第三章　禁忌搜索

一. 前言

1. 组合优化问题

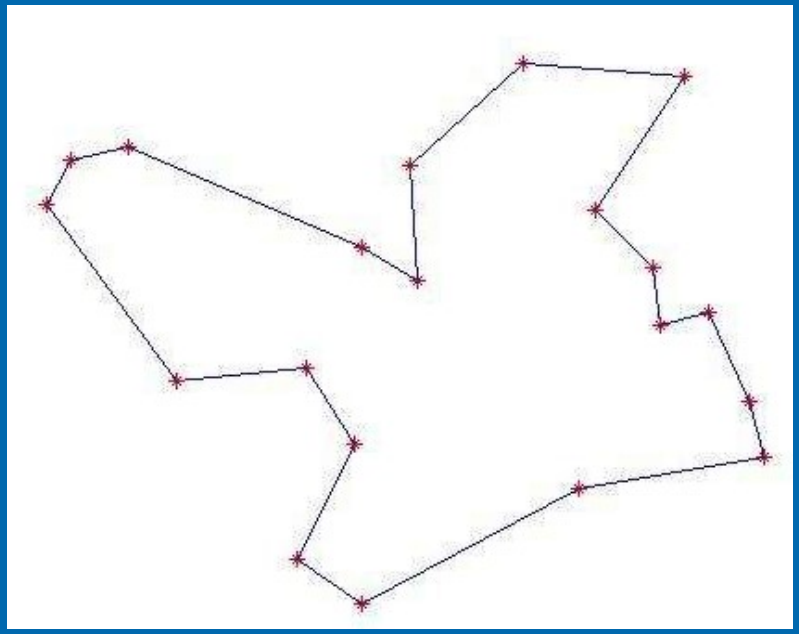
- 定义
  - ✓ “最优化问题似乎自然地分成两类：一类是连续变量的问题，另一类是离散变量的问题。具有离散变量的问题，我们称它为组合的。”——来自百度百科
  - ✓ **“A combinatorial optimization problem can be modeled as that of minimizing a cost function or maximizing a profit function over a finite set of discrete variables.”**

一. 前言

1. 组合优化问题

- 典型的组合优化问题
  - ✓ 旅行商问题
  - ✓ 生产调度问题
  - ✓ 背包问题
  - ✓ 装箱问题
  - ✓ 聚类问题

旅行商问题



旅行商问题（**Travel Salesman Problem, TSP**）可以描述为：假设有一个商人要拜访若干个城市，他必须选择一条所要走的路径；路径的限制是每个城市只能拜访一次，而且最后要回到原来出发的城市；路径选择的目标是所选择的路径长度最短。**TSP**是运筹学领域最著名问题之一。

一. 前言

2. 两个重要的概念

- 解空间

对于一个组合优化问题来说，解空间定义为：任一可能（可行）解可以访问的搜索空间

举例：对于n个城市的旅行商问题来说，一个解可以用自然数序列{1,2,…,n}的某种排列来表达，于是解空间则是由该自然数序列的所有排列组合构成。

{1,2,3,4}、{2,3,1,4}都是上述4城市TSP的解

一. 前言

2. 两个重要的概念

- 邻域

令S表示组合优化问题的一个解，则其邻域N(S)可以定义为：S执行一次邻域移动可以访问的所有解的集合

举例：对于4个城市的旅行商问题来说，采用上述的基于排列的解表达方法，邻域移动定义为 *2-opt*，即对S中2个元素进行对换

S=(1,2,3,4)，则领域N(S)={(2,1,3,4), (3,2,1,4), (4,2,3,1), (1,3,2,4), (1,4,3,2), (1,2,4,3)}

一. 前言

2. 两个重要的概念

- 邻域

2-opt可以扩展为***k-opt***，即对**k**个元素按一定规则互换，这种映射方式也被成为基于置换的邻域移动；除此以外，还有基于插入的邻域移动和基于翻转的邻域移动。

例如，**S=(1,2,3,4)**
若执行一次基于插入的邻域移动，则可获得**S'=(3,1,2,4)**
若执行一次基于翻转的邻域移动，则可获得**S'=(4,3,2,1)**

练　习

定义邻域移动为：***2-opt***

若当前解为**[4 2 3 5 1]**，下列解是否在其邻域内?

[4 3 2 5 1]　[4 3 5 1 2]　[4 3 3 5 1]

[5 2 3 4 1]　[1 2 3 5 4]　[3 4 2 5 1]

一. 前言

2. 局域搜索

- 基本思想：从一个初始解开始，每次迭代总是遍历其邻域，若不存在比当前最优解更好的邻域解，则返回当前最好解，算法停止；否则用最好的邻域解替换当前解。如此循环直至算法停止。类似于模拟爬山的过程，因此也被称之为爬山算法。

一. 前言

2. 局域搜索

- 算法流程

%%参数定义

S：当前解

S^*：当前最优解

f^*：当前最优函数值

$N(S)$：S的邻域

一. 前言

%%初始化

产生（或构造）一个初始解S_0；

令$S \leftarrow S_0$，$f^* \leftarrow f(S_0)$，$S^* \leftarrow S_0$；

%%迭代过程

Repeat

令$S \leftarrow \text{argmin}_{S' \in N(S)}[f(S')]$；

若$f(S) < f^*$，则令$f^* \leftarrow f(S)$，$S^* \leftarrow S$；

否则，算法停止；

End.

一. 前言

2.局域搜索

- 示例

5个城市的对称旅行商问题，选定城市1为起点

$$D=(d_{ij})=\begin{bmatrix}0 & 10 & 15 & 6 & 2\\ 10 & 0 & 8 & 13 & 9\\ 15 & 8 & 0 & 20 & 15\\ 6 & 13 & 20 & 0 & 5\\ 2 & 9 & 15 & 5 & 0\end{bmatrix}$$

初始解为S=(1,2,3,4,5)，f(S)=45，定义邻域移动为*2-opt*
S*=(1,2,3,4,5)，f*(S)=45

一. 前言

2.局域搜索

- 示例

第1步

N(S)={(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)}

对应目标函数为{43, 45, 60, 60, 59, 44}

S*=(1,3,2,4,5)，f*(S)=43，S=(1,3,2,4,5)

一. 前言

2. 局域搜索

- 示例

第2步

N(S)={(1,2,3,4,5), (1,4,2,3,5), (1,5,2,4,3), (1,3,4,2,5), (1,3,5,4,2), (1,3,2,5,4)}

对应目标函数为**{45, 44, 59, 59, 58, 43}**

S*=(1,3,2,4,5)，*f(S)=43**

若不设定出发城市，则算法结果是否会不一样？

一. 前言

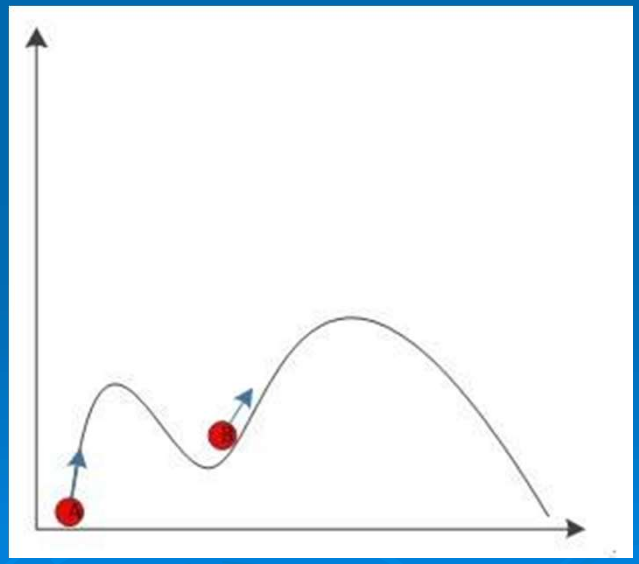
2. 局域搜索

- 优劣性
  - ① 通用易实现，易于理解
  - ② 搜索效果依赖于初始点和邻域结构，极易陷入局优

一. 前言

2. 局域搜索

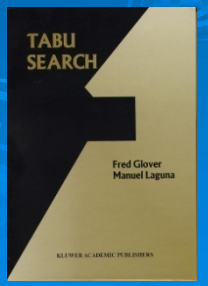
- 优劣性
  - ① 通用易实现，易于理解
  - ② 搜索效果依赖于初始点和邻域结构，极易陷入局优



二. 基本算法

1. 算法思想

- 人类在选择过程中具有记忆功能，比如走迷宫时，当发现有可能又回到某个地点的时候总会有意识地避开先前选择的方向而选择其他的可能性，这样就可以确定性的避开迂回搜索
- 通过接受劣解来逃离局优
- **Tabu Search，简称TS**



二. 基本算法

2. 构成要素

- 解的表达
  - ① 采用恰当的数据结构来表示问题的解
  - ② 初始解的产生：随机产生或者采用启发式方法产生一个初始解
  - ③ 适值函数的构造：往往直接将目标函数作为适值函数

二. 基本算法

2. 构成要素

- 邻域移动

① 组合优化问题中邻域移动可以定义为某种排列置换
② 邻域移动定义是灵活的，目的是便于搜索

局部搜索：当前解附近区域搜索

全局搜索：远离当前解区域搜索

二. 基本算法

2. 构成要素

- 禁忌表(T表)：防止搜索出现循环
  - ① 将移动作为禁忌对象
  - ② 表的长度称为**Tabu-Size**，可以用来控制局域搜索和广域搜索
  - ③ 表是动态更新的：把最新的解记入，最老的解从表中释放（解禁）

二. 基本算法

2. 构成要素

- 选择策略
  - ① 当前解S每一步总是移动到邻域N(S)中未被禁忌的最优解
  - ② 为了保证算法具有跳出局优的能力

二. 基本算法

2. 构成要素

- 渴望水平
  - ① 若被禁忌的邻域移动会使当前解达到超出渴望水平的要求，则这种邻域移动将不受T表限制
  - ② 渴望水平一般选取为历史上所能达到的最优函数值，其主要目的是为了破禁

选择策略和渴望水平构成了禁忌搜索的两大核心规则

二. 基本算法

2. 构成要素

- 停止准则
  - ① 设定最大迭代次数
  - ② 得到满意解
  - ③ 若干次迭代未获得更好的解

二. 基本算法

3. 算法流程

%%参数定义

S：当前解

S^*：当前最优解

f^*：当前最优函数值

$N(S)$：S的邻域

$\tilde{N}(S)$：$N(S)$的可行子集（未被禁忌以及突破渴望水平的邻域解）

T：禁忌表

二. 基本算法

%%初始化

产生（或构造）一个初始解S_0;

令$S \leftarrow S_0$，$f^* \leftarrow f(S_0)$，$S^* \leftarrow S_0$，$T \leftarrow \emptyset$;

%%迭代过程

While 停止条件不被满足 **do**

令$S \leftarrow \text{argmin}_{S' \in \tilde{N}(S)}[f(S')]$;

若$f(S) < f^*$，则令$f^* \leftarrow f(S)$，$S^* \leftarrow S$;

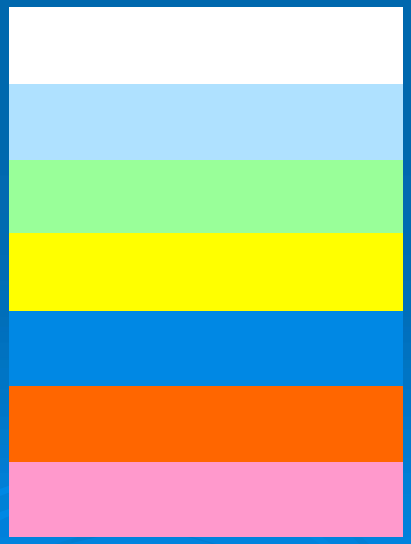
更新禁忌表T;

Endwhile.

三. 算法举例

1. 问题提出

由7层不同的绝缘材料构成的一种绝缘体，应如何排列顺序，可获得最好的绝缘性能?



三. 算法举例

2. 算法设计

- 解的表达：顺序编码
- 初始解的产生：随机产生，如**2-5-7-3-4-6-1**
- 适值函数：极大化目标值
- 邻域移动：**2-opt**
- 其他参数：禁忌对象为邻域移动方式，**T**表长度设为**3**，最大迭代次数设为**5**

三. 算法举例

① 初始表

初始编码：2-5-7-3-4-6-1

$f(x)=10$，$x^*=x$，$A=f(x^*)=16$，$T=\emptyset$

移动	$\Delta f(x)$
5, 4	**6**
7, 4	4
3, 6	2
2, 3	0
4, 1	-1
……	……

T表	
1	∅
2	∅
3	∅

结论：交换4和5，即解变为：2-4-7-3-5-6-1

三. 算法举例

② 迭代1

编码：2-4-7-3-5-6-1

$f(x) = 16$，$x^* = x$，$A = f(x^*) = 16$

移动	$\Delta f(x)$
3, 1	**2**
2, 3	1
3, 4	-1
7, 1	-2
6, 1	-4
……	……

T表	
1	4, 5
2	∅
3	∅

结论：交换1和3，解变为：2-4-7-1-5-6-3

三. 算法举例

③ 迭代2

编码：2-4-7-1-5-6-3

$f(x) = 18$，$x^* = x$，$A = f(x^*) = 18$

移动	$\Delta f(x)$
1, 3	-2
2, 4	-4
7, 6	-6
4, 5	-7
5, 3	-9
……	……

被禁忌

T表	
1	1, 3
2	4, 5
3	Ø

结论：因交换1和3已在禁忌表中，故只能交换2和4

三. 算法举例

④ 迭代3

编码：4-2-7-1-5-6-3

$f(x) = 14$，$A = f(x^*) = 18$

移动	$\Delta f(x)$
4, 5	6
5, 3	2
7, 1	0
1, 3	-3
2, 6	-6
……	……

因$f(x)=20>A=18$，此时渴望水平发生作用，破禁。交换4和5。

T表	
1	2, 4
2	1, 3
3	4, 5

结论：因渴望水平发挥作用，交换4和5

三. 算法举例

⑤ 迭代4

编码：5-2-7-1-4-6-3

$f(x) = 20$，$x^* = x$，$A = f(x^*) = 20$

移动	$\Delta f(x)$
7, 1	0
4, 3	-3
6, 3	-5
5, 4	-6
2, 6	-8
……	……

T表	
1	4, 5
2	2, 4
3	1, 3

结论：交换7和1

三. 算法举例

⑥ 迭代5

编码：5-2-1-7-4-6-3

$f(x) = 20$

T表	
1	1, 7
2	4, 5
3	2, 4

结论：迭代已到5次，得到最优解

$$x^* = \{5, 2, 7, 1, 4, 6, 3\}$$

$$f(x^*) = 20$$

四. 标准算法

1. 基于概率的邻域搜索

- 利用邻域的一个随机子集替代完整邻域
- 伪代码如下：

While 停止条件不被满足 **do**

令$S \leftarrow \mathbf{argmin}_{S' \in \tilde{N}(S)}[f(S')]$;

若$f(S) < f^*$，则令$f^* \leftarrow f(S)$，$S^* \leftarrow S$;

更新禁忌表T;

Endwhile.

用$\tilde{N}'(S)$代替$\tilde{N}(S)$， $\tilde{N}'(S)$表示$\tilde{N}(S)$的一个随机子集

四. 标准算法

2. 频数表

- 频数表是用来记忆不同方向的移动次数，从而加以惩罚（比如2-opt，记录每对交换的发生次数），从而提高搜索方向的多样性
- 令$\boldsymbol{S \leftarrow argmin}_{S' \in \tilde{N}(S)}[f(S') + \alpha E(S, S')]$，其中$E(S, S')$表示由$S$移动到$S'$的出现频数，$\alpha$为惩罚因子

注：惩罚因子α的取值一般应远小于目标值（1%目标值或1‰目标值），α越大分散性越好，广域搜索能力强，但会损坏邻域搜索。

课堂作业

5个城市旅行商问题，城市间距离矩阵如下：

$$D=(d_{ij})=\begin{bmatrix}0 & 10 & 15 & 6 & 2\\ 10 & 0 & 8 & 13 & 9\\ 15 & 8 & 0 & 20 & 15\\ 6 & 13 & 20 & 0 & 5\\ 2 & 9 & 15 & 5 & 0\end{bmatrix}$$

初始解为(1,2,3,4,5)；

领域移动方式为2-opt；

若T表长度设为3，最大迭代次数设为5，请给出完整的禁忌搜索迭代过程。

五. 算法应用实例

1. 基础数据

- 全国31个省会城市的坐标数据如下：

```
C = [1304 2312; 3639 1315; 4177 2244; 3712 1399; 3488 1535; 3326 1556;
     3238 1229; 4196 1044; 4312  790; 4386  570; 3007 1970; 2562 1756;
     2788 1491; 2381 1676; 1332  695; 3715 1678; 3918 2179; 4061 2370;
     3780 2212; 3676 2578; 4029 2838; 4263 2931; 3429 1908; 3507 2376;
     3394 2643; 3439 3201; 2935 3240; 3140 3550; 2545 2357; 2778 2826;
     2370 2975]
```

五. 算法应用实例

2. TS算法设置

- 顺序编码，编码长度**N=31**
- 初始解随机产生
- 邻域移动方式为**2-OPT**，邻域大小**Ca=200**
- 紧急对象为邻域移动方式，禁忌表长**TabuL=22**
- 最大迭代次数**G=1000**

五. 算法应用实例

3. TS算法仿真过程

① 初始化城市数量$N=31$，禁忌长度$TabuL=22$，邻域大小$Ca=200$，最大迭代次数$G=1000$。

② 计算任意两个城市的距离矩阵D；随机产生初始解S_0，计算其适值，并将其赋给当前最优解$bestsofar$。

③ 随机产生Ca个候选邻域解，计算候选邻域解的适值。

五. 算法应用实例

3. TS算法仿真过程

④ 判断当前最好的候选邻域解是否满足破禁准则：若满足，则用其替代当前解成为新的当前解，并更新禁忌表$\boldsymbol{Tabu}$和禁忌长度$\boldsymbol{TabuL}$，然后转步骤⑥；否则，继续以下步骤。

⑤ 选择未被禁忌的最好候选邻域解作为新的当前解，同时更新禁忌表$\boldsymbol{Tabu}$和禁忌长度$\boldsymbol{TabuL}$。

⑥ 判断是否满足终止条件：若满足，则结束搜索过程，输出当前最优解；若不满足，则继续进行迭代优化。

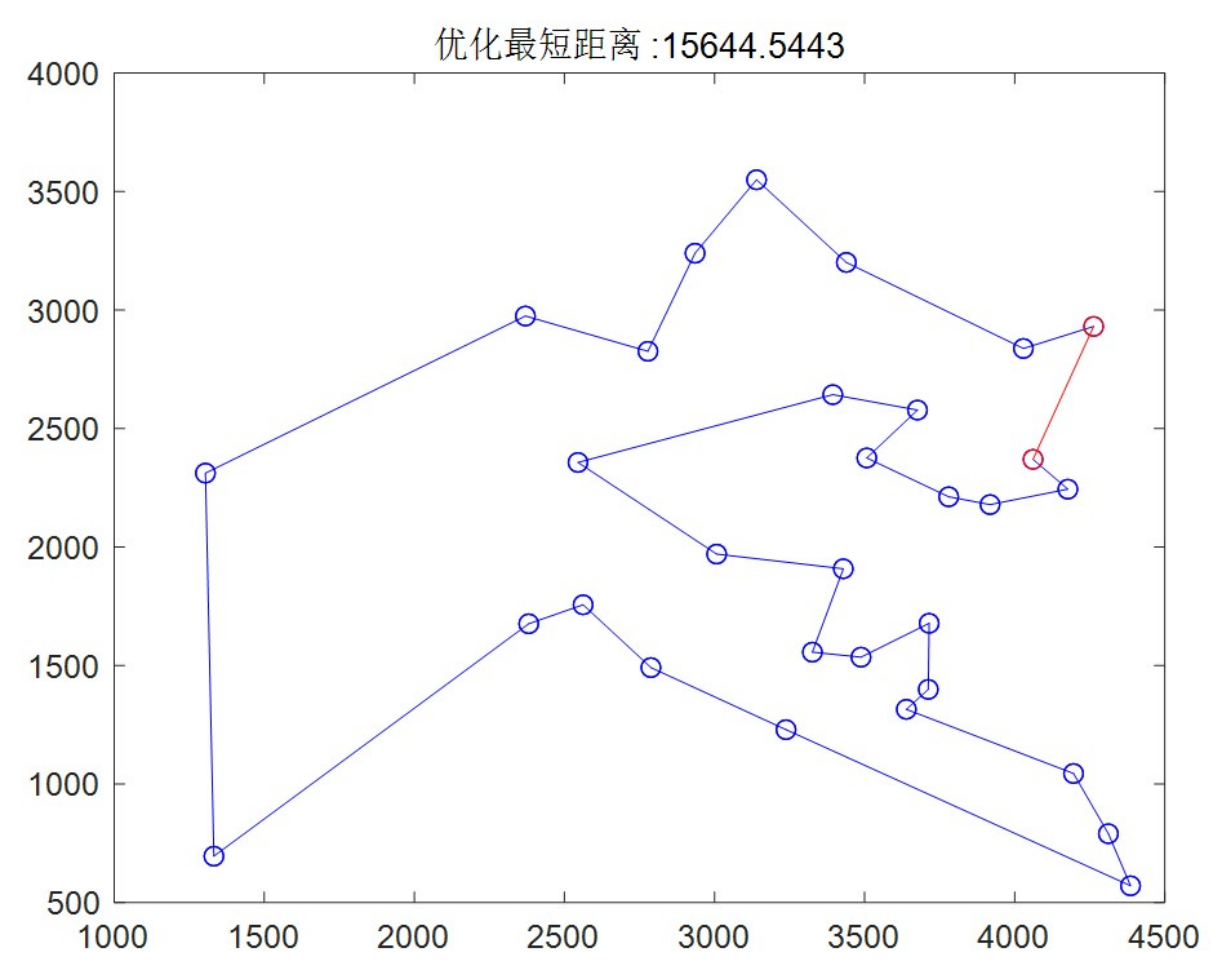
五. 算法应用实例

4. Matlab代码

- 参见附件文档

五. 算法应用实例

5. 仿真实验结果



编程作业

作业要求：

- 根据参考代码，调试程序。
- 分析禁忌表长度对TS算法性能的影响。
- 分析初始解对TS算法性能的影响。

提示：

1. 目前代码中禁忌表长为22，尝试将其分别设置为10、15、25和30时，分析算法性能会发生什么样的变化？
2. 当前代码中初始解随机产生，尝试利用贪婪启发式产生初始解，分析是否有助于改善算法性能？