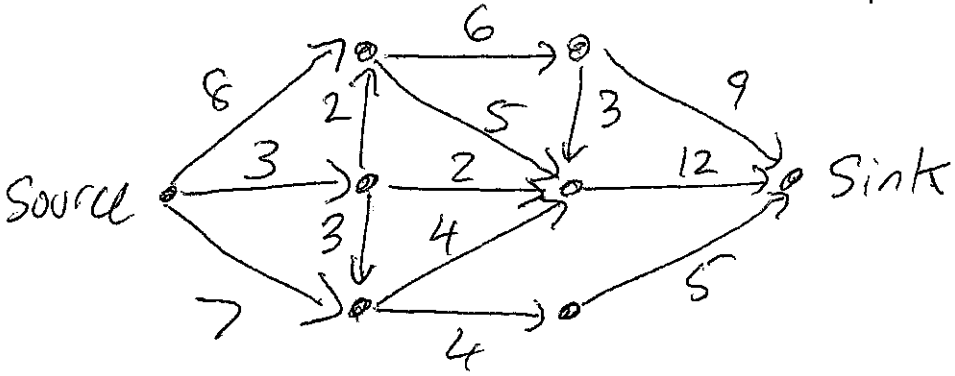
Network Flow

Printed Notes (9 pages)
Ford Fulkerson.java (extremely succinct)

Graph

number = capacity



Max Flow Problem

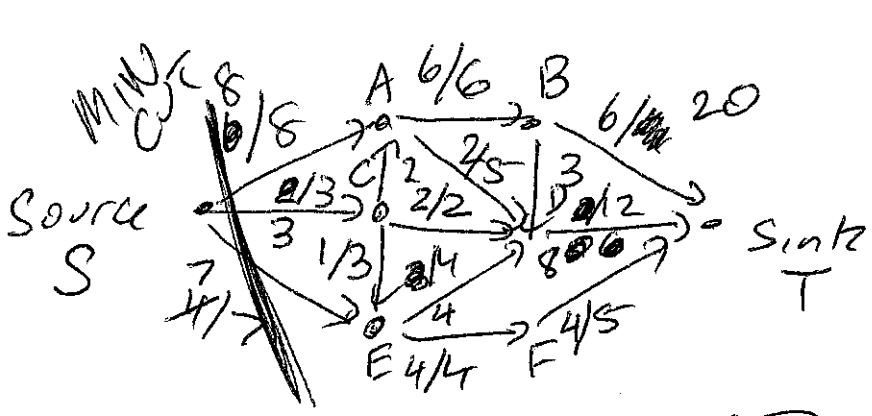
Generic Solution

While (an augmenting path exists)
    send flow through that
    augmenting path

Augmenting Path is a valid path to add some flow from source to sink

TWO TYPES OF EDGES

(1) forward
(2) backward



CUT SPACE/BDFT

A → B 6 3/8/15(2)
C → D 2
E → D 4
E → F 4
A → D 5

MAX FLOW ≤ 6+2+4+4+5 = 21

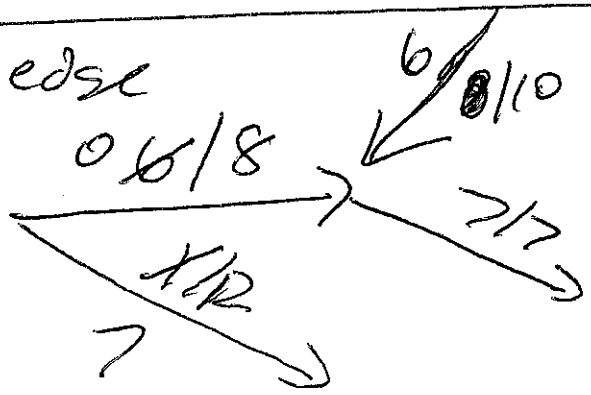
S → A → B → T	6
S → C → D → T	2
S → E → F → T	4
S → E → D → T	3
S → C → E → D → T	1
S → A → D → T	2

18

MAXFLOW ≤ VALUE OF ANY CUT

MAXFLOW = MINCUT

Backedge

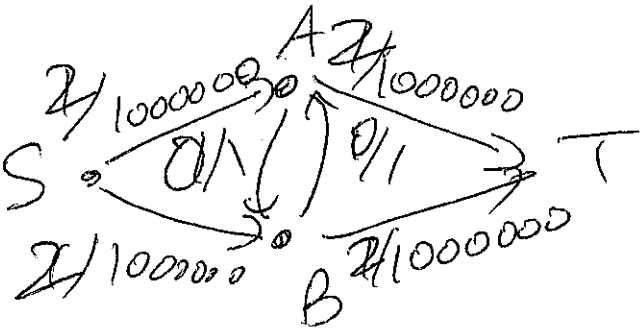


Purpose of a backedge - to allow you to FIX MISTAKES!!!

When looking for an augmenting path, I have to also look for backedges

HOW TO FIND AUGMENTING PATHS? DFS or BFS

Ford-Fulkerson - DFS (might be exponential)
Edmunds-Karp - BFS (polynomial)
Dinic's - BFS and DFS (fastest practice)

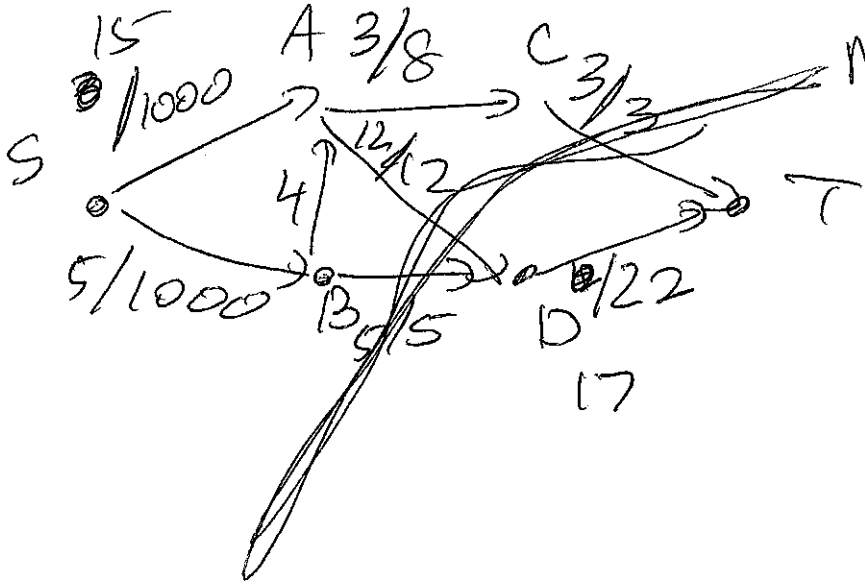


Bottleneck - WHAT ROADS ARE PREVENTING MORE TRAFFIC ~~FROM~~ FROM MOVING THROUGH?

MINUMUM CUT PROBLEM

CUT is separating (partitioning) the vertices into 2 sets, one that contains S and another that contains T.

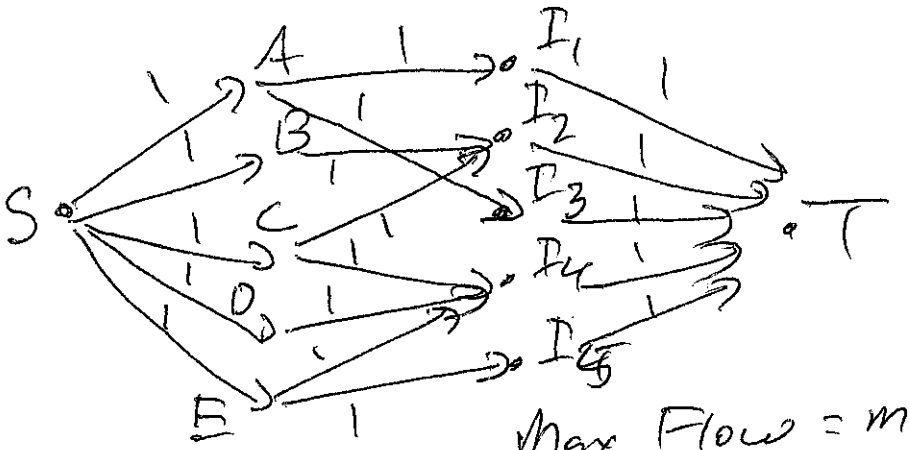
Old Graph: {S, A, C, E}, {B, D, F, T}



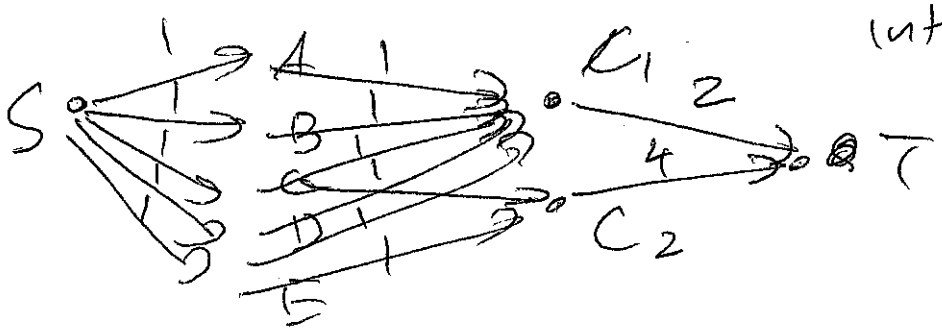
Max Flow Apps

3/8/18

Bipartite Matching



Max Flow = most people who could get an internship.



Museum Guard

48 shifts (30 minutes each)

12-12:30

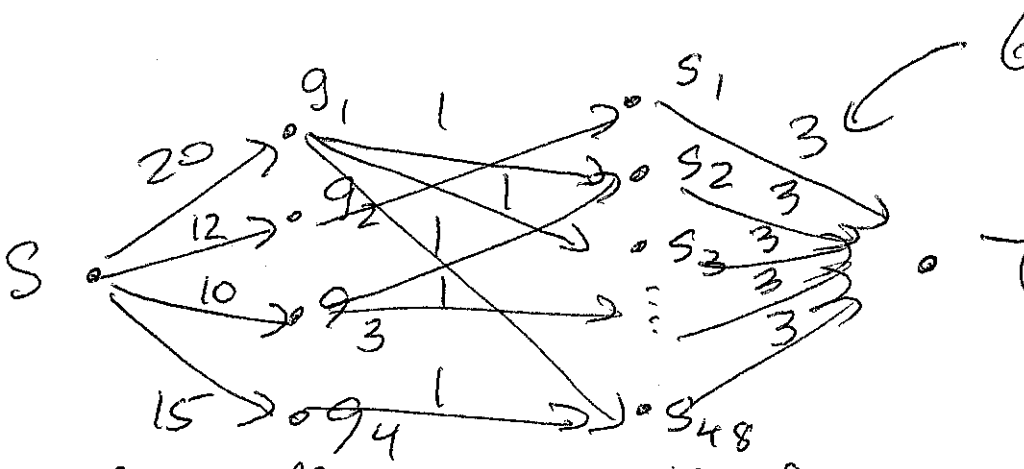
12:30-1 ...

GOAL: Maximize the min # of guards at any shift.

(4,12,1,8) (1)

(3,2,4,2) (2)

Guards - each guard (max # shifts) specific shifts they can make it.



Run Max Flow If flow = 3×48
= 144
then it's possible to schedule 3 guards for each shift.
If it's less it's impossible.