CSCI-4470 HW-5

Nov 14 2023

Homework 5

Jun Wang
ID: 811574679

Upload a soft copy of your answers (single pdf file) to the submission link on elc before the due date. The answers to the homework assignment should be your own individual work. Make sure to show all the work/steps in your answer to get full credit. Answers without steps or explanation will be given zero.

Extra credit: There is 5 percentage extra credit if you don't submit hand written homework including the figures. You can use latex or any other tool to write your homework. For figures you can use any drawing tool and include the figure as a jpeg or a png file in your latex file.

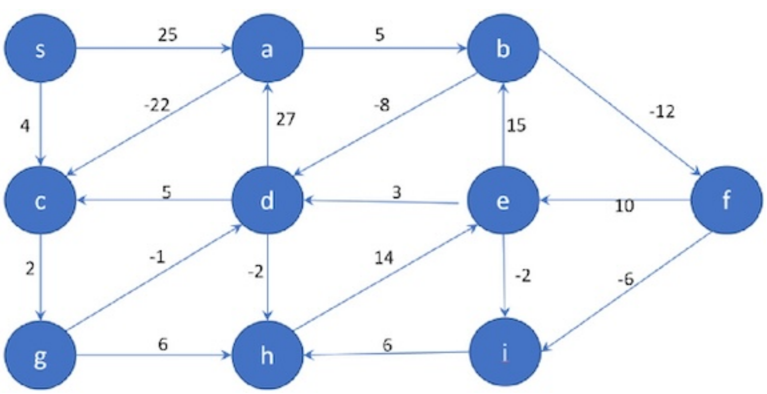
1.(10 points) Let $G = (V, E)$ be a weighted, directed graph. Assume that G is initialized using `Initialize-Single-Source(G,s)`. Prove that if a sequence of relaxation steps ever sets $s.\pi$ to a non-Nil value, then G contains a negative weight cycle.

```
INITIALIZE-SINGLE-SOURCE(G, s)
for each vertex ve G.V
  v.d= ∞
  v.π = NIL
s.d = 0
```

If $s.\pi$ in a graph G is set to a non-NIL value after a sequence of relaxation steps, it implies the presence of a negative weight cycle in G. This is because:

1. $s.\pi$ being non-NIL suggests a path back to the source s, which shouldn't exist in shortest-path contexts.
2. A cycle leading back to s and causing $s.\pi$ to update must be reducing the distance to s, indicating negative total weight.
3. Therefore, a non-NIL $s.\pi$ signifies a negative weight cycle in G.

2. (20 points) Consider the graph G:

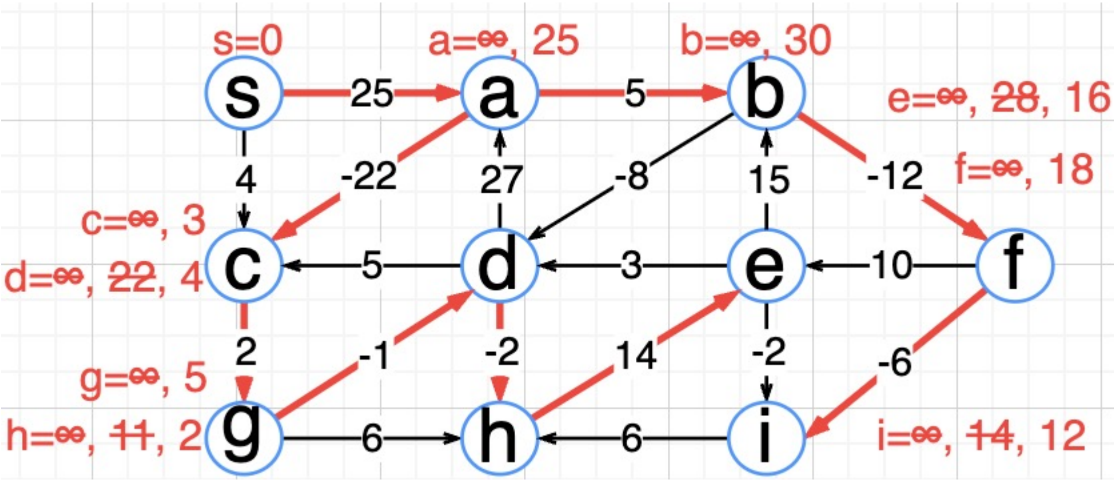


Use **Bellman-Ford** to find the single source shortest path graph starting at vertex s. Please process the edges in the following order:
$(s, a), (a, b), (a, c), (s, c), (d, a), (d, c), (b, d), (e, d), (e, b), (b, f),$
$(f, e), (c, g), (g, d), (g, h), (d, h), (h, e), (i, h), (e, i), (f, i).$
Draw the subgraph showing shortest path from s to all the other vertices.

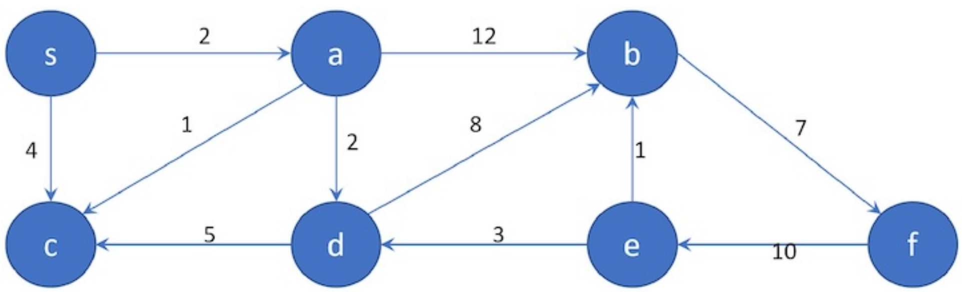
$(s, a), 0 + 25 = 25, \infty > 25$ (Relaxing). $(a, b), 25 + 5 = 30, \infty > 30$ (Relaxing),
$(a, c), 25 + (-22) = 3, \infty > 3$ (Relaxing). $(s, c), 0 + 4 = 4, 4 > 3$ (Relaxing),
$(d, a), \infty + 27 = \infty,\ \infty > 25$ (Relaxing). $(d, c), \infty + 5 = \infty,\ \infty > 3$ (Relaxing),
$(b, d), 30 + (-8) = 22,\ \infty > 22$ (Relaxing). $(e, d), \infty + 3 = \infty,\ \infty > 22$ (Relaxing),
$(e, b), \infty + 15 = \infty, \infty > 30$ (Relaxing). $(b, f), 30 + (-12) = 18, \infty > 18$ (Relaxing),
$(f, e), 18 + 10 = 28, 28 > 16$ (Relaxing). $(c, g), 3 + 2 = 5, \infty > 5$ (Relaxing),
$(g, d), 5 + (-1) = 4, 22 > 4$ (Relaxing). $(g, h), 5 + 6 = 11, \infty > 11$ (Relaxing),
$(d, h), 4 + (-2) = 2, 11 > 2$ (Relaxing). $(h, e), 2 + 14 = 16, 28 > 16$ (Relaxing),

$(i, h), \infty + 6 = \infty, \infty > 2$ (Relaxing). $(e, i), 16 + (-2) = 14,\ \infty > 14$ (Relaxing), $(f, i), 18 + (-6) = 12, 14 > 12$ (Relaxing).

Solution is: $\{(s = 0), (a = 25), (b = 30), (c = 3), (d = 4), (e = 16), (f = 18), (g = 5), (h = 2), (i = 12)\}$



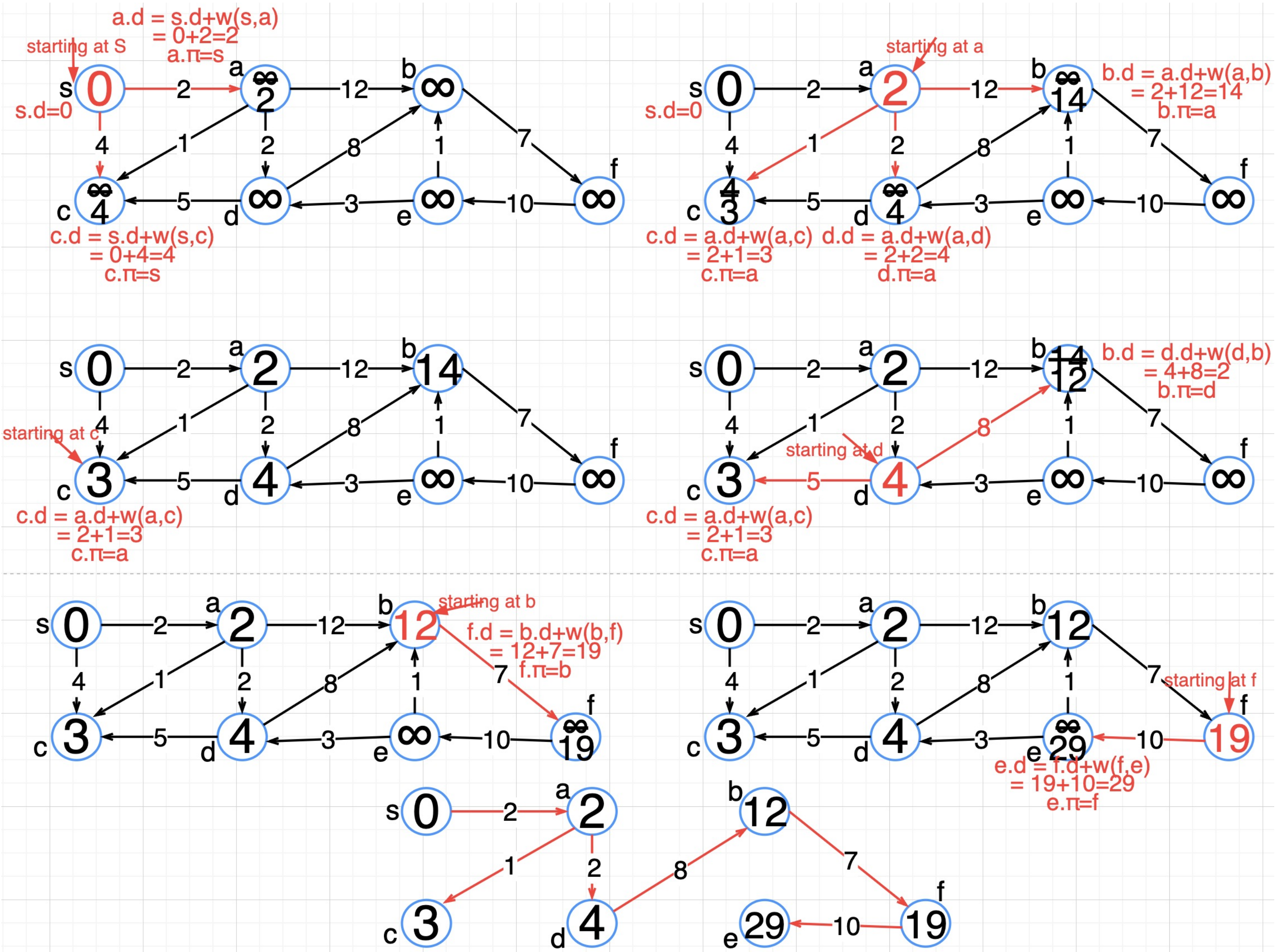
3. (20 points) Using s as the source, run Dijkstra's algorithm to find the SSSP sub-graph of the graph G shown below:



Specify the order in which vertices are added to S in the algorithm.

$V = \{s, a, b, c, d, e, f\}$, $E = \{(s, a, 2), (s, c, 4), (a, b, 12), (a, c, 1), (a, d, 2), (b, f, 7), (d, b, 8), (d, c, 5), (e, b, 1), (e, d, 3), (f, e, 10)\}$

- Starting from vertex s, , Set $s.d = 0$ (distance from source to source is 0). For all other vertices V, set $V.d = \infty$ and $V.\pi = \text{NIL}$.
- $s.\pi$ is not defined as s is the starting point. The vertex s keep s tracking of all vertices that its visited.



The shortest path is $\{s=0, a=2, c=3, d=4, b=12, f=19, e=29\}$
The order of vertices added to s is: $\{s, a, c, d, b, f, e\}$

4. (20 points) Find a feasible solution or determine that no feasible solution exists, By the Linear objective function $Ax \leq b, x \geq 0$.
for the following system of difference constraints: The given system of inequalities for all constraints $X_j - X_i \leq b_k$

$$\text{subject to} \begin{cases} X_1 - X_5 \leq 2 \\ X_3 - X_4 \leq 0 \\ X_5 - X_2 \leq -4 \\ X_2 - X_3 \leq 2 \\ X_4 - X_1 \leq 11 \end{cases}$$

Include the graph constructed from the difference constraints model in your solution.

Step 1. **Create Vertices**: For each variable X_i, create a vertex V_i in the graph.

- **Vertices**: $V_0, V_1, V_2, V_3, V_4, V_5$

Step 2. **Add a Source Vertex**: Add an additional vertex V_0 which will act as a source vertex.

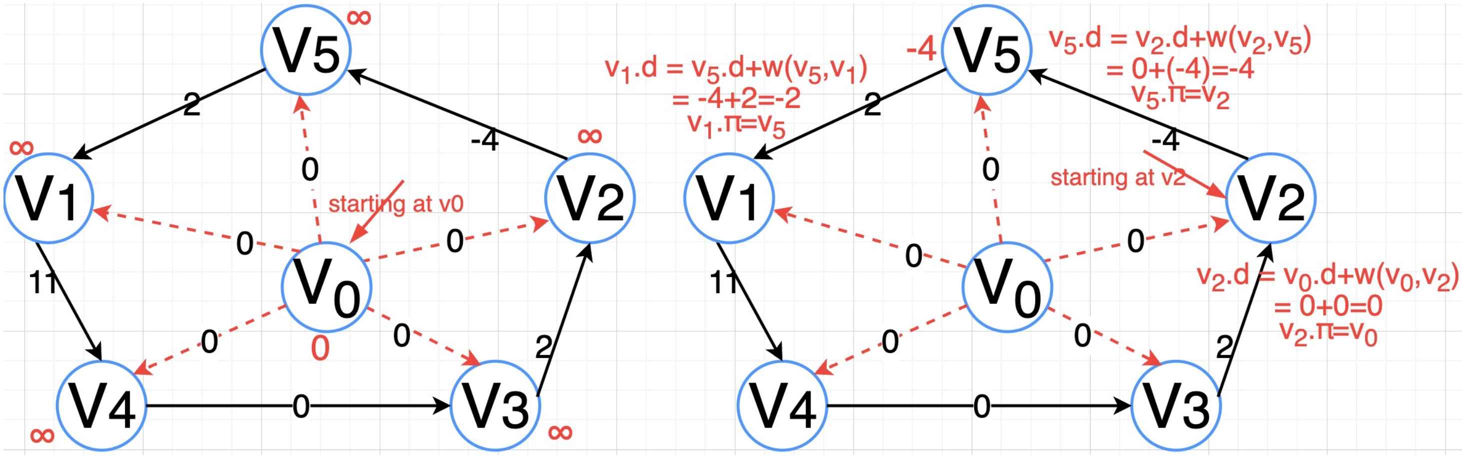
- A new Source Vertex V_0

Step 3. **Connect Source to All Vertices**: Connect V_0 to all other vertices V_i with edges of weight 0.

- **Connect V_0 to All Vertices**: Add edges with weight 0 from V_0 to V_1, V_2, V_3, V_4, V_5.

Step 4. **Add Edges for Constraints**: For each constraint $X_j - X_i \leq b_k$, add a directed edge from vertex V_i to vertex V_j with weight b_k.

- $X_1 - X_5 \leq 2$: Add edge from V_5 to V_1 with weight 2. $X_3 - X_4 \leq 0$: Add edge from V_4 to V_3 with weight 0.
- $X_5 - X_2 \leq -4$: Add edge from V_2 to V_5 with weight -4. $X_2 - X_3 \leq 2$: Add edge from V_3 to V_2 with weight 2.
- $X_4 - X_1 \leq 11$: Add edge from V_1 to V_4 with weight 11.



The feasible solution for the given system is:

- $V_0.d = 0, V_1.d = -2, V_2.d = 0, V_3.d = 0, V_4.d = 0, V_5.d = -4$

5. (20 points) In class we have talked about **Extend** algorithm that computes the shortest path between pair of vertices. You can start with the following definition of Extend, if `A = Extend(L,W)` then

$$a_{ij}^{LW} = \min_{1 \leq k \leq n} (l_{ik} + w_{kj})$$

Prove that **Extend** is associative. In other words, if you have 3 matrices $A, B,$ and C, then `Extend(A, Extend(B, C)) = Extend((Extend(A, B), C))`. Specifically, show if

- **D = Extend(A, B), E = Extend(D, C), F = Extend(B, C), and G=Extend(A,F), then E = G.**

```
EXTEND(L,W)
n = L.rows
Let L' = (l_{ij}') be a new n × n matrix
for i = 1 to n
   for j = 1 to n
      l_{ij}' = ∞
      for k = 1 to n
         l_{ij}' = min(l_{ij}', l_{ik} + w_{kj})
return L'
```

1. **Definition of Extend**: The result of `Extend(L, W)` is a matrix where each element l'_{ij} is computed as the minimum of $l_{ik} + w_{kj}$ over all k, for $1 \leq k \leq n$. This means l'_{ij} is the shortest path from i to j through an intermediate vertex.
2. **Computing D**: For d_{ij} in D, we have:

$$d_{ij} = \min_{1 \leq k \leq n} (a_{ik} + b_{kj})$$

   This is the shortest path from i to j using matrices A and B.
3. **Computing E**: For e_{ij} in E, we have:

$$e_{ij} = \min_{1 \leq k \leq n} (d_{ik} + c_{kj}) = \min_{1 \leq k \leq n} (\min_{1 \leq m \leq n} (a_{im} + b_{mk}) + c_{kj})$$

   We substitute d_{ik} with its definition from step 2.
4. **Computing F**: For f_{ij} in F, we have:

$$f_{ij} = \min_{1 \leq k \leq n} (b_{ik} + c_{kj})$$

   This is the shortest path from i to j using matrices B and C.
5. **Computing G**: For g_{ij} in G, we have:

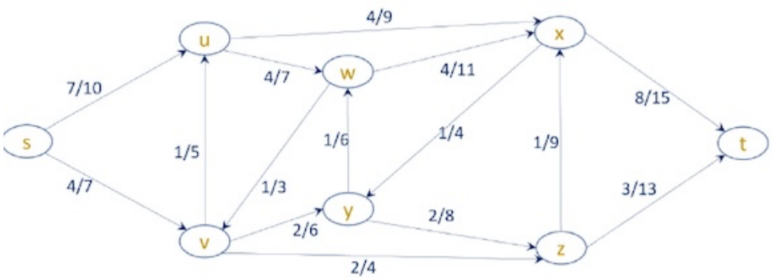
$$g_{ij} = \min_{1\le k\le n}(a_{ik} + f_{kj}) = \min_{1\le k\le n}(a_{ik} + \min_{1\le m\le n}(b_{km} + c_{mj}))$$

We substitute f_{kj} with its definition from step 4.

6. **Proving Associativity**: We need to show that $e_{ij} = g_{ij}$. Both e_{ij} and g_{ij} are expressions of triple nested minima involving elements of matrices A, B, and C. By the properties of the minima operation, we can rearrange the terms without affecting the outcome. Therefore, e_{ij} and g_{ij} are equivalent, proving the associativity of the **Extend** operation.

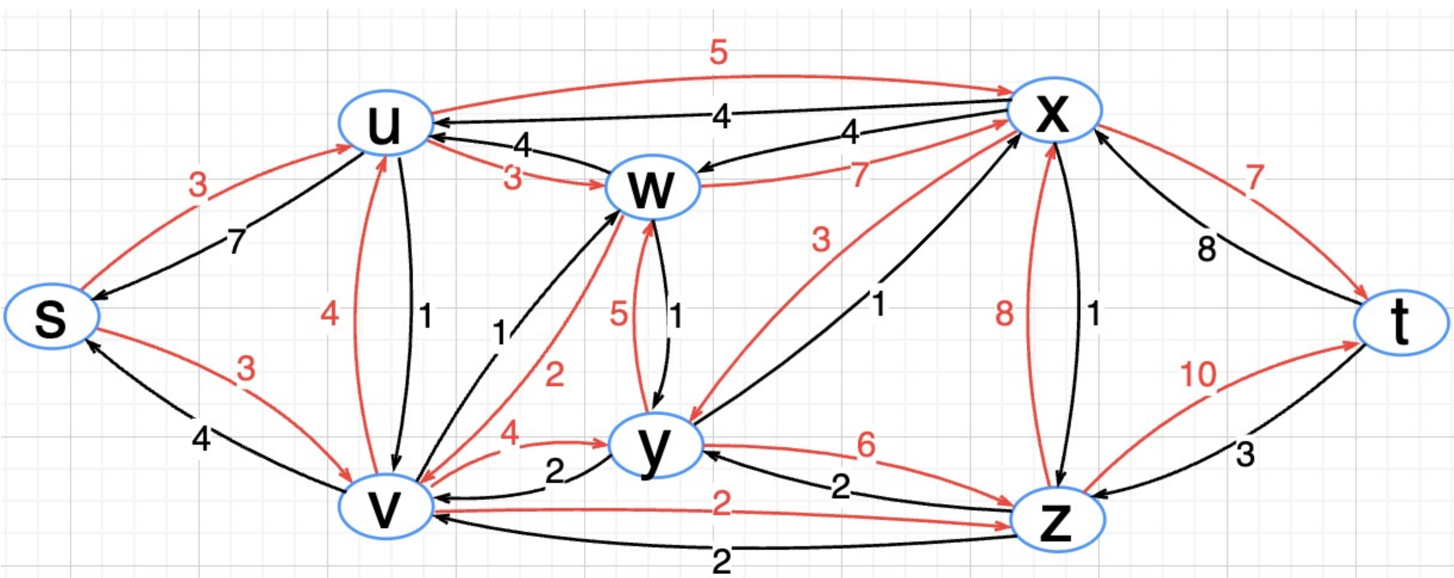
This proves that the **Extend** algorithm is associative,

6. (5 points each) Consider the following flow network.



the Flow network $G =$
$\{(s,u,7/10),(s,v,4/7),(v,u,1/5),(u,w,4/7),(u,x,4/9),(v,y,2/6),(v,z,2/4),(w,v,1/3),(w,x,4/11),(x,y,1/4),(x,t,8/15),(y,w,1/6),(y,z,2/8$

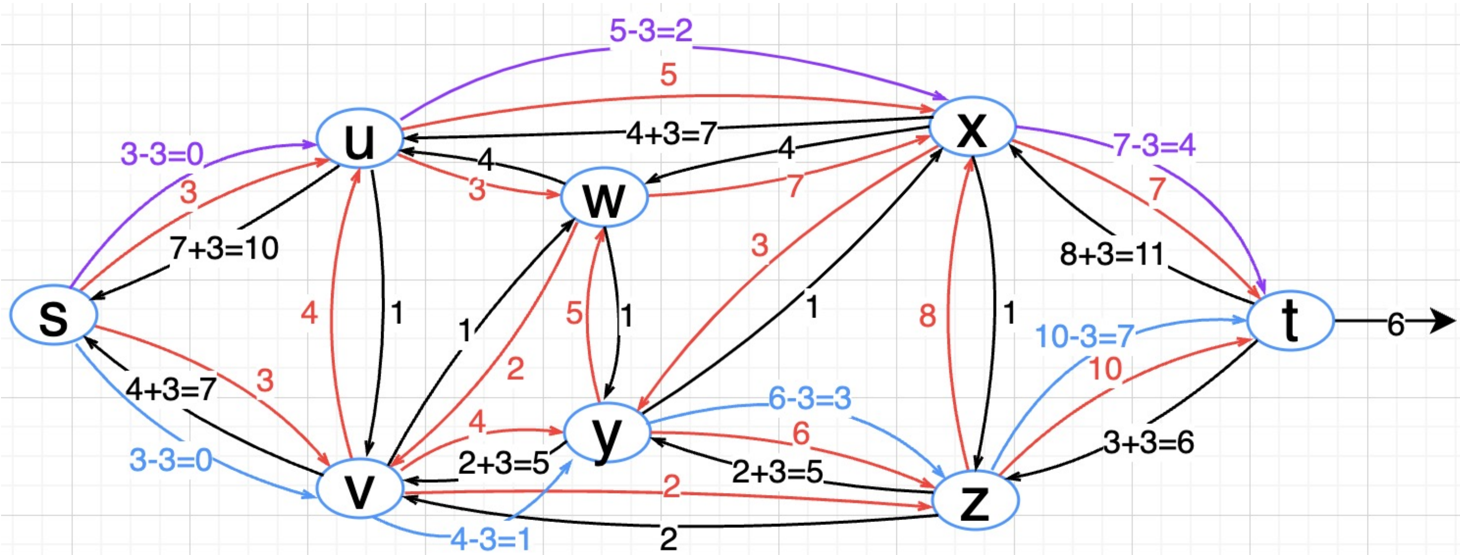
(a) Draw the residual network for this flow.



the value 3/7 3 is the residual capacity, 7 is reverse capacity

The residual network $G_f =$
$\{(s,u,3/7),(s,v,3/4),(v,u,4/1),(u,w,3/4),(u,x,5/4),(v,y,4/2),(v,z,2/2),(w,v,2/1),(w,x,7/4),(x,y,3/1),(x,t,7/8),(y,w,5/1),(y,z,6/2),($

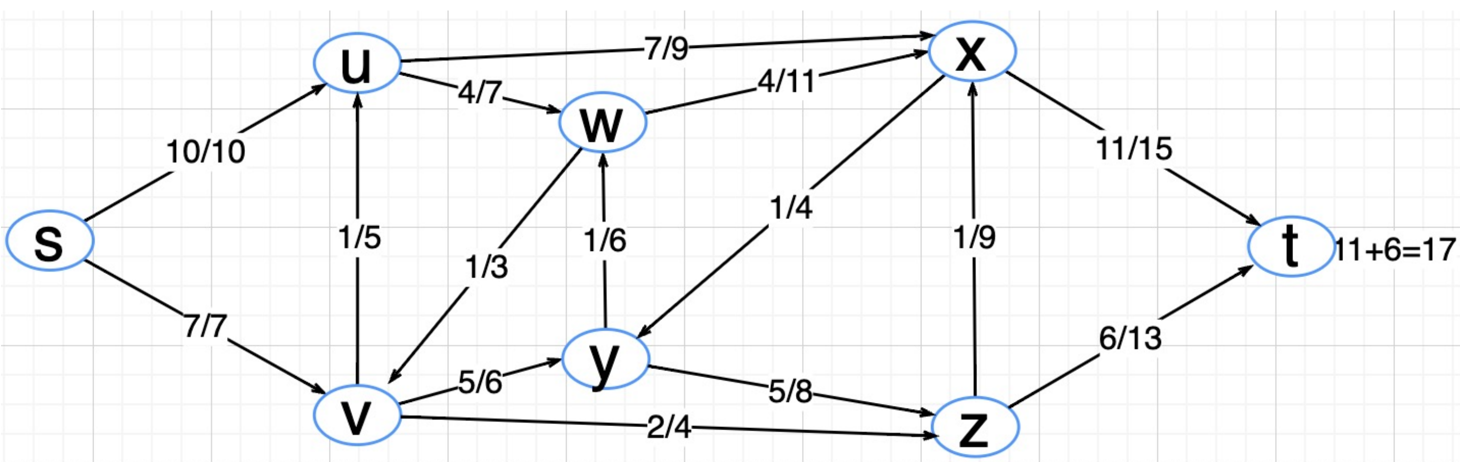
(b) Find an augmenting path that increases at the flow across at least one edge and decreases the flow across another edge. Make the flow be as large as possible for the chosen path.



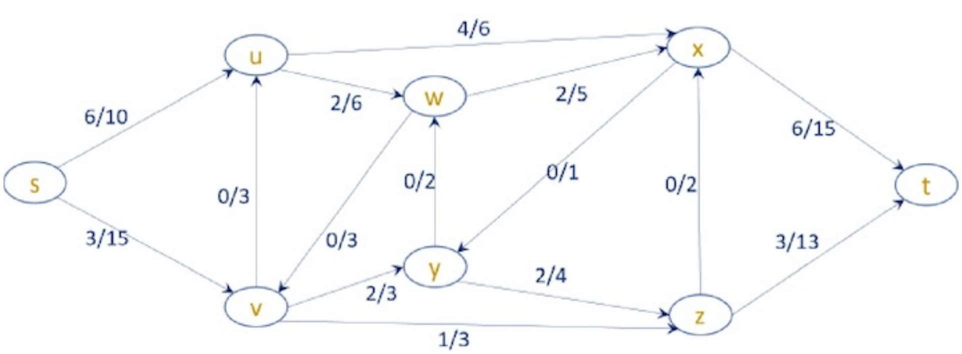
(c) Describe the augmenting path as a ordered list of vertices and state how much flow goes through it.

- $p_1 = \{s, u, x, t\}, c_f(p_1) = 3$
- $p_2 = \{s, v, y, z, t\}, c_f(p_2) = 3$

(d) Finally, draw a new flow graph with the augmented flow.

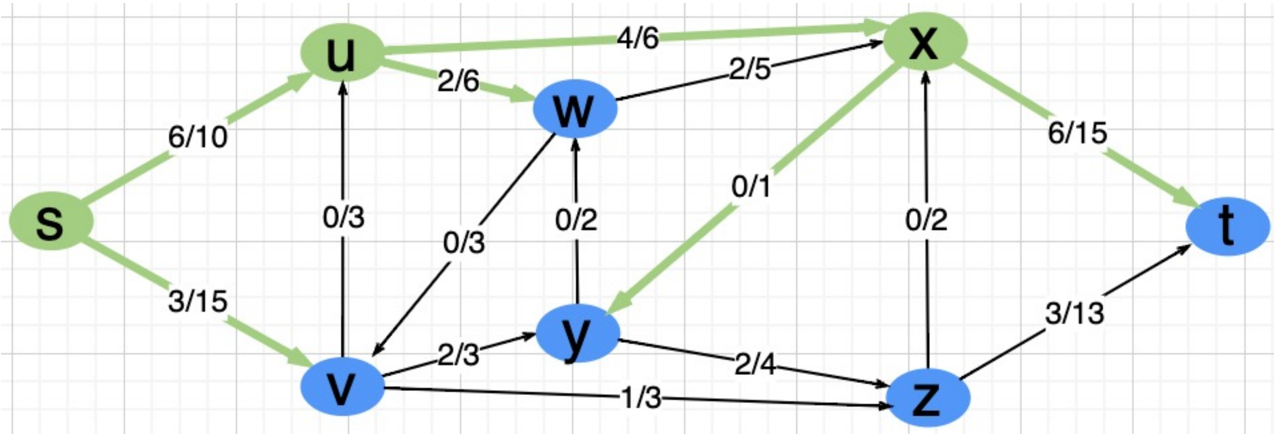


7. Consider the following flow network.



Let $S = \{s, u, x\}$ and $T = \{v, w, y, z, t\}$

(a) What is $f(S, T)$?



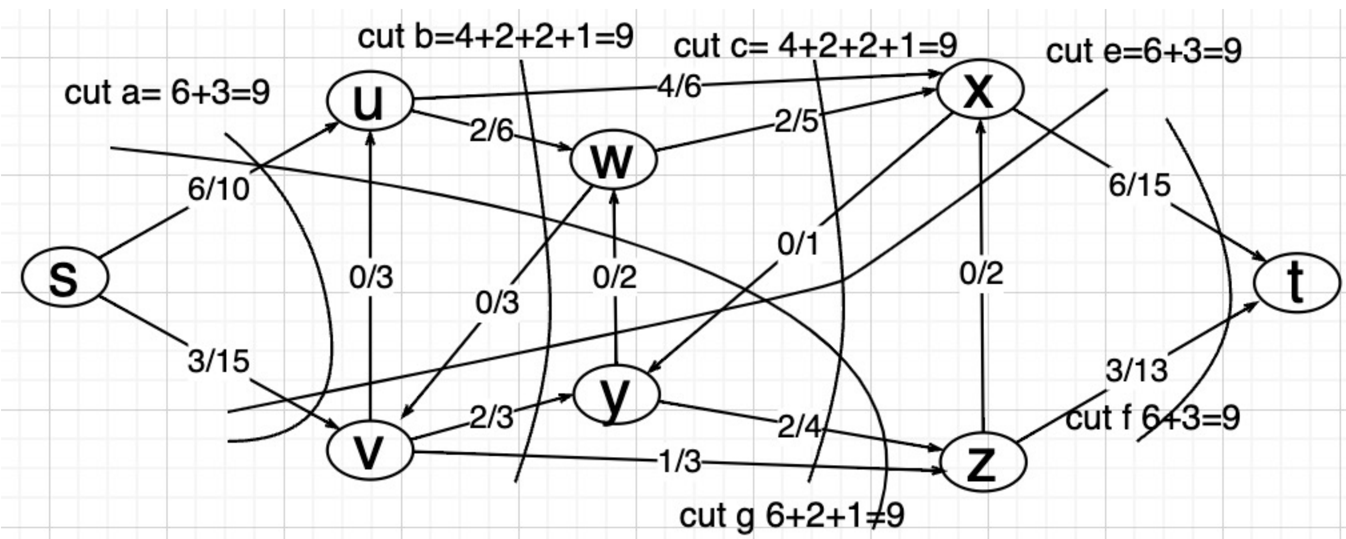
- $f(S,T): 6+3+2+4+0+6=21$

(b) What is $c(S,T)$?

- $c(S,T): 10+15+6+6+1+15=53$

(c) Find a minimum cut for G (Hint: Add augmenting paths until no augmenting path can be found. Which edges are reachable from s in the G_f?

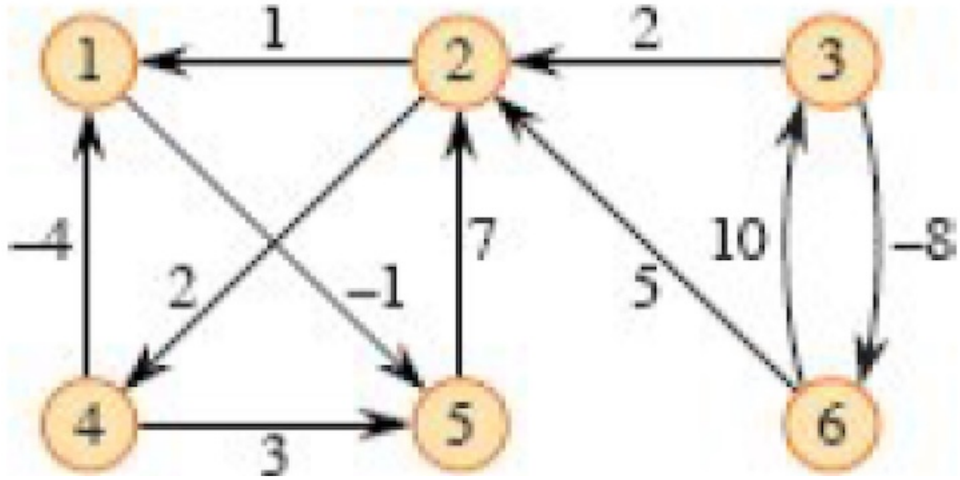
- **Minimum Cut == Maximum Flow**



- From those cut sets, the minimum cut is 9.

8. (30 points) Bonus question Use Johnson's algorithm to find the shortest paths between all pairs of vertices in the graph below. Show the values of h and $\hat{w}$ computed by the algorithm.

The graph $G=(V,E), V=\{s,1,2,4,,4,5,6\}, E=$
$\{(1,5,-1),(2,1,1),(2,4,2),(3,2,2),(3,6,-8),(4,1,-4),(4,5,3),(5,2,7),(6,2,5),(6,3,10),(s,1,0)(s,2,0),(s,3,0),(s,4,0),(s,5,0),(s,6,0)\}$



Step 1: Apply Bellman-Ford, to find all $h(1)$,$h(2)$,$h(2)$,$h(3)$,$h(4)$, and $h(5)$, which are the shortest paths from source s to vertices.

$h(1) = \delta(s,1) = 0 + (-4) = -4$

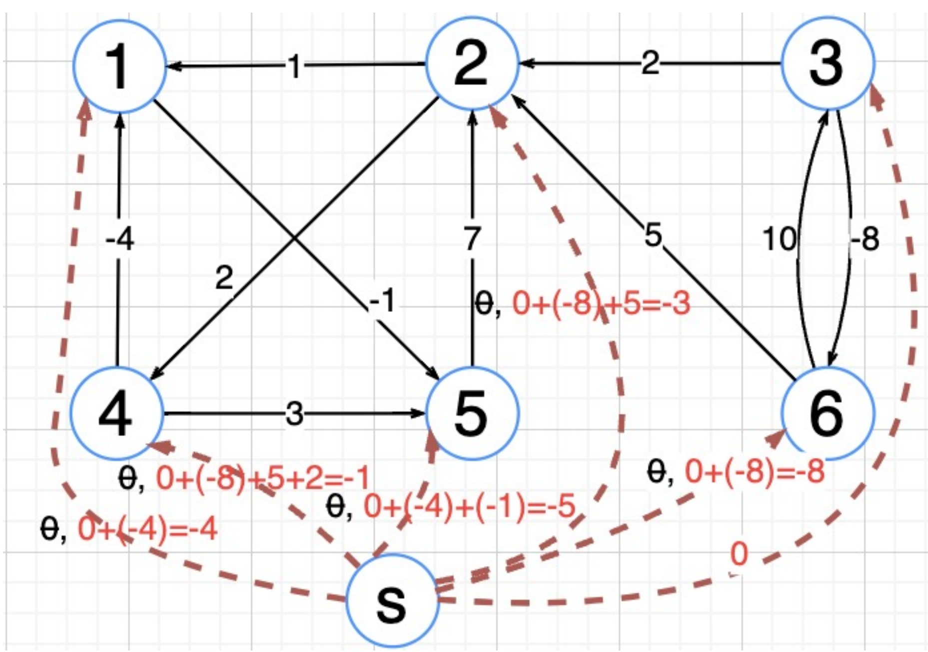
$h(2) = \delta(s,2) = 0 + (-8) + 5 = -3$

$h(3) = \delta(s,3) = 0$

$h(4) = \delta(s,4) = 0 + (-8) + 5 + 2 = -1$

$h(5) = \delta(s,5) = 0 + (-4) + (-1) = -5$

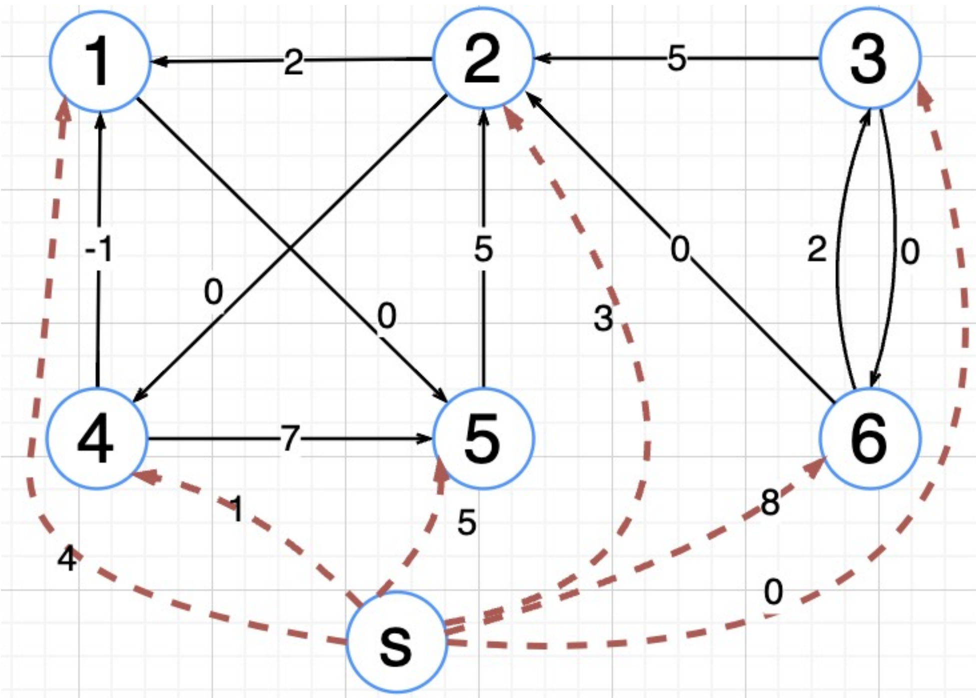
$h(6) = \delta(s,6) = 0 + (-8) = -8$



Step 2: Reweighting Edges $\hat{w}(u,v) = w(u,v) + h(u) - h(v)$

$\hat{w}(1,5) = -1 + (-4) - (-5) = 0$
$\hat{w}(2,1) = 1 + (-3) - (-4) = 2$
$\hat{w}(2,4) = 2 + (-3) - (-1) = 0$
$\hat{w}(3,2) = 2 + (0) - (-3) = 5$
$\hat{w}(3,6) = -8 + (0) - (-8) = 0$
$\hat{w}(4,1) = -4 + (-1) - (-4) = -1$
$\hat{w}(4,5) = 3 + (-1) - (-5) = 7$
$\hat{w}(5,2) = 7 + (-5) - (-3) = 5$
$\hat{w}(6,2) = 5 + (-8) - (-3) = 0$
$\hat{w}(6,3) = 10 + (-8) - (0) = 2$
$\hat{w}(s,1) = 0 + (0) - (-4) = 4$
$\hat{w}(s,2) = 0 + (0) - (-3) = 3$
$\hat{w}(s,3) = 0 + (0) - (0) = 0$

$\hat{w}(s,4) = 0 + (0) - (-1) = 1$
$\hat{w}(s,5) = 0 + (0) - (-5) = 5$
$\hat{w}(s,6) = 0 + (0) - (-8) = 8$



Since There is a negative weight in the reweighted graph, Dijkstra's Algorithm can be performance.