Chapter 3

Huffman Coding

Huffman code is a particular type of optimal prefix code that is commonly used for lossless data compression. The process of finding or using such a code proceeds by means of Huffman coding.

ASCII code ➔ fixed length code, it take a lot of space.

A B C D E F

- 6 characters ➔ each character need 8 bits

  1000 character ➔ 8000 bits
- But we need 3 bit to represent 6 character

  1000 character ➔ 3000 bit
- Some character occurs more frequency than the other.

If we have file with 100,000 character

	A	B	C	D	E	F	
Frequency	45	13	12	16	9	5	freq/1000

Type	A	B	C	D	E	F	Size
ASCII	1000001	1000010	1000011	1000100	1000101	1000110	800,000 bit = 100,000 byte = **100 Kbyte**
Fixed Length	000	001	010	011	100	101	300,000 bit = 37,500 byte = **36.6 Kbyte**
Variable Length	0	101	100	111	1101	1100	1 * 45,000 + 3 * 13,000 + 3 * 12,000 + 3 * 16,000 + 4 * 9000 + 4 * 5000 = 244,000 bit = 28,000 byte = **27.3 Kbyte**

Example:

A C D F B

01001111100101

0|100|1111|100|101

A C D F B

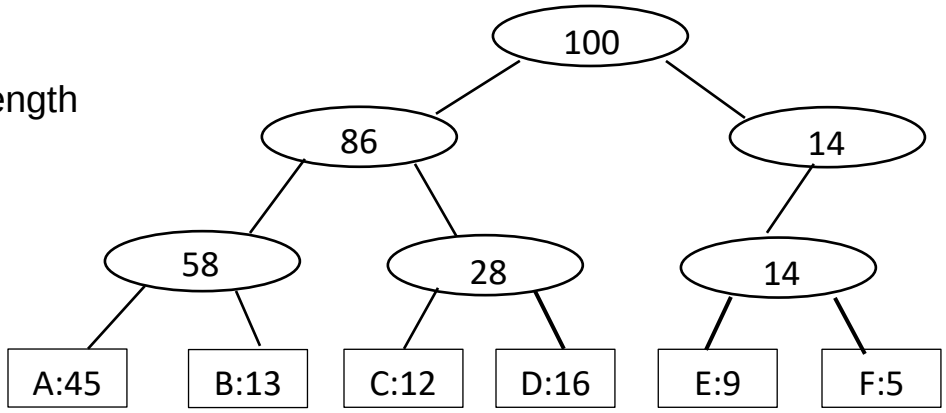
Leaves➔ given character set

To code a character trace a path from root to leaves

Left child ➔ 0

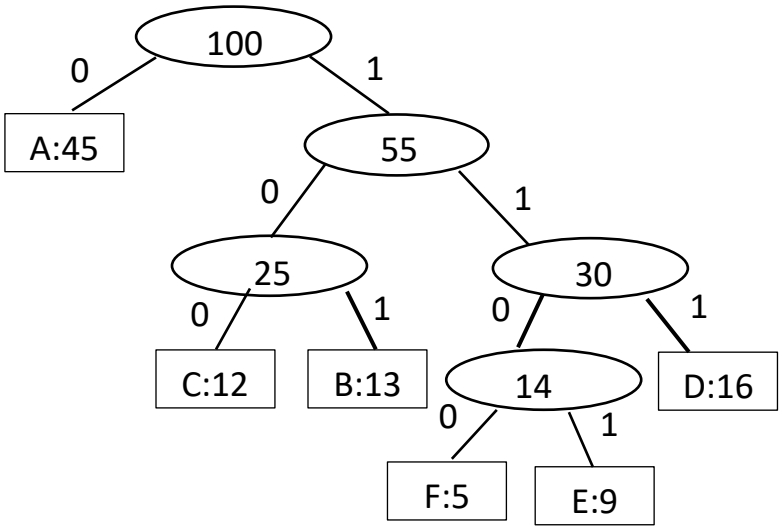
Right child ➔ 1

Fixed Length

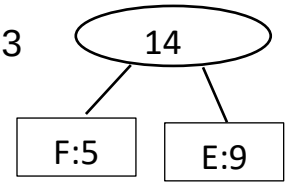


Not necessary to be full tree

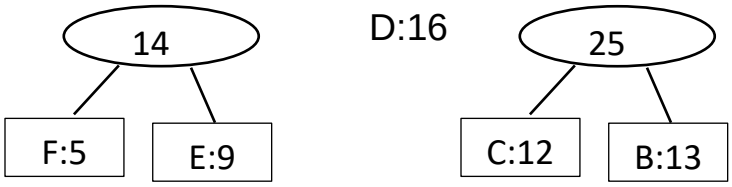
Variable Length

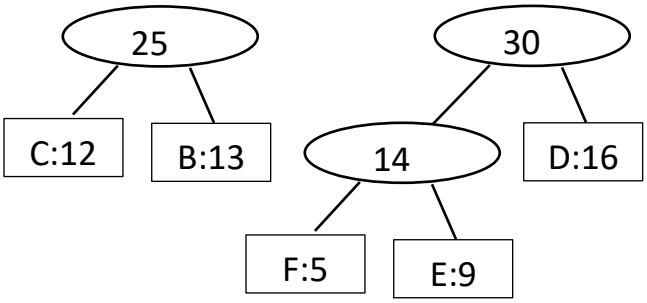


F: 5 E:9 C:12 B:13 D:16 A:45

C:12 B:13 

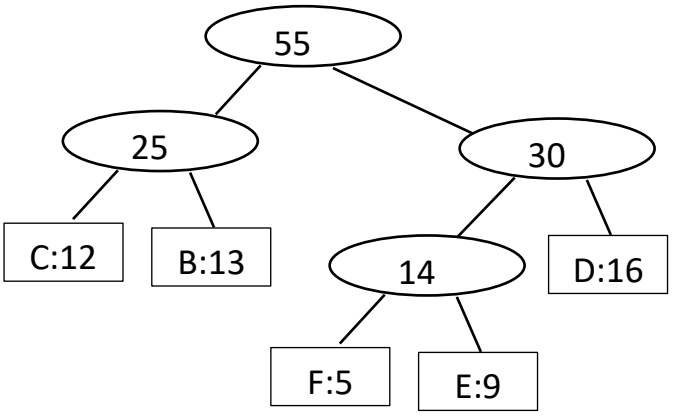
D:16 A:45

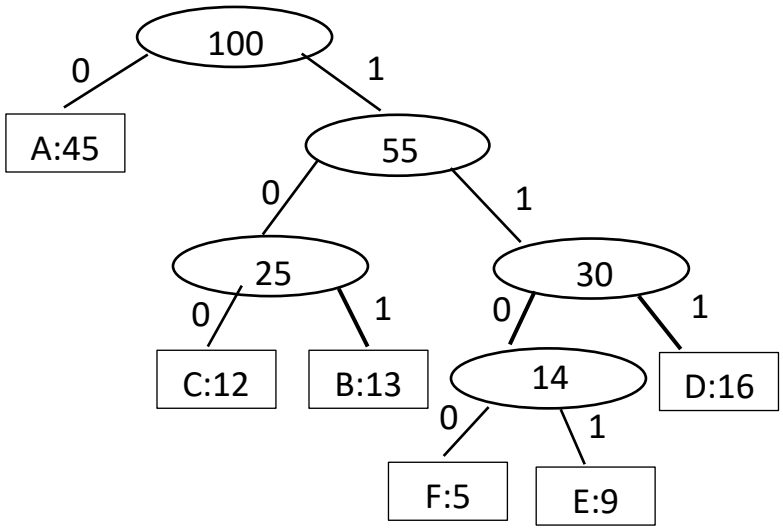




A:45

A:45





Algorithm

n = number of leaves

Q → Priority Queue

Insert (Q, Character)

for (i = 1; i < n; i++)

z = allocate new node;

x = deleteMin(Q) ;

y = deleteMin(Q);

z.leftChild ← x

z.rightChild ← y

freq(z) = freq(x) + freq(y);

insert (Q, z)

end for