

《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

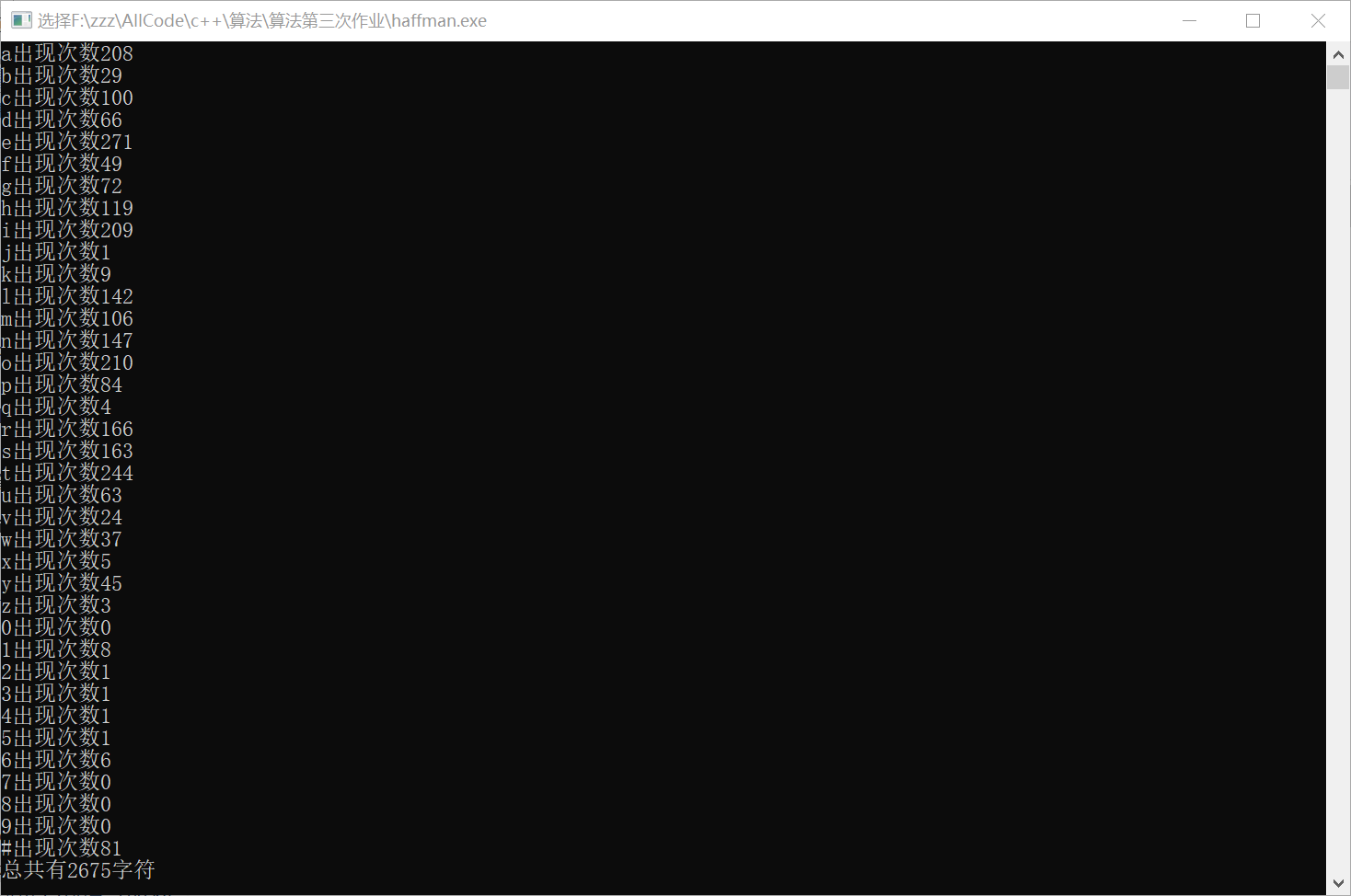
证明关键点2： T’ 所表示的C’的前缀码的码长B(T’)是最短/最优的

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。

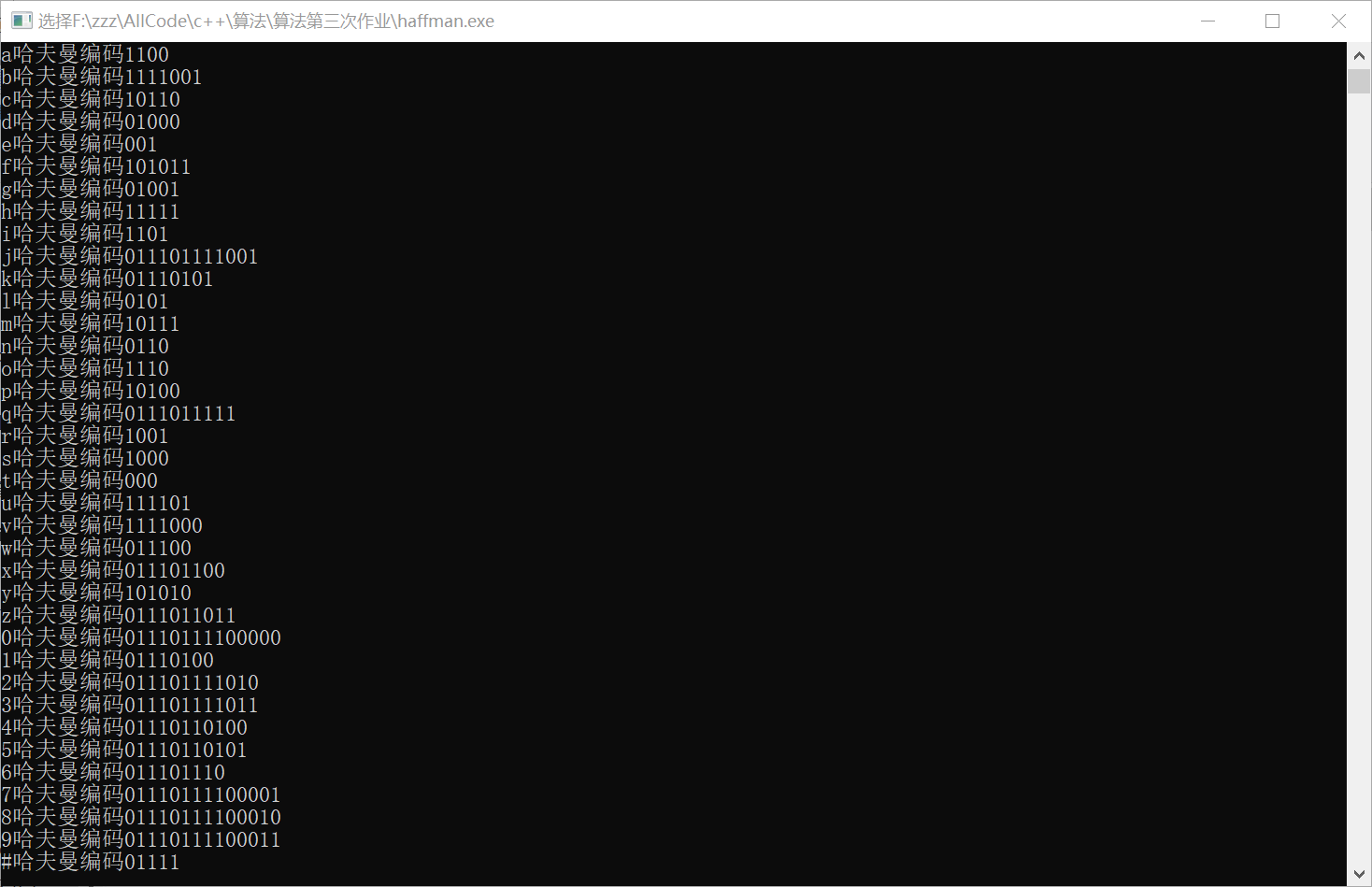
**2.3 实验结果**

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：



哈夫曼编码：



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



**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：

文本

描述已自动生成

3.3.2 42基站565845到565667最短路径

文本

中度可信度描述已自动生成

**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**

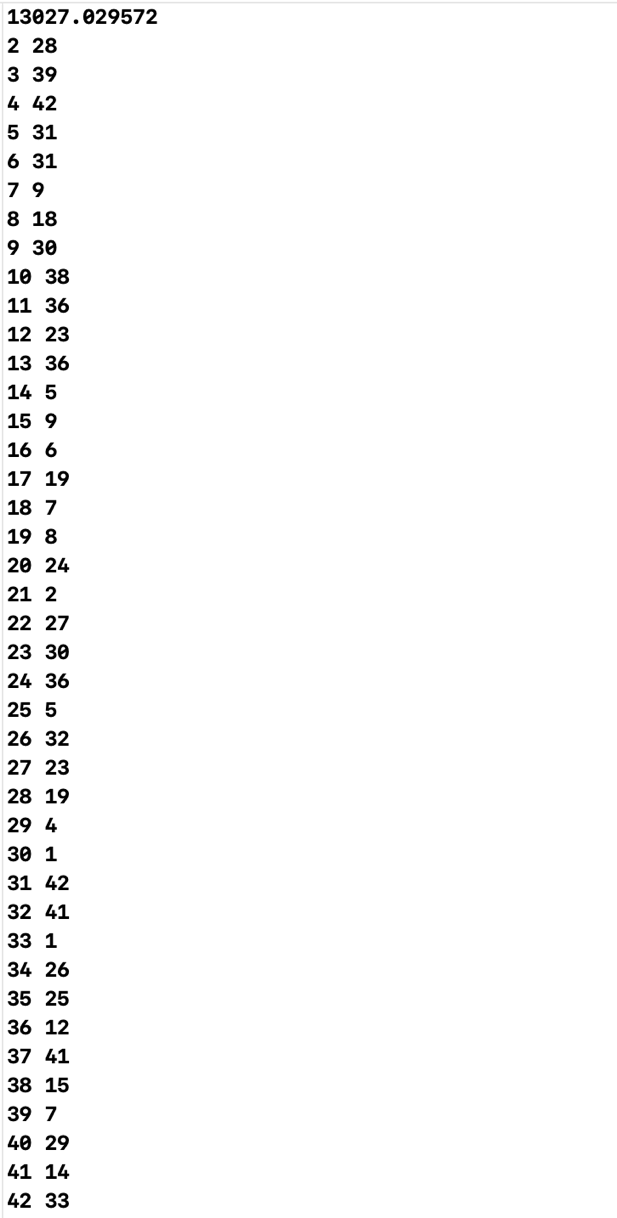
，这与T是任意最小生成树的假设相矛盾

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



图表

描述已自动生成

**4.3.2 22基站**

表格

中度可信度描述已自动生成

图表, 雷达图

描述已自动生成

**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。