIT CS&E

期望的线性性质

$$\mathrm{E}\left[\sum_{i=1}^{n} a_i X_i\right] = \sum_{i=1}^{n} a_i \cdot \mathrm{E}[X_i]$$

证明：

$$\mathrm{E}[X+Y] = \sum_x \sum_y (x+y)\Pr[X=x \wedge Y=y]$$

$$= \sum_x \sum_y x\Pr[X=x \wedge Y=y] + \sum_x \sum_y y\Pr[X=x \wedge Y=y]$$

$$= \sum_x x \sum_y \Pr[X=x \wedge Y=y] + \sum_y y \sum_x \Pr[X=x \wedge Y=y]$$

$$= \sum_x x\Pr[X=x] + \sum_y y\Pr[Y=y]$$

全概率公式

$$= \mathrm{E}[X] + \mathrm{E}[Y]$$

HIT CS&E

期望的线性性质

$$\mathrm{E}\left[\sum_{i=1}^{n} a_i X_i\right] = \sum_{i=1}^{n} a_i \cdot \mathrm{E}[X_i]$$

证明：

$$\mathrm{E}[X+Y] = \mathrm{E}[X] + \mathrm{E}[Y]$$

$$\mathrm{E}[cX] = \sum_x x\Pr[cX=x]$$

$$= c\sum_x \frac{x}{c}\Pr\left[X=\frac{x}{c}\right]$$

$$= c\sum_{x'} x'\Pr[X=x']$$

$$= c\mathrm{E}[X]$$

HIT CS&E

期望的线性性质

$$\mathrm{E}\left[\sum_{i=1}^{n} a_i X_i\right] = \sum_{i=1}^{n} a_i \cdot \mathrm{E}[X_i]$$

证明：

$$\mathrm{E}[X+Y] = \mathrm{E}[X] + \mathrm{E}[Y]$$

$$\mathrm{E}[cX] = c\mathrm{E}[X]$$

在此基础上，对n做数学归纳，得出结论

注：证明过程未涉及变量间是否独立
线性性质对任何依赖关系都成立

HIT CS&E

线性性质的应用实例

猴子在打字机上随机连续地敲出一个长度为10亿的字符串
这个字符串中“proof”平均出现多少次呢？

$X_i=1$ —— “proof”出现在位置i
$X_i=0$ —— “proof”未出现在位置i
$X=\sum_i X_i$ —— “proof”出现的总次数

$\Pr[X_i=1] = 1/26^5$
$\Pr[X_i=0] = 1-1/26^5$
$\mathrm{E}[X_i] = 1/26^5$

$$\mathrm{E}[X] = \mathrm{E}\left[\sum_{i=1}^{10^9-4} X_i\right] = \sum_{i=1}^{10^9-4} \mathrm{E}[X_i] = (10^9-4)\mathrm{E}[X_1] = (10^9-4)/26^5 \approx 84$$

HIT CS&E

Markov不等式

对于任意非负随机变量X,

$$\Pr[X\geq t] \leq \frac{\mathrm{E}[X]}{t}$$

对任意$t>0$成立

证明：

$$\Pr[X\geq t] = \int_t^{+\infty} \Pr[X=x]\mathrm{d}x$$

$$= \int_t^{+\infty} 1\cdot\Pr[X=x]\mathrm{d}x$$

$$\leq \int_t^{+\infty} \frac{x}{t}\cdot\Pr[X=x]\mathrm{d}x$$

$$= \frac{1}{t}\int_t^{+\infty} x\cdot\Pr[X=x]\mathrm{d}x$$

$$\leq \frac{1}{t}\int_0^{+\infty} x\cdot\Pr[X=x]\mathrm{d}x$$

$$= \frac{\mathrm{E}[X]}{t}$$

HIT CS&E

Markov不等式

对于任意非负随机变量X,

$$\Pr[X\geq t] \leq \frac{\mathrm{E}[X]}{t}$$

对任意$t>0$成立

推广

对任意随机变量X和非负函数$h: X\to R^+$

$$\Pr[h(X)\geq t] \leq \frac{\mathrm{E}[h(X)]}{t}$$

对任意$t>0$成立

HIT CS&E

方差

定义. 离散随机变量X的方差定义为

$\mathrm{Var}[X] = \mathrm{E}[(X-\mathrm{E}[X])^2]$

方差的算术平方根称为X的标准差，记为$\delta[X]$

$\delta[X] = (\mathrm{Var}[X])^{1/2}$

$$
\begin{aligned}
\mathrm{Var}[X] &= \mathrm{E}[(X-\mathrm{E}[X])^2] \\
&= \mathrm{E}[X^2-2X\cdot\mathrm{E}[X]+(\mathrm{E}[X])^2] \\
&= \mathrm{E}[X^2]-2\,\mathrm{E}[X\cdot\mathrm{E}[X]]+\mathrm{E}[(\mathrm{E}[X])^2] \\
&= \mathrm{E}[X^2]-2\,\mathrm{E}[X]\cdot\mathrm{E}[X]+(\mathrm{E}[X])^2 \\
&= \mathrm{E}[X^2]-(\mathrm{E}[X])^2
\end{aligned}
$$

HIT CS&E

二项分布的方差

定义.若随机变量X满足

$\Pr[X=1]=p$　　$\Pr[X=0]=1-p$

则称X服从两点分布

$\mathrm{E}[X] = 1\cdot\Pr[X=1]+0\cdot\Pr[X=0]=p$

$$
\begin{aligned}
\mathrm{Var}[X] &= \mathrm{E}[(X-\mathrm{E}(X))^2] \\
&= (1-p)^2\cdot\Pr[X=1]+(0-p)^2\cdot\Pr[X=0] \\
&= (1-p)^2\cdot p+p^2\cdot(1-p) \\
&= p(1-p)
\end{aligned}
$$

HIT CS&E

Chebyshev不等式

对任意随机变量X,

$$\Pr[\,|X-E[X]|\ge t\,]\le\frac{\mathrm{Var}[X]}{t^2}$$

对任意$t>0$成立

证明：令 $Y=(X-\mathrm{E}[X])^2$，Y是非负随机变量

$|X-\mathrm{E}[X]|\ge t \iff Y\ge t^2$

对Y和t^2运用Markov不等式即得结论

HIT CS&E

对任意随机变量X,

$$\Pr[\,|X-E[X]|\ge t\,]\le\frac{\mathrm{Var}[X]}{t^2}$$

对任意$t>0$成立

$t'=t\cdot\delta[X]$

$(\delta[X])^2=\mathrm{Var}[X]$

对任意随机变量X,

$$\Pr[\,|X-E[X]|\ge t\cdot\delta[X]\,]\le\frac{1}{t^2}$$

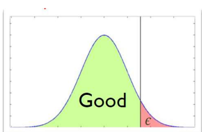
对任意$t>0$成立

HIT CS&E

尾概率界(Tail Bound)

Tail Bound:

$\Pr[X>t]<\varepsilon$



理想情况

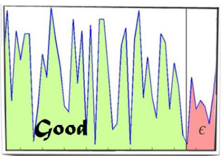
很多分析任务要求给出尾概率的界限

如：

（1）随机算法运行时间超过阈值的概率

（2）某种代价（如最大负载）

（3）极端情况出现的概率



糟糕情况

HIT CS&E

尾概率界(Tail Bound)

Tail Bound:

$\Pr[X>t]<\varepsilon$

将n个球放进n个箱子：

Pr[第一个箱子内球个数 > t]

$$
\begin{aligned}
&\le \binom{n}{t}\left(\frac{1}{n}\right)^t \\
&= \frac{n!}{t!(n-t)!n^t} \\
&= \frac{1}{t!}\cdot\frac{n(n-1)(n-2)\cdots(n-t+1)}{n^t} \\
&= \frac{1}{t!}\cdot\prod_{i=0}^{t-1}\left(1-\frac{i}{n}\right) \\
&\le \frac{1}{t!} \\
&\le \left(\frac{e}{t}\right)^t
\end{aligned}
$$

分析手段1:

计数并计算概率

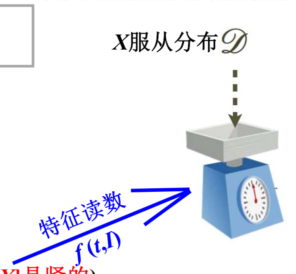
涉及大量计算

涉及计算方法的有效性

HIT CS&E

尾概率界(Tail Bound)

Tail Bound:
Pr[$X>t$] < ε



分析手段2:
将尾概率的计算归结为分析X某种特征I

Pr[$X>t$] < $f(t,I)$

分析工具:
Markov不等式(仅知$E[X]$是紧的)
Chebyshev不等式
Chernoff界 (待讲)
其他概率不等式 (待讲)

HIT CS&E

尾概率界(Tail Bound)

Tail Bound:
Pr[$X>t$] < ε

分析手段2:
将尾概率的计算归结为分析X某种特征I

将n个球放进n个箱子:
$X_i=1$ 第i个球落入第一个箱子
$X_i=0$ 第i个球未落入第一个箱子
$X=\sum_i X_i$ 第一个箱子中球总数

$\Pr[X_i=1]=1/n$ $\Pr[X_i=0]=1-1/n$
$E[X_i]=1/n$
$E[X]=\sum_i E[X_i]=1$

Pr[第一个箱子内球个数 > t]
$=\Pr[X>t]$
$\le E[X]/t$
$=1/t$

你试试Chebyshev不等式呢?

HIT CS&E

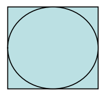
2.2 数值随机算法

- 计算π值
- 计算定积分

HIT CS&E

计算π值

- 数学基础
  - 设有一个半径为r的圆及其外切四边形



  - 向正方形随机地投掷n个点, 设k个点落入圆内
  - 投掷点落入圆内的概率为 $(\pi r^2)/(4r^2)=\pi/4$.
  - 用k/n逼近$\pi/4$, 即$k/n\approx\pi/4$, 于是$\pi\approx(4k)/n$.
  - 我们可以令$r=1$用投掷n个点的方法计算π

HIT CS&E

- 算法

1. $K=0$;
2. For i=1 To n Do
3. 随机地产生四边形中的一点(x, y);
4. If $x^2+y^2\le 1$ Then $k=k+1$;
5. Return $(4k)/n$

- 时间复杂性=$O(n)$
  - 不是输入的大小, 而是随机样本的大小
- 解的精确度
  - 随着随机样本大小n增加而增加

问题:样本数n和精度之间能建立关联关系吗?

HIT CS&E

计算定积分

- 问题
  - 计算积分 $\int_a^b g(x)dx$
- 数学基础
  - 令$f(x)$是区间$[a, b]$上的一组独立、同分布的随机变量$\{\xi_i\}$的任意密度函数
  - 令$g^*(x)=g(x)/f(x)$, 则$\{g^*(\xi)\}$是密度为$f(x)$的随机变量集合, 而且

$$E(g^*(\xi_i))=\int_a^b g^*(x)f(x)dx=\int_a^b g(x)dx=I$$

$$E(g^*(\xi_i)) = \int_a^b g^*(x) f(x)dx = \int_a^b g(x)dx = I$$

– 由强大数定律 $\Pr\left(\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n} g^*(\xi_i)=I\right)=1$

– 我们可以用 $\left(\frac{1}{n}\sum_{i=1}^{n} g^*(\xi_i)\right)$ 来近似计算I

– 令 $f(x)=1/(b-a)$ $a \le x \le b$

– 索求积分可以由如下I'来近似计算I

$$I' = \frac{1}{n}\sum_{i=1}^{n} g^*(\xi_i) = \frac{1}{n}\sum_{i=1}^{n} g(\xi_i)/f(\xi_i) = \frac{1}{n}\sum_{i=1}^{n}(b-a)g(\xi_i)$$

HIT CS&E

- 算法
  1. I=0;
  2. For i=1 To n
  3. 随机产生[a, b]中点x;
  4. I=I+$g(x)$;
  5. Return $(b-a)*I/n$
- 时间复杂性=$O(n)$
  - 不是输入的大小, 而是随机样本的大小
- 解的精确度
  - 随着随机样本大小n增加而增加

问题：样本数n和精度之间能建立关联关系吗？

HIT CS&E

2.3 随机选择与拉斯维加斯算法

- 问题的定义
- 随机算法
- 算法的性能分析
- Las Vegas算法

HIT CS&E

问题的定义

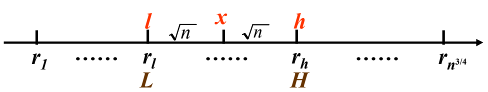
- 输入: $S=\{x_1, x_2, \ldots, x_n\}$,
  整数k, $1 \le k \le n$.
- 输出: S中第k个最小元素.

记号
  Rank(Q, t) =集合Q中的元素t的rank
  (第k小元素的rank是k)
  min(Q, i) = 集合Q中第i个最小元素.

HIT CS&E

随机算法

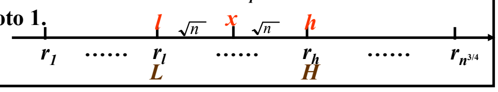
- 基本思想
  - 从S中随机地抽取$n^{3/4}$个样本存入R, 排序R
  - S中第k最小元素可能成为R中$x=kn^{3/4}/n$最小元素
  - 为了解决误差问题, 我们考察区间[$x-n^{1/2}$, $x+n^{1/2}$]



  - 把S中属于[L, H]数据存入P
  - 在P中查找$min(S, k)$

LAZYSELECT(S, k)

1. R=独立、均匀、可放回地从S随机选取的$n^{3/4}$元素;
2. 在$O(n)$时间内排序R;
3. $x=(k/n)n^{3/4}$; /* $(k/n)n^{3/4}=kn^{-1/4}$ */
4. $l=\max\{\lfloor x-\sqrt{n}\rfloor, 0\}$; $h=\min\{\lceil x+\sqrt{n}\rceil, n^{3/4}\}$;
5. $L=min(R, l)$; $H=min(R, h)$;
6. $L_p=Rank(S,L)$, $H_p=Rank(S,H)$; /*L和H与S元素比较*/
7. $P=\{y\in S \mid L\le y\le H\}$;
8. If $min(S, k)\in P$ and $|P|\le 4n^{3/4}+1$
   /* $max(S, k)\in P$可由$L_p\le k\le H_p$确定 */
9. Then 排序P, $min(S, k)=min(P, (k-L_p))$, 算法结束;
10. ELSE goto 1.



HIT CS&E

- 算法的性能分析

定理. 算法执行1-9步一遍就可求出$min(S,k)$的概率是$1-O(n^{-1/4})$,
即算法需要$O(n)$次比较就可求出$min(S,k)$的概率是$1-O(n^{-1/4})$.

证明. 若算法执行1-9一遍可求出$min(S, k)$, 则第6步需$2n$次比较, 其他步需$O(n)$次比较, 总需$O(n)$次比较.

往证算法执行1-9一遍可求出$min(S, k)$的概率是$1-O(n^{-1/4})$.

算法执行1-9一遍可求出$min(S, k)$的条件是:

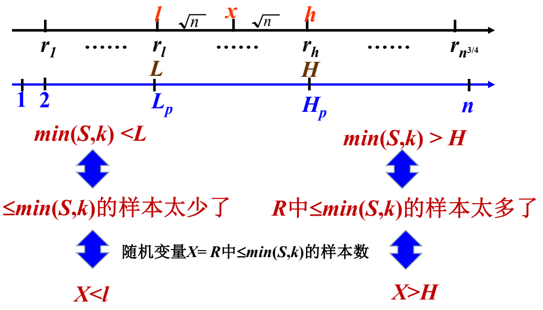
(1). $min(S, k)$在L和H之间即P包含$min(S, k)$,

(2). $|P|\le 4n^{3/4}+1$.

我们首先来计算上述两个条件失败的概率.

A. 计算条件(1)不成立的概率（计算思路）

条件(1)不成立 ⇔ $k<L_p$或$k>H_p$ ⇔ $L>min(S, k)$或$H<min(S, k)$



A. 计算条件(1)不成立的概率（形式化证明）

条件(1)不成立当且仅当$L>min(S, k)$或$H<min(S, k)$.

令$X_i=1$如果第i个随机样本$\le min(S, k)$, 否则$X_i=0$.

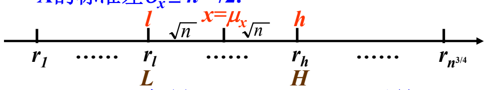
于是, $P(X_i=1)=k/n$, $P(X_i=0)=1-k/n$.

令 $X=\sum_{1\le i\le n^{3/4}} X_i$ 是R中小于等于$min(S, k)$ 的样本数. 我们有

X的数学期望$\mu_x=n^{3/4}k/n=kn^{-1/4}$,

X的方差$\sigma_x^2= n^{3/4}(k/n)(1-k/n) \le n^{3/4}/4$,

X的标准差$\sigma_x \le n^{3/8}/2$.



如果$L>min(S, k)$, $X<l$. 如果$H<min(S, k)$, $X\ge h$. 于是

$P(L>min(S, k)$或$H<min(S, k)))=P(X<l$ 或$X>h)=P(|X-\mu_x|>n^{1/2})$

$P(H<min(S, k))=P(X\ge h)=P(X>h)+P(X=h)=P(|X-\mu_x|\ge n^{1/2})+(n^{3/4}+1)^{-1}$.

应用Chebyshev不等式, 又由$2n^{1/8}\sigma_x \le n^{1/2}$, 我们有

$P(|X-\mu_x|>n^{1/2})\le P(|X-\mu_x|>2n^{1/8}\sigma_x)\le 1/(2n^{1/8})^2=O(n^{-1/4})$. 于是

$P(L>min(S, k))=P(H<min(S, k))=O(n^{-1/4})$.

B. 计算P包含$min(S, k)$但$|P|\le 4n^{3/4}+1$不成立的概率

令$k_l=\min\{0, k-2n^{3/4}\}$, $k_h=\max\{k+2n^{3/4}, n\}$.

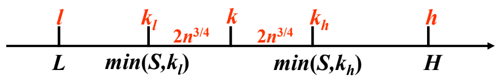
"P包含$min(S, k)$但$|P|\le 4n^{3/4}+1$不成立" 发生当且仅当 $L<min(S, k_l)$或$H>min(S, k_h)$.

类似于上面A中的分析,

$P(L<min(S, k_l))=P(H>min(S, k_h))=O(n^{-1/4})$.

由A和B, "算法执行1-9一遍就可以求出$min(S, k)$"不成立的概率是$O(n^{-1/4})$.

即, "算法执行1-9一遍就可以求出$min(S, k)$"的概率是$1-O(n^{-1/4})$.



HIT CS&E

Las Vegas算法

Las Vegas算法(随机算法类 LV Algorithm)

- 算法不会产生不正确的解
- 算法一旦得到问题的解，则该解是正确的
- 算法得到解的概率$p>0$
- 但算法运行过程可能不能产生问题的解
- 反复运行算法，运行时间不确定，最终可以得到问题的解
- 找到正确解需要运行算法的遍数与p相关(后续章节)
- 如：LazySelect算法1-9步可视为一个LV算法
  - ➢ 一旦找到返回解，该元素就是目标元素
  - ➢ 它运行一遍可能无法找到解
  - ➢ 反复运行，最终必然得到问题的解
- 一般用来刻画Yes-No型问题的随机算法

HIT CS&E

实验2

- 比较3种中位数选择算法的性能
  - 算法1： 排序后选择
  - 算法2： 确定型中位数线性时间选择
    《算法设计与分析》第3章
  - 算法3：中位数选择随机算法
- 实验内容
  - 实现3种算法
  - 数据集寻找或生成
  - 运行时间比较，比较准确度（如何衡量）？
  - 扩展性比较
  - 以恰当、准确、规范地表述实验结果

HIT CS&E

2.4 素数测试与蒙特卡罗算法

- 问题的定义
- 随机算法设计
- 算法的性能分析
- 蒙特卡罗算法
- 简单的概率放大
- 蒙特卡罗 Vs 拉斯维加斯

HIT CS&E

问题的定义

- 输入
  - 一个正整数N
- 输出
  - N是否素数

HIT CS&E

随机算法的设计

- 基本思想
  - 对N进行m次测试
  - 如果有一次测试成功, 则回答N是合数
  - 如果m次测试均失败, 则回答N是素数
  - 回答N是合数时, 答案百分之百正确
  - 回答N是素数时, 答案正确的概率是$1-2^{-m}$

- 随机算法

1. 随机地选择m个数$\{b_1, b_2, \ldots, b_m\}$, 满足 $1\leq b_1, b_2, \ldots, b_m \leq N$;
2. For i=1 To m Do
3. If $W(b_i)$成立 Then Return (N是合数);
4. Return (N是素数)

$W(b_i)$定义如下:

(1) $b_i^{N-1} \neq 1 \ mod\ N$, 或

(2) $\exists j[(N-1)/2^j=k$是整数, $1 < (b_i^k-1$与N的最大公因子$) < N]$.

HIT CS&E

例1. 给定N=12. 选择测试数集{2, 3, 7}

测试 2: $2^{12-1} = 2048 \neq 1 \mod 12$, $W(2)$成立.

N是合数.

HIT CS&E

例2. 给定N=11,选择测试数集{2, 5, 7}

测试 2: $2^{11-1} = 1024 = 1 \mod 11$,

测试 5: $5^{11-1} = 9765625 = 1 \mod 11$,

测试 7: $7^{11-1} = 282475249 = 1 \mod 11$,

结论: 11可能是素数

答案正确的概率为$1-2^{-3}$

HIT CS&E

算法性能的分析

定理1. (1) 如果对于任意$1\le b<N$, $W(b)$成立, 则N是合数.

(2) 如果N是合数, 则$(N-1)/2\le|\{b \mid 1\le b<N, W(b)\}|$

*(1)说明算法是正确的.

*(2)说明, 如果N是合数, 则至少一半$b(b<N)$使$W(b)$成立

定理2. 算法的回答"N是素数"正确的概率是$1-2^{-m}$.

HIT CS&E

蒙特卡罗算法

蒙特卡罗算法(随机算法类 Monte Carlo Algorithm)

- 用于刻画Yes-No型计算问题
- 运行时间是固定的
- 算法得到正确解的概率$p>0$
- 算法得到错误解的概率$1-p>0$
- 单面错误蒙特卡罗算法 (MC1算法)
  - ➢ 算法输出*Yes*-结论可靠
  - ➢ 算法输出No-结论可能是错的
- 如：N是合数吗？
  - ➢ Yes-可靠
  - ➢ No-发生错误的可能性不超过2^{-m}
- 双面错误蒙特卡罗算法 (MC2算法)
  - ➢ 算法输出*Yes*-结论可能是错的
  - ➢ 算法输出No-结论可能是错的

HIT CS&E

MC1算法的成功率放大

MC1算法的成功率放大

- 用于刻画Yes-No型计算问题
- 运行时间是固定的
  - ➢ Answer = Yes, 算法总输出*Yes*
  - ➢ Answer = No, Pr[算法输出No]≥ε
- 重复运行t次 $\quad t=O(\frac{1}{\epsilon}\log\frac{1}{\delta})$
  - ➢ 如果t次均输出Yes, 则最终输出Yes
  - ➢ 如果有一次输出no，则最终输出no
  - ➢ Pr[算法犯错] = Pr[answer =no但输出Yes]

    $\le(1-\varepsilon)^t$

    $\le\delta$
- 算法犯错的概率可以减小到任意指定的值δ
- 可以建立重复遍数t与ε, δ之间的关系

HIT

MC2算法的成功率放大

MC2算法的成功率放大

- 用于刻画Yes-No型计算问题
- 运行时间是固定的
  - ➢ Answer = Yes, Pr[算法输出*Yes*] ≥ 1/2+ε
  - ➢ Answer = No, Pr[算法输出No] ≥ 1/2+ε
- 重复运行t次 $\quad t=O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$
  - ➢ 输出占多数的答案

  Pr[恰有i次答案是正确的]=$\binom{t}{i}\left(\frac{1}{2}+\epsilon\right)^i\left(\frac{1}{2}-\epsilon\right)^{t-i}$

  Pr[算法最终犯错] = Pr[正确次数≤t/2]

$$=\sum_{i=0}^{\lfloor t/2\rfloor}\binom{t}{i}\left(\frac{1}{2}+\epsilon\right)^i\left(\frac{1}{2}-\epsilon\right)^{t-i}\le\sum_{i=0}^{\lfloor t/2\rfloor}\binom{t}{i}\left(\frac{1}{2}+\epsilon\right)^{t/2}\left(\frac{1}{2}-\epsilon\right)^{t/2}$$

$$=\left(\frac{1}{4}-\epsilon^2\right)^{t/2}\sum_{i=0}^{\lfloor t/2\rfloor}\binom{t}{i}\le\left(\frac{1}{4}-\epsilon^2\right)^{t/2}2^t=(1-4\epsilon^2)^{t/2}=\delta$$

HIT

MC2算法的成功率放大

- 用于刻画Yes-No型计算问题
- 运行时间是固定的
  - ➢ Answer = Yes, Pr[算法输出*Yes*] ≥ 1/2+ε
  - ➢ Answer = No, Pr[算法输出No] ≥ 1/2+ε
- 重复运行t次
  - ➢ 输出占多数的答案

  Pr[恰有i次答案是正确的]=$\binom{t}{i}\left(\frac{1}{2}+\epsilon\right)^i\left(\frac{1}{2}-\epsilon\right)^{t-i}$

  Pr[算法最终犯错] = Pr[正确次数≤t/2] ≤δ

  $t=O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$
- 算法犯错的概率可以减小到任意指定的值δ
- 可以建立重复遍数t与ε, δ之间的关系

HIT CS&E

蒙特卡罗 Vs 拉斯维加斯

两大类随机算法

Monte Carlo	Las Vegas
	
• 运行时间固定 • 是否正确是随机的	• 运行时间是随机的 • 得到的解是正确的 • 也可能得不到解

HIT CS&E

由Las Vegas算法构造MC1算法

Tail Bound:
Pr[$X>t$] < ε

算法$B(x)$
1. 调用$A(x)$运行$aT(n)$步
2. 若$A(x)$获得解，则返回该解
3. 否则返回0

单面错误

Las Vegas算法→蒙特卡罗算法
- A是一个Las Vegas算法
  - 最坏期望运行时间$T(n)$
  - 得到的解是正确解
- B是一个蒙特卡罗算法
  - 固定的运行时间$aT(n)$
  - 如果得到解，则解是正确的
  - 可能返回错误解

Pr[$B(x)$未获正确解]
$=\Pr[A(x)>aT(n)]$
$<\dfrac{A(x)\text{的期望运行时间}}{aT(n)}$
$=\dfrac{T(n)}{aT(n)}$
$=\dfrac{1}{a}$

HIT CS&E

2.5 随机排序与舍伍德算法

- 问题的定义
- 随机算法
- 算法性能的分析
- 舍伍德算法

HIT CS&E

问题的定义

- 输入
  - $S=\{x_1, x_2, \ldots, x_n\}$
- 输出
  - 排序的S

HIT CS&E

随机算法

- 基本思想
  - 采用随机抽样的方法确定集合的划分点
  - 把集合划分为两个子集合
  - 分别递归地在每个子集合上使用随机排序算法

HIT CS&E

- 算法
  1. 均匀等可能地在S中随机抽取一个样本y;
  2. $\forall x\in S$与y比较, 把S划分为如下两个集合:
     $S_1=\{ x \mid x\in S, x<y \}$, $S_2=\{ x \mid x\in S, x>y \}$;
  3. 递归地排序S_1和S_2;
  4. 顺序输出排序的S_1, y, S_2;

HIT CS&E

算法性能的分析

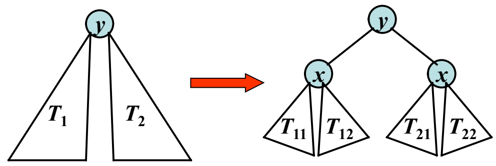
- 基本概念
  - $S_{(i)}$表示S中阶为i的元素
    例如，$S_{(1)}$和$S_{(n)}$分别是最小和最大元素
  - 随机变量X_{ij}定义如下:
    $X_{ij}=1$如果$S_{(i)}$和$S_{(j)}$在运行中被比较, 否则为0
  - X_{ij}是$S_{(i)}$和$S_{(j)}$的比较次数
  - 算法的比较次数为$\sum_{i=1}^{n}\sum_{j>i}X_{ij}$
  - 算法的平均复杂性为$E[\sum_{i=1}^{n}\sum_{j>i}X_{ij}]=\sum_{i=1}^{n}\sum_{j>i}E[X_{ij}]$

HIT CS&E

- 计算$E[X_{ij}]$
  - 设p_{ij}为$S_{(i)}$和$S_{(j)}$在运行中比较的概率, 则
    $E[X_{ij}]=p_{ij}\times 1+(1-p_{ij})\times 0=p_{ij}$

关键问题成为求解p_{ij}

- 求解P_{ij}
  - 我们可以用树表示算法的计算过程



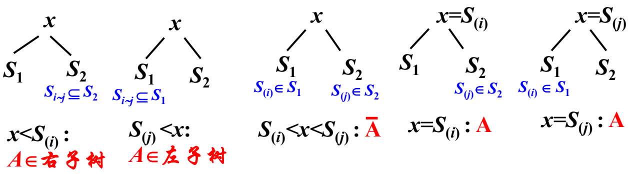
- 我们可以观察到如下事实:
  - 一个子树的根必须与其子树的所有节点比较
  - 不同子树中的节点不可能比较
  - 任意两个节点至多比较一次

HIT CS&E

计算p_{ij}

$S_{i\sim j}=\{S_{(i)}, S_{(i+1)}, S_{(i+2)},\ldots,S_{(j)}\}$

A: $X_{ij}=1$——$S_{(i)}$与$S_{(j)}$在算法运行过程中发生比较(随机事件)



B: 算法首次选中S_{ij}中的元素作为划分元素
可能在不同位置发生，但必然发生

$\Pr[A|B] = 2/(j-i+1)$

$\Pr[A] = \sum_\alpha \Pr[A|B_\alpha]\Pr[B_\alpha] = [2/(j-i+1)]\sum_\alpha \Pr[B_\alpha] = 2/(j-i+1)$

HIT CS&E

综上所述

$$E[T(n)] = \mathrm{E}\Big[\sum_{i=1}^{n}\sum_{j>i}X_{ij}\Big] = \sum_{i=1}^{n}\sum_{j>i}\mathrm{E}[X_{ij}] = \sum_{i=1}^{n}\sum_{j>i}p_{ij}$$

$$= \sum_{i=1}^{n}\sum_{j>i}\frac{2}{j-i+1} = \sum_{i=1}^{n}\sum_{k=2}^{n-i+1}\frac{2}{k} \le \sum_{i=1}^{n}\sum_{k=1}^{n}\frac{2}{k}$$

$$\le 2\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{1}{k} = 2n\sum_{k=1}^{n}\frac{1}{k}$$

$$=O(n\log n)$$

定理. 随机排序算法的期望时间复杂性为$O(n\log n)$

HIT CS&E

舍伍德算法

舍伍德算法(随机算法类 Sherwood Algorithm)

- 确定型算法的随机化
- 消除算法在最好实例和最坏实例间差别
- 如：QuickSort算法也可以确定性地选择划分元素
  - ➢ 最好时间复杂度为$O(n\log n)$
  - ➢ 最坏时间复杂度为$O(n^2)$
  - ➢ 随机选择划分元素后，期望时间复杂度为$O(n\log n)$
  - ➢ 先设计了确定型算法，随机化之后才得QuickSort
- 舍伍德算法总能得到问题的正确解

HIT CS&E

实验3

实验步骤

1.严格按照右框实现算法

2.生成11个大小为10^6的整数数据集
  第i个子集中存在元素重复$10^6\times 10\times i\%$
  $i=0,1,2,\ldots,10$

3.在各个实验数据集上运行算法
  观察实验现象

4.调用编程语言库函数中的快速排序算法
  在各个实验数据集上运行算法，观察现象

5. 解释实验现象发生的原因，改进算法及其实现

```
QuickSort (A, p, r)
    if p < r
        q = Rand_Partition (A, p, r)
        QuickSort (A, p, q-1)
        QuickSort (A, q+1, r)
Rand_Partition (A, p, r)
    i = Random (p, r)
    exchange A[r] with A[i]
    x = A[r]
    i = p – 1
    for j = p to r – 1
        if A[j] <= x
        i = i + 1
        exchange A[i] with A[j]
    exchange A[i+1] with A[r]
    return i + 1
```

HIT CS&E

2.6 最小割与概率放大技术

- 问题定义
- 随机算法
- 算法性能的分析
- 成功概率放大技术

HIT CS&E

问题定义

- 输入:
  - 无向多重连通图G
- 输出
  - 一个Min-Cut

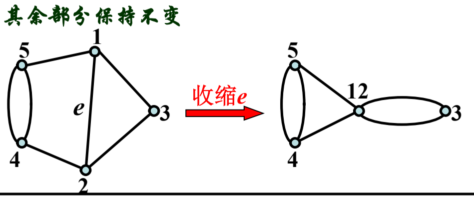
- 图G的一个Cut是一组边, 从G中删除这个Cut将导致两个或多个连通分量
- Cut的大小是其边数, 多重边重复计算
- 最小Cut是具有最少边的Cut

HIT CS&E

随机算法

- 基本概念
  - Cut可以视为节点集的划分$V=(C, V-C)$, Cut是所有G中连接C和$V-C$的边集合.
  - 图G的边(x, y)的contraction:
    - 用新节点代替节点x和y或边(x, y),
    - $\forall v \in V$, 用边(v, z)代替边(x, v)或(y, v),
    - G的其余部分保持不变

例如:



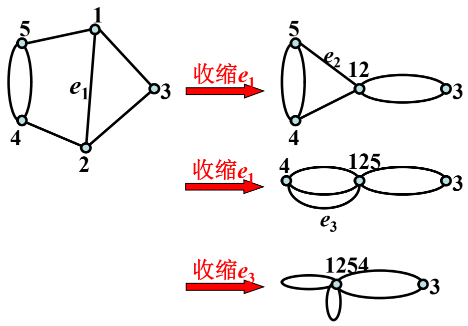
HIT CS&E

- 我们用$G/(x, y)$表示G的边(x, y)的收缩
- 边集合$F \subseteq G$收缩记作G/F
- 图G/F的节点集合表示为V/F
- 图G/F的节点集合表示为E/F

HIT CS&E

- 随机算法CONTRACTION
  1. $H=G$;
  2. While $|H(V)| > 2$ Do
  3. 随机地从$H(E)$中选择一条边(x, y);
  4. $F=F \cup \{(x, y)\}$;
  5. $H=H/(x, y)$;
  6. Cut=连接H中两个元节点的G的所有边.

例



Cut={(1,3), (2, 3)}

HIT CS&E

算法的性能分析

定理1. 如果算法的输入是具有n个节点的多重图, 则算法的时间复杂性为$O(n^2)$.

证明. 一次边收缩需要$O(n)$时间.

至多进行$O(n)$次收缩.

于是, 算法时间复杂性为$O(n^2)$.

注意:

我们仅证明了在$O(n^2)$时间内算法能够求出一个*Cut*, 但是这个Cut不一定是优化的.

HIT CS&E

引理1. 如果k是min-cut的大小, 则G至少有$kn/2$条边.

证. 如果$|G(E)|<kn/2$, 则存在一个度小于k的节点p.

删除与p相关连的k条, 把G划分为两个连通分量, 其一是仅包含p.

于是, 与p相关连的边集合是一个cut.

但是这个cut的大小$<k$, 与min-cut大小为k矛盾.

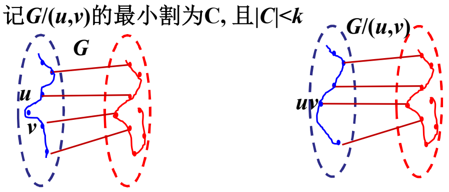
引理2. 算法输出的cut是连接两个剩余节点的没有被收缩过的边.

证. 从算法定义可以看到, 算法输出的cut是连接两个剩余节点的没有被收缩的边的集合.

HIT CS&E

引理3. 设图G的min-cut的大小为k, 则$G/(u,v)$是G收缩边(u,v)后得到的图。如果(u,v)不是最小割中的边，则$G/(u,v)$的最小割至少为k.

证. 反证法. 记$G/(u,v)$的最小割为C, 且$|C|<k$



从$G/(u,v)$删除C中的边, 得到两个顶点子集V_1,V_2, $uv\in V_2$

从G删除C中的边, 得顶点子集V_1=,$V_3=V_2-\{uv\}\cup\{u,v\}$

C也是G的割，且$|C|<k$。矛盾

定理2. 设C是一个min-cut, 其大小为k. 在算法结束时, C中无边被收缩过的概率大于$2/n^2$.

证. A_i表示第i步没有选中C的边, $1\leq i\leq n-2$.

在第1步, 选中的边在C中的概率至多为$k/(kn/2)=2/n$, 即

$Pr(A_1)\geq 1-2/n$.

在第2步, 若A_1发生, 则至少有$k(n-1)/2$条边(每次收缩减少一个节点), 选中C中边的概率为$2/(n-1)$, 即

$Pr(A_2|A_1)\geq 1-2/(n-1)$.

在第i步, 若A_1至A_{i-1}发生, 则有$n-i+1$个节点, 即至少有$k(n-i+1)/2$条边, 于是

$Pr(A_i|\cap_{1\leq j\leq i-1}A_j)\geq 1-2/(n-i+1)$

最后我们有

$Pr(\cap_{1\leq i\leq n-2}A_i)\geq \Pi_{1\leq i\leq n-2}(1-2/(n-i+1))=2/n(n-1)>2/n^2$

HIT CS&E

放大成功率：简单重复

算法Amplify

1. $S=E$
2. For i=1 to $n^2/2$ Do
3. S_i = CONTRACTION(G);
4. If $|S|>|S_i|$ Then $S=S_i$;
5. Return S

推论1. 算法Amplify的运行时间为$O(n^4)$，它不能发现一个min-cut的概率为

$$\left(1-\frac{2}{n^2}\right)^{n^2/2}<\frac{1}{e}$$

HIT CS&E

算法Amplify

1. $S=E$
2. For i=1 to $n^2/2$ Do
3. S_i = CONTRACTION(G);
4. If $|S|>|S_i|$ Then $S=S_i$;
5. Return S

这有意义吗？

最小割问题的精确算法的时间复杂度是多少？ $O(n^3)$

造成这种现象的原因是什么呢？ 获得正确解的概率太低

怎么才能提高获得正确解的概率呢？

HIT CS&E

成功率低下的原因

严格单调递减

Pr[CONTRACT成功]≥ $\frac{n-2}{n}\ \frac{n-3}{n-1}\ \frac{n-4}{n-2}\cdots\frac{3}{5}\ \frac{2}{4}\ \frac{1}{3}\ \geq \frac{2}{n^2}$

最后一次收缩未选中最小割中的边的概率

……

第3次收缩未选中最小割中的边的概率

第2次收缩未选中最小割中的边的概率

第1次收缩未选中最小割中的边的概率

HIT CS&E

成功率放大：重复关键操作

图的规模较小
未选中最小割中边概率较小

调用精确算法

Pr[CONTRACT成功]≥ $\frac{n-2}{n}\ \frac{n-3}{n-1}\ \frac{n-4}{n-2}\cdots\frac{3}{5}\ \frac{2}{4}\ \frac{1}{3}\ \geq \frac{2}{n^2}$

随机收缩

图的规模较大
不会选中最小割中的边的概率较大

HIT CS&E

算法DetRan

1. While $|V| > d(n)$ Do　　　　//$d(n)=n^{2/3}$
2. 　　随机选择一条边进行收缩；
3. 调用精确算法求得最小割S；
5. Return S

第2步每次时间复杂度为$O(n)$　　　　执行n-$d(n)$遍

第3步每次时间复杂度为$O(d^3(n))$

总时间为$O(n^2)+O(d^3(n))$

结论：取$d(n)= n^{2/3}$, 算法DetRan的时间复杂度为$O(n^2)$

HIT CS&E

算法DetRan

1. While $|V| > d(n)$ Do
2. 　　随机选择一条边进行收缩；
3. 调用精确算法求得最小割S；
5. Return S

$$\Pr[\text{DetRan获得最小割}]\geq \frac{n-2}{n}\ \frac{n-3}{n-1}\ \frac{n-4}{n-2}\cdots\frac{d(n)+1}{d(n)+3}\ \frac{d(n)}{d(n)+2}\ \frac{d(n)-1}{d(n)+1}$$

$$=\frac{d(n)}{n}\ \frac{d(n)-1}{n-1}$$

$$\approx\frac{n^{4/3}}{n^2}\qquad d(n)=n^{2/3}$$

$$=\frac{1}{n^{2/3}}$$

HIT CS&E

算法Amplify2

1. $S=E$
2. For i=1 to $n^{2/3}$ Do
3. 　S_i = DetRan(G);
4. 　If $|S|>|S_i|$ Then $S= S_i$;
5. Return S

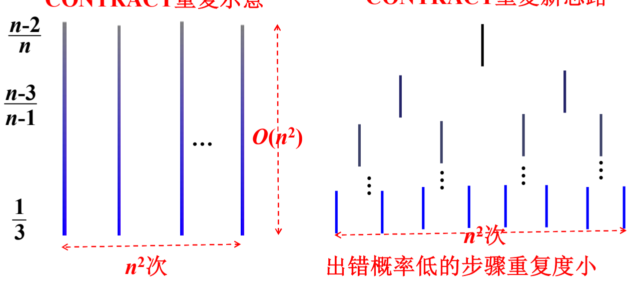
$$\Pr[\text{Amplify2未得最小割}]\leq \left[1-\frac{1}{n^{2/3}}\right]^{n^{2/3}}\approx e^{-1}$$

结论：独立运行DetRan算法$n^{2/3}$遍, 时间复杂度为$O(n^{8/3})$
找到最小割的概率至少为1-e^{-1}

HIT CS&E

成功率放大：关键操作重复策略

CONTRACT重复示意　　　　CONTRACT重复新思路



整体时间复杂度$O(n^4)$

出错概率低的步骤重复度小
出错概率高的步骤重复度大

怎么才能实现这种想法呢？

算法RepTree(G)　　　　//顶点个数记为n

1. If $n\leq 6$ Then 用确定型算法求最小割S,返回
2. $h=\lceil n-n/2^{1/2}\rceil$;
3. 随机独立收缩G中n-h条边得图G_1;　　$O(n^2)$
4. 随机独立收缩G中n-h条边得图G_2;　　$O(n^2)$
5. S_1= RepTree(G_1);
6. S_2 = RepTree(G_2);
7. Return $min(S_1, S_2)$;

$T(n)= 2T(n/2^{1/2})+O(n^2)$

$T(n)= O(n^2\log n)$

算法RepTree(G)　　　　//顶点个数记为n

1. If $n\leq 6$ Then 用确定型算法求最小割S,返回
2. $h=\lceil n-n/2^{1/2}\rceil$;
3. 随机独立收缩G中n-h条边得图G_1;
4. 随机独立收缩G中n-h条边得图G_2;
5. S_1= RepTree(G_1);
6. S_2 = RepTree(G_2);
7. Return $min(S_1, S_2)$;

$$\Pr[G_1\text{仍含最小割}]\geq \frac{n-2}{n}\,\frac{n-3}{n-1}\,\frac{n-4}{n-2}\cdots\frac{h+1}{h+3}\,\frac{h}{h+2}\,\frac{h-1}{h+1}=\frac{h(h-1)}{n(n-1)}\geq 1/2$$

HIT CS&E

RepTree获得正确解的概率

Pr(n) = 算法在规模为n的图上获得正确解的概率

Pr($n/2^{1/2}$) = 算法在G_1上获得正确解的概率| G_1含最小割

Pr($n/2^{1/2}$) = 算法在G_2上获得正确解的概率| G_2含最小割

Pr[G_1,G_2均不包含最小割] ≤ (1-1/2)(1-1/2)= 1/4

Pr[算法找不到正确解|G_1,G_2含最小割]
≤Pr[未找到正确解|G_1含最小割]· Pr[未找到正确解|G_2含最小割]
=[1-Pr($n/2^{1/2}$)]^2

Pr(n) = 1- Pr[算法无法获得正确解的概率]
≥3/4- [1-Pr($n/2^{1/2}$)]^2

解得： Pr(n)= Ω(1/ log$n$)

HIT CS&E

结论

结论：算法RepTree的时间复杂度为$O(n^2\log n)$
找到最小割的概率至少为Ω(1/ log$n$)

重复运行RepTree算法log$n$遍的时间开销为$O(n^2\log^2 n)$，找到正确解的概率为$1-e^{-1}$

重复运行RepTree算法$\log^2 n$遍的时间开销为$O(n^2\log^3 n)$，找到正确解的概率接近于1