北京邮電大学
Beijing University of Posts and Telecommunications

《算法分析—第四章实验》

实验报告

姓　　名　　袁洁　　胡敏臻

学　　号　　2019211426　2019211424

班　　级　　2019211307　2019211307

专　　业　　计算机科学与技术

1 实验内容

1.1 基于贪心法的凸多边形三角划分

利用给出的 21 凸多边形顶点数据、 29 凸多边形顶点数据，以顶点间的地理距离作为连接 2 个顶点的边、弦的权值，对这 2 个凸多边形采用贪心法/启发式方法进行三角剖分。

*本题已在上一章中完成了启发式方法的凸多边形三角划分，在本章中不再多加赘述。

1.2 哈夫曼编码

利用 “附件 2.哈夫曼编码输入文本”给出的文本信息。

方案 1：将文本中的数字 0-9、空格、标点符号用“#”替换，统计 26 个英文字母及#出现的频率，对{a, b, c,..,x, y, z, #}，设计哈夫曼编码。按照哈夫曼编码，对输入文本重新编码。计算采用哈夫曼编码，输入文本需要的存储比特数，并与定长编码方式需要的存储比特数进行比较。

要求：给出如下结果

1. {a, b, c,..,x, y, z, #}中各成员在文本中的出现频率和哈夫曼编码
2. 采用哈夫曼编码、定长编码，输入文本需要的存储比特数

方案 2：文本中的数字 0-9、空格、标点符号等不替换，统计 26 个英文字母、数字 0-9、空格、标点符号的出现的频率。对{a, b, c,..,x, y, z, 0,…,9, 空格,标点符号}，设计哈夫曼编码。按照哈夫曼编码，对输入文本重新编码。计算采用哈夫曼编码，输入文本需要的存储比特数，并与定长编码方式需要的存储比特数进行比较。

要求：给出如下结果

1. {a, b, c,..,x, y, z, 0,…,9, 空格,标点符号}中各成员在文本中的出现频率和哈夫曼编码
2. 采用哈夫曼编码、定长编码，输入文本需要的存储比特数

1.3 单源最短路径

从昆明 LTE 网络中，选取部分基站，计算基站间的距离，在部分基站间引入边，得到

1）22 个基站顶点组成的图
2）42 个基站顶点组成的图

要求：

对 22 个基站顶点组成的图，以基站 567443 为源点

1. 计算 567443 到其它各点的单源最短路径；
2. 计算 567443 到 33109 的最短路径

对 42 个基站顶点组成的图，以基站 565845 为源点

1. 计算 565845 到其它各点的单源最短路径
2. 计算 565845 到 565667 的最短路径

1.4 最小生成树

从昆明 LTE 网络中，选取部分基站，计算基站间的距离，在部分基站间引入边，得到

1）22 个基站顶点组成的图
2）42 个基站顶点组成的图

生成这 2 个图的最小生成树，要求：

1. 采用 K 算法，或 P 算法；
2. 给出最小生成树的成本/代价/耗费 cost
3. 做图，呈现出最小生成树

2 哈夫曼编码

2.1 设计思路

哈夫曼编码也是最优前缀码。对字母表中的每个字符规定一个 0、1 串作为其代码，要求任一字符的代码都不是其它字符代码的前缀。根据文件中字符出现的频率表建立用变长的 0、1 串（前缀码）表示各字符的最优表示方式。出现频率高的字符用较短的编码，出现频率较低的字符用较长的编码。

构造哈夫曼树的原理：

1. 以编码字符集中每个字符 c 的出现频率 f(c),作为贪心选择依据，对字符集进行由小到大的排序，每个字符对应于一个只包含一个结点的子树
2. 先合并最小频率的 2 个字符对应的子树，计算合并后的子树中这个字符出现的频率总和
3. 重新排序各个子树
4. 对上述排序后的子树序列进行合并
5. 重复上述过程，将全部结点合并成 1 棵完整的二叉树，称为编码树 T
6. 对二叉树中的边赋予 0、1，得到各字符的变长编码

2.2 算法正确性证明

2.2.1 贪心选择性证明

假设：

1) b、c 是 T 中最深叶子且互为兄弟，且 f(b)<=f(c)；

2) 已知 C 中 2 个最小频率字符 f(x)<=f(y)， 但在 T 中，x、y 有可能并非最深结点。由于 x、y 具有最小频率, 故 f(x)<=f(b), f(y)<=f(c)

在 T 中交换 b 和 x 的位置得到 T1，在 T1 中交换 c 和 y 的位置，得到 T’。可以证明 B(T)—B(T1) ≤0，即第一步交换不会增加平均码长；B(T1)—B(T’) ≤0，即第二步交换也不会增加平均码长。故 T’的码长仍然是最短的，即 T’是最优前缀码，并且其最小频率的 x、y 具有最深的深度（最长的编码)，且只有最后一位不同。

2.2.2 最优子结构证明

对 T 中 2 个互为兄弟的叶节点 x、y，z 为其父节点，将 z 看做频率为 f(z)=f(x)+f(y)的字符，则 T’=T-{x, y}是子问题 C’= (C-{x,y}) ∪{z} 的最优编码。

证明关键点 1：T 的平均码长 B(T)可用子树 T’的平均码长 B(T’) 表示。B(T) = B(T’) + 1*f(x) + 1*f(y) 递推表达式，原问题最优值与子问题最优值之间的关系

证明关键点 2： T’ 所表示的 C’的前缀码的码长 B(T’)是最短/最优的

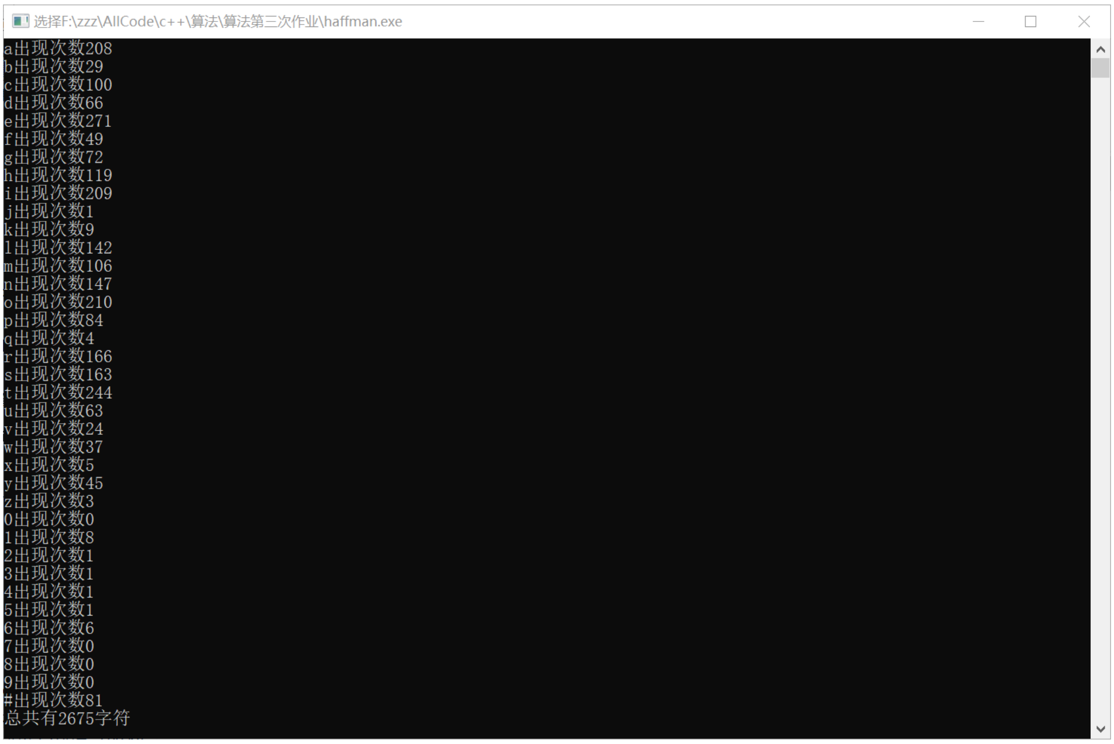
采用反证法证明关键点 2 假设：有另一个 T’’，是子问题 C’的最优前缀码，即 B(T’)>B(T’’)，节点 z 在 T’’ 还是一个叶节点。在 T’’ 中将 z 替换为其子节点

x、y，得到 T’”， T’”是关于原问题 C 的 1 个解，同时 B(T’”)<B(T)，与 T 是最优解矛盾。

2.3 实验结果

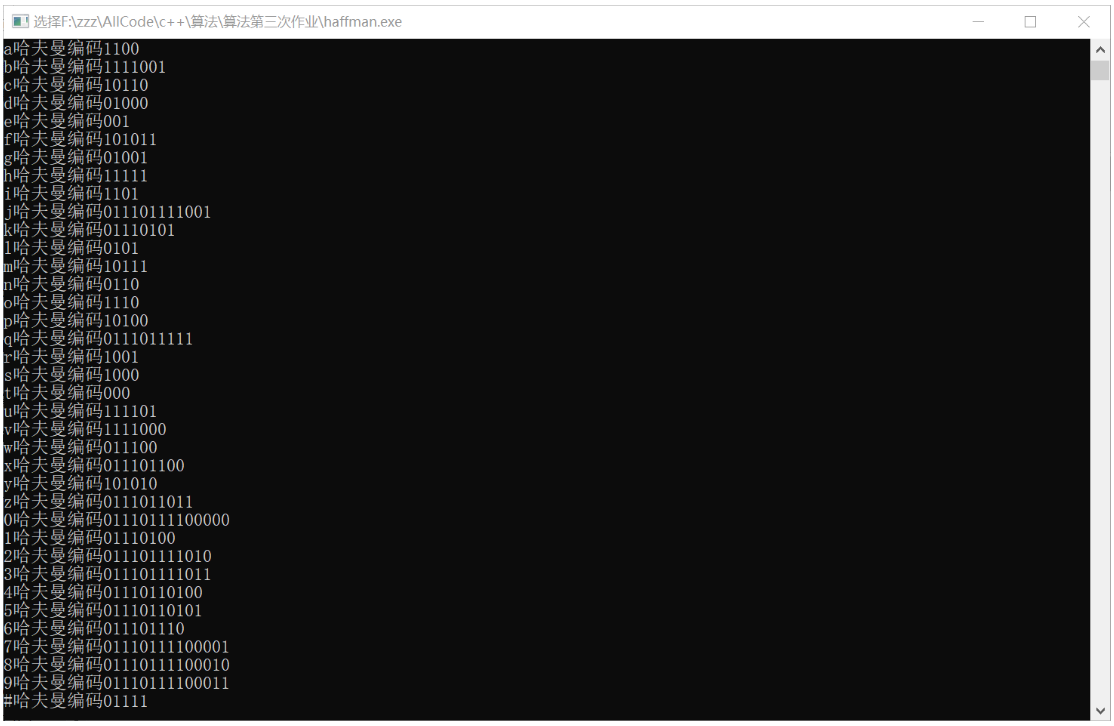
2.3.1 {a, b, c,..,x, y, z,0,…,9, #}中各成员在文本中的出现频率和哈夫曼编码

出现频率：



```
选择F:\zzz\AllCode\c++\算法\算法第三次作业\haffman.exe
a出现次数208
b出现次数29
c出现次数100
d出现次数66
e出现次数271
f出现次数49
g出现次数72
h出现次数119
i出现次数209
j出现次数1
k出现次数9
l出现次数142
m出现次数106
n出现次数147
o出现次数210
p出现次数84
q出现次数4
r出现次数166
s出现次数163
t出现次数244
u出现次数63
v出现次数24
w出现次数37
x出现次数5
y出现次数45
z出现次数3
0出现次数0
1出现次数8
2出现次数1
3出现次数1
4出现次数1
5出现次数1
6出现次数6
7出现次数0
8出现次数0
9出现次数0
#出现次数81
总共有2675字符
```

哈夫曼编码：



```
选择F:\zzz\AllCode\c++\算法\算法第三次作业\haffman.exe
a哈夫曼编码1100
b哈夫曼编码1111001
c哈夫曼编码10110
d哈夫曼编码01000
e哈夫曼编码001
f哈夫曼编码101011
g哈夫曼编码01001
h哈夫曼编码11111
i哈夫曼编码1101
j哈夫曼编码011101111001
k哈夫曼编码01110101
l哈夫曼编码0101
m哈夫曼编码10111
n哈夫曼编码0110
o哈夫曼编码1110
p哈夫曼编码10100
q哈夫曼编码0111011111
r哈夫曼编码1001
s哈夫曼编码1000
t哈夫曼编码000
u哈夫曼编码111101
v哈夫曼编码1111000
w哈夫曼编码011100
x哈夫曼编码011101100
y哈夫曼编码101010
z哈夫曼编码0111011011
0哈夫曼编码01110111100000
1哈夫曼编码01110100
2哈夫曼编码011101111010
3哈夫曼编码011101111011
4哈夫曼编码01110110100
5哈夫曼编码01110110101
6哈夫曼编码011101110
7哈夫曼编码01110111100001
8哈夫曼编码01110111100010
9哈夫曼编码01110111100011
#哈夫曼编码01111
```

2.3.2 采用哈夫曼编码、定长编码，输入文本需要的存储比特数

```
哈夫曼编码存储比特数11563
定长编码存储比特数16050
```

3 单源最短路径

3.1 设计思路

设置集合 S={i}⊆V，记录已经得到最短路径的顶点 i（已经求出 v 至 i 的最短路径），对图 G(V, E)中某一个顶点 u∈V，将从源 v 到 u 且中间只经过 S 中的顶点的路径称为从源点 v 到 u 的特殊路径，用数组 dist 记录 v 到图中各点 u 的特殊路径长度，记为 dist[u]。采用贪心选择策略，从 V-S 中挑选具有最小 dist[uk] 的顶点 uk，将 uk 加入 S，S={uk}∪S，当 S=V 时，获得源点 v 至图中全部其它 n-1 个顶点的最短路径，算法结束。

3.2 算法正确性证明

3.2.1 贪心选择性证明

在迭代求解过程中，顶点 u 是遇到的第 1 个满足 **d(v,u) < dist[u]** 的顶点，即：d(v,u) ≠dist[u]，全局最优路径经过 S 之外的顶点，设从 v 到 u 的全局最短路径上，第 1 个属于 V-Si 的顶点为 x，对 v 到 u 的全局最短路径 d(v ,u) ，根据 d(v, x) + distance(x, u) = d(v ,u) ，distance(x, u) >0，有 **dist[x]< dist[u]**，但是根据路径 p 构造方法，在下图所示情况下，u、x 都在集合 Si 之外，即 u、x 都属于 V-Si，贪心选择 S 外顶点时，u 被选中，并没有选 x，根据扩展 Si 的原则：选 dist 最小的顶点加入 Si，说明此时

dist[u] ≤ dist[x]

矛盾。

3.2.2 最优子结构证明

对顶点 u，考察将顶点 u 加到集合 Si 之前和之后，dist[u]的变化，添加 u 之前对应的顶点集合为 Si，加入 u 之后的顶点集合为 Si+1，对另外 1 个节点 i，考察 u 的加入对 dist[i]的影响：

情况 1. 假设添加 u 后, 出现 1 条从 v 到 i 的新路，该路径先由 v 经过老 Si 中的顶点到达 u，再从 u 经过一条直接边到达 I。如果 dist[u]+ c[u][i] < 原来的 dist[i]，则算法用 dist[u] + c[u][i] 替代 dist[i]，得到新的 dist[i]；否则， dist[i]不更新。

情况 2. 如果新路径如下图所示，先经过 u，再回到 Si 中的 x，由 x 直接到达 i。x 处于老的 Si 中，故 dist[x]已经是由 v 到 x 的最短路径的长度，x 比 u 先加入 Si，dist[x]=d(v,x)是全局最短路径，因此 dist[x] ≤ dist[u]＋path(u,x)。此时，从源点 v 到 i 的最短路径 dist[i]=dist[x]+c[x,i]小于路径（v, u, x, i）的长度，因此算法更新 dist[i]时不受路径(v, u, x, i)影响，即 u 的加入对 dist[i]无影响。

因此，无论算法中 dist[u]的值是否变化，它总是关于当前顶点集合 S 的到顶点 u 的最短路径。虽然只针对子集 S，不一定是全局最优。

也就是说：对于加入 u 之前、之后的新老 S 所对应的 2 个子问题，算法执行过程保证了 dist[u]始终是 u 相对于 S 的最优解。当算法结束时，S=V，dist(u)成为全局最优解。

3.3 实验结果

3.3.1 22 基站 567443 到 33109 最短路径：

```
567443
567443 566750 567439 33109 Program
    ended with exit code: 0
```

3.3.2 42 基站 565845 到 565667 最短路径

```
565845
565845 567526 567500 565675 565551
    565633 565667 Program ended with
    exit code: 0
```

4 最小生成树

4.1 设计原理

Prim 算法：

Step1. 设置顶点集合 S={1}，边集合 T=ϕ

Step2. 当 S⊂V，即 S 是 V 的真子集时，作如下的贪心选择。选取满足：i∈S，j ∈V-S，且 c[i][j]最小的边<i, j>，将顶点 j 添加到 S 中，边<i, j>加到边集 T 中

Step3. 重复上述过程，直到 S=V 为止

4.2 算法正确性证明

4.2.1 贪心选择性证明

假设对 G 的任意一个最小生成树 T，针对点集 U 和 V-U， (u, v) E 为横跨这 2 个点集的最小权边， T 不包含该最小权边<u, v>，但 T 包括节点 u 和 v，将<u, v>添加到树 T 中，树 T 将变为含回路的子图，并且该回路上有一条不同于<u, v> 的边<u’, v’>, u’∈U, v’ ∈V-U，将 T 中的边<u’, v’>替换为(u, v)，得到 T‘，由于对边和，耗费满足 c[u][v] ≤ c[u’][v’]，因此用较小耗费的边<u,v>替换后得到的树 T’的耗费更小，即：

T’耗费 ≤ T 的耗费

，这与 T 是任意最小生成树的假设相矛盾

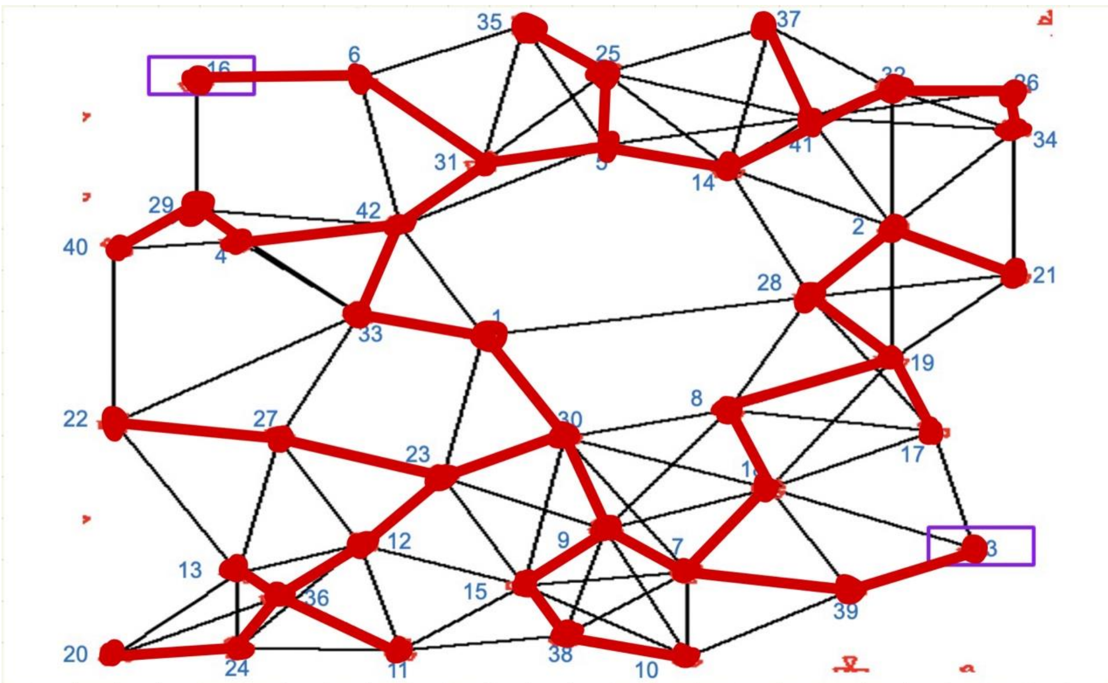
4.2.2 最优子结构证明

假设对 G 的任意一个最小生成树 T，针对点集 U 和 V-U，U 为已加入最小生成树的顶点，子问题为在 V-U 中构建最小生成树，如果 V-U 有耗费更小的生成树，则将原解的子树替换，依旧得到最优解。

4.3 实验结果

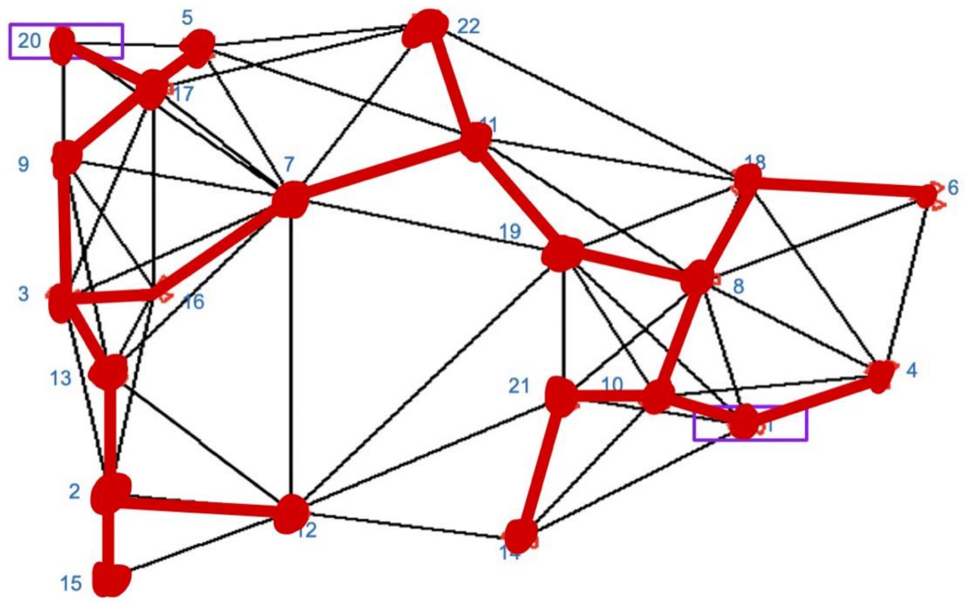
4.3.1 42 基站

```
13027.029572
2 28
3 39
4 42
5 31
6 31
7 9
8 18
9 30
10 38
11 36
12 23
13 36
14 5
15 9
16 6
17 19
18 7
19 8
20 24
21 2
22 27
23 30
24 36
25 5
26 32
27 23
28 19
29 4
30 1
31 42
32 41
33 1
34 26
35 25
36 12
37 41
38 15
39 7
40 29
41 14
42 33
```



4.3.2 22 基站

```
6733.567764
2 13
3 16
4 1
5 17
6 18
7 11
8 10
9 3
10 1
11 19
12 2
13 3
14 21
15 2
16 7
17 9
18 8
19 8
20 17
21 10
22 11
```



5 时间、空间复杂性分析

5.1 哈夫曼编码

时间复杂度为 O(nlogn)

5.2 单源最短路径

时间复杂度为 O(n^2)

空间复杂度 O(n)

5.3 最小生成树

时间复杂度为 O(n^2)

空间复杂度 O(n)

6 实验总结

在本次实验中，每个任务我们都共同付出了较多的努力。每个人的贡献度为 50%+50%。

本次实验完成了三个任务。除了在上一章中已经实现的启发式凸多边形最优三角划分外，我们实现了哈夫曼编码，单源最短路径中的 Dijkstra 算法和最小生成树中的 Prim 算法。这三个算法都是贪心策略中较为典型的问题。在自己尝试编写之后，对这些算法更为熟悉。通过比较哈夫曼最优前缀编码和定长前缀编码，我们发现，哈夫曼编码更加节约空间，减少开销。

改进思路：

1、在构建哈夫曼树时，在书上所给方法中，需要将所有的权重在每次排列后都全排列。在最后实现的时候，没有采用排序的方法，而是直接从已有的结点中挑出最小的两个结点。这样挑选的复杂度是 O(n)，而排序的平均时间复杂度为 O(nlogn)，因此，直接选择权重最小的两个结点反而比先排序要更快一点。并且本次构建哈夫曼树构建的范围为{a,b,c…,x,y,z,0…,9,#}，在方案 1 的基础之上范围扩大，一共需要表示 37 个叶结点。