CS5800 Algorithms ***A***

Solutions for Exam on November 15, 2019

Ravi Sundaram

- **Logistics:** Read all these instructions. Write your name on the top of every page. Write your name in uppercase in the blank provided on this page. If you need extra space, use the blank facing page.
- **Duration, Problems and Points:** This exam lasts 90 minutes and has 5 problems totaling 100 points. Make sure to read all problems to decide an order to attempt them that would allow quickest progress.
- **Known algorithms:** While describing a new algorithm, you may use any algorithm covered in class as a subroutine without elaboration.
- **Proofs and analyses:** You do not need to provide a proof of correctness and/or an analysis of the running time *unless* explicitly asked to.
- **Partial Credit:** Points are determined on the basis of both the correctness and the clarity of description. Show your work, as partial credit may be given. Present the best algorithm you can, even if it doesn't meet all desired properties.
- **Closed-book exam:** This exam is closed-book. No sources of any kind, electronic or physical, can be consulted during the course of this exam. No collaboration of any kind is permitted.
- **Policy on cheating:** Students who violate the above rules on scholastic honesty are subject to disciplinary penalties. Any student caught cheating will *immediately* receive an **F** (failing grade), and the case will be forwarded to the Office of Student Conduct and Conflict Resolution.

  Sign below and confirm that you are strictly abiding by the rules of this test.

Signature: ____________________

NAME: ____________________

Question	1	2	3	4	5	Total
Points	10	20	20	20	30	100

Best of luck!

Problem 1 (True/False) 10

Indicate whether the following statements are True or False by filling in the table with T or F respectively:

a) For a directed Graph $G = (V, E)$ with capacities on edges and vertices s and t, every flow from s to t is at most the capacity of any $s - t$ cut of G.

b) If the capacities on the edges of a graph are all integral, then the maximum flow and the minimum cut found by the Ford-Fulkerson algorithm are both integers.

c) Given a network flow graph with irrational flow values, the Ford-Fulkerson algorithm is guaranteed to terminate.

d) A flow f is a max flow if there are no augmenting paths in the residual graph..

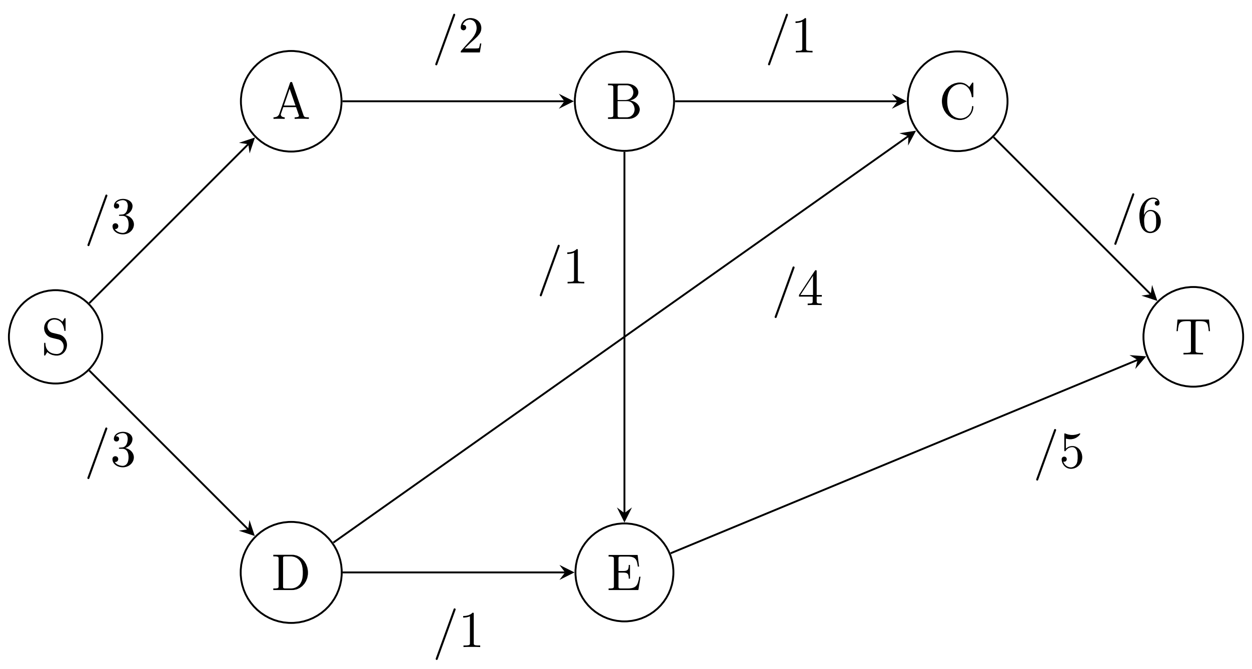
e) The problem of finding the maximum number of edge-disjoint paths, the maximum number of node-disjoint paths, and maximum bipartite matching can all be solved via a reduction to maximum flow.

Solution:

Question	a	b	c	d	e
T/F	T	T	F	T	T

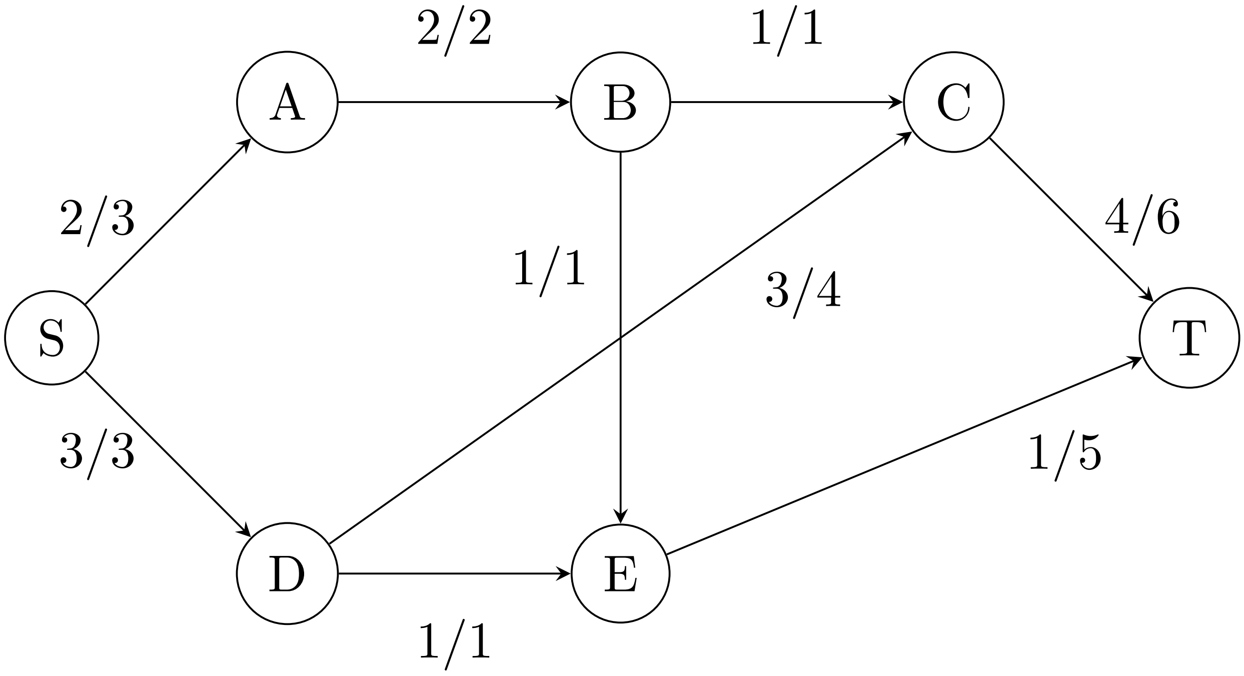
■

Problem 2 (Using Edmonds-Karp algorithm, calculate the maximum flow through the graph shown below. Show the augmenting path and residual flow after each iteration.) 20



Solution:

A possible solution:



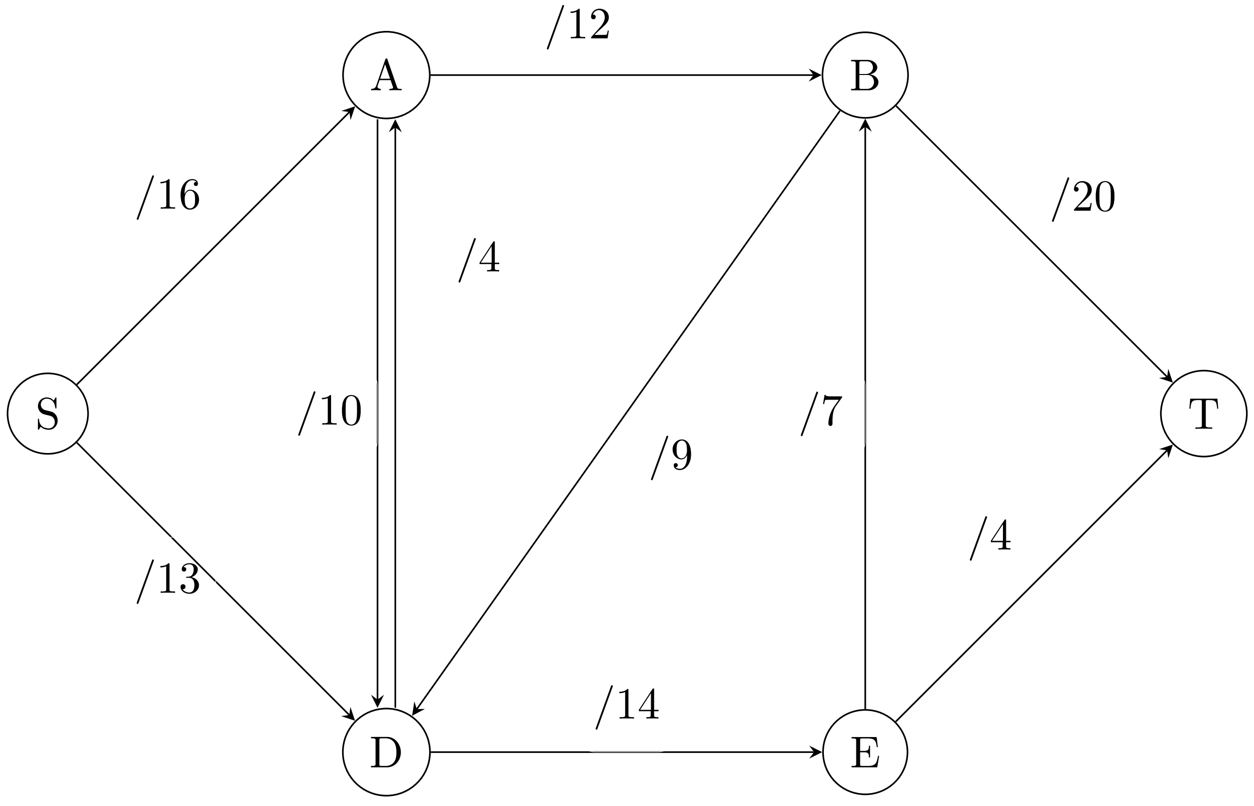
Path : Flow
$S \to D \to C \to T : 3$
$S \to A \to B \to C \to T : 1$
$S \to A \to B \to E \to T : 1$

■

Page left blank intentionally (use for scratch work)

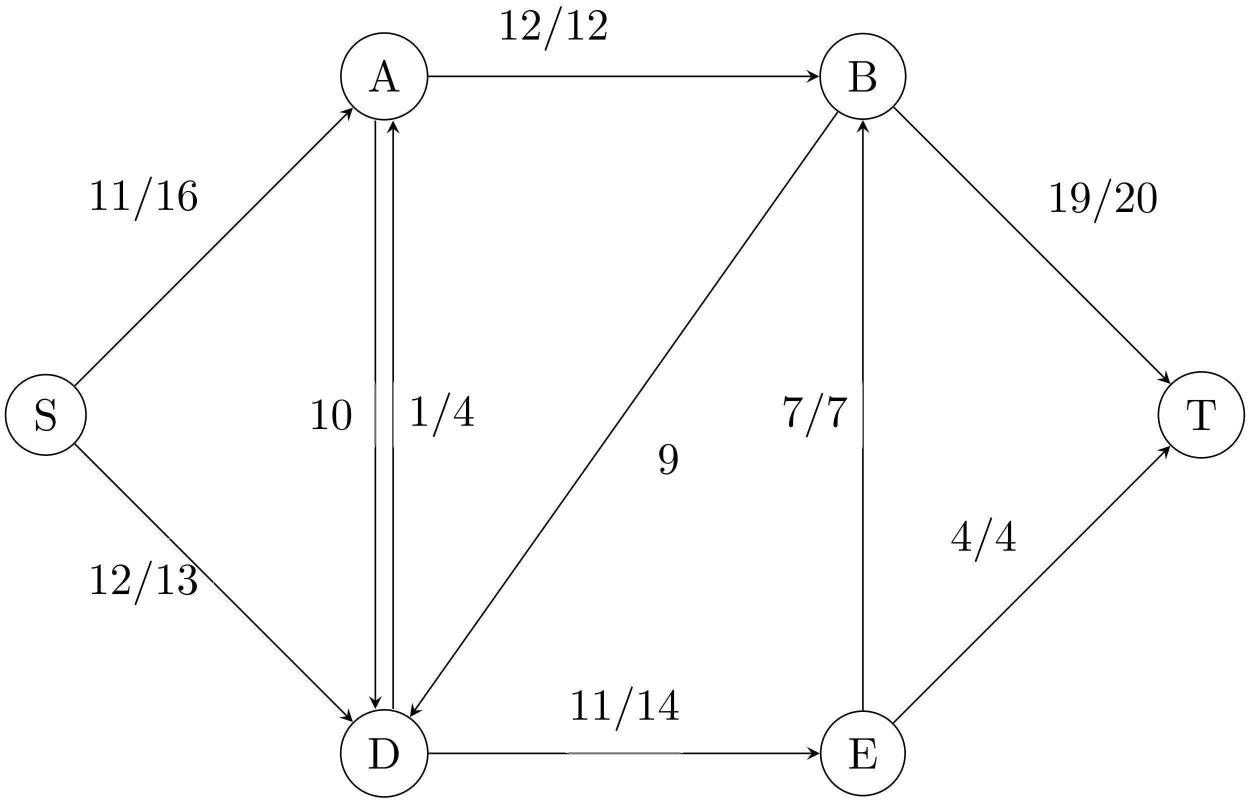
Problem 3 (Electrical Power Distribution) 20

You are an engineer working in an electrical power distribution company. The graph below shows a high voltage electrical network operated by your company, in which S generates and sends electrical power in Megawatts to various cities. Nodes represent cities and edges represent power lines. Sometimes, certain events may cause the power output of a generator to spike, which could overload all connected nodes downstream in the network. This is called a cascading effect and can cause huge loss to equipment if not checked. Your job is to identify possible powerlines from the set $L = \{SA, AD, DA, BD, DE\}$ on which to install circuit breakers, in order to minimize the cascading effect from reaching T during peak power flow through the network. As high voltage circuit breakers are expensive, you want to use as few of them as possible. Show the maximum flow of electrical power through this network and suggest a subset of L to install these breakers on.



Solution:

A possible solution:

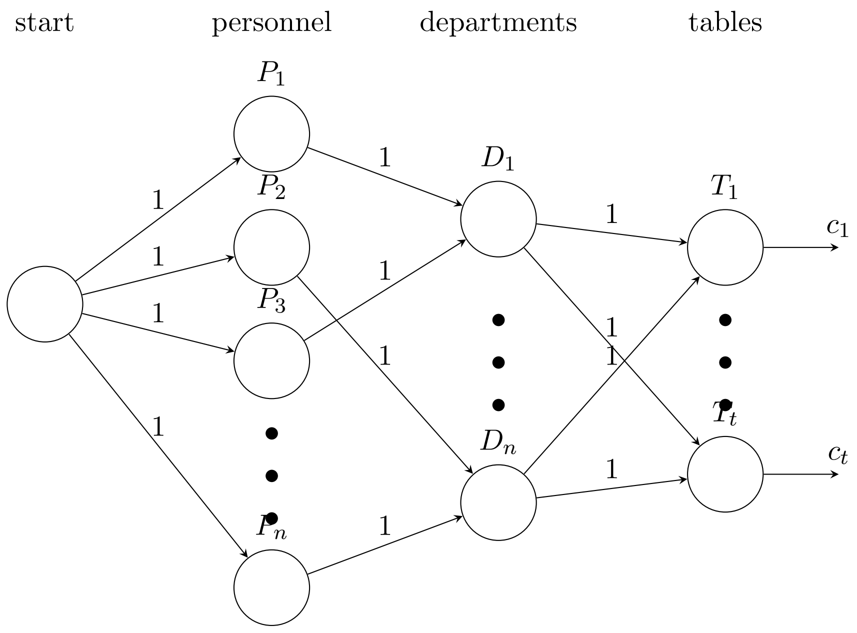


$\{SA, DA, DE\}$ ■

Problem 4 (Table arrangements) 20

You are organizing a big dinner party for your company retreat. To increase social interaction among the various departments in your company, you would like to set up a seating arrangement so that no two members of the same departments are at the same table. Show how to formulate the problem of finding a seating arrangement that meets this objective as a maximum flow problem. Assume that the company has n departments and department i has m_i members. Also assume that t tables are available and that the table j has a seating capacity of c_j.

Solution:



Number of Edges $|E| = (n + t + nt)$
Number of Vertices $|V| = (2 + n + t)$
Running time $= O(|V||E|^2) = O((n + t)(n + t + nt)^2)$ ■

Problem 5 (Subset Sum)) **30**

You are given a set of positive integers $A = \{a_1, a_2, \ldots, a_m\}$ and a non-negative integer target value n. You need to determine if there is a subset $S \subseteq A$ such that the sum of the elements in S equals the target n.

For example, if $A = \{2, 3, 5\}$ and $n = 8$ then we output TRUE since $S = \{3, 5\}$ is a valid subset that adds up to n. However, for $A = \{2, 3, 5\}$ if the input $n = 6$ then we output FALSE since no subset of A adds up to n.

Come up with a dynamic programming algorithm that given a set A and a target value n correctly outputs if there is a subset $S \subseteq A$ that sums up to n.

Note: *Use the DP template: Brief English description* of your logic*, Bellman Equation (including the base case), iterative approach, number of subproblems, time for each subproblem, final running time, final value to return.*

Solution:

The problem can be broken up into smaller sub-problems which require a subset of A that sums upto a number smaller than n.

Let $S[i][j]$ be *True* if there is a subset of $\{a_1, a_2, \ldots, a_j\}$ which adds up to i. Then, we want to know if $S[n, m]$ is $True$.

Bellman equations:

$$S[i, j] = \begin{cases} S[i - a_j, j - 1] \vee S[i, j - 1] \text{ if } i - a_j \geq 0 \\ S[i, j - 1] \text{ otherwise} \end{cases}$$

where $\forall 1 \leq j \leq m, S[0, j] = True$

Iterative approach: Fill a table S rowwise from row 0 upto row n using the given Bellman equations.

There are n*m subproblems, each of which takes time $O(1)$ to solve.
Hence the total running time is $O(n * m)$. Return S[n,m] in the end. ■