**Sıkıştırma Projesi**

Arda Talu

Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü

170202037 [arda.starkiler@gmail.com](mailto:arda.starkiler@gmail.com)

**Özet**

Projenin amacı kullanıcı tarafından verilmiş olan bir oku.txt dosyasını okuyarak gerekli bilgileri alıp algoritmalar yardımıyla istenilen metnin sıkıştırılmasını sağlamaktır.

**1.Giriş**

Sıkıştırma Projesinde amaç, kullanıcıdan alınan metin dosyasının çok daha az yer kaplayacak şekilde sıkıştırma yönteminin bulunmasıdır.

Bu projede karakter tabanlı LZ77, DEFLATE sıkıştırma algoritmaları kullanılarak dosya sıkıştırma uygulaması yapmanız istenmektedir.

**2.Algoritma**

Program başlatıldığında oku.txt dosyası okunur.

Okunan oku.txt dosyası algoritmalar tarafından okunur, işlemden geçirilir.

Sıkıştırılıp encode edilmiş dosya encode.txt dosyasına, decode edilmiş dosya da decode.txt dosyasına yazdırılır.

**3.Sözde Kod**

1. Başla
2. stdio.h kütüphanesini import et
3. stdlib.h kütüphanesini import et
4. stdint.h kütüphanesini import et
5. string.h kütüphanesini import et
6. Tanımla MAX\_TREE\_HT 50
7. Tanımla OFFSETBITS 5
8. Tanımla LENGTHBITS (8-OFFSETBITS)
9. Tanımla OFFSETMASK ((1 << (OFFSETBITS)) - 1)
10. Tanımla LENGTHMASK ((1 << (LENGTHBITS)) - 1)
11. Tanımla GETOFFSET(x) (x >> LENGTHBITS)
12. Tanımla GETLENGTH(x) (x & LENGTHMASK)
13. Tanımla OFFSETLENGTH(x,y) (x << LENGTHBITS | y)
14. Struct token
15. uint8\_t offset\_len
16. char c
17. int prefix\_match\_length(char \*s1, char \*s2, int limit)
18. int l = 0
19. İken (\*s1++ == \*s2++ && l < limit)
20. l++
21. return l
22. void lz77memcpy(char \*s1, char \*s2, int size)
23. İken (size--)
24. \*s1++ = \*s2++
25. char \*decode(struct token \*tokens, int numTokens, int \*pcbDecoded)
26. int cap = 1 << 3
27. \*pcbDecoded = 0
28. char \*decoded = malloc(cap)
29. int i
30. İçin (i = 0; i < numTokens; i++)
31. int offset = GETOFFSET(tokens[i].offset\_len)
32. int l = GETLENGTH(tokens[i].offset\_len)
33. char c = tokens[i].c
34. Eğer (\*pcbDecoded + l + 1 > cap)
35. cap = cap << 1
36. decoded = realloc(decoded, cap)
37. Eğer (l > 0)
38. lz77memcpy(&decoded[\*pcbDecoded], &decoded[\*pcbDecoded - offset], l)
39. \*pcbDecoded += l
40. decoded[\*pcbDecoded] = c
41. \*pcbDecoded = \*pcbDecoded + 1
42. return decoded
43. struct token \*encode(char \*text, int limit, int \*numTokens)
44. int cap = 1 << 3
45. int \_numTokens = 0
46. struct token t
47. char \*lookahead, \*search
48. struct token \*encoded = malloc(cap \* sizeof(struct token))
49. İçin (lookahead = text; lookahead < text + limit; lookahead++)
50. search = lookahead - OFFSETMASK
51. Eğer (search < text)
52. search = text
53. int max\_len = 0
54. char \*max\_match = lookahead
55. İçin (; search < lookahead; search++)
56. int l = prefix\_match\_length(search, lookahead, LENGTHMASK)
57. Eğer (l > max\_len)
58. max\_len = l
59. max\_match = search
60. Eğer (lookahead + max\_len >= text + limit)
61. max\_len = text + limit - lookahead - 1
62. t.offset\_len = OFFSETLENGTH(lookahead - max\_match, max\_len)
63. lookahead += max\_len
64. t.c = \*lookahead
65. Eğer (\_numTokens + 1 > cap)
66. cap = cap << 1
67. encoded = realloc(encoded, cap \* sizeof(struct token))
68. encoded[\_numTokens++] = t
69. Eğer (numTokens)
70. \*numTokens = \_numTokens
71. return encoded
72. char \*file\_read(FILE \*f1, int \*size)
73. char \*content
74. fseek(f1, 0, SEEK\_END)
75. \*size = ftell(f1)
76. content = malloc(\*size)
77. fseek(f1, 0, SEEK\_SET)
78. fread(content, 1, \*size, f1)
79. return contents
80. int main(void)
81. FILE \*f1
82. int metin\_boyutu = 8, token\_sayisi, decode\_boyutu
83. char \*kaynak = "aaaaaaaa", \*decoded\_metin = ""
84. struct token \*encoded\_metin
85. Eğer (f1 = fopen("oku.txt", "rb"))
86. kaynak = file\_read(f1, &metin\_boyutu)
87. fclose(f1)
88. encoded\_metin = encode(kaynak, metin\_boyutu, &token\_sayisi)
89. Eğer (f1 = fopen("encoded.txt", "wb"))
90. fwrite(encoded\_metin, sizeof(struct token), token\_sayisi, f1)
91. fclose(f1)
92. decoded\_metin = decode(encoded\_metin, token\_sayisi, &decode\_boyutu)
93. Eğer (f1= fopen("decoded.txt", "wb"))
94. fwrite(decoded\_metin, 1, decode\_boyutu, f1)
95. fclose(f1)
96. Yaz("Original Size: %d, Encode Size: %d, Decode Size: %d", metin\_boyutu, token\_sayisi \* sizeof(struct token), decode\_boyutu)
97. return 0
98. Bitiş

**4.Sonuç**

Kullanıcıya istediği metnin sıkıştırılmış bir versiyonunu LZ77, Deflate ve Huffman Algoritmaları yardımıyla sunan bir program yazılmıştır.

**5. Karşılaşılan Problemler**

Kullanılması istenen algoritmaların bir miktar iç içe olması kodlamada zorluklar çıkartmıştır. LZ77, Deflate ve Huffman ağacı için gerekli algoritmalar birbirine yakınlıklar ve kopukluklar barındırıyor. Aynı anda çalıştıramadım en iyi sonucu LZ77 algoritmasında alabildiğimi düşünüyorum.

**6.Kaynakça**

[1]<https://en.wikipedia.org/wiki/LZ77_and_LZ78>

[2]<https://ysar.net/algoritma/lz77.html>

[3]<https://tr.qwe.wiki/wiki/DEFLATE>

[4]<https://github.com/madler/zlib/blob/master/deflate.c>

[5]<https://www.programiz.com/dsa/huffman-coding>

[6]<https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/>

[7]<https://tools.ietf.org/html/rfc1951>

**7. Ekran Görüntüleri**











