 **南京航空航天大学**

**离散数学实验**

**实验报告**

|  |  |
| --- | --- |
| 学生姓名 | 马睿 |
| 学 号 | 161930131 |
| 学 院 | 计算机科学与技术学院/人工智能学院 |
| 专 业 | 软件工程 |
| 授课老师 | 郝洁 |

二〇二零年十月

**目录**

**一、问题描述及分析 1**

**二、算法描述1**

1.最邻近法1

2.最小生成树法2

3.分支定界法3

4.模拟退火算法4

5.蚁群算法6

6.遗传算法7

7.禁忌搜索9

**三、测试用例12**

**四、总结与体会19**

1. **问题描述及分析**

**TSP问题**（Traveling Salesman Problem，旅行商问题）：有若干个城市，任何两个城市之间的距离都是确定的，现要求一旅行商从某城市出发必须经过每一个城市且只在一个城市逗留一次，最后回到出发的城市，问如何事先确定一条最短的线路以保证其旅行的费用最少？

从图论的角度来看，该问题实质是在一个带权完全无向图Kn中，找一个权值最小的Hamilton回路。问题中的城市作为图中的顶点，城市之间的道路是图中的边，路径的距离为边的权值。若点与点之间中不存在道路时，将其边权设为∞即可。

由于该问题的可行解是所有顶点的全排列，随着顶点数的增加，会产生组合爆炸。早期的研究者使用精确算法求解该问题，常用的方法包括：分枝定界法、线性规划法、动态规划法等。但是，随着问题规模的增大，精确算法将变得无能为力，因此，在后来的研究中，国内外学者重点使用近似算法或启发式算法，主要有遗传算法、模拟退火法、蚁群算法、禁忌搜索算法、贪婪算法和神经网络等。

1. **算法描述**

解决TSP问题的方法主要分为两种：**传统解决方法**以及**智能优化算法**。其中，传统的优化算法是一种局部搜索算法，一般得到局部最优解，很难达到全局最优解，并且很难适用于大规模的最优化问题；相反，智能优化算法弥补了传统优化算法的缺陷，但是大部分算法需要设定许多参数，参数的大小会很大程度影响算法的时间效率。在本次实验中，传统算法我使用了：最小生成树法、最邻近法、分支定界法；智能优化算法我使用了：遗传算法、模拟退火法、蚁群算法、禁忌搜索算法。因为大部分地方已经在程序中写明了注释，所以这里仅描述一下算法的主要思想和步骤。

设G=<V,E,W>为n阶无向完全带权图，各边所带权均为正数。**所用的数据结构为n\*n阶的二维数组**，第i行j列元素的值代表顶点i和顶点j之间的权值大小。

1. **最邻近法**

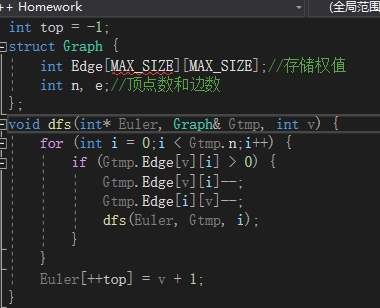
因为该方法较为简单且书上有说明，这里不再赘述。

1. **最小生成树法**

因为书上有该方法的具体说明，这里不再赘述。

在实现该算法时，**求最小生成树**我使用了Prim算法，因为所研究的G大多为稠密图，Prim算法更适合。

**求欧拉回路**那一步骤我没有使用Tarjan算法，而使用了如下算法。



其中Euler数组用来存放所求的欧拉回路，该数组实质上为一个栈，top用来指向栈顶元素，初始值为-1；图Gtemp与原图不同，**图Gtmp中存储着将最小生成树加平行边之后的欧拉图：若顶点i,j之间存在边，则Edge[i][j]与Edge[j][i]的值为2**（代表有两条边）。

因为Gtmp中必定存在欧拉回路，所以一定是边不重的圈的并，因此从一个顶点出发直接走（无论是不是桥），直到某个顶点没有边可走了（该顶点是第一个无边可走的顶点），此时说明回到了出发点（也就是欧拉回路的终点），则将该顶点及其之前的遍历过的顶点（且这些顶点无边可走了）放入栈中，一直退回到有边可以走的顶点（该顶点为某些圈的公共顶点），然后继续走，直到遍历完所有边。最后就会得到一条欧拉回路。

接着删除掉Euler数组中重复的顶点，并在最后加入Euler[0]（也就是起点，因为在删除过程中终点和起点相同，并将其删除了，所以最后要重新添加起点），得到一条哈密顿回路。最后将哈密顿回路数组Euler中每个相邻顶点的权值累加，得到一个近似解。

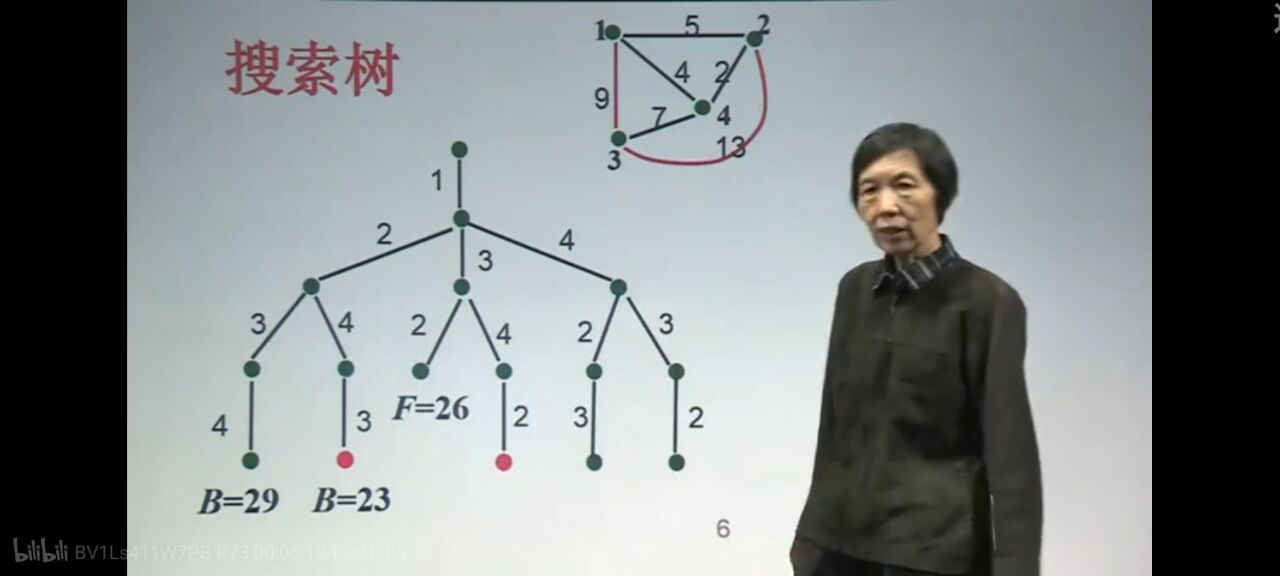
1. **分支定界法**

该算法实质上运用了深度优先搜索以及回溯法。设起点为v0，并设最小界为∞。**从v0出发进行深度优先搜索，并计算该搜索路径下的路径长度**，直到满足以下条件：

1)当前路径长度大于等于最小界，直接回溯。

2)当前路径中顶点个数已达到最大，此时需要比较该哈密顿回路下的路径长度是否小于最小界，若小于最小界，则将最小界更新为该哈密顿回路的路径长度，然后回溯；否则不做任何改变，直接回溯。

**最终回溯到起点v0，算法结束。**

以下图为例，设最小界为ans，初始值为∞，当前路径长度为len：

从顶点1开始算法，dfs至顶点2、3、4，此时len=29，最小界ans>len，则更新最小界，ans=29；然后回溯到顶点2，dfs至顶点4、3，此时len=23，更新ans=23。然后回溯至顶点1，dfs至顶点3、2，此时len=26，虽然没有遍历完所有顶点，但是len>ans，所以不需要继续往下进行，直接回溯至顶点3……

关键算法如下所示：

#define INFINITY 32767

int\* path, \* anspath;//前者用来存储当前的路径，如果符合条件则将其赋值给anspath

int ans;//最小界

int preans;//用来检测最小界是否有变化，若有变化,则说明路径要更新

int start;//起点

void dfs(int x, int temp, int cnt)

{

int i;

vis[x] = true;

path[cnt] = x + 1;

if (cnt == n - 1)//已访问的顶点数（因为cnt初始值为0）

{

preans = ans;

temp += graph[x][start - 1];

ans = ans < temp ? ans : temp;//取最小的

if (preans != ans) memcpy(anspath, path, sizeof(int) \* n);

return;

}

for (i = 0; i < n; i++)

{

if (graph[x][i] != INFINITY && !vis[i])

{

temp += graph[x][i];

if (temp < ans)

dfs(i, temp, cnt + 1);

temp -= graph[x][i];

vis[i] = false;

}

}

}

1. **模拟退火算法**

模拟退火算法来源于固体退火原理。

物理退火: 将材料加热后再经特定速率冷却，目的是增大晶粒的体积，并且减少晶格中的缺陷。材料中的原子原来会停留在使内能有局部最小值的位置，加热使能量变大，原子会离开原来位置，而随机在其他位置中移动。退火冷却时速度较慢，使得原子有较多可能可以找到内能比原先更低的位置。

模拟退火: 其原理也和固体退火的原理近似。模拟退火算法从某一较高初温出发，伴随温度参数的不断下降,结合概率突跳特性在解空间中随机寻找目标函数的全局最优解，即在局部最优解能概率性地跳出并最终趋于全局最优。

（1）加温过程：增强粒子运动，消除系统原本可能存在的非均匀态；

（2）等温过程：对于与环境换热而温度不变的封闭系统，系统状态的自发变化总是朝向自由能减少的方向进行，当自由能达到最小时，系统平衡；

（3）冷却过程：使粒子热运动减弱并逐渐趋于有序，系统能量逐渐下降，从而得到低能的晶体结构。

该过程采用**Metropolis方法**进行模拟：

温度恒定为T时，当前状态i转为新状态j，如果j状态的能量小于i，则接受状态j为当前状态；否则，如果概率p=exp（-(Ej-Ei)/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。

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。

本次实验中，使用模拟退火算法实现TSP的参数选取如下：

1)控制参数初值：t0 = 2000，即为初始温度

2)停止准则：连续2次循环中对路径无任何变动（优化或恶化的）时即停止算法运行。

3)冷却进度表中的控制参数t的衰减因子：α=0.7

4)Mapkob链长L：定长2000，即为循环次数

5)新解的产生：随机交换两个序号。任选序号u和v (u<v)，将u和v的顺序交换。

（Π1…Πu-1ΠuΠu+1 …Πv-1ΠvΠv+1… Πn）

变为

（Π1…Πu-1ΠvΠu+1 …Πv-1ΠuΠv+1… Πn）

程序过程为

**1)按顺序1-n的顺序初始化path;**

**2)将s置为0，L置为2000，从0到L循环，由当前路径生成新路径;**

**3)更新当前解，若新解路径长度小于当前解，就用新解替换当前解，否则用模拟退火算法判断是否接受新解。温度T进行递减。**

**4)若循环L次后解都没被替换，则s++，否则s = 0;**

**5)当s为2的时候，也就是连续两个Mapkob链没有变化时，或T（温度值）已降低至设定值时，退出程序，当前找到解就是最优解。**

1. **蚁群算法**

蚁群算法是受到对真实蚂蚁群觅食行为研究的启发而提出。生物学研究表明：一群相互协作的蚂蚁能够找到食物和巢穴之间的最短路径,而单只蚂蚁则不能。生物学家经过大量细致观察研究发现,蚂蚁个体之间的行为是相互作用相互影响的。蚂蚁在运动过程中,能够在它所经过的路径上留下一种称之为信息素的物质,而此物质恰恰是蚂蚁个体之间信息传递交流的载体。蚂蚁在运动时能够感知这种物质,并且习惯于追踪此物质爬行,当然爬行过程中还会释放信息素。一条路上的信息素踪迹越浓,其它蚂蚁将以越高的概率跟随爬行此路径,从而该路径上的信息素踪迹会被加强,因此,由大量蚂蚁组成的蚁群的集体行为便表现出一种信息正反馈现象。某一路径上走过的蚂蚁越多,则后来者选择该路径的可能性就越大。蚂蚁个体之间就是通过这种间接的通信机制实现协同搜索最短路径的目标的。

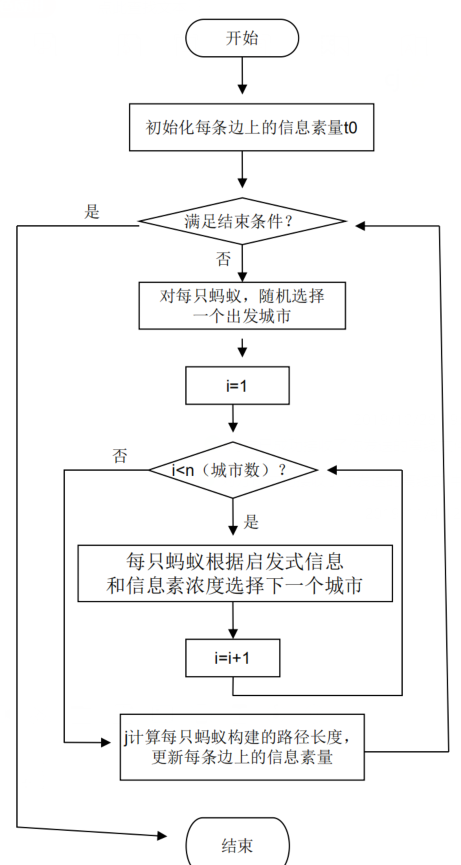
**该算法最重要的一点是蚂蚁如何寻找下一个要走的顶点。**

每只蚂蚁在寻找它所经过的路径中的下一个城市时，需要依据两个重要的信息：**信息素和启发式信息**。TSP 问题中的信息素表示在访问城市i后直接访问城市j的期望度，即该i,j之间存在多少信息素；启发式信息值一般与城市i和城市j的距离成反比。

然后根据轮盘赌选择法选择下一步要走的城市，并添加至路径中。接着根据路径的长度计算该次行走过程中留下多少信息素（路径长度越长，该路径上留下的信息素越少），再将之前留下的信息素进行一定程度上的挥发，并与之相加，更新该路径下的信息素。

最终达到最大迭代次数后，蚁群会走向信息素最多的那条最优路径，算法结束。

程序过程如下图所示：



1. **遗传算法**

遗传算法是模拟达尔文生物进化论的自然选择和遗传学机理的生物进化过程的计算模型，通过模拟自然进化过程搜索最优解。遗传算法是从代表问题可能潜在的解集的一个种群（population）开始的，初代种群产生之后，按照适者生存和优胜劣汰的原理，逐代（generation）演化产生出越来越好的近似解，在每一代，根据问题域中个体的适应度（fitness）大小选择个体，并借助于自然遗传学的遗传算子（genetic operators）进行组合交叉（crossover）和变异（mutation），产生出代表新的解集的种群。这个过程将导致种群像自然进化一样的后生代种群比前代更加适应于环境，末代种群中的最优个体经过解码（decoding），可以作为问题近似最优解。在TSP问题中，适应度一般与一般与城市i和城市j的距离成反比；种群可以理解为解向量的个数（与蚁群类似）。

程序过程如下：

1)**初始化：**设置进化代数计数器t=0、设置最大进化代数T、交叉概率、变异概率，M个个体作为初始种群；

2)**个体评价：**计算种群中各个个体的适应度；

3)**选择运算：**将选择算子作用于群体。以个体适应度为基础，选择最优个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。该实验中为轮盘赌选择法；

4)**交叉运算：**在交叉概率的控制下，对群体中的个体两两进行交叉。该实验中采用顺序交叉的方法：

先从第一双亲中随机选一个子串，然后将子串复制到一个空字串的相应位置，产生一个原始后代；接着删去第二双亲中子串已有的城市，得到原始后代需要的其它城市的顺序；最后按照这个城市顺序，从左到右将这些城市定位到后代的空缺位置上。

如下图所示：

5)**变异运算**：在变异概率的控制下，对群体中的个体两两进行变异，即对某一个体的基因进行随机调整，在该实验中是将该个体的随机两座城市i,j进行调换；

6)经过选择、交叉、变异运算之后得到下一代群体P1。还需要将这一代中最优解保留下来，就要用到另一个算子——**最优保存**算法。它是指将群体中最优的一部分个体不经过选择, 交叉和变异操作, 直接进入下一代, 以避免优秀个体损失。若当前群体中最优个体比历史最优个体适应度还高, 则以当前群体最优个体作为历史最优个体; 否则使用历史最优个体替换当前群体最差个体。

7)重复以上1-6，直到遗传代数为T，以进化过程中所得到的具有最大适应度个体作为最优解输出，终止计算。

1. **禁忌搜索**

禁忌搜索方法是是对人类智力过程的一种模拟。该算法通过引入一个灵活的存储结构和相应的禁忌准则来避免迂回搜索，并通过藐视准则来赦免一些被禁忌的优良状态，进而保证多样化的有效搜索以实现全局优化。

我觉得这样一个故事可以很大程度上帮助理解这一算法：兔子们找到了泰山，它们之中的一只就会留守在这里，其他的再去别的地方寻找。当兔子们再寻找的时候，一般地会有意识地避开泰山，因为他们知道，这里已经找过，并且有一只兔子在那里看着了。这就是禁忌搜索中“禁忌表”的含义。那只留在泰山的兔子一般不会就安家在那里了，它会在一定时间后重新回到找最高峰的大军，因为这个时候已经有了许多新的消息，泰山毕竟也有一个不错的高度，需要重新考虑，这个归队时间，在禁忌搜索里面叫做“禁忌长度”；如果在搜索的过程中，留守泰山的兔子还没有归队，但是找到的地方全是华北平原等比较低的地方，兔子们就不得不再次考虑选中泰山，也就是说，当一个有兔子留守的地方优越性太突出，超过了“best so far”的状态，就可以不顾及有没有兔子留守，都把这个地方考虑进来，这就叫“特赦准则”。

部分概念：

1)禁忌表：

禁忌表包括禁忌对象和禁忌长度。由于在每次对当前解的搜索中，需要避免一些重复的步骤，因此将某些元素放入禁忌表中，这些元素在下次搜索时将不会被考虑，这些被禁止搜索的元素就是禁忌对象。禁忌长度则是禁忌表所能接受的最多禁忌对象的数量

2) 特赦规则

在禁忌搜索算法的迭代过程中，会出现侯选集中的全部对象都被禁忌，或有一对象被禁，但若解禁则其目标值将有非常大的下降情况。在这样的情况下，为了达到全局最优，我们会让一些禁忌对象重新可选。这种方法称为特赦，相应的规则称为特赦规则。

3) 评价函数

用来评价当前解的好坏，TSP问题中是总旅程距离

程序过程如下：

**1)给以禁忌表H=∅ 并选定一个初始解x和最优解bestx；**

**2)在x的邻域N(x)中选出满足不受禁忌的候选集Can\_N(x)（邻域的解是通过变换x中城市的位置得到的）；在Can\_N(x)中选一个评价值最佳的解x1，x=x1；若路径x1的长度小于bestx，则令bestx=x1，否则不更新bestx;**

**若N(x)中存在一个解，它受禁忌的影响，但是它的评价值比bestx更好，则将其特赦。**

**3)将x1更新至禁忌表H，重复STEP2。满足停止规则时，停止计算，输出结果，一般停止规则为达到最大迭代次数。**

1. **测试样例**

测试数据：(教材P217，设a、b、c、d、e序号分别为1、2、3、4、5)

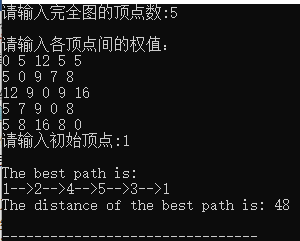
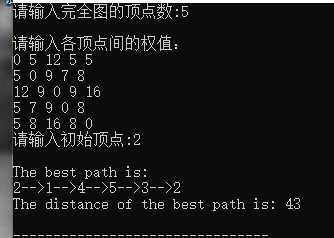
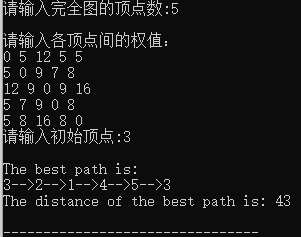
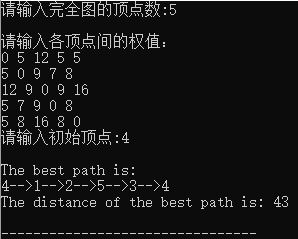
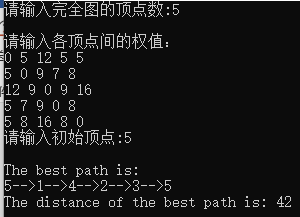
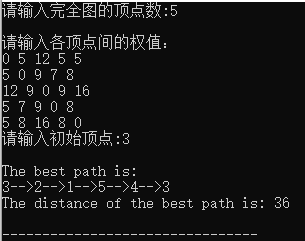
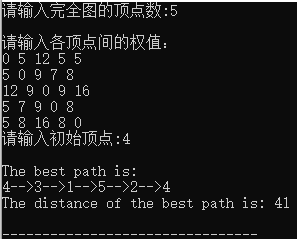
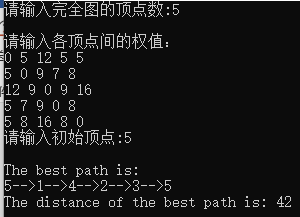
0 5 12 5 5

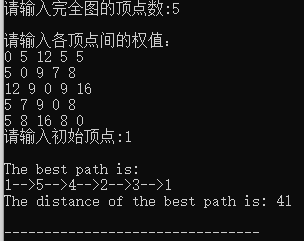
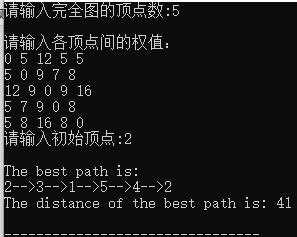
5 0 9 7 8

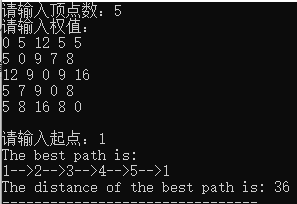
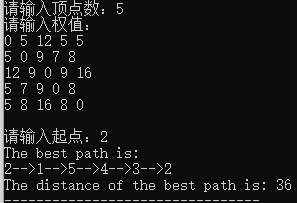
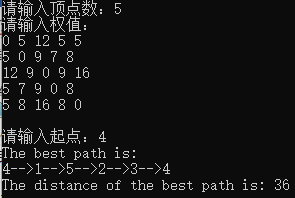
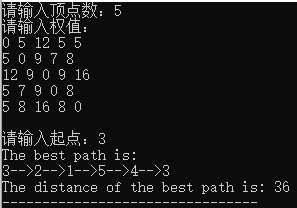
12 9 0 9 16

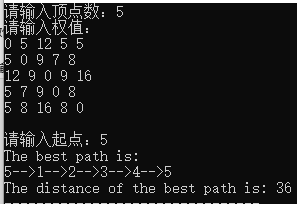
5 7 9 0 8

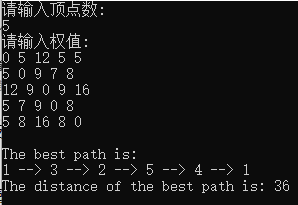
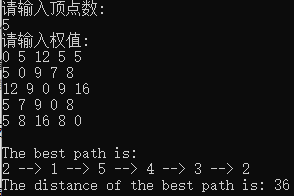
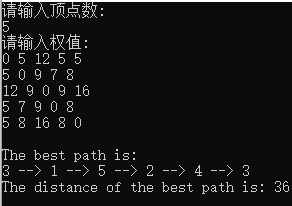
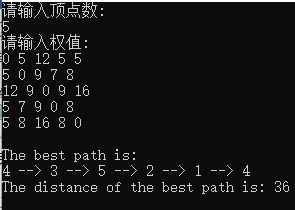
5 8 16 8 0

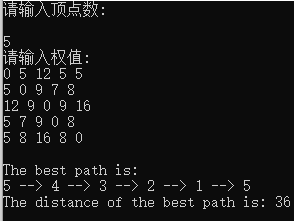
1. **最邻近法**
2. **最小生成树法**

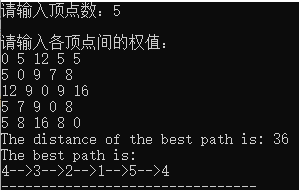
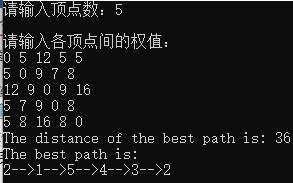
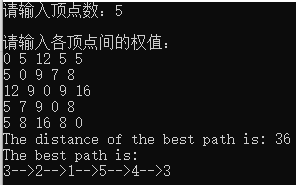
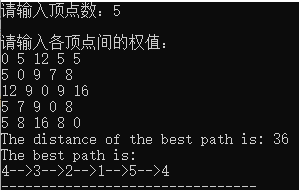


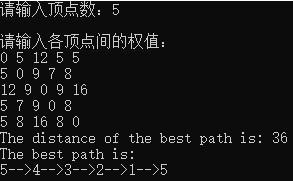
1. **分支定界法**

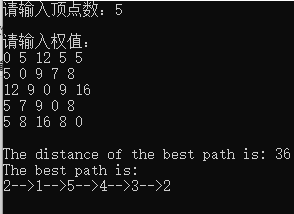
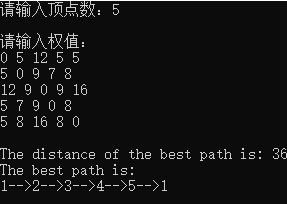
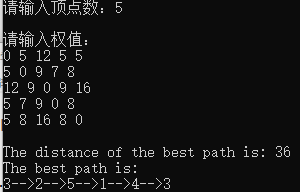
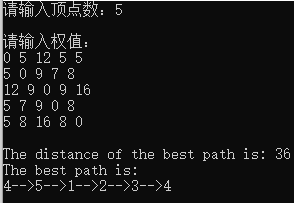
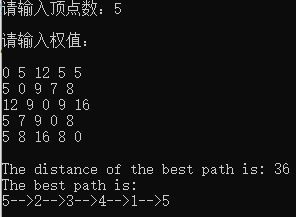


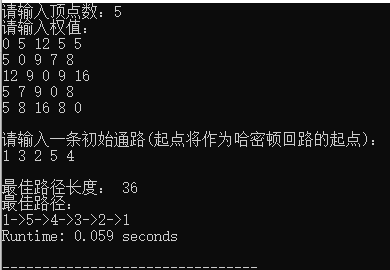
1. **模拟退火算法**

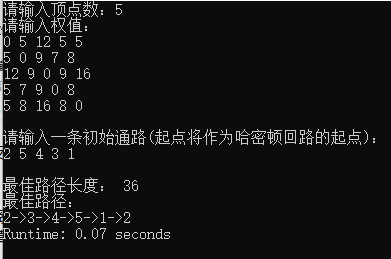


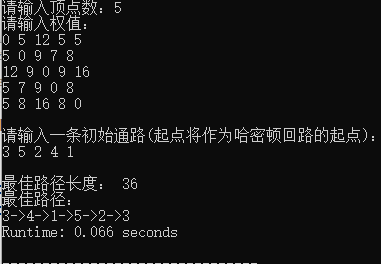
1. **蚁群算法**

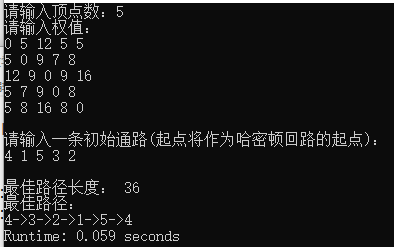


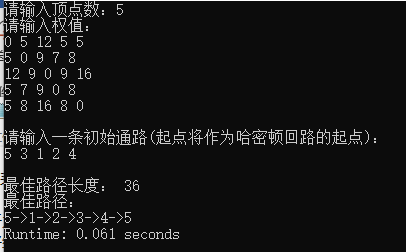
1. **遗传算法**
2. **禁忌搜索**







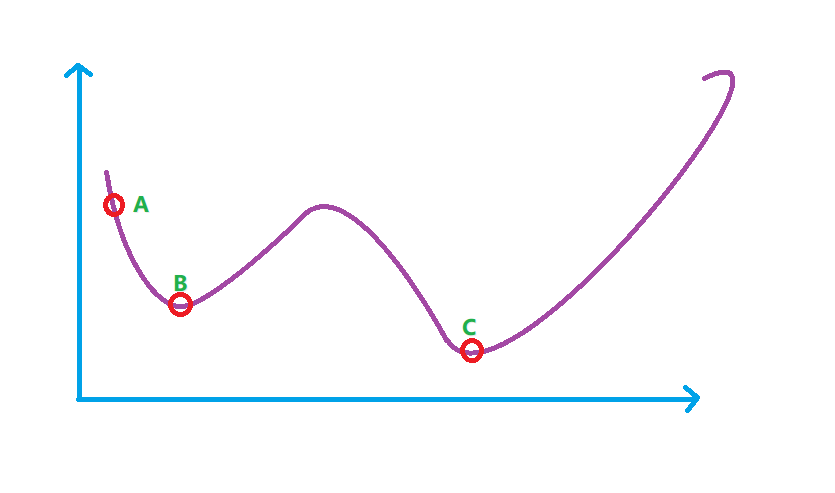




1. **总结与体会**

**总结:**

从测试结果来看，最邻近法和最小生成树法最终得到的结果大多都是局部最优解而非全局最优解，其他算法都能得到全局最优解。



可以看做是爬山算法，传统算法一般都是在寻求局部最优解，即从A走到B时，发现左右两边没有比B更优的解了，算法结束。

然而启发算法在B处，会以一定的概率去向右去寻找更优解（可能存在也可能不存在）。

本想在测试时对每个算法的时间效率进行计算，但是仔细想了想似乎没有太大意义，因为测试的数据也不是非常的规模，对时间的要求不是非常大，所以就着重关注每种算法得到解的近似程度。

**体会：**这次实验是在国庆期间完成的，时间非常充裕，因此看了许多相关的算法。在此之前没有接触过“启发式算法”，通过这次的学习让我体会到了它的魅力。它不依赖于问题，许多问题都可以通过启发式算法来解决。但是相对的，它的算法也更复杂，在实现每个算法时需要在网上收集许多资料。同时，因为不了解背后的数学原理，所以我只知其然,而不知其所以然，没有对这些算法有一个更深层次的理解。

总体而言，此次实验令我收获颇多，不仅是知识层面上，更多的是在算法的思想上。