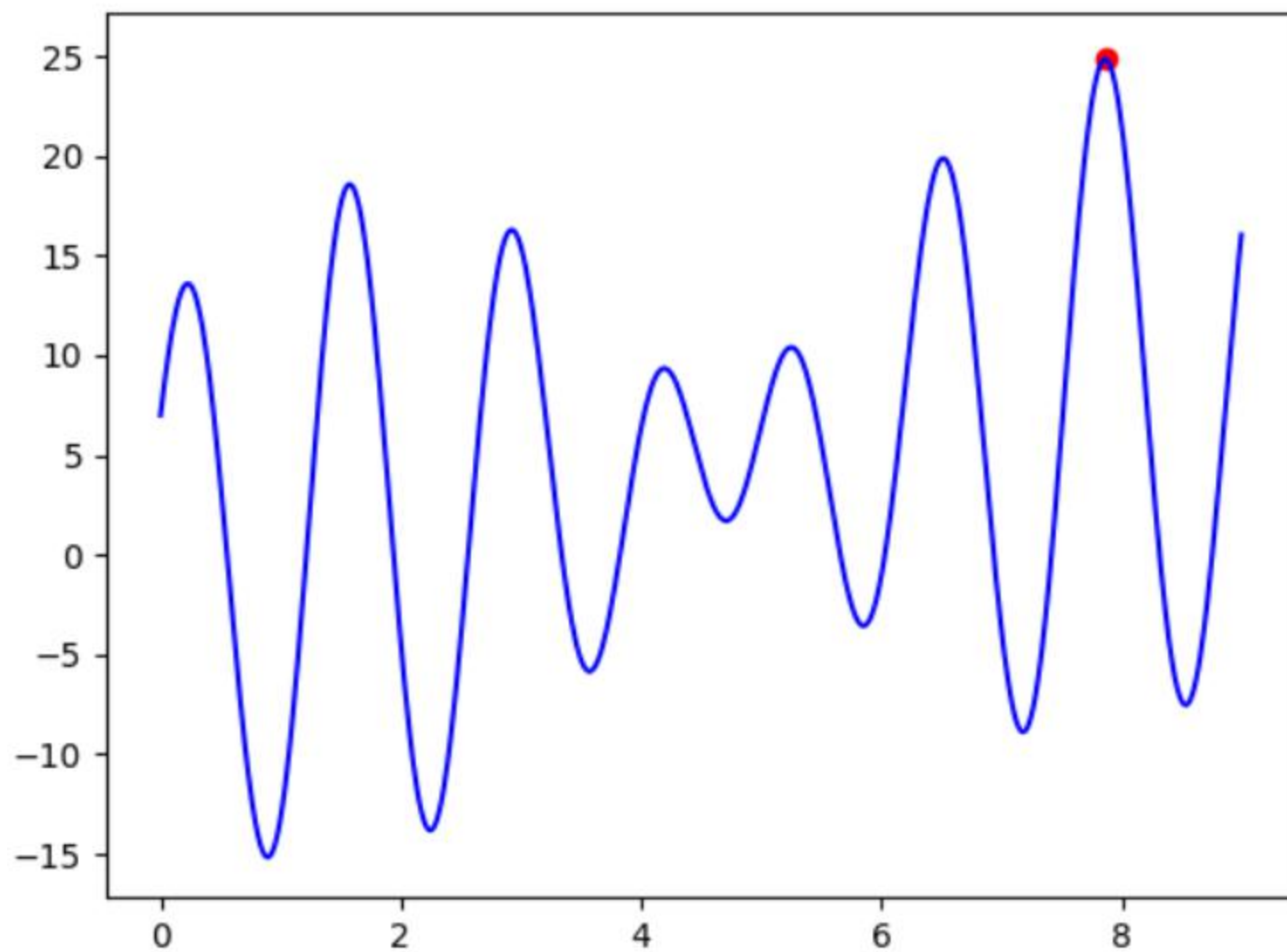


---

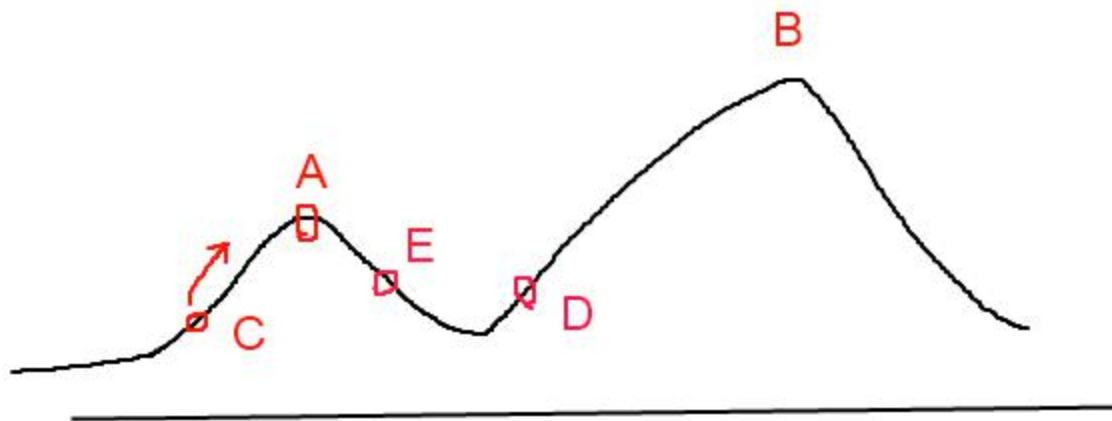
# 模拟退火算法

## Simulated Annealing



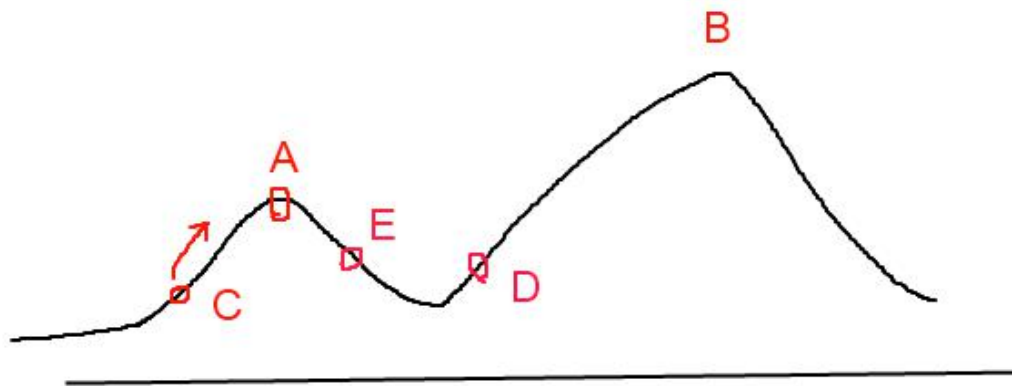
# 爬山算法

爬山算法是一种简单的贪心搜索算法，该算法每次从当前解的临近解空间中选择一个最优解作为当前解，直到达到一个局部最优解。



## 爬山算法

如图所示：假设C点为当前解，爬山算法搜索到A点这个局部最优解就会停止搜索，因为在A点无论向那个方向小幅度移动都不能得到更优的解。



## 模拟退火算法

在搜索到局部最优解A后，会以**一定的概率**接受到E的移动。也许经过几次这样的不是局部最优的移动后会到达D点，于是就跳出了局部最大值A。

# 关于优化问题

---

## 传统的优化方法（局部优化）

共轭梯度法、拟牛顿法、单纯形方法

## 全局优化方法

漫步法（Random Walk）、模拟退火法、GA

- ◆ **算法的提出**

模拟退火算法最早的思想由Metropolis等（1953）提出，

1983年Kirkpatrick等将其应用于组合优化

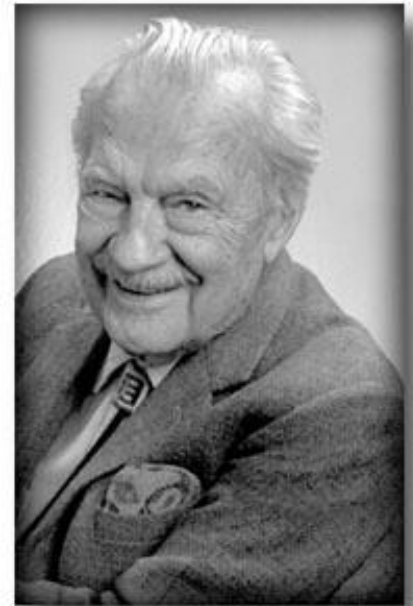
- ◆ **算法的目的**

解决NP(Non-deterministic Polynomial)

复杂性问题；

克服优化过程陷入局部极小；

克服初值依赖性。



Nick Metropolis

# 退火工艺：

---

退火是将金属和合金加热到适当温度，保持一定时间，然后缓慢冷却的热处理工艺。

## 退火的目的：

- ①降低钢的硬度，提高塑性，以利于切削加工及冷变形加工。
- ②细化晶粒，消除因铸、锻、焊引起的组织缺陷，均匀钢的组织 and 成分，改善钢的性能或为以后的热处理作组织准备。
- ③消除钢中的内应力，以防止变形和开裂

- 
- 相似性：
    - 金属  $\leftrightarrow$  问题
    - 能量状态  $\leftrightarrow$  成本函数
    - 温度  $\leftrightarrow$  控制参数
    - 完整排列的晶体结构  $\leftrightarrow$  问题的最优解



# 算法的基础

---

- 1、源于对固体退火过程的模拟；
- 2、采用Metropolis接受准则；

# 算法的基本思想

---

模拟退火算法采用类似于物理退火的过程：

1. 先在一个高温状态下（相当于算法随机搜索），
2. 然后逐渐退火，
3. 在每个温度下（相当于算法的每一次状态转移）徐徐冷却（相当于算法局部搜索），
4. 最终达到物理基态（相当于算法找到最优解）

## 什么是退火：

- ✓ 指将固体加热到足够高的温度，使分子呈随机排列状态，
- ✓ 然后逐步降温使之冷却，最后分子以低能状态排列，固体达到某种稳定状态。



---

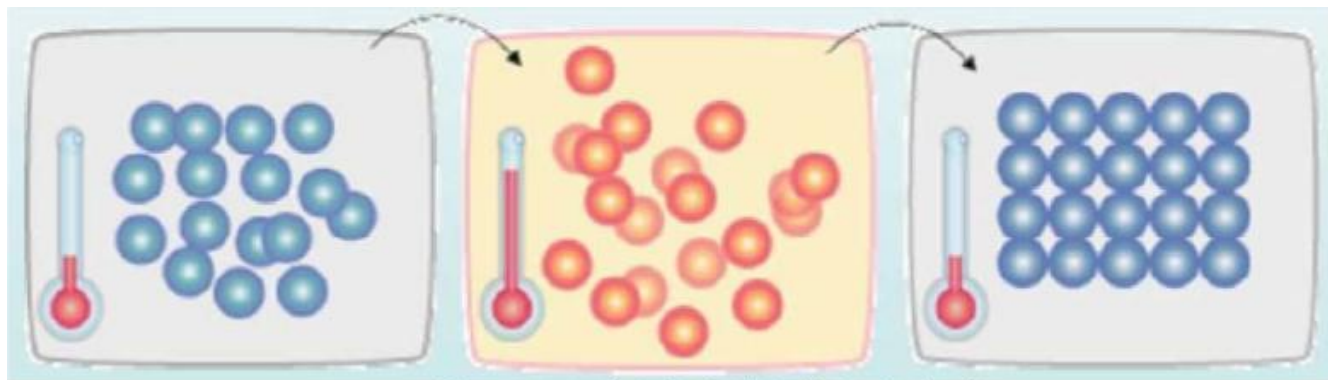
**热力学中的退火现象指物体逐渐降温时发生的物理现象：**

温度越低，物体的能量状态越低，到达足够的低点时，液体开始冷凝与结晶，在结晶状态时，系统的能量状态最低。缓慢降温（退火，annealing）时，可达到最低能量状态；但如果快速降温（淬火，quenching），会导致不是最低能态的非晶形。

**大自然知道慢工出细活：**

缓缓降温，使得物体分子在每一温度时，能够有足够时间找到安顿位置，则逐渐地，到最后可得到最低能态，系统最稳定。

# 物理退火过程



物理退火过程的发展阶段

1. **加温过程**。其目的是增强粒子的热运动，使其偏离平衡位置。当温度足够高时，固体将溶解为液体，溶解过程与系统的熵增过程联系，系统能量也随温度的升高而增大，使得每一粒子的状态都具有充分的**随机性**。
2. **等温过程**。物理学的知识告诉我们，对于与周围环境交换热量而温度不变的封闭系统，系统状态的自发变化总是朝自由能减少的方向进行，当自由能达到最小时，系统达到**平衡态**。
3. **冷却过程**。目的是使粒子的热运动减弱并渐趋有序，系统能量**逐渐**下降，从而得到低能的晶体结构。在常温时达到基态，内能减为**最小**。

## 数学表述:

- 物体加热至一定温度后，所有分子在状态空间D中自由运动，随着温度的下降，分子逐渐停留在不同的状态。在温度最低时，分子重新以一定的结构排列。
- 在温度T, 分子停留在状态r满足Boltzmann概率分布:

$$P\{\bar{E} = E(r)\} = \frac{1}{Z(T)} \exp\left(-\frac{E(r)}{k_B T}\right)$$

$\bar{E}$ 表示分子能量的一个随机变量， $E(r)$ 表示状态 $r$ 的能量， $k_B > 0$ 为Boltzmann常数。

$Z(T)$ 为概率分布的标准化因子:

$$Z(T) = \sum_{s \in D} \exp\left(-\frac{E(s)}{k_B T}\right)$$

在同一个温度 $T$ ，选定两个能量 $E_1 < E_2$ ，有

$$P\{\bar{E} = E_1\} - P\{\bar{E} = E_2\} = \frac{1}{Z(T)} \exp\left(-\frac{E_1}{k_B T}\right) \left[ 1 - \exp\left(-\frac{E_2 - E_1}{k_B T}\right) \right]$$

$>0$

$<1$

在同一个温度，分子停留在能量小的状态的概率比停留在能量大的状态的概率要大。



## 状态与温度的关系例子

- 四个能量点  $r = 1, 2, 3, 4$
- 三个温度点  $t = 20, 5, 0.5$

	$r = 1$	$r = 2$	$r = 3$	$r = 4$
$t = 20$	0.269	0.256	0.243	0.232
$t = 5$	0.329	0.269	0.221	0.181
$t = 0.5$	0.865	0.117	0.016	0.002

(1) 在同一个温度，分子停留在能量小状态的概率大于停留在能量大状态的概率；

(2) 温度越高，不同能量状态对应的概率相差越小；温度足够高时，各状态对应概率基本相同；

(3) 随着温度的下降，能量最低状态对应概率越来越大；温度趋于0时，其状态趋于1。



- **Metropolis准则（1953）——以概率接受新状态**  
固体在恒定温度下达到热平衡的过程可以用Monte Carlo方法加以模拟，该方法简单，但必须大量采样才能得到比较精确的结果，计算量很大。

# Monte Carlo模拟退火过程

---

- ✓ 蒙特卡罗(Monte Carlo)方法，或称计算机随机模拟方法，是一种基于“随机数”的计算方法。
- ✓ 数学家冯·诺伊曼用驰名世界的赌城—摩纳哥的Monte Carlo—来命名这种方法
- ✓ 民意测验的人不是征询每一个登记选民的意见，而是通过对选民进行小规模抽样调查来确定可能的优胜者。其基本思想是一样的。
- ✓ 它需要一个良好的随机数源。这种方法往往包含一些误差，但是随着随机抽取样本数量的增加，结果也会越来越精确。

## ● Metropolis准则（1953）——以概率接受新状态

- 若在温度  $T$ ，当前状态  $i \rightarrow$  新状态  $j$
- 若  $E_j < E_i$ ，则接受  $j$  为当前状态；
- 否则，若概率  $p = \exp[-(E_j - E_i)/k_B T]$  大于  $[0, 1)$  区间的随机数，则仍接受状态  $j$  为当前状态；若不成立则保留状态  $i$  为当前状态。

# Metropolis准则

在温度 $t$ ，初始状态 $i$ ，该状态的能量为  $E_i$ ，

随机选取某个粒子的位移随机地产生一微小变化，得到一个  
新状态 $j$ ，新状态的能量为  $E_j$

$$\left\{ \begin{array}{ll} E_j < E_i & \text{则接受新状态}j \\ E_j > E_i & \text{则考虑到热运动的影响} \end{array} \right.$$

$$r = \exp\left(-\frac{E_j - E_i}{kt}\right) \quad \xi \in [0,1) \text{ 随机数}$$

若  $r > \xi$ ，则接受  $j$ ，否则接受  $i$ 。

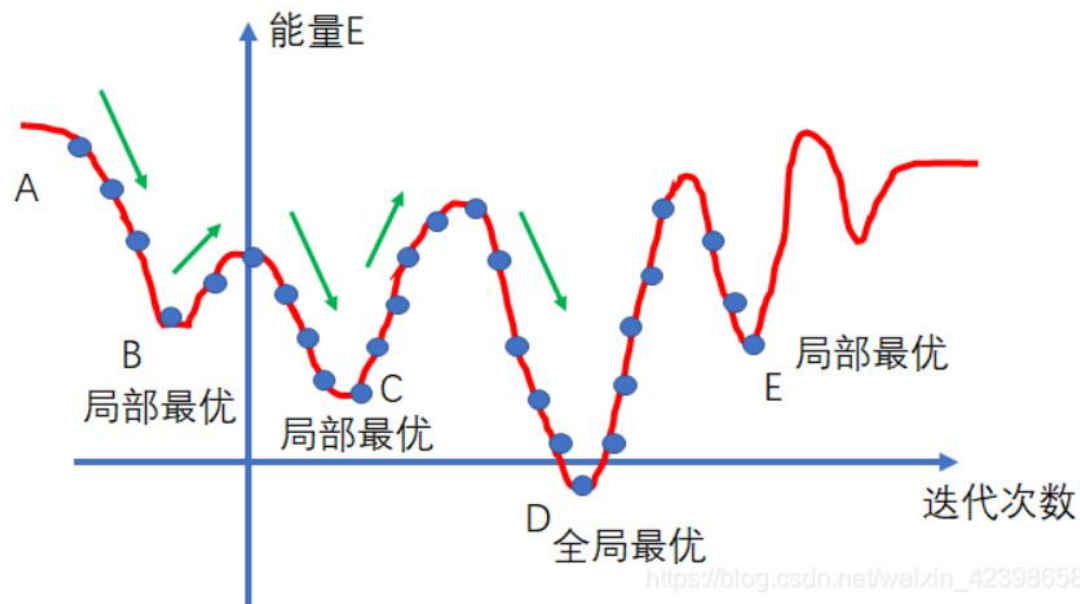
## 这种重要性采样过程的特点：

- 在高温下可接受与当前状态能量差较大的新状态；
- 在低温下基本只接受与当前能量差较小的新状态，
- 当温度趋于零时，就不能接受比当前状态能量高的新状态。

这种接受准则通常称为**Metropolis准则**。

如此反复，达到系统在此温度下的热平衡。这个过程称作**Metropolis抽样过程**。

**Metropolis抽样过程**就是在一确定温度下，使系统达到热平衡的过程。



1. 假设开始状态在A，随着迭代次数更新到B的局部最优解，  
 $B < A$  说明接近最优解了，
2. 状态到达B后，发现下一步能量上升了，  
(如果是梯度下降则是不允许继续向前的)
3. 而模拟退火准则以一定的概率跳出这个坑  
**这个概率的设计十分重要**

## 退火过程（降温过程）

在Metropolis抽样过程中温度 $T$ 缓慢的降低。

模拟退火过程：

通过 $T$ 参数的变化使状态收敛于最小能量处。

- $T$ 参数的选择对于算法最后的结果有很大影响。
- 初始温度和终止温度设置的过低或过高都会延长搜索时间
- 降温步骤太快，往往会漏掉全局最优点，使算法收敛至局部最优点。
- 降温步骤太慢，则会大大延长搜索全局最优点的计算时间，从而难以实际应用。因此， $T$ 可以理解为一个控制参数。

# 模拟退火与组合优化问题的相似性

---

退火过程的状态 → 组合优化问题的解

能量 → 目标值

能量的取舍 → 目标值的取舍

能量的最小值 → 目标值的最小值

根据这种相似性，并依据Metropolis准则进行迭代就形成了  
模拟退火算法



# 优化问题与物理退火的类比

优化问题	物理退火
解	分子状态
最优解	能量最低状态
目标函数	能量
设定初始温度	熔解过程
Metropolis抽样	等温过程
温度的下降	冷却过程

## 物理退火过程

### 物理退火过程

- 物体内部的状态
- 状态的能量
- 温度
- 熔解过程
- 退火冷却过程
- 状态的转移
- 能量最低状态

## 类比关系

### 模拟退火算法

- 问题的解空间
- 解的质量
- 控制参数
- 设定初始温度
- 控制参数的修改
- 解在邻域中的变化
- 最优解

## 模拟退火算法

# 由固体退火到模拟退火：实现形式

---

模拟退火算法来源于固体退火原理：

1. 将固体加温至充分高，再使其以足够慢的速度冷却，
2. 用原子或晶格空位的移动来释放内部残留应力，
3. 通过这些原子排列重组的过程来消除材料中的差排。
  
4. 加温时，固体内部粒子随温度升高变为无序状，内能增大，
5. 而缓慢冷却时粒子趋于有序，在每个温度都达到平衡态，按照物理规律，
6. 最终在常温时达到基态，内能减为最小。

## 算法基本步骤:

1. 确定退火过程的温度初值  $T_0$  ,随机生成该温度下的粒子状态及一个解  $x_0$  , 计算目标函数值 $f$  即该状态下的内能  $E_{x_0}$  。
2. 进入下一个冷却温度。
3. 对当前的解  $x_i$  添加扰动, 及产生个新的解  $x_j$  , 计算相应的内能  $E_{x_i}$  及  $E_{x_j}$  , 根据 Metropolis准则判断是否接受新的解, 更新局部最优解和全局最优解。
4. 在不同的冷却温度  $T_i$  下, 重复m次对x的扰动, 即产生m个新的解, 并执行2~3.
5. 判断T是否已经到达了最低温度限制  $T_{limit}$  , 一般设定为  $0.01 \sim 5$  , 是则终止算法; 否继续执行2~4.

## 算法实质分为两层循环:

1. 在任一温度下随机扰动产生新解，计算目标函数值的变化，决定是否接受。由于算法初始温度比较高，这样使 $E$ 增大的新解在初始时也可能被接受，因此能跳出局部极小值，
2. 然后通过缓慢地降低温度，算法可能收敛到全局最优解。

### 1、新解的产生

要求尽可能地遍及解空间的各个区域，这样，在某一恒定温度下，不断产生新解时，就可能跳出局部最优解。

### 2、收敛的一般条件：

- 初始温度足够高；
- 热平衡时间足够长；
- 终止温度足够低；
- 降温过程足够缓慢；

### 3、参数的选择：

#### (1) 初始值 $T_0$

$T_0$ 越大越好，但为了减少计算量，要根据实际情况选择；

#### (2) 控制参数 $T$ 的衰减函数

常用 $T_{k+1} = aT_k$ ,  $a$  的取值范围：0.5~0.99；

#### (3) Markov链长度

$L_k = 100n$ ,  $n$ 为问题规模。

## 优点:

计算过程简单，通用，鲁棒性强，适用于并行处理，可用于求解复杂的非线性优化问题。

## 缺点:

收敛速度慢，执行时间长，算法性能与初始值有关及参数敏感等缺点。



# SA机理

---

- 优化问题的解视为固体的状态；
- 随机给定优化问题的初始解；
- 给定初始温度；
- 根据当前的解产生新的解；
- 依据Metropolis准则对两个解进行取舍；
- 重复以上两步直达到热平衡；
- 降低温度继续上述过程直到温度降到最低，最后的状态就认为是问题的解。

## ● 基本步骤

给定初温 $t=t_0$ ，随机产生初始状态 $s=s_0$ ，令 $k=0$ ；

**Repeat**

**Repeat**

产生新状态 $s_j=\text{Genete}(s)$ ；

if  $\min\{1, \exp[-(C(s_j)-C(s))/t_k]\} \geq \text{randrom}[0,1]$   $s=s_j$ ；

**Until** 抽样稳定准则满足；

退温 $t_{k+1}=\text{update}(t_k)$ 并令 $k=k+1$ ；

**Until** 算法终止准则满足；

输出算法搜索结果。

## ● 影响优化结果的主要因素

给定初温 $t=t_0$ ，随机产生初始状态 $s=s_0$ ，令 $k=0$ ；

Repeat

Repeat

产生新状态 $s_j=\text{Genete}(s)$ ;

if  $\min\{1, \exp[-(C(s_j)-C(s))/t_k]\} \geq \text{randrom}[0,1]$   $s=s_j$ ;

Until 抽样稳定准则满足;

退温 $t_{k+1}=\text{update}(t_k)$ 并令 $k=k+1$ ;

三函数两准则

Until 算法终止准则满足;

初始温度

输出算法搜索结果。

## 三函数两准则

状态产生函数

状态接受函数

退温函数

抽样稳定准则

退火结束准则

# 1 状态产生函数（邻域函数）

设计的出发点：

尽可能保证产生的候选解遍布全部解空间。

两部分 { 产生候选解的方式  
候选解产生的概率分布

---

候选解的产生方式由问题的性质决定，通常在当前状态的邻域结构内以一定概率方式产生，

而邻域函数和概率方式可以多样化设计，其中概率分布可以是均匀分布、正态分布、指数分布、柯西分布等。

## 2 状态接受函数 $\min[1, \exp(-\Delta C / t)]$

目的： 尽可能接受优化解

状态接受函数一般以概率的方式给出，不同接受函数的差别主要在于接受概率的形式不同。

# 设计状态接受概率，应遵循的原则：

---

- 固定温度下，接受使目标函数值下降的候选解的概率要大于使目标函数值上升的候选解的概率。
- 随温度的下降，接受使目标函数值上升的解的概率要逐渐减小。
- 当温度趋于零时，只能接受目标函数值下降的解。



### 3 初温 $t_0$

实验表明：初温值只要选择充分大，获得高质量解的概率就大！但花费计算时间增加。

- ✓初温的选择要足够高。
- ✓初温的确定应折衷考虑优化质量和优化效率。

---

初始温度

温度更新函数

内循环终止准则

外循环终止准则

退火历程

**(annealing schedule)**

## 4 温度更新函数

即温度的下降方式，用于在外循环中修改温度值。

### 衰减量 “以小为宜”

- ✓ 实验表明降温速度越慢，获得高质量解的几率越大，但花费的计算时间将同时增加。
- ✓ 温度高时下降的慢些，温度低时下降的快些。

$$t_{k+1} = \lambda t_k \quad 0 < \lambda < 1$$

$\lambda = 0.95$  Kirkpatrick首先提出  
被Johnson,Bonomi及Lutton采用  
 $\lambda$  取0.5至0.99之间

Nahar及Skiscim等人把  $[0, t_0]$  划分成K个小区间，温度更新函数为

$$t_k = \frac{K - k}{K} t_0, k = 1, 2, \dots, K$$

## 5 内循环终止准则 (Metropolis抽样稳定准则)

用于决定在各温度下产生候选解的数目。

非时齐SAA: 每个温度下只产生一个或少量候选解。

时齐算法——常用Metropolis抽样稳定准则包括:

- 检验目标函数值的均值是否稳定
- 连续若干步目标函数值的变化较小
- 按一定的步数抽样

---

具体应与问题规模成比例。

实验表明高温时迭代次数越多越好，低温时迭代次数可以适当减少。

## 6 外循环终止准则 (算法终止准则)

用于决定算法何时结束。

✓理论上要求温度终值趋于零  $e_r = \left| \frac{f^* - f_{opt}}{f_{opt}} \right|$

✓通常的做法包括:

设置终止温度的阈值

设置外循环迭代次数(6-50)

算法搜索到的最优值连续若干步保持不变

检验系统熵是否稳定

## 总结

三个参数  $t_0$  和  $L_k$  值均有显著影响。

过大的  $t_0$  值、 $\lambda$  值和  $L_k$  值均能导致过长的 CPU 时间。因而在最终解的质量有待较大

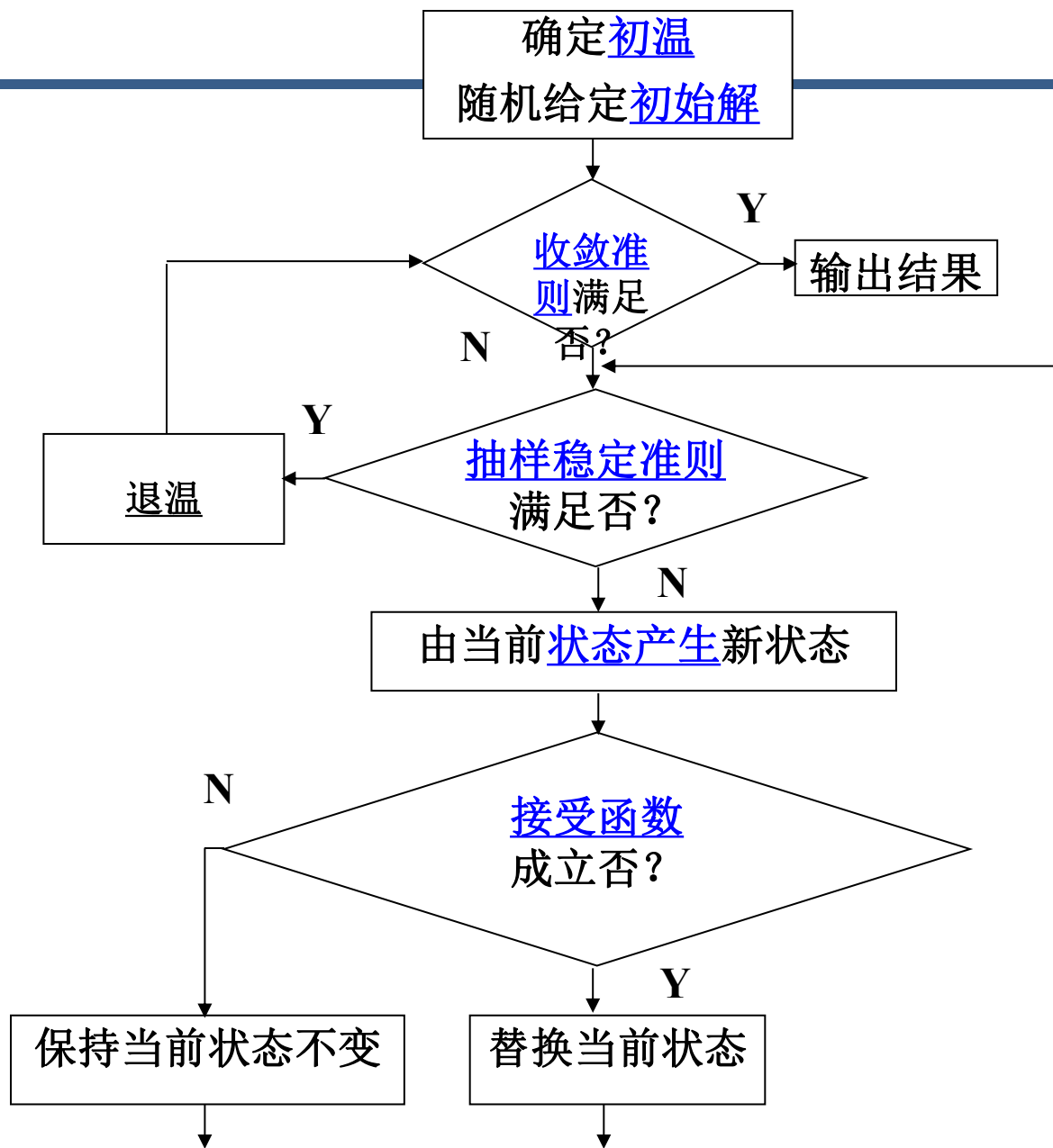
$t_0$  值和  $\lambda$  值予以保证的前提下，选取较小的  $L_k$  值可以抑制 CPU 时间上升的态势。



# SA流程

## 关键环节

- 1 初温、初始解
- 2 状态产生函数
- 3 状态接受函数
- 4 退温函数
- 5 抽样稳定准则
- 6 收敛准则



# 模拟退火算法基本要素和设定方法

功能意义	基本要素	设置方法
影响模拟退火算法全局搜索性能的重要因素之一。 实验表明，初温越大，获得高质量解的几率越大，但花费的计算时间将增加。	初始温度	<ol style="list-style-type: none"> <li>1、均匀抽样一组状态，以各状态目标值的方差定初温</li> <li>2、随机产生一组状态，以两两状态间最大差值定初温</li> <li>3、利用经验公式给出初温</li> </ol>
状态空间与状态产生函数。 邻域函数（状态产生函数）应尽可能保证产生的候选解遍布全部解空间。	邻域函数	<p>候选解一般采用按照某一概率密度函数对解空间进行随机采样来获得。</p> <p>概率分布可以是均匀分布、正态分布、指数分布等等</p>
指从一个状态 $X_k$ （一个可行解）向另一个状态 $X_{new}$ （另一个可行解）的转移概率，通俗的理解是接受一个新解为当前解的概率	接受概率	<p>一般采用Metropolis准则</p> $P_{ij}^T = \begin{cases} 1, & \text{if } E(j) \leq E(i) \\ e^{-\frac{E(j)-E(i)}{KT}} = e^{-\frac{\Delta E}{KT}}, & \text{otherwise} \end{cases}$
指从某一较高温状态 $t_0$ 向较低温状态冷却时的降温管理表，或者说降温方式	冷却控制	<ol style="list-style-type: none"> <li>1、经典模拟退火算法的降温方式 <math>t_k = \frac{t_0}{\lg(1+k)}</math></li> <li>2、快速模拟退火算法的降温方式 <math>t_k = \frac{t_0}{1+k}</math></li> </ol>
内层平衡也称Metropolis抽样稳定准则，用于决定在各温度下产生候选解的数目	内层平衡	<ol style="list-style-type: none"> <li>1、检验目标函数的均值是否稳定</li> <li>2、连续若干步的目标值变化较小</li> <li>3、预先设定的抽样数目，内循环代数</li> </ol>
算法的终止条件	终止条件	<ol style="list-style-type: none"> <li>1、设置终止温度的阈值</li> <li>2、设置外循环迭代次数</li> <li>3、算法搜索到的最优值连续若干步保持不变</li> <li>4、检验系统熵是否稳定</li> </ol>

# SA特点

---

- 可以保证全局最优
- 特别适合组合优化问题
- 可以随机选择初始解
- 简单易行，通用性好

---

模拟退火算法在求解规模较大的实际问题时，往往存在以下缺点：

（1）收敛速度比较慢。

（2）尽管理论上只要计算时间足够长，模拟退火法就可以保证以概率1收敛于全局最优点。但是在实际算法的实现过程中，由于计算速度和时间的限制，在优化效果和计算时间二者之间存在矛盾，因而难以保证计算结果为全局最优点，优化效果不甚理想。

（3）在每一温度下很难判定是否达到了平衡状态。

## 4.1 模拟退火算法的优缺点

- **模拟退火算法的优点**

质量高；

初值鲁棒性强；

简单、通用、易实现。

- **模拟退火算法的缺点**

由于要求较高的初始温度、较慢的降温速率、较低的终止温度，以及各温度下足够多次的抽样，因此优化过程较长。

## 5.2 算法的实现

### 1、组合优化问题的求解

#### 数学模型

一个商人欲到  $n$  个城市推销商品，每两个城市  $i$  和  $j$  之间的距离为  $d_{ij}$ ，如何选择一条道路使得商人每个城市走一遍后回到起点且所走路径最短。

## 解空间

$$S = \{(\pi_1, \pi_2, \dots, \pi_n) \mid (\pi_1, \pi_2, \dots, \pi_n) \text{ 为 } (1, 2, \dots, n) \text{ 的循环排列} \}$$

## 目标函数

$$f(\pi_1, \pi_2, \dots, \pi_n) = \sum_{i=1}^n d_{\pi_i \pi_{i+1}}$$

## 初始解

选为  $(1, 2, \dots, n)$

## 新解的产生

(5 4 1 7 9 8 6 2 3)

1、互换操作(SWAP) (5 8 1 7 9 4 6 2 3)

随机交换两个不同城市的位置。

2、逆序操作(INV) (5 8 9 7 1 4 6 2 3)

两个不同随机位置的城市逆序。

3、插入操作(INS) (5 8 4 1 7 9 6 2 3)

随机选择某个城市插入到不同随机位置。



初值  $t_0 = \frac{-\Delta f}{\ln p} \quad p = 0.8$

衰减函数  $t_{k+1} = \lambda t_k \quad 0 < \lambda < 1$

马氏链长  $L_k = 20$

停止准则

设计终止温度的阈值

设计外循环迭代次数

算法搜索到的最优值连续若干步保持不变

# 例1 30城市TSP问题 ( $d^*=423.741$ by D B Fogel)

## ● TSP Benchmark 问题

41 94;37 84;54 67;25 62;

7 64;2 99;68 58;71 44;54

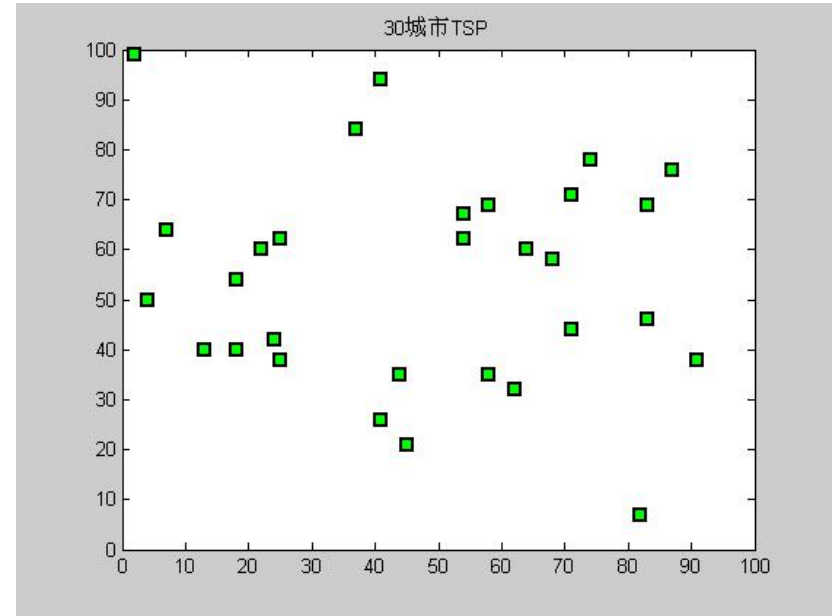
62;83 69;64 60;18 54;22

60;83 46;91 38;25 38;24

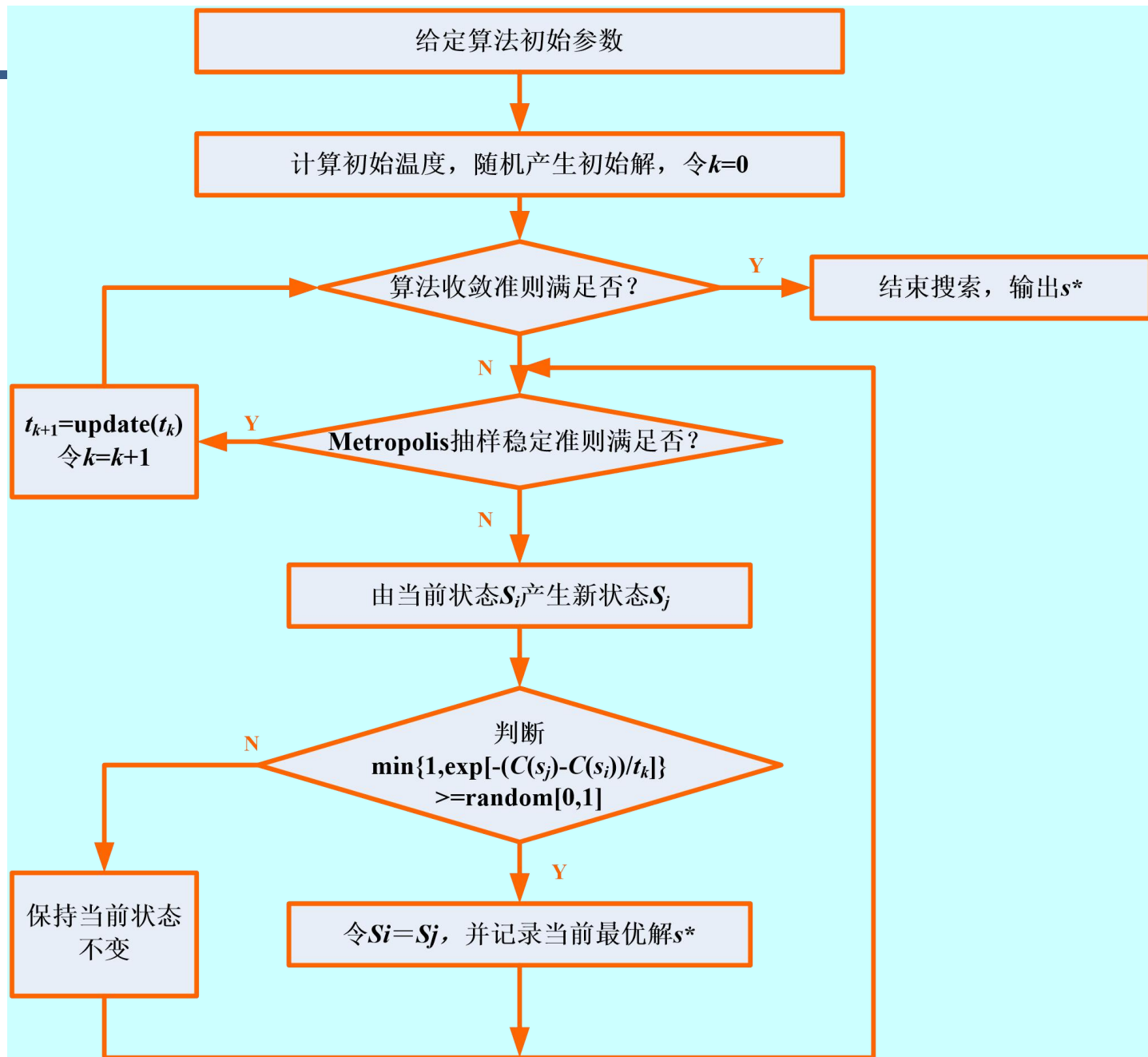
42;58 69;71 71;74 78;87

76;18 40;13 40;82 7;62 32;

58 35;45 21;41 26;44 35;4 50



# ● 算法流程



---

- 初始温度的计算

```
for i=1:100
```

```
    route=randperm(CityNum);
```

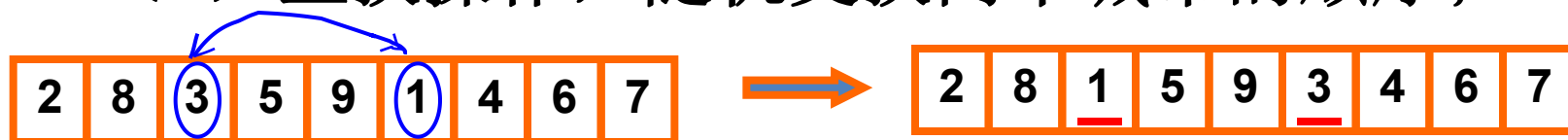
```
    fval0(i)=CalDist(dislist,route);
```

```
end
```

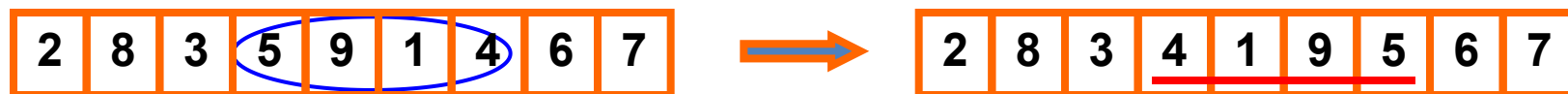
```
t0=-(max(fval0)-min(fval0))/log(0.9);
```

## ● 状态产生函数的设计

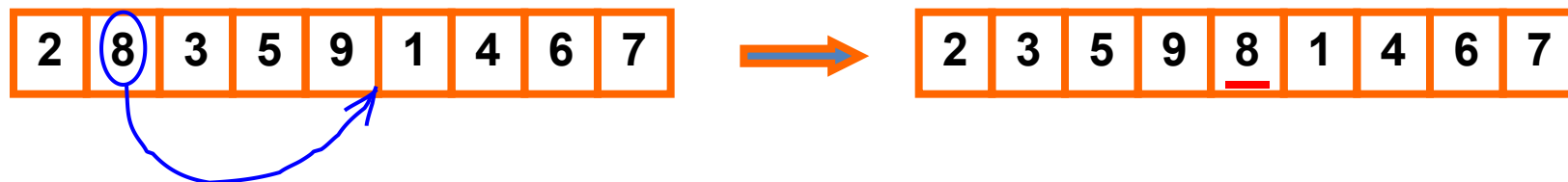
(1) 互换操作，随机交换两个城市的顺序；



(2) 逆序操作，两个随机位置间的城市逆序；



(3) 插入操作，随机选择某点插入某随机位置。



---

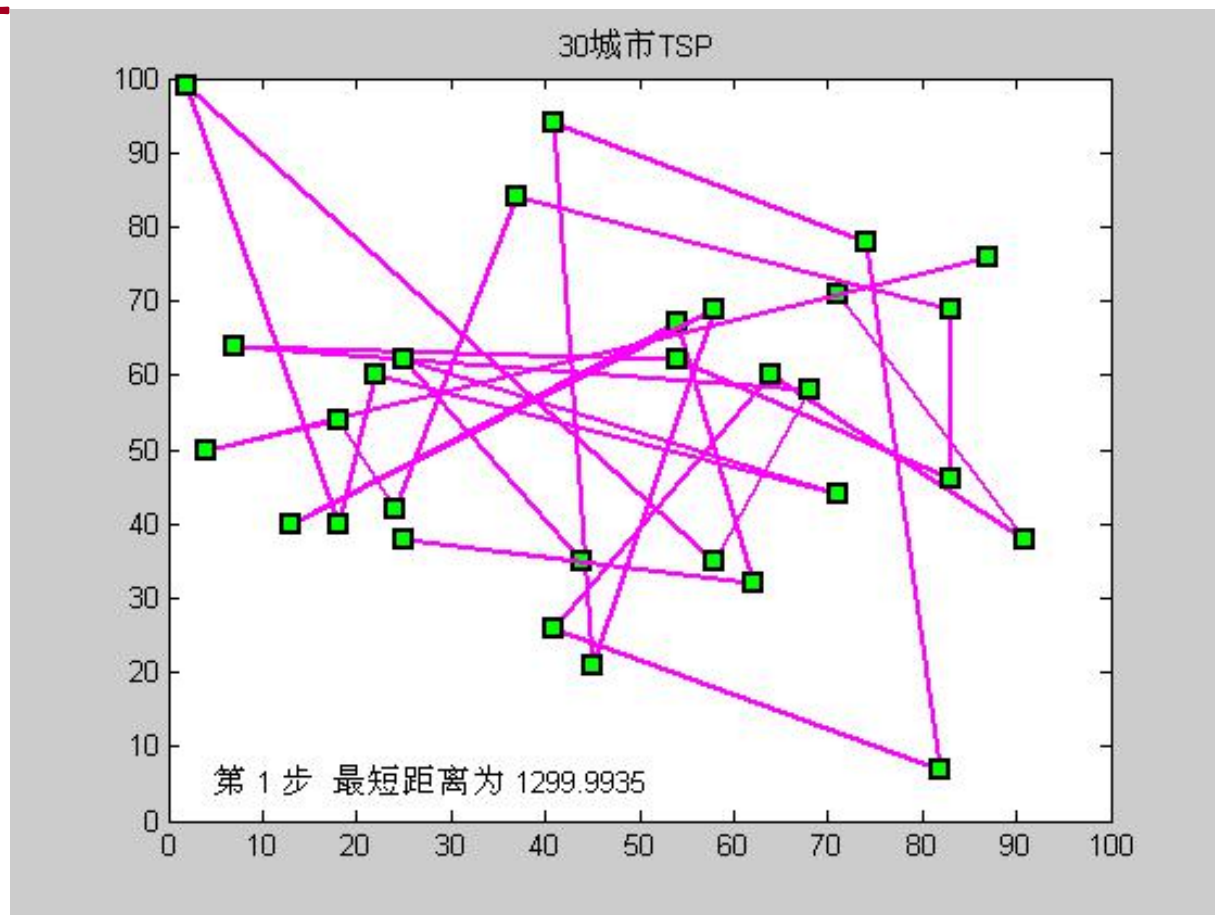
- **参数设定**

截止温度  $tf=0.01;$

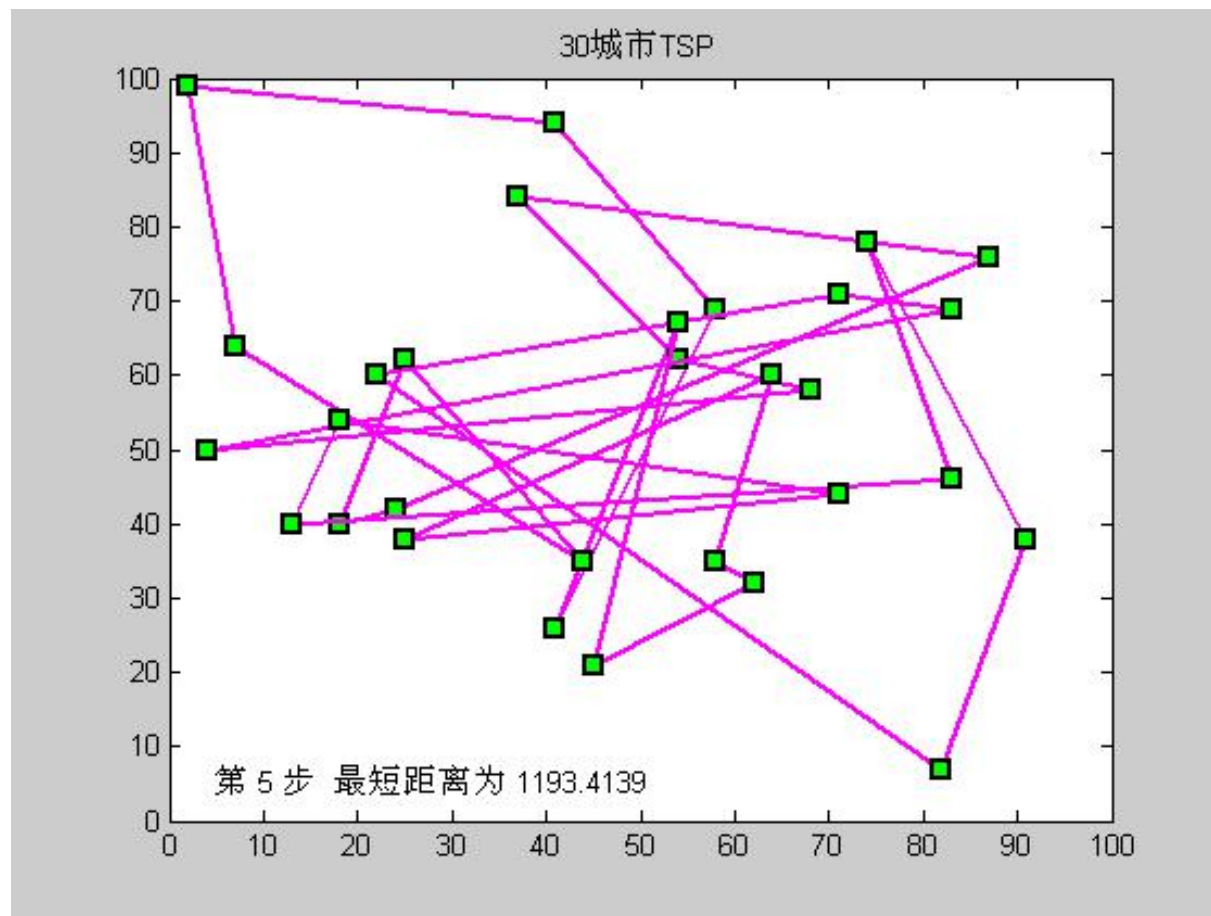
退温系数  $\alpha=0.90;$

内循环次数  $L=200*CityNum;$

- 运行过程

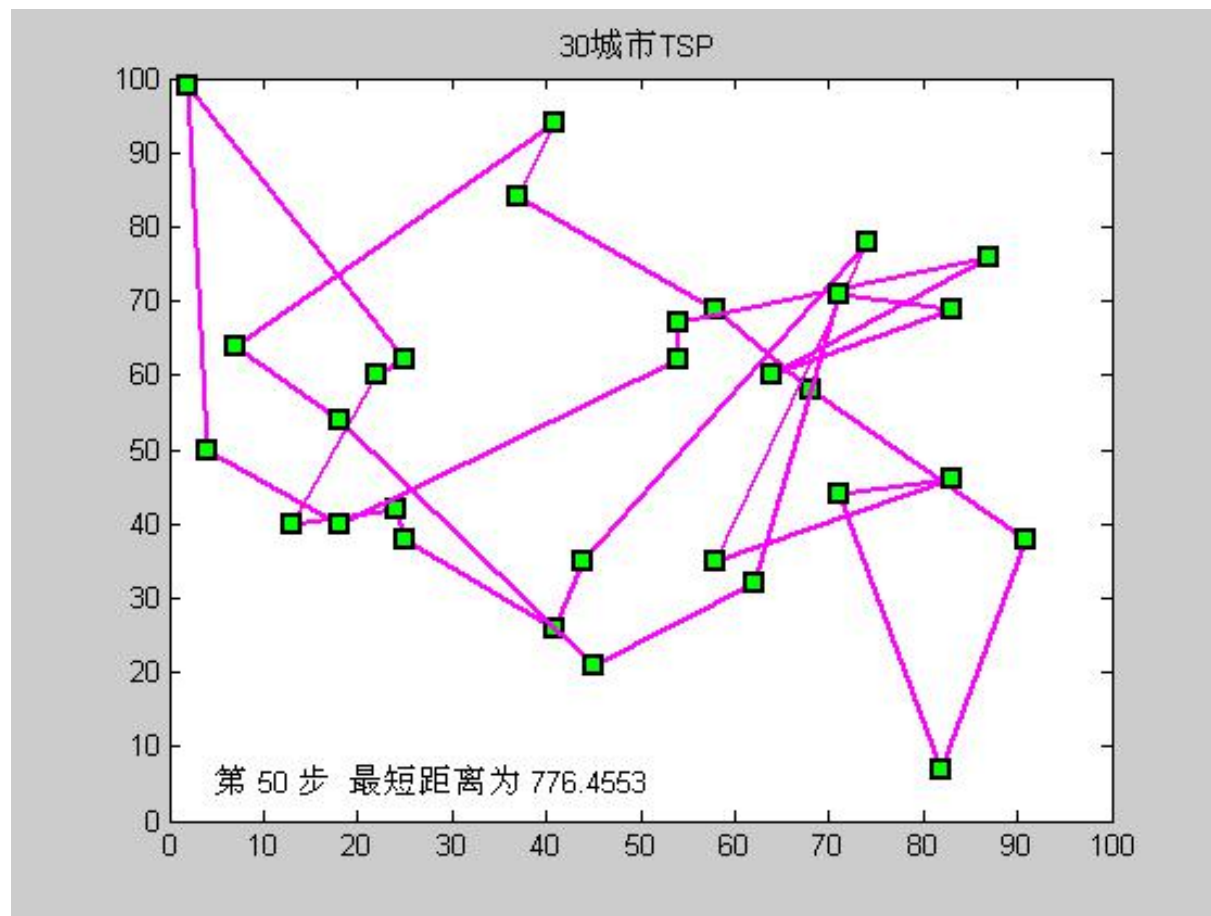


- 运行过程

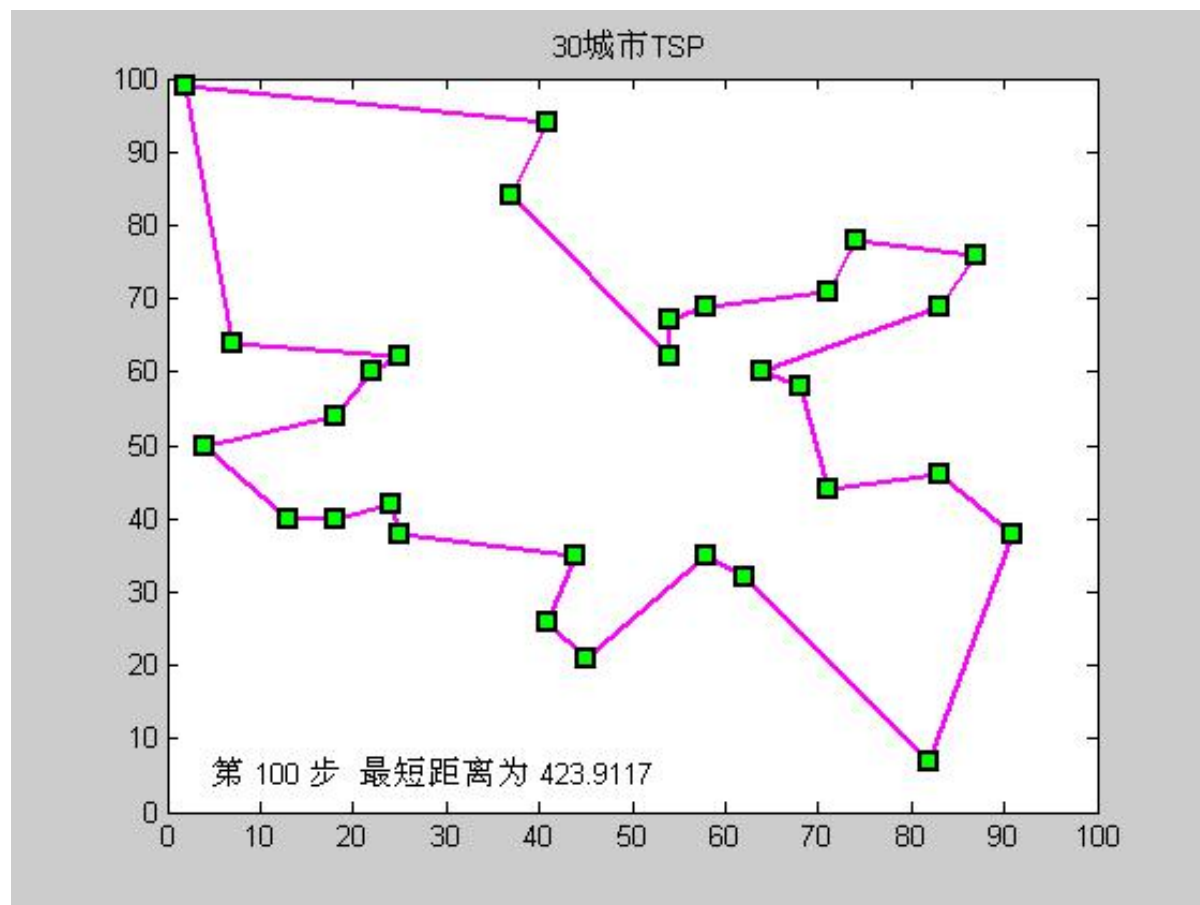




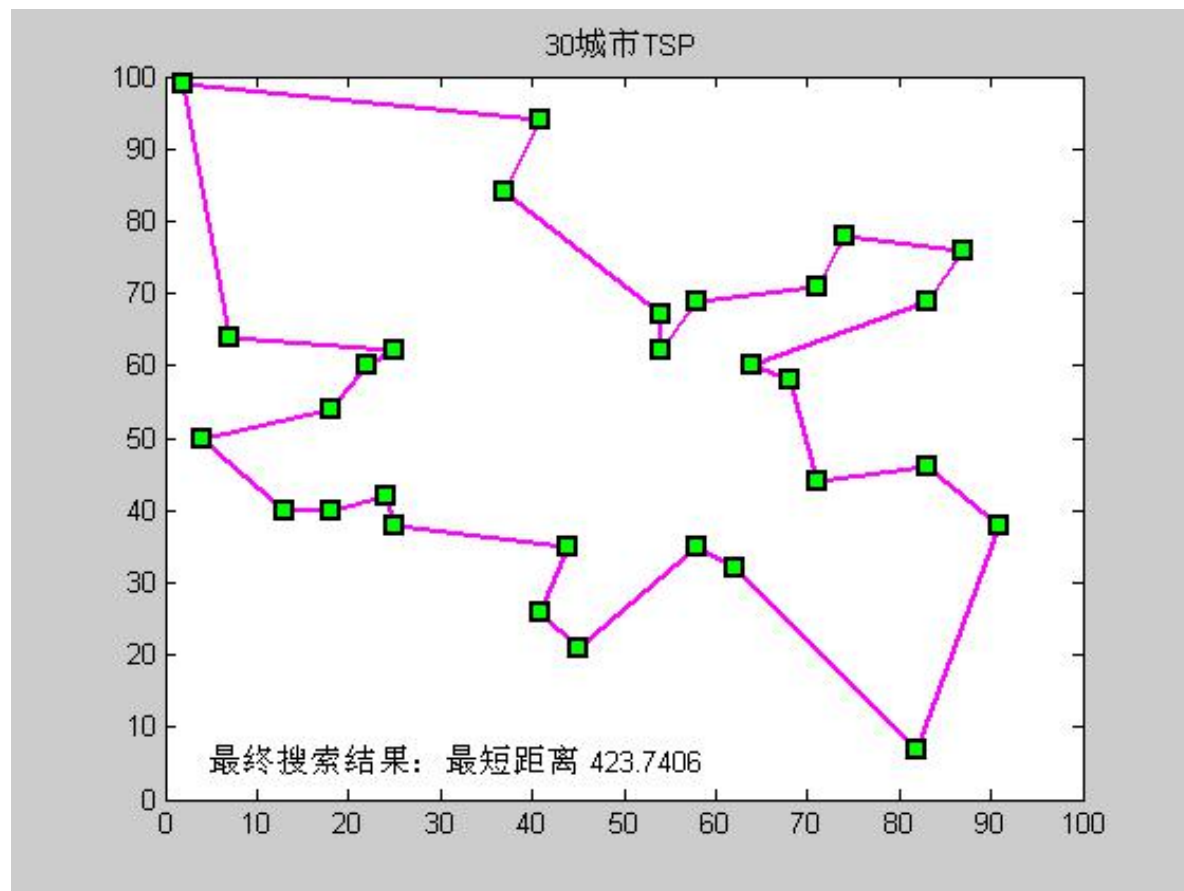
- 运行过程



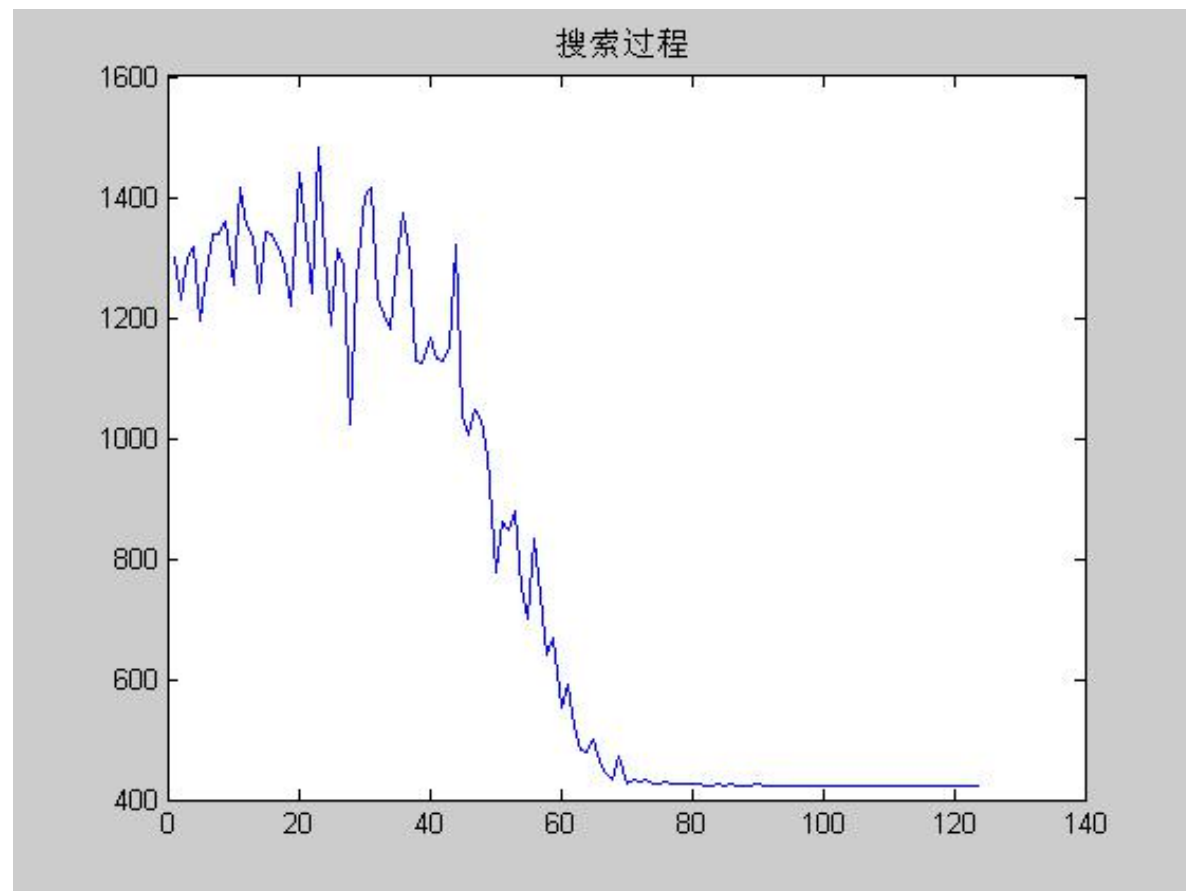
- 运行过程



## ● 运行过程



- 运行结果



## 例2 模拟退火求解背包问题

已知背包的装载量为 $c=8$ ，现有 $n=5$ 个物品，它们的重量和价值分别是 $(2, 3, 5, 1, 4)$ 和 $(2, 5, 8, 3, 6)$ 。试使用模拟退火算法求解该背包问题，写出关键的步骤。

求解：假设问题的一个可行解用0和1的序列表示，例如 $i=(1010)$ 表示选择第1和第3个物品，而不选择第2和第4个物品。用模拟退火算法求解此例的关键过程如图所示：

# 运行步骤

已知：

物体个数：  $n=5$

背包容量：  $c=8$

重量  $w = (2, 3, 5, 1, 4)$

价值  $v = (2, 5, 8, 3, 6)$

第一步：初始化。假设初始解为  $i=(11001)$ ，初始温度为  $T=10$ 。计算  $f(i)=2+5+6=13$ ，最优解  $s=i$

第三步：降温，假设温度降为  $T=9$ 。如果没有达到结束标准，则返回第二步继续执行

假设在继续运行的时候，从当前解  $i=(10110)$  得到一个新解  $j=(00111)$ ，这时候的函数值为  $f(j)=8+3+6=17$ ，这是一个全局最优解。可见上面过程中接受了劣解是有好处的。

第二步：在  $T$  温度下局部搜索，直到“平衡”，假设平衡条件为执行了3次内层循环。

(2-1) 产生当前解  $i$  的一个邻域解  $j$ （如何构造邻域根据具体的问题而定，这里假设为随机改变某一位的0/1值或者交换某两位的0/1值），假设  $j=(11100)$   
要注意产生的新解的合法性，要舍弃那些总重量超过背包装载量的非法解

(2-2)  $f(j) = 2+5+8=15 > 13=f(i)$ ，所以接受新解  $i=j$ ； $f(i)=f(j)=15$ ；而且  $s=i$ ；  
要注意求解的是最大值，因此适应值越大越优

(2-3) 返回 (2-1) 继续执行。

(a) 假设第二轮得到的新解  $j=(11010)$ ，由于  $f(j) = 2+5+3=10 < 15=f(i)$ ，所以需要计算接受概率  $P(T)=\exp((f(j)-f(i))/T) = \exp(-0.5) = 0.607$ ，假设  $\text{random}(0,1) > P(T)$ ，则不接受新解

(b) 假设第三轮得到的新解  $j=(10110)$ ，由于  $f(j) = 2+8+3=13 < 15=f(i)$ ，所以需要计算接受概率  $P(T)=\exp((f(j)-f(i))/T) = \exp(-0.3) = 0.741$ ，假设  $\text{random}(0,1) < P(T)$ ，则接受新解  
按照一定的概率接受劣解，也是跳出局部最优的一种手段

(2-4) 这时候， $T$  温度下的“平衡”已达到（即已经完成了3次的邻域产生），结束内层循环

# 函数优化问题的求解

求解  $f = x + 10\sin(5x) + 7\cos(4x)$  在  $x \in [0, 9]$  上的最大值

```
% 目标函数
```

```
function f=target(x)
```

```
f=x + 10 * sin(5 .* x) + 7 * cos(4 .* x);
```

```
end
```

```
lower_bound = 0;
```

```
higher_bound = 9;
```

# 1参数设置

```
x_new = lower_bound+rand*(higher_bound-lower_bound);  
x_new = x_new*0.1;    % 从x较小开始搜索，直至找到最值
```

```
x_current = x_new;    % 初始化局部最优x  
x_best = x_new;       % 初始化全局最优x  
y_current = inf;      % 初始化局部最优y  
y_best = inf;         % 初始化全局最优y
```

```
t = 1000;  
tf = 3;  
ratio = 0.9;  
Markov_length = 20;
```



## 2 程序内外循环

```
while t>tf
```

```
    for i=1:Markov_length
```

内循环：温度  $t$  不变的情况下，执行

```
        end
```

**Metropolis**准则判断

```
        if flag==0
```

```
            break
```

**flag**：控制内循环程序停止

```
        else
```

外循环：温度  $t$  不断的降低

```
            t=t*ratio;
```

```
        end
```

```
    end
```

### 3 程序内循环：温度 $t$ 不变，执行Metropolis采样 准则判断

```
for i=1:Markov_length
```

```
    x_new = x_new+new;  
    if (x_new<lower_bound||x_new>higher_bound)  
        x_new = x_new-3*new;  
    end
```

```
    y_new=-target(x_new);      % 寻找最大值，
```

```
% Metroplis准则寻找最大值
```

```
    flag=y_current;           % 用于控制退出循环条件  
    if y_new < y_current  
        x_current = x_new;  
        y_current = y_new;  
        if y_new < y_best  
            x_best = x_new;  
            y_best = y_new;  
        end  
    else  
        if rand<exp(-(y_new-y_current)/t)  
            x_current = x_new;  
            y_current = y_new;  
        else  
            x_new=x_current;  
        end  
    end
```

```
end
```

解的产生：在其领域内的扰动

**flag:** 控制内循环程序停止

```
    if abs(flag-y_current)<0.001  
        flag=0;  
        break  
    end
```

# 关于优化问题

---

## 传统的优化方法（局部优化）

共轭梯度法、拟牛顿法、单纯形方法

## 全局优化方法

漫步法（Random Walk）、模拟退火法、GA

## 模拟退火与传统迭代最优算法的比较：

（1）当系统在非零温度下时，从局部最优中跳出是非常可能的，因此不会陷入局部最优。

（2）系统最终状态的总特征可以在较高温度下看到，而状态的好的细节却在低温下表现，因此，模拟退火是自适应的。

## 传统的优化方法

- 1、依赖于初始条件。
- 2、与求解空间有紧密关系，促使较快地收敛到局部解，但同时解域有约束，如可微或连续。利用这些约束，收敛快。
- 3、有些方法，如Davidon-Fletcher-Powell直接依赖于至少一阶导数；共轭梯度法隐含地依赖于梯度。

## 全局优化方法

- 1、不依赖于初始条件；
- 2、不与求解空间有紧密关系，对解域，无可微或连续的要求。求解稳健，但收敛速度慢。能获得全局最优。适合于求解空间不知的情况