CS310: Homework 6

Scott Fenton

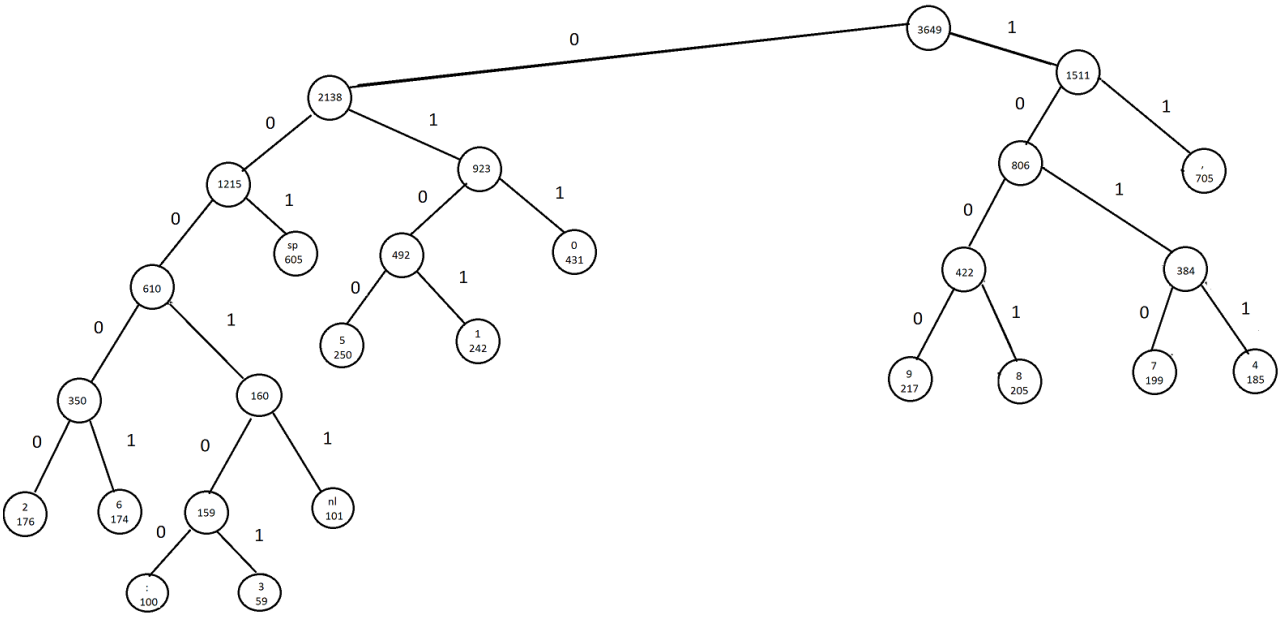
April 13, 2017

Exercise (1). The following table has the frequencies of symbols in a file, where nl and sp stand for newline and space, respectively.

Symbols	:	sp	nl	,	0	1	2	3	4	5	6	7	8	9
Frequencies	100	605	101	705	431	242	176	59	185	250	174	199	205	217

Exercise (1A). Applying Huffmans algorithm to the data, draw the prefix code tree. The leaves should be annotated with the symbols and their weights, and internal nodes should be annotated with the total weights of their substrees. When merging two subtrees, put the lower-weight subtree on the right.

Solution (1A). Below is the Huffman Tree



Exercise (1B). Make a table of four columns: symbols, codes, frequencies, total bits, like Slide 20 of lecture notes.

Solution (1B). Below is the table

Char	Frequency	Code	Total bits
,	705	11	1410
sp	605	001	1815
0	431	011	1293
5	250	0100	1000
1	242	0101	968
9	217	1000	868
8	205	1001	820
7	199	1010	796
4	185	1011	740
2	176	00000	880
6	174	00001	870
nl	101	00011	505
:	100	000100	600
3	59	000101	354

Exercise (1C). Compress this 7-symbol text: '04: 19,'. Show the result in both binary and hexadecimal.

Solution (1C). The 7-symbol text converted to binary and hex.

Binary: 01110110001000010101100011
Hexadecimal: 1D88563

Exercise (1D). Decompress the binary string 11001110111100000.

Solution (1D). The decompressed string is COMMA SPACE COMMA ZERO COMMA TWO.

String with quotes: ', ,0,2'

Exercise (1E). In preparation for fast decoding, sort the codes by lexicographical order like Slide 27 of lecture notes.

Solution (1E). Char and Code in lexicographical order

Char	Code
2	00000
6	00001
:	000100
3	000101
nl	00011
sp	001
5	0100
1	0101
0	011
9	1000
8	1001
7	1010
4	1011
,	11

Exercise (1F). Take the top three codes after sorting, and construct the decode array for these three codes, using . . . to skip repetitive symbols in the decode array, like Slide 28 of lecture notes.

Solution (1F). Char and bit pattern

Char	Bit Pattern
2	00000000
2	...
2	00000111
6	00001000
6	...
6	00001111
:	00010000
:	...
:	00010011

Exercise (2A). Apply the Burrows-Wheeler Transform on this string: abracadabra, using the methods on Slides 47 and 48 of lecture notes. Generate all rotations of the string.

Solution (2A). All Rotations

All rotations
abracadabra
aabracadabr
raabracadab
braabracada
abraabracad
dabraabraca
adabraabrac
cadabraabra
acadabraabr
racadabraab
bracadabraa

Exercise (2B). Reorder the rotations by lexicographical order.

Solution (2B). Lexicographical order

Lexographical order
aabracadabr
abraabracad
abracadabra
acadabraabr
adabraabrac
braabracada
bracadabraa
cadabraabra
dabraabraca
raabracadab
racadabraab

Exercise (2C). Output the last column and the index I of the original string. Note that I is zero-based. See Slide 47.

Solution (2C). The Index I of the original string is 2.

Last Column
r
d
a
r
c
a
a
a
a
b
b

Exercise (2D). Write the contents of the three arrays F, L, and T. See Slide 48.

Solution (2D). Contents of FLT:

First Column	Last Column	Transform
a	r	9
a	d	8
a	a	0
a	r	10
a	c	7
b	a	1
b	a	2
c	a	3
d	a	4
r	b	5
r	b	6