高级搜索

- 一般认为，NP完全问题的算法复杂性是指数级的。当问题规模达到一定程度时，以前这些算法显得无能为力。
- 局部搜索算法、模拟退火算法和遗传算法等是较新发展起来的算法，算法引入了随机因素，不一定能找到最优解，但一般能快速找到满意的解。
- 主要内容：

✓ 基本概念

✓ 局部搜索算法

✓ 模拟退火算法

组合优化问题

- 优化问题

  设x的决策变量，D为x的定义域，$f(x)$是指标函数，$g(x)$是约束条件集合。则优化问题可以表示为，求解满足$g(x)$的$f(x)$最小值问题。即：

$$\min_{x \in D}(f(x)|g(x))$$

- 组合优化问题

  如果在定义域上，满足条件$g(x)$的解是有限的，则优化问题称为组合优化问题。

- 现实世界中很多问题属于组合优化问题，或者可以转化为组合优化问题求解，如旅行商问题、皇后问题。

组合优化问题举例

- TSP问题

  从某个城市出发，经过n个指定的城市，每个城市只能且必须经过一次，最后回到出发城市，如何安排旅行商的行走路线以使总路程最短？

- 约束机器排序问题

  n个加工量为$d_i(i=1,2,\ldots\ n)$的产品在一台机器上加工，机器在第t个时段的工作能力为c_t,完成所有产品加工的最少时段数。

- 指派问题

  一家公司经理准备安排N名员工去完成N项任务，每人一项。由于各员工的特点不同，不同的员工去完成同一项任务时获得的回报是不同的。如何分配工作方案可以获得最大收益？

组合优化问题举例

- 0-1背包问题

  设有一个容积为b的背包，n个体积分别为$a_i(i=1,2,\ldots\ n)$,价值分别为$c_i\ (i=1,2,\ldots\ n)$的物品，如何以最大的价值装包？

- 装箱问题

  如何用个数最少的尺寸为1的箱子装进n个尺寸不超过1的物品？

- SAT问题

  称 判定一个公式是否存在一个模型的问题为可满足性问题(以后简称为SAT问题)。如果一个公式存在模型，则称该公式是可满足的，否则称为不可满足的。

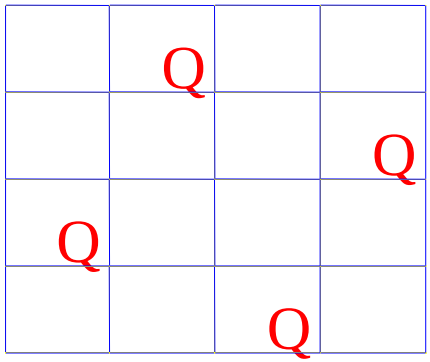
组合优化问题举例

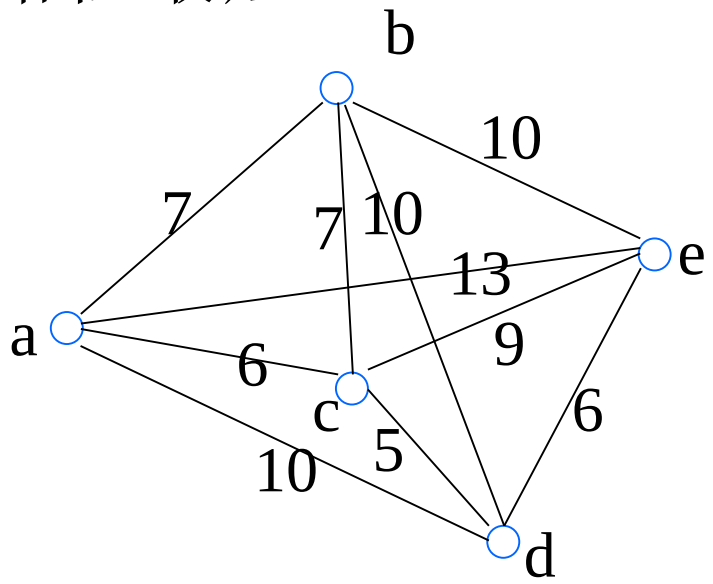
- 皇后问题

  在n×n的国际象棋棋盘上，摆放n个皇后，使得n个皇后之间不能相互“捕捉”？

- TSP问题

  从某个城市出发，经过n个指定的城市，每个城市只能且必须经过一次，最后回到出发城市，如何安排旅行商的行走路线以使总路程最短？





问题规模与算法复杂度

- 问题的规模通常用输入数据量n来衡量。如旅行商问题的城市数目、皇后问题的皇后数目等。

- 当问题规模比较小时，由于组合优化问题的解是有限的，总可以通过枚举法获得问题的最优解。但当问题的规模比较大时，其状态数往往呈指数级增长，很难再通过枚举方法获得问题的解。

- 对于同一个问题，不同的求解方法，其效率是不同的，差别可能会非常大。

- 通常用算法的时间复杂度来评价一个求解方法的好坏。

常用的算法复杂性函数

- 多项式时间算法

  $O(\log n)$、$O(n)$、$O(n\log n)$）、$O(n^2)$

- 指数时间算法

  $O(2^n)$、$O(n!)$、$O(n^n)$

- 旅行商问题和皇后问题，用枚举法的时间复杂度为$O(n!)$

- 完备算法的复杂性在通常情况下仍是指数型的。

- 对指数时间算法，当问题的规模大到一定程度时，因为所花费的时间太长了，以至于不能求解。

时间复杂性函数比较-数值说明

$h(n)$ \ n	10	20	30	40	100
n	10ns	20ns	30ns	40ns	100ns
$n\log n$	10ns	26ns	44.3ns	64.1ns	200ns
n^2	100ns	400ns	900ns	1.6us	10us
2^n	1.0us	1.9ms	1.1s	18.3min	4.0世纪
$n!$	3.6ms	77.1年	8.4×10^{13}世纪	2.6×10^{29}世纪	3.0×10^{139}世纪

大规模组合优化问题的求解策略

- 一些难以求解的组合优化问题，至今尚未找到多项式时间算法来获得问题的最优解，如旅行商问题。
- 实际上，在很多情况下追求最优解不一定有意义，一个满意解就可以了。如买西瓜。
- 求满意解的基本思想是：引入随机因素，每次运行并不能保证求得问题的最优解，但经过多次运行后，一般总能得到一个与最优解相差不太大的满意解，以放弃每次必然找到最优解的目标，来换取算法时间复杂度的降低，以适合于求解大规模的优化问题。

邻域的概念

- 在组合优化问题中，变量x的一个取值看作一个点；如果是多变量的问题，一组变量的一个取值组合可以看作一个点，即:点就是一个候选解。
- 邻域，简单地说就是一个点附近的其他点的集合。在距离空间中，邻域一般定义为以该点为中心的一个圆。
- 在组合优化问题中，可将邻域定义为：

设D是问题的定义域，若存在一个映射N，使得：

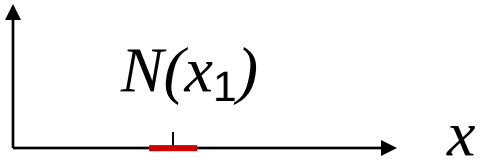
$$N:\ S\in D\rightarrow N(S)\in 2^{D}$$

则称$N(S)$为S的邻域，称$S'\in N(S)$为S的邻居。

常见的几种邻域

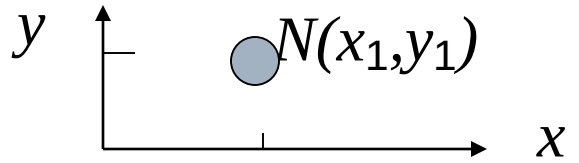
- 一维空间中

  点x_1附近的一个小区间



- 二维空间中

  点$<x_1,y_1>$附近的一个小区域



- 三维或多维空间中

- 邻域可定义为该点为中心的一个圆球体，即

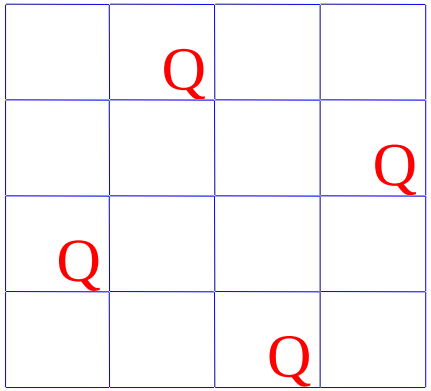
$$N(<x'_1,\ldots,x'_n>)=\{<x_1,\ldots,x_n>: (x_1-x'_1)^2+\ldots+(x_n-x'_n)^2=r\}$$

- 或定义与该点的欧氏距离，即：

$$N(<x'_1,\ldots,x'_n>)=\{<x_1,\ldots,x_n>: |(x_1-x'_1)|+\ldots+|(x_n-x'_n)|=d\}$$

四皇后问题的邻域

- 定义S=(s_i)表示四皇后问题的一个可能的解，其中s_i表示在第i行、第s_i列有一个皇后，如图，有S=(2，4，1，3)



- 定义映射N为棋盘上任意两个皇后的所在行或列进行交换。
- S的邻居共有6个，所有邻居的集合就是S的邻域。
- 如图，N（S）={（4，2，1，3），（1，4，2，3），（3，4，1，2），（2，1，4，3），（2，3，1，4），（2，4，3，1）}

旅行商问题的邻域

- 用一个城市序列表示一个可能的解，
- 通过交换两个城市的位置S的邻居。
- 设$S=(x_1,x_2,\ldots,x_{i-1},x_i,x_{i+1},\ldots,x_{j-1},x_j,x_{j+1},\ldots,x_n)$。则通过交换$x_i$和$x_j$两个城市的位置可以得到S的一个邻居：

  $S'=(x_1,x_2,\ldots,x_{i-1},x_j,x_{i+1},\ldots,x_{j-1},x_i,x_{j+1},\ldots,x_n)$
- 也可以采用逆序的方式获取S的邻居，即通过交换x_i和x_j两个城市之间的城市次序来得到S的邻居：

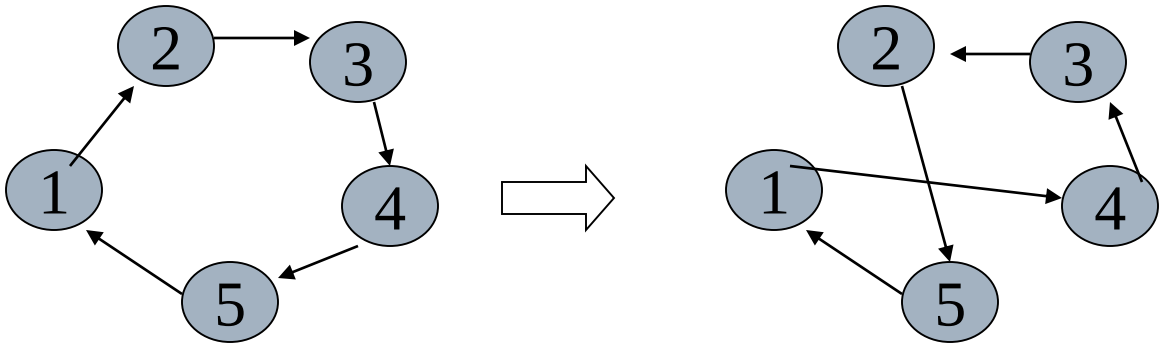
  $S'=(x_1,x_2,\ldots,x_{i-1},x_i,x_{j-1},\ldots,x_{i+1},x_j,x_{j+1},\ldots,x_n)$

旅行商问题的邻域-示例

- 以5城市TSP为例，可以用一个城市序列S表示一个可能解$<c_1,c_2,c_3,c_4,c_5>$，其中c_i表示第i个城市。
- 定义映射N为交换城市序列中的任意两个城市，即S中任意两个元素交换位置，这样可得到S的所有邻居，所有邻居的集合就是S的邻域。
- 以S=<1,2,3,4,5,>为例，其邻域$N(S)$ = {<2,1,3,4,5>, <3,2,1,4,5>, <4,2,3,1,5>,<5,2,3,4,1>, <1,3,2,4,5>, <1,4,3,2,5>, <1,5,3,4,2>, <1,2,4,3,5>, <1,2,5,4,3>, <1,2,3,5,4>}

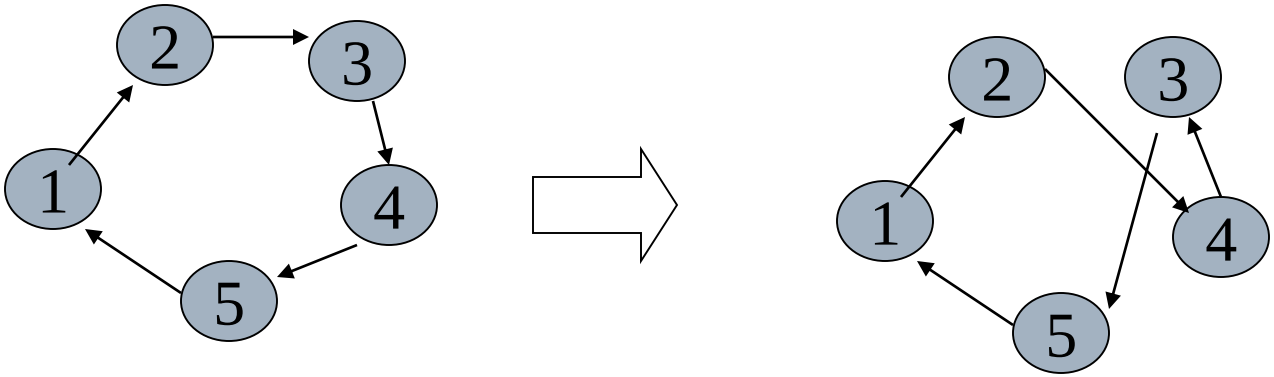
旅行商问题的邻域-示例

- 以<1,2,3,4,5,>交换为<1,4,3,2,5>后路径变化示意图如下：



旅行商问题的邻域-示例

- 还可以定义映射N'为逆序交换获得S的所有邻居，即S中任意两个元素之间的城市逆序重排。
- 以S=<1,2,3,4,5,>为例，将城市2和5之间的城市逆序重排，得：<1,2,4,3,5>
- 以<1,2,3,4,5,>交换为<1,2,4,3,5>后路径变化示意图如下：



局部最优与全局最优

- 在一个邻域内的最优解成为局部最优解
- 在整个定义域上的最优解成为全局最优解
- 最优解可以是求最大值，也可以是求最小值，思想是一样的。以后的论述中，一般假定求最小值。

局部搜索算法

- 局部搜索算法是从爬山法改进而来的。
- 爬山法：在没有任何有关山顶的其他信息的情况下，沿着最陡的山坡向上爬。
- 局部搜索算法的基本思想：在搜索过程中，始终选择当前点的邻居中与离目标最近者的方向搜索。

爬山算法

1, $n := s$;

2, LOOP: IF GOAL(n) THEN EXIT(SUCCESS);

3, EXPAND(n) →{m_i},计算$h(m_i)$,
next$n$=$min\{h(m_i)\}$

4, IF $h(n)$<h(next$n$) THEN EXIT(Fail);

5, n:=next$n$;

6, GO LOOP;

该算法在单峰的条件下，必能达到山顶。

局部搜索算法

（1）随机选择一个初始的可能解$x_0 \in D$，$x_b=x_0$,$P=N(x_b)$;

//D是问题的定义域，x_b用于记录到目标位置的最优解，P为x_b的邻域。

（2）如果不满足结束条件，则：*//结束条件为循环次数或P为空等*

（3）Begin

（4）选择P的一个子集P'，x_n为P'的最优解

// P'可根据问题特点，选择适当大小的子集。可按概率选择

（5）如果$f(x_n)<f(x_b)$，则$x_b=x_n$，$P=N(x_b)$，转(2)

// 重新计算P，f(x)为指标函数

（6）否则$P=P-P'$，转(2)

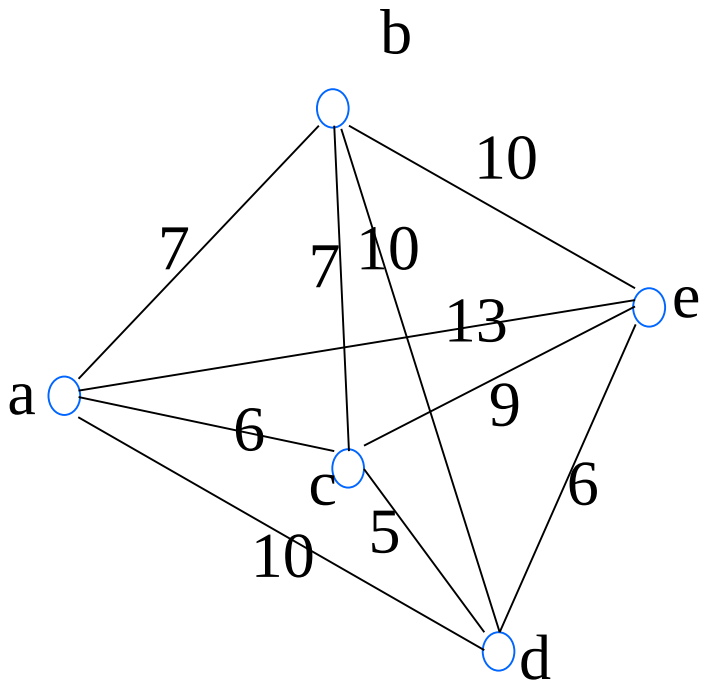
（7）End

（8）输出计算结果

（9）结束

用局部搜索方法求解TSP问题

- 假设从a出发，用一个城市序列表示一个可能的解
- 设初始生成的可能解为 $x_0=(a,b,c,d,e)$，则 $f(x_b)=f(x_0)=38$
- 选择两个城市间的位置交换方式来得到一个可能解的邻域，并在算法第4步选择从P中随机选择一个元素的方法，则 $P=\{(a,c,b,d,e)，(a,d,c,b,e)，(a,e,c,d,b)，(a,b,d,c,e)，(a,b,e,d,c)，(a,b,c,e,d)\}$



求解TSP问题-过程

- 第1次循环

P'={从P中随机选择一个元素},设为$x_1=(a,c,b,d,e)$,则
$f(x_1)=42$，　$f(x_1)> f(x_b)$,P=P- $\{x_1\}$={ (a,d,c,b,e), (a,e,c,d,b), (a,b,d,c,e), (a,b,e,d,c), (a,b,c,e,d)}

- 第2次循环

P'={从P中随机选择一个元素},设为$x_2= (a,d,c,b,e)$,则
$f(x_2)=45$，　$f(x_2)> f(x_b)$, P=P- $\{x_2\}$={ (a,e,c,d,b), (a,b,d,c,e), (a,b,e,d,c), (a,b,c,e,d)}

- 第3次循环

P'={从P中随机选择一个元素},设为$x_3= (a,e,c,d,b)$,则
$f(x_3)=44$，　$f(x_3)> f(x_b)$, P=P- $\{x_3\}$={ (a,b,d,c,e), (a,b,e,d,c), (a,b,c,e,d)}

求解TSP问题-过程

- 第4次循环

P’={从P中随机选择一个元素},设为x_4= (a,b,d,c,e),则
$f(x_4)$=44， $f(x_4)> f(x_b)$, P=P- $\{x_4\}$={ (a,b,e,d,c), (a,b,c,e,d)}

- 第5次循环

P’={从P中随机选择一个元素},设为x_5= (a,b,e,d,c),则
$f(x_5)$=34， $f(x_5)< f(x_b)$, x_b= (a,b,e,d,c),P=$N(x_5)$={ (a,e,b,d,c), (a,d,e,b,c), (a,c,e,d,b), (a,b,d,e,c), (a,b,c,d,e), (a,b,e,c,d)}

- 第6次循环

P’={从P中随机选择一个元素},设为x_6= (a,e,b,d,c),则
$f(x_6)$=44， $f(x_6)>f(x_b)$, P=P-$\{x_6\}$={ (a,d,e,b,c), (a,c,e,d,b), (a,b,d,e,c), (a,b,c,d,e), (a,b,e,c,d)}

求解TSP问题-过程

- 第7次循环

P’={从P中随机选择一个元素},设为x_7= (*a*,*d*,*e*,*b*,*c*),则
$f(x_7)$=39， $f(x_7)>f(x_b)$, P=P-{x_7}={ (*a*,*c*,*e*,*d*,*b*), (*a*,*b*,*d*,*e*,*c*), (*a*,*b*,*c*,*d*,*e*), (*a*,*b*,*e*,*c*,*d*)}

- 第8次循环：

P’={从P中随机选择一个元素},设为x_8= (*a*,*c*,*e*,*d*,*b*),则
$f(x_8)$=38, $f(x_8)>f(x_b)$, P=P-{x_8}={ (*a*,*b*,*d*,*e*,*c*), (*a*,*b*,*c*,*d*,*e*), (*a*,*b*,*e*,*c*,*d*)}

- 第9次循环

P’={从P中随机选择一个元素},设为x_9= (*a*,*b*,*d*,*e*,*c*),则
$f(x_9)$=38, $f(x_9)>f(x_b)$, P=P-{x_9}={ (*a*,*b*,*c*,*d*,*e*), (*a*,*b*,*e*,*c*,*d*)}

求解TSP问题-过程

- 第10次循环

  P’={从P中随机选择一个元素},设为x_{10}= (a,b,c,d,e),则
  $f(x_{10})$=38, $f(x_{10})$>$f(x_b)$, P=P-{x_{10}}={ (a,b,e,c,d)}

- 第11次循环

  P’={从P中随机选择一个元素},设为x_{11}= (a,b,e,c,d),则
  $f(x_{11})$=41, $f(x_{11})$>$f(x_b)$, P=P-{x_{11}}={}

- P为空，算法结束，得到结果为x_b= (a,b,e,d,c), $f(x_b)$=34.

局部最优问题

- 现实问题中，f在D上往往有多个局部的极值点。
- 一般的局部搜索算法一旦陷入局部极值点，算法就在该点处结束，这时得到的可能是一个糟糕的结果。
- 解决的方法就是每次并不一定选择邻域内最优的点，而是依据一定的概率，从邻域内选择一个点。
- 指标函数优的点，被选中的概率大，指标函数差的点，被选中的概率小。
- 考虑归一化问题，使得邻域内所有点被选中的概率和为1。

局部最优问题

- 当前点的一个邻居被选中的概率可以由邻域中所有邻居的指标函数值计算得到。
- 当求解的最优值为极大值时，$x_i \in N(x_i)$被选中的概率可以定义为：

$$P_{max}(x)=f(x_i) / \sum f(x_j) \qquad (7\text{-}3)$$

- 当求解的最优值为极小值时，$x_i \in N(x_i)$被选中的概率可以定义为：

$$P_{min}(x_i)=(1- P_{max}(x_i)) / (|N(x)|-1) \qquad (7\text{-}4)$$

选择概率计算-数值说明

- 下面用一个数值例子，说明选择概率计算。
- 设当前点x的邻居有5个： x_1, x_2, x_3, x_4, x_5, 其对应的目标函数值分别为： $f(x_1)=10, f(x_2)=12, f(x_3)=14, f(x_4)=16, f(x_5)=12$
- 记$F= f(x_1)+ f(x_2)+ f(x_3)+ f(x_4)+ f(x_5)= 10+12+14+16+10=62$
- 邻居各自的选择概率为：

- $P_{MAX}(x_1) =10/F=5/31$
- $P_{MAX}(x_2) =12/F=6/31$
- $P_{MAX}(x_3) =14/F=7/31$
- $P_{MAX}(x_4) =16/F=8/31$
- $P_{MAX}(x_5) =10/F=5/31$

显然这5个邻居的选择概率之和为1，且x_4被选择的概率最大， x_1和x_5被选择的概率最小 。

局部搜索算法1——克服局部最优

（1）随机选择一个初始的可能解x_0属于D，$x_b=x_0$,$P=N(x_b)$;

//D是问题的定义域，xb用于记录到目标位置的最优解，P为xb的邻域。

（2）如果不满足结束条件，则：//结束条件为循环次数或P为空等

（3）Begin

（4）对于所有的x属于P，计算指标函数f(x)，并按式（7-3）或（7-4）计算每一个点x的概率

（5）依计算的概率值，从P中随机选择一个点x_n，$x_b=x_n$，$P=N(x_b)$，转（2）继续

// 重新计算P，f(x)为指标函数

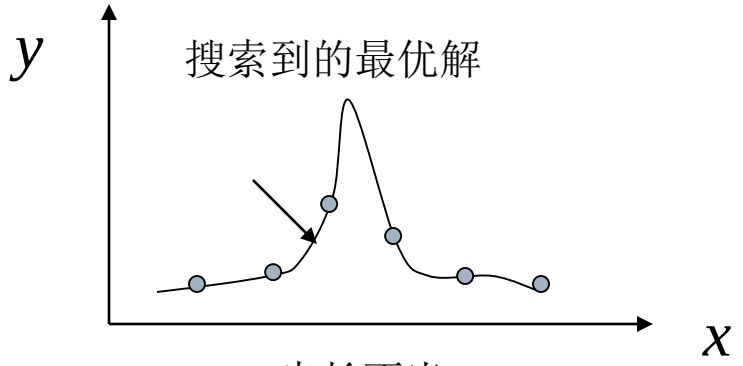
（6）End

（7）输出计算结果

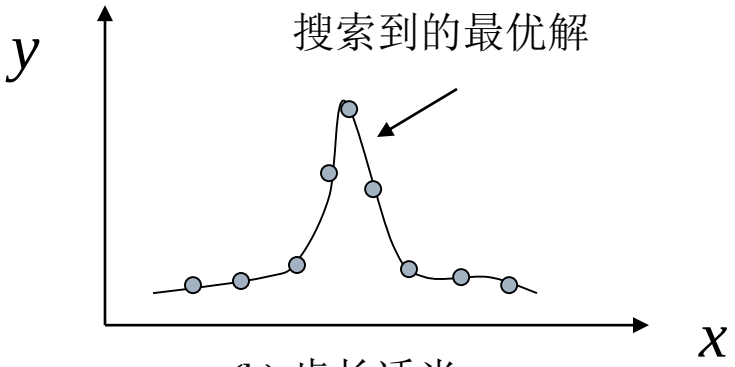
（8）结束

步长问题

- 在距离空间中，邻域可以简单定义为距离当前点固定距离的点。固定距离称为步长。
- 如果步长选择得不合适，即使是单极值的指标函数，一般的局部搜索算法也可能找不到一个可以接受的解。
- 步长太小会使得搜索耗费太多的时间。也不知道步长小到什么程度合适。
- 解决方法是将固定步长的搜索方法改进为动态步长，开始时选择比较大的步长，随着搜索的进行，逐步减小步长。



(a) 步长不当



(b) 步长适当

局部搜索算法2——可变步长

（1）随机选择一个初始的可能解x_0属于D，$x_b=x_0$,$P=N(x_b)$;
//D是问题的定义域，x_b用于记录到目标位置的最优解，P为x_b的邻域。

（2）如果不满足结束条件，则： //结束条件为循环次数或P为空等

（3）Begin

（4）选择P的一个子集P‘，x_n为P’的最优解

（5）如果$f(x_n)<f(x_b)$，则$x_b=x_n$

（6）按某种策略改变步长，计算$P=N(x_b)$，转（2） 继续

（7）否则P=P-P‘，转（2）

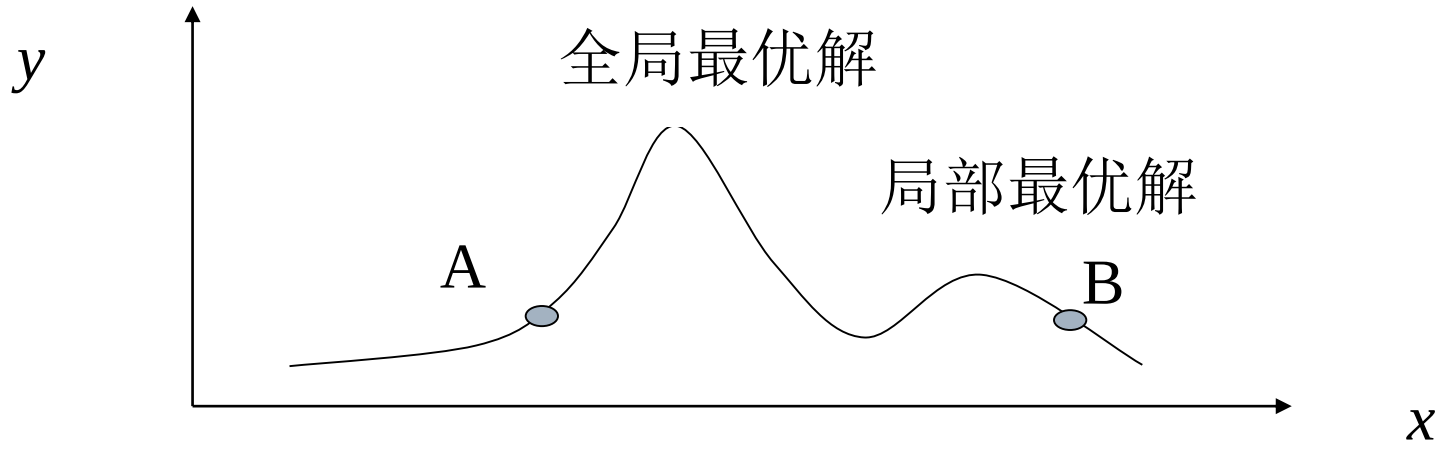
（8）End

（9）输出计算结果

（10）结束

起始点问题

- 一般的局部搜索算法是否能找到全局最优解，与初始点的位置有很大的依赖关系。
- 解决的方法就是随机生成一些初始点，从每个初始点出发进行搜索，找到各自的最优解。再从这些最优解中选择一个最好的结果作为最终的结果。
- 起始点位置影响搜索结果示意图



初始点的位置影响搜索结果

局部搜索算法3——多次起始点

（1）k=0

（2）随机选择一个初始的可能解x_0属于D，$x_b=x_0$,$P=N(x_b)$;

（3）如果不满足结束条件，则：

（4）Begin

（5）选择P的一个子集P‘，x_n为P’的最优解

（6）如果$f(x_n)<f(x_b)$，则$x_b=x_n$，$P=N(x_b)$，转（3）

（7）否则P=P-P‘，转（3）

（8）End

（9）k=k+1

（10）如果k达到了指定的次数，则从k个结果中选择一个最好的结果，否则转（2）

（11）输出结果

（12）结束

局部搜索算法求解皇后问题

J. Gu博士，美籍华人，现在香港理工大学任教。二十世纪80年中，研究局部搜索算法求解NP问题方面成就巨大，他首先提出了求解SAT问题的局部搜索算法，并相继提出了一系列的改进算法，可求解上百变元的3-SAT问题。皇后问题也是一个NP问题，回溯算法难于求解100皇后问题，然而他局部搜索算法已可求解百万级的皇后问题。

皇后搜索算法

（1）随机地将N 个皇后分布在棋盘上，使得棋盘的每行、每列只有一个皇后。

（2）计算皇后间的冲突数Conflicts

（3）如果冲突数等于0等转（6）

（4）对于棋盘上的任意两个皇后，交换他们的位置，如果交换后的冲突数减少，则接受这种交换，更新Conflicts

（5）如果陷入了局部极小，即交换了所有的皇后后，冲突数仍不能下降，则转（1）

（6）输出结果

（7）结束

皇后数	100	500	1000	2000	5000	10000	30000
平均时间	5	5	12	28	170	900	10000

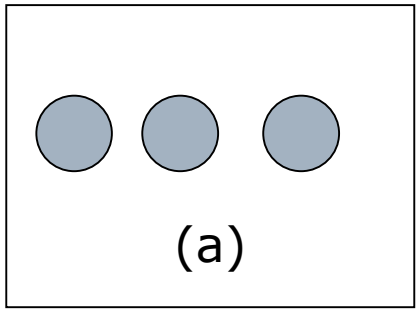
模拟退火算法（simulated annealing）

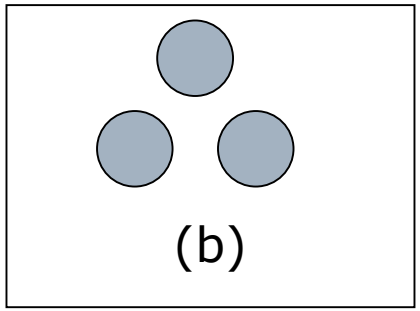
- 模拟退火算法是局部搜索算法的一种扩展；
- 算法思想最早由Metropolis在1953年提出；
- 1983年Kirkpatrick等人成功将模拟退火算法用于求解组合优化问题；
- 模拟退火算法来源于固体退火原理。

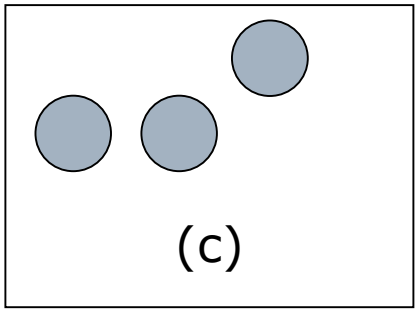
固体退火过程

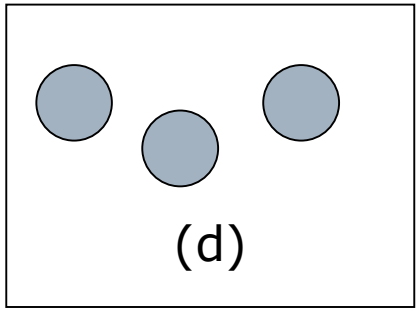
退火过程是一种物理现象。当对金属块进行加热时，粒子的热运动不断增加，随温度的不断升高，粒子逐渐脱离开其平衡位置，变得越来越自由，当到达金属的熔点时，粒子排列从原来的有序状态(有型固态)变为完全的无序状态(无型液态)。这是金属熔解过程。退火过程与熔解过程刚好相反。随着温度的下降，粒子的热运动逐渐减弱，粒子逐渐停留在不同的位置，其排列也从无序向有序方向发展，直到温度很低时，粒子重新以一定的结构排列。粒子不同的排列结构对应着不同的能量水平，如果退火过程是缓慢进行的，也就是说温度的下降是非常缓慢的话，使得在每个温度下，粒子的排列都达到一种平衡状态，则当温度趋于0时，系统(金属铸件)的能量将趋于最小值(无泡、无裂、表面光滑、强度最大等)。

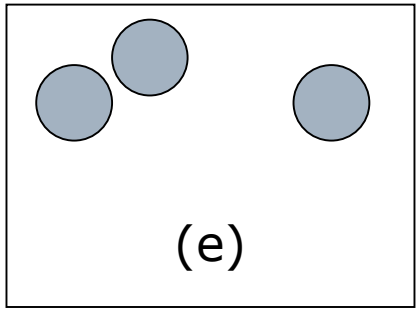
三个粒子的各种结构例子

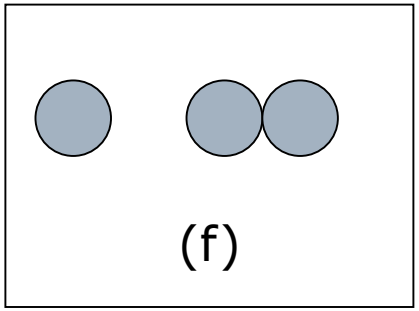












.....

退火过程建模理论-1

- 用粒子的排列或相应的能量表示物体所处的状态，在温度T下，物体(系统)所处的状态具有一定的随机性。主流趋势是系统向能量较低的状态发展，但粒子的不规则热运动妨碍系统准确落入低能状态。
- Metropolis的从状态i转换到状态j的准则：
  如果$E(j) \le E(i)$，则状态转换被接受；
  如果$E(j) > E(i)$，则状态转换被接受的概率为

$$e^{\frac{E(i)-E(j)}{KT}}$$

其中$E(i)$、$E(j)$分别表示状态i、j下的能量，T是温度，$K > 0$是波尔兹曼常数；

Metropolis准则表达了退火过程的能量变化。

退火过程建模理论-2

- 系统处于某个状态i的概率服从Boltzmann分布：

$$P_i(T) = \frac{e^{-\frac{E(i)}{KT}}}{Z_T} \text{其中，} \qquad Z_T = \sum_{j \in S} e^{-\frac{E(i)}{KT}}$$

Z_T为归一化因子，S表示所有可能状态集。

- 系统向能量低发展的概率大于向能量高发展的概率。
在给定温度T下，设有系统有i、j两个状态，
且$E(i) < E(j)$, 根据Boltzmann分布有：

退火过程建模理论-3

$$P_i(T) - P_j(T) = \frac{e^{-\frac{E(i)}{KT}}}{Z_T} - \frac{e^{-\frac{E(j)}{KT}}}{Z_T}$$

$$= \frac{1}{Z_T} e^{-\frac{E(i)}{KT}} \left(1 - \frac{e^{-\frac{E(j)}{KT}}}{e^{-\frac{E(i)}{KT}}}\right)$$

$$= \frac{1}{Z_T} e^{-\frac{E(i)}{KT}} \left(1 - e^{-\frac{E(j)}{KT} + \frac{E(i)}{KT}}\right)$$

$$= \frac{1}{Z_T} e^{-\frac{E(i)}{KT}} \left(1 - e^{-\frac{E(j) - E(i)}{KT}}\right)$$

退火过程建模理论-4

由于$E(i)<E(j)$，所以有 $e^{-\frac{E(j)-E(i)}{KT}}<1$

于是有：$P_i(T)-P_j(T)>0$

即在任何温度下，系统处于能量低的状态的概率大于处于能量高的状态的概率。

- 当温度很高时，系统处于各个状态的概率基本相等，接近于平均值，与所处状态的能量几乎无关。因为

$$\lim_{T\to\infty}(P_i(T))=\lim_{T\to\infty}\left[\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)}{KT}}}\right]=\frac{1}{|S|}$$

其中|S|为系统所有可能状态数。

退火过程建模理论-5

- 当温度很低时，系统以等概率趋近几个能量最小的状态，而系统处于其它状态的概率几乎为0。因为

$$\lim_{T\to\infty}(P_i(T)) = \lim_{T\to\infty}\left[\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S} e^{-\frac{E(j)}{KT}}}\right]$$

$$= \lim_{T\to\infty}\left[\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S} e^{-\frac{E(j)}{KT}}}\cdot\frac{e^{-\frac{E_m}{KT}}}{e^{-\frac{E_m}{KT}}}\right]$$

退火过程建模理论-5

$$= \lim_{T \to \infty} \left[\frac{e^{-\frac{E(i)-E_m}{KT}}}{\sum_{j \in S} e^{-\frac{E(j)-E_m}{KT}}} \right]$$

$$= \lim_{T \to \infty} \left[\frac{e^{-\frac{E(i)-E_m}{KT}}}{\sum_{j \in S_m} e^{-\frac{E(j)-E_m}{KT}} + \sum_{j \notin S_m} e^{-\frac{E(j)-E_m}{KT}}} \right]$$

退火过程建模理论-6

若$E(j)-E_m>0$,则有

$$-\frac{E(j)-E_m}{KT}<0,\text{且}\lim_{T\to 0}\left(-\frac{E(j)-E_m}{KT}\right)\to-\infty$$

于是$$\lim_{T\to 0}\left(e^{-\frac{E(j)-E_m}{KT}}\right)\to e^{-\infty}=\frac{1}{e^{+\infty}}\to 0$$

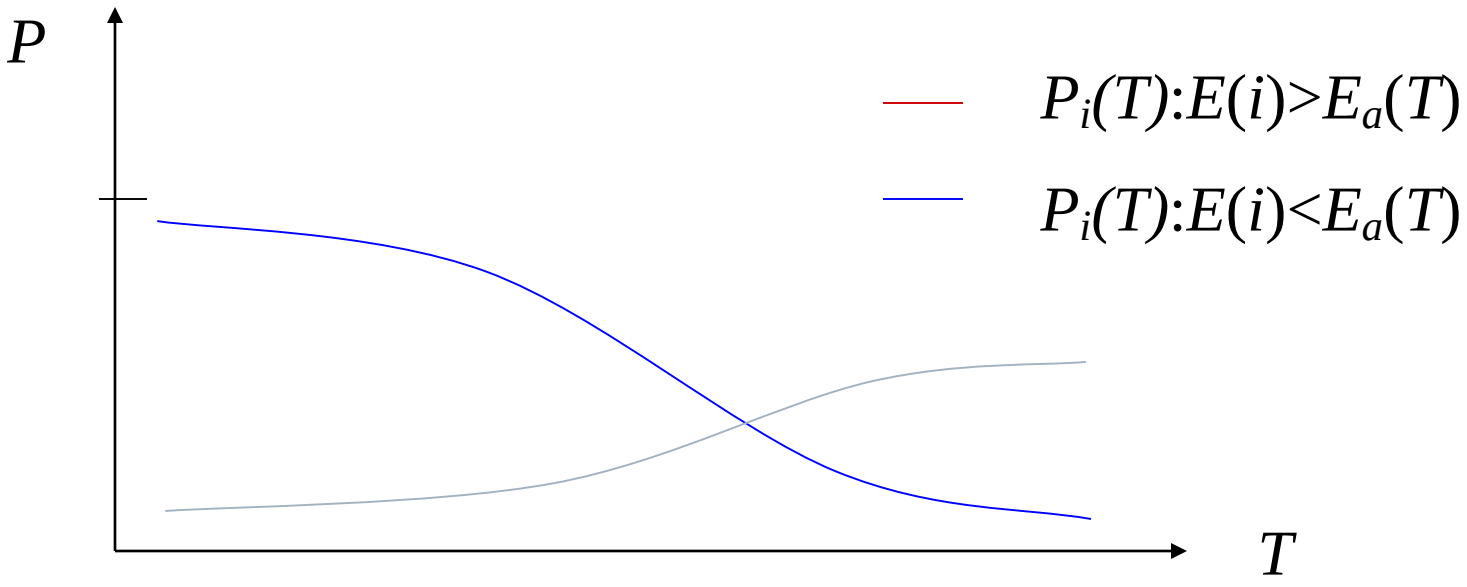
退火过程建模理论-7

$$\text{所以} \lim_{T\to 0}\left[\frac{e^{-\frac{E(i)-E_m}{KT}}}{\sum_{j\in S_m} e^{-\frac{E(j)-E_m}{KT}} + \sum_{j\notin S_m} e^{-\frac{E(j)-E_m}{KT}}}\right]$$

$$= \lim_{T\to 0}\left[\frac{e^{-\frac{E(i)-E_m}{KT}}}{\sum_{j\in S_m} e^{-\frac{E(j)-E_m}{KT}}}\right] = \begin{cases} \frac{1}{|S_m|} & \text{如果} i\in S_m \\ 0 & \text{如果} i\notin S_m \end{cases}$$

退火过程建模理论-8

- 系统落入能量较低的状态的概率是随温度T单调下降的，而系统落入能量较高的状态的概率是随温度T单调上升的。换句话说，系统落入能量较低的状态的概率是随温度T下降而单调上升，而系统落入能量较高的状态的概率是随温度T下降而单调下降的。



退火过程建模理论-9

退火过程建模理论的结论：

- 在温度很高时，系统基本处于无序，以等概率落入各个状态。
- 在给定温度下，系统向能量低发展的概率大于向能量高发展的概率。这样在同一温度下，如果系统交换得足够充分，则系统会趋向于落入能量较低的状态。
- 随温度T下降，系统落入能量较低的状态的概率单调上升，而系统落入能量较高的状态的概率是单调下降的。而只有那些能量小于期望值的状态，落入的概率才随温度下降而增加，其它状态的落入概率随温度下降而减小。随能量期望值的逐步下降，能量小于期望值的状态数目逐步减少，当温度趋于0时，系统只能处于能量最小的状态，处于其它状态的概率趋于0。因此最终系统将以概率1处于能量最小的一个状态。

退火过程建模理论-10

退火过程中，系统将以概率1处于能量最小的一个状态的条件有3个：

1. 初始温度足够高；
2. 在每一个温度下，状态的交换必须足够充分；
3. 温度T的下降必须足够缓慢。

这三个条件是模拟退火过程的难点和关键参数。

组合优化问题与固体退火过程

固体退火过程	组合优化问题
物理系统的一个状态	组合优化问题的解
状态的能量	解的指标函数
能量最低状态	最优解
温度	控制参数

模拟退火算法的基本思想

由初始解i和控制参数初值t开始，对当前解重复“产生新解→计算目标函数差→接受或舍弃”的迭代，并逐步衰减t值，算法终止时的当前解即为所得近似最优解，这是基于蒙特卡罗迭代求解法的一种启发式随机搜索过程。

退火过程由冷却进度表(Cooling Schedule)控制，包括控制参数的初值t及其衰减因子Δt、每个t值时的迭代次数L和停止条件S。

模拟退火算法描述

(1) 随机选择一个解i，$k=0$，$t_0=T_{max}$(初始温度充分大)，计算$f(i)$;

(2) 如果满足结束条件，则转(13)

(3) **Begin**

(4) 如果在该温度内达到了平衡条件，则转(11)

(5) **Begin**

(6) 从i的邻域$N(i)$中随机选择一个解j，计算指标函数$f(j)$

(7) 如果$f(j)<f(i)$，则$i=j$，$f(i)=f(j)$，转(4)

(8) 计算 $P_t(i \to j) = e^{-\frac{f(j)-f(i)}{t}}$

(9) 如果$P_t(i \to j)>Random(0,1)$,则$i=j$，$f(i)=f(j)$，转(4)

(10) **End**

(11) $t_{k+1}=Drop(t_k),k=k+1$

(12) **End**

(13) 输出结果

(14) 结束

模拟退火算法说明

- 内循环模拟在给定温度下系统达到热平衡的过程;
- 每次内循环随机产生一个新解，然后按Metropolis准则随机地接受该解;
- Random(0,1)是一个[0,1]上均匀分布的随机数发生器;
- 外循环模拟温度的下降过程,控制接受劣解的概率发展趋势;
- Drop(t_k)是一个温度下降函数，温度缓慢下降;

模拟退火算法与局部搜索算法

- 模拟退火算法是局部搜索算法的扩展
- 模拟退火算法按Metropolis准则随机地接受一些劣解；
- 当温度比较高时，接受劣解的概率大；
- 在初始高温下，几乎以100%的概率接受劣解；
- 随着温度下降，接受劣解的概率逐渐减小；
- 当温度趋于0时，接受劣解的概率趋于0；
- 有利于从局部最优解中跳出，求得问题的全局最优解。

模拟退火算法分析

- 模拟退火算法具有渐近收敛性，只要合适地构造产生概率$G_t(i,j)$和接受概率$A_t(i,j)$和邻域，就可以保证以概率1收敛于全局最优解；
- 模拟退火算法以概率1找到全局最优解的基本条件，是初始温度必须足够高，状态交换必须足够充分，温度下降必须足够缓慢。
- 模拟退火算法逐渐达到最优解的能力是以搜索过程的无限次状态转移为前提的，算法复杂性仍然是指数时间的。
- 模拟退火算法解的质量与算法的运行时间成正比。对于很多实际问题，求最优解意义不大，因此只要通过确定一些参数或者准则，可以在一个多项式时间内求一个满意解就可以了。

初始温度的选取

- 模拟退火算法要求初始温度足够高；
- 初始温度应根据具体的问题而定；
- 一个合适的初始温度应保证接受概率P_0近似等于1，即：

$$e^{-\frac{f(j)-f(i)}{t_0}} = e^{-\frac{\Delta f}{t_0}} \approx 1$$

这样给定P_0，可以计算t_0。

- 可以仿照固体的升温过程，得到一个合适的初始温度。

温度的下降方法

模拟退火算法要求温度缓慢下降，下降方法有：

- 等比例下降

$$t_{k+1} = \alpha \cdot t_k, k = 0,1,..., \quad 0 < \alpha < 1$$

- 等值下降

$$t_{k+1} = t_k - \Delta t, \quad \Delta t = \frac{t_0}{K}$$

- 基于距离参数的下降

$$t_{k+1} = \frac{t_k}{1 + \frac{t_k \ln(1+\delta)}{3\sigma_{t_k}}}, k = 0,1,...$$

每一温度下的停止准则

- 模拟退火算法要求产生足够的状态交换；
- 如果用L_k表示在温度t_k下的迭代次数，则L_k应使得系统在这一温度下基本达到平稳状态；
- 采用固定长度的停止准则

在每一个温度下，都使用相同的L_k； L_k 的选择与问题相关，如n个城市的TSP问题，如果采用交换两个城市的方法产生邻域，邻域大小为n(n-1)/2, L_k 可以选取cn或cn^2，其中c为常数；

- 采用基于接受概率的停止准则

✓ 规定接受次数R；

✓ 规定接受率R；

✓ 规定相邻两代的解的指标函数的差

算法的终止原则

合理的结束条件，应使得算法收敛于问题的某一个近似解，同时应能保证解具有一定的质量，并且应在一个可以接受的有限时间内停止求解。常用的终止方法有：

- 零度法
- 循环总控制法
- 无变化控制法
- 接受概率控制法
- 邻域平均概率控制法
- 相对误差估计法

用模拟退火算法求解TSP问题

TSP (Travelling Salesman Problem)：设有n个城市，用数码1,...,n代表。城市i和城市j之间的距离为d(i，j) i, j= 1,...,n．TSP问题是要找遍访每个域市恰好一次的一条回路，且其路径总长度为最短。

算法模型如下：

（1）解空间：解空间S是遍访每个城市恰好一次的所有回路，是n个城市的所有排列的集合，当规定了出发城市时，S的规模为(n-1)!。

（2）目标函数：此时的目标函数即为访问所有城市的路径总长度或称为代价函数，我们要求此代价函数的最小值。

（3）新解的产生：用两个城市间的逆序交换方式得到问题的一个新解。

（4）指标函数差：只涉及两对城市间的变化

（5）新解的接受准则

（6）参数确定：初始温度、迭代次数、衰减系数、停止准则