Middlebury

CSCI 201: Data Structures

Fall 2024

Lab 6: Huffman Compression

Goals for today:

- Manually track a priority queue to build a Huffman Tree.
- Encode a message using a Huffman Coding Tree.
- Decode a sequence of bits using a Huffman Coding Tree.

CSCI 201 Friday, November 1, 2024 Lab 6

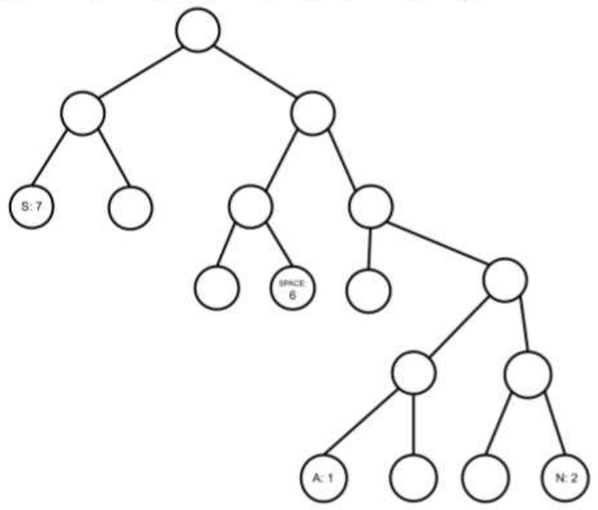
DATA STRUCTURES

Huffman Coding • Priority Queues • Binary Trees • Bits

HuffmanCoding

Consider the priority queue [A:1, T:1, G:2, N:2, H:5, SPACE:6, P:6, S:7, I:8] formed from the frequencies of characters in an input string.

1. *Complete the Huffman Coding Tree below by (1) filling in the nodes and (2) labelling the edges (0 for a left edge, 1 for a right edge).*



2. *Write the encoding (bit sequence) for each character underneath the leaf nodes.*

3. *Use your Huffman Coding Tree to decode the following bit string:*

11101100010010101001011110010100100011101010010001110110
01111111111010100100011101100111111111101010010001110000

WordSearch

priority, height, node, Huffman, complete, heap, binary, queue, tree, full, path, bit

C H C O M P L E T E
Q Q V E E E Y M K Q
B P U K I N Q F H W
C I R E E Y I U E B
P N N I U R X L A I
A X C A O E N L P T
T X U J R R H O M T
H H K Q T Y I C D R
H U F F M A N T B E
Z F H E I G H T Y E

What if we want to send the text bananabread?

11 characters

DEC	OCT	HEX	BIN	Symbol	HTML Number	HTML Name	Description
97	141	61	01100001	a	a		Lowercase a
98	142	62	01100010	b	b		Lowercase b
99	143	63	01100011	c	c		Lowercase c
100	144	64	01100100	d	d		Lowercase d
101	145	65	01100101	e	e		Lowercase e
102	146	66	01100110	f	f		Lowercase f
103	147	67	01100111	g	g		Lowercase g
104	150	68	01101000	h	h		Lowercase h
105	151	69	01101001	i	i		Lowercase i
106	152	6A	01101010	j	j		Lowercase j
107	153	6B	01101011	k	k		Lowercase k
108	154	6C	01101100	l	l		Lowercase l
109	155	6D	01101101	m	m		Lowercase m
110	156	6E	01101110	n	n		Lowercase n
111	157	6F	01101111	o	o		Lowercase o
112	160	70	01110000	p	p		Lowercase p
113	161	71	01110001	q	q		Lowercase q
114	162	72	01110010	r	r		Lowercase r
115	163	73	01110011	s	s		Lowercase s
116	164	74	01110100	t	t		Lowercase t
117	165	75	01110101	u	u		Lowercase u
118	166	76	01110110	v	v		Lowercase v
119	167	77	01110111	w	w		Lowercase w
120	170	78	01111000	x	x		Lowercase x
121	171	79	01111001	y	y		Lowercase y
122	172	7A	01111010	z	z		Lowercase z

https://www.ascii-code.com/

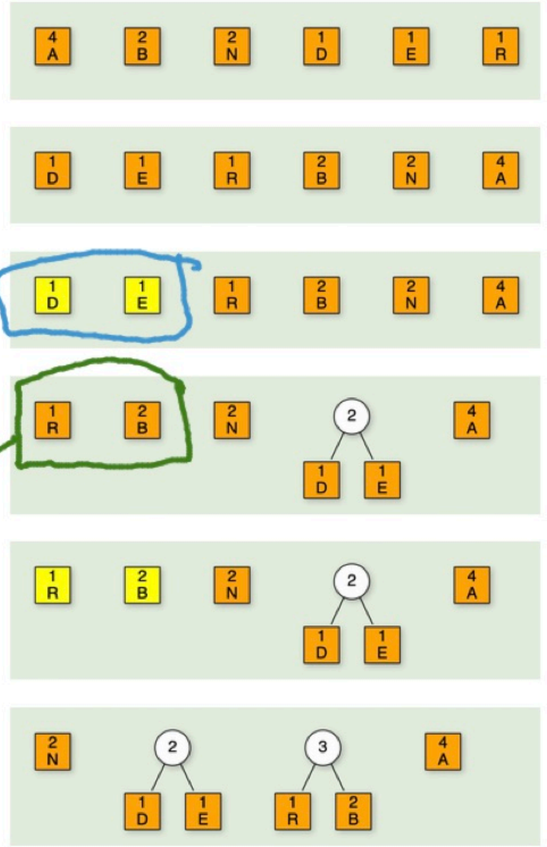
b: 01100010 a: 01100001 .....

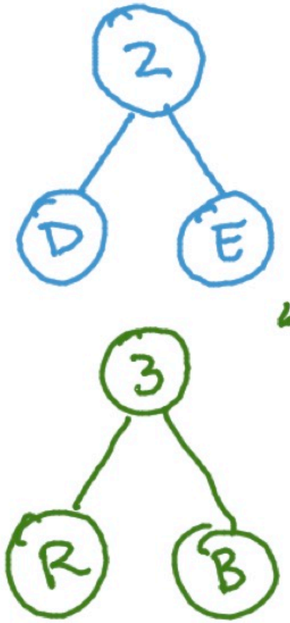
encode string = 8 x n characters = 88 bits

Instead, we can use a variable-length encoding: use a different number of bits for each character.

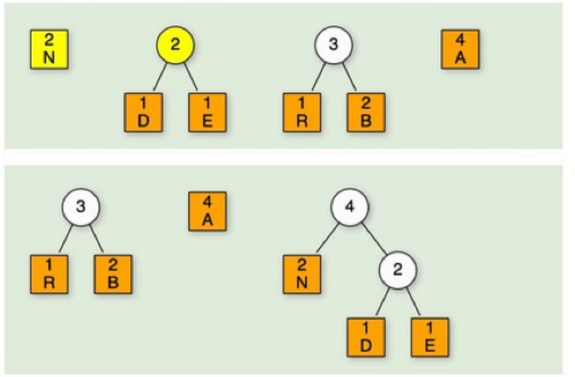
more frequent characters → less bits to represent

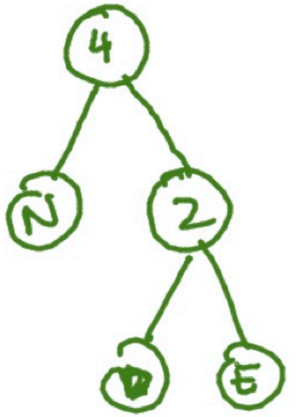
Huffman coding: encode less (more) frequent characters with more (less) bits.



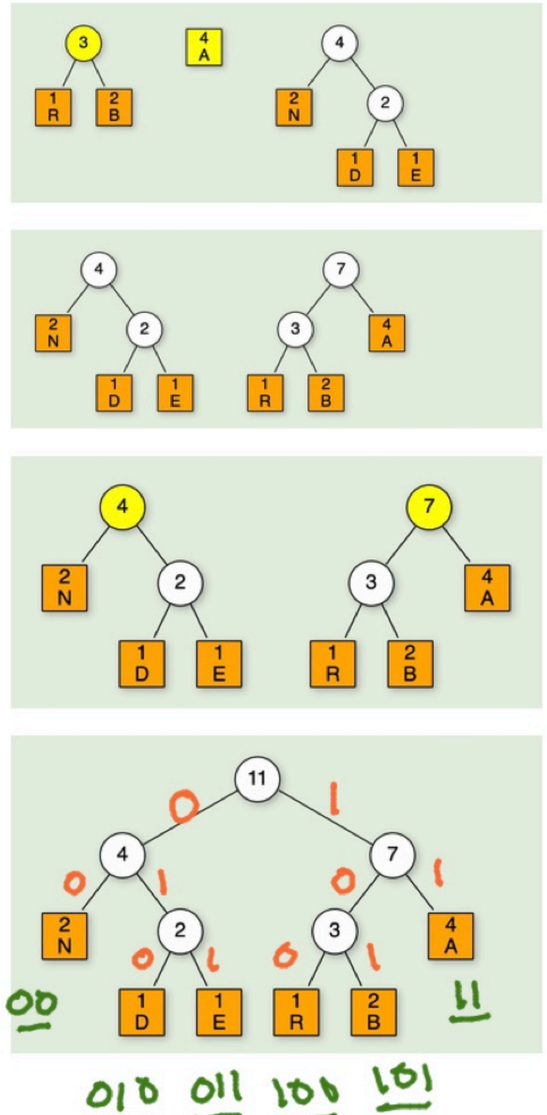


1. Count frequency of each character.
2. Insert characters into **priority queue** (lower frequency has higher priority).
3. `while` priority queue is not empty:
    1. Extract (and remove) top **two** items from priority queue.
    2. Create a new internal node.
    3. Add value (frequency) of two items and assign to internal node.
    4. Make the left child of the new node the first (lower) item.
    5. Make the right child of the new node the second (higher) item.
    6. Insert new node into priority queue.





Completing the Huffman tree for **bananabread**.



Letter	Frequency	Code
a	4	11
b	2	101
n	2	00
d	1	010
e	1	011
r	1	100

b a n a n a b r e a d
101 11 00 11 00 11 101 100 011 11 010

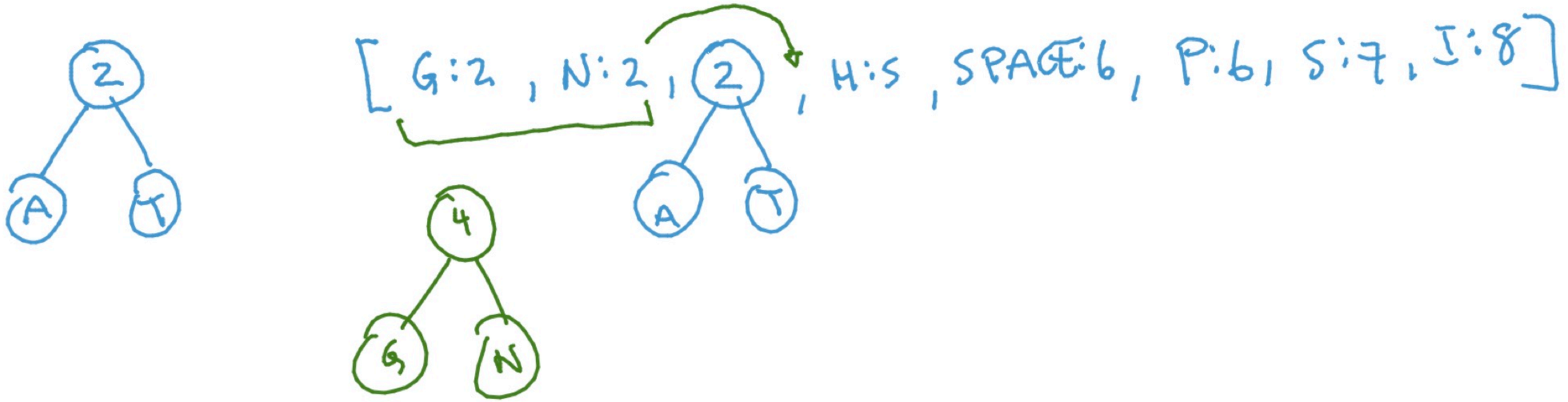
27 bits

traverse tree to retrive characters (until reaching a leaf)

Consider this priority queue.

[A:1, T:1, G:2, N:2, H:5, SPACE:6, P:6, S:7, I:8]

A few steps in building the Huffman Coding Tree:



For the rest of the lab:

- Complete the worksheet to build the Huffman tree for the given frequency of characters.
- Use your tree to decode a sequence of bits.
- Submit worksheet results to Canvas quiz (unlimited attempts).
- Nothing to submit to Gradescope for this lab.

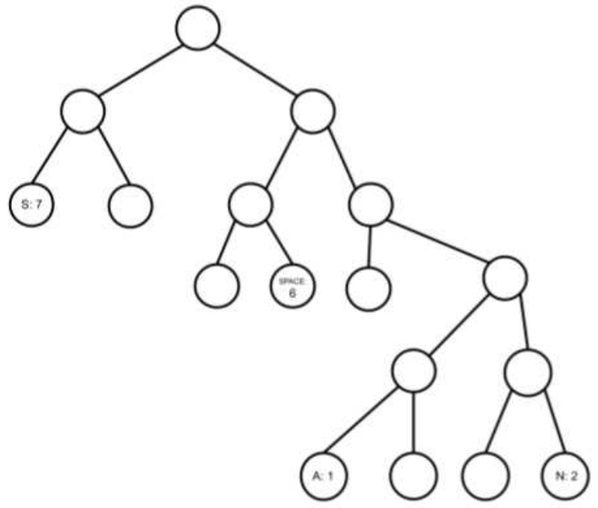
DATA STRUCTURES

Huffman Coding • Priority Queues • Binary Trees • Bits

HuffmanCoding

Consider the priority queue [A:1, T:1, G:2, N:2, H:5, SPACE:6, P:6, S:7, I:8] formed from the frequencies of characters in an input string.

1. *Complete the Huffman Coding Tree below by (1) filling in the nodes and (2) labelling the edges (0 for a left edge, 1 for a right edge).*



2. *Write the encoding (bit sequence) for each character underneath the leaf nodes.*

3. *Use your Huffman Coding Tree to decode the following bit string:*

11101100010010101001011110010100100011101010010001110110
01111111111010100100011101100111111111101010010001110000

WordSearch

priority, height, node, Huffman, complete, heap, binary, queue, tree, full, path, bit

C H C O M P L E T E
Q Q V E E E Y M K Q
B P U K I N Q F H W
C I R E E Y I U E B
P N N I U R X L A I
A X C A O E N L P T
T X U J R R H O M T
H H K Q T Y I C D R
H U F F M A N T B E
Z F H E I G H T Y E