Huffman codes

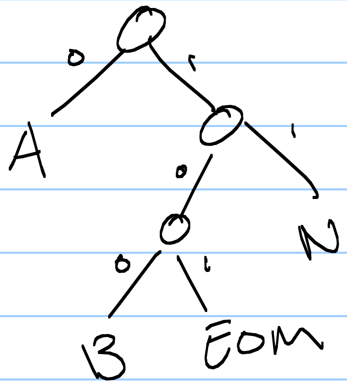
Note Title 12/1/2009

Announcements

- Program due ~~tomorrow~~ Saturday
- Next program up - take a look over break
- Take out paper & pen
- Final is Monday the 13th

Transmitting information

prefix free



Is this a good way to transmit info?

1000110110101 ← 15 bits + tree

B A N A N A EOM

ASCII: $8 \times 6 = 48$

Idea

We wish to transmit information using as few bits as possible.

Standard ASCII → 8 bits per character

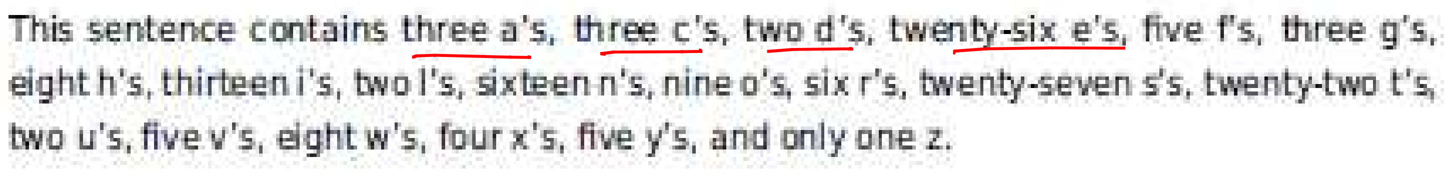
How can we do better?

- Can figure out which characters we actually need.
- More common letters get shorter bitstrings.

S, T, M
short

Q, Z
long

First - what do we need?



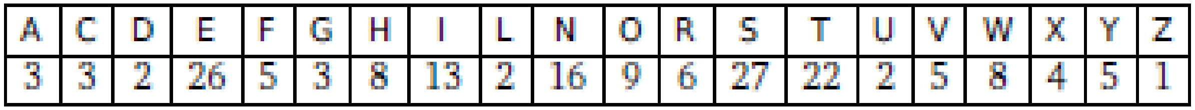
This sentence contains three a's, three c's, two d's, twenty-six e's, five f's, three g's, eight h's, thirteen i's, two l's, sixteen n's, nine o's, six r's, twenty-seven s's, twenty-two t's, two u's, five v's, eight w's, four x's, five y's, and only one z.

Letters:

A, C, D, E, F, G, H, I, L, N, O, R, S, T, U, V, W, X, Y, Z

(ignoring punctuation, spaces...)

Frequency Analysis



A	C	D	E	F	G	H	I	L	N	O	R	S	T	U	V	W	X	Y	Z
3	3	2	26	5	3	8	13	2	16	9	6	27	22	2	5	8	4	5	1

Which ones do we want to use few bits for?

Which ones can use lots of bits?

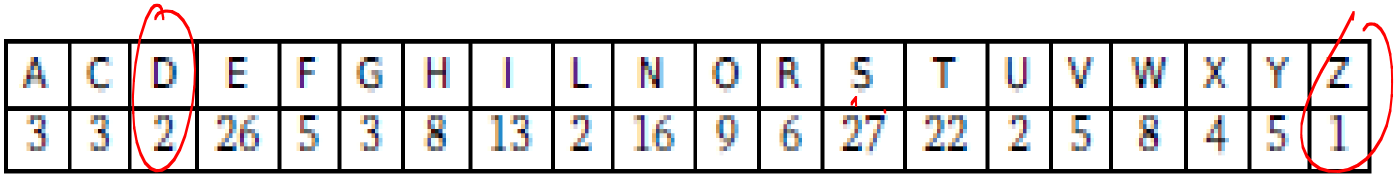
Z, D, A, C, G

build bottom

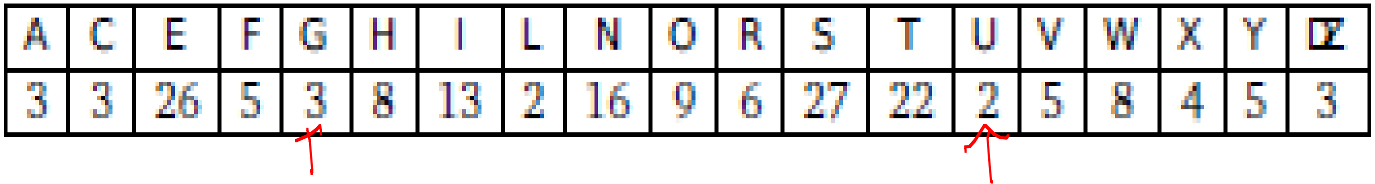
Huffman's algorithm:

Make two least frequent characters

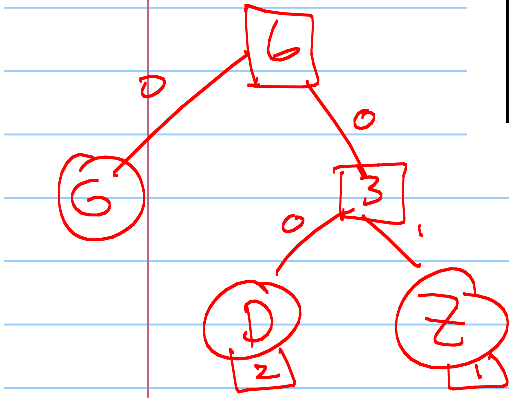
↳ merge them into character + recurse

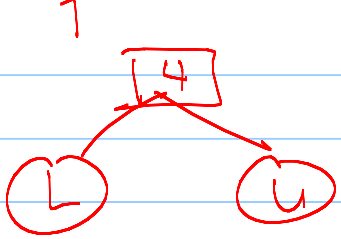


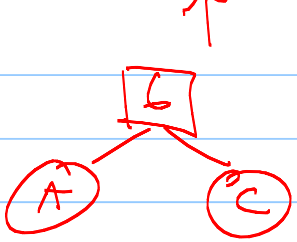
A	C	D	E	F	G	H	I	L	N	O	R	S	T	U	V	W	X	Y	Z
3	3	2	26	5	3	8	13	2	16	9	6	27	22	2	5	8	4	5	1



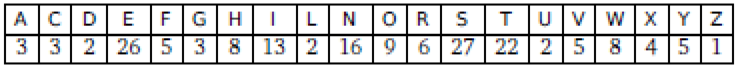
A	C	E	F	G	H	I	L	N	O	R	S	T	U	V	W	X	Y	DZ
3	3	26	5	3	8	13	2	16	9	6	27	22	2	5	8	4	5	3





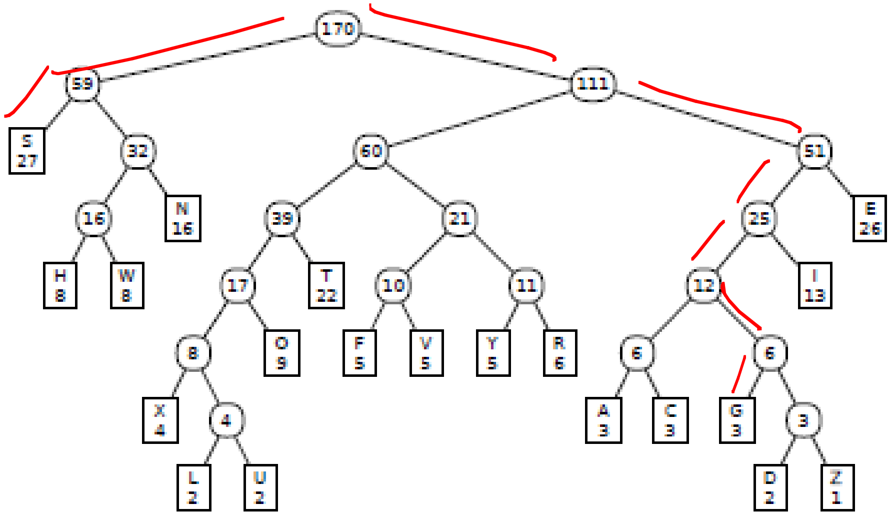


In the end, this:



A	C	D	E	F	G	H	I	L	N	O	R	S	T	U	V	W	X	Y	Z
3	3	2	26	5	3	8	13	2	16	9	6	27	22	2	5	8	4	5	1

turns into a decoder tree (like in program)

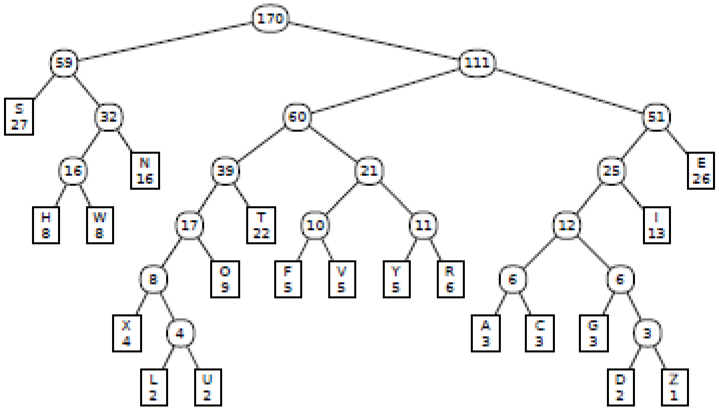


to send an S,
transmit 00

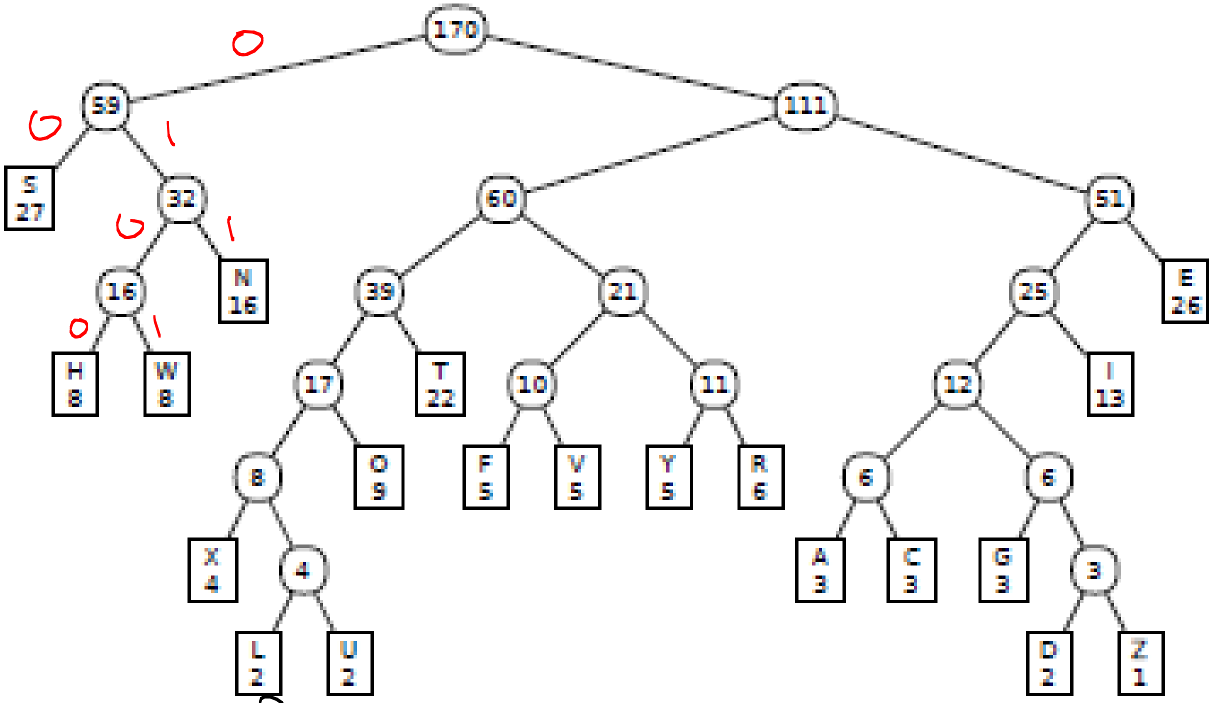
to send G: 110010

Original message:

1001	0100	1101	00	00	111	011	1001	111	011	110001	111	110001	10001	011	1001	110000	1101	...
T	H	I	S	S	E	N	T	E	N	C	E	C	O	N	T	A	I	



Exercise: 01001111000010100001010001



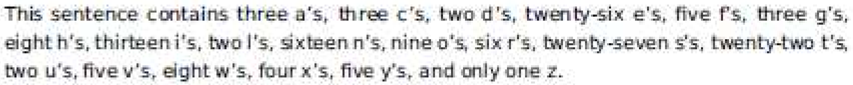
Message? HELLO

How many bits? 26

ASCII? 40

Why do all this again?

170 letters



This sentence contains three a's, three c's, two d's, twenty-six e's, five f's, three g's, eight h's, thirteen i's, two l's, sixteen n's, nine o's, six r's, twenty-seven s's, twenty-two t's, two u's, five v's, eight w's, four x's, five y's, and only one z.

How does ASCII do?

170 x 8 = 1360

Using our tree:

642 bits

(to send tree: 20 char, 8 bits each, 80 bits for tree)

Good to know:

Huffman trees are optimal.
They use fewest possible
bits for any message.
(greedy)