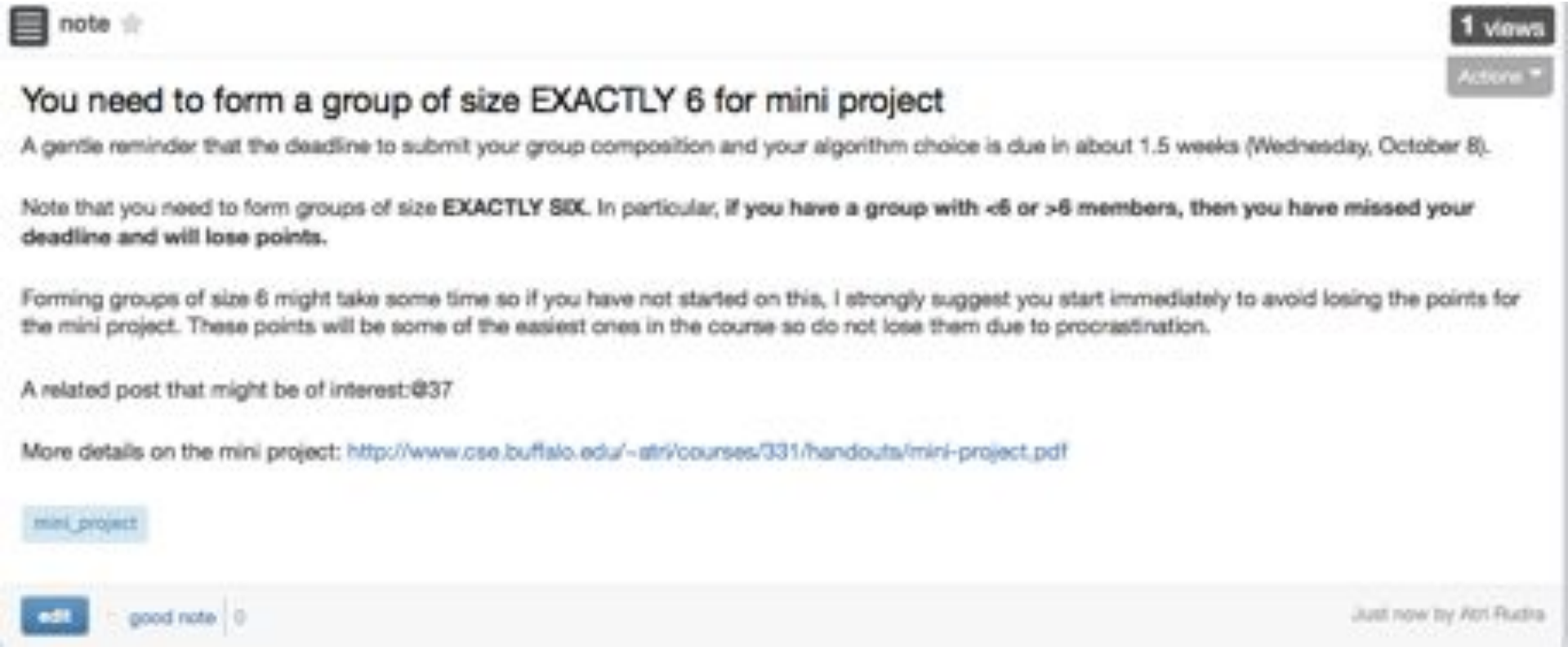
Lecture 15

CSE 331

Sep 29, 2014

Group/Algo registration deadline

BOTH DUE BY NEXT WEDNESDAY!


note

1 Views

Actions

You need to form a group of size EXACTLY 6 for mini project

A gentle reminder that the deadline to submit your group composition and your algorithm choice is due in about 1.5 weeks (Wednesday, October 8).

Note that you need to form groups of size **EXACTLY SIX**. In particular, **if you have a group with <6 or >6 members, then you have missed your deadline and will lose points.**

Forming groups of size 6 might take some time so if you have not started on this, I strongly suggest you start immediately to avoid losing the points for the mini project. These points will be some of the easiest ones in the course so do not lose them due to procrastination.

A related post that might be of interest:@37

More details on the mini project: http://www.cse.buffalo.edu/~atri/courses/331/handouts/mini-project.pdf

mini_project

edit good note 0

Just now by Atri Rudra

HW related stuff

HW 4 posted on piazza last Friday

Graded HW 2 available for pickup starting earlier today

Will hand out solutions to HW 3 at the END of the lecture

Today's agenda

Run-time analysis of BFS (DFS)

Stacks and Queues

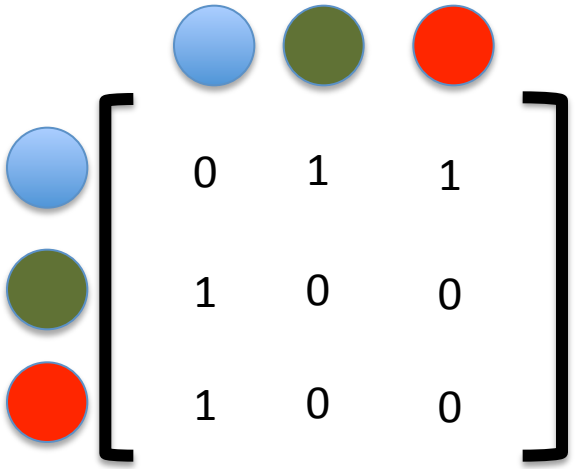
Last in First out

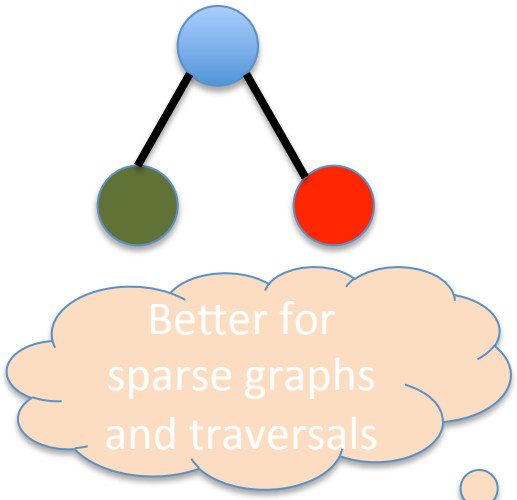
First in First out

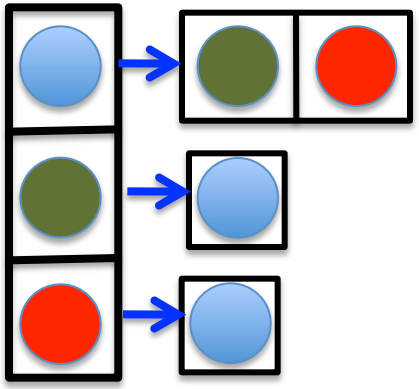
But first…

How do we represent graphs?

Graph representations







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)

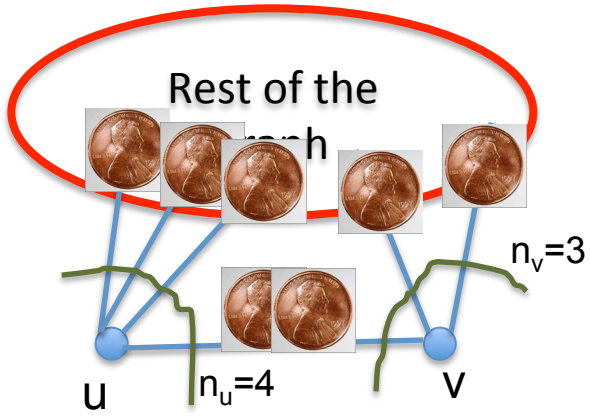
Questions?

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

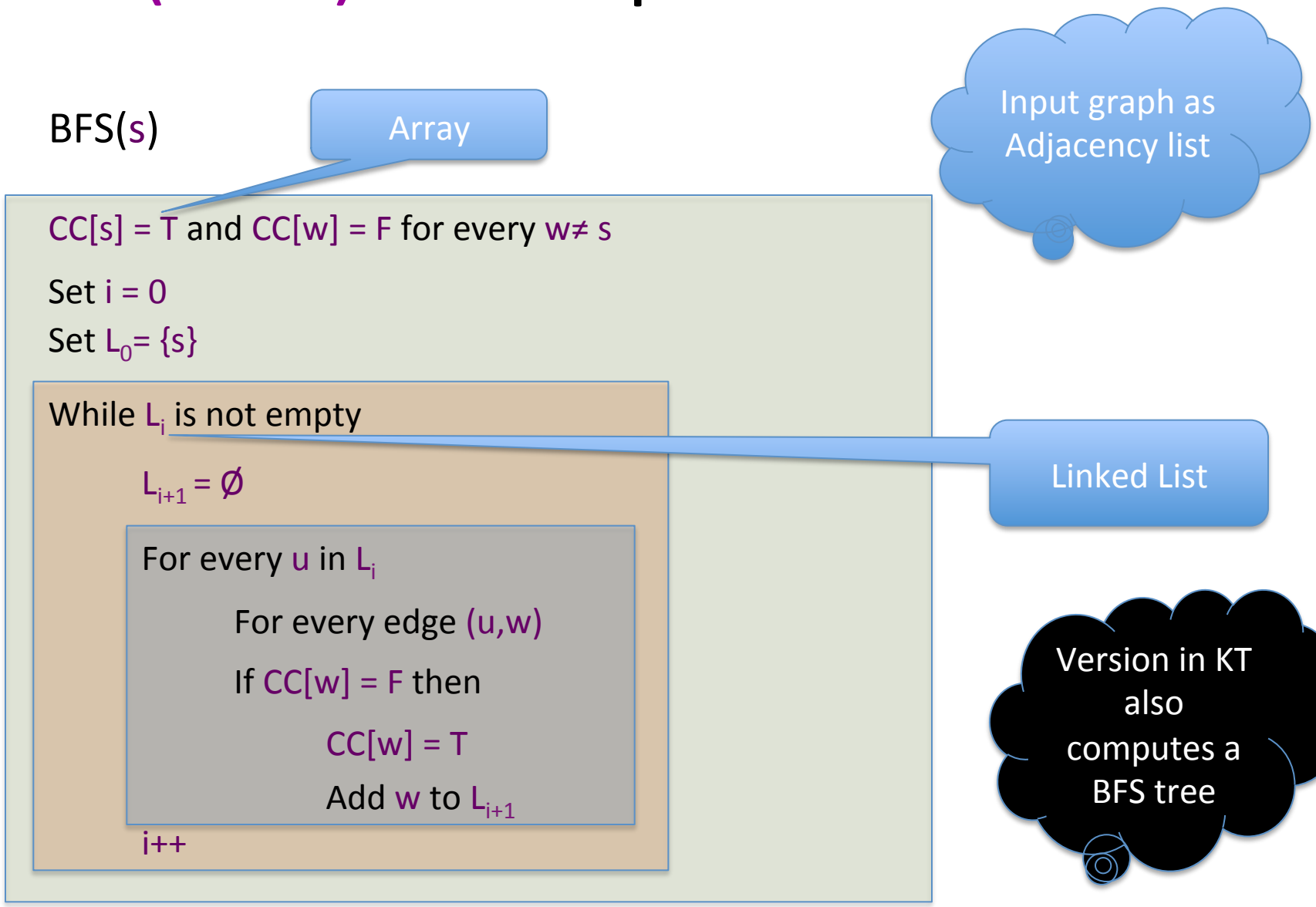
L_{j+1} set of vertices not chosen yet but are connected to L_j

Stop when new layer is empty

Use linked lists

Use CC[v] array

O(m+n) BFS Implementation



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

All the layers as one

BFS(s)

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

Set i = 0

Set L_0= {s}

While L_i is not empty

L_{i+1} = Ø

For every u in L_i

For every edge (u,w)

If CC[w] = F then

CC[w] = T

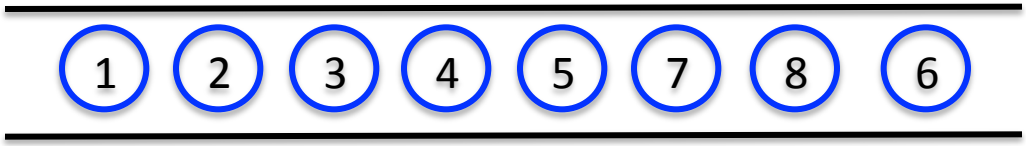
Add w to L_{i+1}

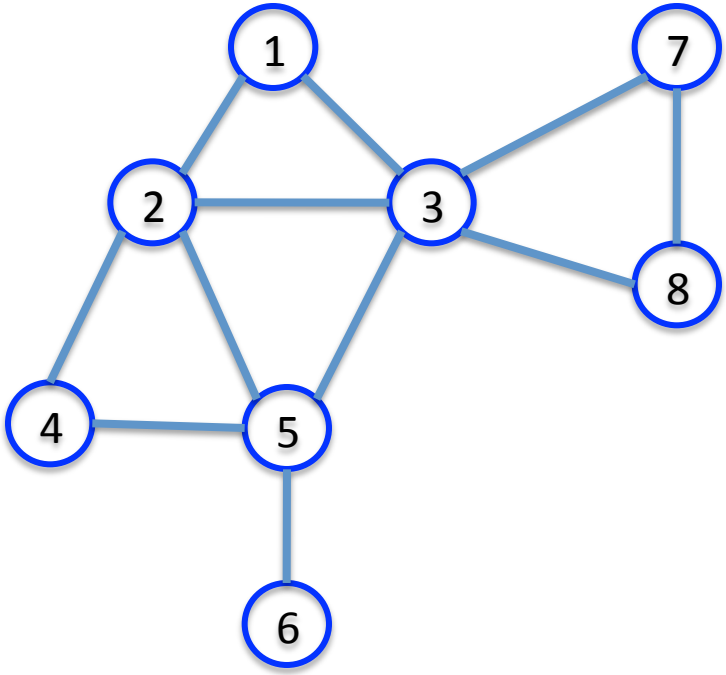
i++

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

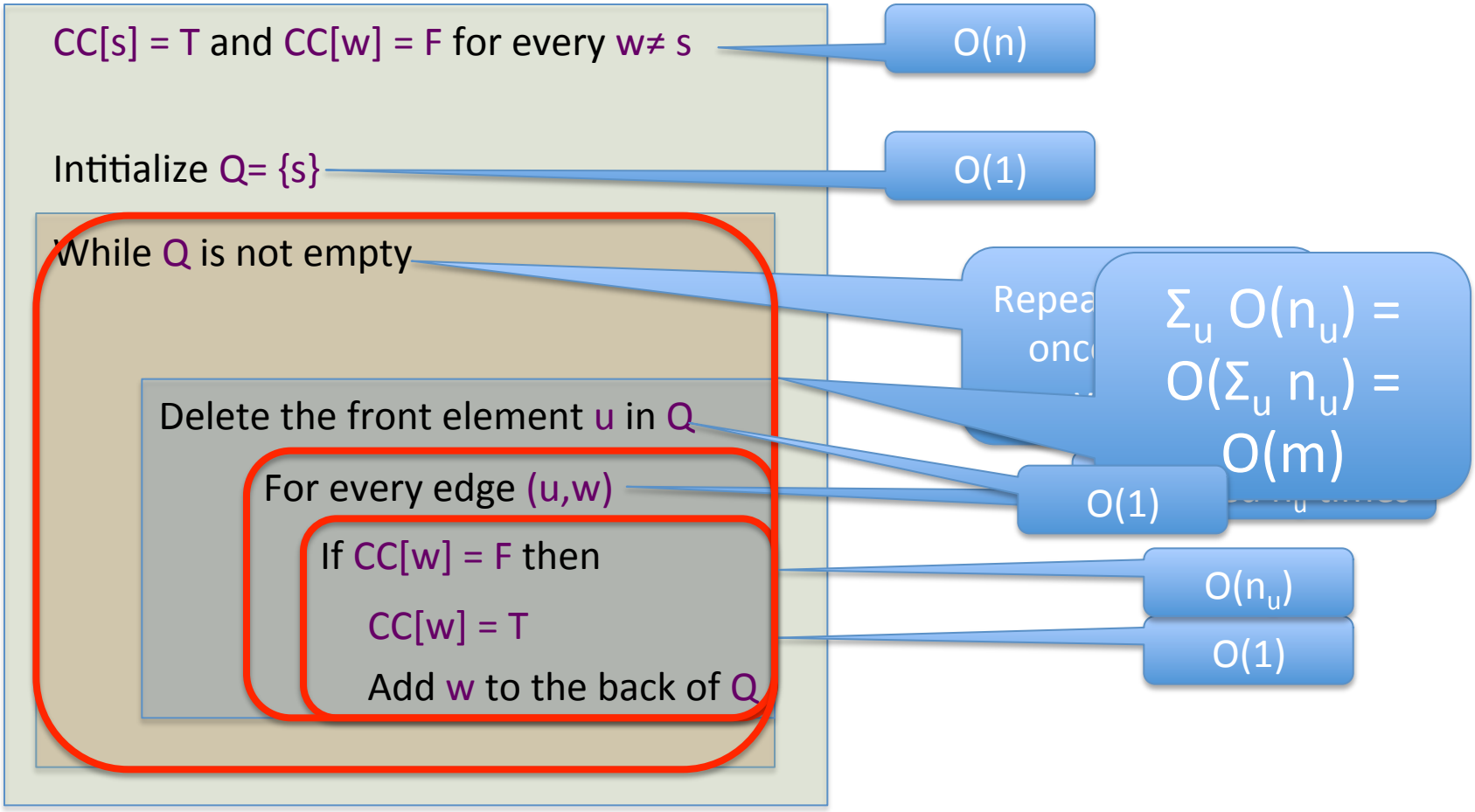
An illustration





Queue O(m+n) implementation

BFS(s)

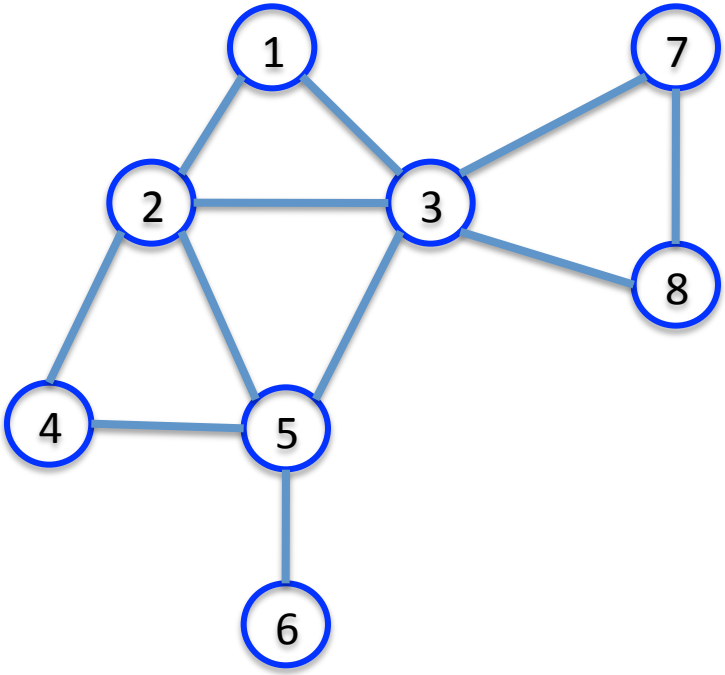


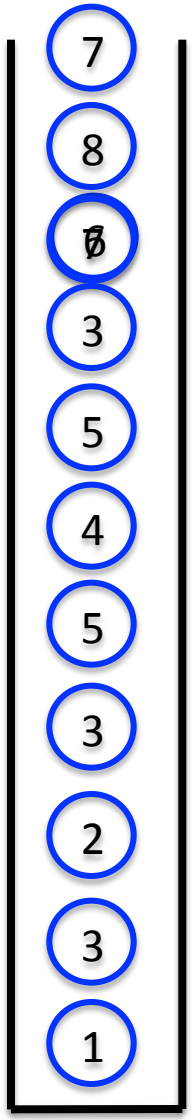
Questions?

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 $\hat{S}$ = {s}

While $\hat{S}$ is not empty

Pop the top element u in $\hat{S}$

For every edge (u,w)

If CC[w] = F then

CC[w] = T

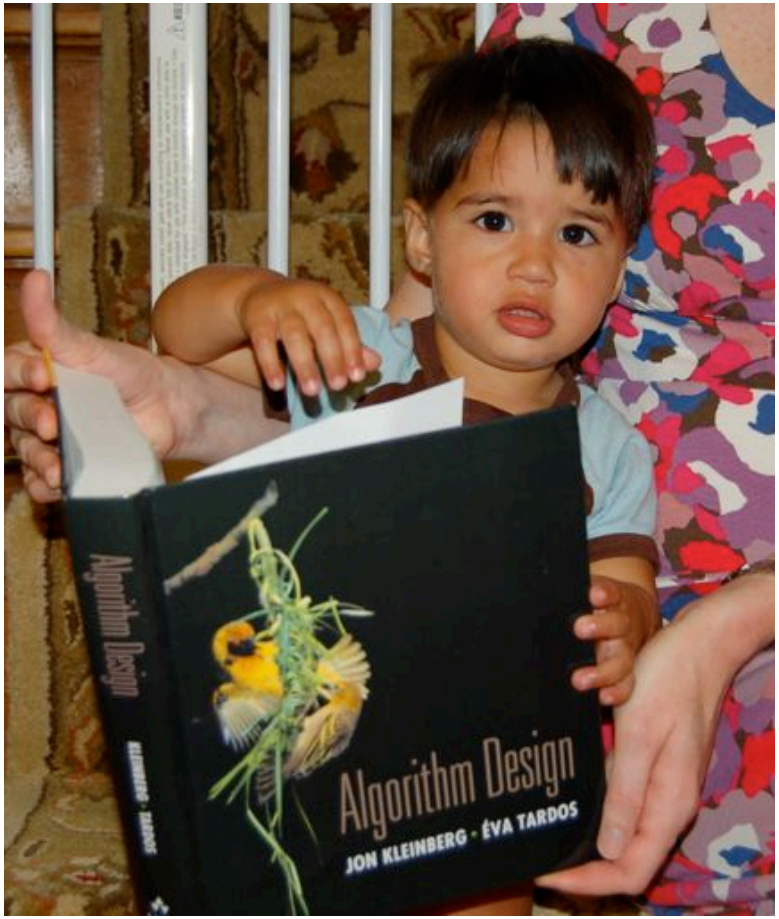
Push w to the top of $\hat{S}$

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

Questions?

Reading Assignment

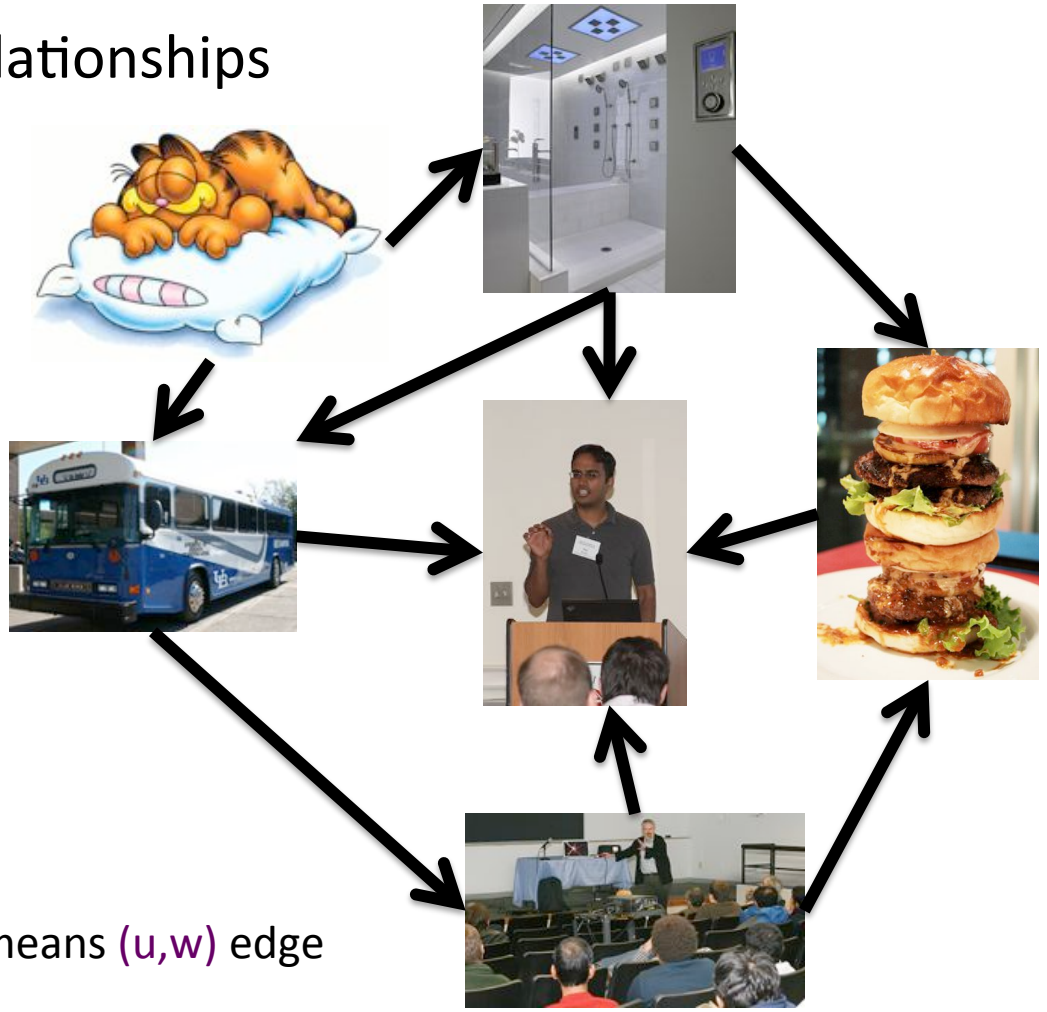
Sec 3.3, 3.4 and 3.5 of [KT]



Directed graphs

Model asymmetric relationships

Precedence relationships

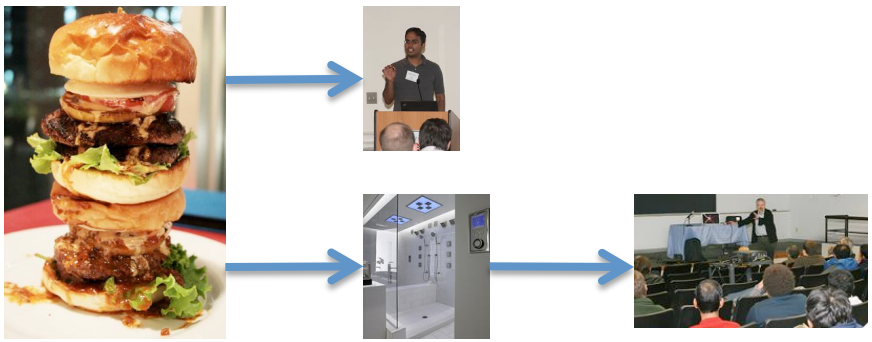


u needs to be done before w means (u,w) edge

Directed graphs

Adjacency matrix is not symmetric

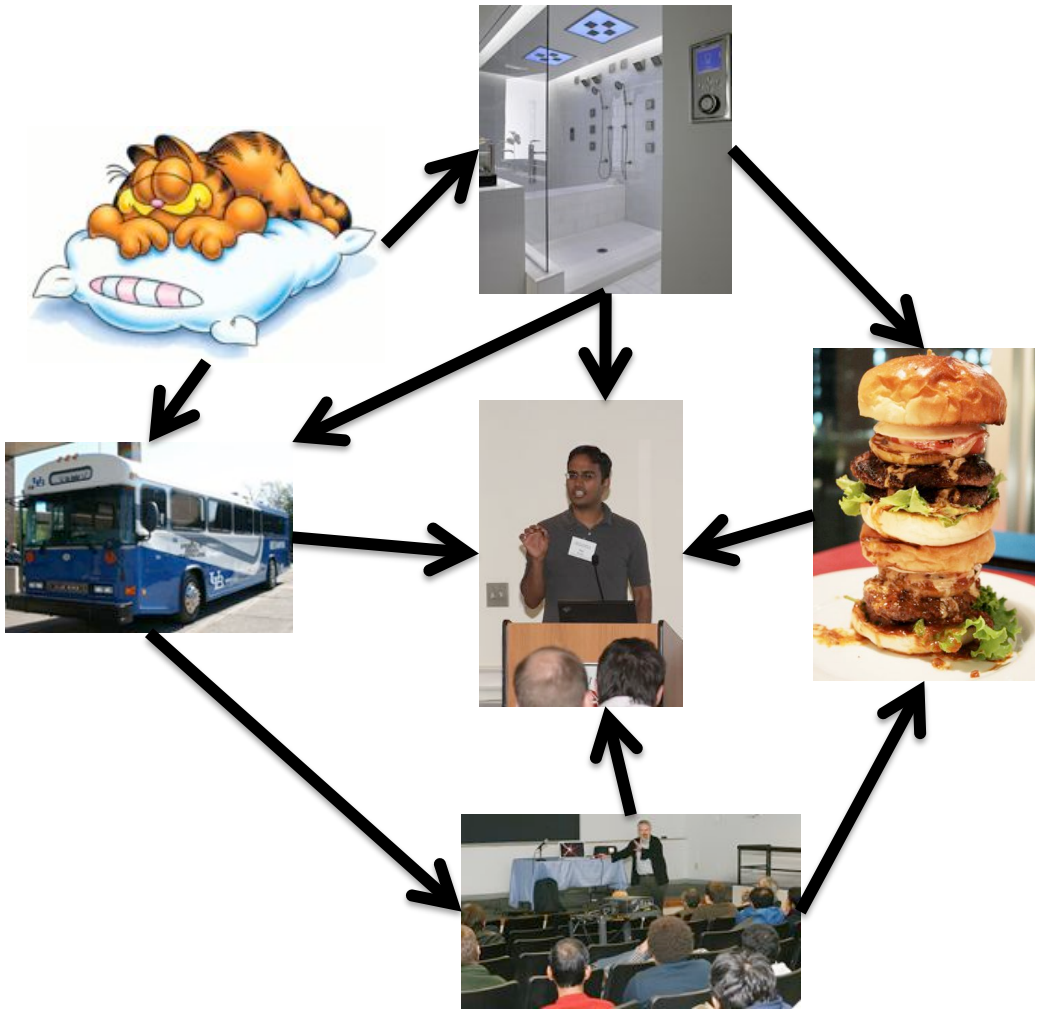
Each vertex has two lists in Adj. list rep.



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"