《计算机模拟》

第7讲 – 马尔可夫链蒙特卡洛方法

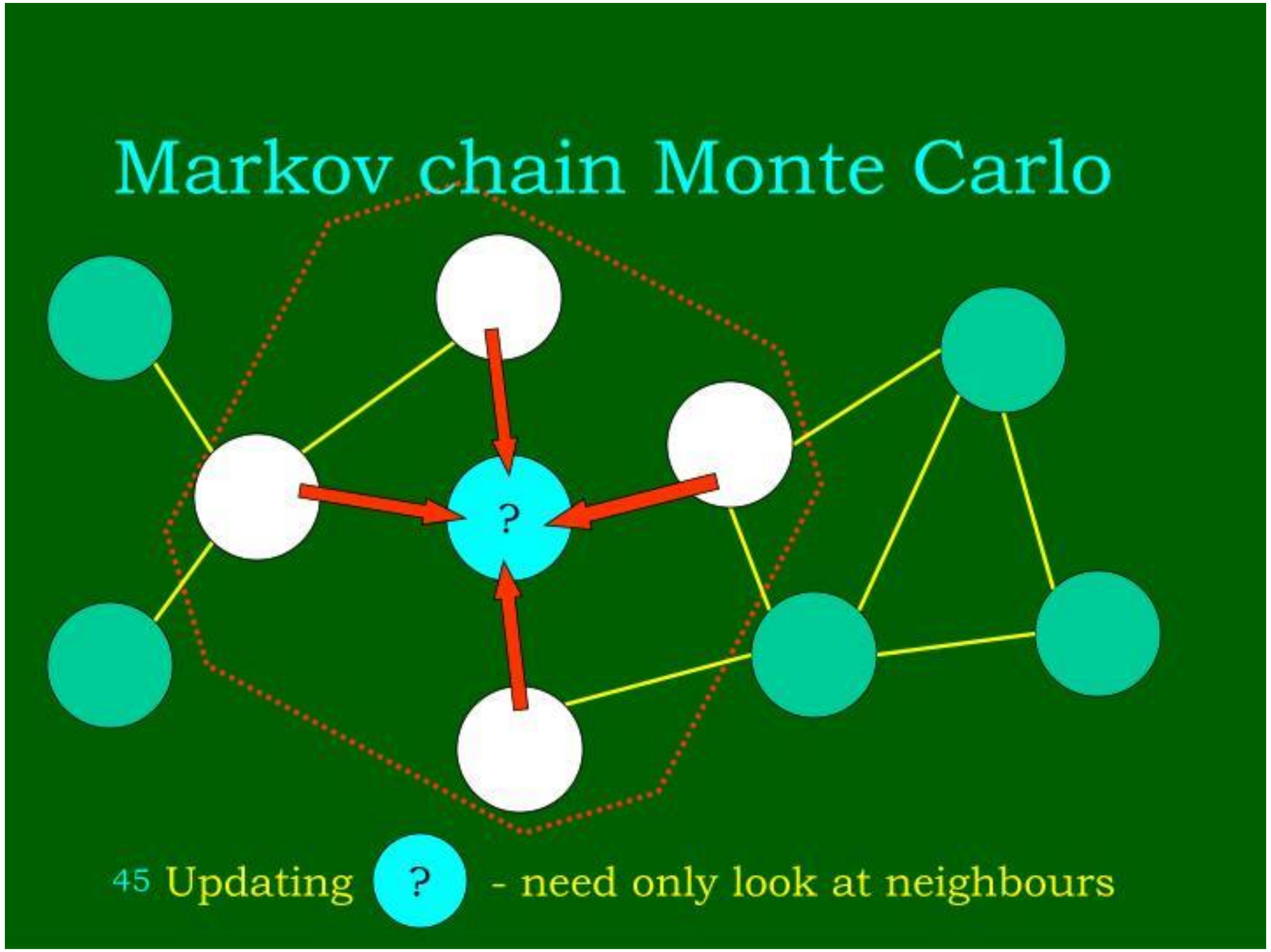
胡贤良
浙江大学数学科学学院

本讲内容

1. Markov Chain 简介

2. Markov Chain Monte Carlo：

   Metropolis-Hastings算法

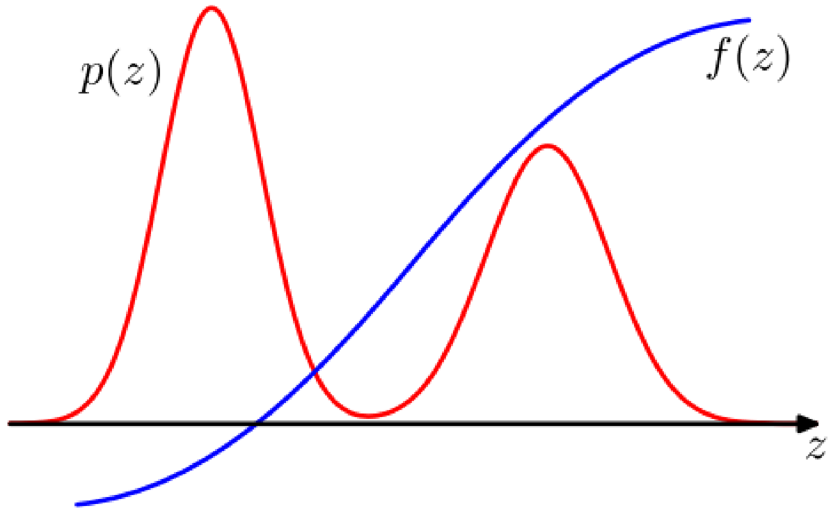


简单Monte Carlo方法局限性

- MC方法的核心：构造较容易采到的分布 $p(x)$的数据样本，即**抽样**
- 考虑计算随机变量的期望值。若随机数x_i的生成服从概率分布 $p(x)$，

则计算 f(x) 的期望：

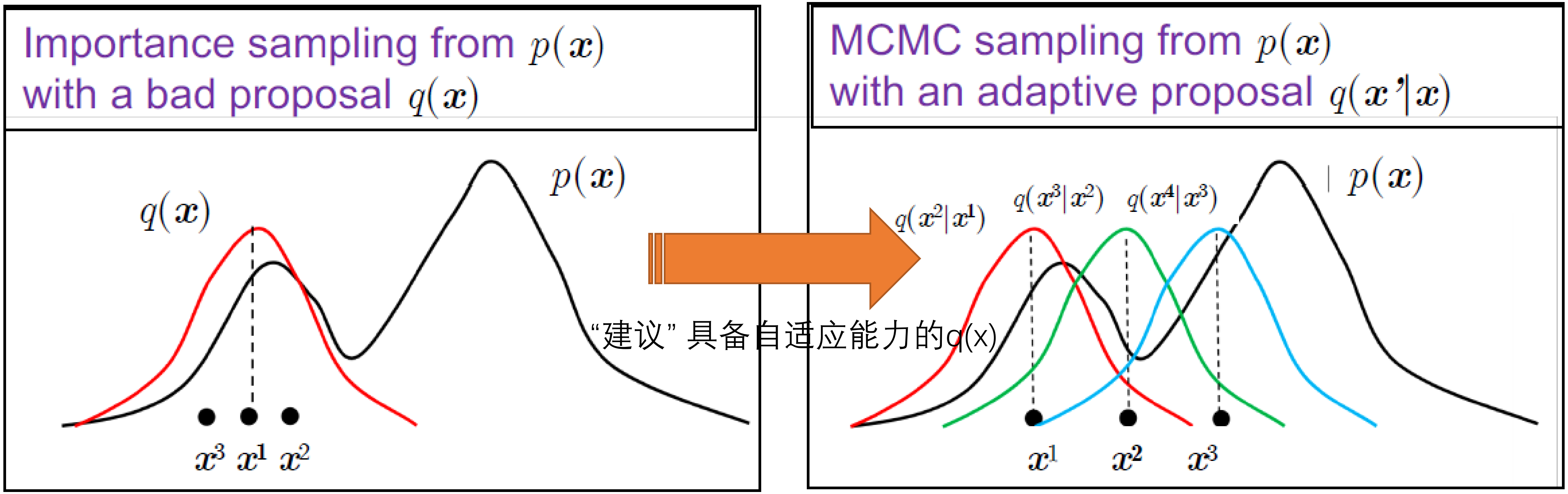
$$E_N(f) = \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$



- 直接抽样：高维Rare events抽样效率太低！
- 拒绝-接受抽样与重要性抽样：
  - 如果建议概率密度函数q(x)与真实的概率密度p(x)差别较大，效率也很低
  - 另一方面，构造一个好的q(x)是困难的！

有没有一种能动态自适应(adaptive)的方法，

能给出形式固定的q(x) ??

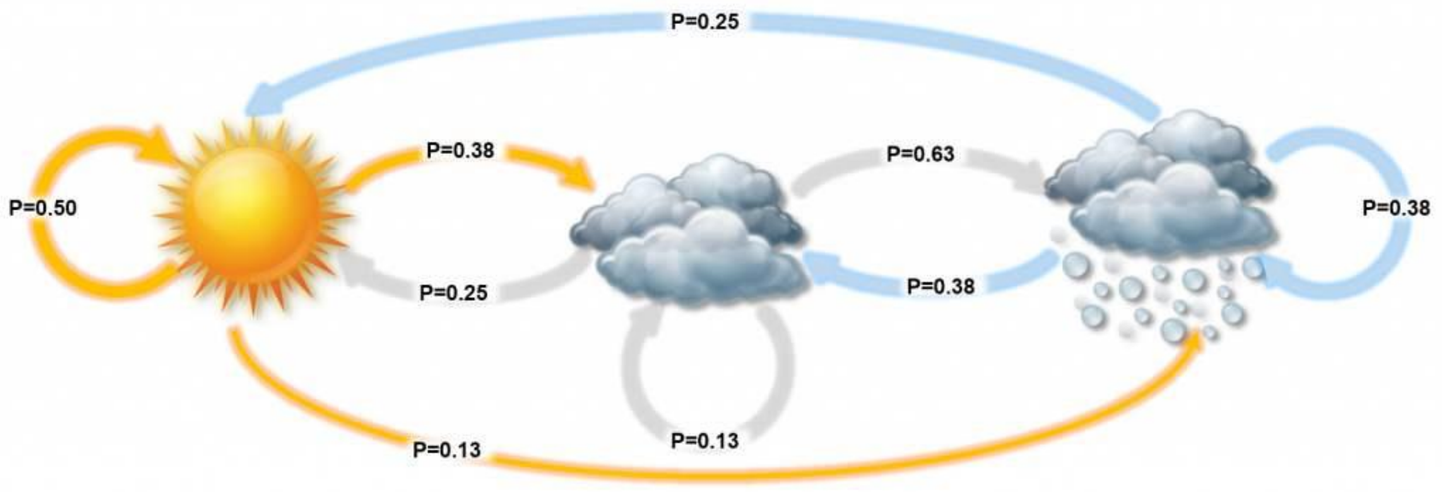


该图摘录自：Sargur Srihari《Markov Chain Monte Carlo Sampling》

Markov过程

- 未来的演变不依赖于其以往的演变，考虑随机过程$X = \{X_t, t \in R\}$为例：

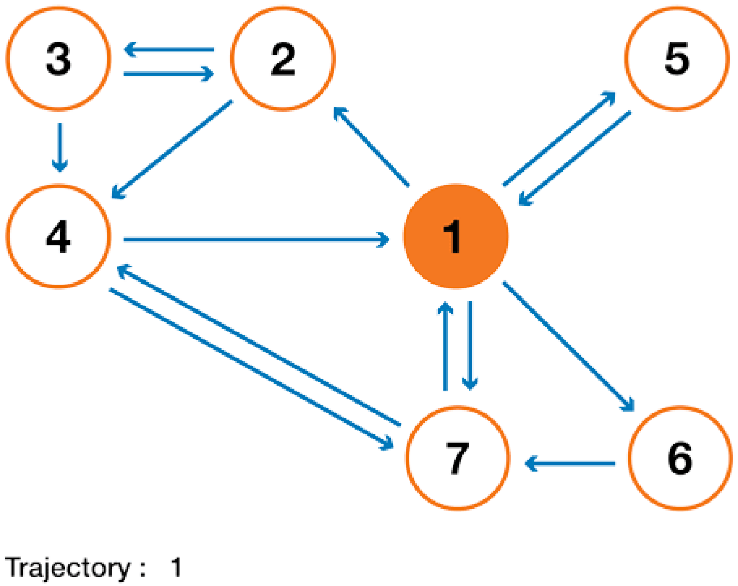
$$P(X_{t+1} = x_{t+1} \mid X_0 = x_0, \cdots, X_t = x_t) = P(X_{t+1} = x_{t+1} \mid X_t = x_t)$$



Markov Chain (马尔可夫 链)

将多个试验结果按时间标记为一系列“前后相继” (随机过程)的**状态**:

$$x^{(0)}, x^{(1)}, x^{(2)}, \cdots, x^{(n)}, x^{(n+1)}, \cdots$$



马尔可夫性(**无记忆性**): 下一状态概率分布**只能由**当前状态决定，在时间序列中它前面的事件均与之无关!

- **离散时间**马尔可夫链(**discrete-time** Markov chain): 它描述了从状态x_i到状态x_j的转换的**随机过程**。即假设存在一个独立于时间的固定概率p_{ij}，使得

$$P\left(x^{(n+1)} = i \,\middle|\, x^{(n)} = j, x^{(n-1)} = i_{n-1}, \cdots, x^{(0)} = i_0\right) = p_{ij}, n \geq 0.$$

转移概率矩阵

- 考虑一个特殊的情形：齐次Markov链，概率p_{ij}定义为：

$$p_{ij} \coloneqq P(X_{t+1} = j \mid X_t = i), \qquad i,j \in N$$

表示从当前状态i转移到状态j的概率，p_{ij}与时间无关！

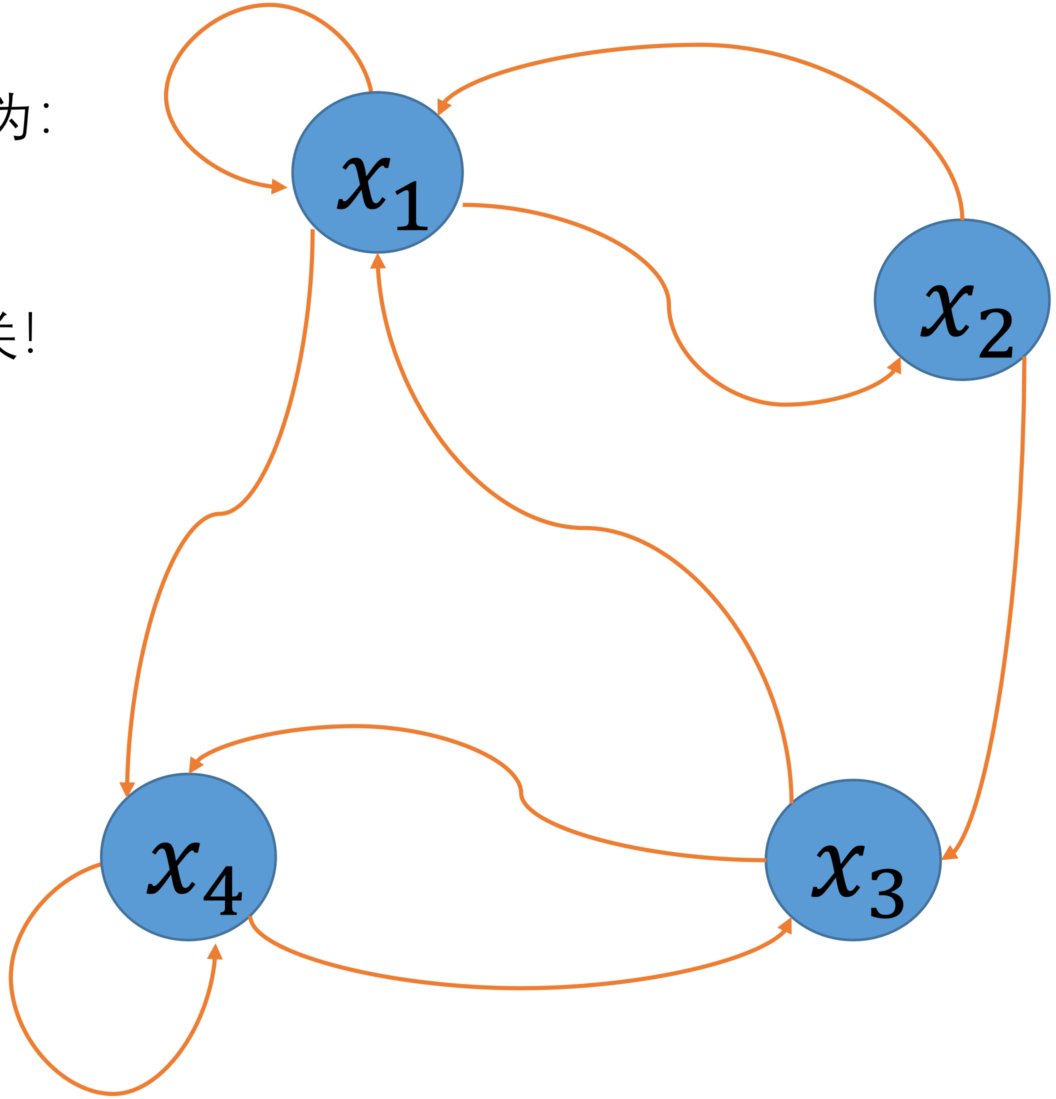
- 显然有

$$p_{ij} \geq 0, \sum_{j=0}^{\infty} p_{ij} = 1, i = 0,1,\cdots.$$

- 由转移概率p_{ij}组成的矩阵

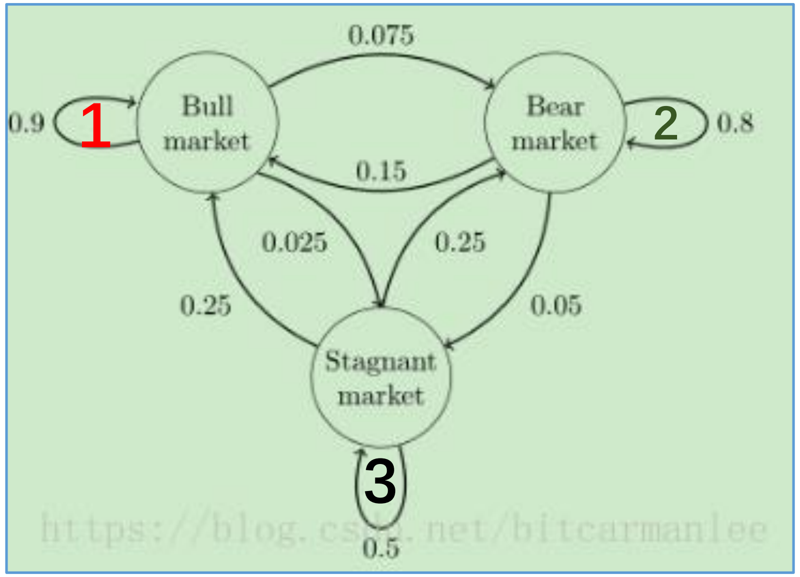
$$\begin{pmatrix} p_{00} & p_{01} & \cdots \\ p_{10} & p_{11} & \cdots \\ \vdots & \vdots & \vdots \end{pmatrix}$$

称为此过程的(一步)**转移概率矩阵**，记为P.



例 1: 平稳分布示例

- 只考虑三种市场(market)状态:




```
P = [ 0.9, 0.075, 0.025; ...
      0.15,  0.8,  0.05; ...
      0.25, 0.25,   0.5];
s = [0.12, 0.23, 0.65];
for i = 1:50
    s = s*P;
end
```

- 状态转化的概率（转移矩阵，行和为1）：

$$P = \begin{pmatrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{pmatrix}$$

```
>> exe_market
Step    2 : ( 0.41270,  0.39215,  0.19515)
Step    4 : ( 0.52203,  0.38257,  0.09540)
Step    6 : ( 0.57128,  0.35655,  0.07217)
Step    8 : ( 0.59608,  0.33788,  0.06605)
Step   10 : ( 0.60923,  0.32669,  0.06408)
Step   12 : ( 0.61636,  0.32035,  0.06329)
Step   14 : ( 0.62026,  0.31683,  0.06292)
Step   16 : ( 0.62239,  0.31488,  0.06273)
Step   18 : ( 0.62357,  0.31381,  0.06262)
Step   20 : ( 0.62421,  0.31322,  0.06257)
Step   22 : ( 0.62457,  0.31290,  0.06254)
Step   24 : ( 0.62476,  0.31272,  0.06252)
Step   26 : ( 0.62487,  0.31262,  0.06251)
Step   28 : ( 0.62493,  0.31257,  0.06251)
Step   30 : ( 0.62496,  0.31254,  0.06250)
Step   32 : ( 0.62498,  0.31252,  0.06250)
Step   34 : ( 0.62499,  0.31251,  0.06250)
Step   36 : ( 0.62499,  0.31251,  0.06250)
Step   38 : ( 0.62500,  0.31250,  0.06250)
Step   40 : ( 0.62500,  0.31250,  0.06250)
Step   42 : ( 0.62500,  0.31250,  0.06250)
Step   44 : ( 0.62500,  0.31250,  0.06250)
Step   46 : ( 0.62500,  0.31250,  0.06250)
Step   48 : ( 0.62500,  0.31250,  0.06250)
Step   50 : ( 0.62500,  0.31250,  0.06250)
```

平稳分布示例:Python版本

```
Python 3.6.8 Shell
File Edit Shell Debug Options Window Help
Python 3.6.8 (tags/v3.6.8:3c6b436a57, Dec 24 2018, 00:16:47) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: D:\Documents\计算机模拟\lec06 - MCMC方法II\demoMarkov.py
Current round: 1 [[0.405   0.4175 0.1775]]
Current round: 2 [[0.4715   0.40875 0.11975]]
Current round: 3 [[0.5156 0.3923 0.0921]]
Current round: 4 [[0.54591   0.375535 0.078555]]
Current round: 5 [[0.567288 0.36101   0.071702]]
Current round: 6 [[0.5826362 0.3492801 0.0680837]]
Current round: 7 [[0.59378552 0.34014272 0.06607176]]
Current round: 8 [[0.60194632 0.33316603 0.06488765]]
Current round: 9 [[0.6079485   0.32790071 0.06415079]]
Current round: 10 [[0.61237646 0.3239544   0.06366914]]
Current round: 57 [[0.62499999 0.31250001 0.0625     ]]
Current round: 58 [[0.62499999 0.31250001 0.0625     ]]
Current round: 59 [[0.62499999 0.3125     0.0625     ]]
Current round: 60 [[0.625   0.3125 0.0625]]
Current round: 61 [[0.625   0.3125 0.0625]]
Current round: 62 [[0.625   0.3125 0.0625]]
Current round: 63 [[0.625   0.3125 0.0625]]
```

- 迭代关系

$$p_{t+1} = p_t P$$

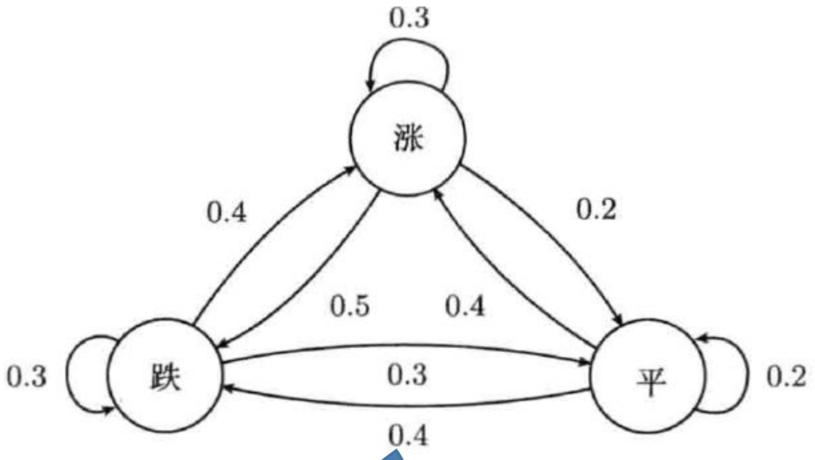
- 考察极限情形$t \to \infty$

$$\pi P = \pi$$

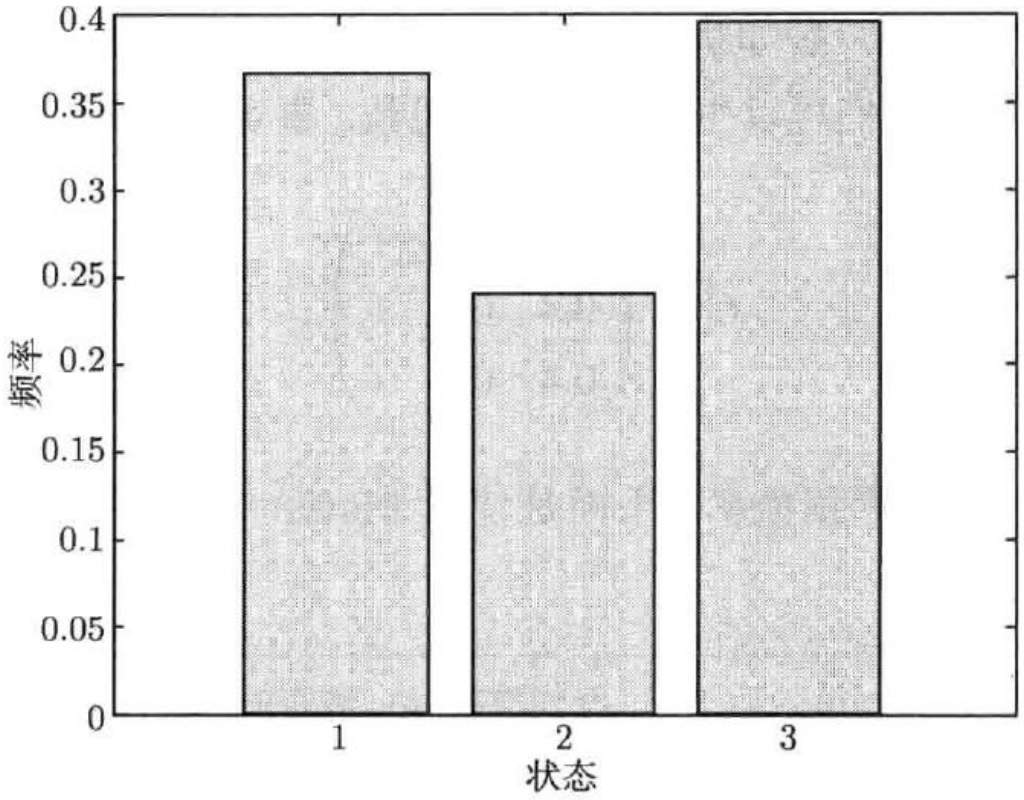
记 $\lim\limits_{t\to\infty} p_t = \pi.$

```python
import numpy as np
P = np.matrix([[0.8,  0.075, 0.125],
               [0.15, 0.8,   0.05 ],
               [0.25, 0.25,  0.5  ]],
              dtype=float)
# try [[0.3,0.4,0.3]] and [[0.7,0.1,0.2]]
s = np.matrix([[0.25,0.25,0.5]],
              dtype=float)
for i in range(50):
    s = s*P
```

例 2



$$p = (0.36415,\ 0.2395,\ 0.39635)。$$




```
n = 2500000;
t0 = 2000000;
P = [0.3, 0.2, 0.5; ...
     0.4, 0.2, 0.4; ...
     0.4, 0.3, 0.3];
C = cumsum(P, 2)
state = ones(1, n);
r = unifrnd(0, 1, 1, n);
i = 1;
for t = 1:n
    j = state(t);
    while (r(t) > C(i, j))
        j = j + 1;
    end
    state(t) = j;
    i = j;
end
h = hist(state(t0 + 1 : n), ...
                        [1, 2, 3]);
p = h / (n - t0);
bar(p);
```

例 3： 周期性转移案例

例 6.3 如图 6.4 所示的转移概率情况就是周期性的，其转移概率矩阵是

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 1 & 0 \end{pmatrix}。$$

容易验证有：$P^3 = P$，则对任意概率向量 α 有：$\alpha P = (\alpha P)P^2$，即该链的周期是 2。

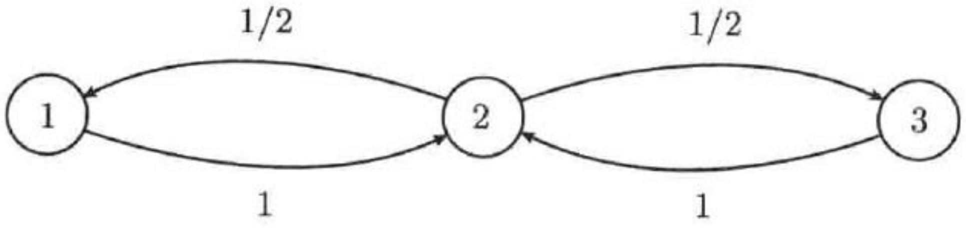


图 6.4 链的周期为 2，对于从任意状态 x 出发再返回到 x 的迭代次数是 2 的倍数

➢ 利用例1的程序，不难得到其不变分布为 $\pi = (0.3636,\ 0.2397,\ 0.3967)$

平稳性分析

是否收敛依赖于:

- 转移矩阵
- 初始状态
- 随机性
- …

马尔科夫链收敛的必要条件:

- 可能的状态数是有限的
- 状态间的转移概率需要固定不变
- 从任意状态能够转变到任意状态
- 不能是简单的循环

Markov链 - 平稳分布/条件

马氏链定理：如果一个非周期的Markov链有状态转移矩阵P，并且它的任何两个状态是连通的，那么$\lim_{n\to\infty} P_{ij}^n = \pi(j)$存在且与i无关，记

$$\lim_{n\to\infty} P_{ij}^n = \pi(j)$$

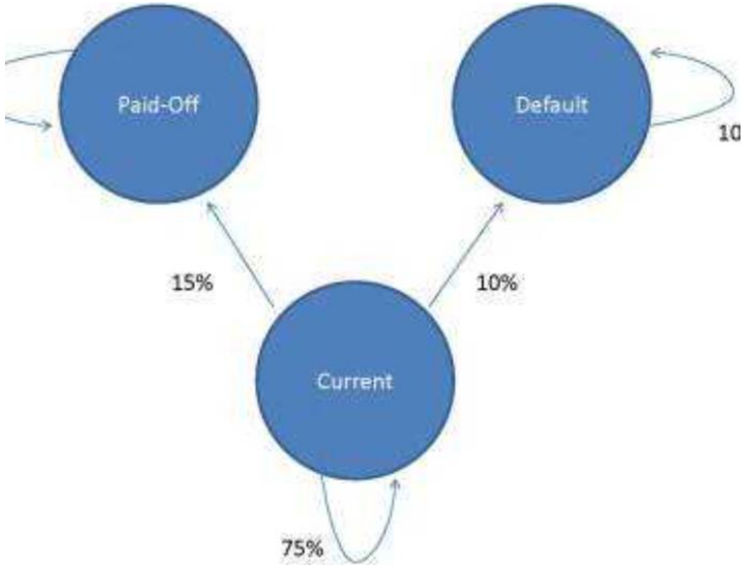
即

$$\lim_{n\to\infty} P^n = \begin{pmatrix} \pi(1) & \pi(2) \cdots \pi(j) \cdots \\ \pi(1) & \pi(2) \cdots \pi(j) \cdots \\ & \cdots \\ \pi(1) & \pi(2) \cdots \pi(j) \cdots \\ & \cdots \end{pmatrix}$$

其中，$\pi(j) = \sum_{i=0}^{\infty} \pi_i P_{ij}$。此外，$\pi$是方程$\pi \mathrm{P} = \pi$的唯一非负解。且

$$\pi = [\pi(1), \pi(2), \cdots, \pi(j), \cdots], \qquad \sum_{i=0}^{\infty} \pi_i = 1.$$

此时，我们称π是一个Markov链的平稳分布。



定理 6.1（Perron-Frobenius）如果概率转移矩阵P满足

$$p_{i,j} > \delta > 0, \forall\, i, j,$$

那么，有

（1）P存在特征值为1且对应的唯一左特征向量$\boldsymbol{w}$严格为正

（2）如果此特征向量被归一化则进一步有

$$\lim_{n\to\infty} P^n = \mathbf{1}\boldsymbol{w}$$

细致平稳条件/定理: $\pi_i p_{i,j} = \pi_j p_{j,i}$

定理：[细致平稳条件] 如果非周期马氏链的转移矩阵P和分布$\pi(x)$ 满足

$$\pi(i)P_{ij} = \pi(j)P_{ji} \qquad \text{for all} \quad i,j \tag{1}$$

则 $\pi(x)$ 是马氏链的平稳分布，上式被称为细致平稳条件(detailed balance condition)。

其实这个定理是显而易见的，因为细致平稳条件的物理含义就是对于任何两个状态i,j，从 i 转移出去到j 而丢失的概率质量，恰好会被从 j 转移回i 的概率质量补充回来，所以状态i上的概率质量$\pi(i)$是稳定的，从而$\pi(x)$是马氏链的平稳分布。数学上的证明也很简单，由细致平稳条件可得

$$\sum_{i=1}^{\infty}\pi(i)P_{ij} = \sum_{i=1}^{\infty}\pi(j)P_{ji} = \pi(j)\sum_{i=1}^{\infty}P_{ji} = \pi(j)$$
$$\Rightarrow \pi P = \pi$$

由于π 是方程 $\pi P = \pi$的解，所以π是平稳分布。

- 所有的 MCMC(Markov Chain Monte Carlo) 方法都是以这个定理作为理论基础的!

安德雷·安德耶维齐·马尔可夫(Андрей Андреевич Марков)

俄国数学家,1856年6月14日出生于梁赞州。成名作：**Markov 链**。

- 1874年入**圣彼得堡**大学，师从**切比雪夫 Tschebyshev**
- 1886年当选为**圣彼得堡**科学院院士
- 1893-1905任**圣彼得堡**大学教授(数论和概率论)
- 1922年7月20日逝世于**圣彼得堡**

他的**同名儿子**A·A·小马尔可夫也是一位著名数学家

2. MCMC方法

接受-拒绝(Acceptance-Rejection)法

1. 选择简单函数$g(x)$，满足$f(x) \leq Mg(x)$，如图所示：
2. 根据$g(x)$生成"建议随机数" y
3. 生成**均匀分布**随机数 u
4. 计算接受准则

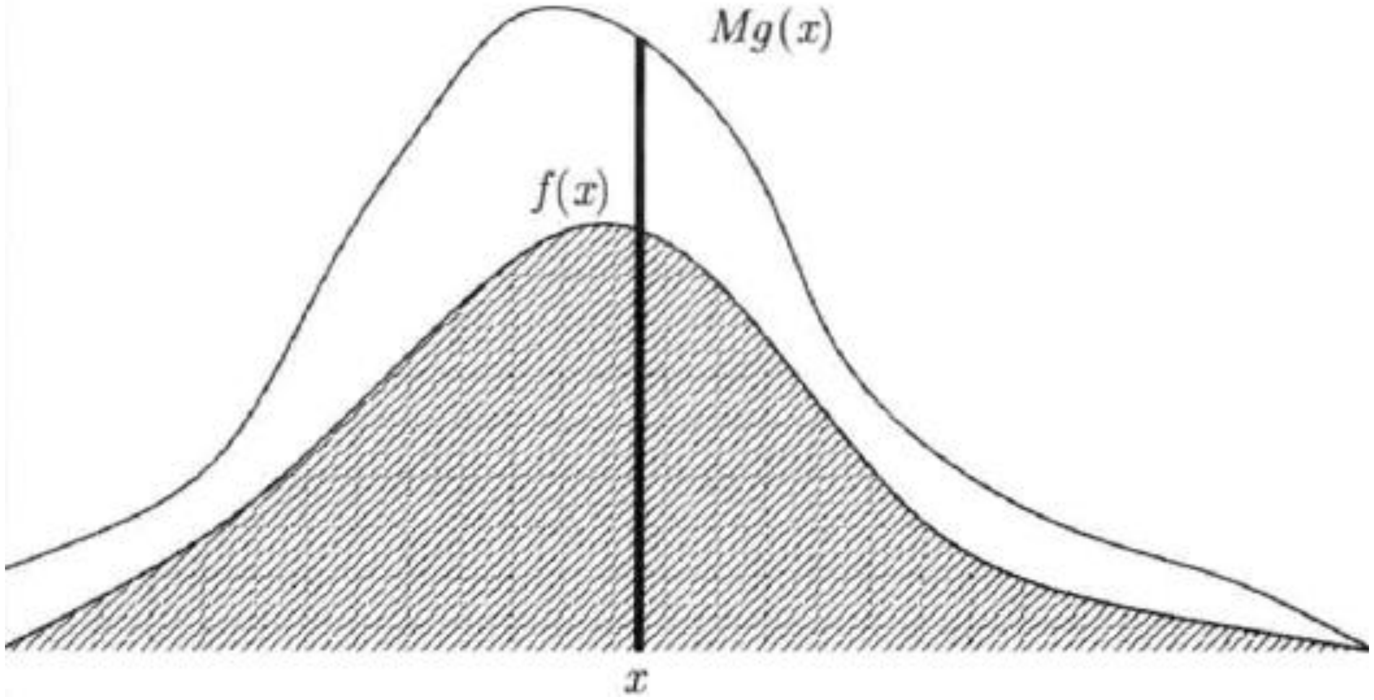
$$h(x) = \frac{f(x)}{Mg(x)},$$

若 $\mathbf{u} < \boldsymbol{h(x)}$，

接受y;

否则，

丢弃y，并转第2步；



例5.3: 无界概率密度函数 - Beta分布抽样

概率密度函数:

$$f(x) = cx^{\alpha-1}(1-x)^{\beta-1}, x \in (0,1)$$

A-R方法不需知归一化常数c

① $G(y) = 2^{\alpha}y^{\alpha}$

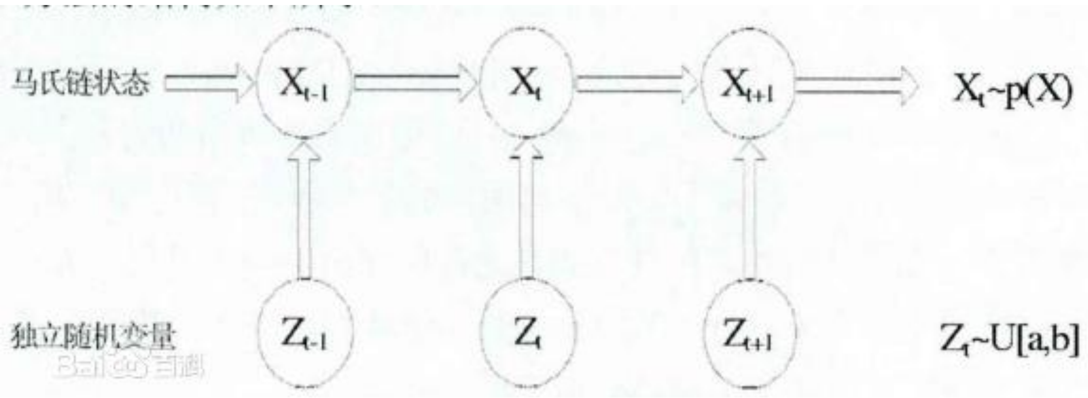
② $h(y) = \left(2(1-y)\right)^{(\alpha-1)}$

```
N = 500000;      alpha = 0.5;
U = unifrnd(0, 1, 1, N);
Y = 0.5 * U.^(1 / alpha);
h = (2 * (1 - Y)).^(alpha - 1);
R = unifrnd(0, 1, 1, N);
X = Y(R < h);

sample = length(X);              bins = 50;
S = unifrnd(0, 1, 1, sample);  S = (S > 0.5);
X = (1 - S).*X + S.*(1 - X)

[Xnumber, Xcenters] = hist(X, bins);
bar(Xcenters, Xnumber / sample);
title('Beta(0.5, 0.5)'); hold on;
dt = 0.005; t = dt : dt : 1 - dt;
z = (t.^(alpha - 1).*((1 - t).^(alpha - 1))) / pi;
scale = 1 / bins; plot(t, scale * z, 'r'); hold off;
```

MCMC 采样



基本思想：构造一条Markov链，使其**平稳分布恰好为给定的概率分布**；通过这条Markov链达到平稳分布时的样本(有效样本)确定随机数的选择。

Algorithm 5 MCMC 采样算法

1: 初始化马氏链初始状态$X_0 = x_0$

2: 对$t = 0, 1, 2, \cdots$, 循环以下过程进行采样

- 第t个时刻马氏链状态为$X_t = x_t$, 采样$y \sim q(x|x_t)$
- 从均匀分布采样$u \sim Uniform[0, 1]$
- 如果$u < \alpha(x_t, y) = p(y)q(x_t|y)$ 则接受转移$x_t \to y$，即$X_{t+1} = y$
- 否则不接受转移，即$X_{t+1} = x_t$

Metropolis – Hastings 抽样

（梅特波利斯 － 黑斯廷斯） 1970

- 定义

$$h(x,y)=\min\left\{1,\frac{f(y)g(x|y)}{f(x)g(y|x)}\right\}$$

- 特殊地，当对称性满足时:

$$h(x,y)=\min\left\{1,\frac{f(y)}{f(x)}\right\}$$

Algorithm 6 Metropolis-Hastings 采样算法

1: 初始化马氏链初始状态$X_0=x_0$

2: 对$t=0,1,2,\cdots$，循环以下过程进行采样

- 第t个时刻马氏链状态为$X_t=x_t$，采样$y\sim q(x|x_t)$
- 从均匀分布采样$u\sim Uniform[0,1]$
- 如果$u<\alpha(x_t,y)=\boxed{\min\left\{\frac{p(y)q(x_t|y)}{p(x_t)p(y|x_t)},1\right\}}$ 则接受转移$x_t\rightarrow y$，即$X_{t+1}=y$
- 否则不接受转移，即$X_{t+1}=x_t$

- Metropolis 算法用一个**对称的**建议分布 $g(\cdot|x_t)$ 来产生潜在转移点y，然后根据**特定的接受拒绝方法**来决定是否转移到该潜在点
- 最初的 Metropolis 算法将 $g(\cdot|x_t)$ 取为对称函数；而 Metropolis-Hasting方法将之推广到非对称情形，且可逆：$g(y|x)>0 \Leftrightarrow g(x|y)>0$

实例 1. 模拟掷双骰子

未归一的目标概率密度函数表

x	2	3	4	5	6	7	8	9	10	11	12
$f(x)$	1	2	3	4	5	6	5	4	3	2	1

（1）易得每个骰子掷一次都有6种情况，那么共有6×6=36种可能，

点数之和为7的有（3，4）；（2，5）；（1，6）；（4，3）；（5，2）；（6，1），共6种，

所以，所求的概率是 $\frac{6}{36}=\frac{1}{6}$.

（2）事件“点数之和小于7”的基本事件有：（1，1）；（2，1）；（1，2）；（1，3）；（3，1）；（1，4）；（4，1）；

（1，5）；（5，1）；（2，2）；（2，3）；（3，2）；（2，4）；（4，2）；（3，3），共计15个，

而所有的基本事件共有36个，

故事件“点数之和小于7”的概率为 $\frac{15}{36}=\frac{5}{12}$.

（3）事件“点数之和等于或大于11”的基本事件有：（5，6）；（6，5）；（6，6），共计3个，

而所有的基本事件共有36个，

故事件“点数之和等于或大于11”的概率为 $\frac{3}{36}=\frac{1}{12}$.

1. 极小邻域法
2. 极大邻域法

1. 极小邻域法 – 6.3.1

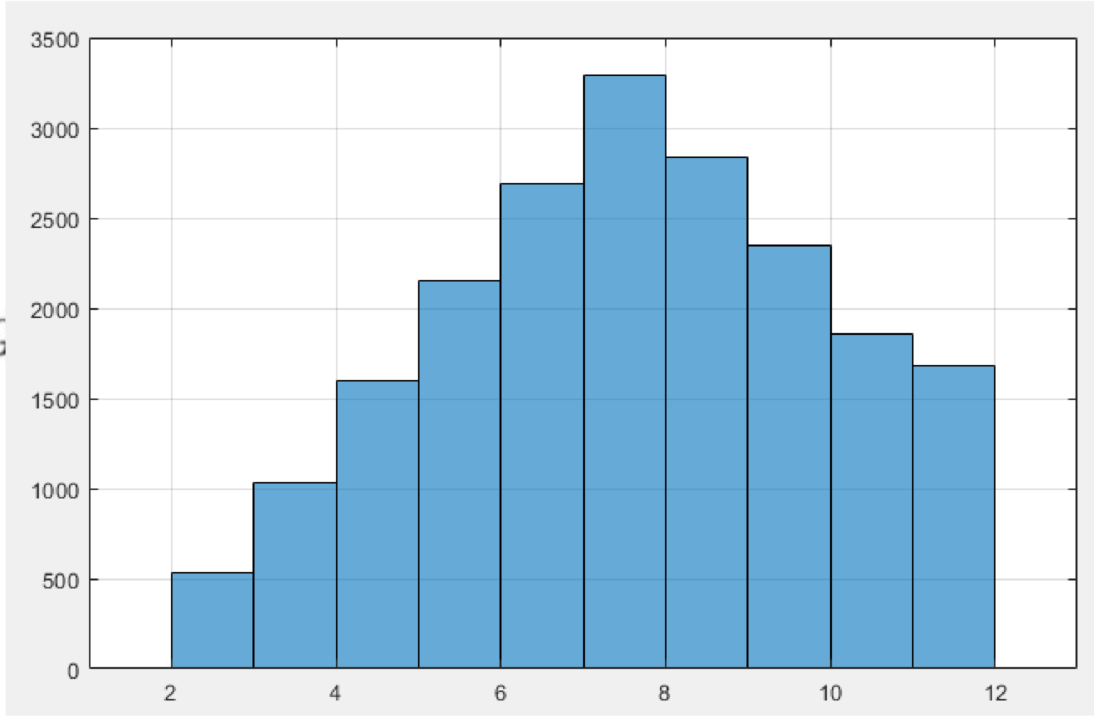
- 对于每个状态x，定义其建议概率密度函数为：

$$g(y|x)=\begin{cases}1/2, & y=\max\{x-1,\ 2\},\\ 1/2, & y=\min\{x+1,\ 12\},\\ 0, & \text{其他。}\end{cases}$$

$$G=\begin{pmatrix}1/2 & 1/2 & 0 & \cdots & 0 & 0 & 0\\ 1/2 & 0 & 1/2 & \cdots & 0 & 0 & 0\\ 0 & 1/2 & 0 & \cdots & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & \cdots & 0 & 1/2 & 0\\ 0 & 0 & 0 & \cdots & 1/2 & 0 & 1/2\\ 0 & 0 & 0 & \cdots & 0 & 1/2 & 1/2\end{pmatrix}$$

极小邻域法实现

x	2	3	4	5	6	7	8	9	10	11	12
$f(x)$	1	2	3	4	5	6	5	4	3	2	1

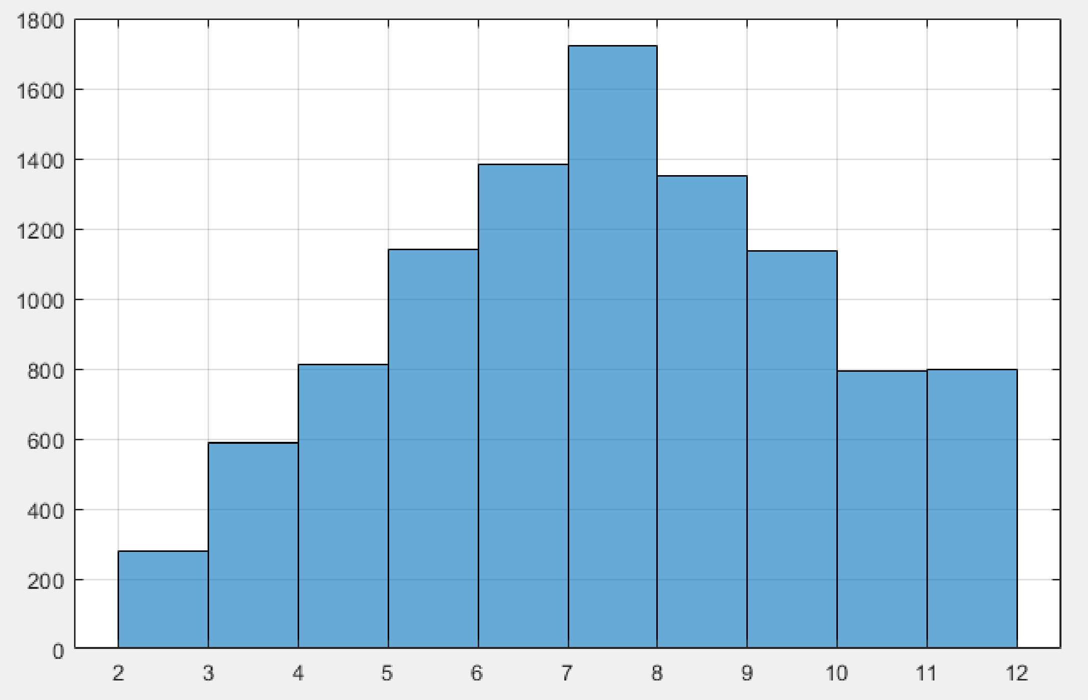



```
f = [1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1];  % no zero!
d = zeros(1, 50000);
x = 5;  % initial value, whatever.
for i = 1:50000
    U = rand();
    if (U < 0.5)
        y = max(x - 1, 2);
    else
        y = min(x + 1, 12);
    end
    h = min(1, f(y - 1) / f(x - 1));
    U = rand();
    if (U < h)
        x = y;
    end
    d(i) = x;
end
histogram(d(30000:end), 2:12);
grid on; axis([1 13 0 3500])
```

2. 极大邻域法 – 6.3.2

```
f = [1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1];
d = zeros(1, 40000);
x = 5;        % initial value
for i = 1:40000
    y = randi(11, 1) + 1;
    h = min(1, f(y - 1) / f(x - 1));
    U = rand();
    if (U < h)
        x = y;
    end
    d(i) = x;
end
a = 1:1:12;
hist(d(30000:end), a);
```

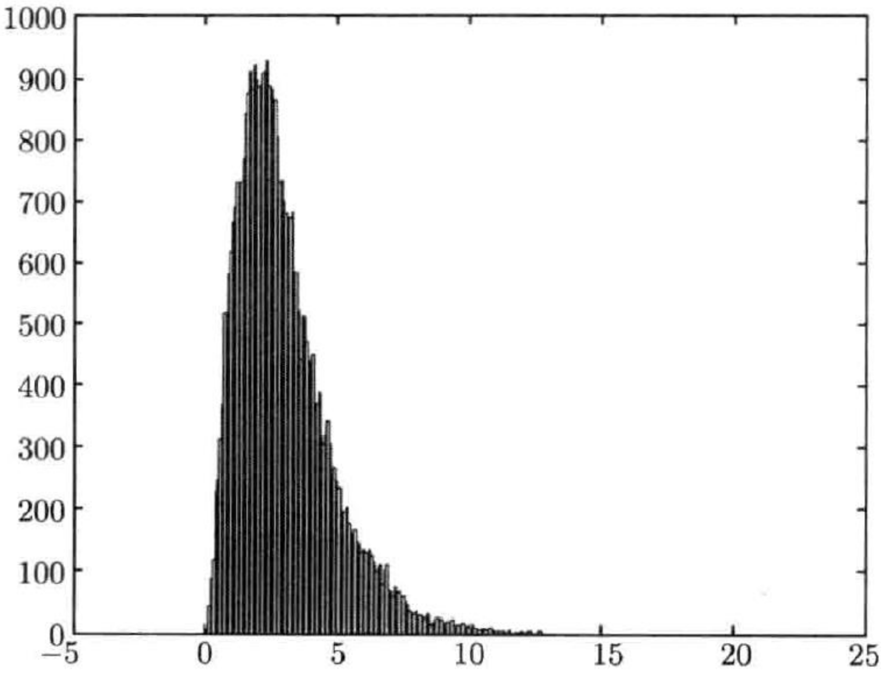
$$G = \begin{pmatrix} \frac{1}{11} & \frac{1}{11} & \cdots & \frac{1}{11} \\ \frac{1}{11} & \frac{1}{11} & \cdots & \frac{1}{11} \\ \vdots & \vdots & & \vdots \\ \frac{1}{11} & \frac{1}{11} & \cdots & \frac{1}{11} \end{pmatrix}$$



例4：连续分布抽样

- 考虑概率密度函数(Γ分布)

$$f(x) = 0.5x^2e^{-x}$$



MCMC 抽样 20000 次产生分布密度为 $f = 0.5x^2\mathrm{e}^{-x}$

```
N = 40000;
f = inline('0.5*x*x*exp(-x)', 'x');
d = zeros(1, N);
x = 2;
for i = 1:N
    y = x - 1 + 2*rand();
    if y < 0
        y = x;
    end
    h = min(1, f(y) / f(x));
    U = rand;
    if (U < h)
        x = y;
    end
    d(i) = x;
end
a = 0:0.08:20;
hist(d(floor(N / 2):end), a);
```

MCMC遍历性定理

- 仅要求Markov链收敛于一个不变分布对于MCMC抽样还不够;
- 需计算收敛的极限，但若只是从Markov链中挑选并不合适;
- 对于特定的情形： O.K.

定理 6.2（遍历性） 若$\{x_1, x_2, \cdots, x_N\}$是一个在有限状态空间Ω上的不可约且非周期的Markov链，其平稳分布为π。设$\xi: \Omega \to \mathbb{R}$为任意映射，则

$$\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \xi(x_i) = \boldsymbol{E}_\pi(\xi),$$

其中，$\boldsymbol{E}_\pi$指相对于分布π的期望。

➢这个定理说明了**MCMC方法的意义**：

人们可以用它来计算期望值，这是MCMC的标准用法!

MCMC收敛性与稳定性

In statistics, we estimate an unknown mean $\mu = E(X)$ of a distribution by collecting n iid samples from the distribution, $X_1, \ldots, X_n$ and using the sample mean

$$\overline{X}(n) = \frac{1}{n}\sum_{j=1}^{n} X_j. \tag{1}$$

Letting $\sigma^2 = Var(X)$ denote the variance of the distribution, we conclude that

$$Var(\overline{X}(n)) = \frac{\sigma^2}{n}. \tag{2}$$

由中心极限定理可知

$$Z_n := \frac{\sqrt{n}}{\sigma}(\bar{X}(n) - \mu)$$

渐进服从正态分布($n \to \infty$)，即

$$\lim_{n\to\infty}\left(\frac{\sqrt{n}}{\sigma}(\bar{X}(n) - \mu) \le X_\alpha\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{X_\alpha} \exp\left(-\frac{t^2}{2}\right) dt,$$

这里，X_α称为正态差.

MCMC 误差估计

- 与普通数值方法误差估计不同，具有随机性！
- 利用（勒维-林德伯格）中心极限定理可知

$$\lim_{n\to\infty} \Pr\left(\left(\bar{X}(n) - \mu \right) \le X_\alpha \frac{\sigma}{\sqrt{n}} \right) = \frac{1}{\sqrt{2\pi}} \int_{-X_\alpha}^{X_\alpha} \exp\left(-\frac{t^2}{2}\right) dt := 1-\alpha,$$

其中， α称为置信概率（显著水平）， $1-\alpha$称为置信水平。

- 统计量估计值的绝对误差为

$$\varepsilon_\alpha = \left| \frac{1}{n} \sum_{i=1}^{n} X(n) - \mu \right| = X_\alpha \frac{\sigma}{\sqrt{n}}$$

MCMC 模拟效率

直接模拟的缺点：收敛速度慢！计算精度低！结果方差大！ 一般可用模拟“费用”衡量：

$$C = \sigma^2 t$$

其中， σ^2是所计算统计量的方差，模拟时间t包括：

1. 数据输入和初始化时间
2. 每次模拟的抽样时间
3. 每次模拟计算统计量的时间
4. 结果输出的时间

MCM的效率与方差和模拟时间成反比，

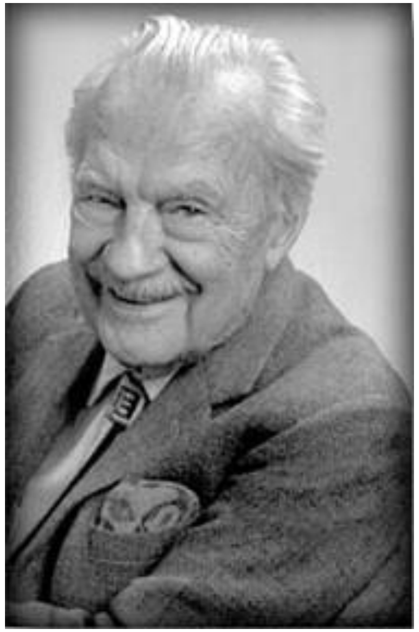
改善途径主要有两种做法：

1. 降低方差σ^2

2. 减少模拟时间t

关于 Metropolis 算法

- Nicolas **Metropolis**, Arianna W. **Rosenbluth**, Marshall N. **Rosenbluth**, Augusta H. **Teller**, and Edward **Teller** : Equation of State Calculations by Fast Computing Machines. The Journal of Chemical Physics, 21, **1087-1092, (1953)**.




为了研究粒子系统的平稳性质， Metropolis 考虑了物理学中常见的波尔兹曼分布的采样问题，首次提出了基于马氏链的蒙特卡罗方法，即Metropolis算法。它是**首个**普适的采样方法，并**启发**了一系列MCMC方法，所以人们把它视为随机模拟技术腾飞的**起点**。这篇文章至今已被引用了**17,000**多次

总结

Markov过程

MCMC

采样$y \sim q(x|x_t)$

Metroplis-Hastings算法

$$h(x,y) = \min\left\{1, \frac{f(y)g(x|y)}{f(x)g(y|x)}\right\}$$

MCMC：回顾 & 展望

1 – 课程简介及概率基础

2 – 随机数生成器

3 – 指定分布随机数/抽样

4 – 离散系统模拟(Poisson过程)

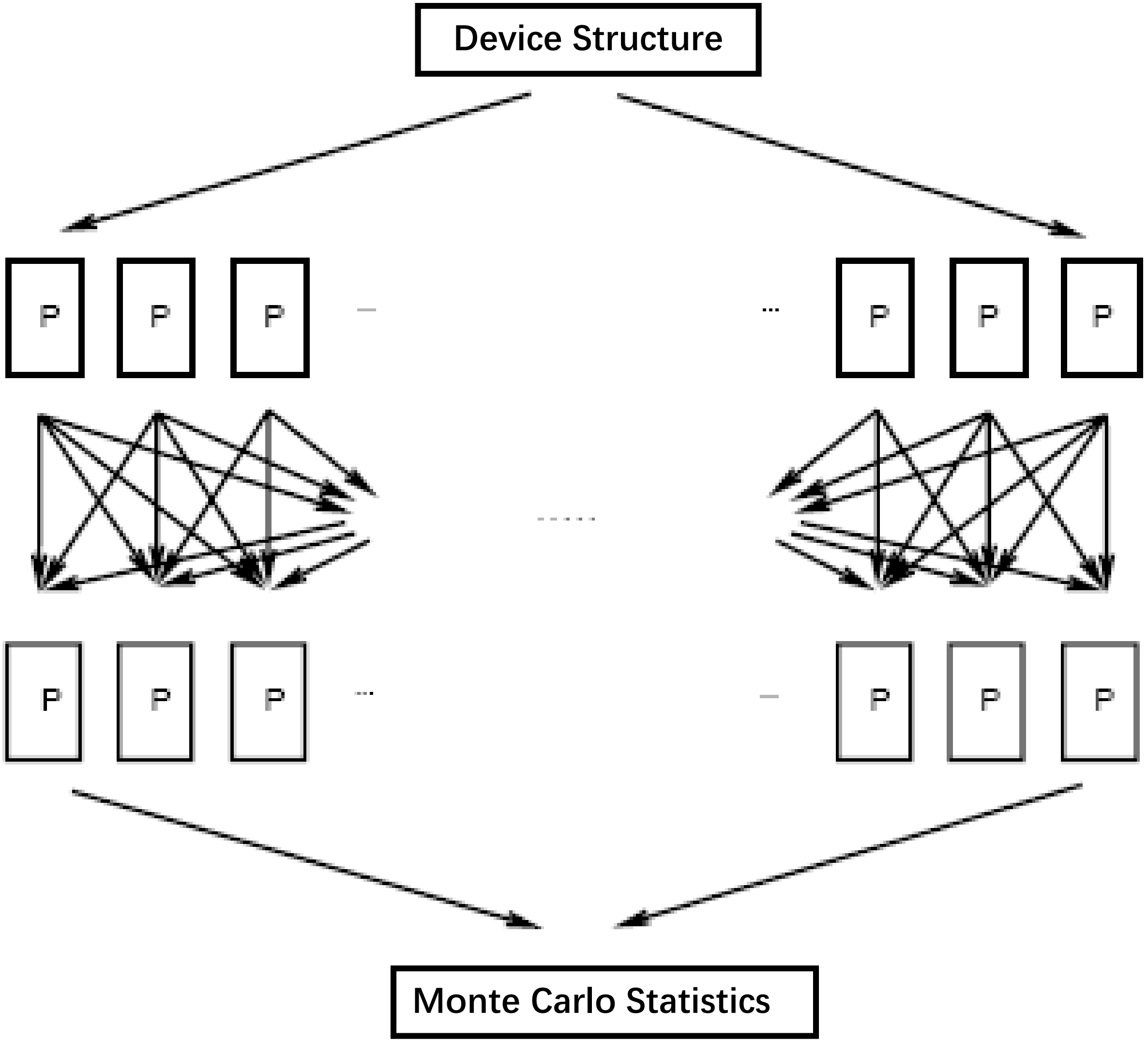
5 – 基本蒙特卡罗(MC)方法

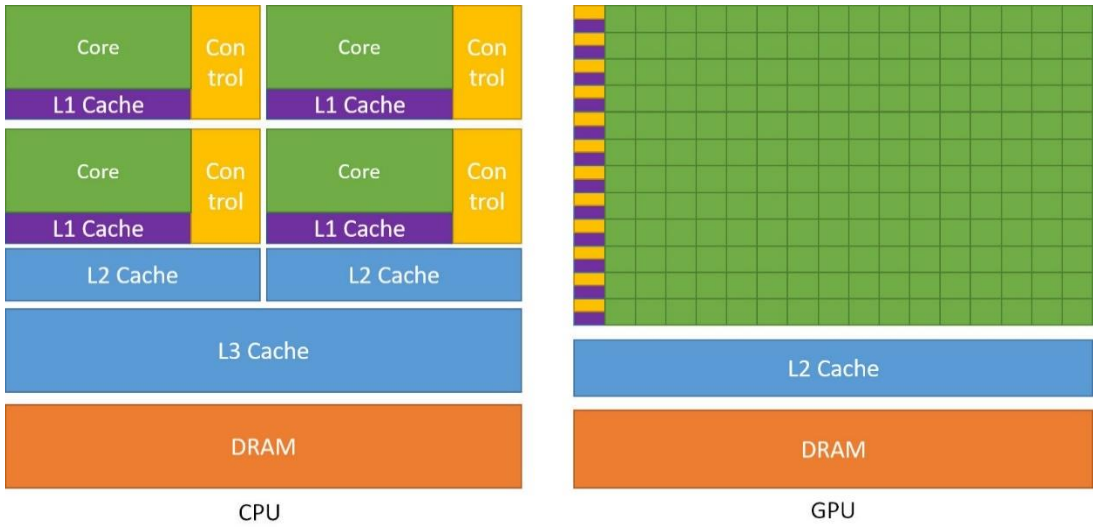
6 – Markov Chain Monte Carlo

Homework 07：

1. **第六章课后题 2、3、4**
2. 请叙述有限状态空间情形下，**对称**Metropolis算法的收敛性
3. **阅读如下文献，在归纳基础上撰写读书心得一篇(篇幅不限)**
    - **C. Andrieu, et.al., An Introduction to MCMC for Machine Learning, Machine Learning, 50, 5–43, 2003.**

Parallel Monte Carlo - 采样的硬件加速





Parallel Monte Carlo - 应用举例

在GPU上使用Monte Carlo方法解决欧式期权定价问题

● 徐磊　徐莹　寇大治

上海超级计算中心 上海 201203 lxu@ssc.net.cn

● 白雪

复旦大学 上海 201203 xueba@fudan.edu.cn

摘要:

GPU计算在最近几年得到了蓬勃的发展，尤其是其向量化的计算处理方式，使一些特定的计算方法得到了极大的计算性能提升，这就包括使用Monte Carlo方法计算的一些应用领域，而在金融计算中就大量的使用到了Monte Carlo方法来处理一些期权问题。本文针对最基本的欧式期权定价问题，使用Monte Carlo方法实现了GPU上的移植和开发，经过测试取得了良好的计算效果。

4. 测试结果与分析

选取PATH数为$1.78*10^7$进行测试，CUDA版本中，设置的curandState规模为2048*366，测试的结果如下：

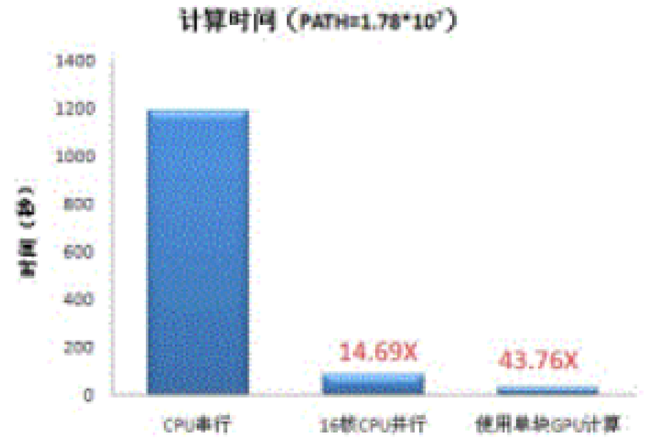


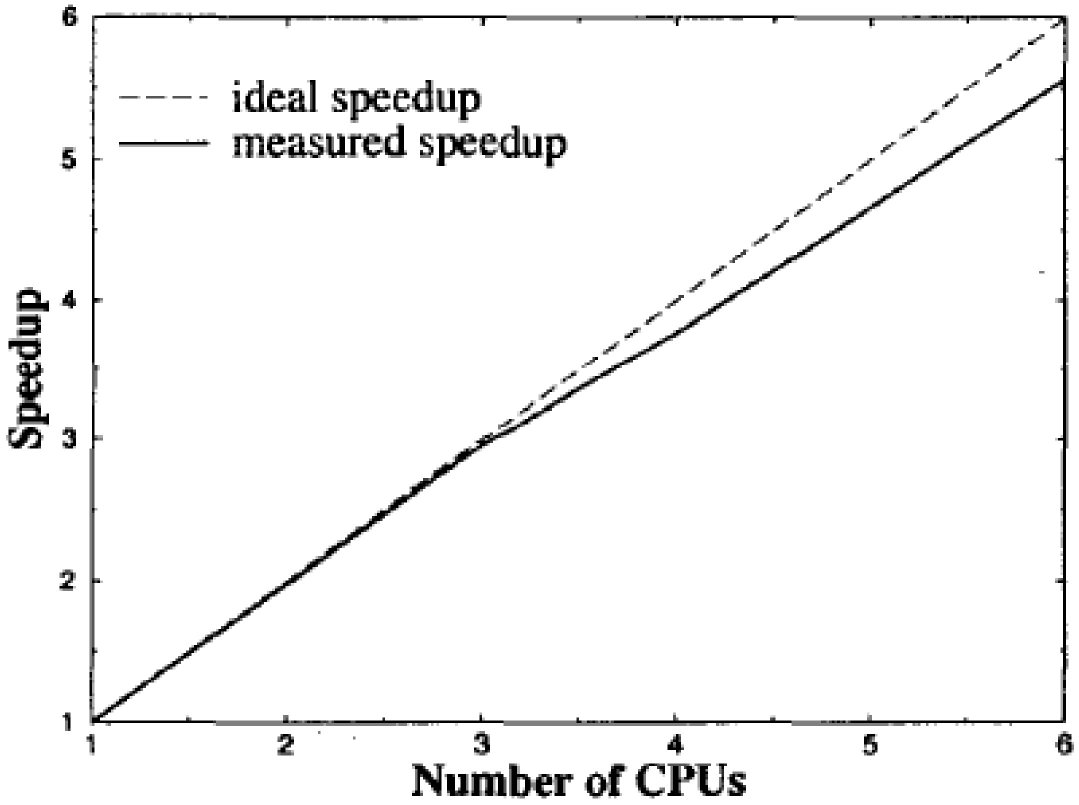
图2 计算时间比较

由上图可以看出，使用np=16的MPI版本，较串行版本有近15倍的加速，而使用单块Tesla S2050显卡，较串行版本有43.76倍的加速。

Parallel Monte Carlo – Pro & Cons

1. Scalability
2. Easy to convert from sequential code

1. Load Balancing
2. 随机数算法的可并行(复现)性



Parallel Monte Carlo Simulation of MBE Growth
Isabel Beichl Y. Ansel Teng James L. Blue
Proceedings of the 9th International Symposium on Parallel Processing 1995, Page(s) 46-52

Parallel Monte Carlo Simulation of Ion Implantation
Andreas Hossinger Erasmus Langer
Proceedings 13th Int.Conf. on Ion Implantation Technology; (2000), Page(s): 203 – 208

- Parallel computing and Monte Carlo algorithms.pdf
- Parallel Monte Carlo Tree Search on GPU.pdf

Don't Trust Parallel Monte Carlo! P Hallekalek
Proceedings of the 1998 Workshop on Parallel and Distributed Simulation Volume 1, Page(s): 82-89