ICS 260 – Fall 2001 – First Midterm

Name: ANSWER KEY

Student ID:

1: 20

2: 20

3: 20

4: 20

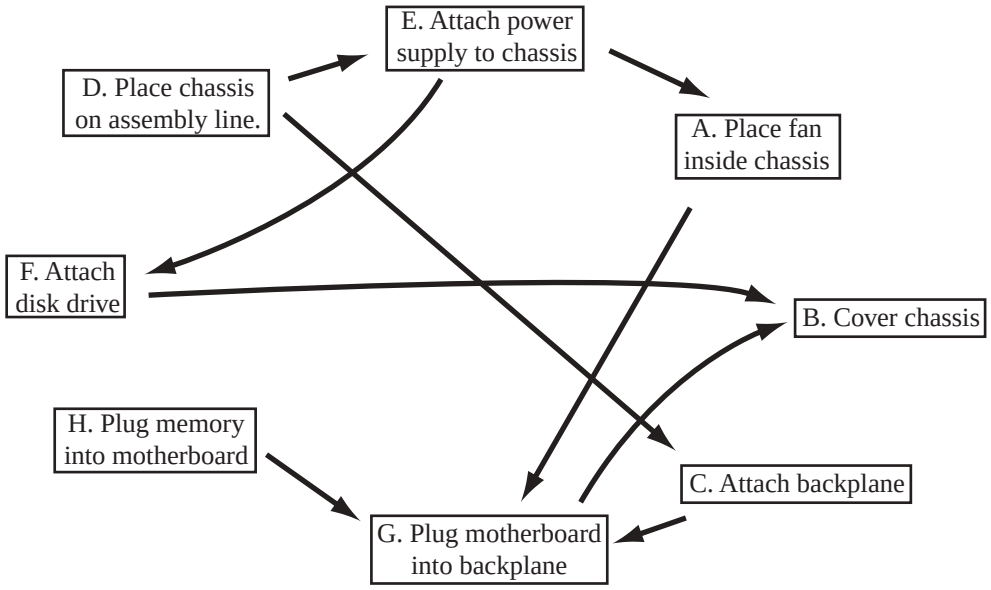
5: 20

6: 20

Total: 120

1. Assembly line scheduling (20 points)

(a) In the following graph, boxes represent tasks that must be performed in the assembly of a computer, and arrows represent constraints that one task must be performed before another (for instance, the disk drive must be attached before the chassis can be covered). Write down a sequence in which all of these tasks can be performed, satisfying all constraints.



D E A F H C G B
(there are many other possible answers)

(b) Circle the phrase among the following that best describes an abstract algorithmic problem corresponding to this assembly line scheduling application.

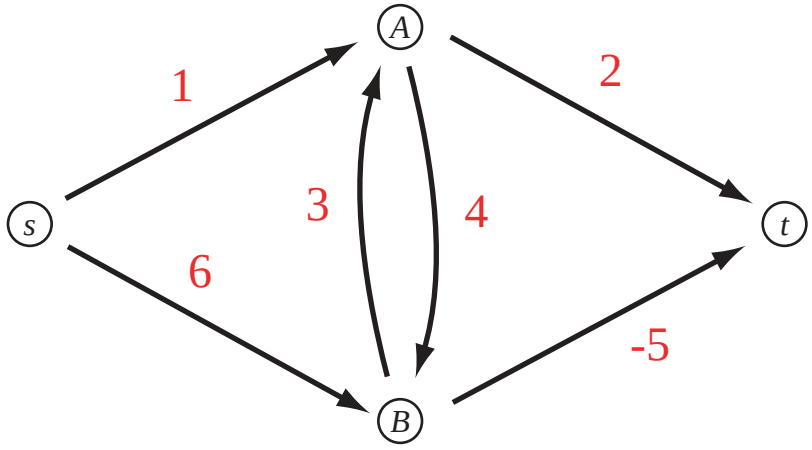
- Breadth first search
- Depth first search
- Maximum flow
- Shortest paths
- TOPOLOGICAL ORDERING
- Uniform packing

(c) If there are t tasks and c constraints, what is the best time bound for solving problems like the one in part (a)? Use O-notation.

$O(t+c)$

2. Shortest paths. (20 points)

(a) Write down lengths for the edges of the following graph, so that Dijkstra's algorithm would not find the correct shortest path from *s* to *t*.



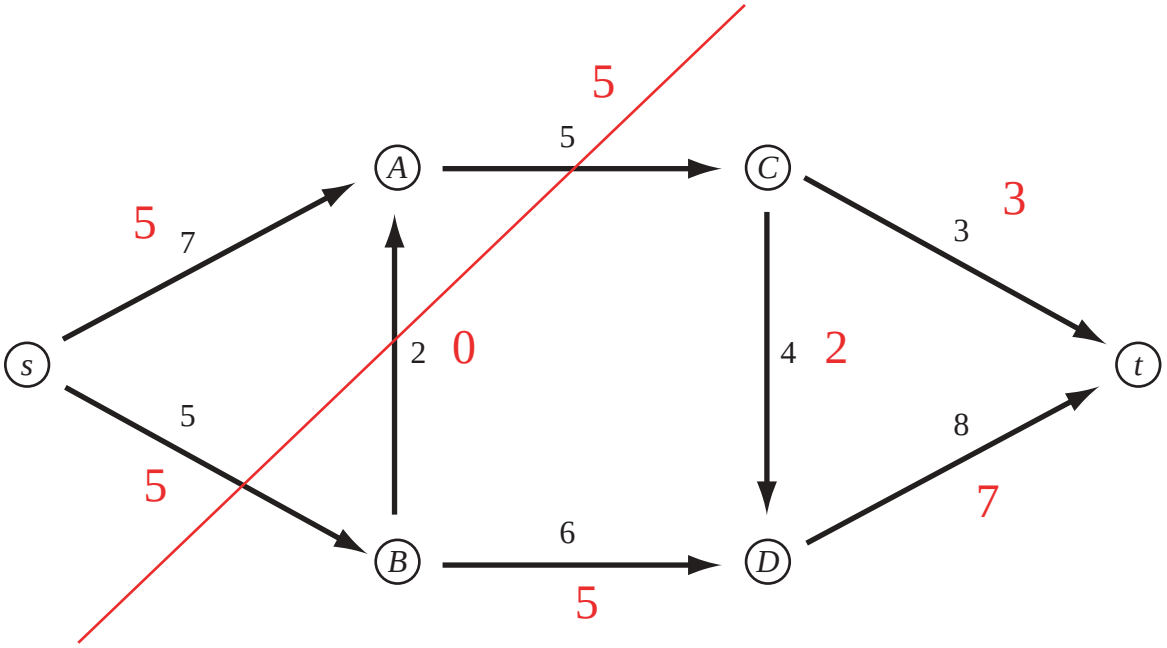
With the weights shown, Dijkstra's algorithm will visit the vertices in the order *s* – *A* – *t* (since the priority of *t* at the last step is 3 while the priority of *B* is 5), so it will return the path *sAt* with length 3 and not find the path *sBt* with length 1.

(b) Which of the shortest path algorithms described in class would be most appropriate for finding paths in the graph of part (a) with the weights you gave? Explain your answer.

Bellman-Ford, because it can handle graphs with negative edge weights and cycles. We can't use the DAG algorithm because this graph is not a DAG.

3. Flow. (20 points)

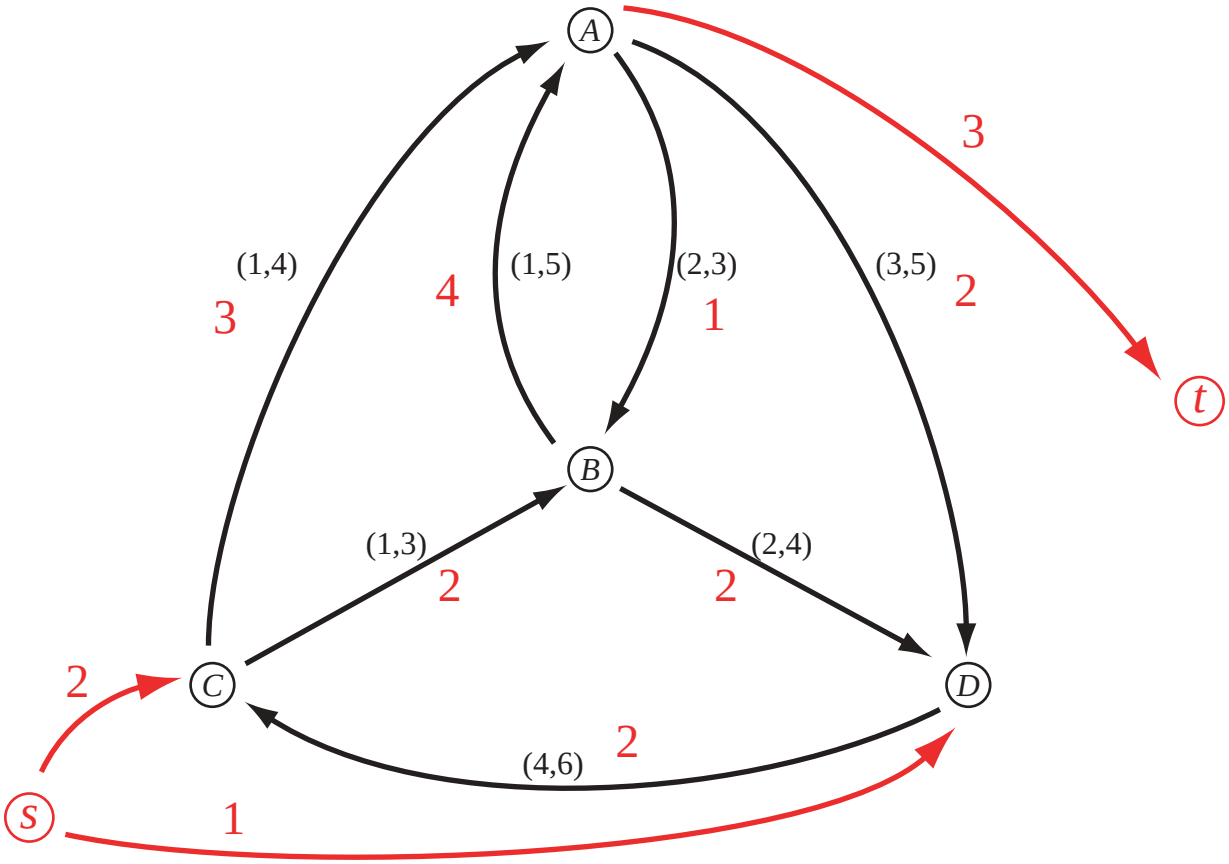
(a) The numbers on the graph below represent edge capacities. Write down flow amounts for a maximum flow in this graph.



(b) Draw or describe the minimum cut for this graph.

4. Feasible circulation. (20 points)

(a) Translate the following feasible circulation problem into a maximum flow problem. Do not solve the flow problem. You may show your answer by adding vertices and edges to the given graph, and by writing capacities on each edge.

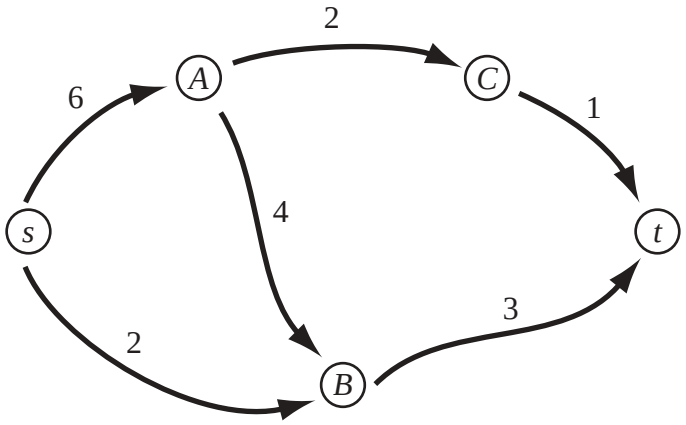


(b) Explain how you would determine whether there is a feasible circulation, if you were given the flow amounts of a maximum flow for your answer to part (a).

If the flow saturates all the edges out of *s* or in to *t* (which will happen in this example if there are three units of flow) then the circulation will be feasible.

5. Preflow push algorithm. (20 points)

Suppose that we are trying to compute a maximum flow from s to t in the following graph (with indicated capacities) using the generic preflow-push algorithm.



(a) For the preflow amounts and distance labels shown in the tables below, fill in the column showing the excess at each vertex

edge	flow amount
$s \to A$	6
$s \to B$	2
$A \to B$	2
$A \to C$	2
$B \to t$	3
$C \to t$	1

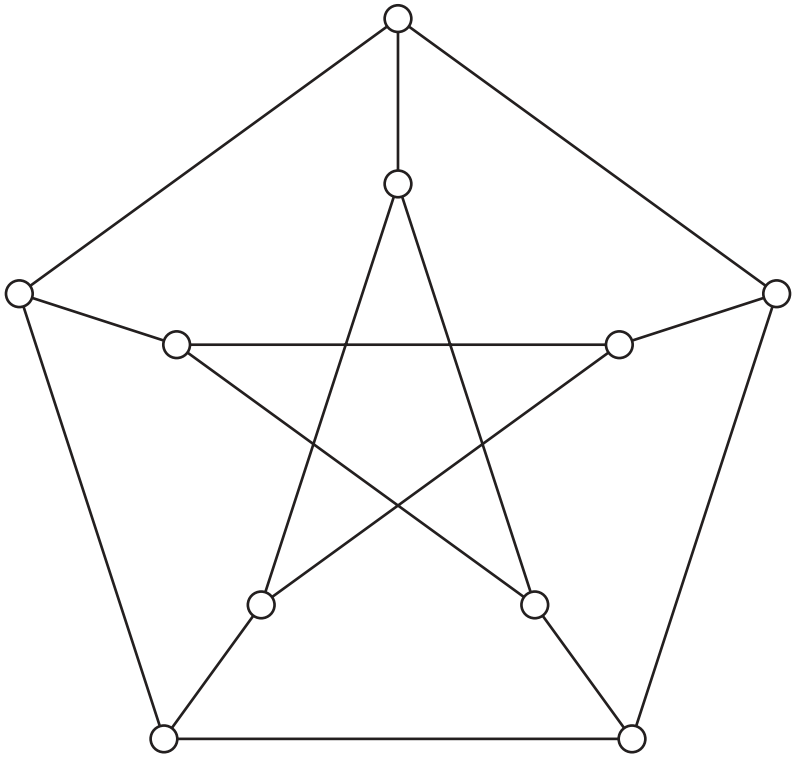
vertex	distance label	excess?
s	5	0
A	2	2
B	3	1
C	1	1
t	0	0

(b) Describe one possible set of changes that could be made to the preflow amounts or distance labels of part (a) by a single step of the generic preflow-push algorithm.

There are three vertices with nonzero excess: A, B, and C. If you chose A, there is no admissable arc (the one from A to C is saturated so not part of the residual graph) and you would change its distance label to 4. If you chose B, there is an admissable arc from B to A in the residual graph, so you would reduce the flow amount on edge A-B from 2 to 1. If you chose C, there is no admissable arc, but there is a residual edge from C to A, so you would change C's distance label to 3. (You only needed to describe one of these three possibilities.)

6. Matching and independent sets. (20 points)

Would the maximum independent set algorithm described in the lecture on matching work for the following graph? Why or why not?



No, because it is not bipartite.

You didn't need to include this in your answer, but in this graph the largest independent set has size four, while the largest matching has five edges; in a bipartite graph these two numbers would add up to the total number of vertices but here they don't. This symmetric ten-vertex graph is known as the Petersen graph.