Huffman Trees

From ASCII Coding to Huffman Coding

char	ASCII	Binary
g	103	1100111
o	111	1101111
p	112	1110000
h	104	1101000
e	101	1100101
r	114	1110010
s	115	1110011
space	32	1000000

- Suppose we want to send a message: “go go gophers”.
- The message is encoded with the same number of bits: 8 bits per characters.
- The message has 13 characters; it requires: 8 bits x 13 = 104 bits
- Can we make our own codes? Can the message being sent less than 104 bits?
- Huffman coding is used to compress data being sent.

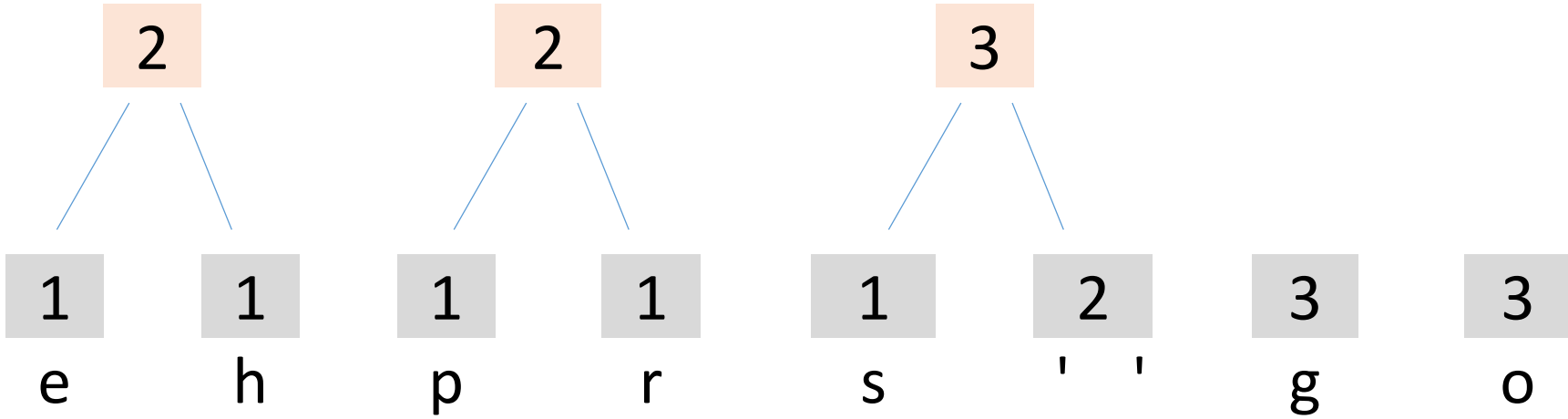
From ASCII Coding to Huffman Coding (cont.)

char	count	Code	Binary (3 bits)
e	1	0	000
h	1	1	001
p	1	2	010
r	1	3	011
s	1	4	100
space	2	5	101
g	3	6	110
0	3	7	111

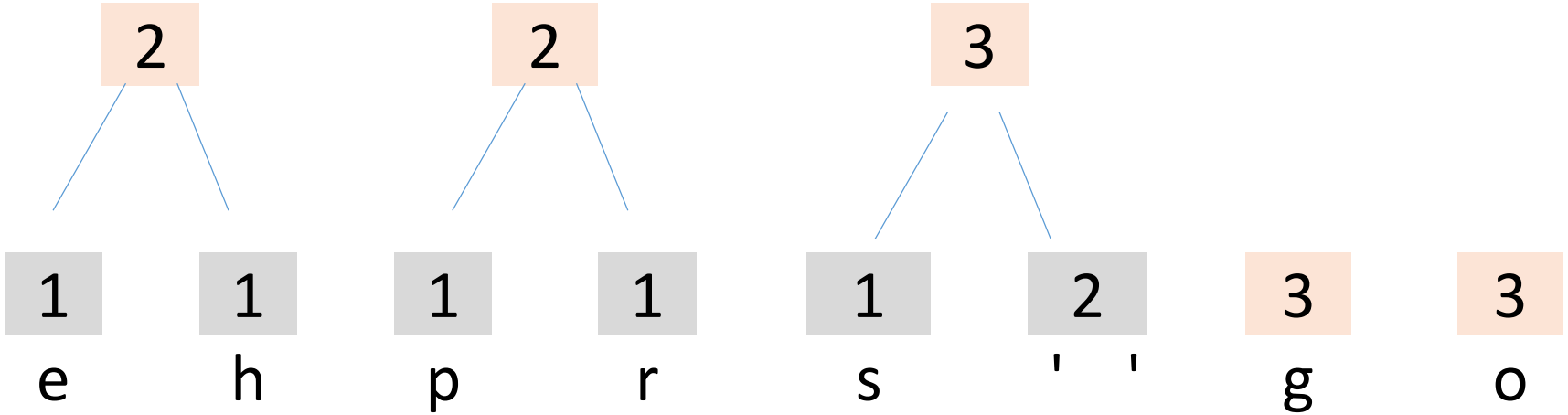
- The message is re-coded with 3 bits per character.
- The message has 13 characters; it requires: 3 bits x 13 = 39 bits
- More bits can be saved if we use fewer than three bits to encode characters like g, o, and space that occur frequently.
- This is the basic idea behind Huffman coding: to use fewer bits for more frequently occurring characters.

1	1	1	1	1	2	3	3
e	h	p	r	s	' '	g	o

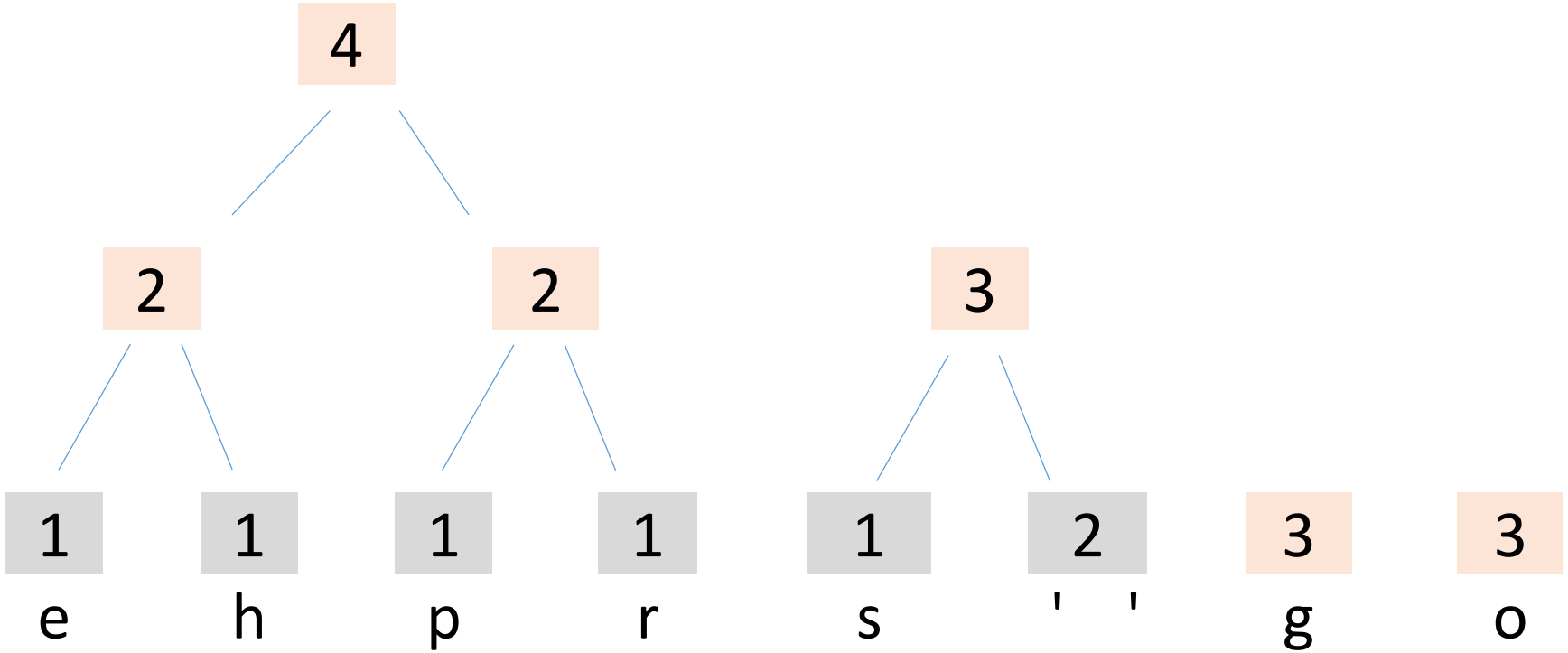
- First, we need to establish the character frequency of the string “go go gophers” from lowest to highest.



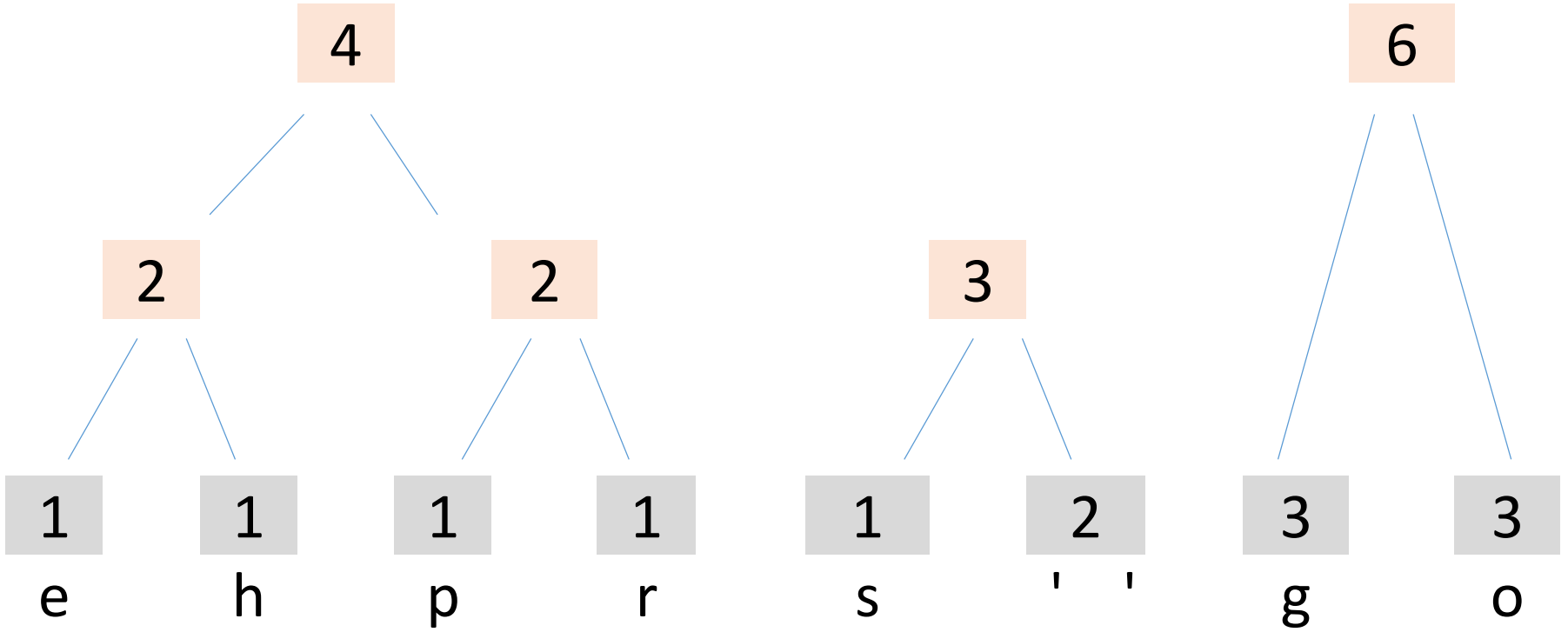
- Create new nodes by adding the two low weight nodes



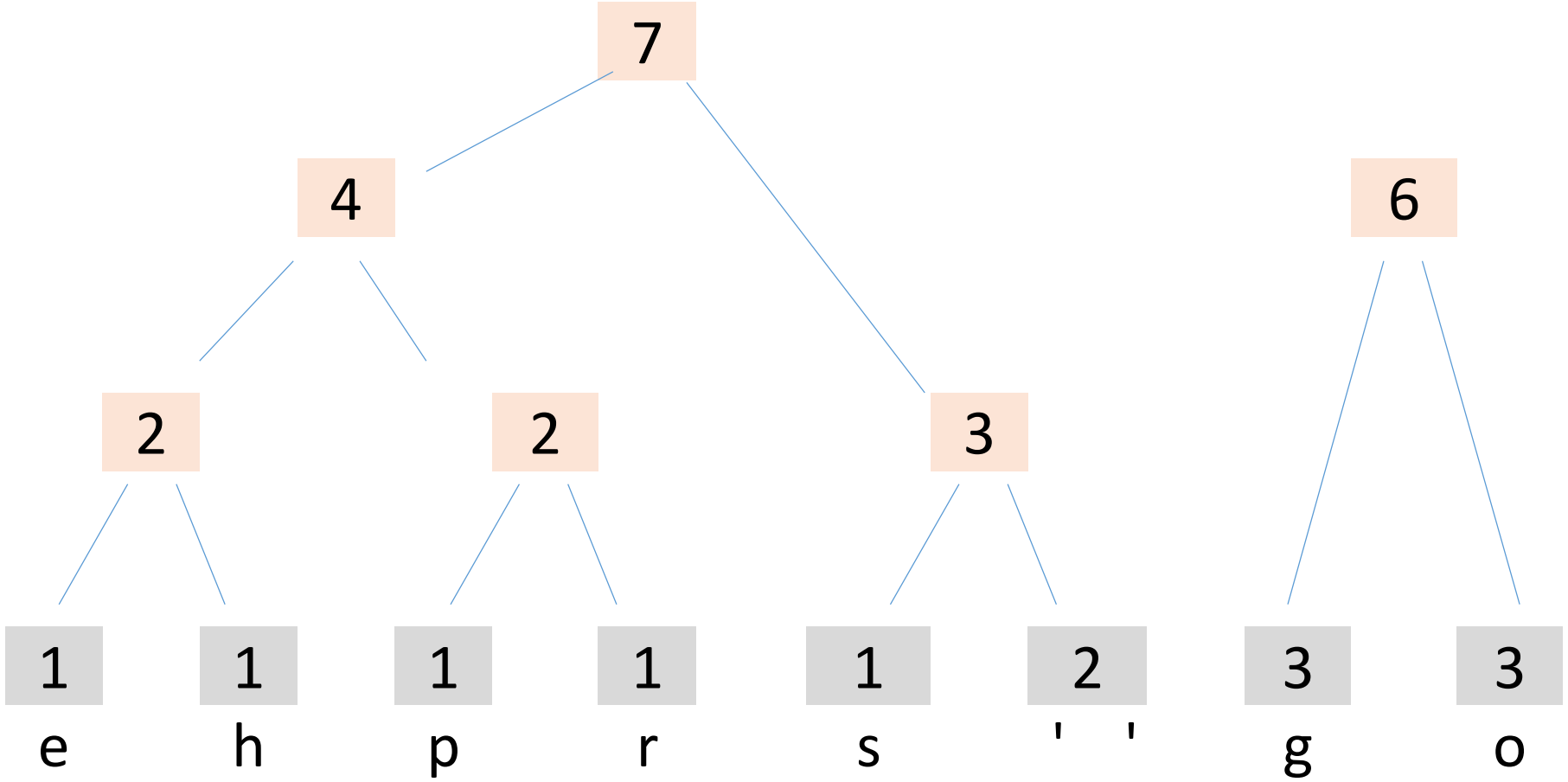
- Create new nodes by adding the two low weight nodes,
- We now have 6 nodes



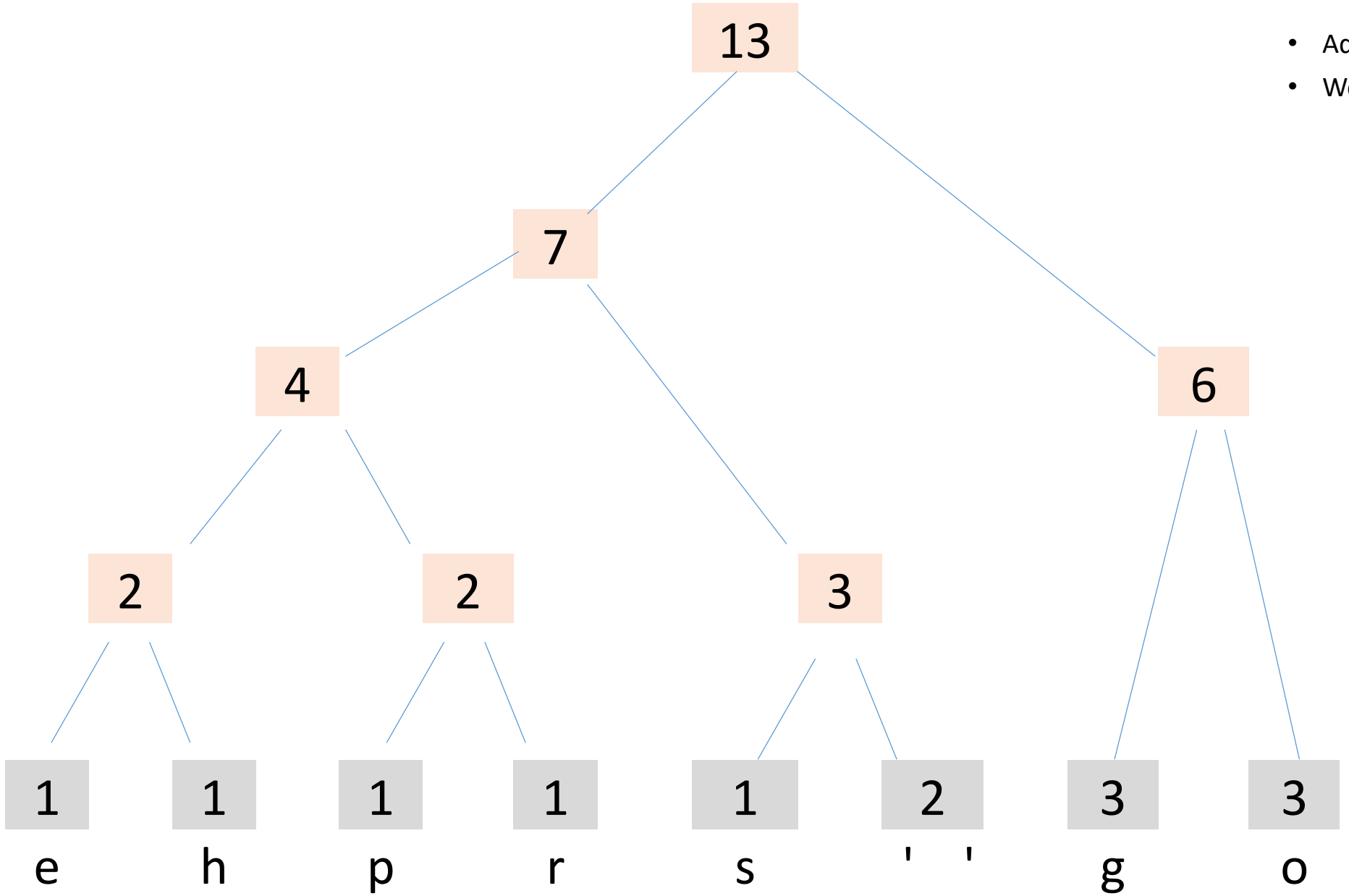
- Again we must choose the two nodes of minimal weight: 2 and 2
- We now have four nodes left: 4, 3, 3 and 3
- Two nodes of minimal weights: 3 and 3



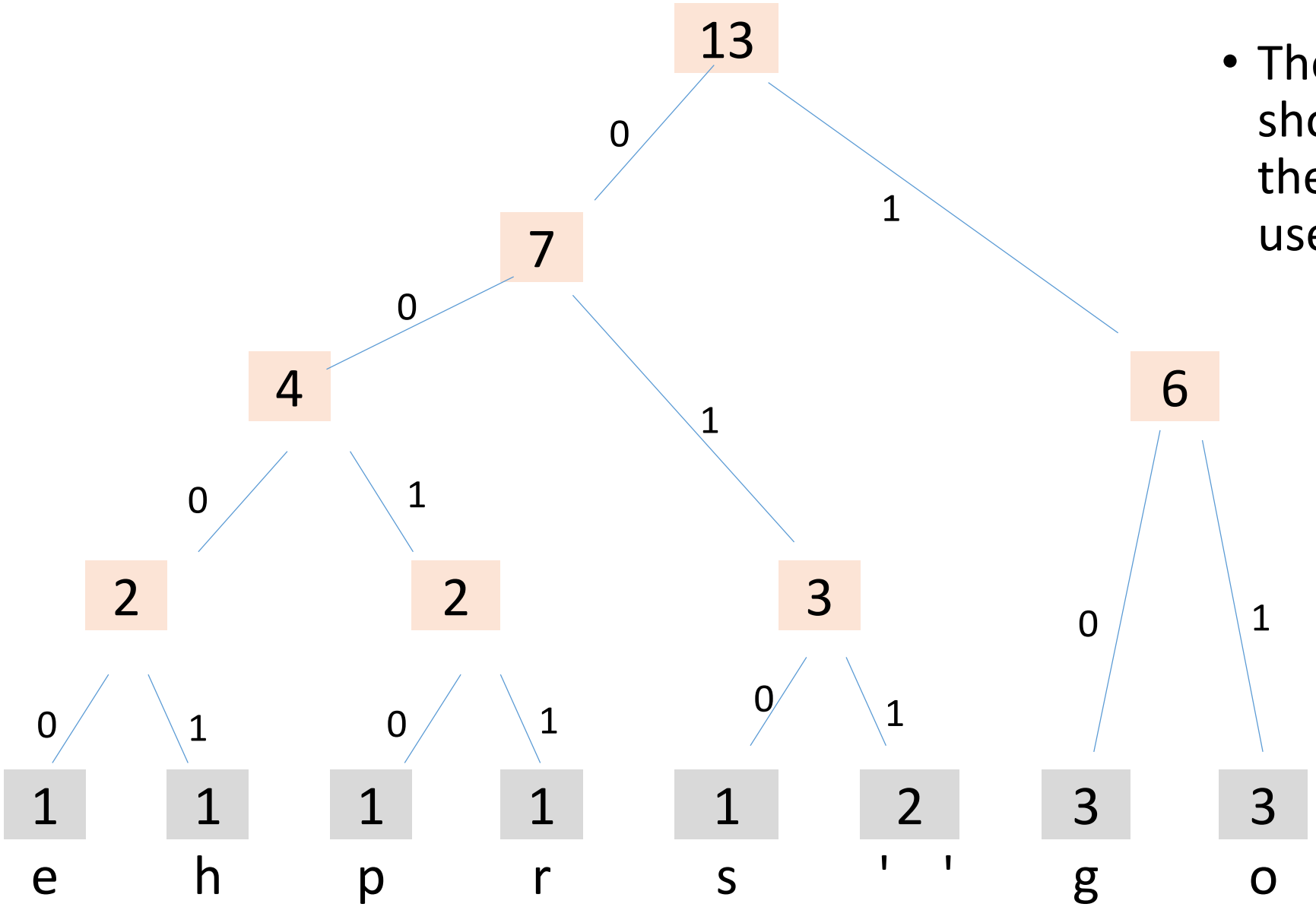
- Adding the two nodes of minimal weight: 3 and 3
- We now have three nodes left: 4, 3 and 6



- Adding 4 and 3, we now have two nodes left: 7 and 6



- Adding the last two nodes: 7 and 6.
- We completed the Huffman tree.



- The character encoding is shown where 0 is use for the left edges, and 1 is used for the right edges.

Huffman Coding

char	count	Huffman Binary	Bit Required
e	1	0000	4
h	1	0001	4
p	1	0010	4
r	1	0011	4
s	1	010	3
space	2	011	6
g	3	10	6
0	3	11	6
			37

- The message - “go go gophers” has 13 characters.
- The message has 13 characters; it requires: 8 bits x 13 = 104 ASCII bits to send this message.
- Huffman coding is used to compress data being sent.
- Huffman code requires 37 bits to send the same message.

Huffman Tree Definition

- Huffman tree is a full binary tree in which each leaf of the tree corresponds to a letter in the given alphabet.
- The weight of the leaf node is the frequency of its associated.
- The goal is to build a tree with the minimum external path weight. The binary tree with minimum external path weight is the one with the minimum sum of weighted path lengths for the given set of leaves.

Steps in Huffman Algorithm

- Huffman coding has four principle steps:

1. Count frequencies – Examine a source file contents and count the number of occurrences of each character, and store them in a map.

2. Build encoding tree – build a binary tree with a particular structure, where each node represents a character and its count of occurrences in the file. A priority queue is used to help build the tree along the way.

3. Build encoding map – Traverse the binary tree to discover the binary encodings of each character.

4. Encode data or read the source file a second time. For each character read, instead of writing the character, write the sequence of zero and one bits used to encode the character. This sequence of bits was determined in the previous steps.

The End!