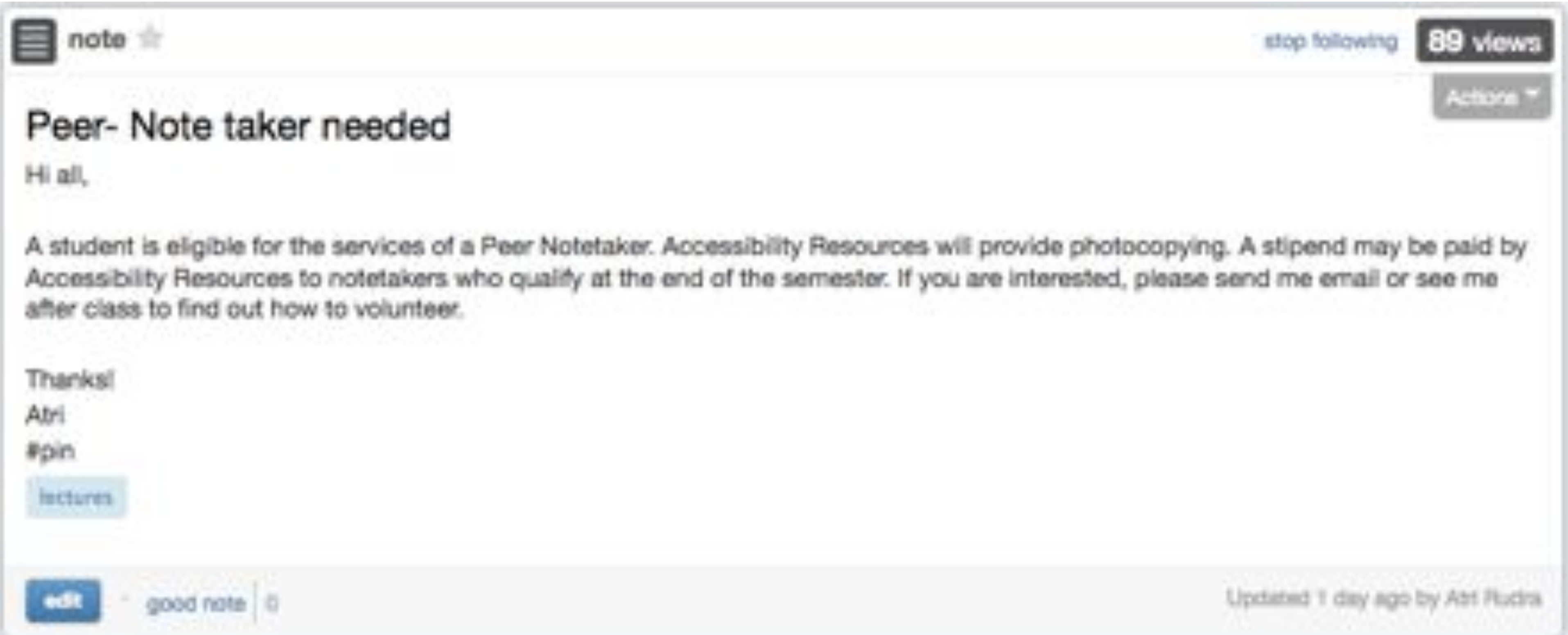
Lecture 14

CSE 331

Sep 30, 2016

Peer Notetaker needed



Mini Project Pitch due WED

note

stop following 59 views

Actions

Mini Project pitch can now be submitted

You can now submit your mini project pitches on Autolab. The deadline is 11:59pm on Wed, Oct 5. More details on the mini project page:

http://www-student.cse.buffalo.edu/~atri/cse331/fall16/mini-project/index.html

The case studies will be assigned on a first come first serve basis. So once your pitch is ready for grading, send me an email. I plan to grade them on a rolling basis.

Important: Since this will be a group submission, things will work a bit differently than for the current submissions that are done individually. In particular, you should form your group **before** submitting your pitch. To create your group do the following:

1. One person in your group has to create it first. To do this, click on the "Group Options":

Options

- View writeup
- Download handout
- View handin history
- Group options

Form your group on Autolab BEFORE submitting your pitch

Do not forget to add URL to your references

HW 4 is now posted

Homework 4

Due by **12:30pm, Friday, October 7, 2016**.

Make sure you follow all the homework policies.

All submissions should be done via Autolab.

Sample Problem

The Problem

This problem is just to get you thinking about graphs and get more practice with proofs.

A **forest** with c components is a graph that is the union of c disjoint trees. The figure below shows for an example with $c = 3$ and $n = 13$ with the three connected components colored blue, read and yellow).

Today’s agenda

Run-time analysis of BFS (DFS)

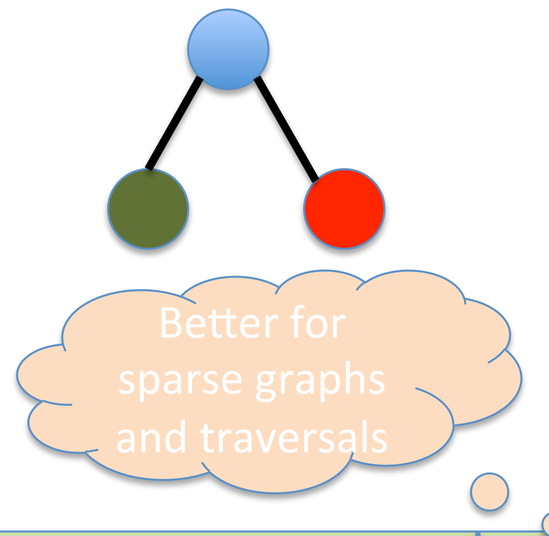
Stacks and Queues

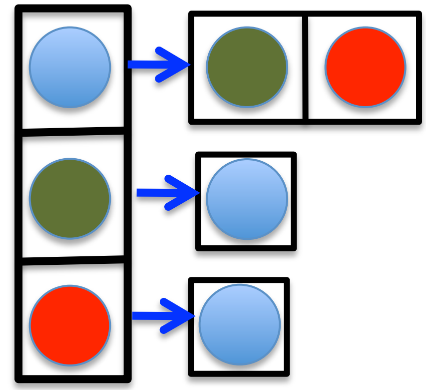
Last in First out

First in First out

Graph representations

$$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$





Adjacency matrix		Adjacency List
O(1)	(u,v) in E?	O(n) [O(n_v)]
O(n)	All neighbors of u?	O(n_u)
O(n^2)	Space?	O(m+n)

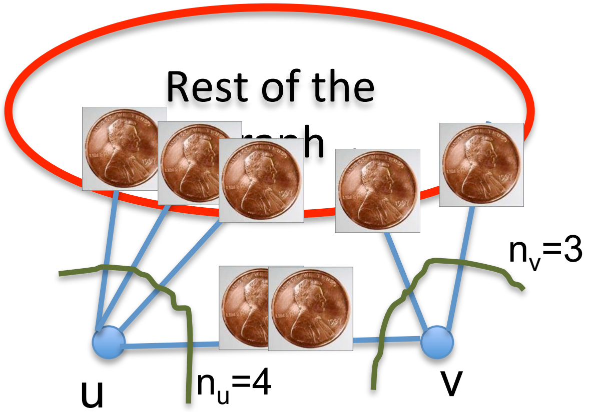
Questions?

2 # edges = sum of # neighbors

$$2m = \Sigma_{u \text{ in } V}\, n_u$$

Give 2 pennies to each edge

Total # of pennies = 2m



Each edges gives one penny to its end points

# of pennies u receives = n_u

Breadth First Search (BFS)

Build layers of vertices connected to s

$L_0 = \{s\}$

Assume $L_0,..,L_j$ have been constructed

L_{j+1} set of vertices not chosen yet but are connected to L_j

Stop when new layer is empty

Use linked lists

Use CC[v] array

Rest of Today’s agenda

Quick run time analysis for BFS

Quick run time analysis for DFS (and Queue version of BFS)

Helping you schedule your activities for the day

O(m+n) BFS Implementation

BFS(s)

Array

Input graph as Adjacency list

CC[s] = T and CC[w] = F for every w≠ s

Set i = 0

Set L_0= {s}

While L_i is not empty

Linked List

$L_{i+1} = Ø$

For every u in L_i

For every edge (u,w)

If CC[w] = F then

CC[w] = T

Add w to L_{i+1}

i++

Version in KT also computes a BFS tree

All the layers as one

BFS(s)

$CC[s]$ = T and $CC[w]$ = F for every $w \neq s$

Set $i = 0$

Set $L_0 = \{s\}$

While L_i is not empty

  $L_{i+1} = \emptyset$

  For every u in L_i

    For every edge (u,w)

    If $CC[w]$ = F then

      $CC[w]$ = T

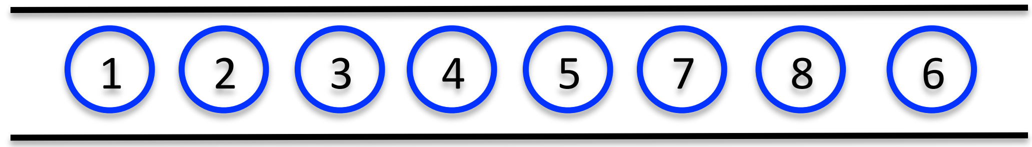
      Add w to L_{i+1}

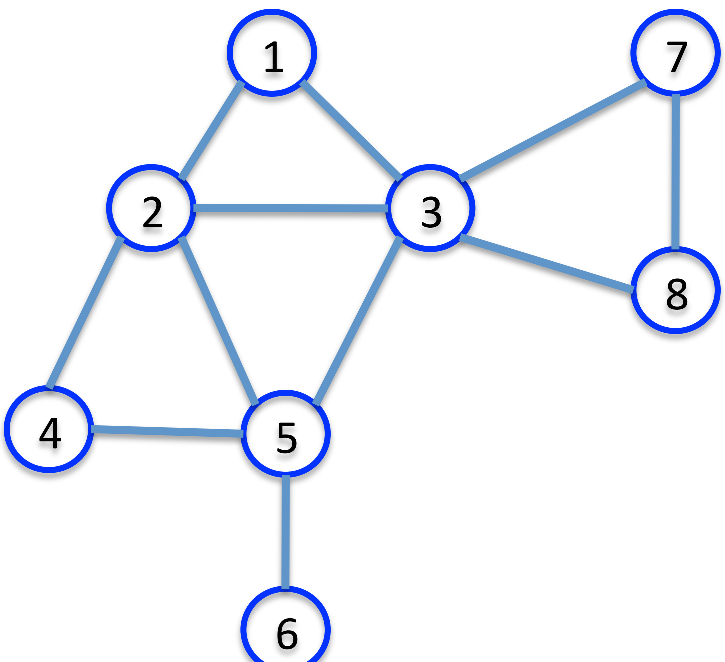
  i++

All layers are considered in first-in-first-out order

Can combine all layers into one queue: all the children of a node are added to the end of the queue

An illustration





Queue O(m+n) implementation

BFS(s)

CC[s] = T and CC[w] = F for every w≠ s

Intitialize Q= {s}

While Q is not empty

Delete the front element u in Q

For every edge (u,w)

If CC[w] = F then

CC[w] = T

Add w to the back of Q

O(n)

O(1)

$\Sigma_u O(n_u) = O(\Sigma_u n_u) = O(m)$

O(1)

$O(n_u)$

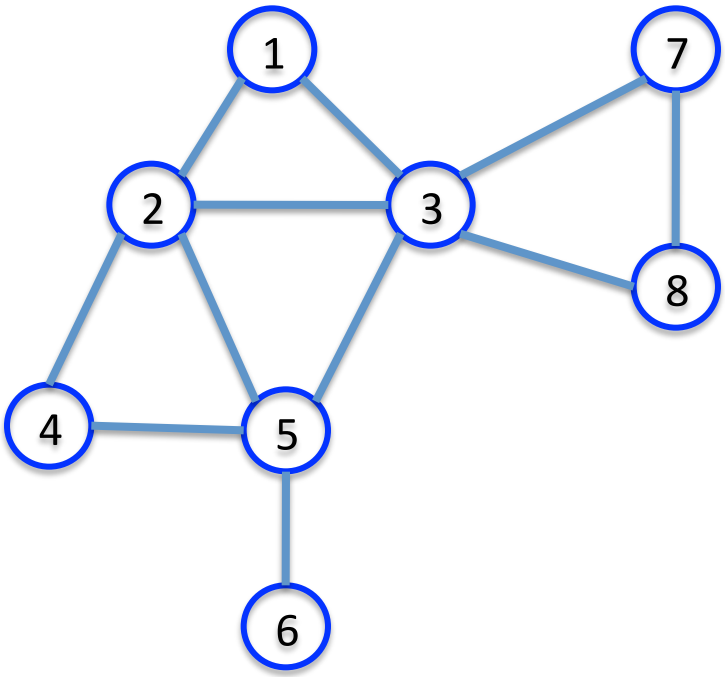
O(1)

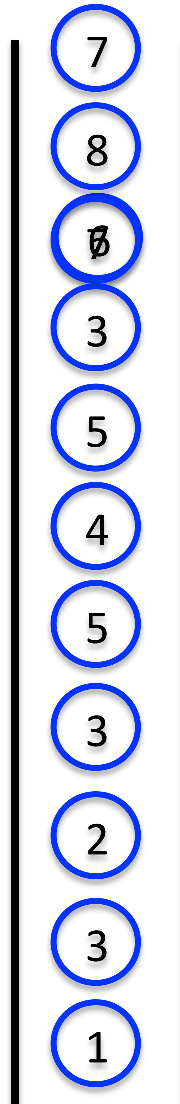
Questions?

Implementing DFS in $O(m+n)$ time

Same as BFS except stack instead of a queue

A DFS run using an explicit stack





DFS stack implementation

DFS(s)

CC[s] = T and CC[w] = F for every w≠ s

Intitialize Ŝ = {s}

While Ŝ is not empty

Pop the top element u in Ŝ

For every edge (u,w)

If CC[w] = F then

CC[w] = T

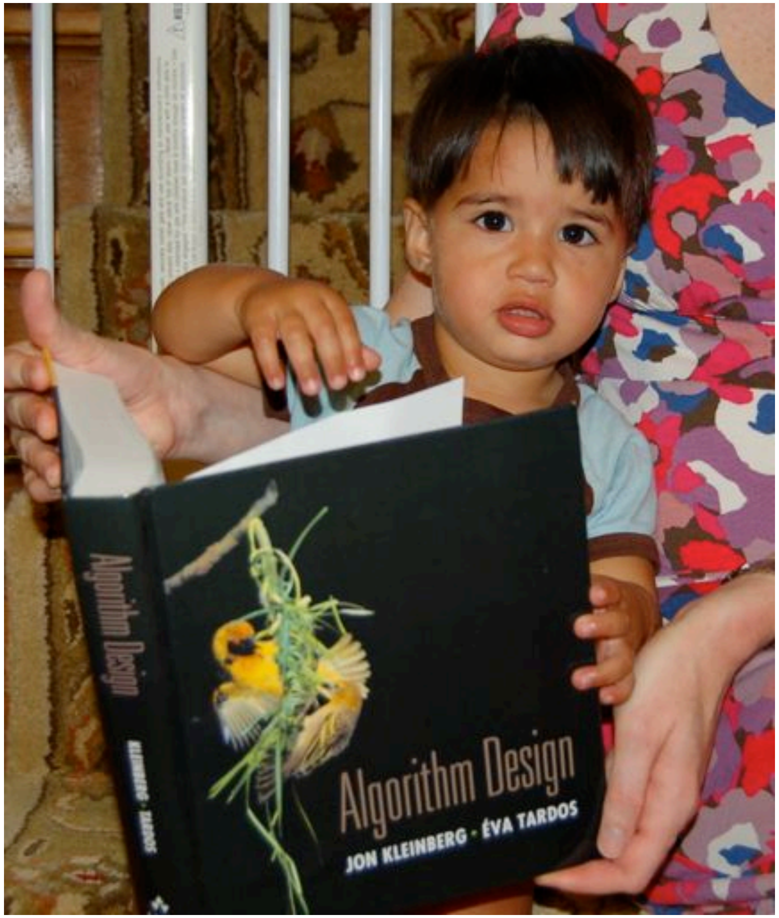
Push w to the top of Ŝ

Same O(m+n) run time analysis as for BFS

Questions?

Reading Assignment

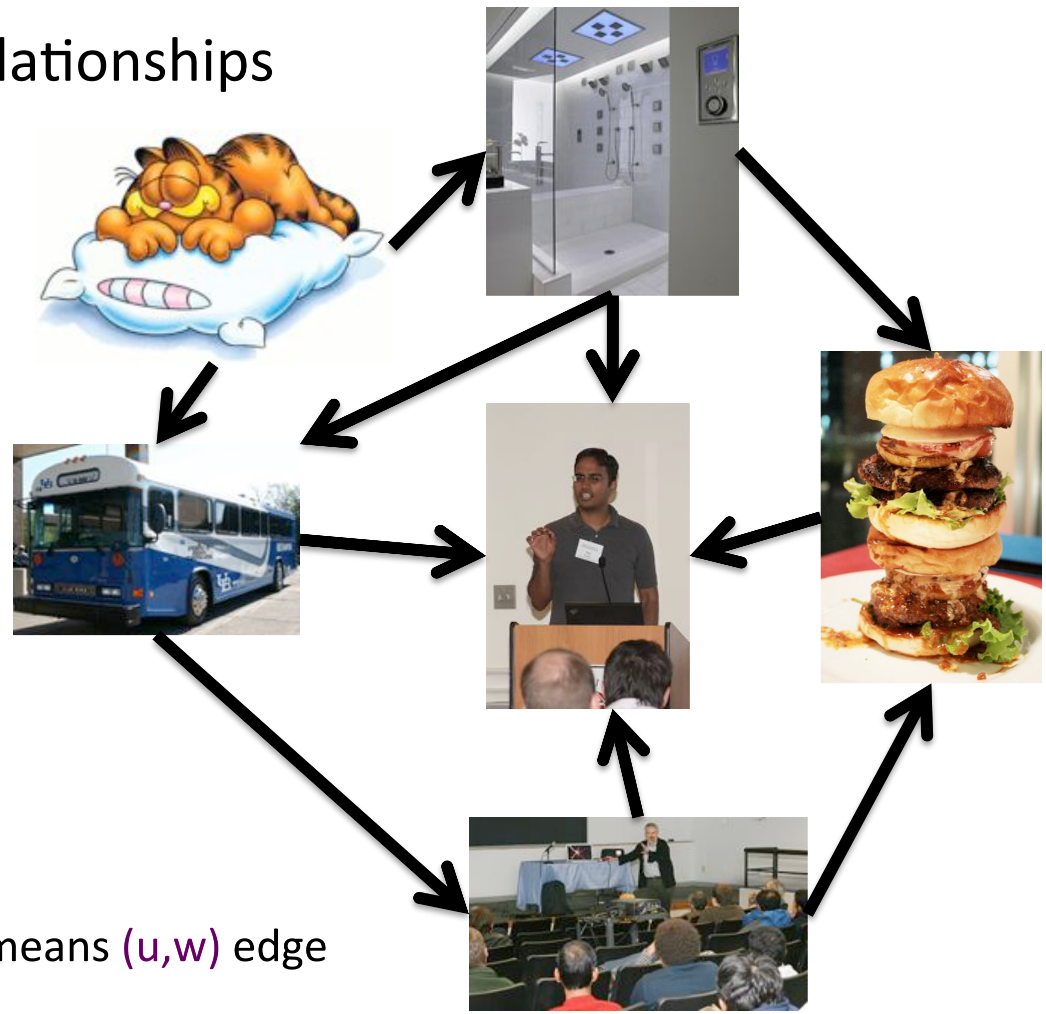
Sec 3.3, 3.4 and 3.5 of [KT]



Directed graphs

Model asymmetric relationships

Precedence relationships

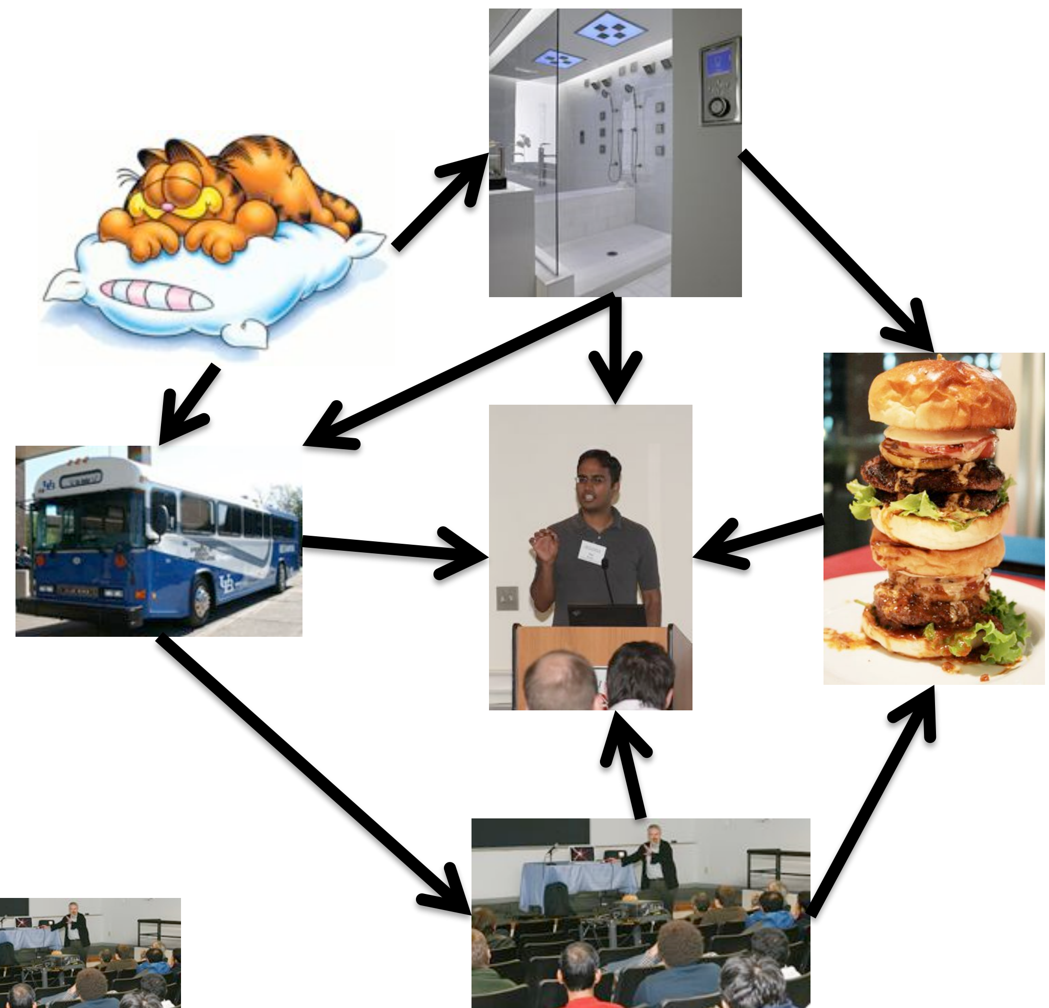


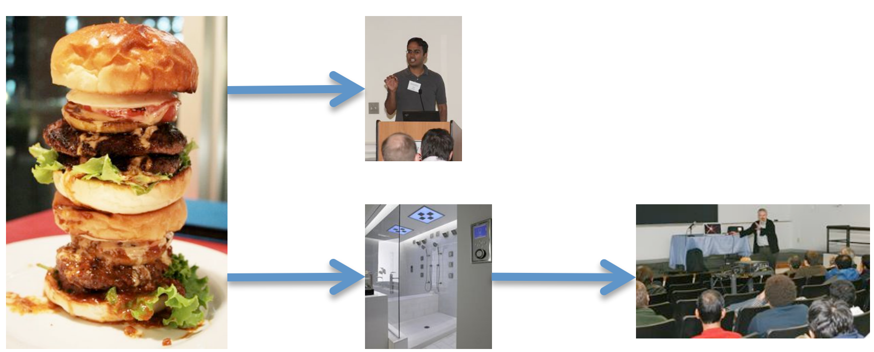
u needs to be done before w means (u,w) edge

Directed graphs

Adjacency matrix is not symmetric

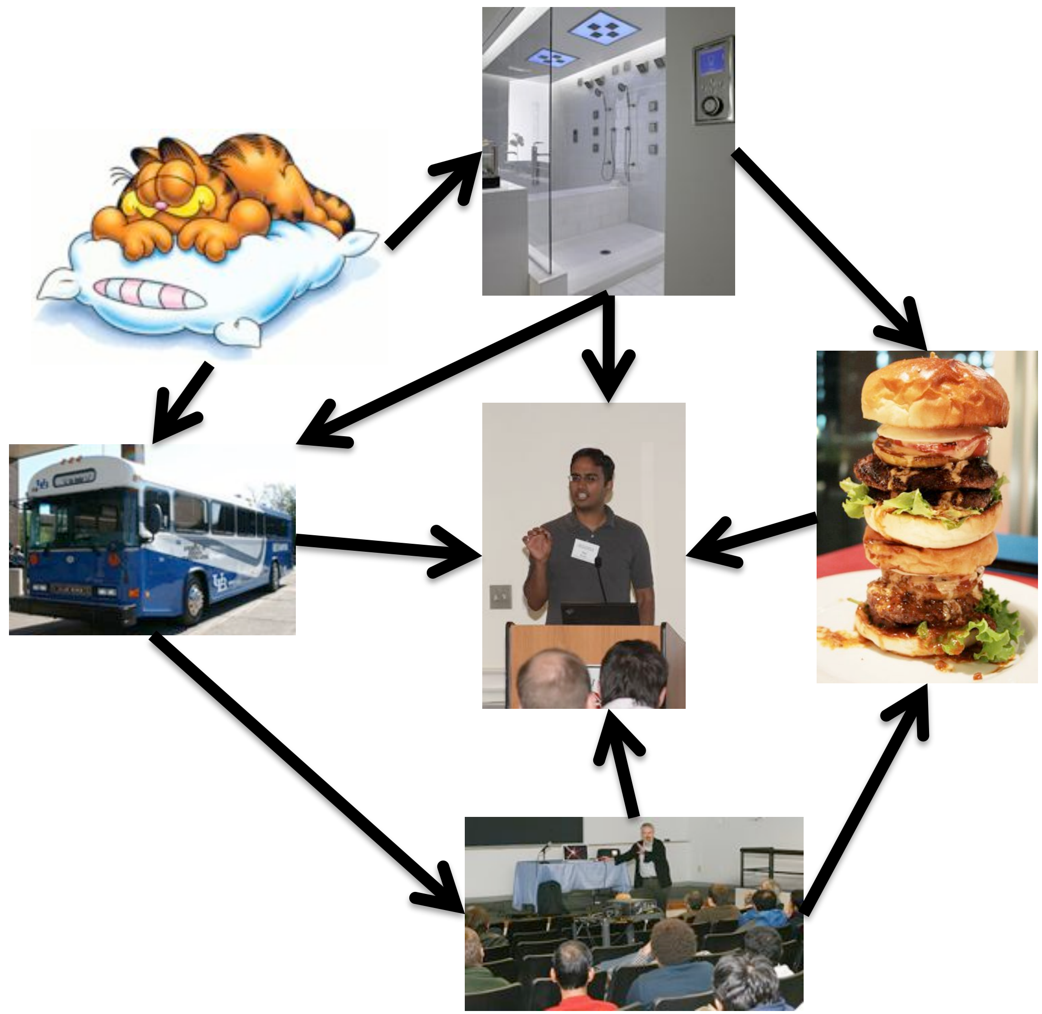
Each vertex has two lists in Adj. list rep.





Directed Acyclic Graph (DAG)

No directed cycles



Precedence relationships are consistent

Topological Sorting of a DAG

Order the vertices so that all edges go "forward"