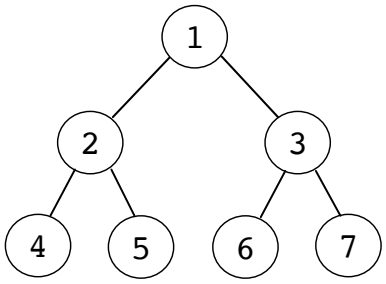
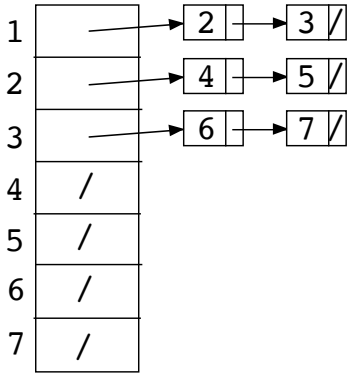
1. 22.1-2 Give an adjacency-list representation for a complete binary tree on 7 vertices. Give an equivalent adjacency-matrix representation. Assume that vertices are numbered from 1 to 7 as in a binary heap. (Edges are directed from parent to child)





Adjacency Matrix:

	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0
2	0	0	0	1	1	0	0
3	0	0	0	0	0	1	1
4	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0

2. 22.1-5 The square of a directed graph G = (V, E) is the graph $G^2 = (V, E^2)$ such that $(u,w) \in E^2$ if and only if for some $v \in V$, both $(u,v) \in E$ and $(v, w) \in E$. That is, G^2 contains an edge between u and w whenever G contains a path with exactly two edges between u and w. Describe efficient algorithms for computing G^2 from G for both the adjacency-list and adjacency-matrix representations of G. Analyze the running times of your algorithms.

G^2 for an adjacency matrix: - Computing G^2 may be done in V^3 time by matrix multiplication:

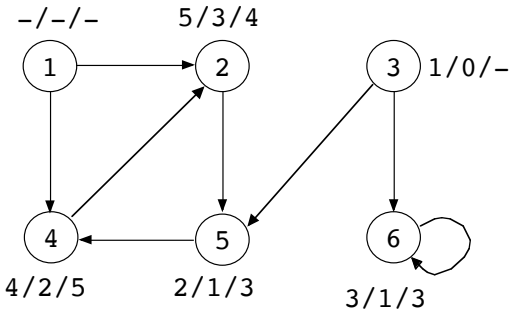
```
for i = 1 to V
  for j = 1 to V {
     G²[i][j] = 0;
     for k = 1 to V
       if (g[i][k] == 1 && g[k][j] == 1) {
           G²[i][j] == 1;
           break;
       }
  }
```

G^2 for an adjacency list:
Procedure G-Square (V[G], E[G])

```
V[G²] ← V[G]
for each u ∈ V[G]
     for each v ∈ Adj[u]
          for each w ∈ Adj[v]
               E[G²] ← {(u, w)} ∪ E[G²]
```

Run time = $O(V^3)$

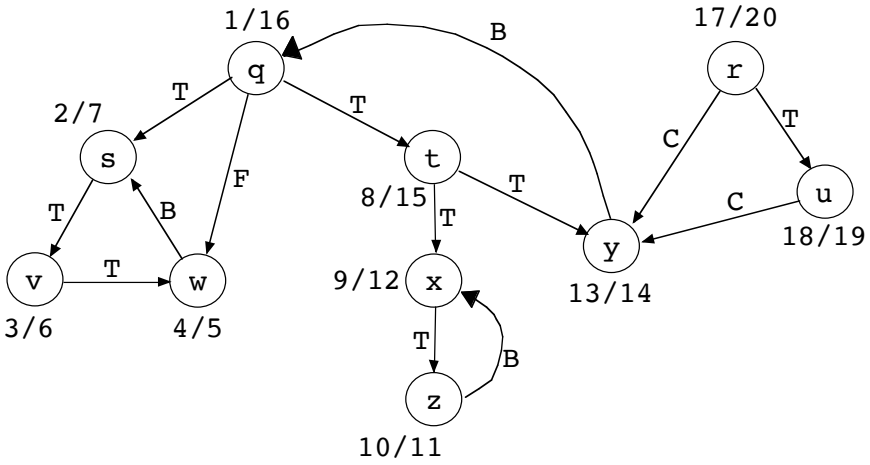
3. 22.2-1 Show the d and Π values that result from running breadth-first search on the directed graph of Figure 22.2(a), using vertex 3 as the source.



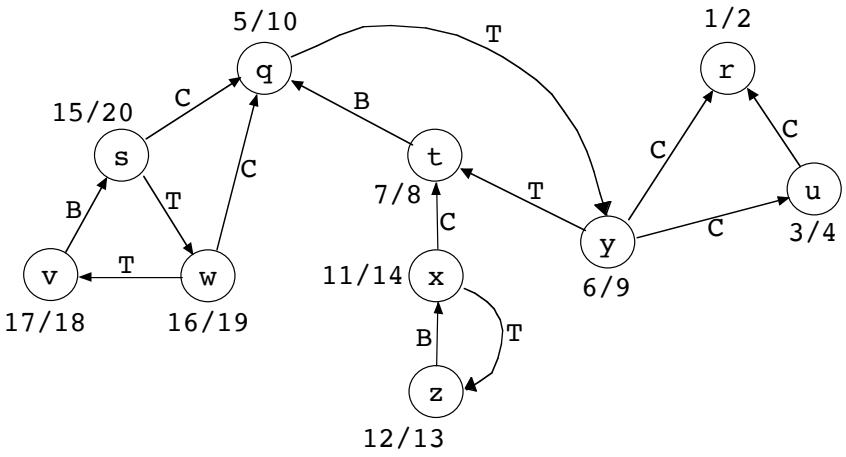
4. 22.2-4 What is the running time of BFS if its input graph is represented by an adjacency matrix and the algorithm is modified to handle this form of input?

Each vertex can be explored once and its adjacent vertices must be determined too. This takes $\Theta(V^2)$ time.

5. Do a DFS on figure 22.6 (p.611). Classify each edge based on the DFS tree you determine.

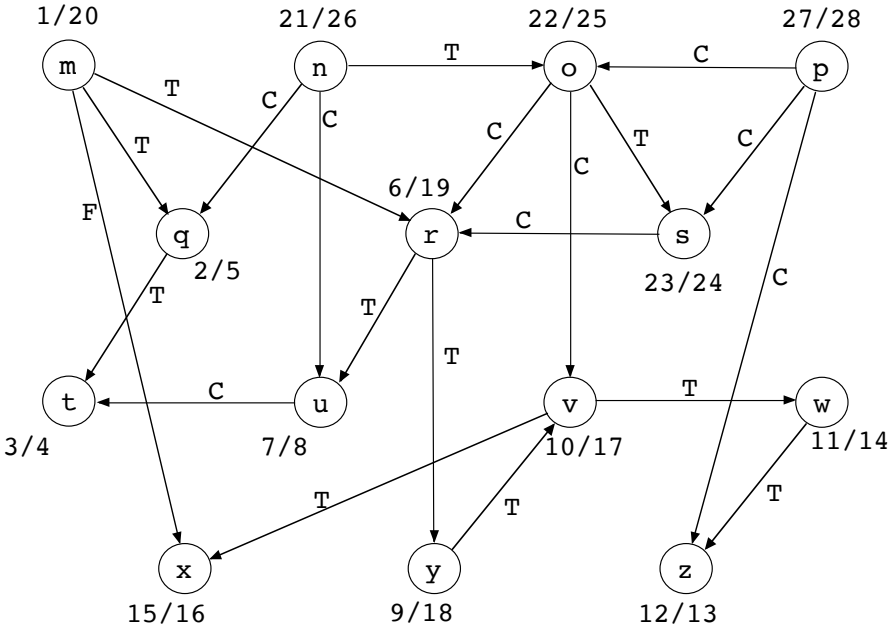


6. Find the strongly connected components in figure 22.6.
From 5., the first DFS gives the list R U Q T Y X Z S U W (reverse order of turning black)



Strongly connected components are {s, w, v}, {q, y, t}, {x, z}, {r}, {u}

7. 22.4-1 Show the ordering of vertices produced by TOPOLOGICAL-SORT when it is run on the dag of Figure 22.8, under the assumption of Exercise 22.3-2.

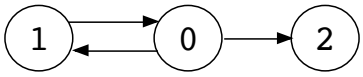


p n o s m r y v x w z u q t

8. 22.5-1 How can the number of strongly connected components of a graph change if a new edge is added?

The number of strongly connected components can be reduced.

9. 22.5-3 Professor Bacon claims that the algorithm for strongly connected components can be simplified by using the original (instead of the transpose) graph in the second depth-first search and scanning the vertices in order of increasing finishing times. Is the professor correct?

Consider 

For this algorithm, the first DFS will give a list 1 2 0 for the second DFS. All vertices will be incorrectly reported to be in the same SCC.

For the CLRS algorithm, the first DFS will give a list 0 2 1 for the second DFS. After reversing edges, the correct SCCs {0, 1} and {2} will be reported.

10. 22.5-4 Prove that for any directed graph G, we have $((G^T)^{SCC})^T = G^{SCC}$. That is, the transpose of the component graph of G^T is the same as the component graph of G.

Since the strongly connected relationship is an equivalence relation, G and G^T will always have the same strongly connected components. If two vertices X and Y are not strongly connected, then there is a unique path from X to Y in G iff there is a unique path from Y to X in G^T.

A similar property will also hold for the component graph and transpose of the component graph, i.e. If two vertices X and Y are not strongly connected in G, then here is a unique path from SCC(G,X) to SCC(G,Y) in $(G)^{SCC}$ iff there is a unique path from SCC(G^T,Y) to SCC(G^T,X) in $(G^T)^{SCC}$).

11. 23.1-1

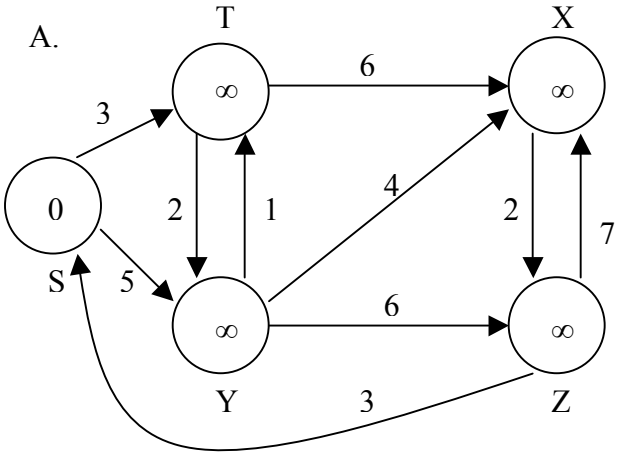
Let (u, v) be a minimum-weight edge in a graph G. Show that (u, v) belongs to some minimum spanning tree of G.

Let A be a subset of some MST T such that (u, v) $\notin$ A. To choose an edge to be added to A, all the edges on the cut are considered and an edge with lowest weight is selected. Since (u, v) is the minimum weight edge in the graph G, it gets selected on some cut.

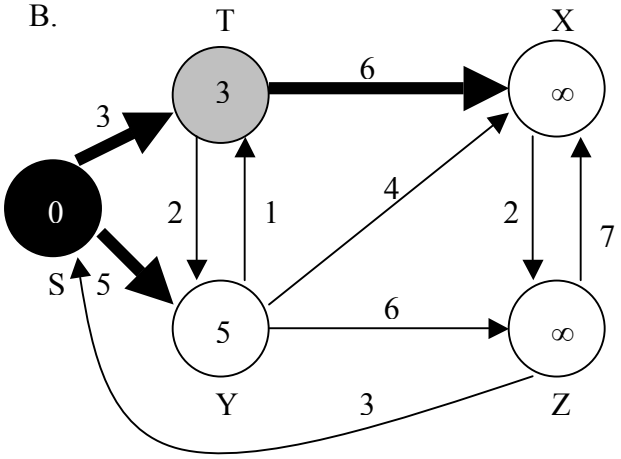
12. 24.3-1 Run Dijkstra's algorithm on the directed graph of Figure 24.2, first using vertex s as the source and then using vertex z as the source. In the style of Figure 24.6, show the d and Π values and the vertices in set S after each iteration of the while loop.

Black vertices are in the set S. The black, thick arrows are the values of Π and the values of d are included are listed inside each node.

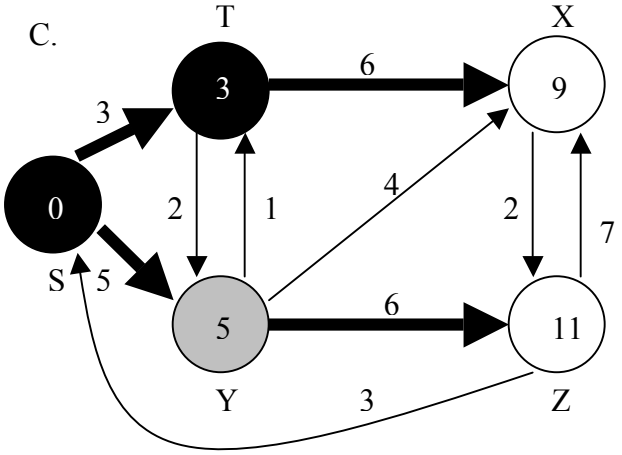
A.



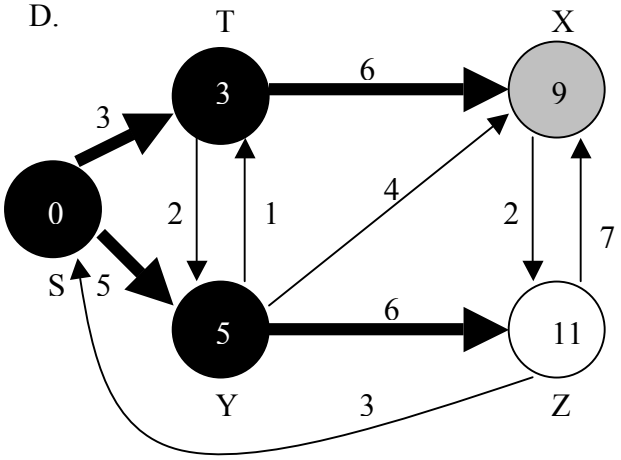
B.



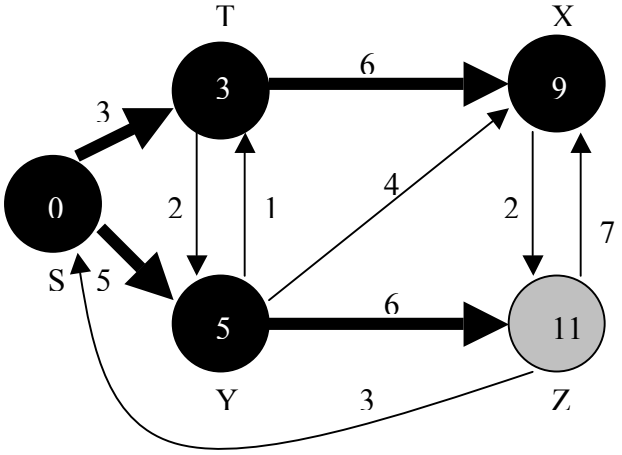
C.



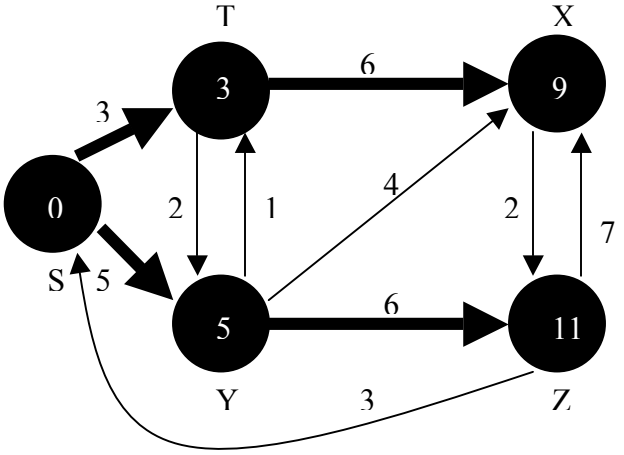
D.



E.

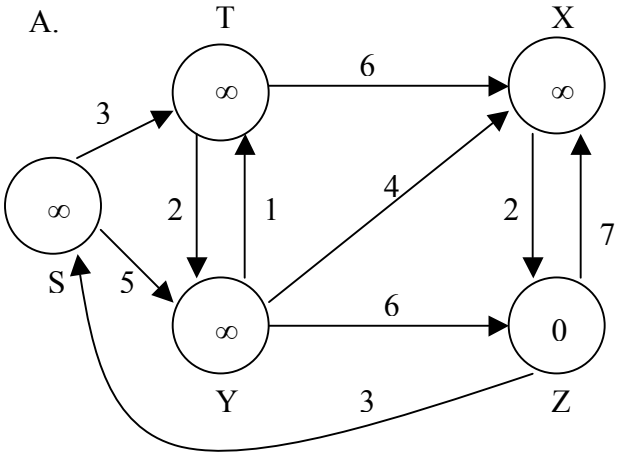


F.

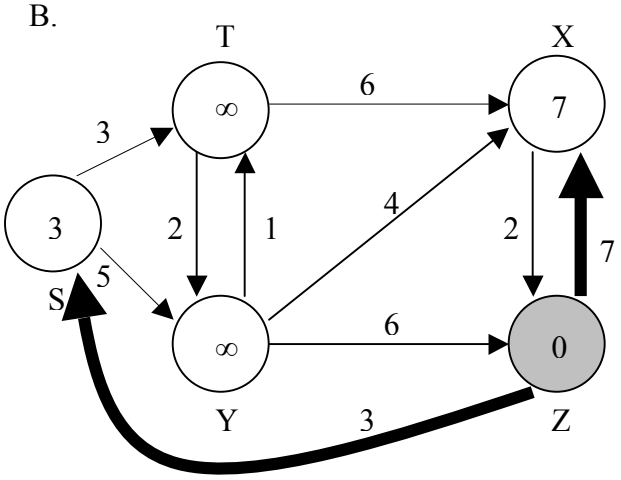


Using Vertex Z as the source.

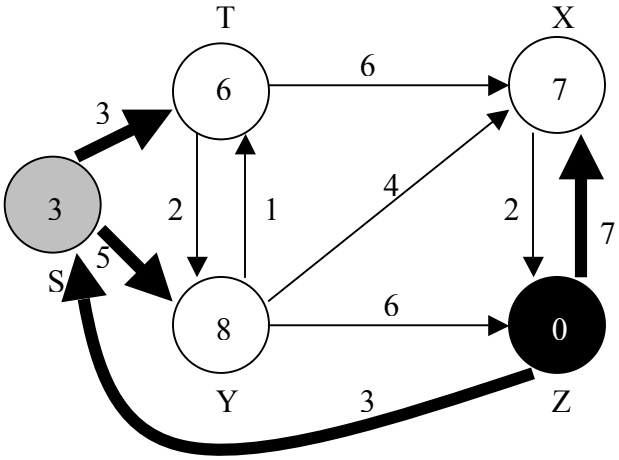
A.



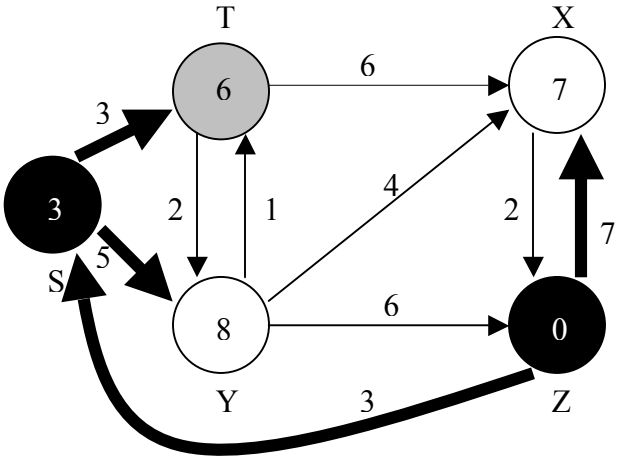
B.



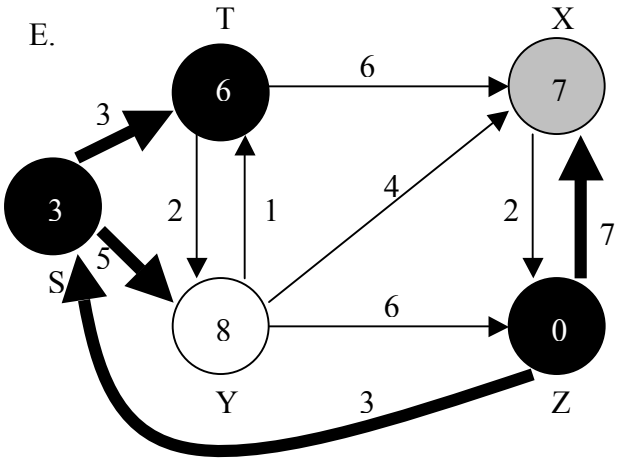
C.



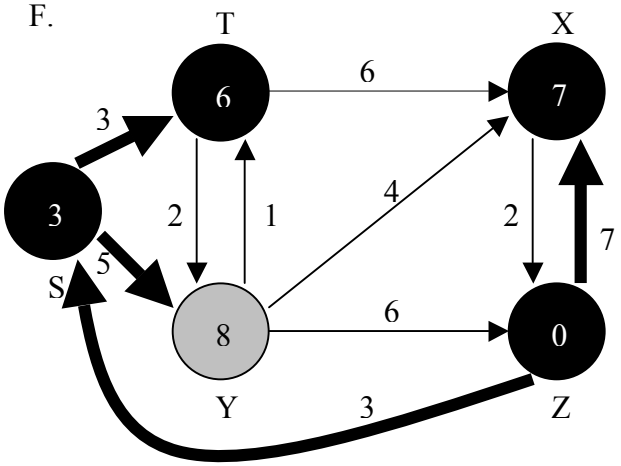
D.



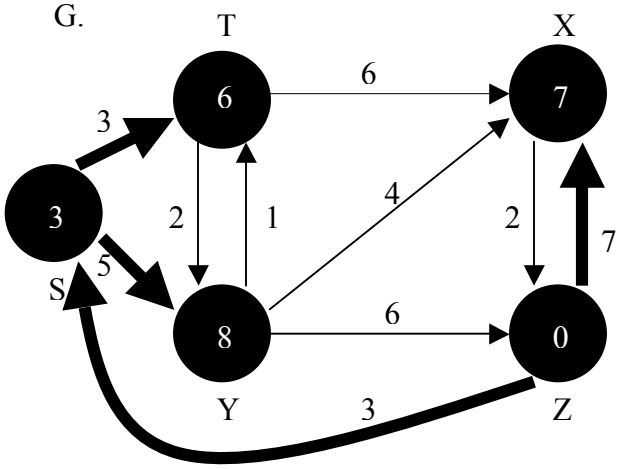
E.


T
X
6
7
3
3
2
1
4
2
7
5
S
8
6
0
Y
3
Z

F.


T
X
6
7
3
3
2
1
4
2
7
5
S
8
6
0
Y
3
Z

G.


T
X
6
7
3
3
2
1
4
2
7
5
S
8
6
0
Y
3
Z

13. a. Determine the transitive closure of the following Boolean matrix by using Warshall's algorithm.

1	0	1	1	0
0	0	1	1	0
1	0	0	0	0
1	0	1	1	1
0	0	0	1	0

T(1) =

1	0	1	1	0
0	0	1	1	0
1	0	1	1	0
1	0	1	1	1
0	0	0	1	0

T(2)=

1	0	1	1	0
0	0	1	1	0
1	0	1	1	0
1	0	1	1	1
0	0	0	1	0

T(3)=

1	0	1	1	0
1	0	1	1	0
1	0	1	1	0
1	0	1	1	1
0	0	0	1	0

T(4)=

1	0	1	1	1
1	0	1	1	1
1	0	1	1	1
1	0	1	1	1
1	0	1	1	1

T(5)=

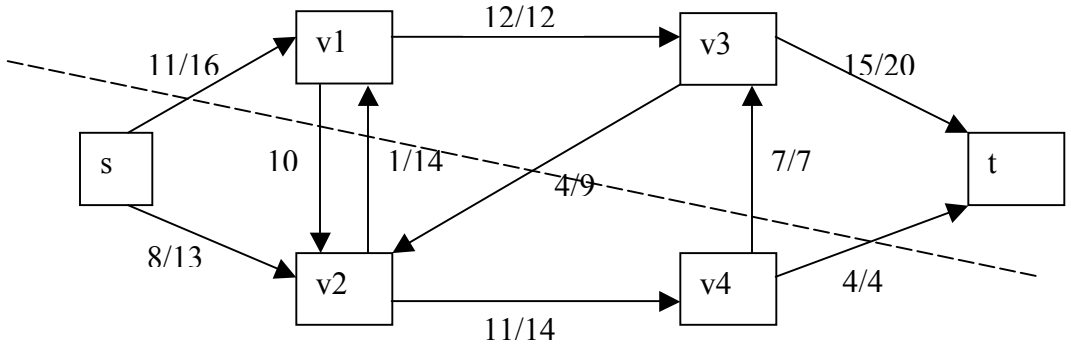
1	0	1	1	1
1	0	1	1	1
1	0	1	1	1
1	0	1	1	1
1	0	1	1	1

b. Convert the matrix to indicate successors and use the version of Warshall's algorithm that allows path tracing.

0	-1	2	3	3
2	-1	2	3	3
0	-1	0	0	0
0	-1	2	3	4
3	-1	3	3	3

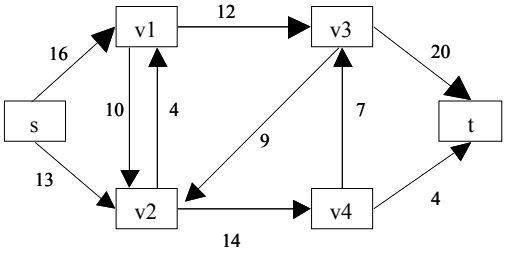
14. 26.2-2

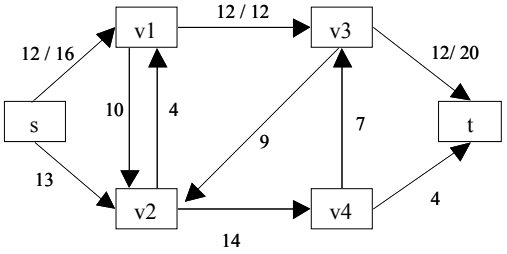
In Figure 26.1(b), which is the flow across the cut ({s, v_2, v_4}, {v_1, v_3, t})? What is the capacity of this cut?

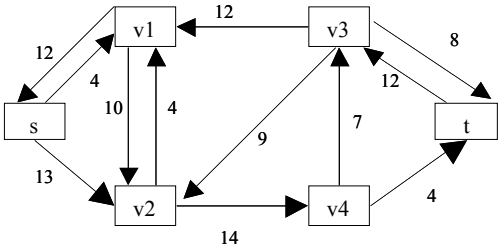


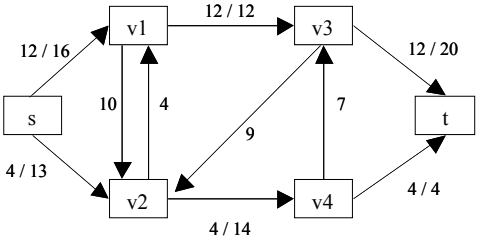
Net flow across the cut is f(s, v1) + f(v2,v1) + f(v2,v3) + f(v3,v4) + f(v4,t) = 11 + 1 + -4 + 7 + 4 = **19** and its capacity is = 16 + 14 +7 + 4 = **41**

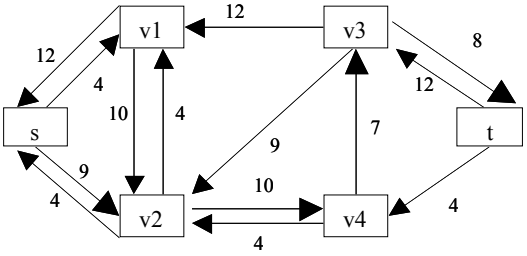
16 26.2-3

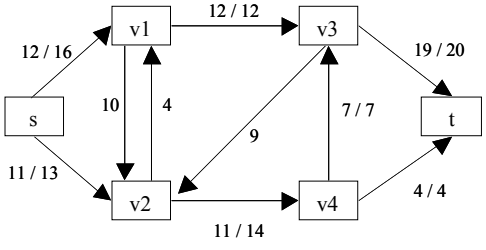


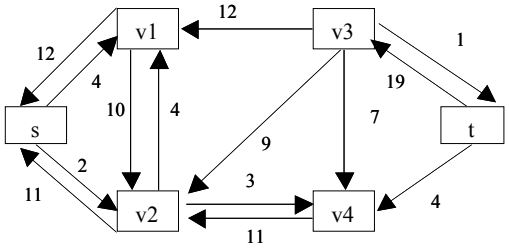




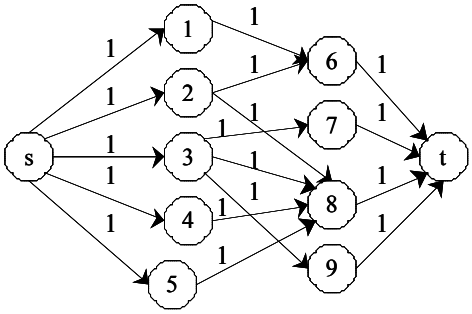








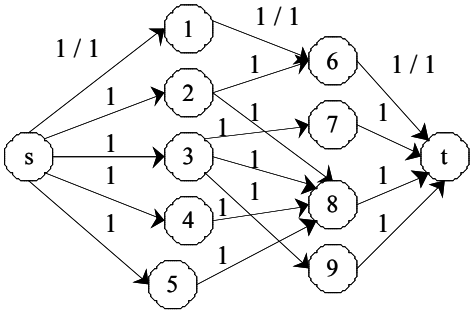
17 26.3-1



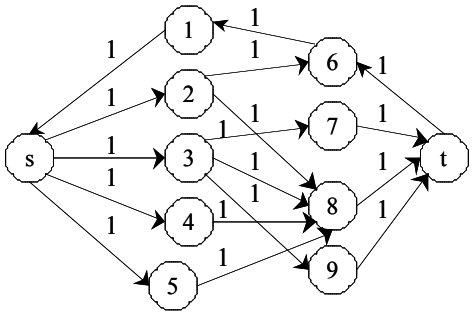
Augmenting path

S ? 1 ? 6 ? t

Flow N/W:

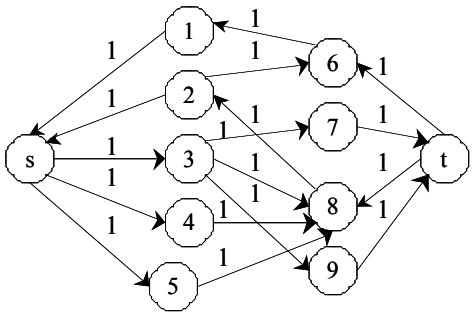


Residual N/W:



Augmenting path

S ? 2 ? 8 ? t



Augmenting path

S ? 3 ? 7 ? t

Final Residual graph: