

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
```

```

5.     for i <- 0 : num - 1 :// 初始化路径
6.         if i != begin
7.             path[j] <- i
8.             j++
9.         endif
10.    endfor
11.
12.    for i <- 1: num - 1 //寻找距离 path[i-1]最近的未访问城市
13.        mindist = 0x7fffffff//给最近距离赋一个较大值
14.        vismin = 0
15.        for j<- i: num
16.            if path[i - 1]和 path[j]的距离 < 当前最近距离
17.                当前最近距离 = path[i - 1]和 path[j]的距离
18.                当前最近城市 = j
19.            endif
20.            交换 path[i] = path[vismin];
21.        endfor
22.    endfor
23. }

```

3. C++代码

```

1. void fun(int begin) { //最近邻的近似算法
2.
3.     double mindist; //最近距离
4.     int vismin, temp; //最近城市, 暂时的存储空间
5.     int i, j=1; //下标存储变量
6.     path[0] = begin; //路径起点
7.     for (i = 0; i < num; i++) // 初始化路径
8.         if (i != begin) {
9.             path[j] = i; j++;
10.        }
11.
12.    for (i = 1; i < num - 1; i++) { //寻找距离 path[i-1]最近的未访问城市
13.        mindist = 0x7fffffff; //给最近距离赋一个较大值
14.        vismin = 0;
15.        for (j = i; j < num; j++)
16.            if (dis(path[i - 1], path[j]) < mindist) { //找最近的城市
17.                mindist = dis(path[i - 1], path[j]);
18.                vismin = j;
19.            }
20.        temp = path[i]; //交换
21.        path[i] = path[vismin];
22.        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

```
270: 187
271: 190
272: 191
273: 192
274: 178
275: 179
276: 180
277: 175
278: 176
279: 242
280: 241

路径长度 3182.087869
所用时间: 0.001000
```

att48.tsp 最优解 10628

```
30: 2
31: 21
32: 15
33: 40
34: 33
35: 28
36: 4
37: 47
38: 38
39: 31
40: 23
41: 9
42: 41
43: 25
44: 3
45: 34
46: 44
47: 1
48: 7

路径长度 40526.421056
所用时间: 0.000000
```

att532.tsp 最优解 27686

```
520: 459
521: 450
522: 447
523: 433
524: 408
525: 398
526: 396
527: 401
528: 490
529: 503
530: 508
531: 521
532: 470

路径长度 112099.445694
所用时间: 0.005000
```

bayg29.tsp 最优解:1610

```
20: 6
21: 22
22: 26
23: 7
24: 23
25: 15
26: 12
27: 28
28: 25
29: 2

路径长度 10211.184727
所用时间: 0.000000
```

bays29.tsp 最优解:2020

```
20: 6
21: 22
22: 26
23: 7
24: 23
25: 15
26: 12
27: 28
28: 25
29: 2

路径长度 10211.184727
所用时间: 0.000000
```

berlin52.tsp 最优解:7542

```
40: 25
41: 46
42: 12
43: 13
44: 51
45: 10
46: 28
47: 29
48: 20
49: 16
50: 41
51: 6
52: 1

路径长度 8980.918279
所用时间: 0.000000
```

bier127.tsp 最优解:118282

```
110: 46
111: 52
112: 48
113: 117
114: 47
115: 45
116: 93
117: 111
118: 110
119: 106
120: 126
121: 92
122: 94
123: 122
124: 96
125: 41
126: 39
127: 97

路径长度 135751.778048
所用时间: 0.001000
```

d1655.tsp 最优解:62128

```
1640: 135
1641: 136
1642: 139
1643: 140
1644: 143
1645: 144
1646: 147
1647: 148
1648: 128
1649: 129
1650: 109
1651: 105
1652: 106
1653: 104
1654: 103
1655: 227

路径长度 74997.191764
所用时间: 0.060000
```

d2103.tsp 最优解:80450

```
2090: 652
2091: 698
2092: 701
2093: 8
2094: 9
2095: 10
2096: 6
2097: 808
2098: 707
2099: 817
2100: 818
2101: 819
2102: 820
2103: 821

路径长度 87188.704914
所用时间: 0.098000
```

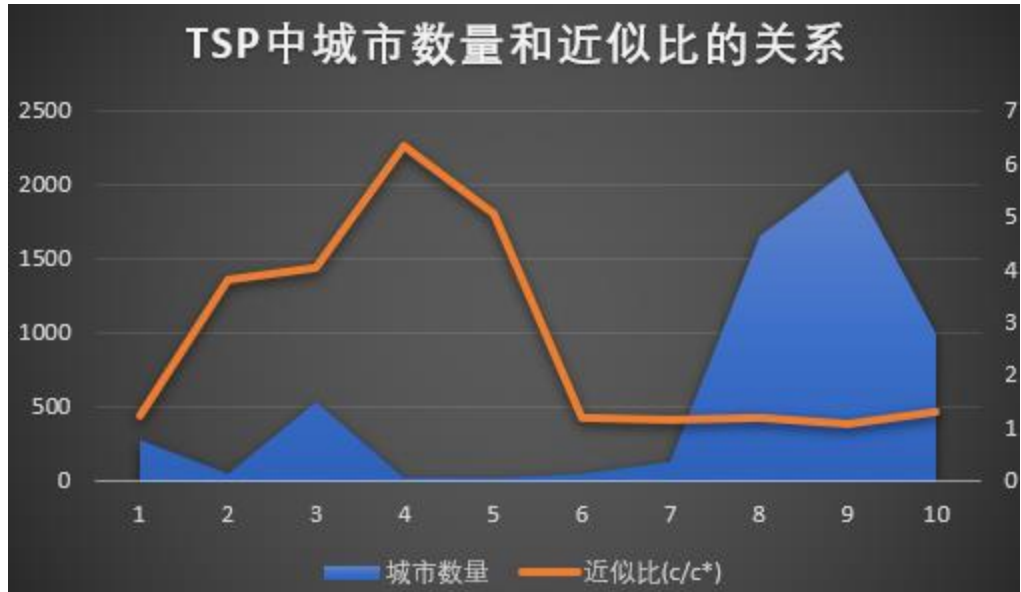
dsj1000.tsp 最优解:18659688

```
990: 406
991: 435
992: 59
993: 130
994: 112
995: 801
996: 536
997: 553
998: 188
999: 472
1000: 894

路径长度 24630960.101473
所用时间: 0.027000
```

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。