5800
Max Flows

mar 23/24 2022

shelat

Max flow

Min Cut

“Consider a rail network connecting two cities by way of a number of intermediate cities, where each link of the network has a number assigned to it representing its capacity. Assuming a steady state condition, find a maximal flow from one given city to the other.”

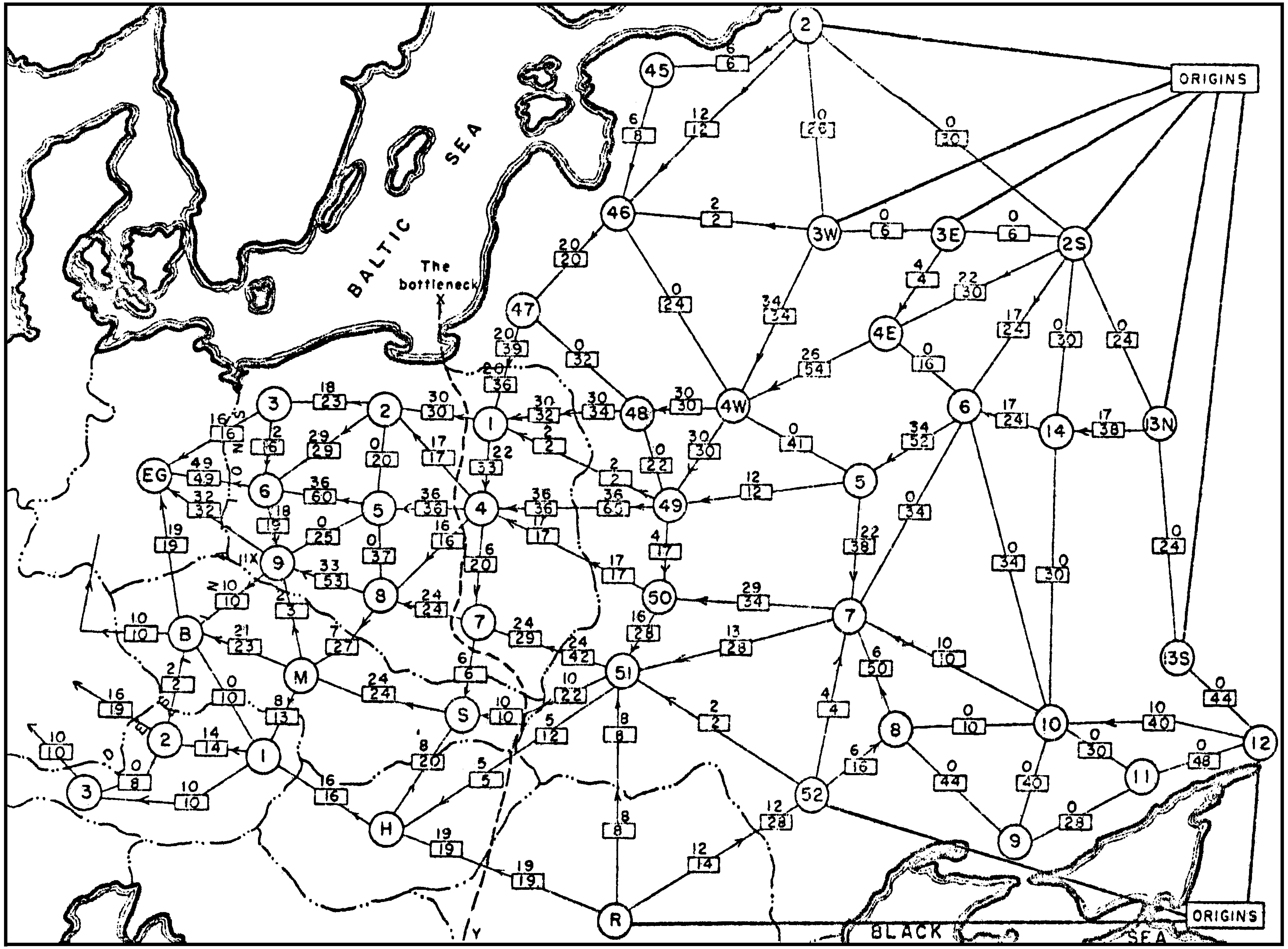


Figure 4 From Harris and Ross [3]: Schematic diagram of the railway network of the Western Soviet Union and East European countries, with a maximum flow of value 163,000 tons from Russia to Eastern Europe and a cut of capacity 163,000 tons indicated as 'The bottleneck'

courtesy Alexander Schrijver

flow networks

$$G = (V, E)$$

source + sink:

capacities:

flow networks

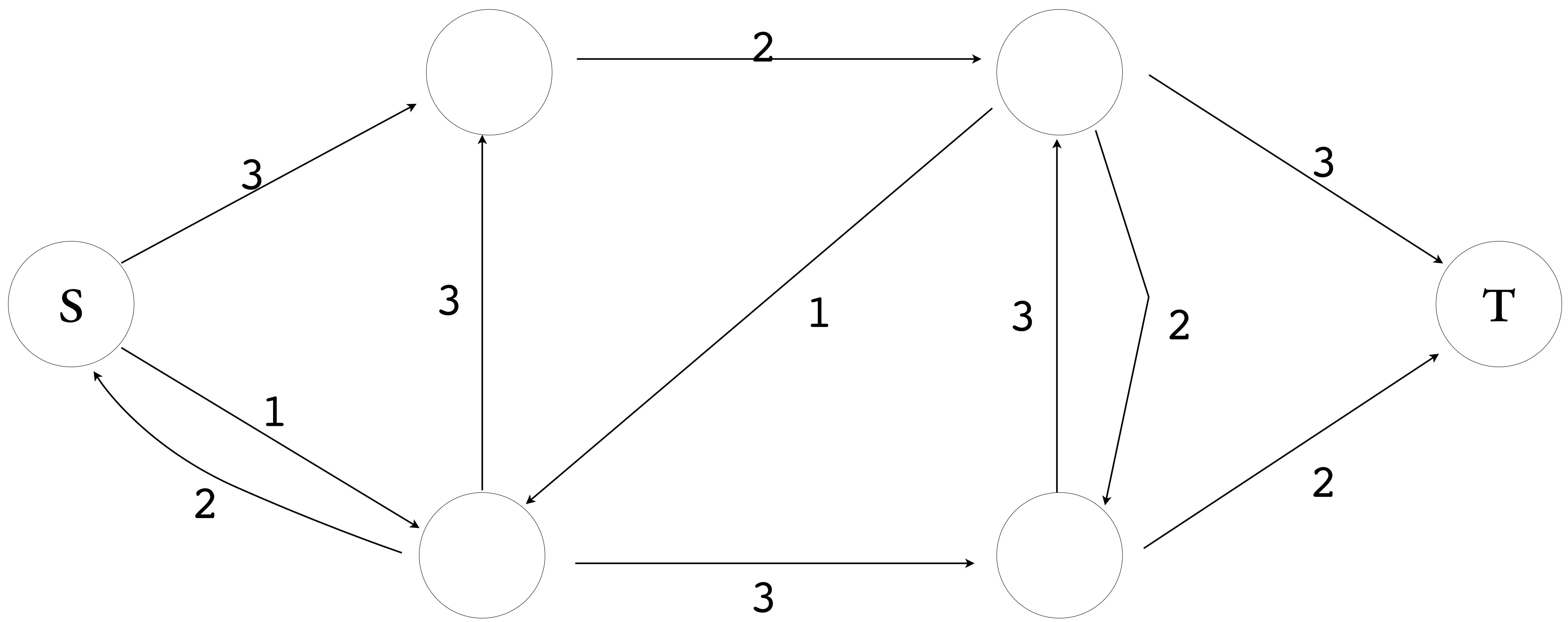
$$G = (V, E)$$

source + sink: node s, and t

capacities: $c(u, v)$

assumed to be 0 if no (u,v) edge

example



flow

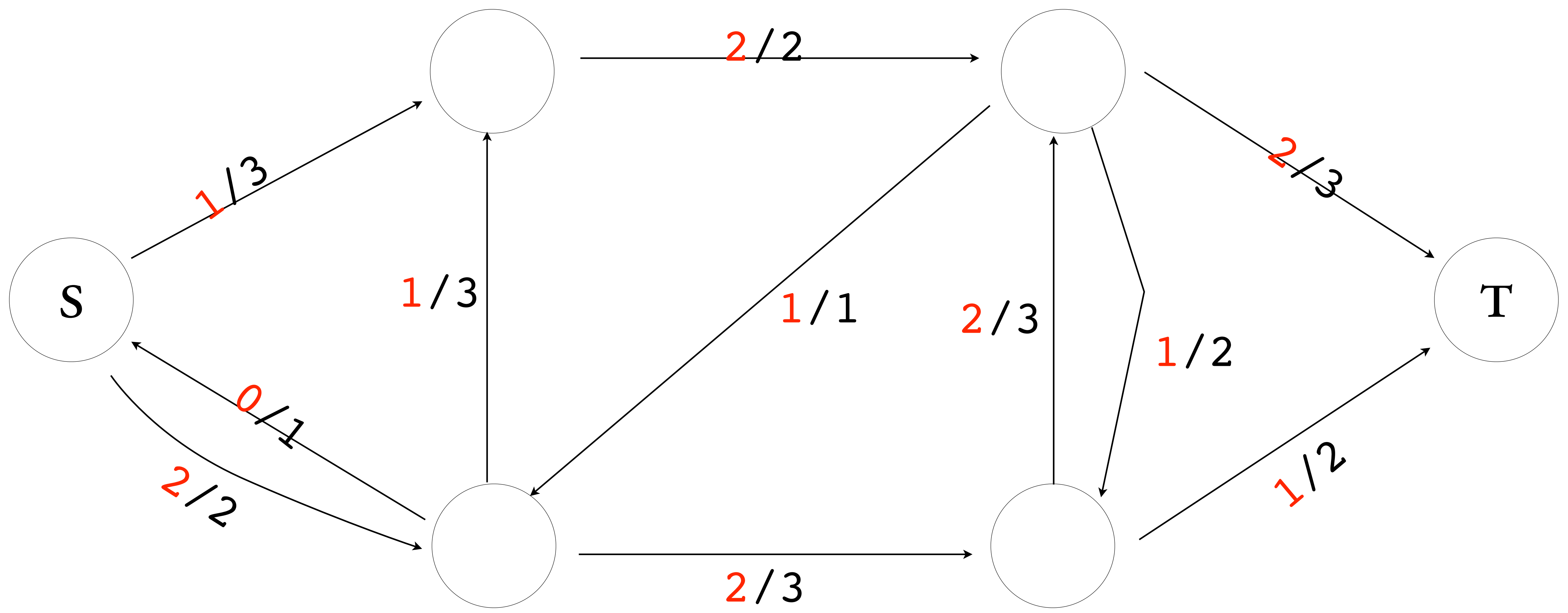
A <u>FLOW</u> IS A MAP FROM EDGES TO NUMBERS:

CAPACITY CONSTRAINT:

FLOW CONSTRAINT:

$$|f| =$$

example



max flow problem

Given a graph $G = (V, E)$ and capacities $c : E \to \mathbb{N}$, compute

max flow problem

Given a graph $G = (V, E)$ and capacities $c : E \to \mathbb{N}$, compute

$$\text{argmax}_f |f|$$

i.e., the maximum flow over all valid flows.

greedy solution?

hundreds of applications

bipartite matching
edge-disjoint paths
node-disjoint paths
scheduling
baseball elimination
resource allocations

will discuss many of these applications soon

Algorithms for max flow

Residual graphs

$$G_f = (V, E_f)$$

Residual graphs

$G_f = (V, E_f)$ A graph derived from G and a valid flow f.

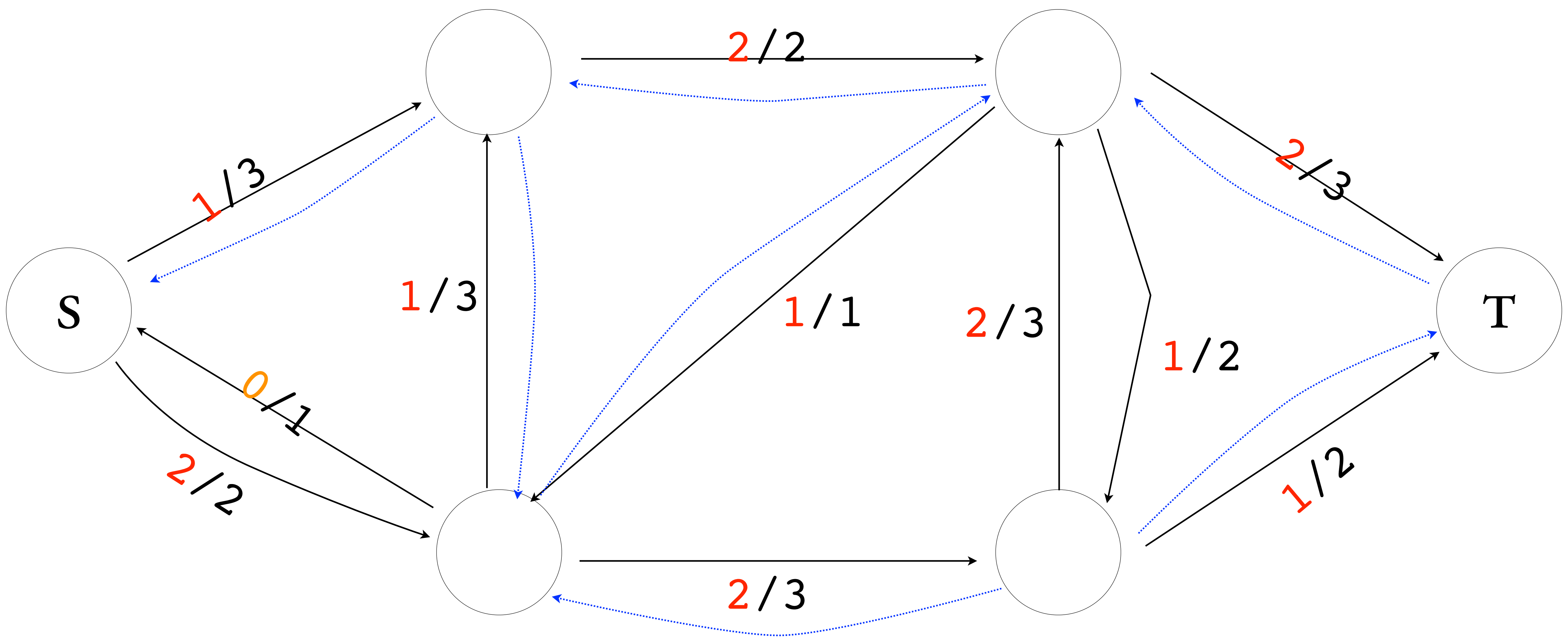
Residual graphs

$$G_f = (V, E_f)$$

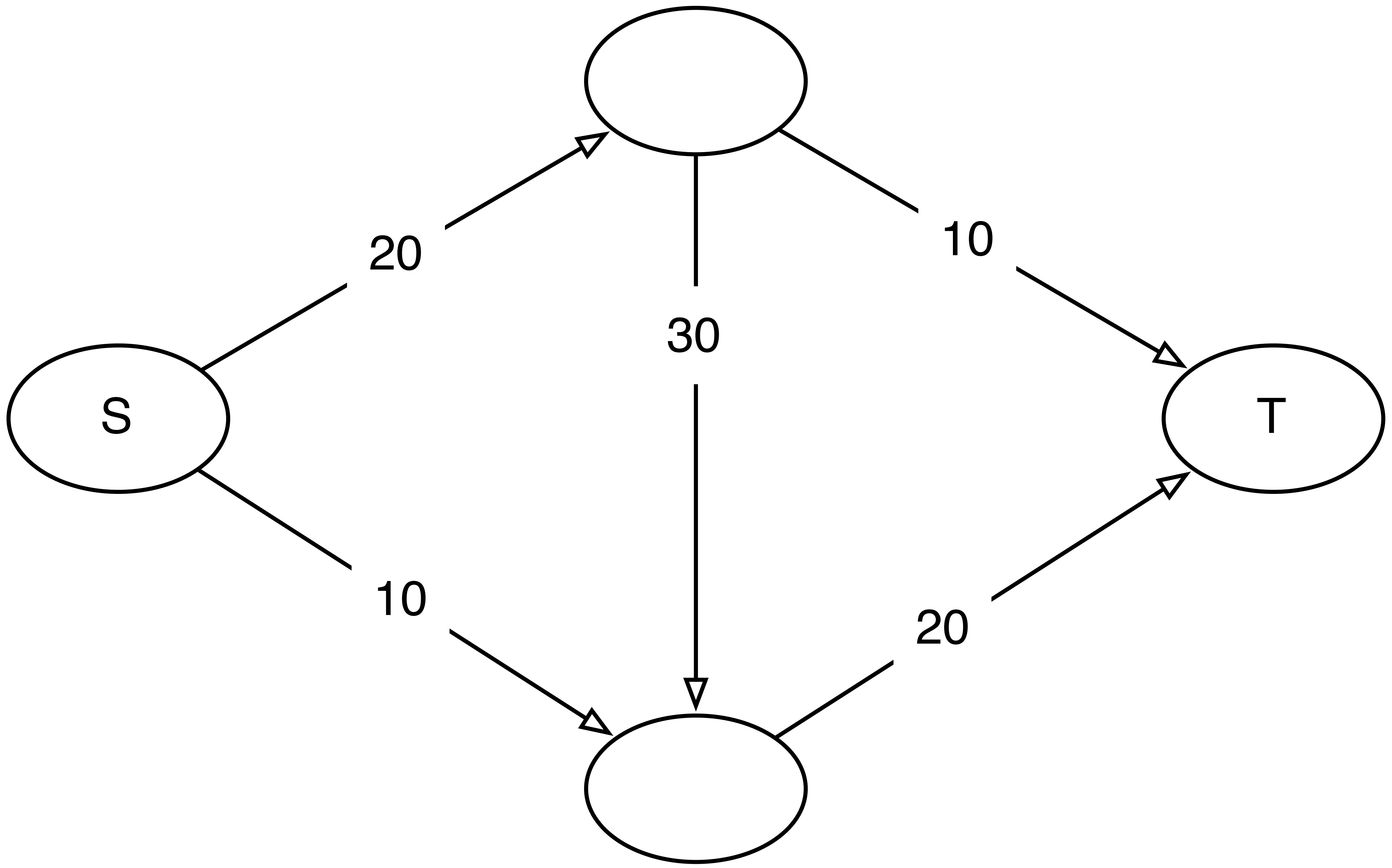
A graph derived from G and a valid flow f.

Same vertices, but difference edges:

example residual graph



why residual graphs ?



augmenting paths

DEF:

augmenting paths

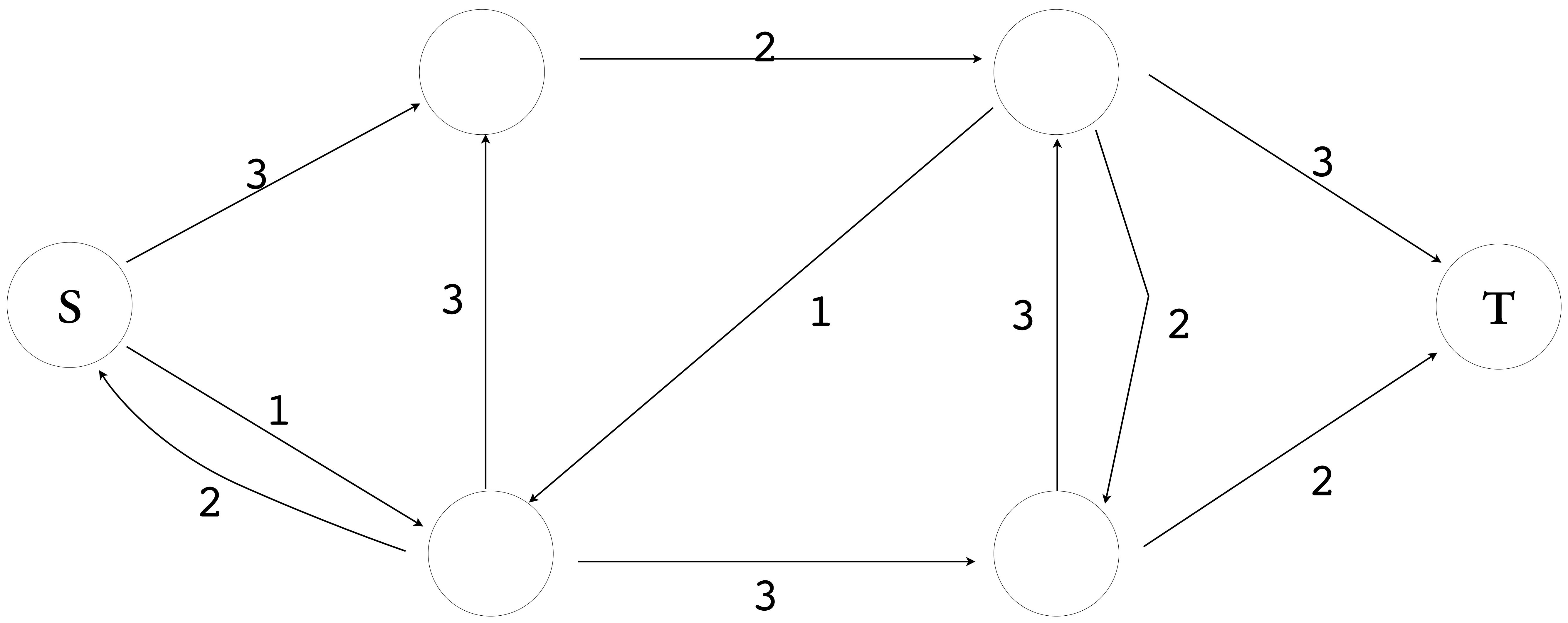
DEF: A path from s to t in the residual graph G_f.

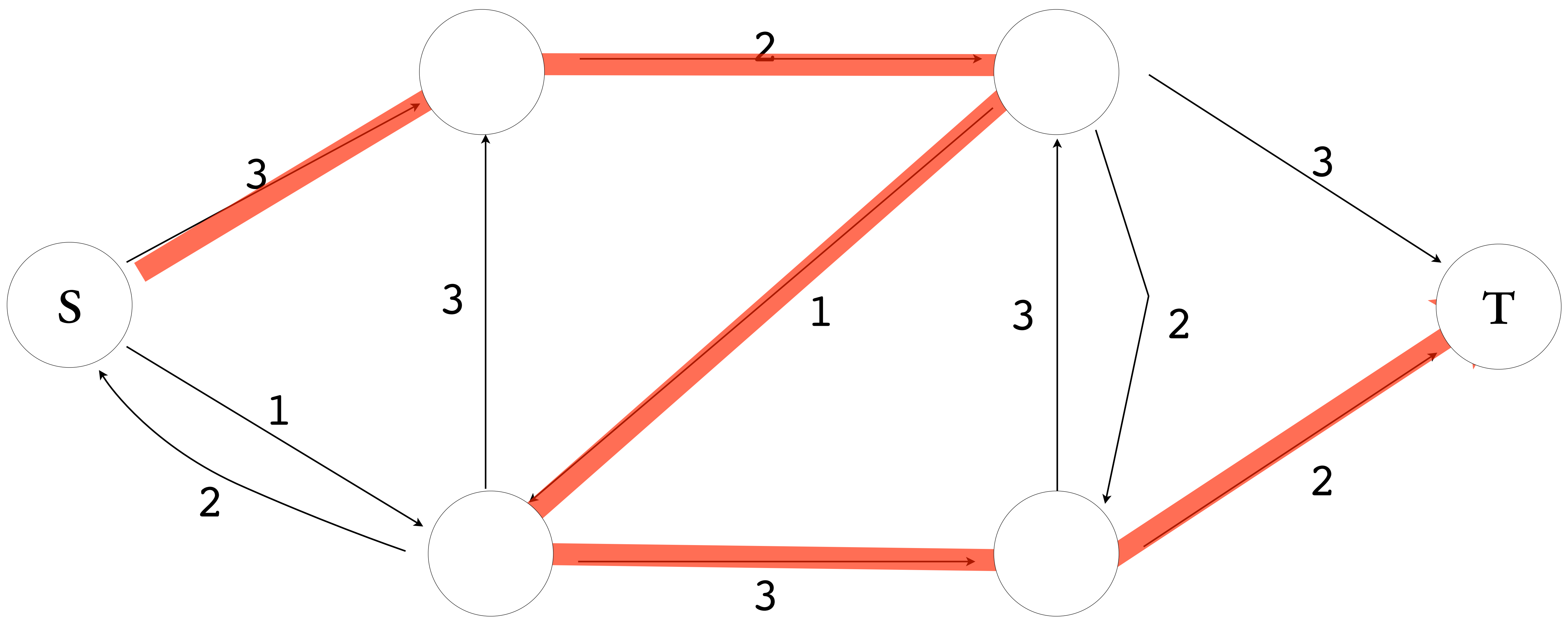
Ford-Fulkerson

INITIALIZE $f(u,v) \leftarrow 0 \; \forall u,v$

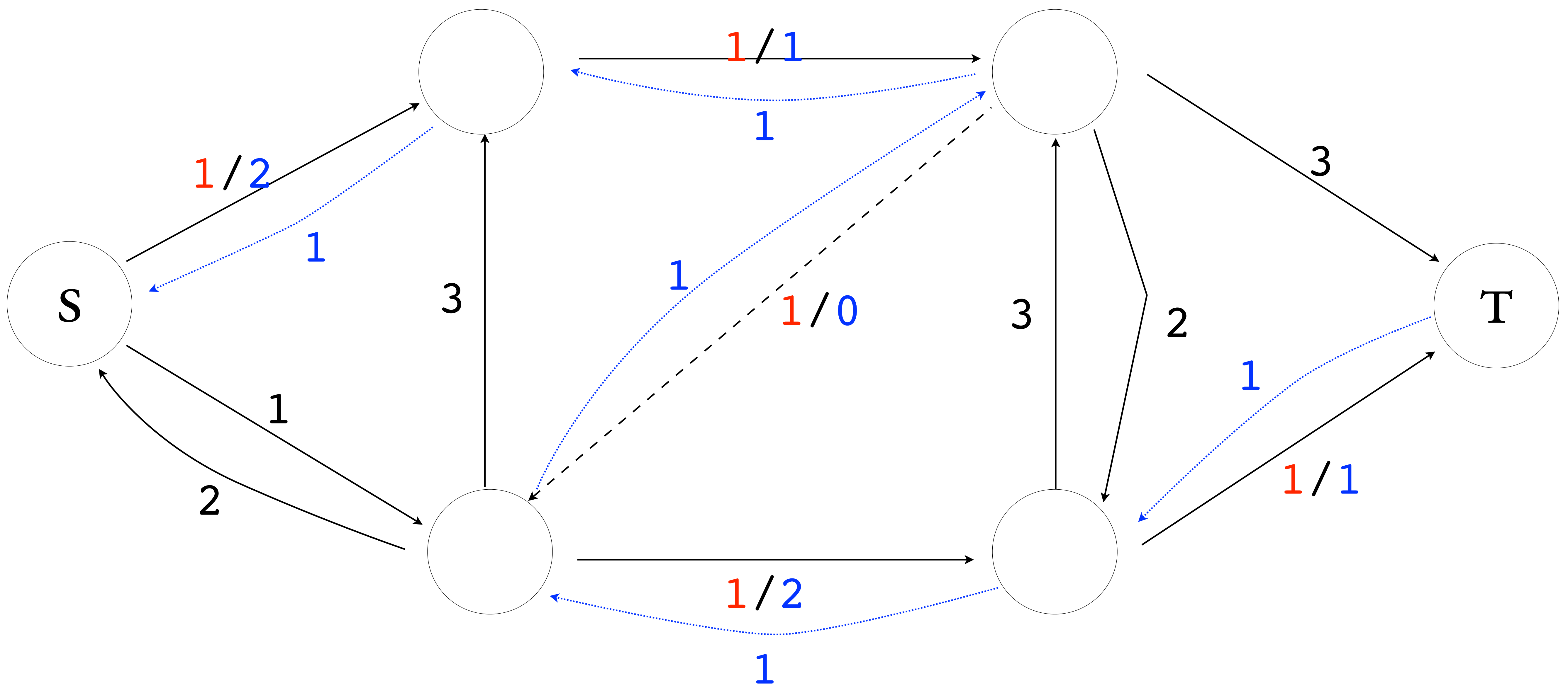
WHILE EXISTS AN AUGMENTING PATH p IN G_f

AUGMENT f WITH $c_f(p) = \min_{(u,v)\in p} c_f(u,v)$

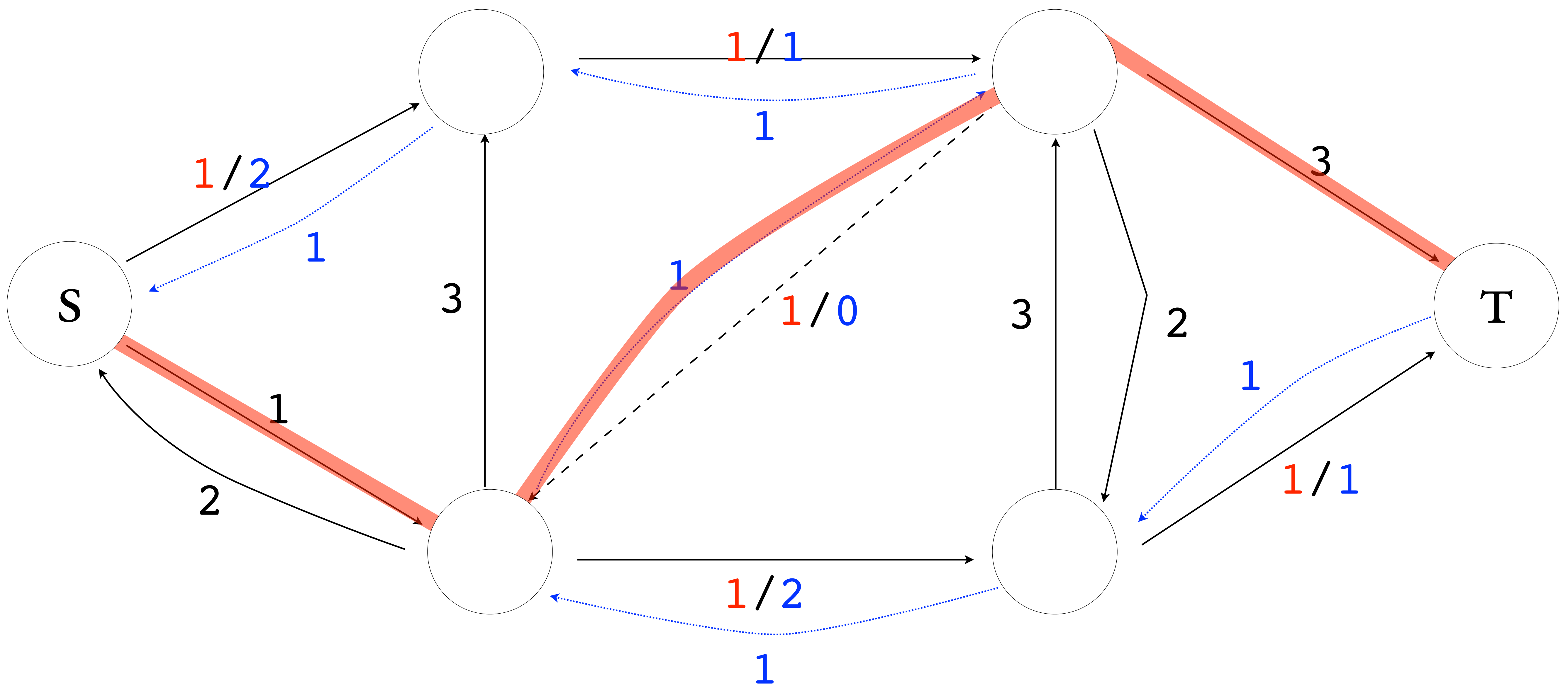

S
T
3
1
2
3
2
1
3
3
2
3
2
3



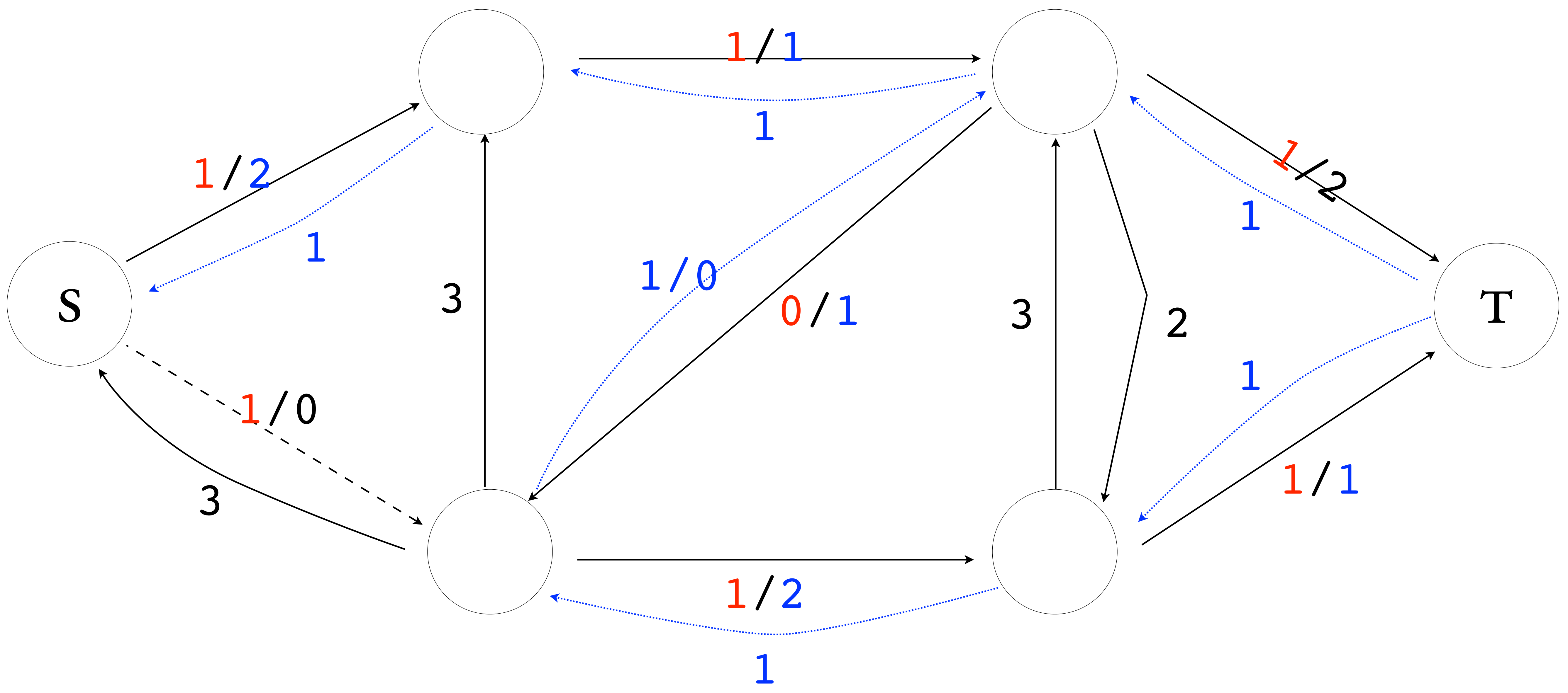
S
T
3
2
3
1
2
1
3
3
3
2
3
2

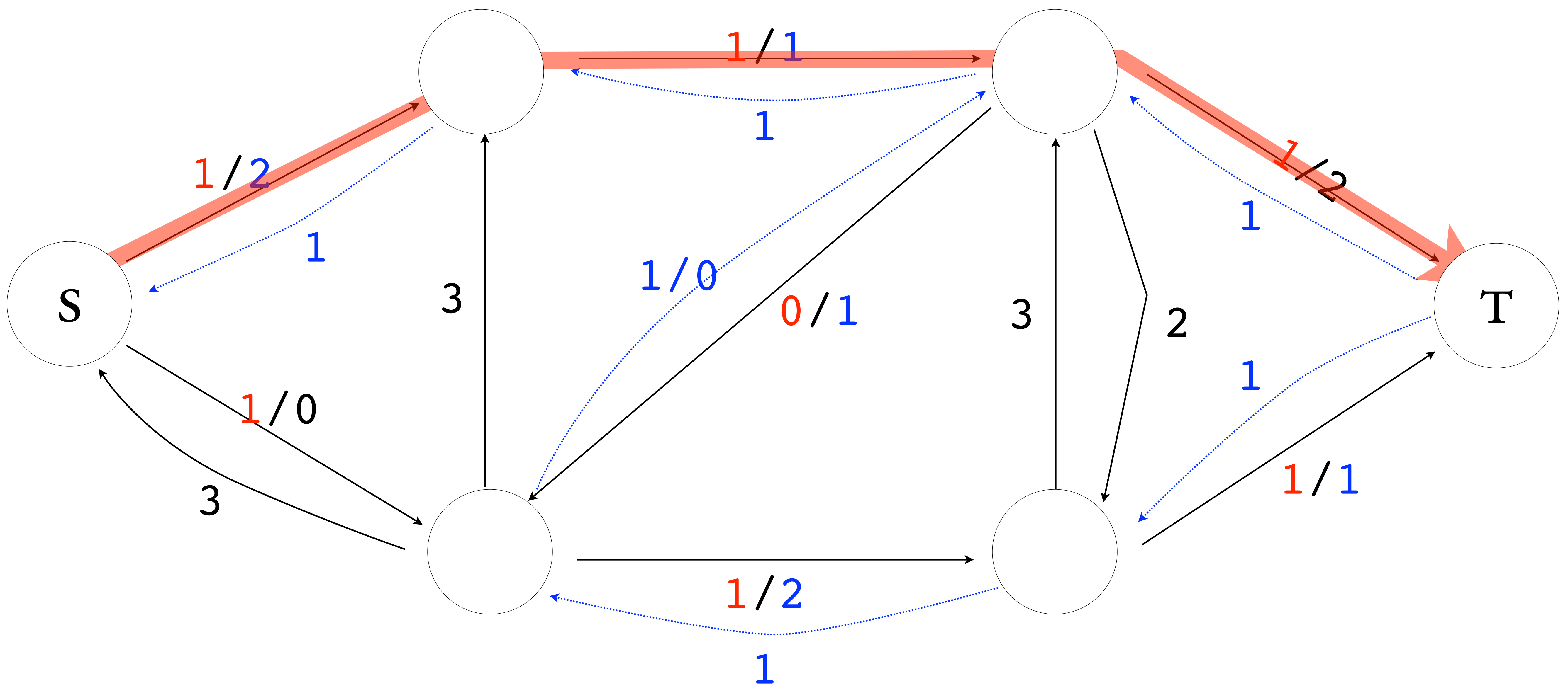


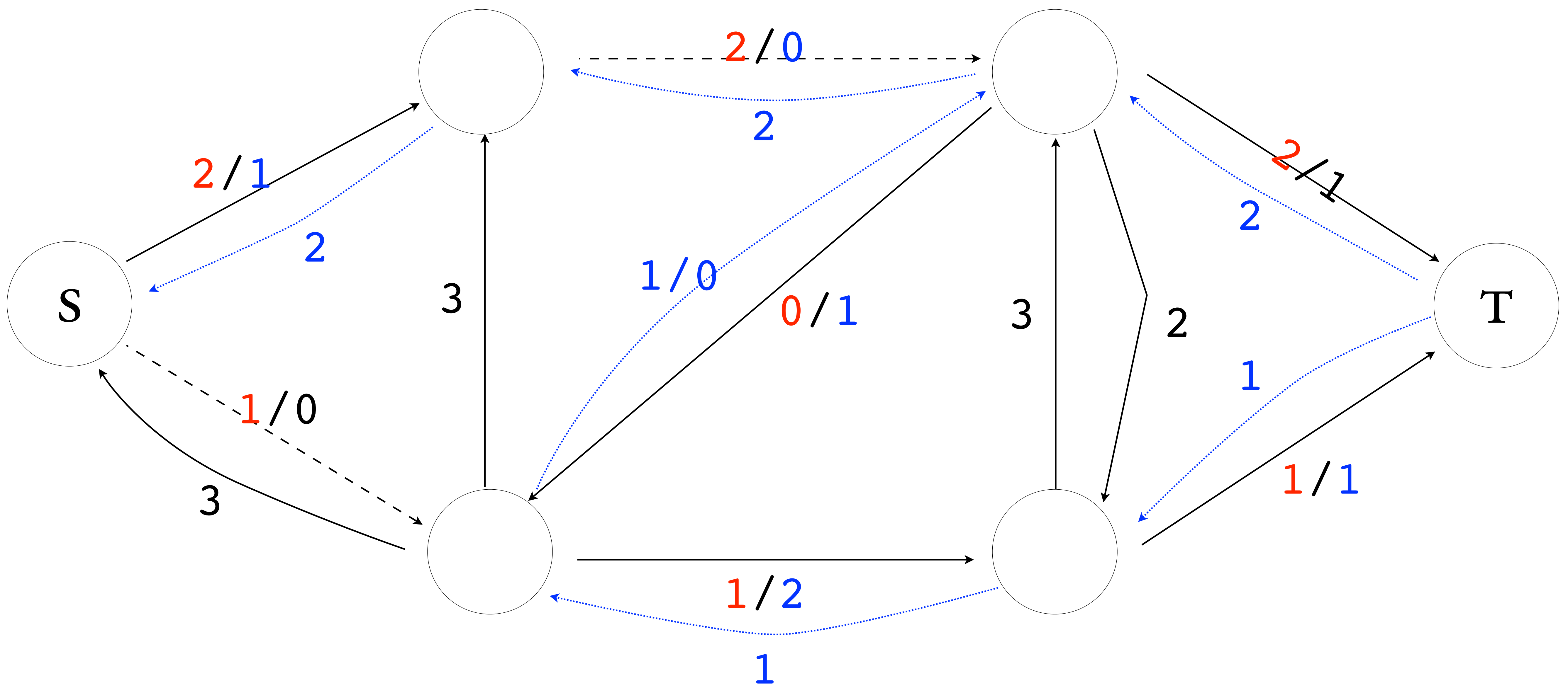
S
T
1 / 2
1
1 / 1
1
3
1
1 / 0
3
3
2
1
1 / 1
1
2
1 / 2
1



S
T
1 / 2
1
3
1
1
2
1 / 1
1
1 / 0
3
2
3
1
1 / 1
1 / 2
1


S
T
1/2
1
1/1
1
3
1/0
0/1
1/2
1
3
2
1
1
1/1
1/0
3
1/2


S
T
1/2
1
1/1
1
1/2
1
3
1/0
0/1
3
2
1
1/0
3
1/1
1/2
1



S
T
2/1
2
1/0
3
3
2/0
2
1/0
0/1
1/2
1
3
2
2/1
2
1
1/1

FORD-FULKERSON

INITIALIZE $f(u,v) \leftarrow 0 \; \forall u,v$

WHILE EXISTS AN AUGMENTING PATH p IN G_f

AUGMENT f WITH $c_f(p) = \min_{(u,v)\in p} c_f(u,v)$

TIME TO FIND AN AUGMENTING PATH:

NUMBER OF ITERATIONS OF WHILE LOOP:

Cuts

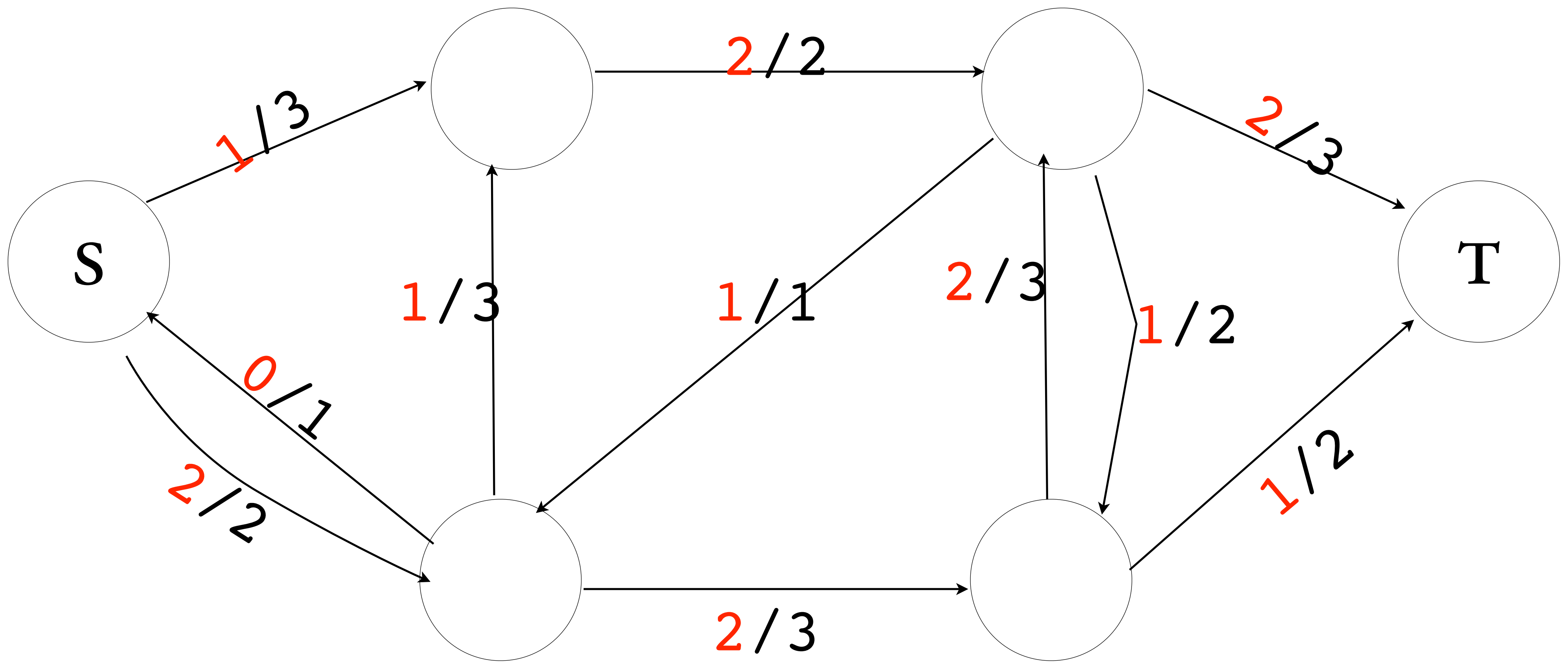
Def of a cut:

cost of a cut:

$$||S, T|| =$$

lemma: [min cut] for any $f, (S, T)$

FOR ANY $f, (S, T)$ IT HOLDS THAT $|f| \leq ||S, T||$

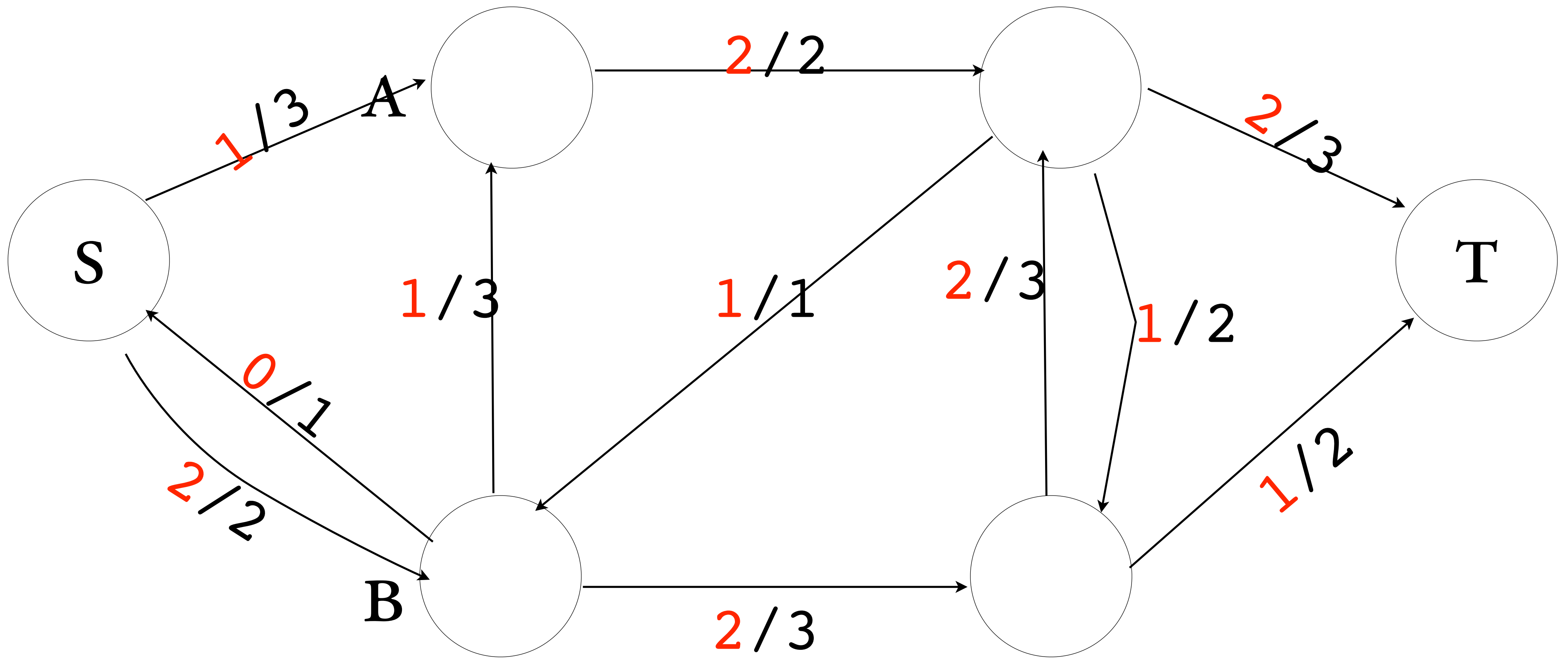


EXAMPLE:

A property to remember

FOR ANY $f, (S,T)$ IT HOLDS THAT $|f| \leq ||S,T||$

PROOF:

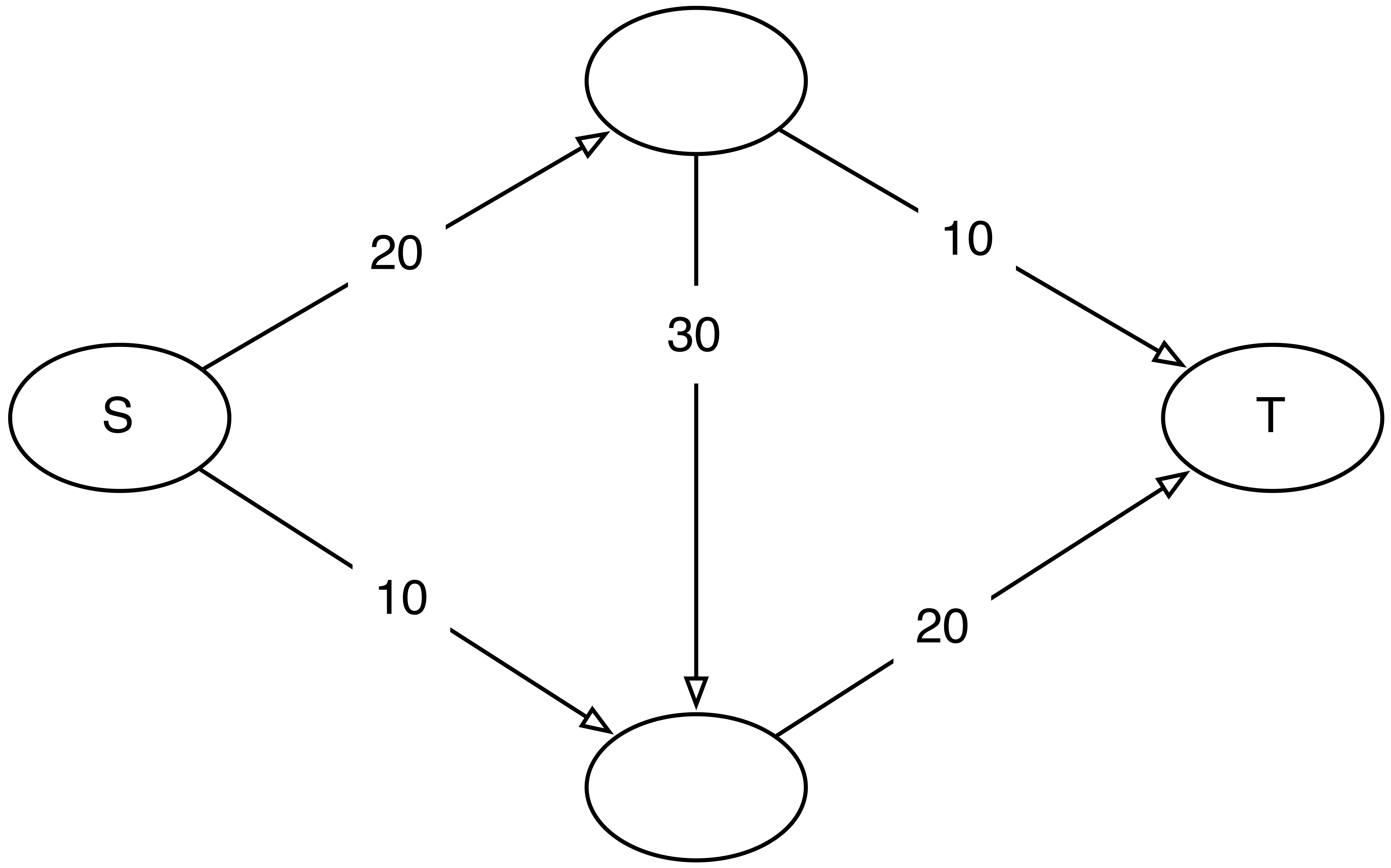


S
A
B
T
1/3
2/2
2/3
1/3
1/1
2/3
1/2
0/1
2/2
2/3
1/2

FOR ANY $f, (S, T)$ IT HOLDS THAT $|f| \leq ||S, T||$

(FINISHING PROOF)

why residual graphs ?



augmenting paths

DEF:

Thm: max flow = min cut

$$\max_{f} |f| = \min_{S,T} ||S,T||$$

If f is a max flow, then Gf has no augmenting paths.

thm: max flow = min cut

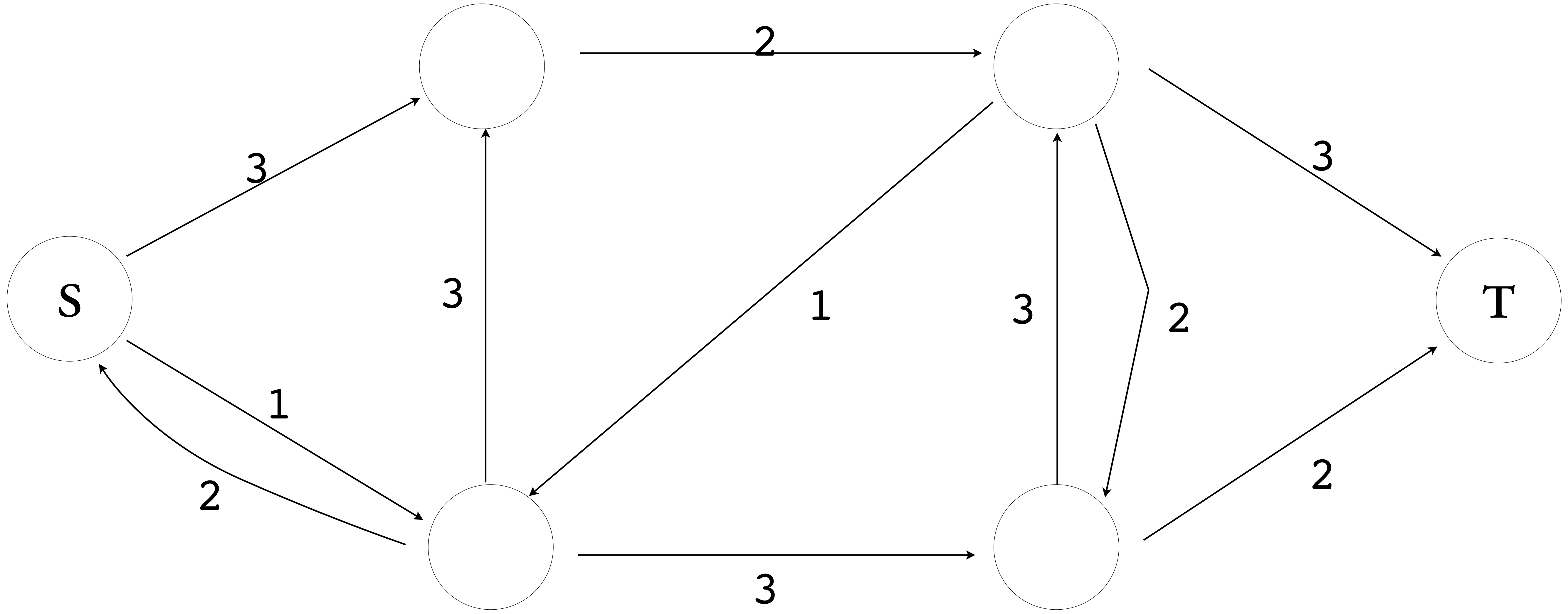
$$\max_{f} |f| = \min_{S,T} ||S,T||$$

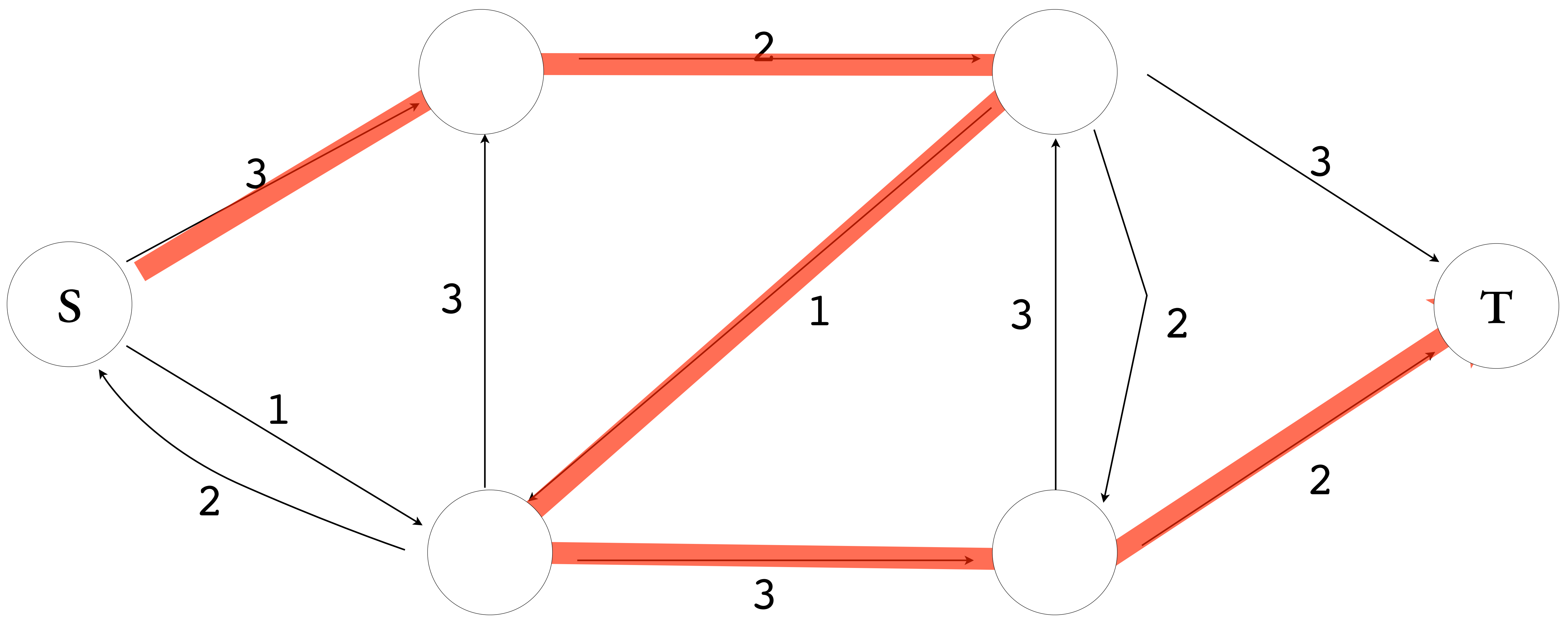
ford-fulkerson

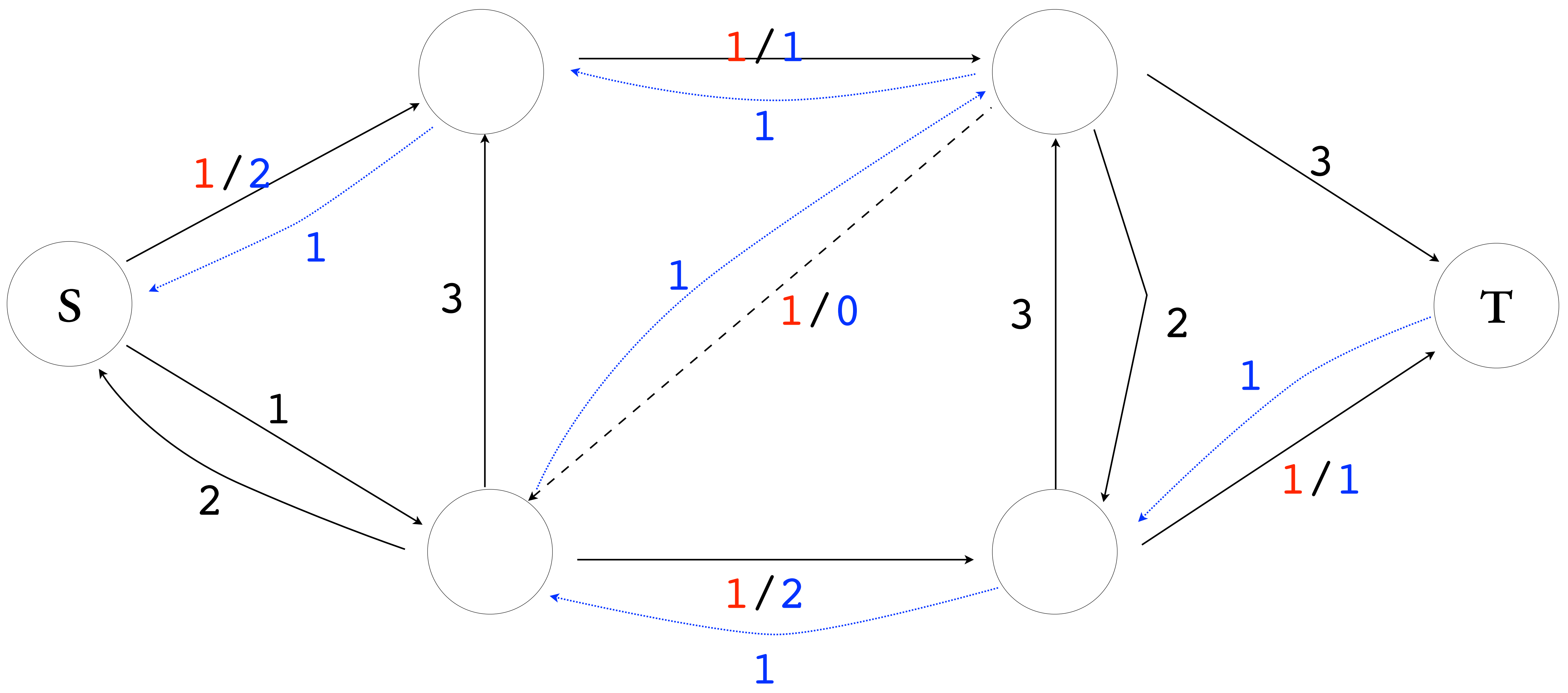
INITIALIZE $f(u,v) \leftarrow 0 \;\forall u,v$

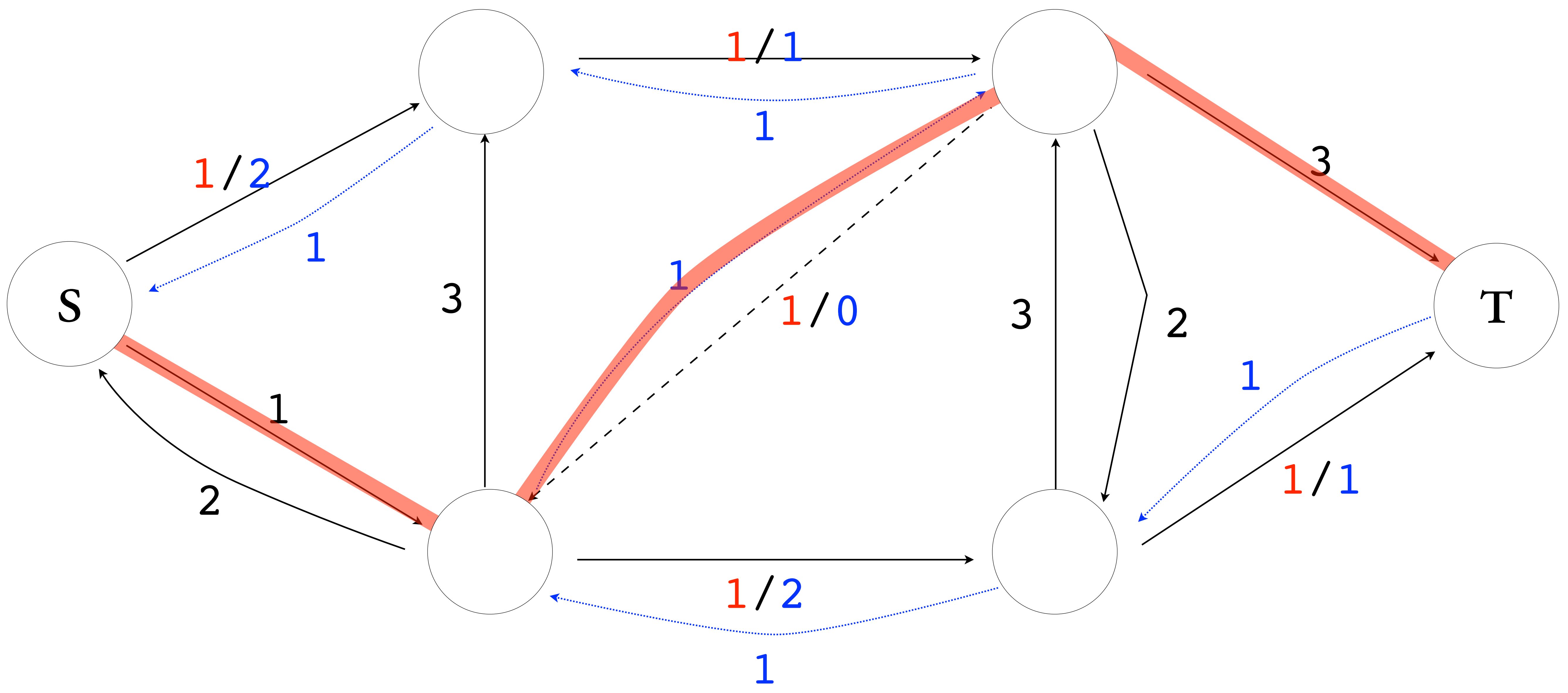
WHILE EXISTS AN AUGMENTING PATH p IN G_f

AUGMENT f WITH $c_f(p) = \min_{(u,v)\in p} c_f(u,v)$

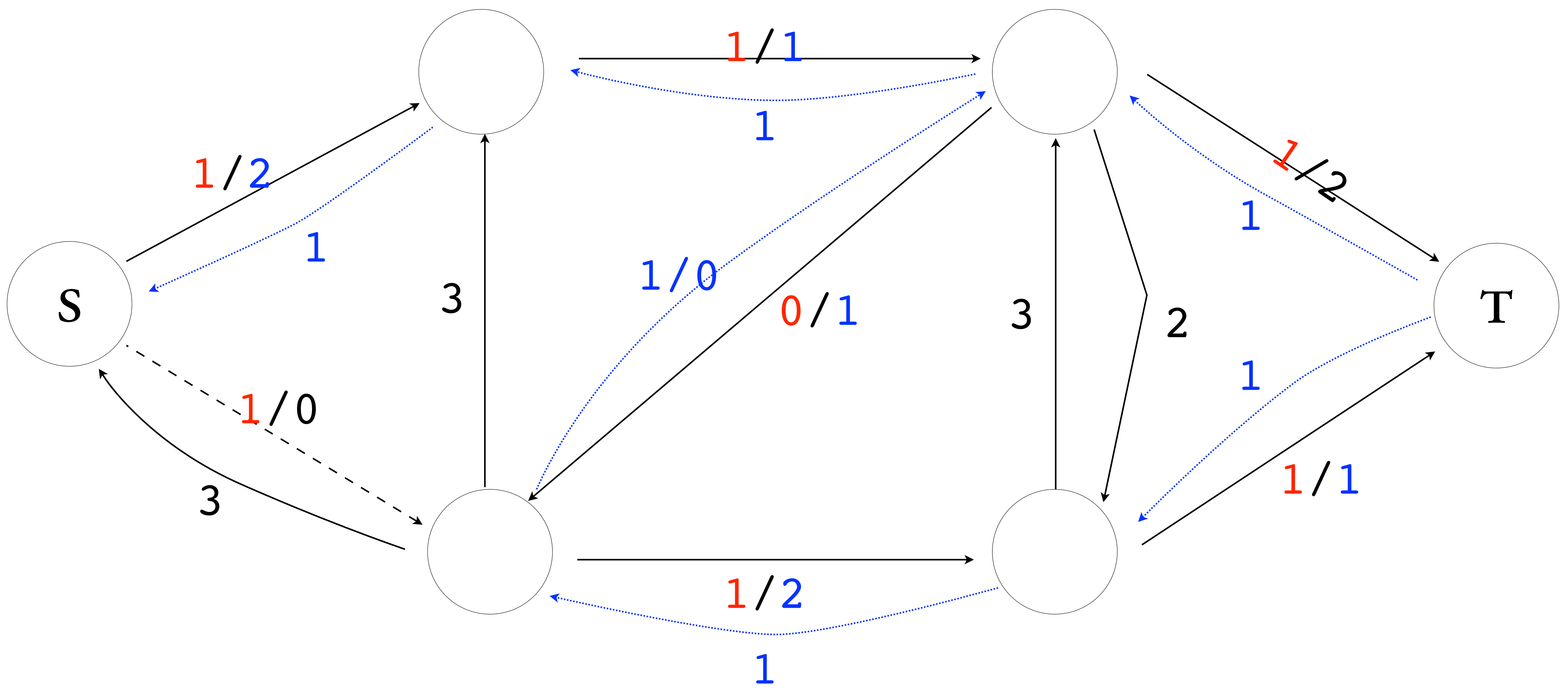

S
T
3
1
2
3
2
1
3
3
2
3
2

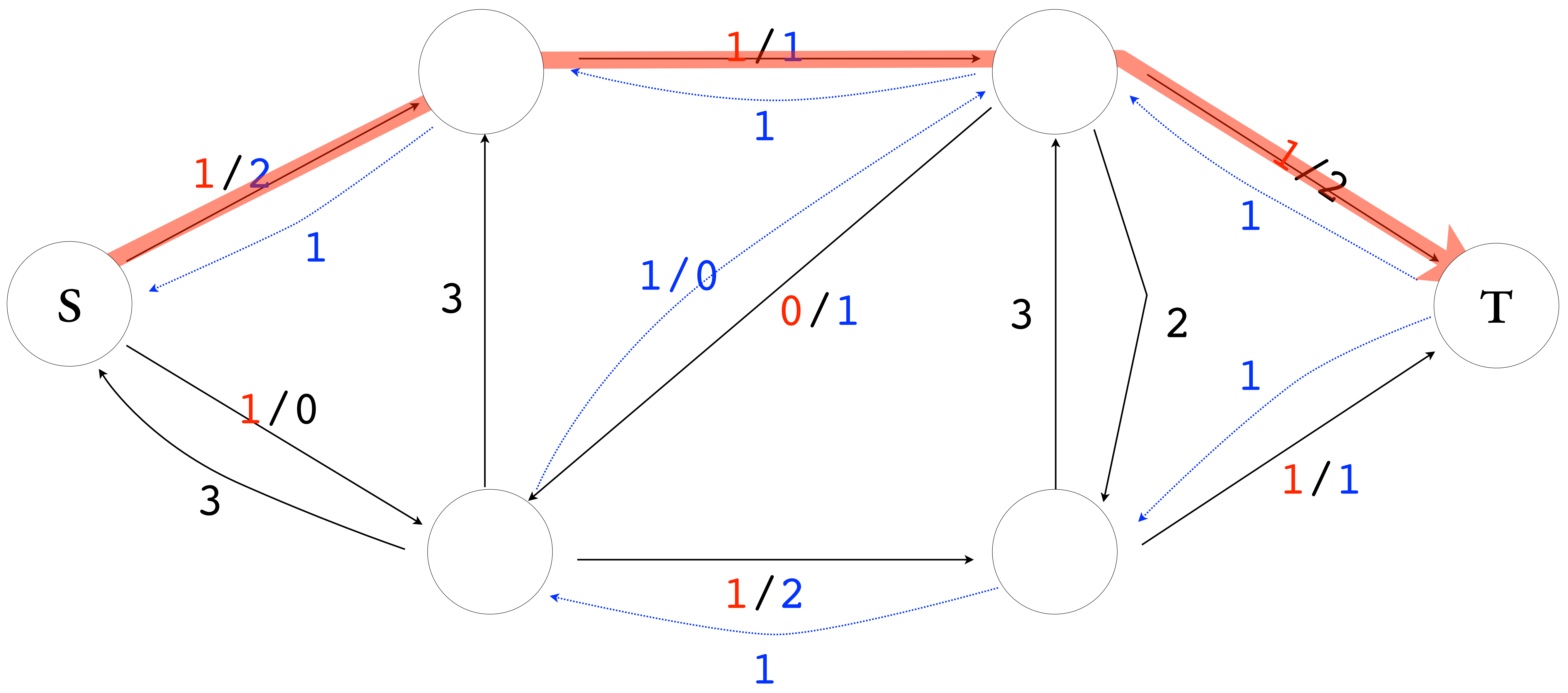

S
T
3
2
3
1
2
1
3
3
2
3
3
2

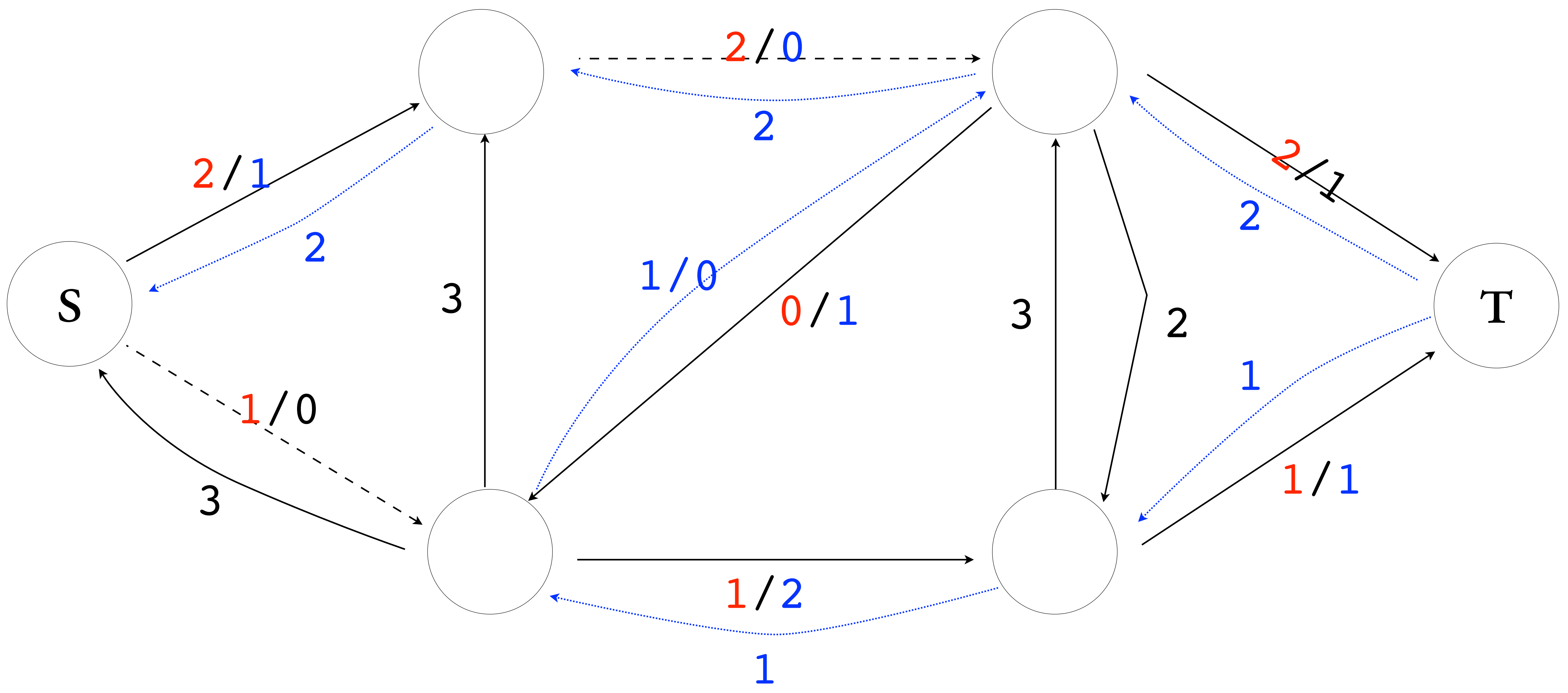

1 / 1
1
3
1 / 2
1
S
3
1
1 / 0
3
2
T
1
1
1 / 1
2
1 / 2
1



S
T
1 / 2
1
3
1 / 1
1
1
1 / 0
3
3
2
1
2
1 / 2
1
1
1 / 1


S
T
1/2
1
1/1
1
3
1/0
0/1
3
2
1/2
1
1
1/0
3
1/2
1
1/1


S
T
1/2
1
1/1
1
1/2
1
3
1/0
0/1
3
2
1
1
1/0
3
1/1
1/2
1



S
T
2/0
2
2/1
2
2/1
2
3
1/0
0/1
3
2
1
1/0
3
1/2
1
1/1

FORD-FULKERSON

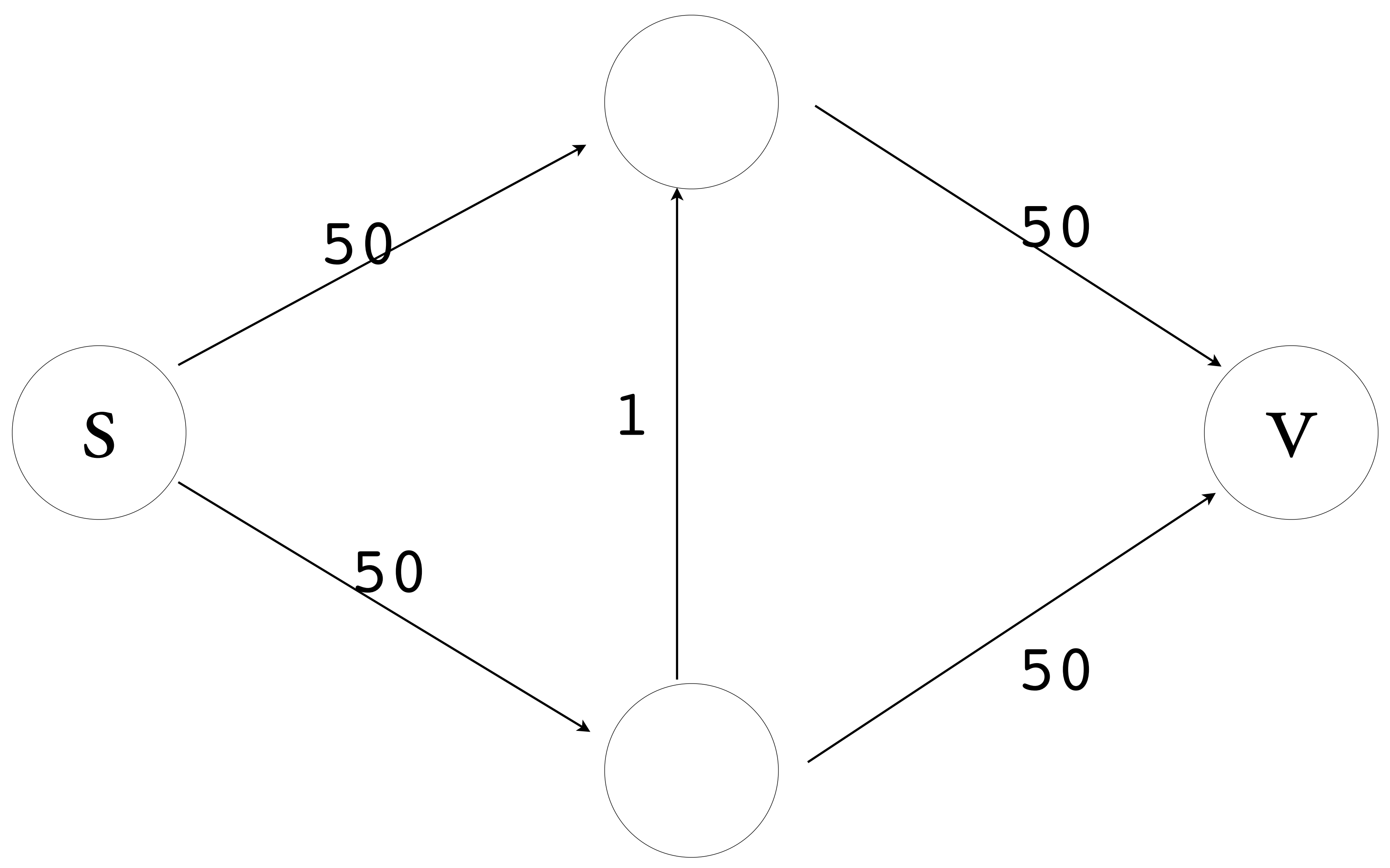
INITIALIZE $f(u,v) \leftarrow 0 \ \forall u,v$

WHILE EXISTS AN AUGMENTING PATH p IN G_f

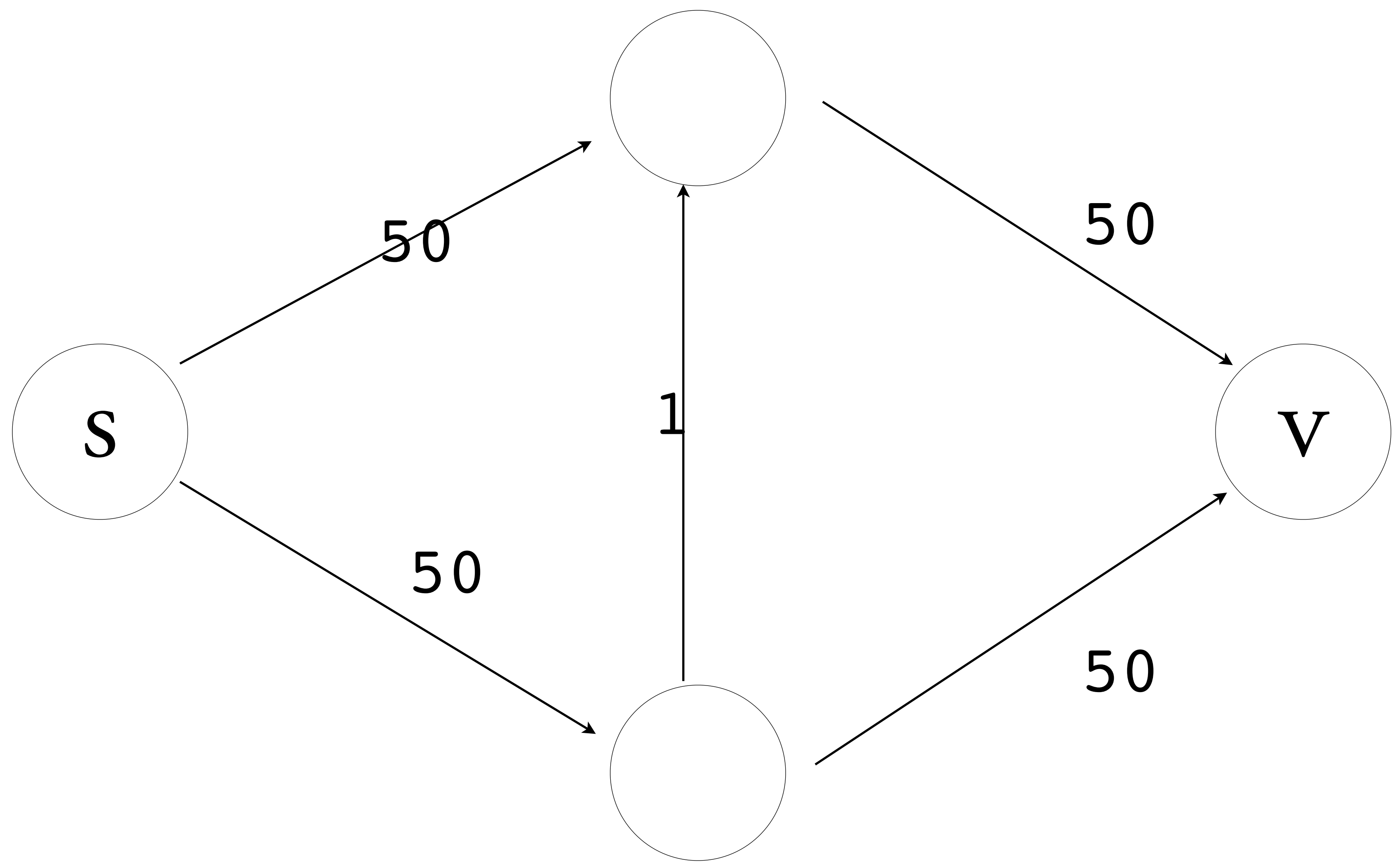
AUGMENT f WITH $c_f(p) = \min_{(u,v)\in p} c_f(u,v)$

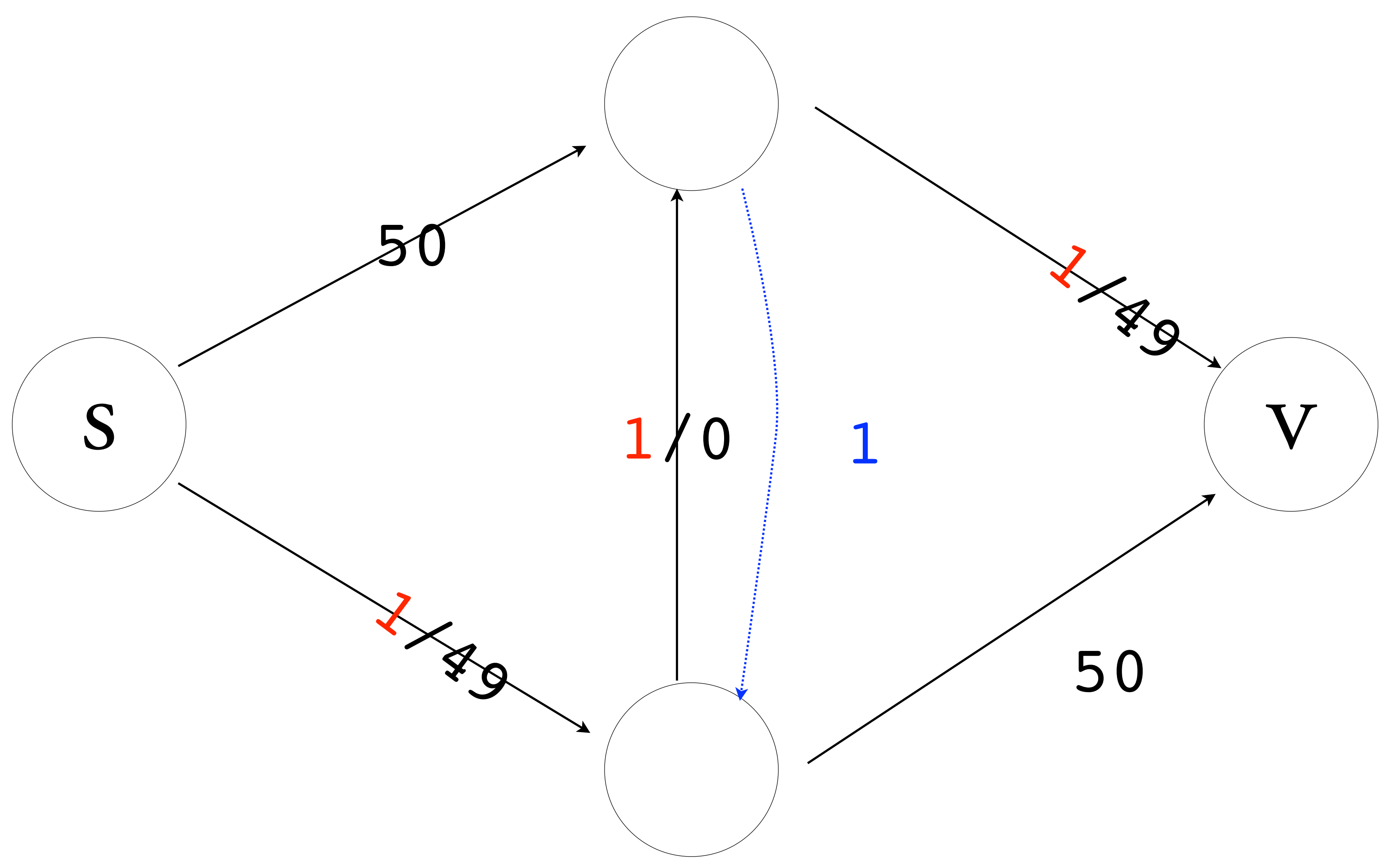
TIME TO FIND AN AUGMENTING PATH:

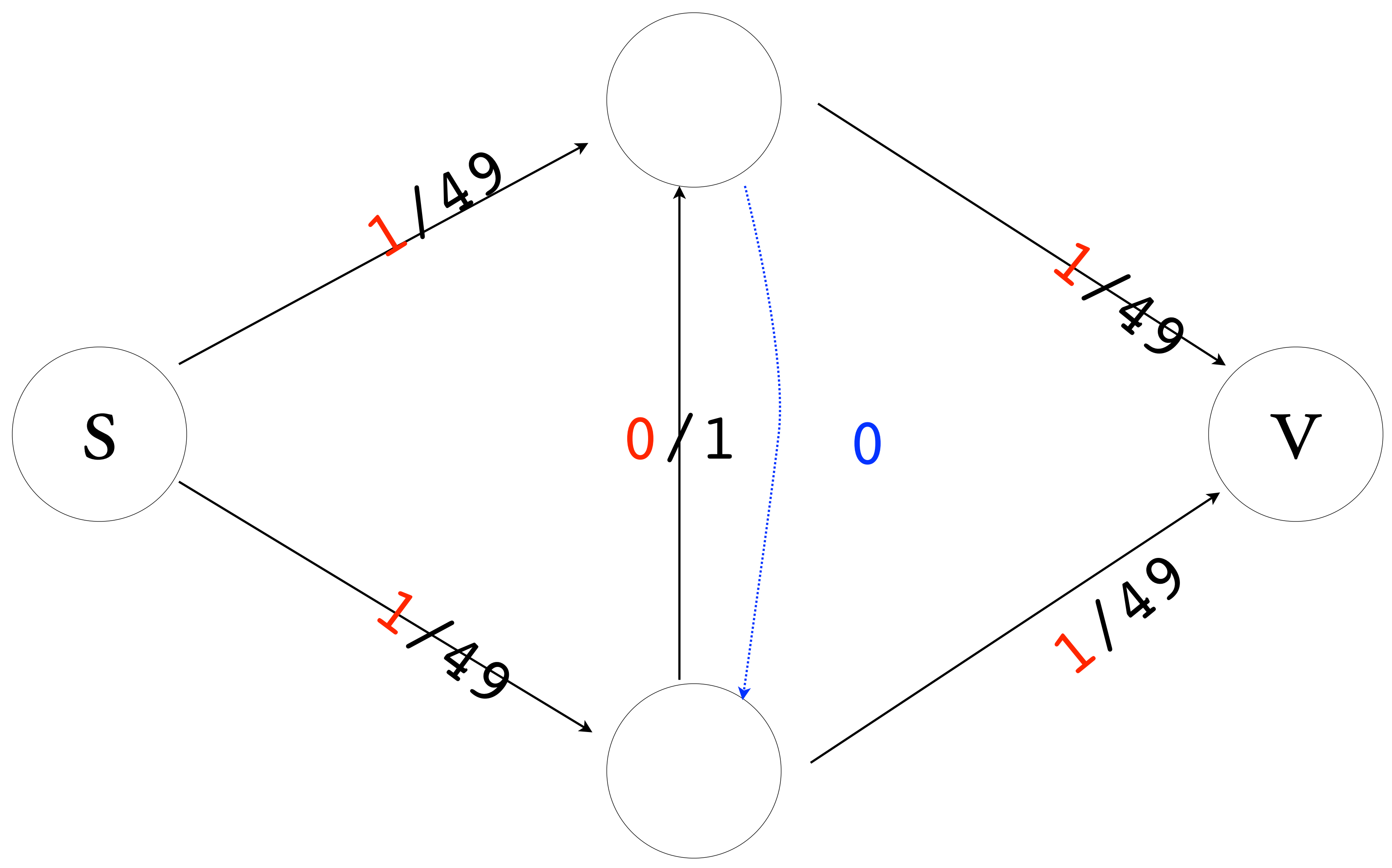
NUMBER OF ITERATIONS OF WHILE LOOP:



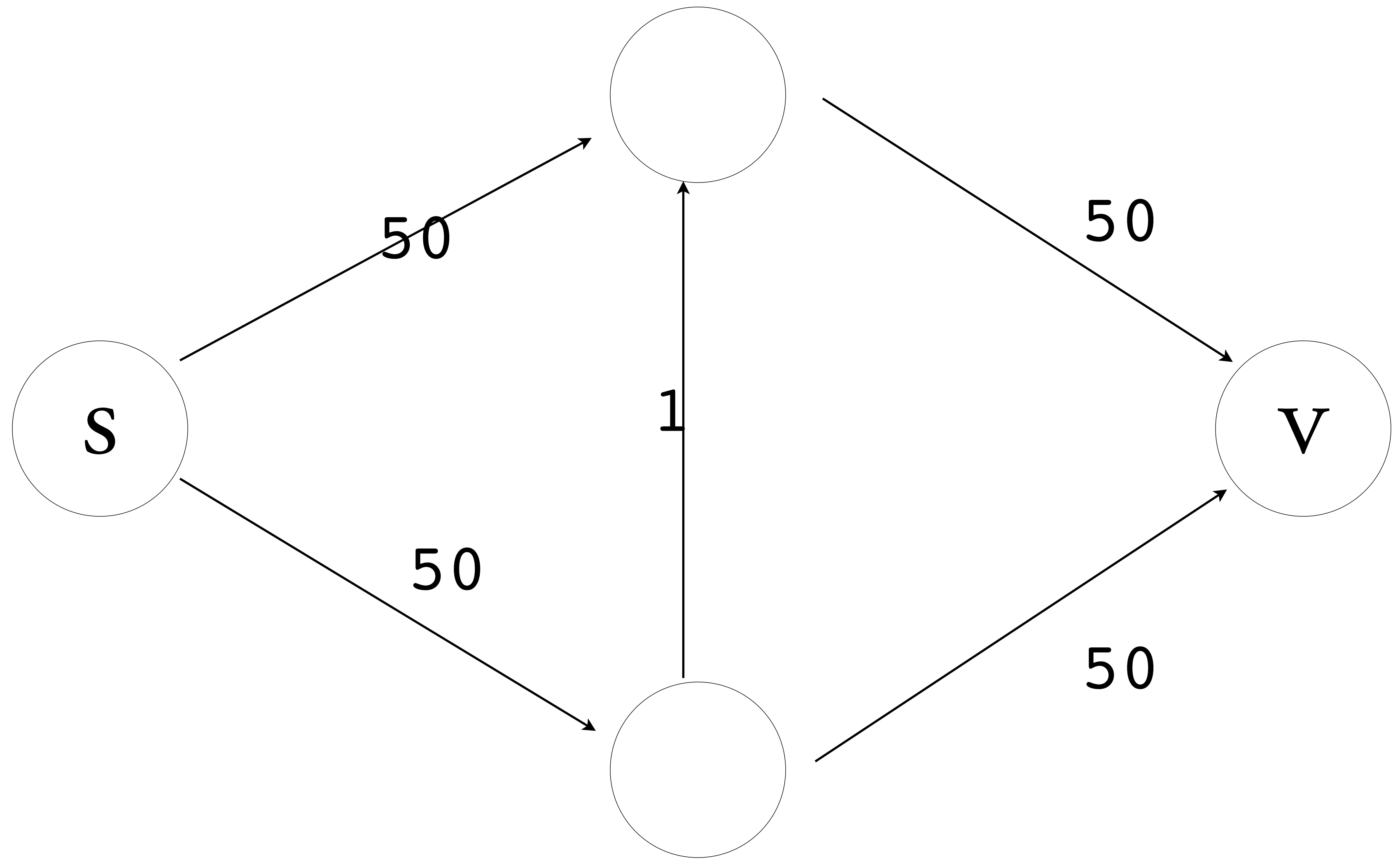
S
V
50
50
50
50
1


S
V
50
50
1
50
50


S
V
50
1/49
1/0
1
1/49
50


1/49
1/49
0/1
0
S
V
1/49
1/49

root of the problem



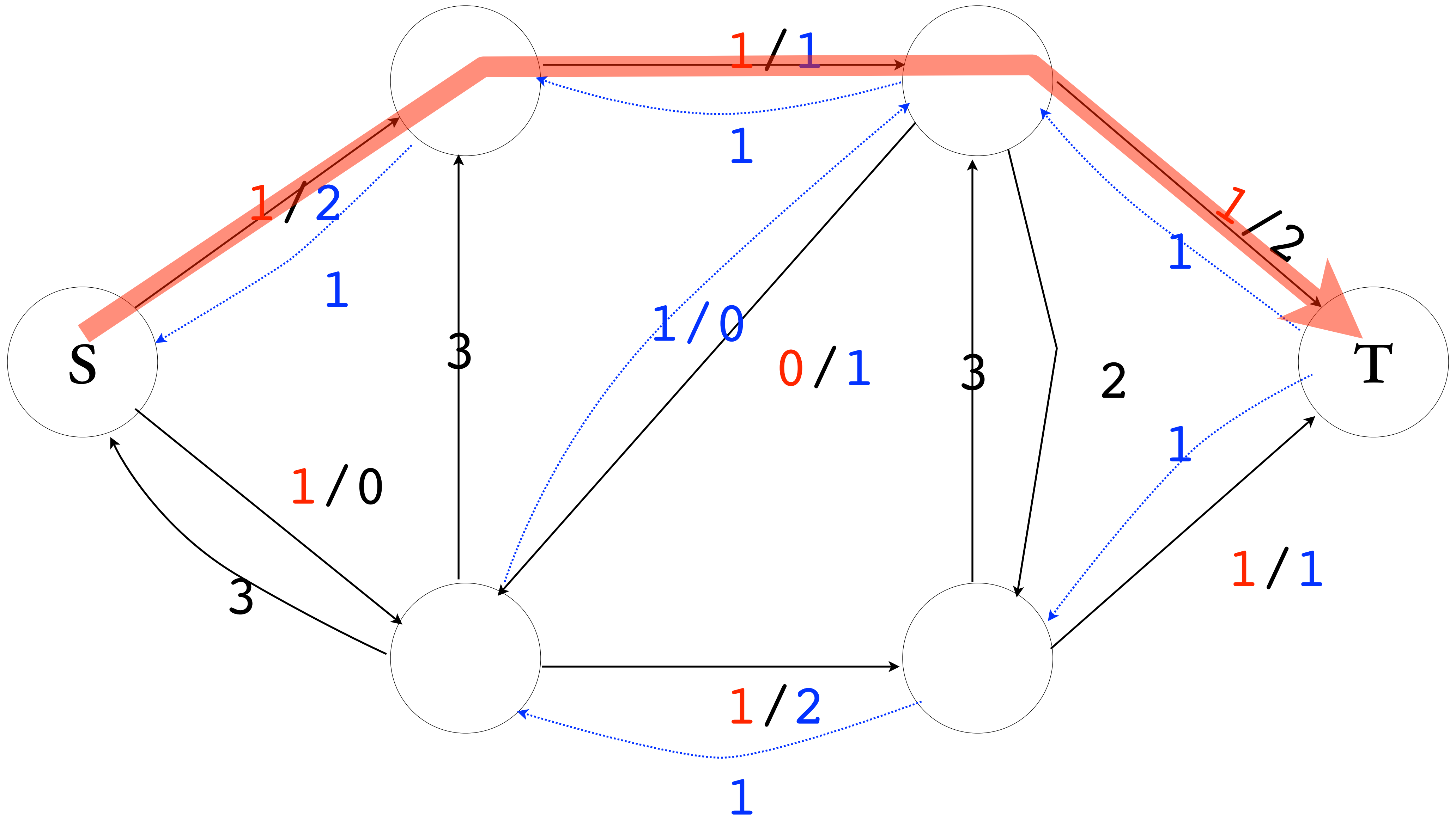
Edmonds-Karp 2

choose path with fewest edges first.

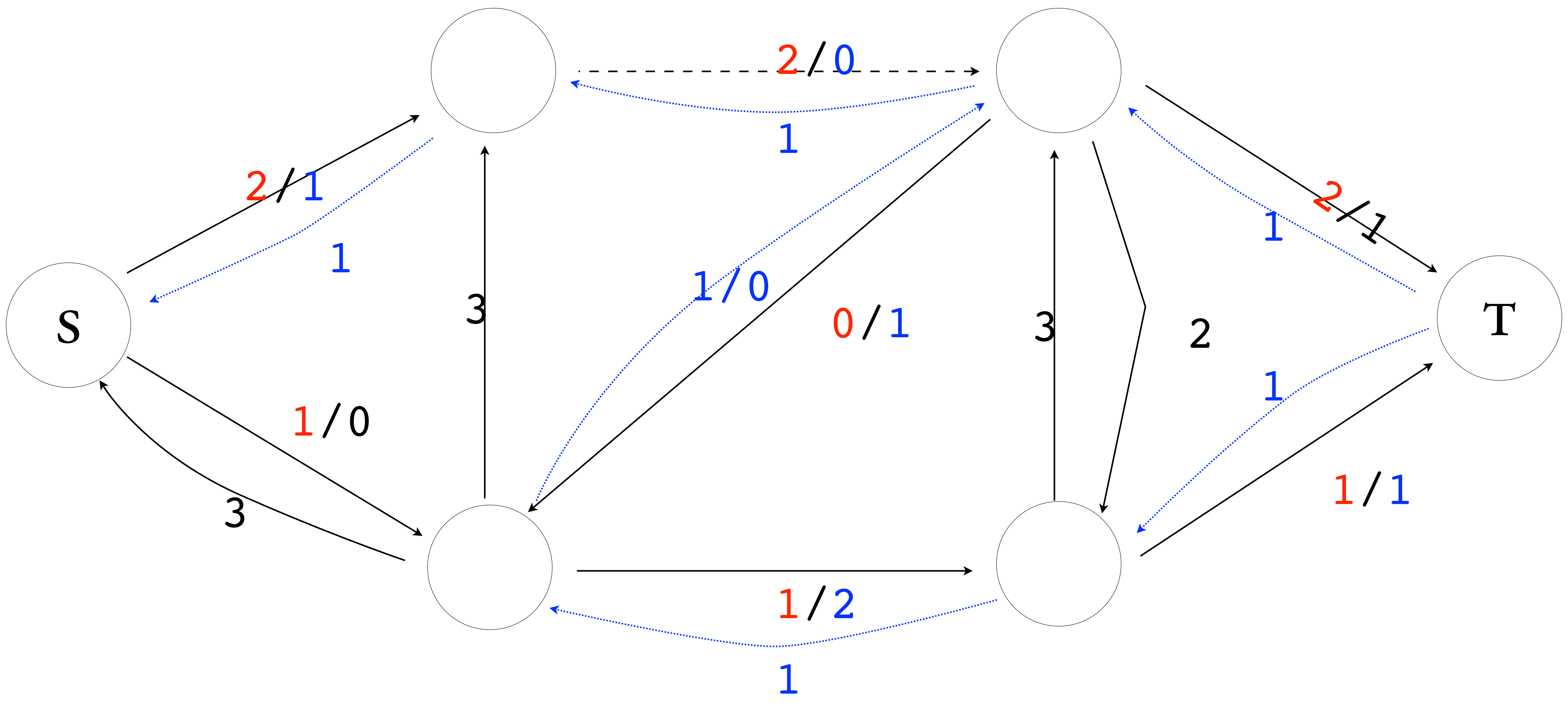
$\delta_f(s,v)$:

$\delta_f(s, v)$ increases monotonically thru exec

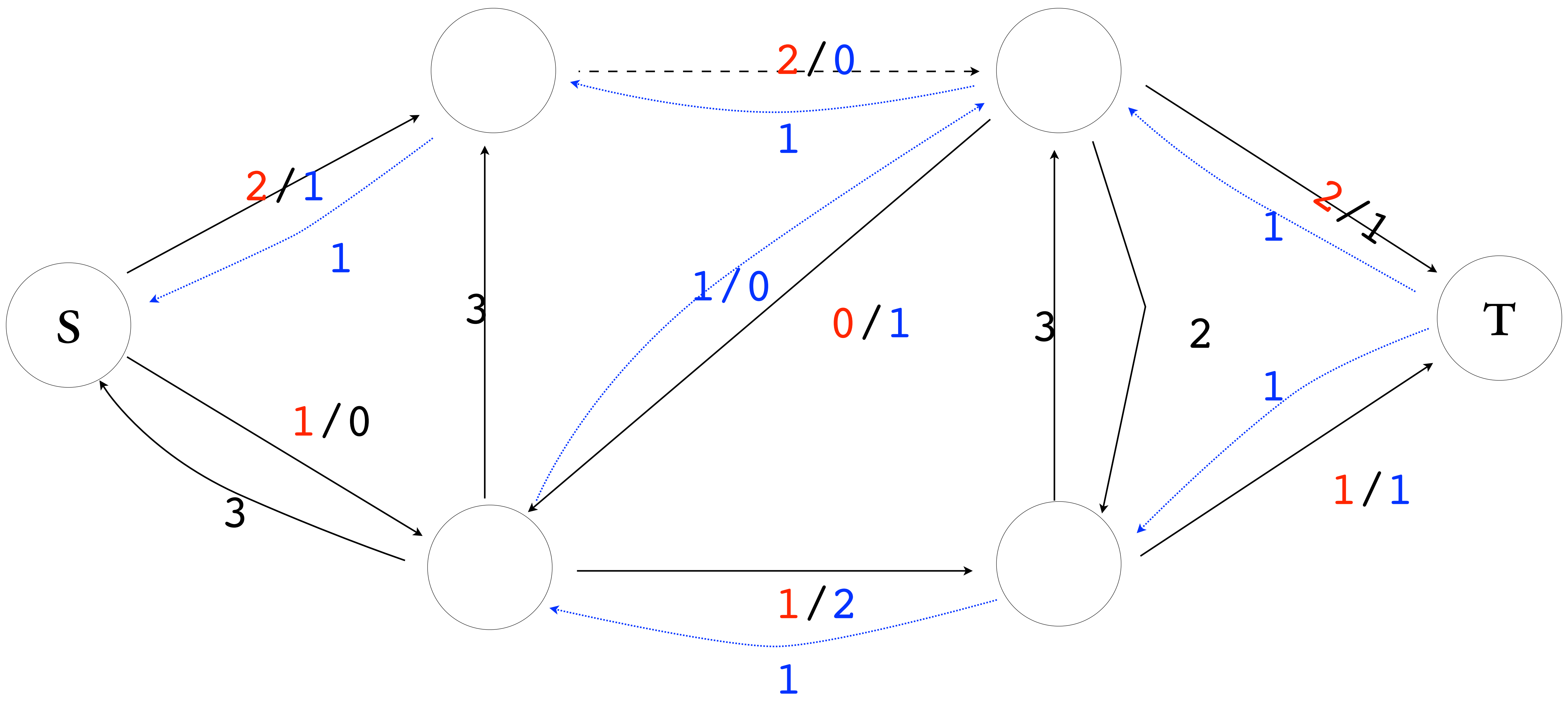
$$\delta_{i+1}(v) \geq \delta_i(v)$$



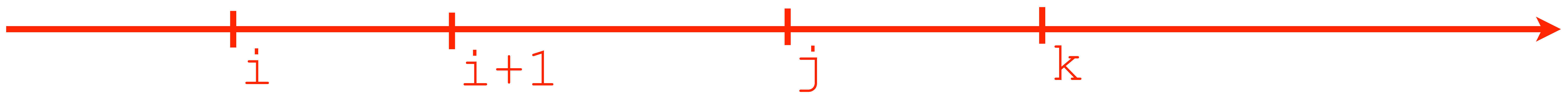
for every augmenting path, some edge is critical.

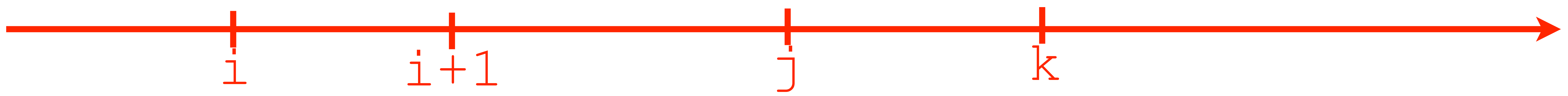


critical edges are removed in next residual graph.

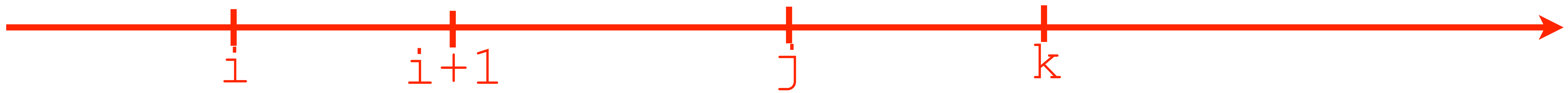


key idea: how many times can an edge be critical?

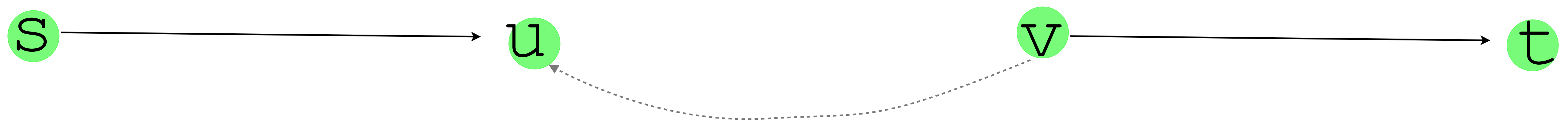

i
i+1
j
k



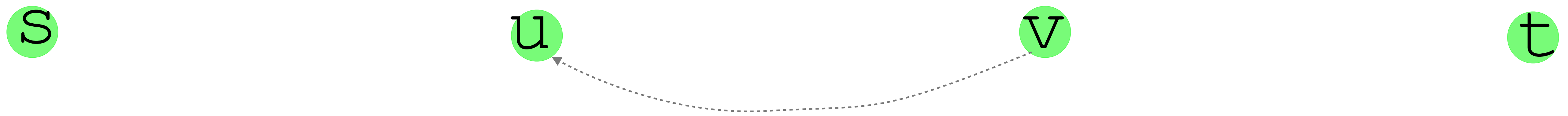
first time (u,v) is critical:

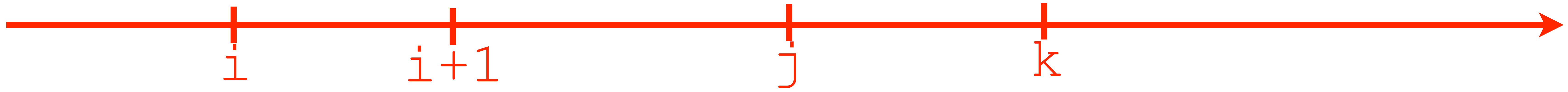


time i+1: (u,v) is critical: $\delta_{i+1}(s,v) \geq \delta_i(s,u) + 1$

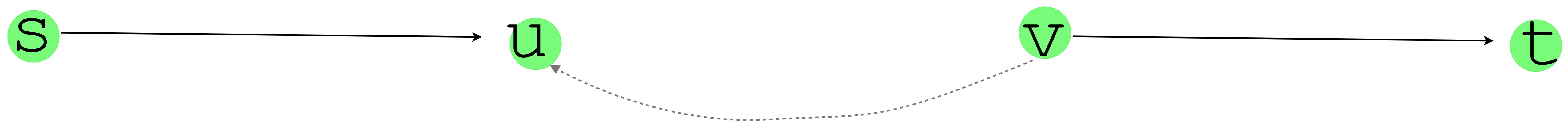


time j: Edge (u,v) STRIKES BACK

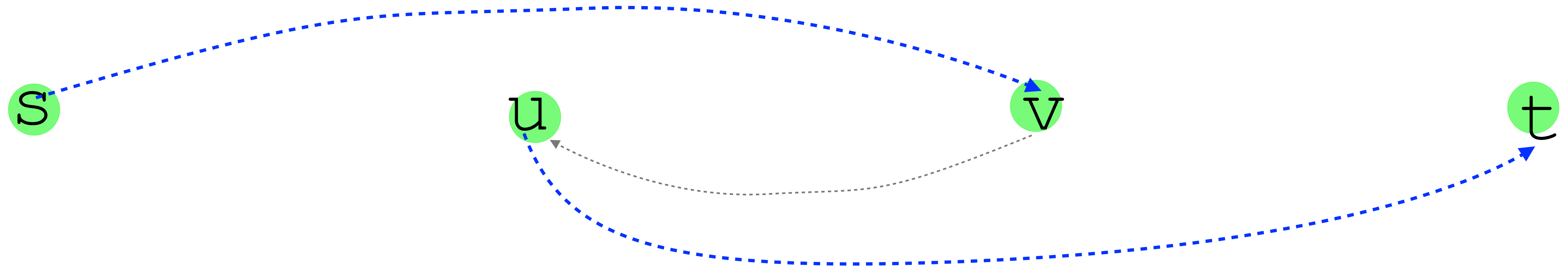




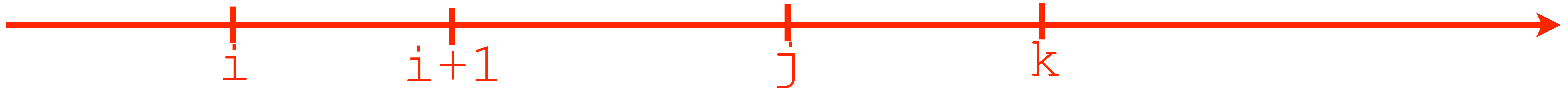
time i+1: (u,v) is critical: $\delta_{i+1}(s,v) \geq \delta_i(s,u) + 1$



time j: Edge (u,v) STRIKES BACK



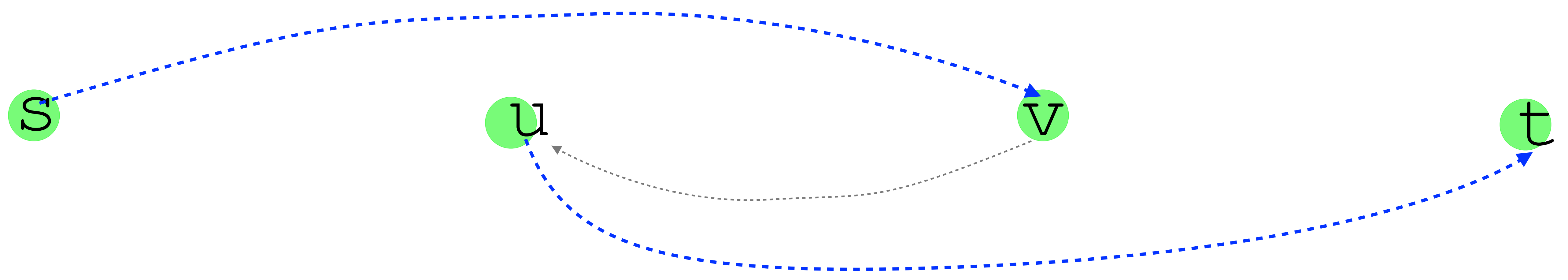
$\delta_j(s,u) = \delta_j(s,v) + 1$

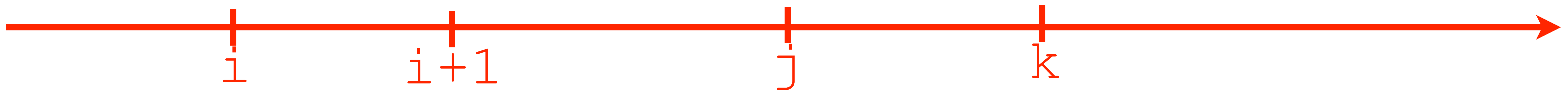


time j: Edge (u,v) STRIKES BACK

$$\delta_{i+1}(s,v) \geq \delta_i(s,u) + 1$$

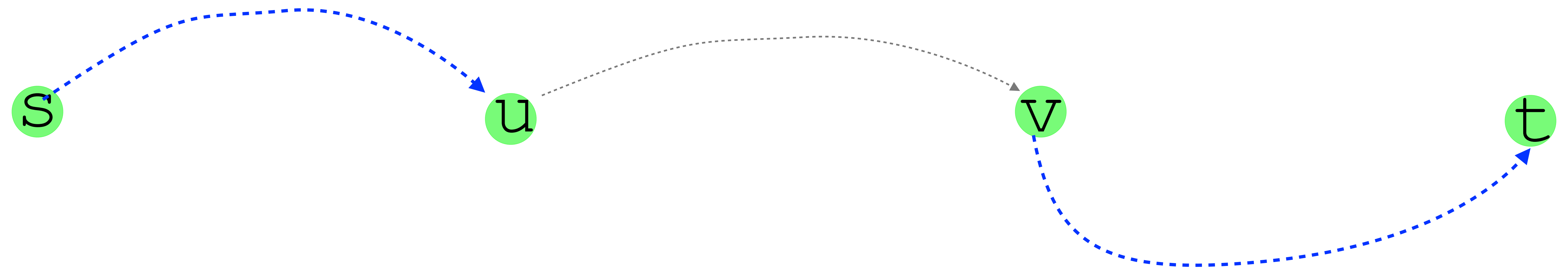
$$\delta_j(s,u) = \delta_j(s,v) + 1$$





time k: RETURN OF THE (u,v) critical

$$\delta_k(s,u) \geq \delta_i(s,u) + 2$$



QUESTION: How many times can (u,v) be critical?

edge critical only times.

there are only edges.

ergo, total # of augmenting paths:

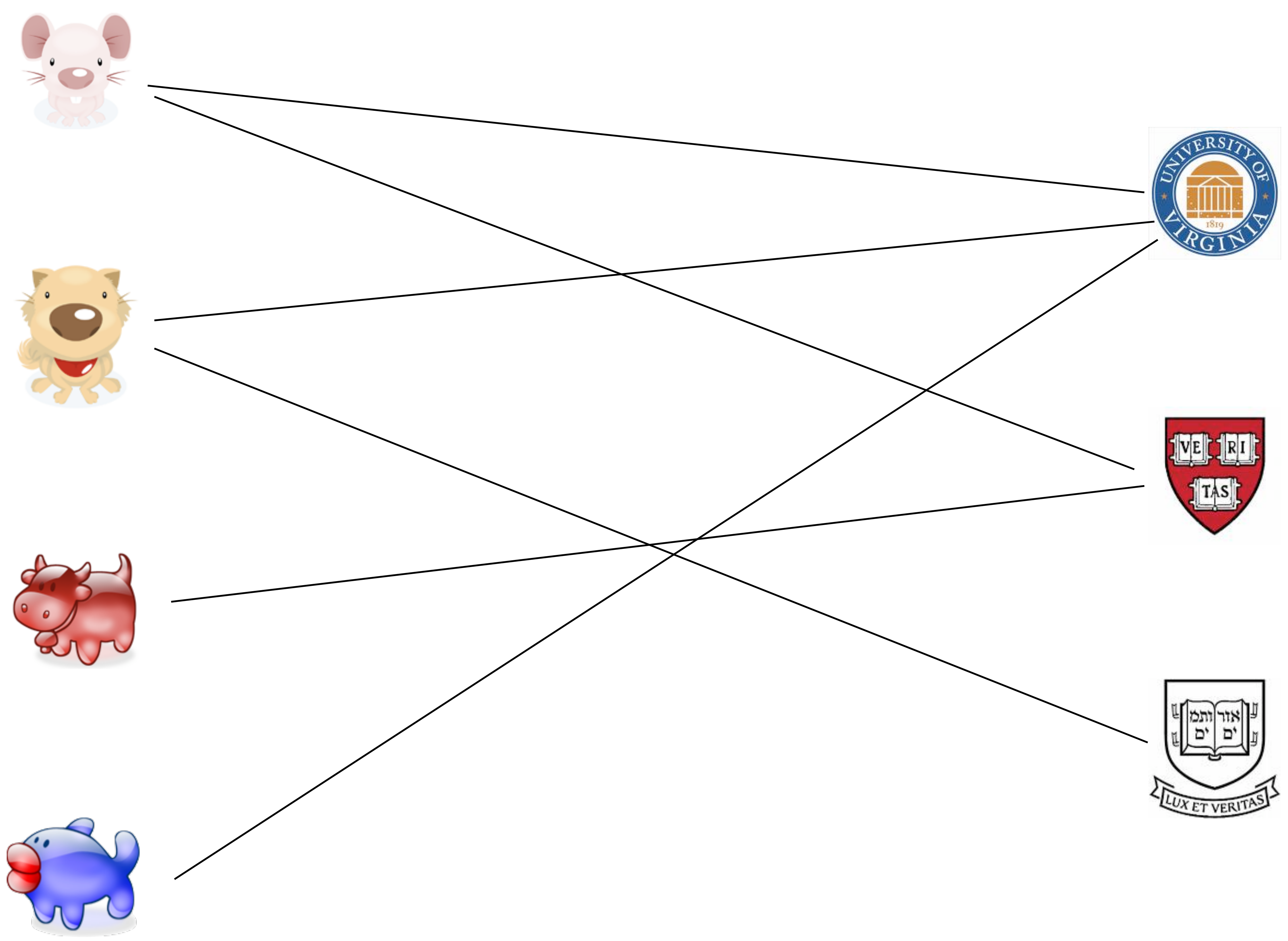
time to find an augmenting path:

total running time of E-K algorithm:

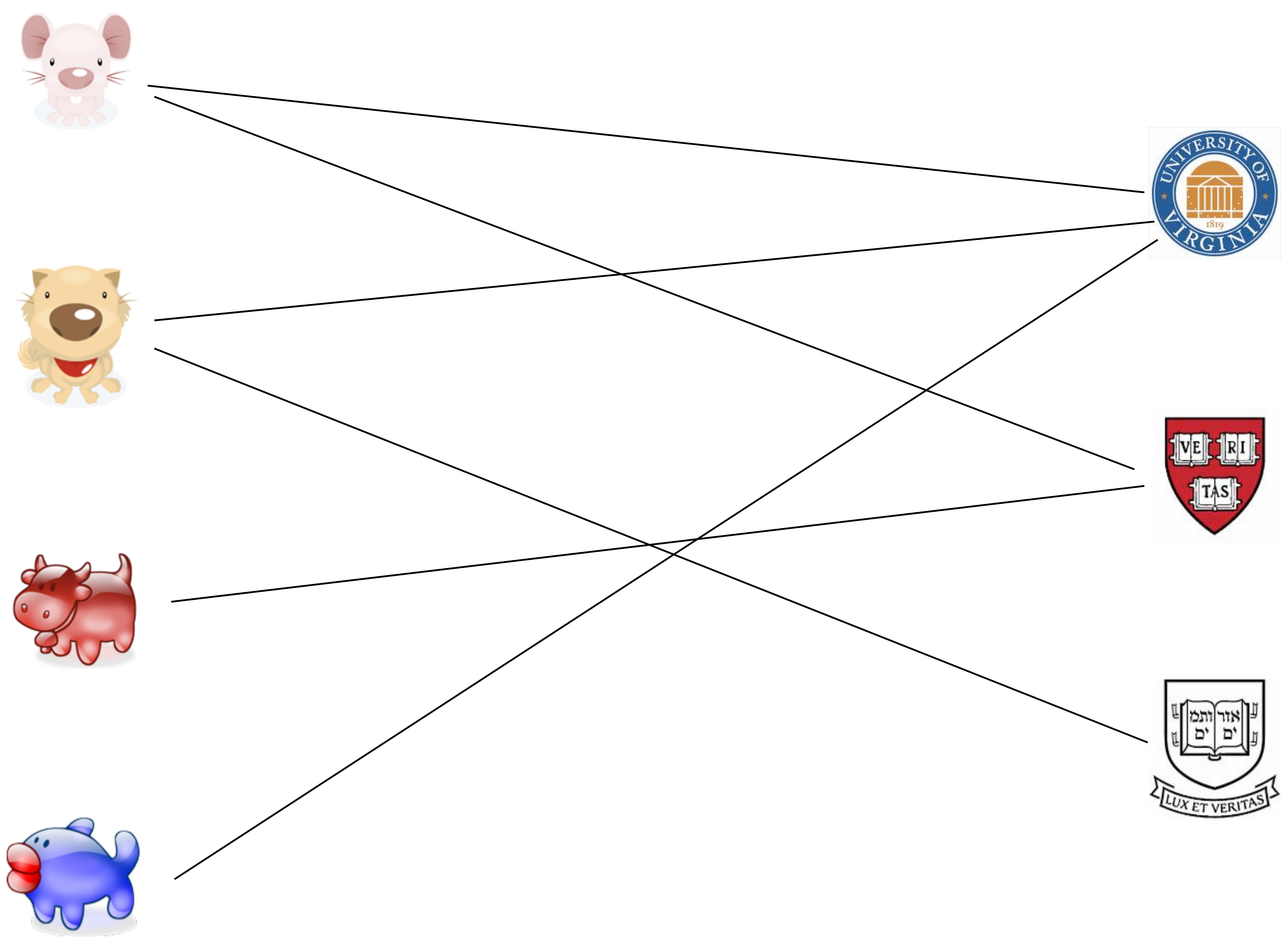
FF	$O(E\|f^*\|)$
EK2	
PUSH-RELABEL	
FASTER PUSH-RELABEL	

Bipartite Matchings

maximum bipartite matching



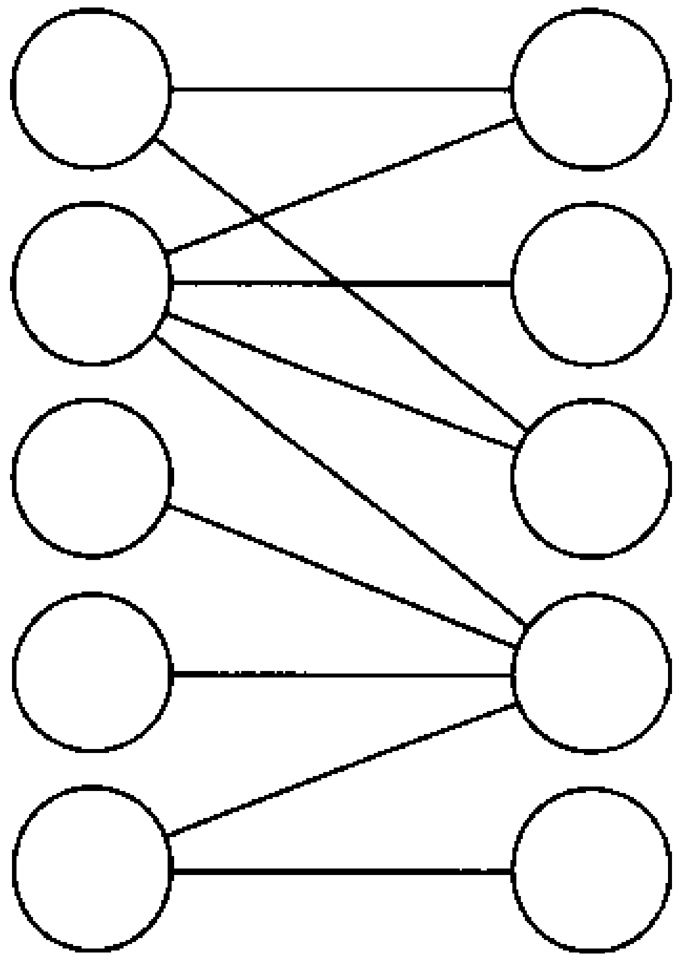
maximum bipartite matching



bipartite matching

PROBLEM:

algorithm

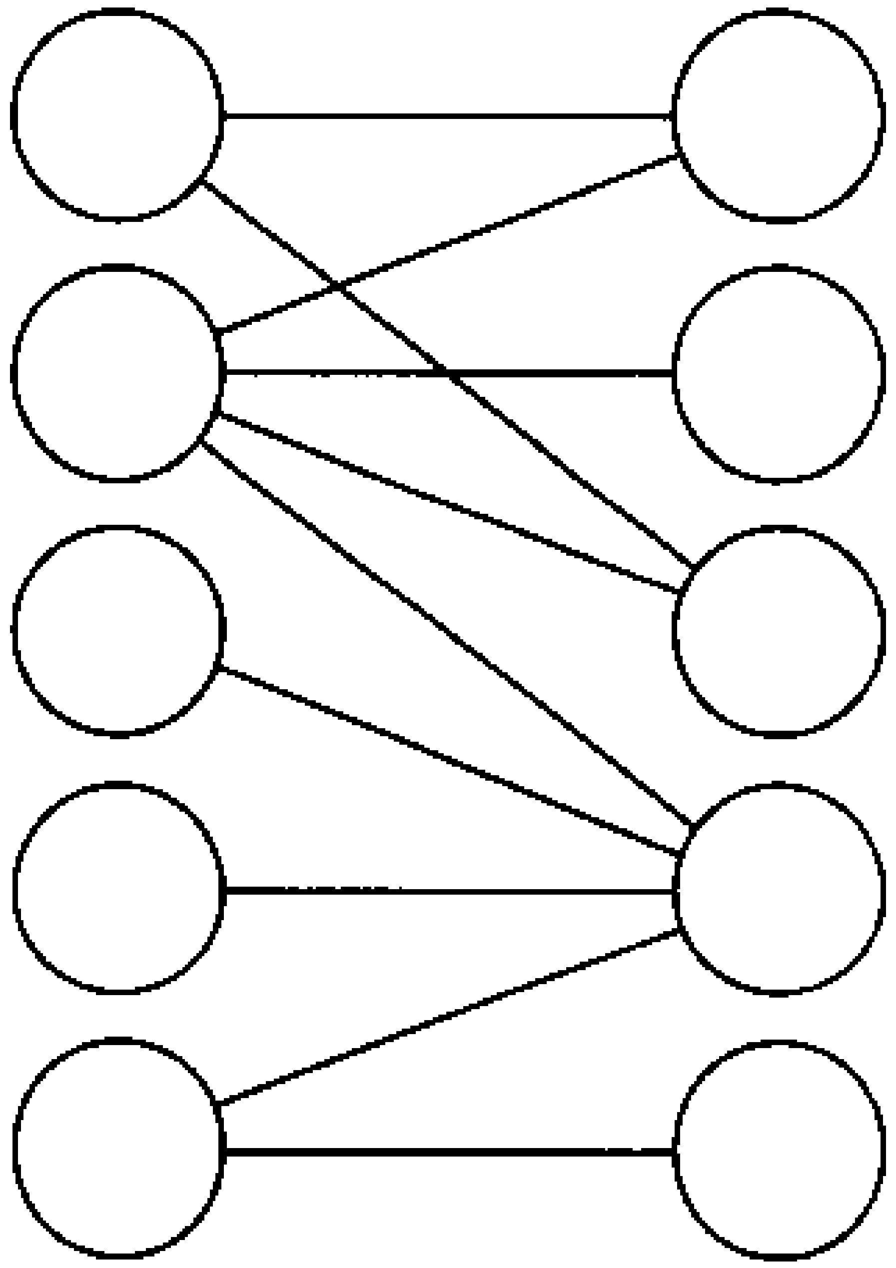


algorithm

1. MAKE NEW G'
FROM INPUT G.

2. RUN FF ON G'

3. OUTPUT ALL MIDDLE EDGES
WITH FLOW F(E)=1.



correctness

IF G HAS A MATCHING OF SIZE K, THEN

correctness

IF G' HAS A FLOW OF K, THEN

integrality theorem

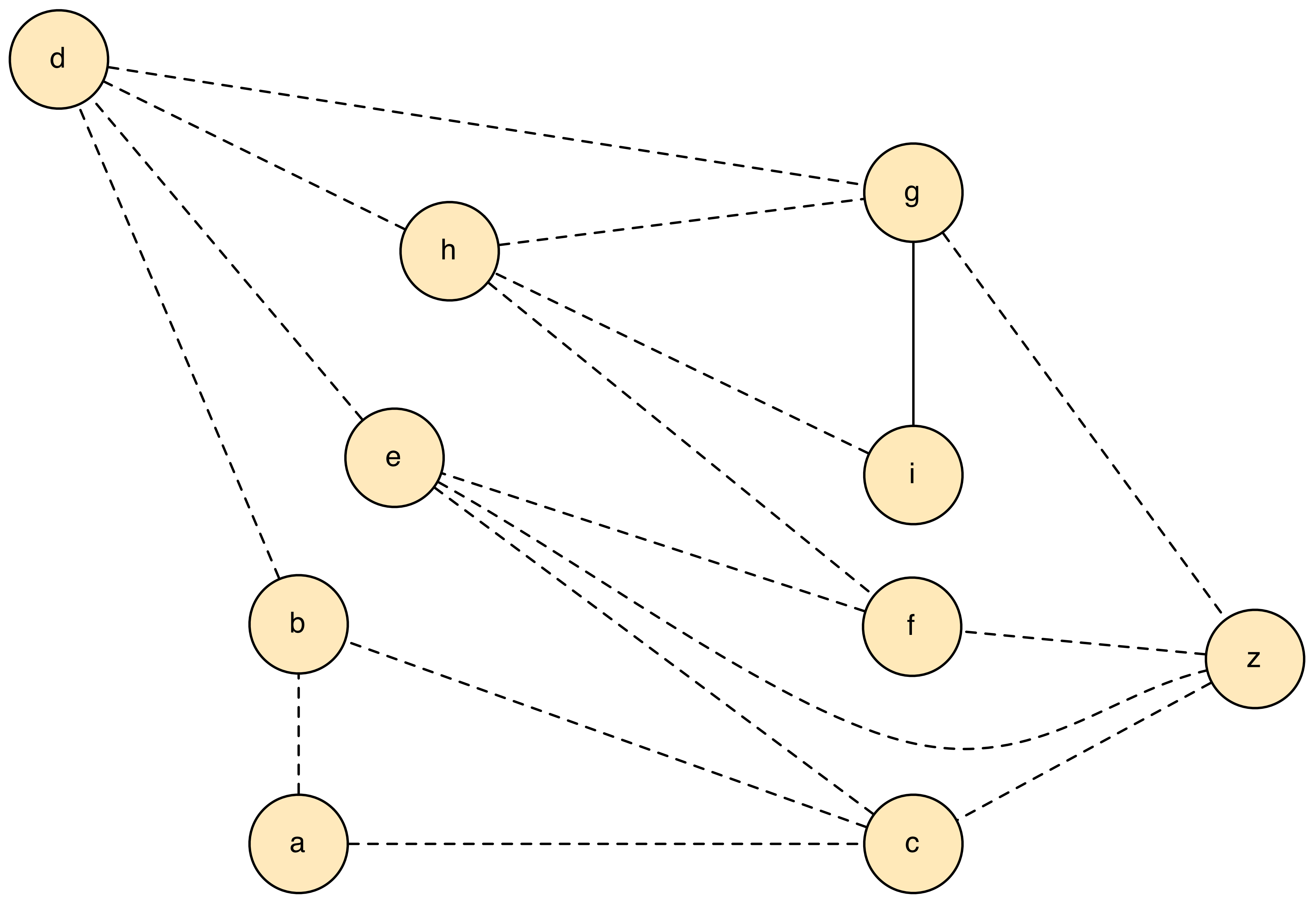
IF CAPACITIES ARE ALL INTEGRAL, THEN

correctness

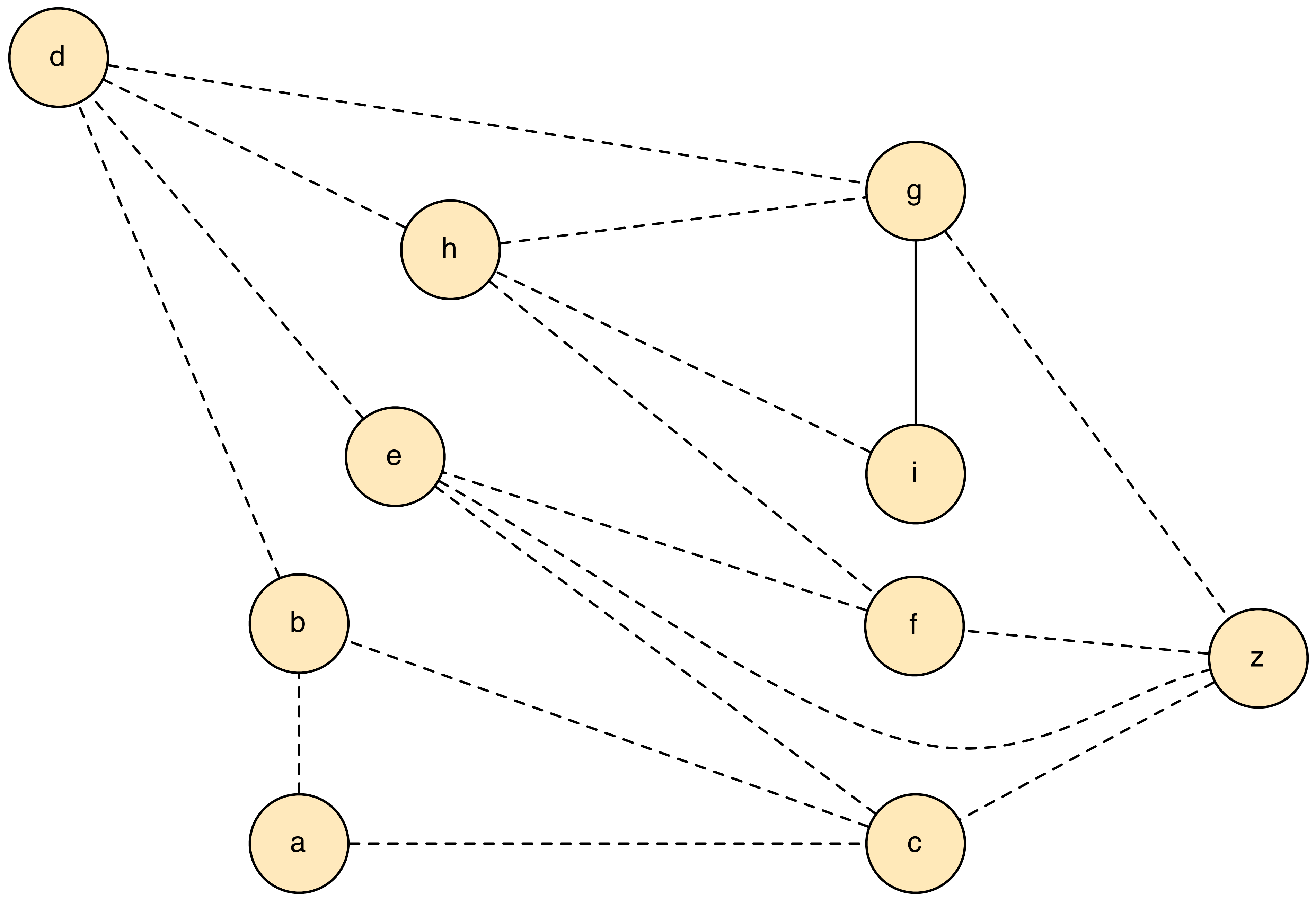
IF G' HAS A FLOW OF K, THEN G HAS K-MATCHING.

running time

edge-disjoint paths



algorithm



1. Compute max flow
2. Remove all edges with $f(e) = 0$.
3. Walk from s.
    1. If you reach a node you have visited before, erase flow along path
    2. If you reach t, add this path to your set, erase flow along path.

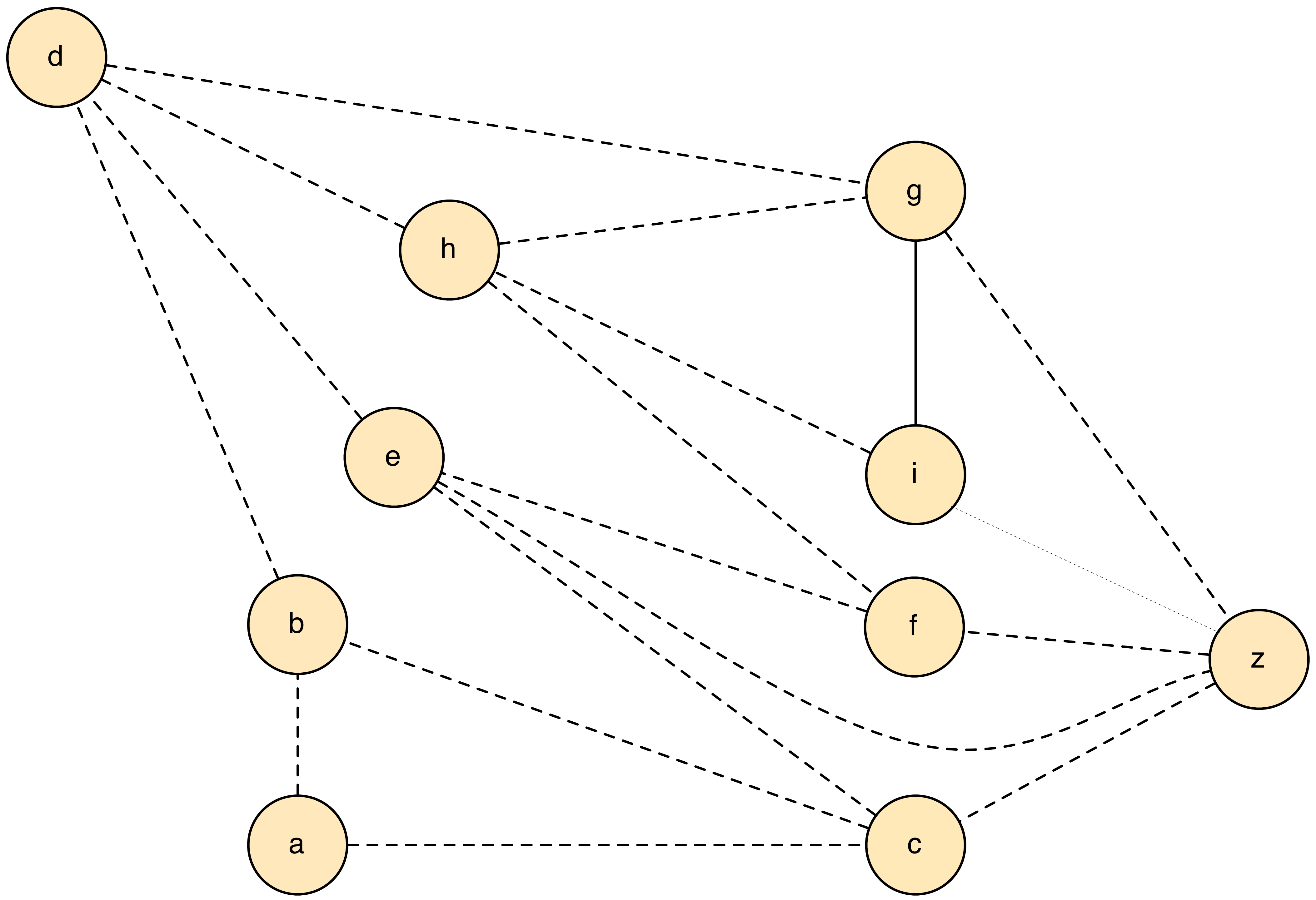
analysis

IF G HAS K DISJOINT PATHS, THEN

analysis

IF G' HAS A FLOW OF K, THEN

vertex-disjoint paths

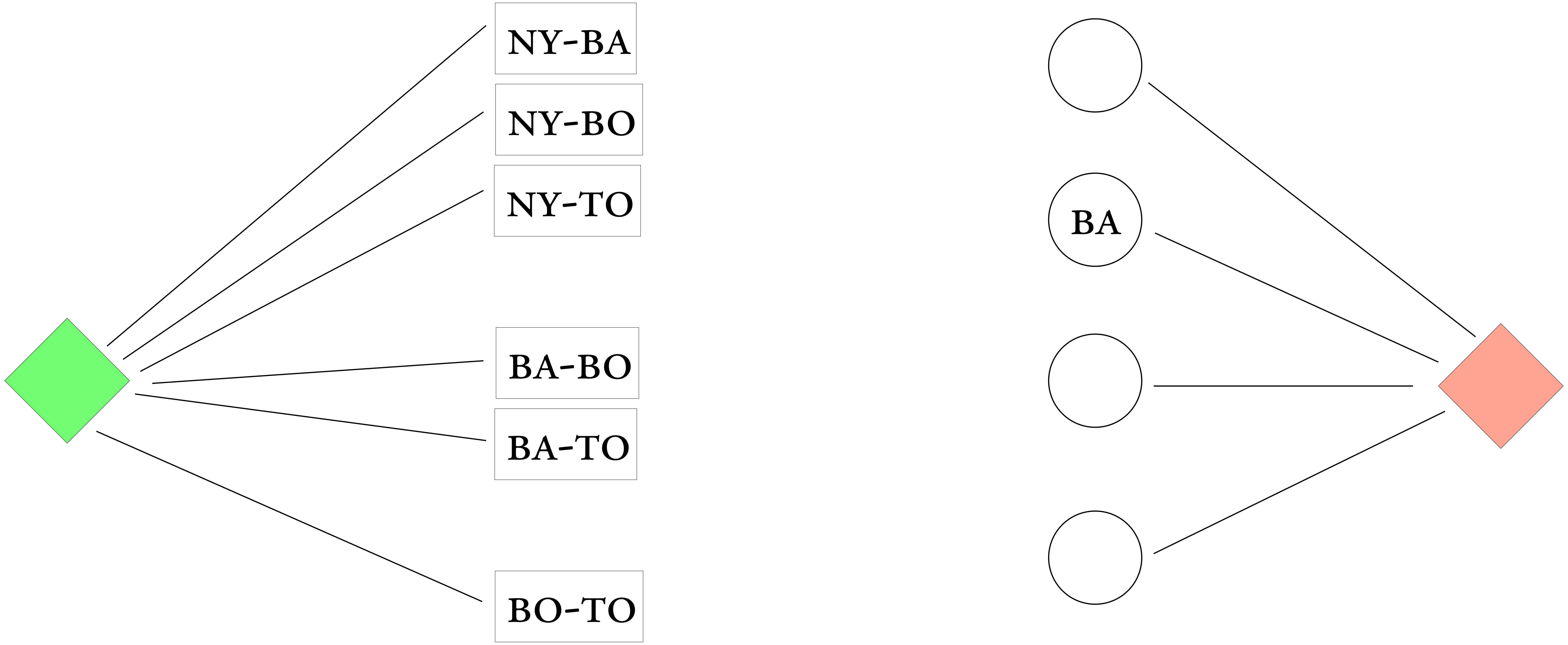


BASEBALL ELIMINATION

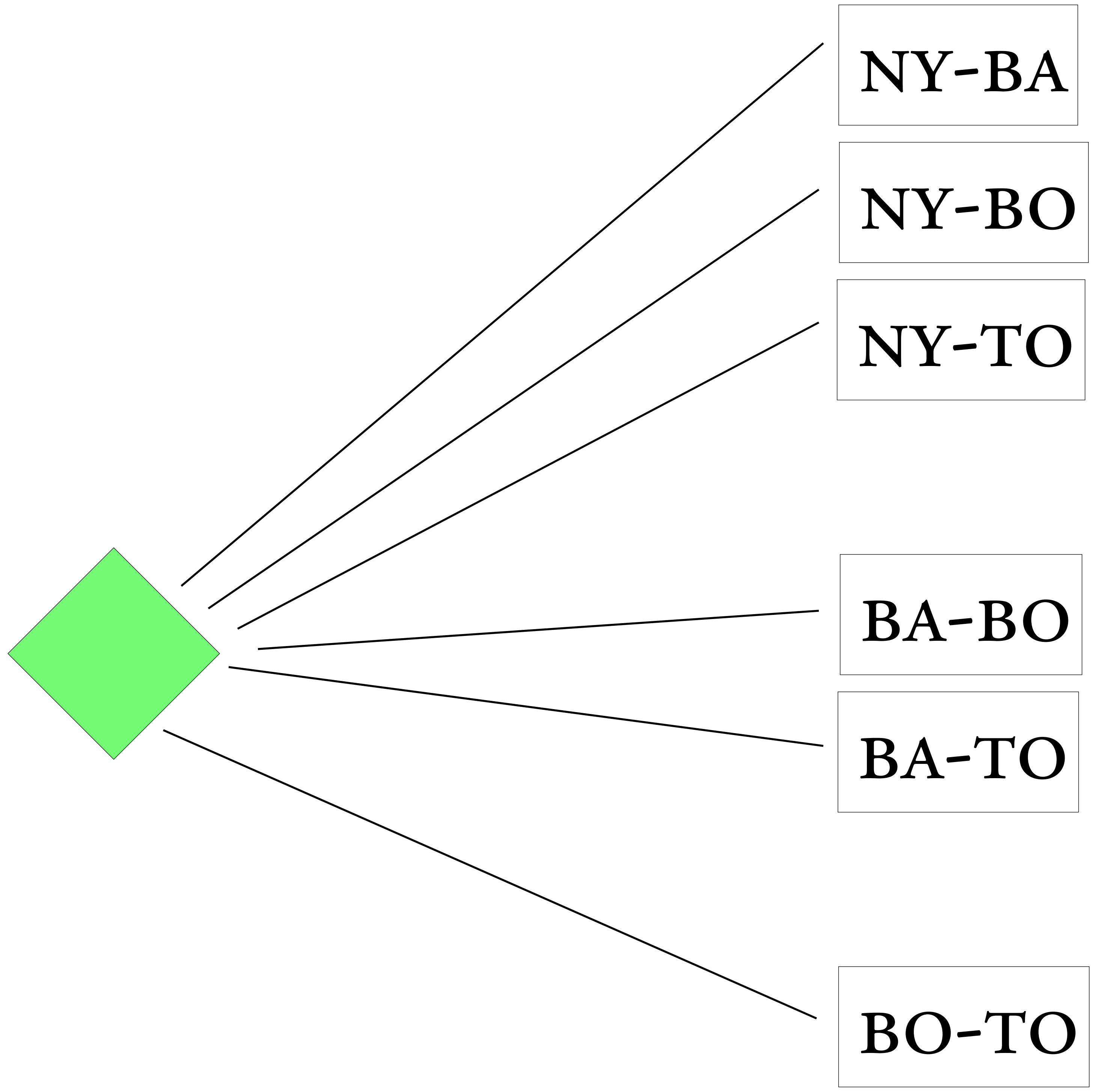
	W	L	Left	Against A	Against P	Against N	Against M
ATL	83	71	8	-	1	6	1
PHL	80	79	3	1	-	0	2
NY	78	78	6	6	0	-	0
MONT	77	82	3	1	2	0	-

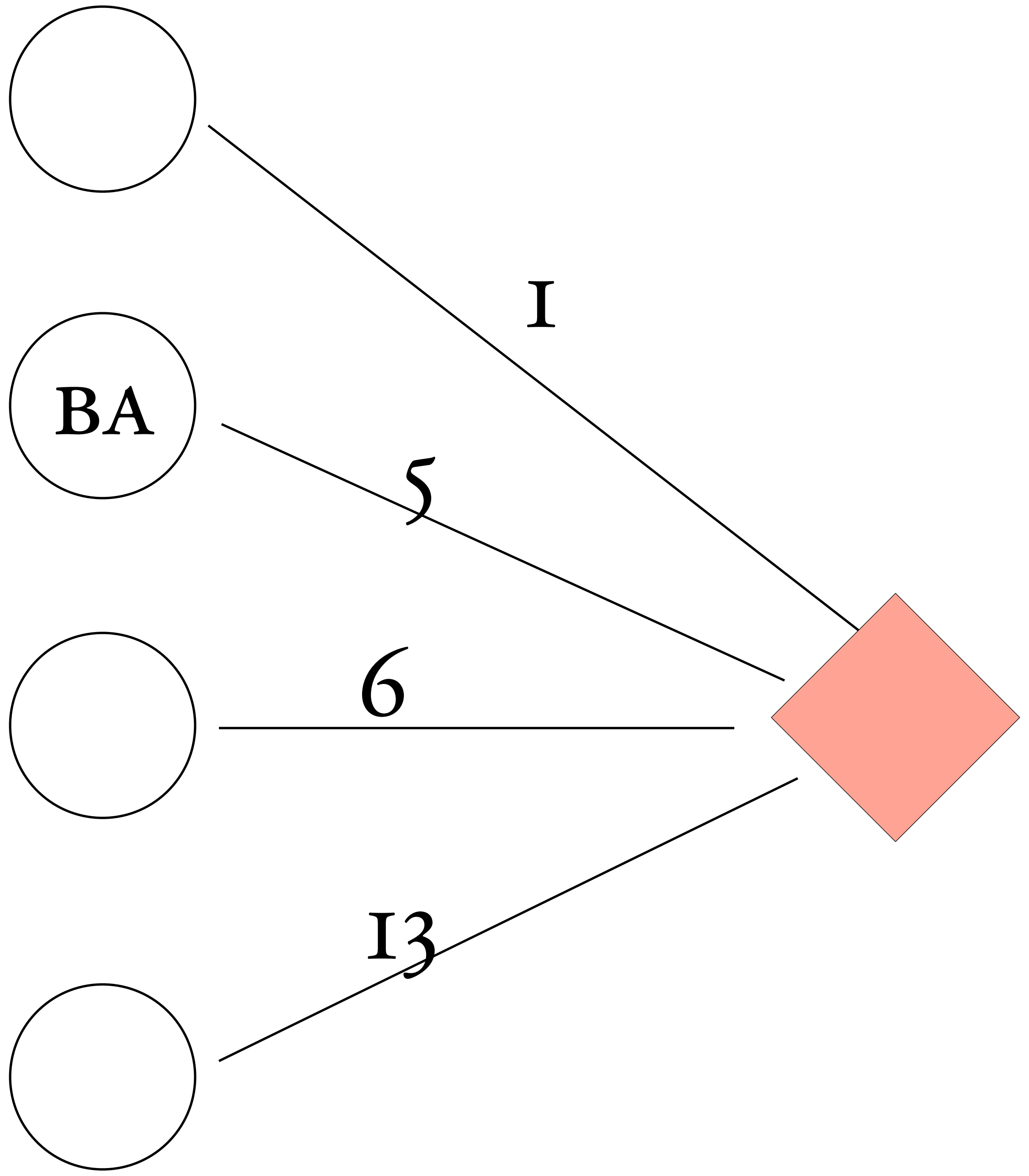
BASEBALL ELIMINATION

				Against				
	W	L	Left	N	B	Bo	T	D
NY	75	59	28		3	8	7	3
BAL	71	63	28	3		2	7	4
BOS	69	66	27	8	2			
TOR	63	72	27	7	7			
DET	49	86	27	3	4			

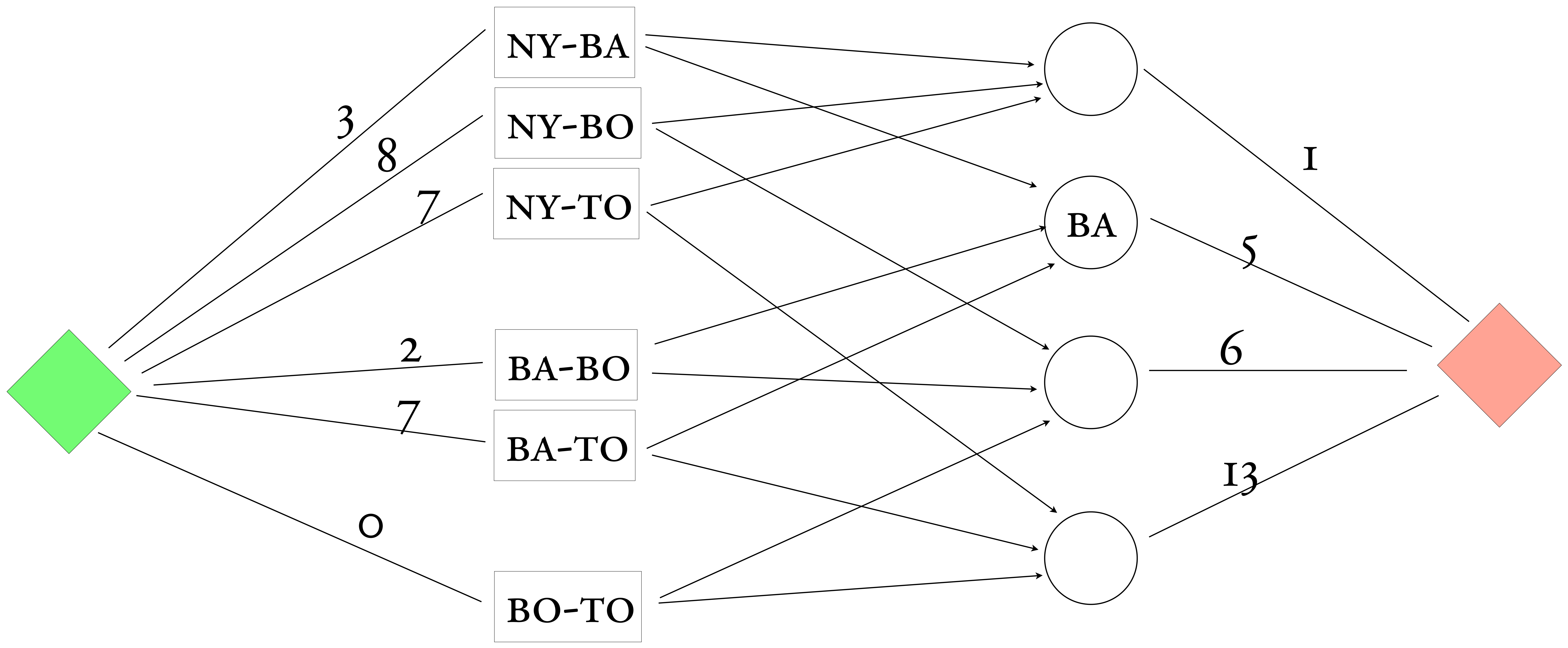


	W	L	Left	N	B	Bo	T	D
NY	75	59	28		3	8	7	3
BAL	71	63	28	3		2	7	4
BOS	69	66	27	8	2			
TOR	63	72	27	7	7			
DET	49	86	27	3	4			





	W	L	Left	N	B	Bo	T	D
NY	75	59	28		3	8	7	3
BAL	71	63	28	3		2	7	4
BOS	69	66	27	8	2			
TOR	63	72	27	7	7			
DET	49	86	27	3	4			



	W	L	Left	N	B	Bo	T	D
NY	75	59	28		3	8	7	3
BAL	71	63	28	3		2	7	4
BOS	69	66	27	8	2			
TOR	63	72	27	7	7			
DET	49	86	27	3	4			