北 京 林 业 大 学

2019 学年—2020 学年第 2 学期计算机算法设计与实践实验报告书

专业：计算机科学与技术(创新实验班)　班级：　计创 18

姓　　名：连月菡　　学　　号：181002222

实验地点：家　　任课教师：王春玲

实验题目：实验 6　TSP 问题的近似算法

实验环境：Visual Studio 2019 Community

一、 实验目的

（1）了解近似算法的局限性；
（2）掌握求解NP完全问题的一种方法：只对问题的特殊实例求解。

二、 实验内容

求解TSP问题的近似算法。

三、 实验步骤

1. 算法思路

选择一个求解 TSP 的近似算法，有很多，在这次实验中考虑使用一种选择最近邻的解法(贪心)，对于这种解法，运用多个测试用例进行测试，检验近似算法的稳定性,查看与最优解的差异。

对于 TSP 问题，应当定义一个结构体,存储一座城市的坐标信息。

a.初始化路径，使得初始的路径中，路径数组下标等于路径名(数字)

b. 令 i=1,遍历当前城市，j=i ,为遍历未访问过的城市,寻找距离 i-1 所对应的城市最近的城市 j,如果 j 的距离比目前的 mindist(最近距离)短，那么用 vismin 保存当前的 j。

c.每次将要结束内循环的遍历时,交换路径中 j 和 vismin 作为下标所对应的城市,以保证内循环 j 所对应的路径下标代表未访问过的城市。

2. 伪代码

```
1.  void fun(int begin) {//最近邻的近似算法
2.
3.       j = 1;//下标存储变量
4.      path[0] = begin;//路径起点
```

```
    for i <-0 : num - 1 :// 初始化路径
        if i != begin
            path[j] <- i
            j++
        endif
    endfor

        for i <- 1: num - 1 //寻找距离 path[i-1]最近的未访问城市
            mindist = 0x7ffffff//给最近距离赋一个较大值
            vismin = 0
                for j<- i: num
                    if path[i - 1]和 path[j]的距离 < 当前最近距离
                        当前最近距离 = path[i - 1]和 path[j]的距离
                        当前最近城市 = j
                    endif
                 交换 path[i] = path[vismin];
                endfor
        endfor
}
```

3. C++代码

```
void fun(int begin) {//最近邻的近似算法

    double mindist;//最近距离
    int vismin, temp;//最近城市，暂时的存储空间
    int i, j=1;//下标存储变量
    path[0] = begin;//路径起点
    for (i = 0; i < num; i++)// 初始化路径
        if (i != begin) {
            path[j] = i;  j++;
        }

    for (i = 1; i < num - 1; i++) {//寻找距离 path[i-1]最近的未访问城市
        mindist = 0x7ffffff;//给最近距离赋一个较大值
        vismin = 0;
        for (j = i; j < num; j++)
            if (dis(path[i - 1], path[j]) < mindist) {//找最近的城市
                mindist = dis(path[i - 1], path[j]);
                vismin = j;
            }
        temp = path[i];//交换
        path[i] = path[vismin];
        path[vismin] = temp;//保证下标靠后的城市
```

```
23.     }
24. }
```

时间复杂度为 O(n^2)

TSP 的最近邻算法近似比为:

$$\frac{2n-(k+1)/2}{2n-k} \leq \frac{2n-(n+1)/2}{2n-n} = \frac{3}{2} - \frac{1}{2n}.$$

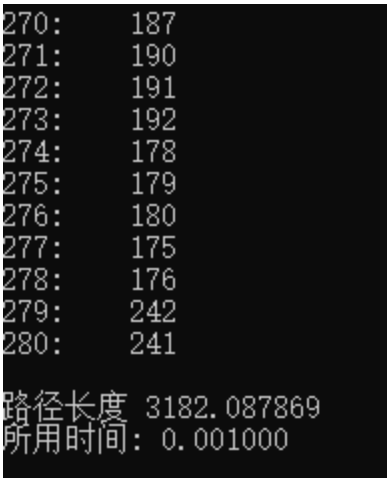
(n 是顶点数, k 是边的长度的个数)

4.测试结果

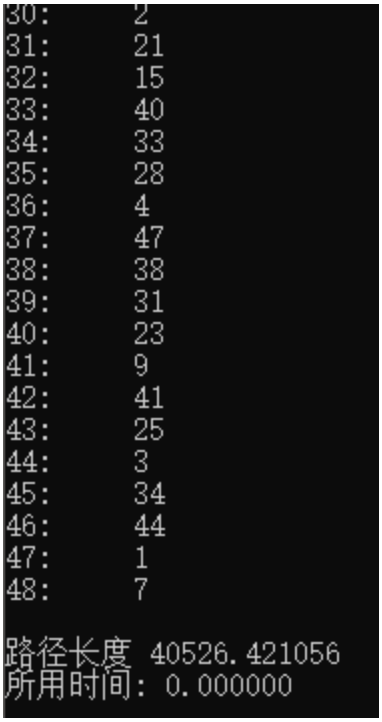
[测试用例下载]

[测试结果标准]

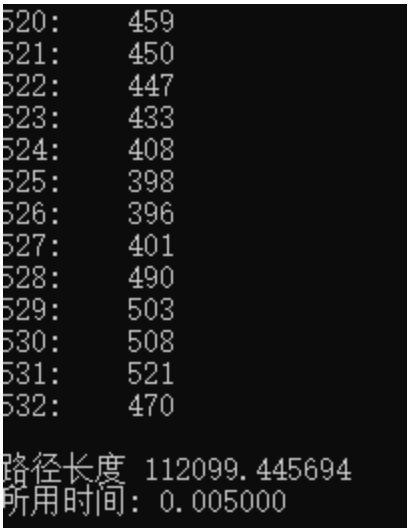
a280.tsp 最优解:2579



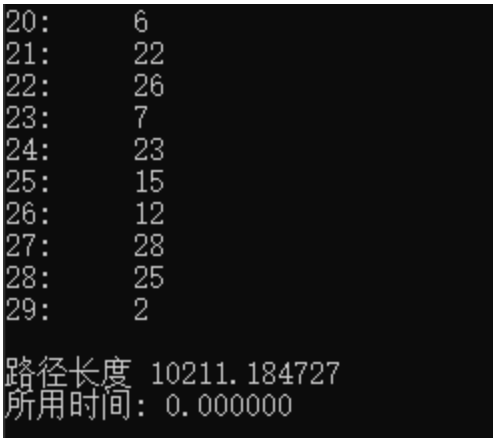
att48.tsp 最优解 10628



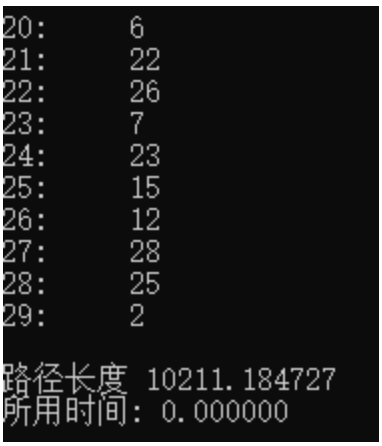
att532.tsp 最优解 27686



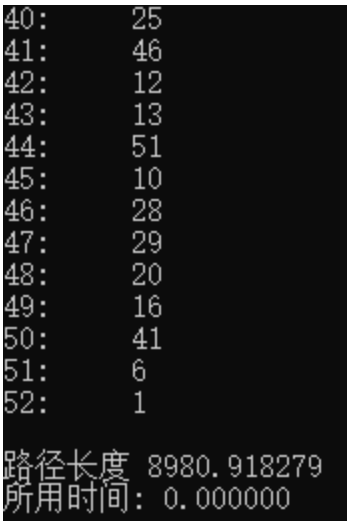
bayg29.tsp 最优解:1610



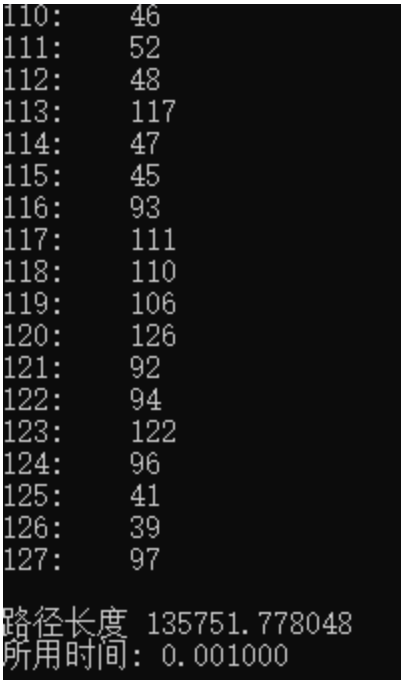
bays29.tsp 最优解:2020



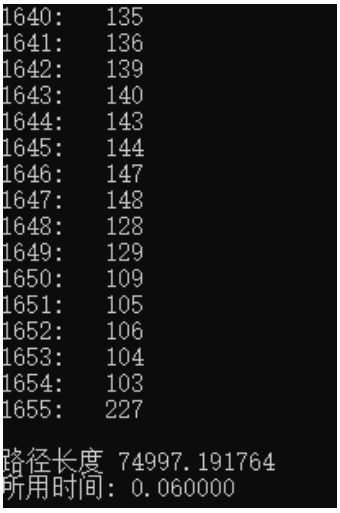
berlin52.tsp 最优解:7542



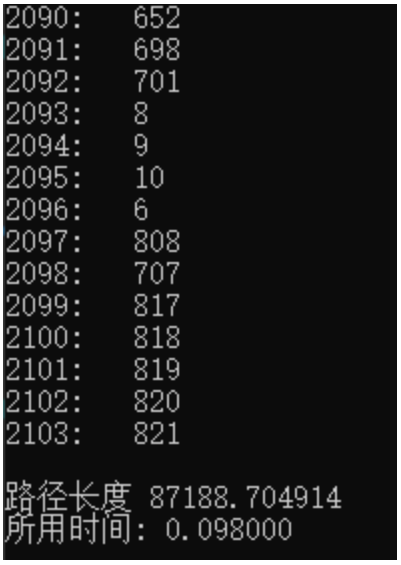
bier127.tsp 最优解:118282



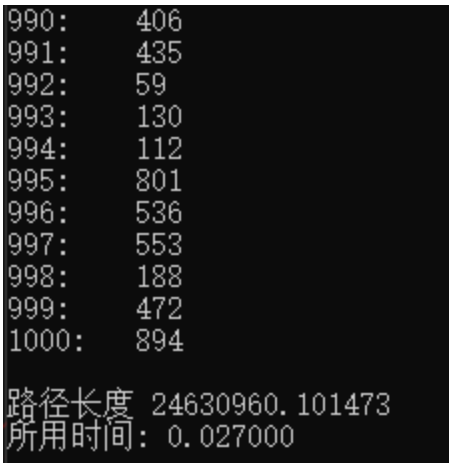
d1655.tsp 最优解:62128



d2103.tsp 最优解:80450

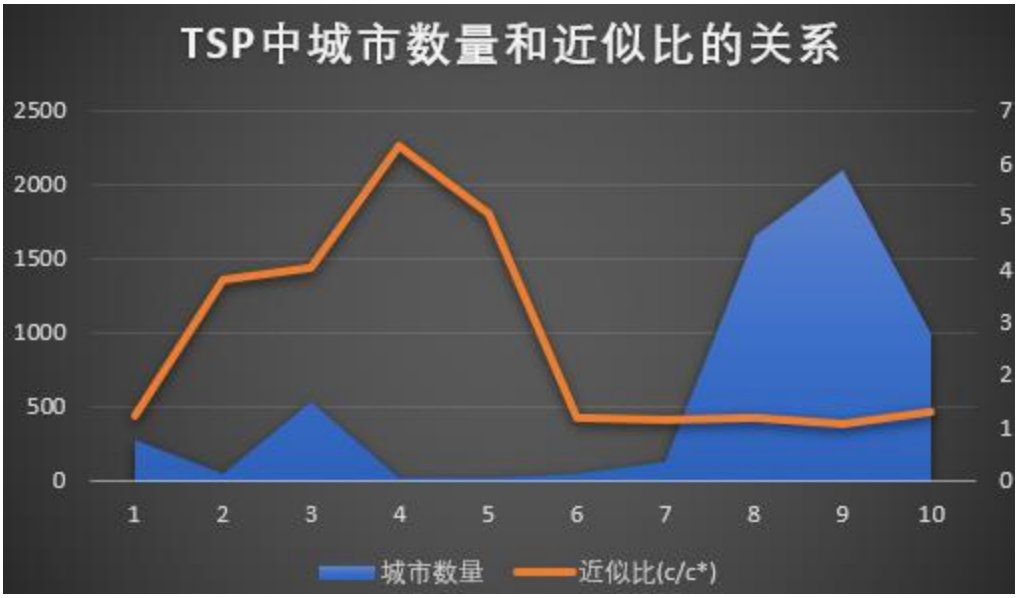


dsj1000.tsp 最优解:18659688



由上, 得到下表:

城市数量	280	48	532	29	29	52	127	1655	2103	1000
最优解c*	2579	10628	27686	1610	2020	7542	118282	62128	80450	18659688
近似最优值c	3182	40526	112099	10211	10211	8981	135752	74997	87189	24630960
近似比(c/c*)	1.23381	3.8131351	4.0489417	6.342236	5.05495	1.1908	1.1477	1.207137	1.08377	1.320009209



可以观察得到，在城市数量较大的时候，近似比趋于稳定，得到的结果较接近最优解。

四、实验心得

（1）理解满足三角不等式的 TSP 问题的特点及应用实例；

（2）设计近似算法求解满足三角不等式的 TSP 问题，并求出近似比；

（3）将求解满足三角不等式的 TSP 问题的近似算法应用于一般的 TSP 问题，考察它的近似性能，理解为什么说 TSP 问题是 NP 问题中最难的一个问题。