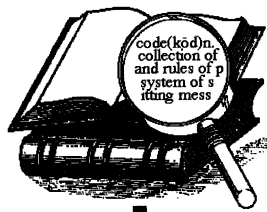


File Compression and Huffman Codes

CLRS Reading: Section 16.3, pages 385 -- 392
 Problem Set: Assignment #8 due Friday, April 11

Q - 1

Binary Code



Coded in (decimal) ASCII

```

99 111 100 101 40 107 111 100 41 110 46
99 111 100 100 101 99 116 105 111 110 32 111 102
97 110 100 32 114 117 108 101 115 32 111 102 112
115 121 115 116 101 109 32 111 102 32 115
105 116 116 105 110 103 32 109 101 115 115
  
```

Coded in (binary) ASCII

```

1100011 1101111 1100100 1100101 0101000 1101011 1101111 1100100 0101001 1101110 0101110
1100011 1101111 1101100 1101100 1100101 1100011 1101100 1101001 1101111 1101110 0100000 1101111 1100110
1100001 1101110 1100100 0100000 1110010 1110101 1101100 1100101 1110011 0100000 1101111 1100110 0100000 1110000
1110011 1111001 1110011 1110100 1100101 1101101 0100000 1101111 1100110 0100000 1110011
1101001 1110100 1110100 1101001 1101110 1100111 0100000 1101101 1100101 1110011 1110011
  
```

Q - 2



Improving on ASCII

- ASCII uses eight bits in order to store 2^8 distinct characters. If our file uses fewer characters, we might do better with a different code.
- For example, rather than store "ABRACADABRA" in eight bit ASCII code, we store it in our own personal five bit binary code (A = 00001, B = 00010, ...)*.

0000100010100100000100011000010010000001000101001000001

*This represents a 37.5% savings over the ASCII encoding. But we can do better if we give up the notion of fixed encoding length.

Q - 3



Variable-Length Encoding

- Assign shortest bit strings to the most commonly used letters: A=0, B=1, R=01, C=10, D=11. So ABRACADADBRA becomes

0 1 01 0 10 0 11 0 1 01 0

- This uses only 15 bits compared to 55 in our previous scheme. But it does require blanks to separate the characters.

Q - 4



Epiphany

- Without the spaces, the encoding of ABRACADABRA is ambiguous:

010101001101010

For example, is the first bit an A or the start of the pair 01 representing an R*?

- But, delimiters aren't needed if no character code is the prefix of another.

*Recall, A=0, B=1, R=01, C=10, D=11.

Q - 5



Prefix Codes

- An improved code for the message ABRACADABRA is given by

A	11
B	00
C	010
D	10
R	011

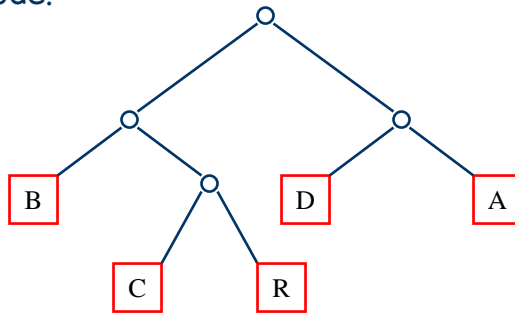
- The message ABRACADABRA is translated as the 25-bit string:

11000111110101110110001111

Q - 6

Coding Trees

- Decoding needs a convenient representation for the prefix code.



- For example, 1100011110101110110001111.

*Recall, A=11, B=00, R=011, C=010, D=10.

Q - 7

Cost of Tree T

Definition

Let T be the code tree corresponding to a prefix code. Define

$$B(T) = \sum_{c \in C} f(c) d_T(c)$$

where $f(c)$ is the frequency of character c and $d_T(c)$ denotes c 's depth in T .

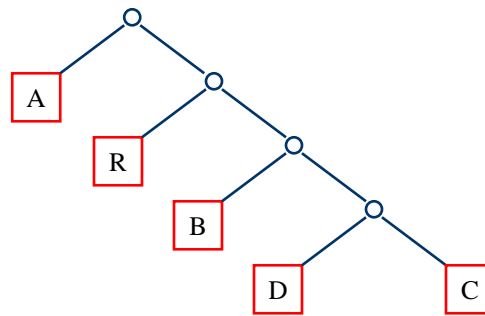
*What is the cost of the ABRACADABRA tree given in the previous slide?

Q - 8

Can We Do Better?

- ABRACADABRA tree yielding 23 bit string

01101001111011100110100.



Q - 9

Huffman Codes

- *Huffman encoding* is a method for constructing a tree which leads to a bit string of minimal length for any given message.
- The first step is to find the frequency counts for the message. For example,

ABRACADABRA Frequency Table

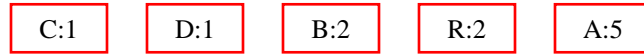
A	5
B	2
C	1
D	1
R	2

Q - 10



Build Tree Bottom Up

- Maintain a priority queue Q keyed on frequency.

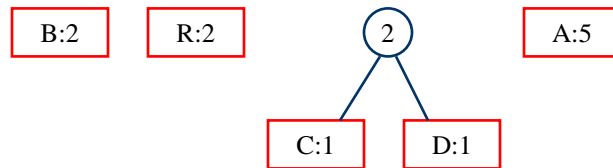


- Remove the two trees with smallest root values and merge them into a new tree.

Q - 11

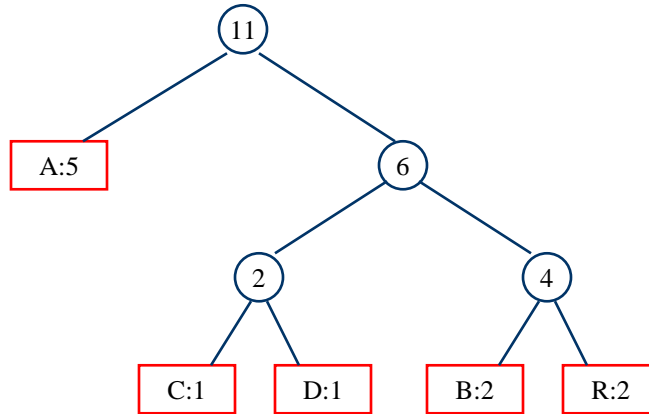


Repeat



Q - 12

To Obtain



Q - 13

Huffman Code

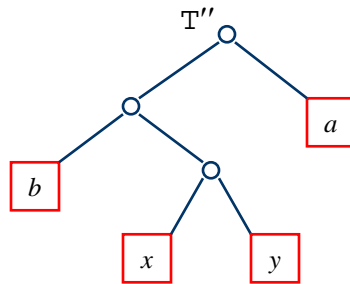
```

Huffman(C)
  n ← |C|
  Q ← C
  for i ← to n-1
    do z ← Allocate-Node()
       x ← left[z] ← Extract-Min(Q)
       y ← right[z] ← Extract-Min(Q)
       f[z] ← f[x] + f[y]
       Insert(Q, z)
  return Extract-Min(Q)
  
```

Q - 14

Greedy-Choice Property

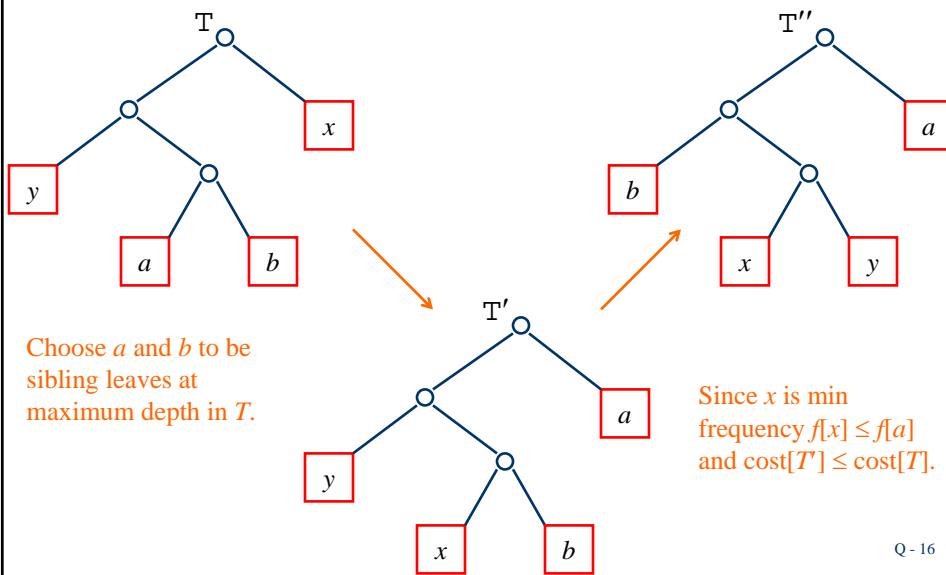
Let $x, y \in \mathcal{C}$ have minimum frequencies.



Then there exists an optimal prefix code for \mathcal{C} in which the codewords for x, y have the same length and differ only by the last bit.

Q - 15

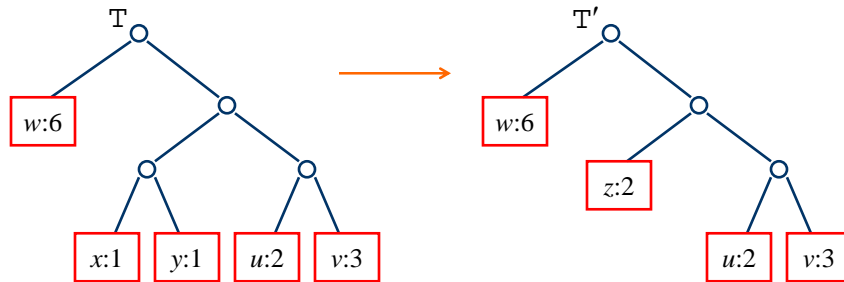
Proof of Greedy-Choice



Q - 16

Optimal-Substructure Property

Let $x, y \in \mathcal{C}$ have minimum frequencies, and T be any optimal prefix code tree for \mathcal{C} in which x, y are siblings.

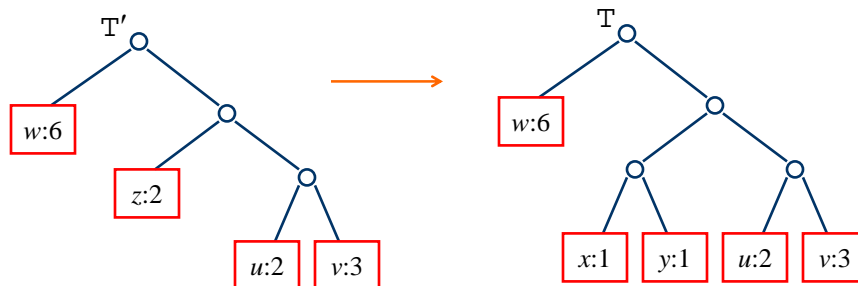


Tree T' obtained from T by replacing the parent of leaves x, y by leaf z , with $f[z] = f[x] + f[y]$, is an optimal prefix code tree for $\mathcal{C}' = \mathcal{C} - \{x, y\} \cup \{z\}$.

Q - 17

And Conversely,

Let $x, y \in \mathcal{C}$ have minimum frequencies, and T' be any optimal prefix code tree for $\mathcal{C}' = \mathcal{C} - \{x, y\} \cup \{z\}$, where $f[z] = f[x] + f[y]$.



Tree T obtained from T' by replacing leaf z by interior node having x, y as children is an optimal prefix code tree for \mathcal{C} .

Q - 18