Lecture 1 02.22.23

Giving you a string of letters like tails.

H ← But if this is a head, this is might suprise you.

T ← come back tell you this is a tail. no interesting.

T T T T T T T T T T T T — T T T T T T T T T T

Something that doesn't happen very often gives you a lot more information when it actually happens.

if i have a string

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

If you do Huffman coding you're going to encode C and D with more bits.

There is actually a way of computing how much information something has

$\log_2 \frac{1}{P_i}$

P_i → probability of it occurring.

2: because information is measure by bits.

Lecture 1 02.22.23

Giving you a string of letters like tails.

H ← But if this is a head, this is might suprise you.

T ← come back tell you this is a tail. no interesting.

T T T T T T T T T T T T _ T T T T T T T T T T

Something that doesn't happen very often gives you a lot more information when it actually happens.

if i have a string

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

If you do Huffman coding you're going to encode C and D with more bits.

There is actually a way of computing how much information something has

$$\log_2 \frac{1}{p_i}$$

p_i: probability of it occurring

2: because information is measure by bits.

For example: Quiz #2 Problem 1.

A: $\log_2 \frac{1}{\frac{20}{64}} = \log_2 \frac{64}{20} = 1.68$ bits.

B: same as A = 1.68 bits

D: $\log_2 \frac{1}{\frac{7}{64}} = \log_2 \frac{64}{7} = 3.19$ bits

E: same as D = 3.19 bits.

F: $\log_2 \frac{64}{3} = 4.415$ bits

G: $\log_2 \frac{64}{3} = 4.415$ bits

H: $\log_2 \frac{64}{2} = 5$ bits

K: 5 bits

A: 20 x 1.68 bits = 33.6 bits

B: 20 x 1.68 = 33.6 bits

D: 7 x 3.19 bits = 22.33 bits

E: 7 x 3.19 bits = 22.33 bits

F: 3 x 4.415 bits = 13.245 bits

G: 3 x 4.415 = 13.245 bits

H: 5 x 2 = 10 bits

K: 5 x 2 = 10 bits

154.35 bits

at least you have so much bits to store the information.

Huffman: 162 bits

H

T T T T T T T T T T T T — T T T T T T T T T T

21 total

$p_H = \frac{1}{21}$ $p_T = \frac{20}{21}$

$\log_2 \frac{1}{\frac{1}{21}} = 4.392$ bits $\log \frac{1}{\frac{20}{21}} = 0.07$ bits

total $4.392 \times 21 + 20 \times 0.07 = 5.79$ bits

if they all "T"s: $p_T = \frac{21}{21} = 1$ $\log_2 1 = 0$ bits

30 tails. 70 heads.

$p_t = \frac{30}{100} = 0.3$ $p_t = 0.7$

$\log_2 \frac{1}{0.3}$ $\log_2 \frac{1}{0.7}$

total $= 30 \log_2 \frac{1}{0.3} + 70 \times \log_2 \frac{1}{0.7} \approx 88.129$ bits

50 tails 50 Heads

$\log_2 \frac{1}{\frac{50}{100}} = \log_2 \frac{1}{\frac{1}{2}} = \log_2 2 = 1$ bit

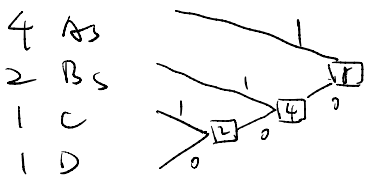
total $= 50 \times 1 + 50 \times 1 = 100$ bits

Huffman coding

A B A C B A A D

There is a message has

4 As
2 Bs
1 C
1 D



1	1	0
10	01	10
100	001	110
000	000	111

There are different Huffman coding.

$4 \times 1 + 2 \times 2 + 3 \times 1 + 3 \times 1 = 14$ bits

Information:

A: $\log_2 \frac{1}{\frac{4}{8}} = \log_2 2 = 1$ bits

B: $\log_2 \frac{1}{\frac{2}{8}} = \log_2 4 = 2$ bits

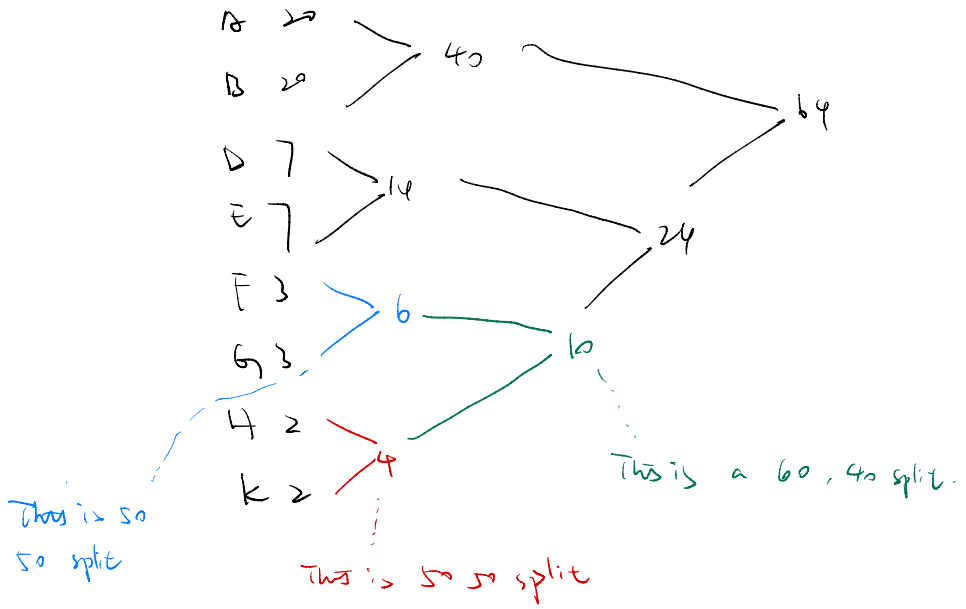
C: $\log_2 8 = 3$ bits

D: $\log_2 8 = 3$ bits

total: $1 \times 4 + 2 \times 2 + 3 \times 1 + 3 \times 1 = 14$ bits

I was always combing two equivalent things. 50! 50

perfect coding. So what make them perfect?



about $\frac{1}{3}$, $\frac{2}{3}$ split

So the fact that you have these split here that are not 50, 50 split. is the reason that you end up with coding 162 bits physical bits to give you 158.33 bits

1 0 0 11 0 10 0 10 11 0 1 0 11 0 1 0 = 1000 bits here

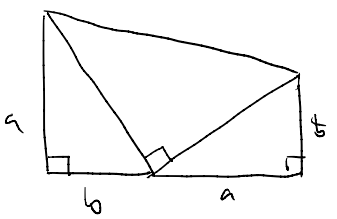
70% get 1

20% get 0

only have 881.3 bits of entropy

So sometimes when you hear people, you can't just count them and expect them exactly even number of them. The thing is did you flip a coin and it was the coin fair where did you have equal likelihood of getting a zero and one every time. does not mean you'll actually come up with an even number. if you flipping a coin come up with an encryption key but if you find out that coin was weighted a little bit more so that it is favored one over the other that's when the entropy of a key goes down and you may have heard of RSA or other people calling bank keys because it didn't have as much entropy in it as they expected it to have and that's what they mean the zero and ones were more predictable than they should have been.

Pythagorean Theorem



$$\frac{1}{2}ab + \frac{1}{2}ab + \frac{1}{2}c^2 = ab + \frac{1}{2}c^2$$

$$\frac{1}{2}(h_1+h_2)h = \frac{1}{2}(a+b)(a+b) = \frac{1}{2}(a^2+2ab+b^2) = \frac{1}{2}a^2 + ab + \frac{1}{2}b^2$$

$$ab + \frac{1}{2}c^2 = \frac{1}{2}a^2 + ab + \frac{1}{2}b^2 \Rightarrow c^2 = a^2 + b^2$$

Euler five funtion.

$\phi(n)$ # of integers $< n$ and relatively prime to n. including 1

$\phi(6)$ ① ~~2~~ ~~3~~ ~~4~~ ⑤ $\phi(6) = 2$

$\phi(15)$ ① ② ~~3~~ ④ ~~5~~ ~~6~~ ⑦ ⑧ ~~9~~ ~~10~~ ⑪ ~~12~~ ⑬ ⑭ $\phi(6) = 8$

$\phi(p) = p-1$ p is prime.

$\phi(p \cdot q) = (p-1)(q-1)$ $p \cdot q$ are prime.

$\phi(15) = \phi(3\times5) = 2\times4 = 8$

$\phi(77) = \phi(7)\cdot\phi(11) = 6\times10 = 60$

Prime factorization

$$n = p_1^{k_1} \cdot p_2^{k_2} \cdot p_3^{k_3} \cdot \cdots \cdot p_n^{k_n}$$

$540 = 2^2 \cdot 3^3 \cdot 5^1$

any integer has a unique prime factorization. it's kind of a daunting thing to say but that's the fundamental theorem of arithmetic that does sounds awfully daunting doesn't it.

/'dɔ:ntiŋ/ adj. 使人气馁的，令人胆怯的.

The question is how would we find 540 knowing that?

$\phi(3) = 2$ $\phi(5) = 4$

$3^2 \to$ $\phi(2^2) = 6$

1 2 ~~3~~ 4 5 ~~6~~ 7 8 ~~9~~

$\phi(3^3)$

$3^3 - 3^2$

1 2 3 4 5 6 7 8 9
10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25 26 27

$5^7 - 5^6$

$\phi(p^k) = p^k - p^{k-1} = p^{k-1}(p-1) = p^k\left(1-\frac{1}{p}\right)$

$\phi(n) = \phi(p_1^{k_1})\,\phi(p_2^{k_2})\,\phi(p_3^{k_3}) \cdots \phi(p_n^{k_n})$

$\phi(n) = p_1^{k_1}\left(1-\frac{1}{p_1}\right) p_2^{k_2}\left(1-\frac{1}{p_2}\right) p_3^{k_3}\left(1-\frac{1}{p_3}\right) \cdots p_n^{k_n}\left(1-\frac{1}{p_n}\right)$

$= p_1^{k_1} p_2^{k_2} p_3^{k_3} \cdots p_n^{k_n}\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right) \cdots \left(1-\frac{1}{p_n}\right)$

$$\phi(n) = n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_n}\right)$$

$540 = 2^2 + 3^2 + 5^1 = \phi(540) = 540\left(1-\frac{1}{2}\right)\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right)$

$= 540 \cdot \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{4}{5}$

$\phi(44)$ $\quad 44 = 2^2 \cdot 11 = 44\left(\frac{1}{2}\right)\left(\frac{10}{11}\right)$

$2 \cdot 3 \cdot 5^2 = 150$

$\phi(150)$ $\quad 150\left(\frac{1}{2} \cdot \frac{2}{3} \cdot \frac{4}{5}\right)$

$2 \cdot 3 \cdot 5^2 \cdot 7 = 1050$ $\quad \phi(1050) = 1050\left(1-\frac{1}{2}\right)\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right)\left(1-\frac{1}{7}\right)$

$= 1050\left(\frac{1}{2}\right)\left(\frac{2}{3}\right)\frac{4}{5} \cdot \frac{6}{7}$

$= 1050 \cdot \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{4}{5} \cdot \frac{6}{7}$

$\phi(p \cdot q) = (p-1 \cdot q-1)$

$\phi(77) = 77\left(\frac{6}{7} \cdot \frac{10}{11}\right) = 60$

1	$2^0 \% 11 = 1$
2	$2^1 \% 11 = 2$
4	$2^2 \% 11 = 4$
8	$2^3 \% 11 = 8$
16	$2^{40} \% 11 = 5$
32	$2^5 \% 11 = 10$
64	$2^6 \% 11 = 9$
128	$2^7 \% 11 = 7$
256	$2^8 \% 11 = 3$
512	$2^9 \% 11 = 6$
1024	$2^{10} \% 11 = 1$
2048	$2^{11} \% 11 = 2$

Fermat's little theorem

There's a theorem that says

$a^p \% p \equiv a$ $\qquad 2^{11} \% 11 = 2$

That's equivalent to say that

$$a^{p-1} \% p = 1$$

$2^{13-1} \% 13 \equiv 1$

$2^{17-1} \% 17 = 1$

$2^{31-1} \% 31 = 1$

Euler's theorem Book Pg32

$a^{\phi(n)} \% n = 1$

if $n = p$, then you will get $a^{p-1} \% p = 1$

11×13

$a^{10 \cdot 12} \% (11 \cdot 13) = 1$

$a^{120} \% 143 = 1$

$a^{\phi(n)} \% n = 1$

$\phi(p \cdot q) = (p-1)(q-1)$ when p and q are prime.

7×17

$a^{6 \cdot 16} \% (7 \cdot 17) = 1$

$a^{96} \% 119 = 1$

$31^{96} \% 119 = 1$

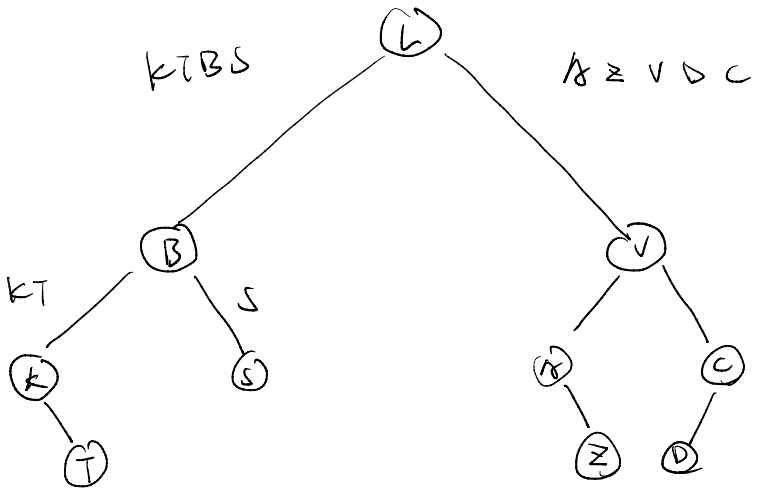
$a^{\phi(n)} \% n = 1$

$a^{\phi(n)+1} \% n = a$

Tree serialization

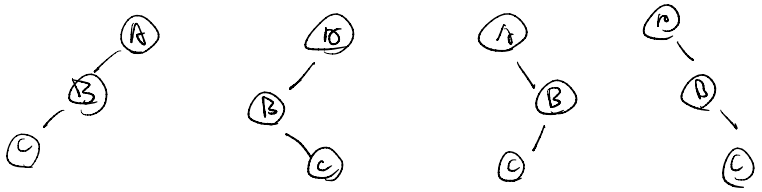
Inorder : K T B S L R Z V D C left parent right

Preorder : L B K T S V R Z C D Draw Tree parent left right



Pre R B C

Post C B R



Inorder C B R B C R R C B R B C

Test #1 Mar 8th

Spring Break Mar 15th

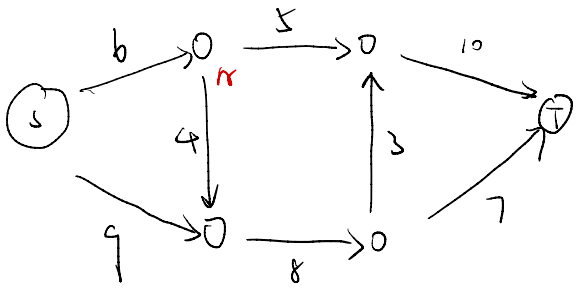
Test #2 Apr. 12th

Test #3 Apr 26th Final project.

Ford Fulkerson Max Flow

Max flow algorithm.

if you have a starting vertex s and you have a graph here



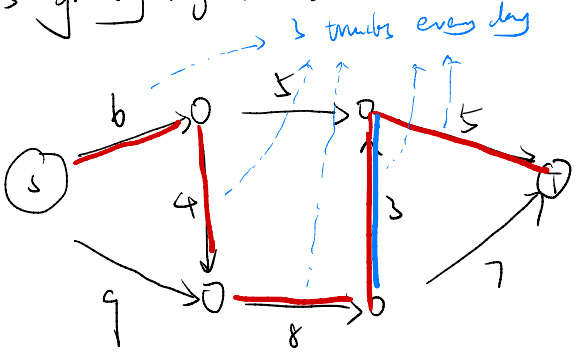
The question is I want to get as much flow from s to T as I can through this whole graph.

Let's pretend these are different cities and I have from s, this is my factory

T is warehouse and I have 6 trucks from s to b

Q: How many truckloads of goods can I get from factory s to my warehouse T.

This is greedy algorithm.

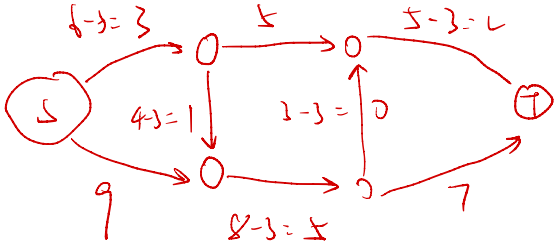


(1)

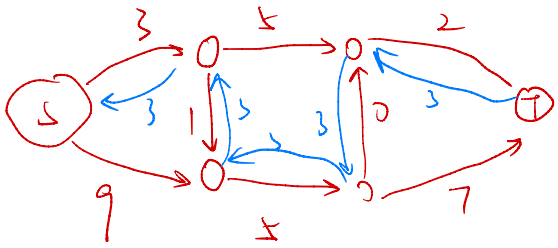
If I picked one path like above, it's probably not a good one. I'm going to use that path. keep in mind I can't store anything at these intermediate locations.

How many truckloads can I get from s to T following the path? 3

redo the graph



Now what I can do then is on my next iteration of the graph I can choose to subtract one of those trucks and only send two trucks, because it might help me find a better path.

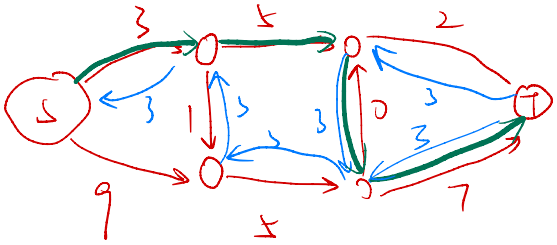


⟵ : going backward.

So to represent that I'm going to show that I have 3 trucks going the opposite direction that I can use. That's effectively subtracting one of the ones that I have already allocated

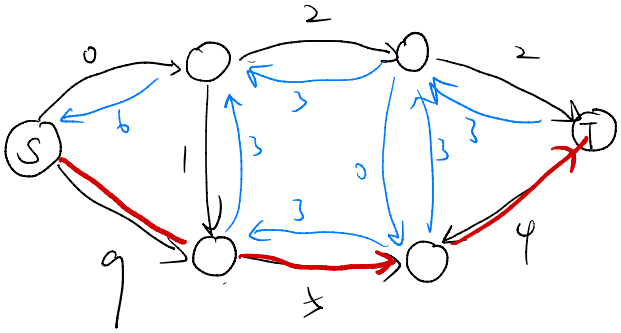
So now you find another path. so the path I'm going to find next is going to be this one

(2)



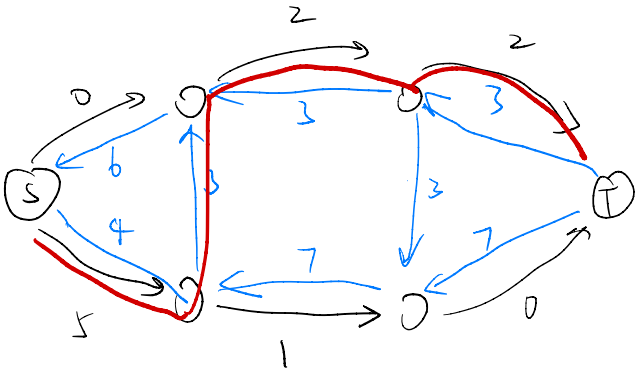
So now I'm going to redraw my graph

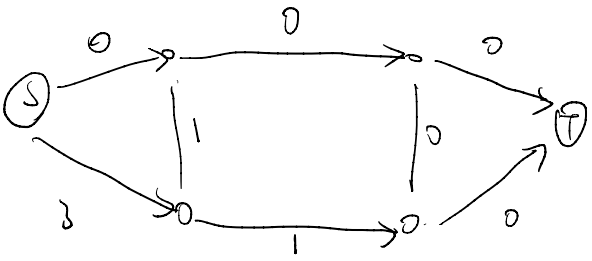
(3)



it's just do the same thing over and over again. Find a path, change the graph, find a path, change the graph

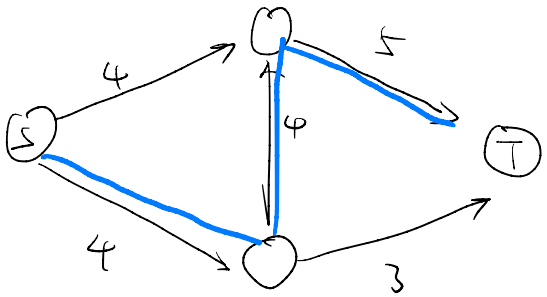
(4)





$3+3+4+2 = 12$ maximum

5

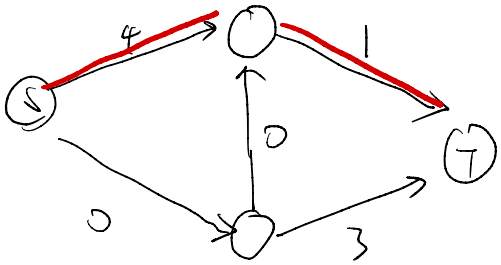


The algorithm will help you solve the problem.

you choose the bad choice

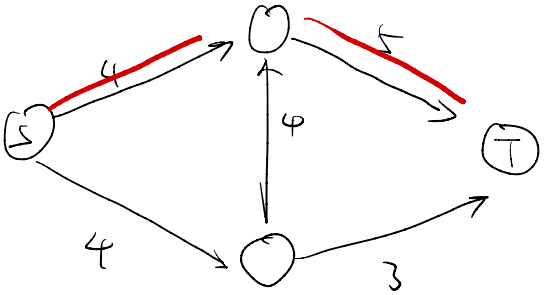
1

bad.

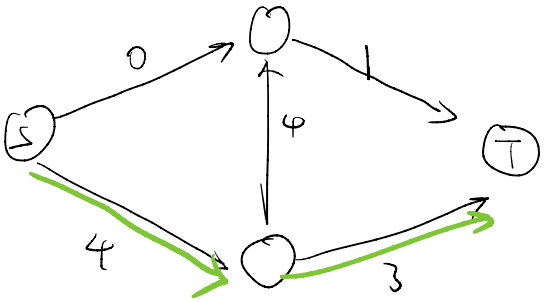


---

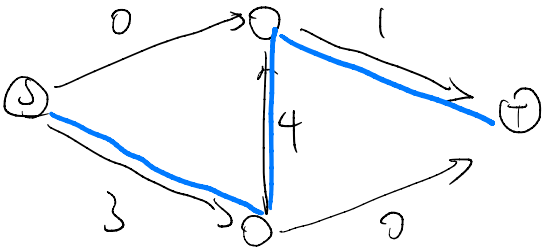
① 4

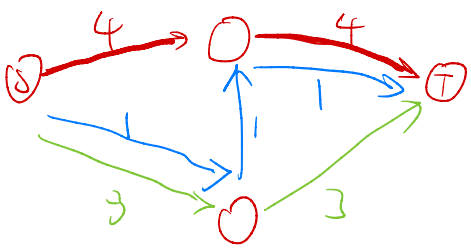


② 3



③ 1





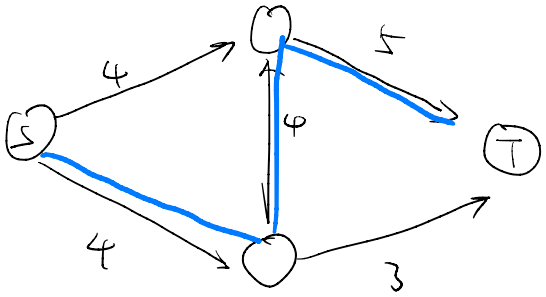
First, you have to understand what the right answer is.

The right answer is have three different flows.

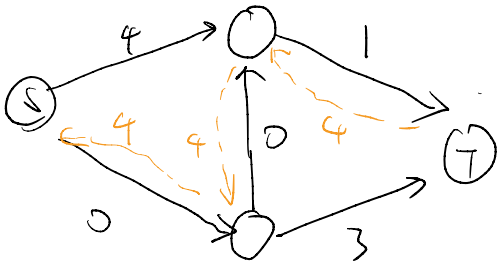
1, 4, 3

if you choose the bad choice, this algorithm doesn't matter if you messed up; it can still give you the right answer.

5

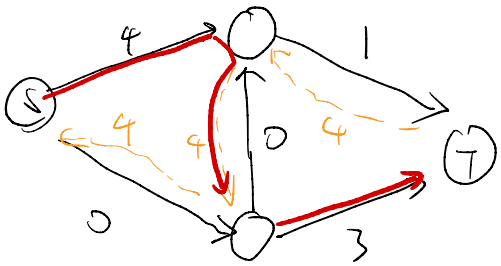


means I'm planning on sending 4 trucks from here to here, I could decide not to send this.

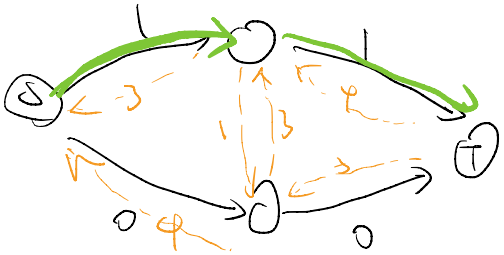


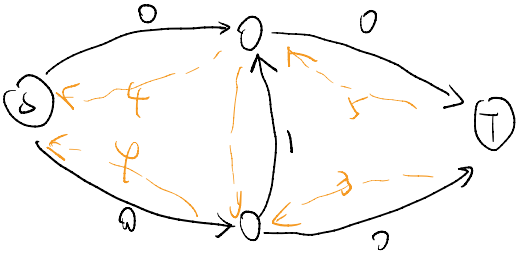
so now we pick another path

3



1





Now say I now have 2 sources & 3 sink