哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称：数据结构与算法

课程类型：必修

实验项目：树形结构及其应用

实验题目：哈夫曼编码与译码方法

实验日期：2020.11.01

班级：1903002

学号：1190200523

姓名：石翔宇

|  |  |  |
| --- | --- | --- |
| 设计成绩 | 报告成绩 | 指导老师 |
|  |  | 张岩 |

**一、实验目的**

哈夫曼编码是一种以哈夫曼树（最优二叉树，带权路径长度最小的二叉树）为基础变长编码方式。其基本思想是：将使用次数多的代码转换成长度较短的编码，而使用次数少的采用较长的编码，并且保持编码的唯一可解性。在计算机信息处理中，经常应用于数据压缩。是一种一致性编码法（又称"熵编码法"），用于数据的无损耗压缩。要求实现一个完整的哈夫曼编码与译码系统。

**二、实验要求及实验环境**

**实验要求**：

1. 从文件中读入任意一篇英文文本文件，分别统计英文文本文件中各字符（包括标点符号和空格）的使用频率；
2. 根据已统计的字符使用频率构造哈夫曼编码树，并给出每个字符的哈夫曼编码（字符集的哈夫曼编码表）；
3. 将文本文件利用哈夫曼树进行编码，存储成压缩文件（哈夫曼编码文件）；
4. 计算哈夫曼编码文件的压缩率；
5. 将哈夫曼编码文件译码为文本文件，并与原文件进行比较。

以下可以不做，供思考，做了可以适当加分

1. 能否利用堆结构，优化的哈夫曼编码算法。
2. 上述 1-5 的编码和译码是基于字符的压缩，考虑基于单词的压缩，完成上述工作，讨论并比较压缩效果。
3. 上述 1-5 的编码是二进制的编码，可以采用 K 叉的哈夫曼树完成上述工作，实现“K 进制”的编码和译码，并与二进制的编码和译码进行比较。

**实验环境：**Windows 10 + Intel i7-9750H + 16GB + MinGW-W64-builds-4.3.5 + VSCode

**三、设计思想**（本程序中的用到的所有数据类型的定义，主程序的流程图及各程序模块之间的调用关系）

1. 物理设计
2. 文章：包括文章长度，每个字符的频率。
3. 哈夫曼编码：包括字符，对应的编码。
4. 哈夫曼树节点：包括字符，频率总和，子节点集合，**支持K叉的哈夫曼树。**
5. 小根堆：包括堆数组，堆长度，**利用堆结构优化哈夫曼编码算法**。
6. 逻辑设计
   1. 主程序流程图



图 1主程序流程图

* 1. 程序模块之间调用关系



图 2程序模块之间调用关系

**四、测试结果**

(斜体为备注内容)

1. 输入数据

I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in the history of our nation.

Five score years ago, a great American, in whose symbolic shadow we stand today, signed the Emancipation Proclamation. This momentous decree came as a great beacon light of hope to millions of Negro slaves who had been seared in the flames of withering injustice. It came as a joyous daybreak to end the long night of bad captivity.

But one hundred years later, the Negro still is not free. One hundred years later, the life of the Negro is still sadly crippled by the manacles of segregation and the chains of discrimination. One hundred years later, the Negro lives on a lonely island of poverty in the midst of a vast ocean of material prosperity. One hundred years later, the Negro is still languished in the corners of American society and finds himself an exile in his own land. And so we've come here today to dramatize a shameful condition.

In a sense we've come to our nation's capital to cash a check. When the architects of our republic wrote the magnificent words of the Constitution and the Declaration of Independence, they were signing a promissory note to which every American was to fall heir. This note was a promise that all men, yes, black men as well as white men, would be guaranteed the "unalienable Rights" of "Life, Liberty and the pursuit of Happiness." It is obvious today that America has defaulted on this promissory note, insofar as her citizens of color are concerned. Instead of honoring this sacred obligation, America has given the Negro people a bad check, a check which has come back marked "insufficient funds."

But we refuse to believe that the bank of justice is bankrupt. We refuse to believe that there are insufficient funds in the great vaults of opportunity of this nation. And so, we've come to cash this check, a check that will give us upon demand the riches of freedom and the security of justice.

We have also come to this hallowed spot to remind America of the fierce urgency of Now. This is no time to engage in the luxury of cooling off or to take the tranquilizing drug of gradualism. Now is the time to make real the promises of democracy. Now is the time to rise from the dark and desolate valley of segregation to the sunlit path of racial justice. Now is the time to lift our nation from the quicksands of racial injustice to the solid rock of brotherhood. Now is the time to make justice a reality for all of God's children.

It would be fatal for the nation to overlook the urgency of the moment. This sweltering summer of the Negro's legitimate discontent will not pass until there is an invigorating autumn of freedom and equality. Nineteen sixty-three is not an end, but a beginning. And those who hope that the Negro needed to blow off steam and will now be content will have a rude awakening if the nation returns to business as usual. And there will be neither rest nor tranquility in America until the Negro is granted his citizenship rights. The whirlwinds of revolt will continue to shake the foundations of our nation until the bright day of justice emerges.

But there is something that I must say to my people, who stand on the warm threshold which leads into the palace of justice: In the process of gaining our rightful place, we must not be guilty of wrongful deeds. Let us not seek to satisfy our thirst for freedom by drinking from the cup of bitterness and hatred. We must forever conduct our struggle on the high plane of dignity and discipline. We must not allow our creative protest to degenerate into physical violence. Again and again, we must rise to the majestic heights of meeting physical force with soul force.

The marvelous new militancy which has engulfed the Negro community must not lead us to a distrust of all white people, for many of our white brothers, as evidenced by their presence here today, have come to realize that their destiny is tied up with our destiny. And they have come to realize that their freedom is inextricably bound to our freedom.

We cannot walk alone.

And as we walk, we must make the pledge that we shall always march ahead.

We cannot turn back.

There are those who are asking the devotees of civil rights, "When will you be satisfied?" We can never be satisfied as long as the Negro is the victim of the unspeakable horrors of police brutality. We can never be satisfied as long as our bodies, heavy with the fatigue of travel, cannot gain lodging in the motels of the highways and the hotels of the cities. We cannot be satisfied as long as the Negro's basic mobility is from a smaller ghetto to a larger one. We can never be satisfied as long as our children are stripped of their selfhood and robbed of their dignity by signs stating "for whites only." We cannot be satisfied as long as a Negro in Mississippi cannot vote and a Negro in New York believes he has nothing for which to vote. No, no, we are not satisfied, and we will not be satisfied until "justice rolls down like waters, and righteousness like a mighty stream."

I am not unmindful that some of you have come here out of great trials and tribulations. Some of you have come fresh from narrow jail cells. And some of you have come from areas where your quest -- quest for freedom left you battered by the storms of persecution and staggered by the winds of police brutality. You have been the veterans of creative suffering. Continue to work with the faith that unearned suffering is redemptive. Go back to Mississippi, go back to Alabama, go back to South Carolina, go back to Georgia, go back to Louisiana, go back to the slums and ghettos of our northern cities, knowing that somehow this situation can and will be changed.

Let us not wallow in the valley of despair, I say to you today, my friends.

And so even though we face the difficulties of today and tomorrow, I still have a dream. It is a dream deeply rooted in the American dream.

I have a dream that one day this nation will rise up and live out the true meaning of its creed: "We hold these truths to be self-evident, that all men are created equal."

I have a dream that one day on the red hills of Georgia, the sons of former slaves and the sons of former slave owners will be able to sit down together at the table of brotherhood.

I have a dream that one day even the state of Mississippi, a state sweltering with the heat of injustice, sweltering with the heat of oppression, will be transformed into an oasis of freedom and justice.

I have a dream that my four little children will one day live in a nation where they will not be judged by the color of their skin but by the content of their character.

I have a dream today!

I have a dream that one day, down in Alabama, with its vicious racists, with its governor having his lips dripping with the words of "interposition" and "nullification" -- one day right there in Alabama little black boys and black girls will be able to join hands with little white boys and white girls as sisters and brothers.

I have a dream today!

I have a dream that one day every valley shall be exalted, and every hill and mountain shall be made low, the rough places will be made plain, and the crooked places will be made straight; "and the glory of the Lord shall be revealed and all flesh shall see it together."

This is our hope, and this is the faith that I go back to the South with.

With this faith, we will be able to hew out of the mountain of despair a stone of hope. With this faith, we will be able to transform the jangling discords of our nation into a beautiful symphony of brotherhood. With this faith, we will be able to work together, to pray together, to struggle together, to go to jail together, to stand up for freedom together, knowing that we will be free one day.

And this will be the day -- this will be the day when all of God's children will be able to sing with new meaning:

My country 'tis of thee, sweet land of liberty, of thee I sing.

Land where my fathers died, land of the Pilgrim's pride,

From every mountainside, let freedom ring!

And if America is to be a great nation, this must become true.

And so let freedom ring from the prodigious hilltops of New Hampshire.

Let freedom ring from the mighty mountains of New York.

Let freedom ring from the heightening Alleghenies of

Pennsylvania.

Let freedom ring from the snow-capped Rockies of Colorado.

Let freedom ring from the curvaceous slopes of California.

But not only that:

Let freedom ring from Stone Mountain of Georgia.

Let freedom ring from Lookout Mountain of Tennessee.

Let freedom ring from every hill and molehill of Mississippi.

From every mountainside, let freedom ring.

And when this happens, when we allow freedom ring, when we let it ring from every village and every hamlet, from every state and every city, we will be able to speed up that day when all of God's children, black men and white men, Jews and Gentiles, Protestants and Catholics, will be able to join hands and sing in the words of the old Negro spiritual:

Free at last! Free at last!

Thank God Almighty, we are free at last!

1. 输出数据

（由于输出数据太长，在此只有部分数据，若想查看全部，请查看同目录文件” Expriment2\_Out.txt”）

'\n' : 0.00482827

' ' : 0.177878

'!' : 0.0006584

'"' : 0.00219467

''' : 0.00120707

',' : 0.00844947

'-' : 0.0009876

'.' : 0.00823

':' : 0.000548667

';' : 0.000109733

'?' : 0.000109733

'A' : 0.0029628

'B' : 0.000438933

'C' : 0.0006584

'D' : 0.000109733

'E' : 0.000109733

'F' : 0.000548667

'G' : 0.0009876

………

'r' : 0000

'\n' : 0001000

'C' : 0001001000

'O' : 00010010010

'E' : 000100100110

'R' : 000100100111

'!' : 0001001010

'z' : 0001001011

'N' : 00010011

'k' : 0001010

'A' : 00010110

'q' : 0001011100

'M' : 0001011101

'L' : 000101111

'f' : 00011

's' : 0010

'n' : 0011

'e' : 010

'b' : 011000

'y' : 011001

'd' : 01101

……..

11001001111101111101011111101101111100101110010101100111110111001111110010001100110000011111101001100010111101111101100110011100111111011100101101011101100111110000011111101001110110111101111110100110001100011000111101011100111101101100110100100111111000001111111011100000101011100100000110011110111001011110111101101011110101100000100111101101000101011111011010101101011001001100101011000001111011100010010011111000111001000011100011000001001001101100111010111110000011111101111011010111110111000001010111001000001100111110010001111110011100110000111001101111011100010010011101000100010001100100100110001010101010111001011010010010000010111011001010011100000010111011110101110011010100111011111110101100000100111101111100010110110101010000010001101000111001110101001111000001111110100111011100100100101110010011001110101011000100111000100011010011100101101101110110110011010011111010010101110010101101110011011011111011100101101011101100110101001110010100010101100110100110111110111……..

I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in the history of our nation.

Five score years ago, a great American, in whose symbolic shadow we stand today, signed the Emancipation Proclamation. This momentous decree came as a great beacon light of hope to millions of Negro slaves who had been seared in the flames of withering injustice. It came as a joyous daybreak to end the long night of bad captivity.

But one hundred years later, the Negro still is not free. One hundred years later, the life of the Negro is still sadly crippled by the manacles of segregation and the chains of discrimination. One hu……

Compression Ratio:

54.849%

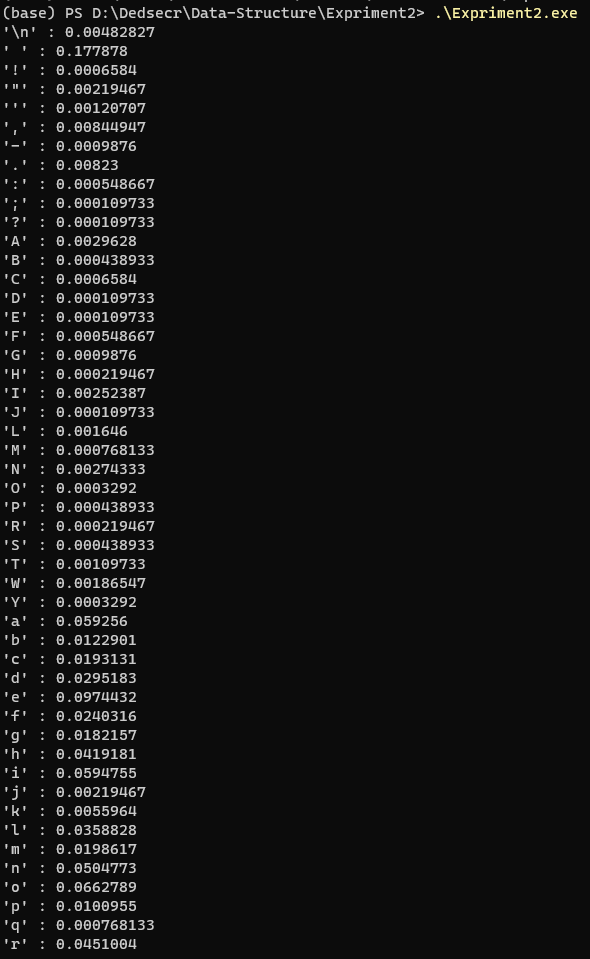


图 3运行结果截图

**五、经验体会与不足**

1. 不足
2. 输入格式对用户较不友好，可改进。
3. 未考虑基于单词的压缩。
4. 内存空间管理能力较弱。
5. 经验体会
   1. 当哈夫曼树叉数在2和3之间时数据压缩率最好。
   2. 要合理设计以尽量实现程序模块复用。

**六、附录：源代码（带注释）**

1. #include <iostream>
2. #include <cstdio>
3. #include <algorithm>
4. #include <string>
5. #include <cstring>
6. #include <stack>
7. #include <queue>
8. #include <map>
9. #include <vector>
10. #include <cmath>
11. **using** **namespace** std;
12. **const** **int** MAXN = 50;
13. string OriginalText;
14. string EncodedText;
15. **struct** Text
16. {
17. **int** Length;
18. **double** Frequency[128];
19. Text()
20. {
21. Length = 0;
22. **for** (**int** i = 0; i < 128; ++i)
23. Frequency[i] = 0;
24. }
25. };
26. **struct** HCode
27. {
28. string Code;
29. **char** Text;
30. };
31. **struct** HuffmanCode
32. {
33. vector<HCode> Code;
34. };
36. map<**char**, string>HFCMap;
37. map<string, **char**>HFCMap\_Reverse;
39. **int** Stack[MAXN], StackP;
40. **int** ChildNum = 2;
41. **struct** TreeNode
42. {
43. **char** Text;
44. **double** Val;
45. vector<TreeNode \*> Ch;
46. TreeNode()
47. {
48. Text = 0;
49. Val = 0;
50. Ch.clear();
51. }
52. };
54. **typedef** TreeNode \*Tree;
55. **typedef** TreeNode \*Node;
56. //小根堆
57. **class** Priority\_Queue
58. {
59. **private**:
60. Tree Heap[MAXN << 2];
61. **int** Length;
63. **public**:
64. Priority\_Queue()
65. {
66. Length = 0;
67. }
68. **bool** empty()
69. {
70. **return** !Length;
71. }
72. **int** size()
73. {
74. **return** Length;
75. }
76. **void** push(Tree x)
77. {
78. Heap[++Length] = x;
79. **int** Pos = Length;
80. **while**(Pos > 1)
81. {
82. **if**(Heap[Pos >> 1]->Val > Heap[Pos]->Val)
83. swap(Heap[Pos >> 1], Heap[Pos]);
84. Pos >>= 1;
85. }
86. }
87. Tree top()
88. {
89. **return** Heap[1];
90. }
91. **void** pop()
92. {
93. **if**(!Length)
94. {
95. cerr << "Error in" << \_\_LINE\_\_;
96. exit(-1);
97. }
98. swap(Heap[1], Heap[Length]);
99. Length--;
100. **if**(Length == 0)
101. **return**;
102. **int** Pos = 1;
103. **while**(Length >= (Pos << 1))
104. {
105. **if**((Pos << 1 | 1) <= Length)
106. {
107. **if**(Heap[Pos]->Val > min(Heap[Pos << 1]->Val, Heap[Pos << 1 | 1]->Val))
108. {
109. **if**(Heap[Pos << 1]->Val < Heap[Pos << 1 | 1]->Val)
110. swap(Heap[Pos << 1], Heap[Pos]), Pos <<= 1;
111. **else** swap(Heap[Pos << 1 | 1], Heap[Pos]), Pos = Pos << 1 | 1;
112. }
113. **else**
114. **break**;
115. }
116. **else**
117. {
118. **if**(Heap[Pos]->Val > Heap[Pos << 1]->Val)
119. swap(Heap[Pos << 1], Heap[Pos]), Pos <<= 1;
120. **else**
121. **break**;
122. }
123. }
124. }
125. };
126. Priority\_Queue Q;
127. //建立哈夫曼树
128. Tree BuildHuffmanTree(Text &text)
129. {
130. **while** (!Q.empty())
131. Q.pop();
132. **for** (**int** i = 0; i < 128; ++i)
133. **if** (abs(text.Frequency[i]) > 1e-8)
134. {
135. Tree T = **new** TreeNode;
136. T->Val = text.Frequency[i];
137. T->Text = i;
138. Q.push(T);
139. }
140. **while** (Q.size() != 1)
141. {
142. **int** End = Q.size();
143. **if** (End > ChildNum)
144. End = ChildNum;
145. Tree T = **new** TreeNode;
146. **for** (**int** i = 1; i <= End; ++i)
147. {
148. Tree now = Q.top();
149. Q.pop();
150. T->Val += now->Val;
151. T->Ch.push\_back(now);
152. }
153. Q.push(T);
154. }
155. **return** Q.top();
156. }
157. //输入文本并计算频率
158. Text GetTextandFrequency()
159. {
160. Text Res;
161. **char** now;
162. OriginalText = "";
163. **while** ((now = getchar()) != EOF)
164. {
165. OriginalText += now;
166. }
167. **int** Length = OriginalText.length();
168. Res.Length = Length;
169. **for** (**int** i = 0; i < Length; ++i)
170. {
171. Res.Frequency[OriginalText[i]] += 1;
172. }
173. **for** (**int** i = 0; i < 128; ++i)
174. {
175. Res.Frequency[i] /= Length;
176. }
177. **return** Res;
178. }
179. **char** HCInt2Char(**int** x)
180. {
181. **if**(x < 10)
182. **return** x + '0';
183. **return** x - 10 + 'A';
184. }
185. string CharForOutput(**char** x)
186. {
187. string Res = "";
188. **if**(x == '\n')
189. Res = "\'\\n\'";
190. **else**
191. {
192. Res = "\'";
193. Res += x;
194. Res += "\'";
195. }
196. **return** Res;
197. }
198. //遍历哈夫曼树以计算哈夫曼编码
199. **void** GetHC(Tree Root, HuffmanCode \*Code)
200. {
201. **int** Size = Root->Ch.size();
202. **if** (!Size)
203. {
204. **if** (!Root->Text)
205. {
206. cerr << "Error in " << \_\_LINE\_\_;
207. exit(-1);
208. }
209. HCode HC;
210. HC.Text = Root->Text;
211. HC.Code = "";
212. **for** (**int** i = 1; i <= StackP; ++i)
213. HC.Code += HCInt2Char(Stack[i]);
214. Code->Code.push\_back(HC);
215. **return**;
216. }
217. **for** (**int** i = 0; i < Size; ++i)
218. {
219. Stack[++StackP] = i;
220. GetHC(Root->Ch[i], Code);
221. StackP--;
222. }
223. }
224. //打印哈夫曼编码
225. **void** PrintHC(HuffmanCode \*Code)
226. {
227. **int** Size = Code->Code.size();
228. **for** (**int** i = 0; i < Size; ++i)
229. {
230. cout << CharForOutput(Code->Code[i].Text) << " : " << Code->Code[i].Code << endl;
231. }
232. }
233. //输出频率
234. **void** PrintFrequency(Text &text)
235. {
236. **for** (**int** i = 0; i < 128; ++i)
237. {
238. **if** (abs(text.Frequency[i]) > 1e-8)
239. {
240. cout << CharForOutput(**char**(i)) << " : " << text.Frequency[i] << endl;
241. }
242. }
243. puts("");
244. }
246. **void** GetMap(HuffmanCode \*Code)
247. {
248. **int** Size = Code->Code.size();
249. **for** (**int** i = 0; i < Size; ++i)
250. {
251. HFCMap[Code->Code[i].Text] = Code->Code[i].Code;
252. HFCMap\_Reverse[Code->Code[i].Code] = Code->Code[i].Text;
253. }
254. }
255. //计算每个字符的哈夫曼编码并输出
256. **void** GetHuffmanCode(Tree Root)
257. {
258. HuffmanCode \*Code = **new** HuffmanCode;
259. StackP = 0;
260. GetHC(Root, Code);
261. GetMap(Code);
262. PrintHC(Code);
263. }
264. //编码并输出编码后文本
265. **int** Encode()
266. {
267. EncodedText = "";
268. **int** Res = 0;
269. **int** Length = OriginalText.length();
270. **for**(**int** i = 0; i < Length; ++i)
271. {
272. Res += HFCMap[OriginalText[i]].length();
273. EncodedText += HFCMap[OriginalText[i]];
274. }
275. cout << EncodedText << '\n';
276. **return** Res;
277. }
278. //解码并输出解码后文本
279. **void** Decode()
280. {
281. string now = "";
282. **int** Length = EncodedText.length();
283. **for**(**int** i = 0; i < Length; ++i)
284. {
285. now += EncodedText[i];
286. **if**(HFCMap\_Reverse[now])
287. {
288. cout << HFCMap\_Reverse[now];
289. now = "";
290. }
291. }
292. puts("");
293. }
294. //求压缩率
295. **void** GetCompressionRatio(**int** Length\_After)
296. {
297. **int** Width\_Before = 8, Width\_After = ceil(log2(1.0 \* ChildNum));
298. **double** Ratio = 1.0 \* Width\_After \* Length\_After / Width\_Before \* 100 / OriginalText.length();
299. printf("Compression Ratio:\n%.3lf%%\n", Ratio);
300. }
302. **int** main()
303. {
304. //freopen("Expriment2\_In.txt", "r", stdin);
305. //freopen("Expriment2\_Out.txt", "w", stdout);
307. ChildNum = 2;
309. Text text = GetTextandFrequency();
310. PrintFrequency(text);
311. puts("");
313. Tree T = BuildHuffmanTree(text);
314. GetHuffmanCode(T);
315. puts("");
317. **int** Length\_After = Encode();
318. puts("");
320. Decode();
321. puts("");
323. GetCompressionRatio(Length\_After);
324. **return** 0;
325. }