Huffman & Fixed-length Code

Without encoding:

cost of transmission, coding & decoding

	Ascii Code	Binary Form	
A	65	01000001	← 8 bits
B	66	01000010	
C	67	⋮	
⋮	⋮		

B C C A B B D D A E C C B B A E D D C C

20 letters, 8 bits per

20 x 8 = 160 bits without encoding (cost of transmission)

Fixed-length:

01

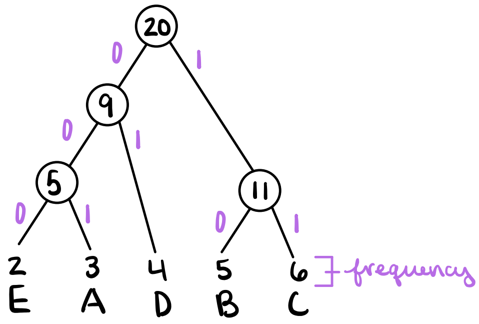
00
01
10
11 } 4 combinations (2^2)

0
1 } 2 combos (2^1)

3 bits → 8 combos (2^3)

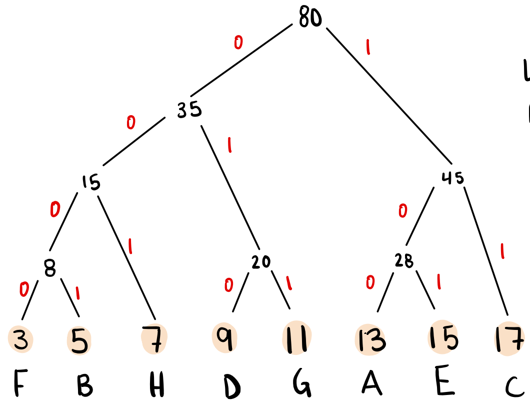
Huffman (Var-Size):

1. arrange letters in order of frequency (small → large)
2. greedy - optimal merge pattern:
   - merge the 2 smaller lists
   - merge the new list with the next smallest list
3. assign code to each character
   - Left branch = 0, Right branch = 1



HUFFMAN CODE:

Character	Count / Frequency
A	13
B	5
C	17
D	9
E	15
F	3
G	11
H	7



LHS branch 0
RHS branch 1

Character	Count / Frequency	Variable-Size-code	Message size for letter
F	3	0000	4 bits * 3 = 12 bits
B	5	0001	4 bits * 5 = 20 bits
H	7	001	3 bits * 7 = 21 bits
D	9	010	3 bits * 9 = 27 bits
G	11	011	3 bits * 11 = 33 bits
A	13	100	3 bits * 13 = 39 bits
E	15	101	3 bits * 15 = 45 bits
C	17	11	2 bits * 17 = 34 bits
	80 bits total	25 bits total	231 bits total

decoding:

+ 8 chars * 8 bits	25 bits (var-size-code) 64 bits (ascii)
	89 bits

+

length of Huffman encoded message:
89 bits + 231 bits = 320 bits

↓ continued

8 chars
$2^3 = 8$ combos
↳ 3 bits

FIXED-LENGTH CODE:

3 bits each

Character	Count / Frequency	Fixed-Size-Code
F	3	000
B	5	001
H	7	010
D	9	011
G	11	100
A	13	110
E	15	111
C	17	101
	80 bits total	

own code → 8 chars * 3 bits = 24 bits

Num bits for table of Code

	24 bits (own)
+ 8 chars * 8 bits	64 bits (ascii)
	88 bits

message size:
length 80 bits * 3 bits = 240 bits

+

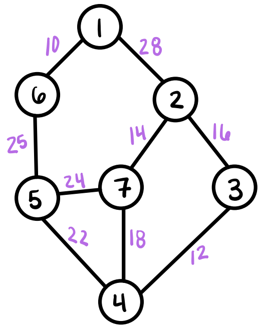
length of fixed-length encoded message:
88 bits + 240 bits = 328 bits

Kruskal's Algorithm

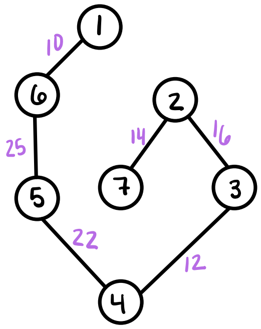
1. min cost edge selected

   and doesn't have to be connected to already-selected edge

2. no cycles



1	10	6	
3	12	4	
2	14	7	
7	16	4	X cycle
5	22	4	
5	24	7	X cycle
5	25	6	



now at 6 edges, done!

$|V| - 1 = |E'|$

$7 - 1 = 6$

Time complexity:

E num edges, and we select min cost edge.

$|V|-1$ edges included in spanning tree.

$$\Theta(|E|(|V|-1))$$

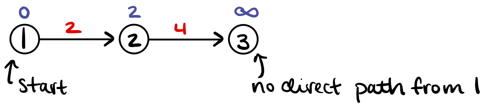
worst case : $\Theta(E n) = (n \times n) = O(n^2)$

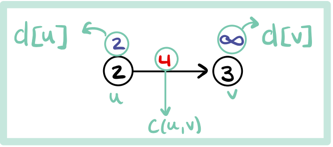
best case : improve w/ Minheap $O(n \log n)$

Dijkstra's Algorithm

DIRECT SHORTEST PATH

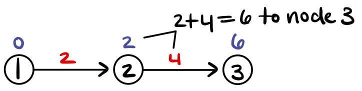
find shortest from start v to other v (any v can be start)



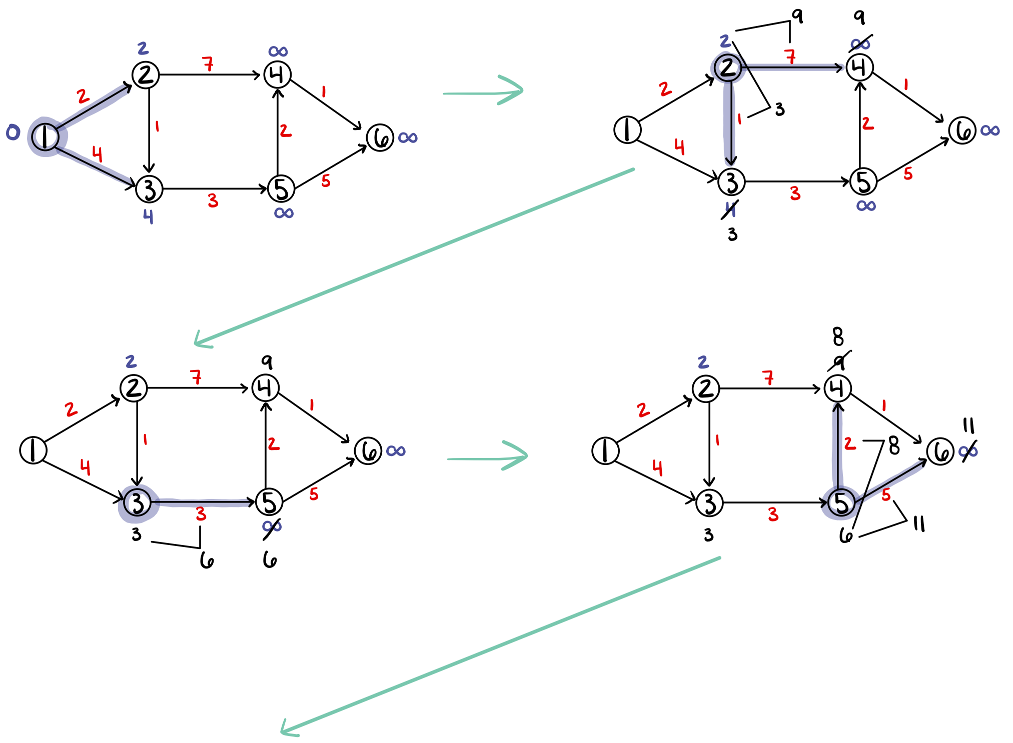


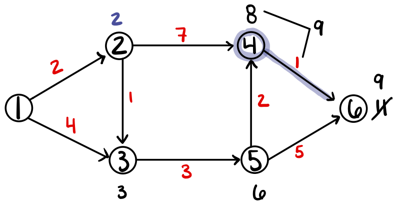
RELAXATION

if d[u] + c(u,v) < d[v] then do relaxation



1. DIRECT SHORTEST PATH
2. mark ∞ for non-direct paths
3. RELAXATION





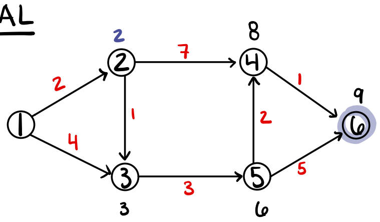
starting point vertex 1 ...

1 to 2 = 2

1 to 3 = 3

1 to 4 = 8

FINAL



Vertex	distance
1	0
②	2
③	3
④	8
⑤	6
⑥	9