Algorithms and Data Structures for Data Science
Graph Implementations 2

CS 277
Brad Solomon

April 12, 2023

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN
Department of Computer Science

This week only: Lab room and OH Changes

Friday April 14th: AE3's Celebration of Teaching in 1306 Everitt

Our lab will be in 2101 Everitt instead!

Office Hour Changes: My OH will be Friday between 3:15 and 4:15

There will not be OH on Thursday April 13th!

Lab Feedback

Still read them and appreciate feedback

lab_huffman needs work in the future in both presentation and content

lab_trees and **lab_avl** were both highly rated

Learning Objectives

Review edge list and adjacency matrix graph implementations

Introduce adjacency list implementation

Discuss the strengths and weaknesses of each implementation

Graphs

Given a roster of students for each class, build a graph which tracks whether there are at least three students in common between two classes

What is a vertex?

What is an edge?

Are the edges directed or undirected?

Are the edges weighted or unweighted?

Graph ADT

Find

getVertices() — return the list of vertices in a graph

getEdges(v) — return the list of edges that touch the vertex v

areAdjacent(u, v) — returns a bool based on if an edge from u to v exists

Insert

insertVertex(v) — adds a vertex to the graph

insertEdge(u, v) — adds an edge to the graph

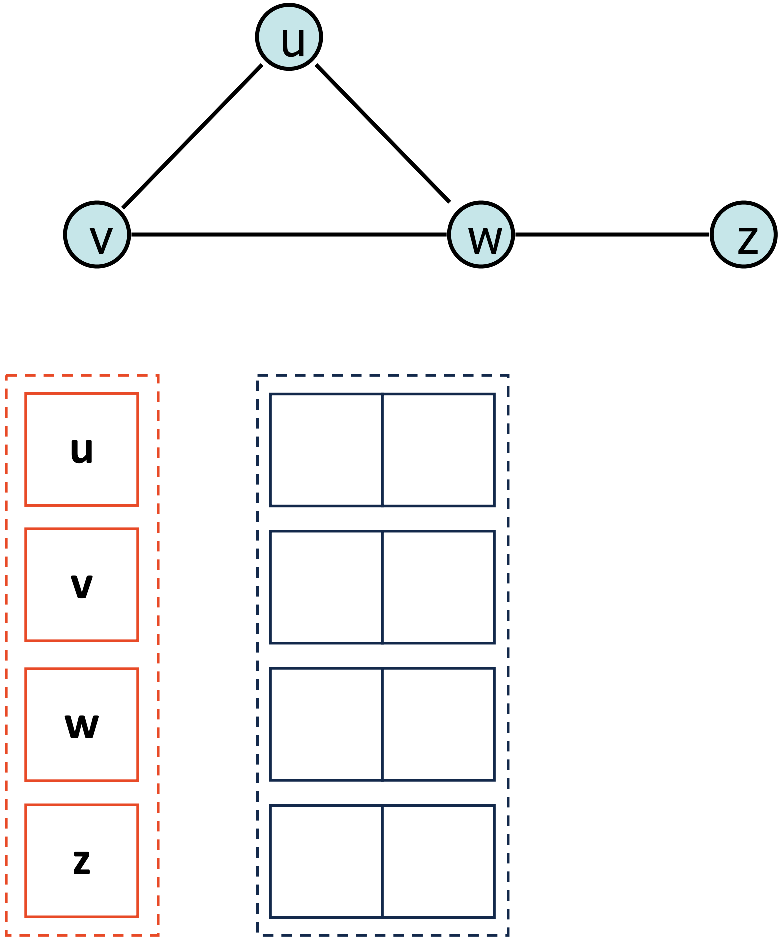
Remove

removeVertex(v) — removes a vertex from the graph

removeEdge(u, v) — removes an edge from the graph

Graph Implementation: Edge List $|V| = n, |E| = m$

The equivalent of an 'unordered' data structure



Vertex Storage:

Not stored at all (recovered from edges)

or

An unordered list of vertices

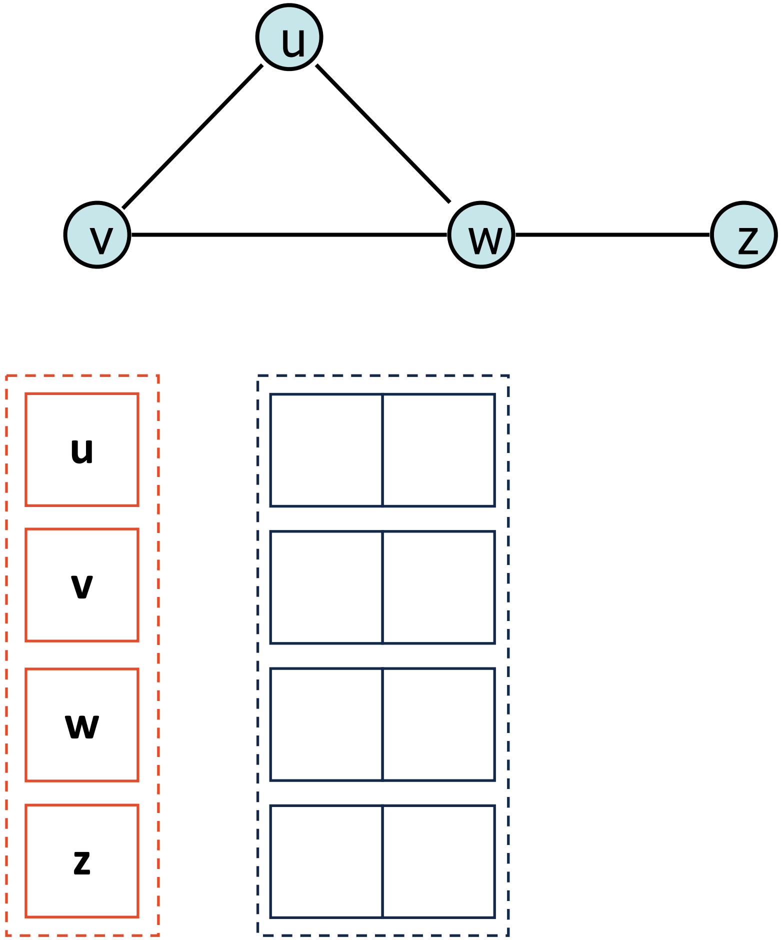
Edge Storage:

An unordered list of edges (as tuples)

[or equivalent]

Graph Implementation: Edge List

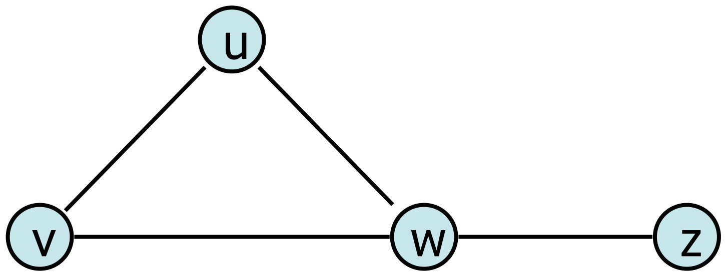
How would our data structure change if…



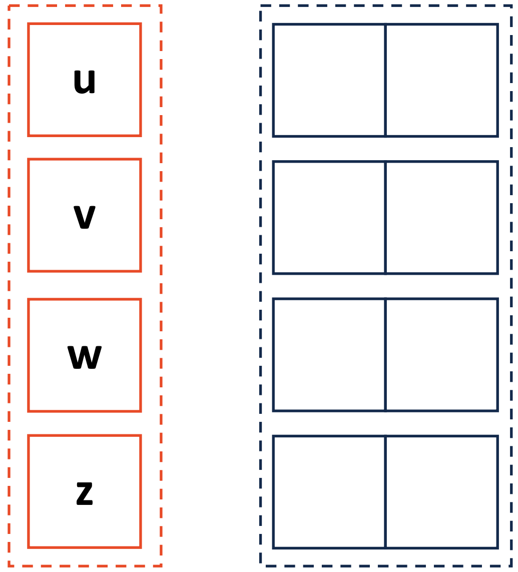
Edges are weighted:

Graph Implementation: Edge List

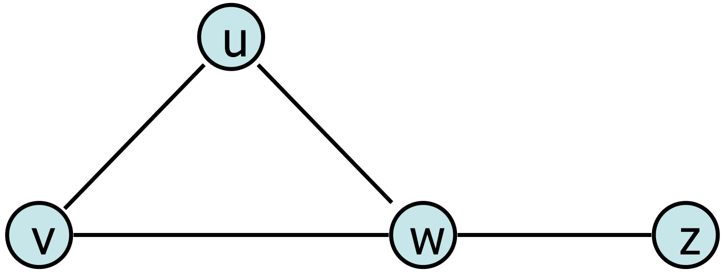
How would our data structure change if…



Edges are directed:

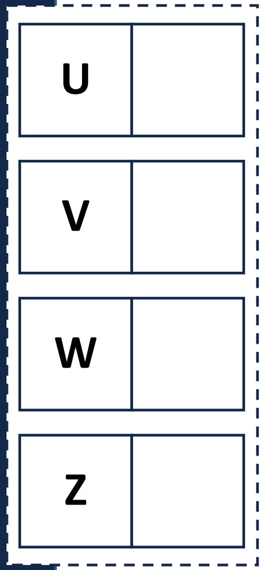


Graph Implementation: Adjacency Matrix



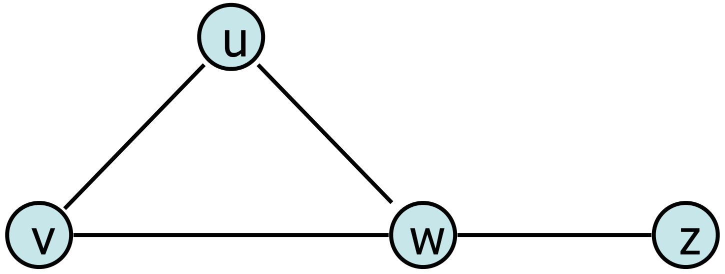
Vertex Storage:

Edge Storage:



	u	v	w	z
u				
v				
w				
z				

Graph Implementation: Adjacency Matrix

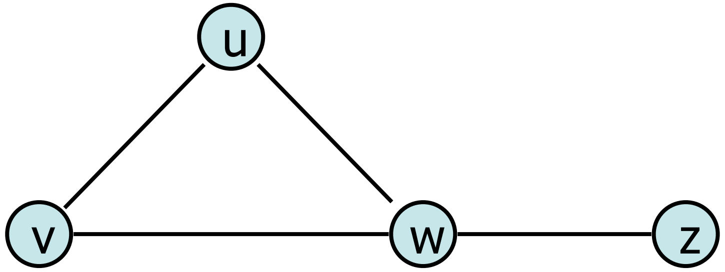


getVertices():

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Graph Implementation: Adjacency Matrix

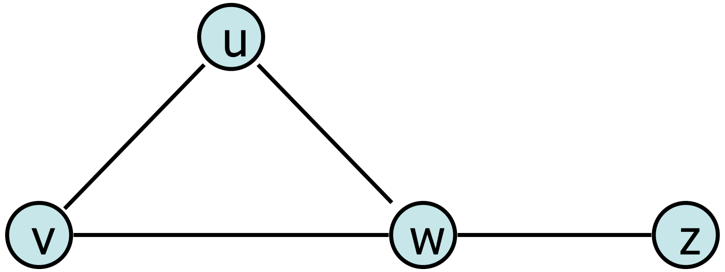


getEdges(v):

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Graph Implementation: Adjacency Matrix

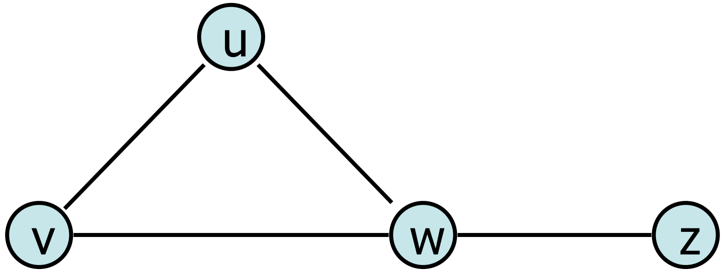


areAdjacent(u, v):

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Graph Implementation: Adjacency Matrix

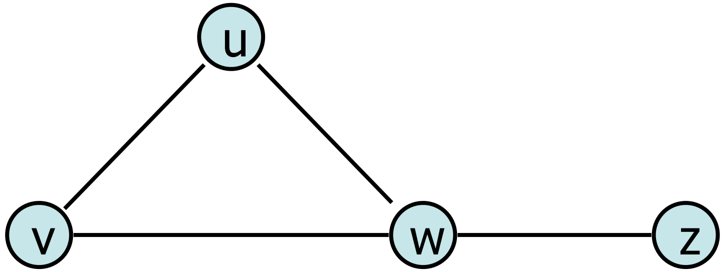


insertVertex(v):

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Graph Implementation: Adjacency Matrix

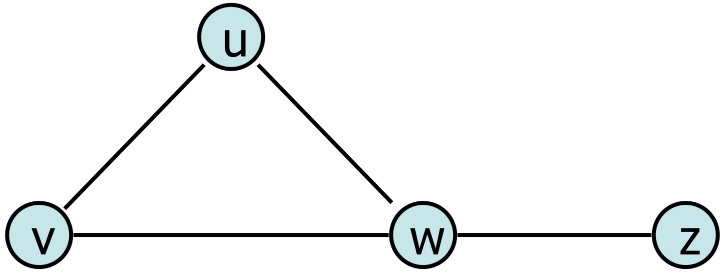


insertEdge(u, v):

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Graph Implementation: Adjacency Matrix



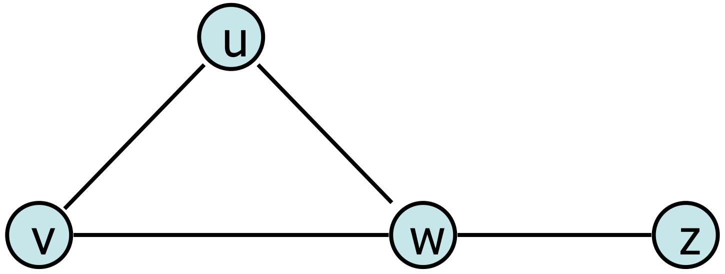
U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

removeVertex(v):

removeEdge(u, v):

Graph Implementation: Adjacency Matrix



U	0
V	1
W	2
Z	3

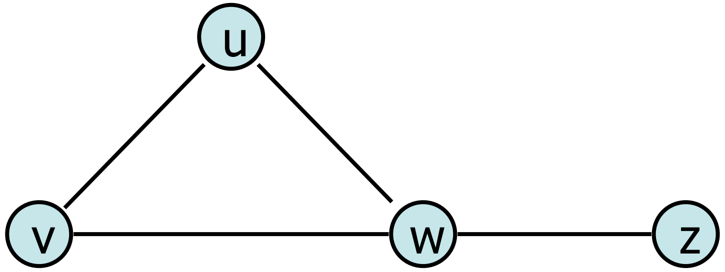
	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Pros:

Cons:

Graph Implementation: Adjacency Matrix

How would our data structure change if…



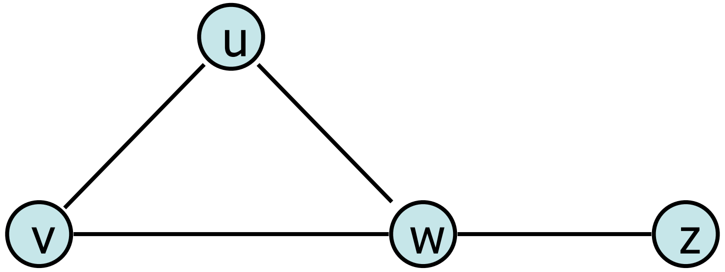
Edges are directed:

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Graph Implementation: Adjacency Matrix

How would our data structure change if…

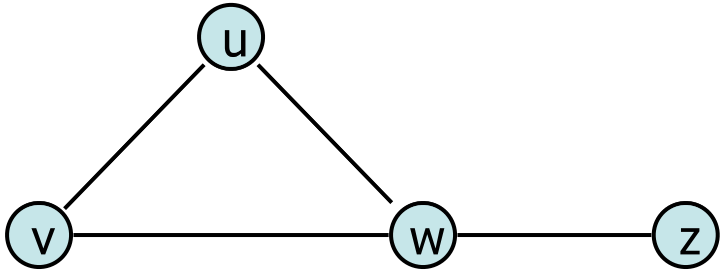


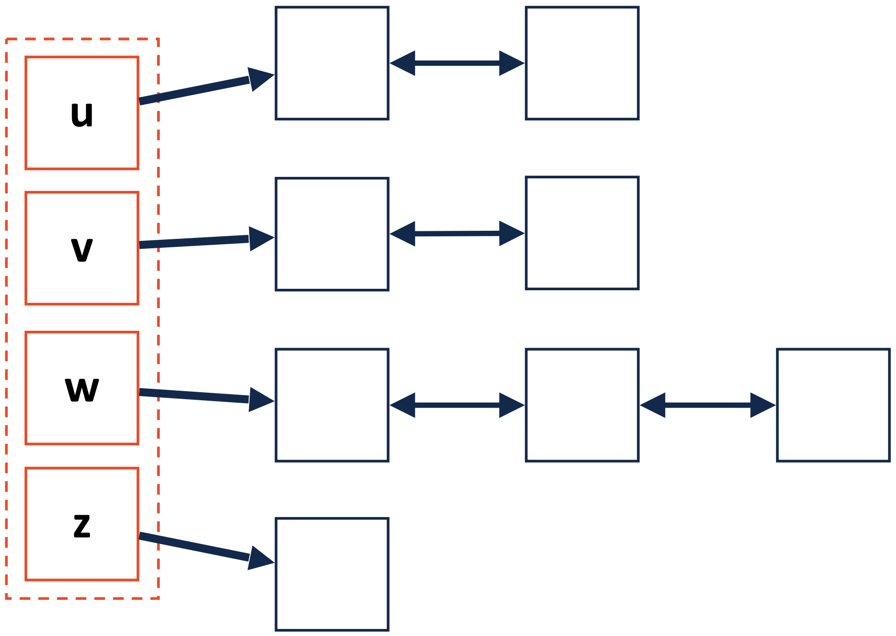
Edges are weighted:

U	0
V	1
W	2
Z	3

	u	v	w	z
u	0	1	1	0
v	1	0	1	0
w	1	1	0	1
z	0	0	1	0

Adjacency List

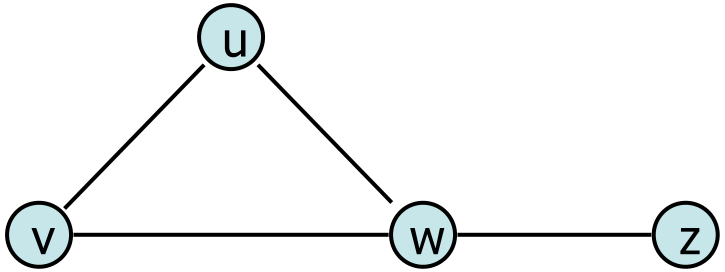


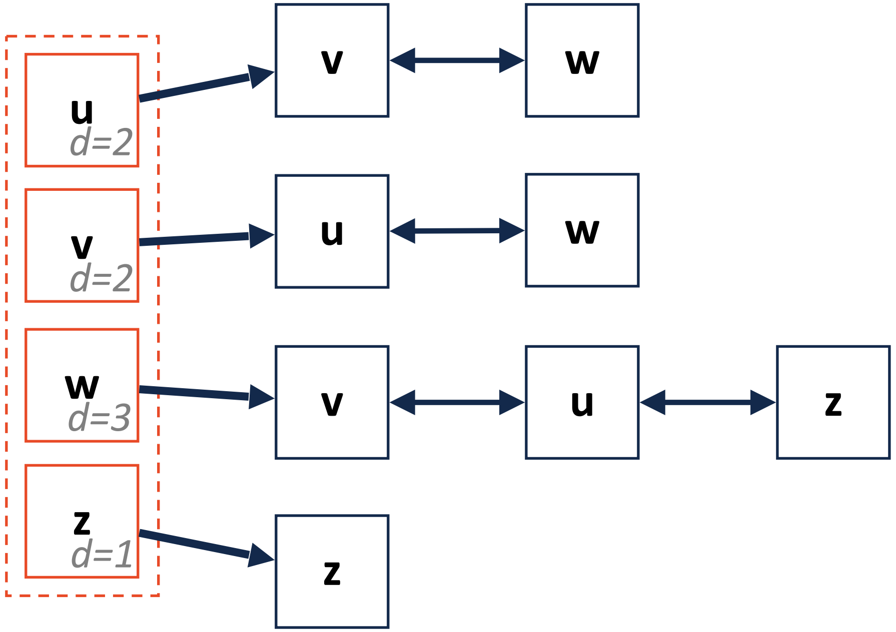


Vertex Storage:

Edge Storage:

Adjacency List

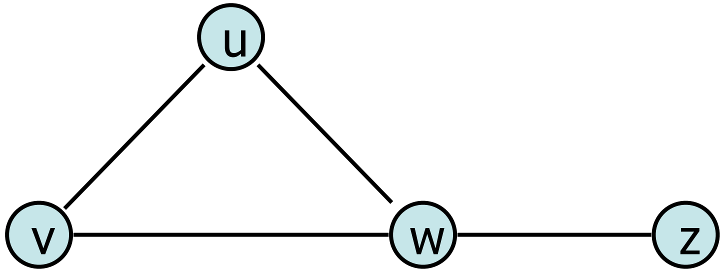


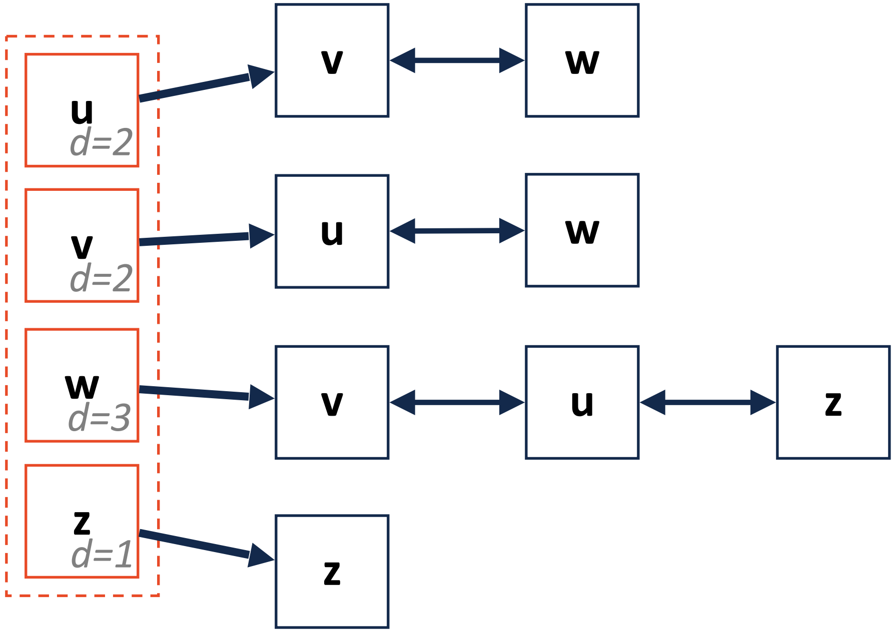


Vertex Storage:

Edge Storage:

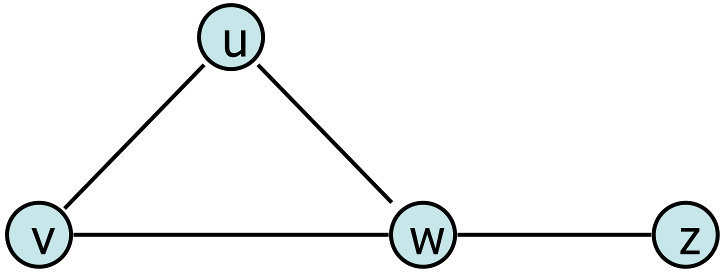
Adjacency List



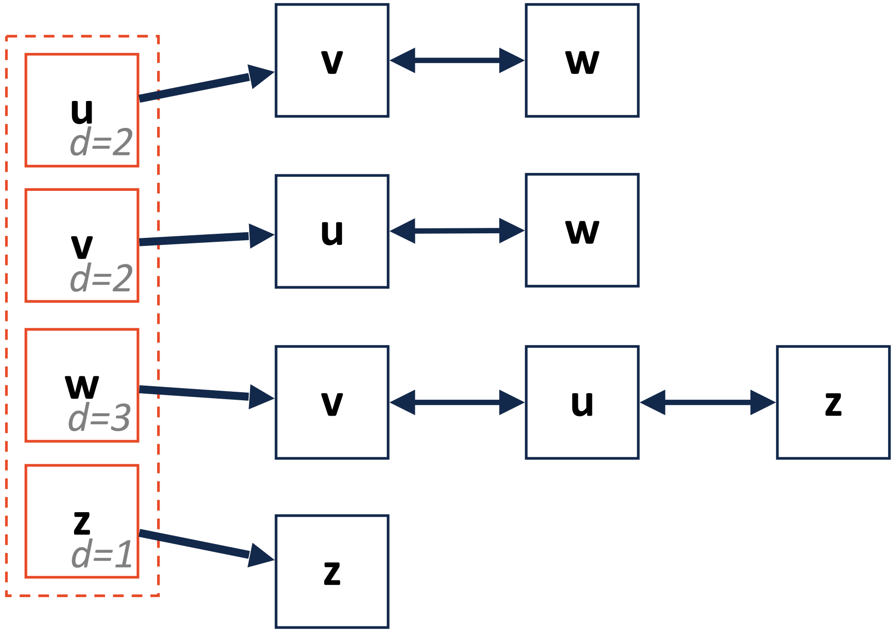


getVertices():

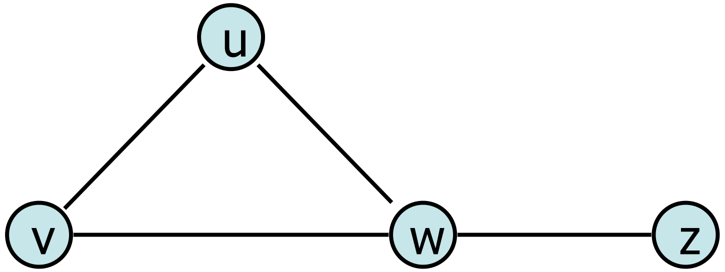
Adjacency List

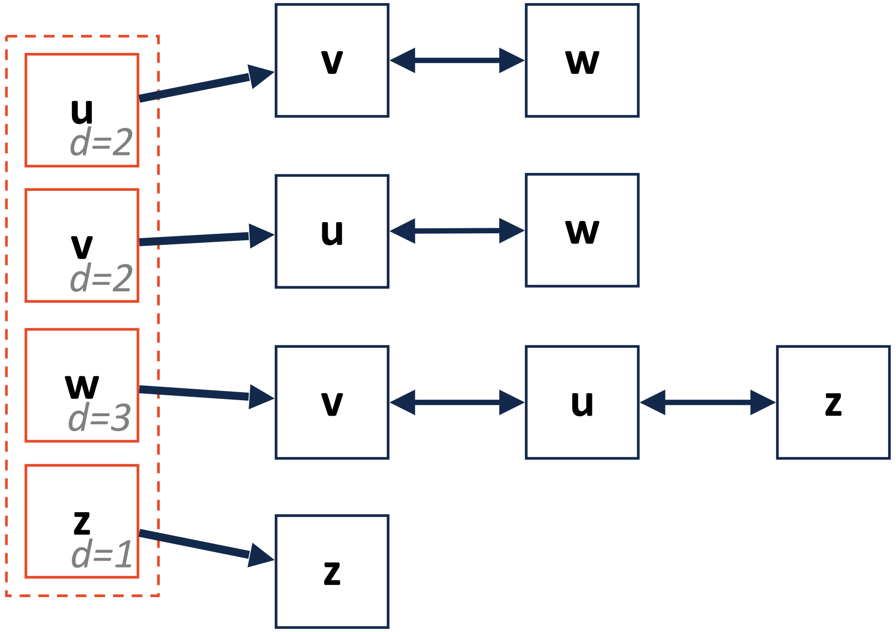


getEdges(v):



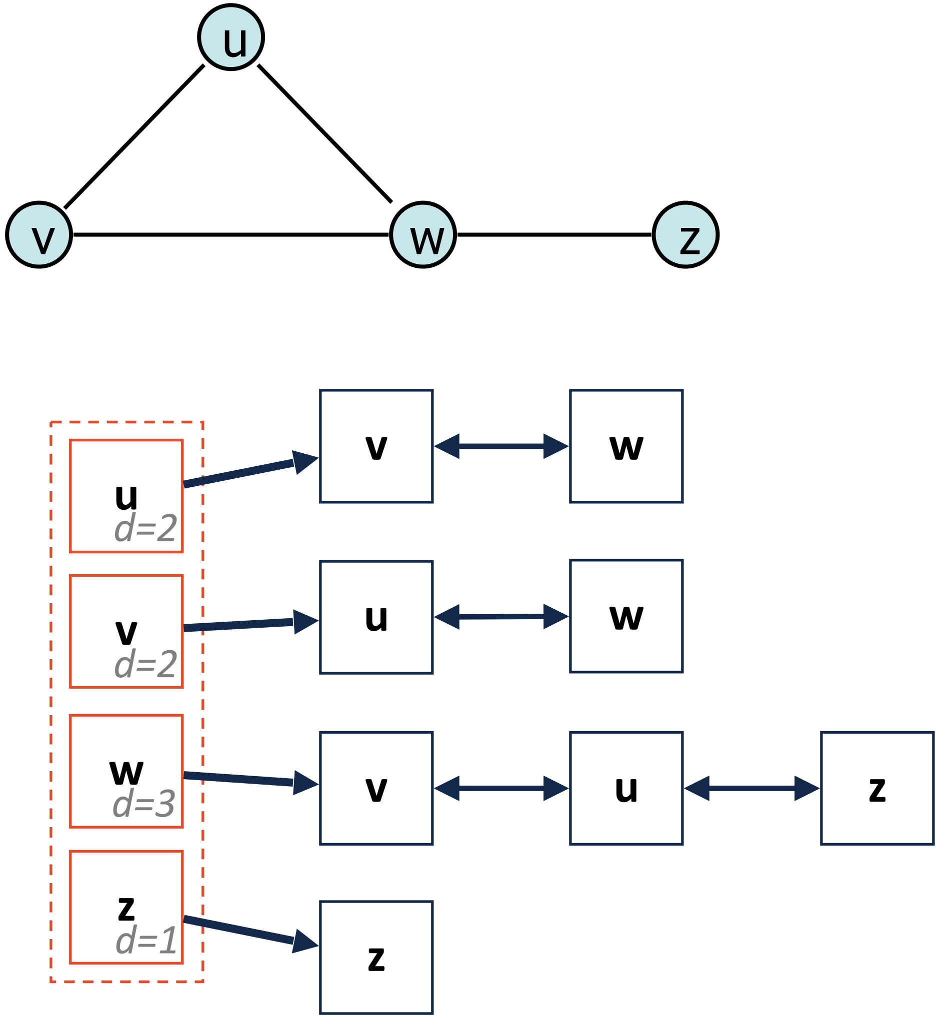
Adjacency List





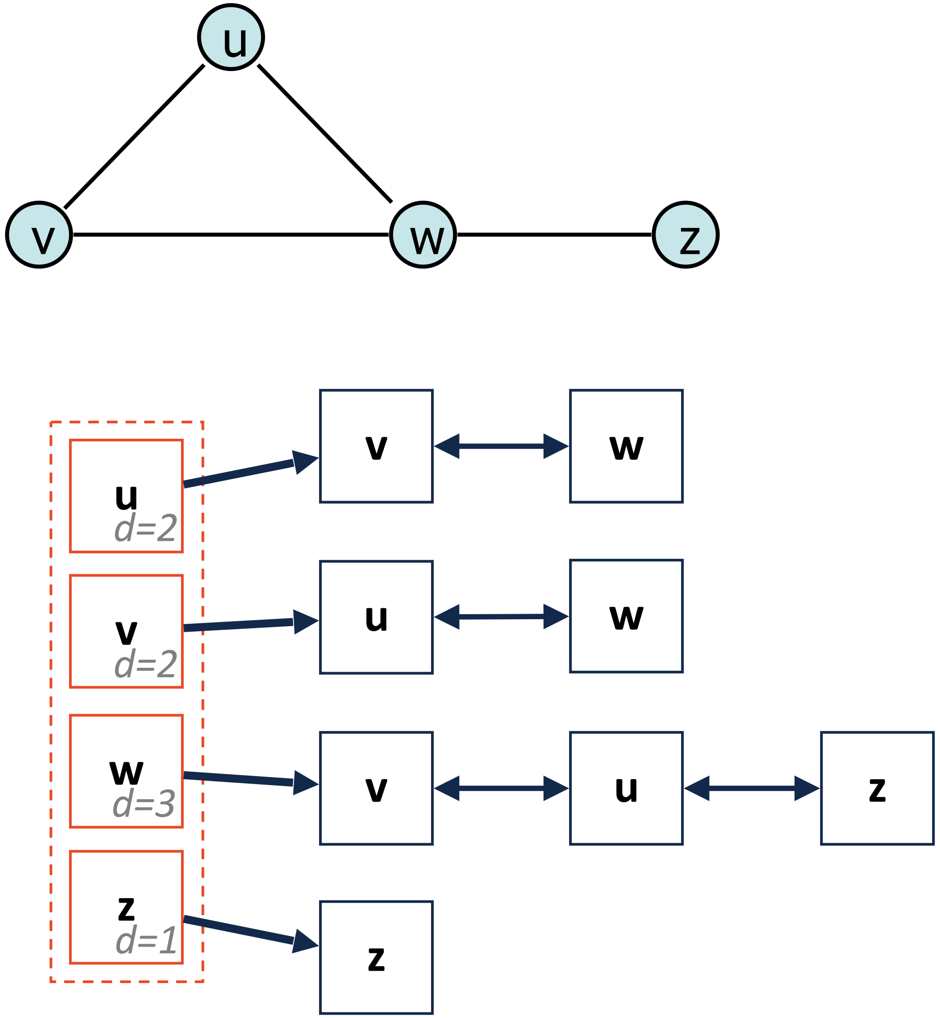
areAdjacent(u, v):

Adjacency List



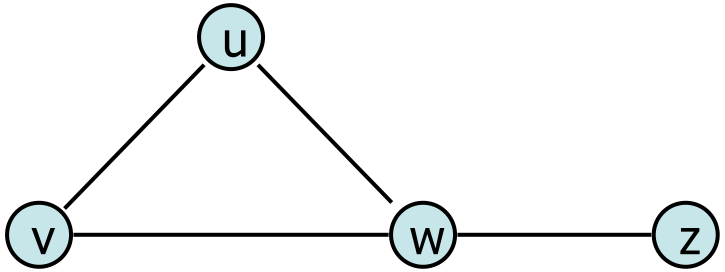
insertVertex(v):

Adjacency List

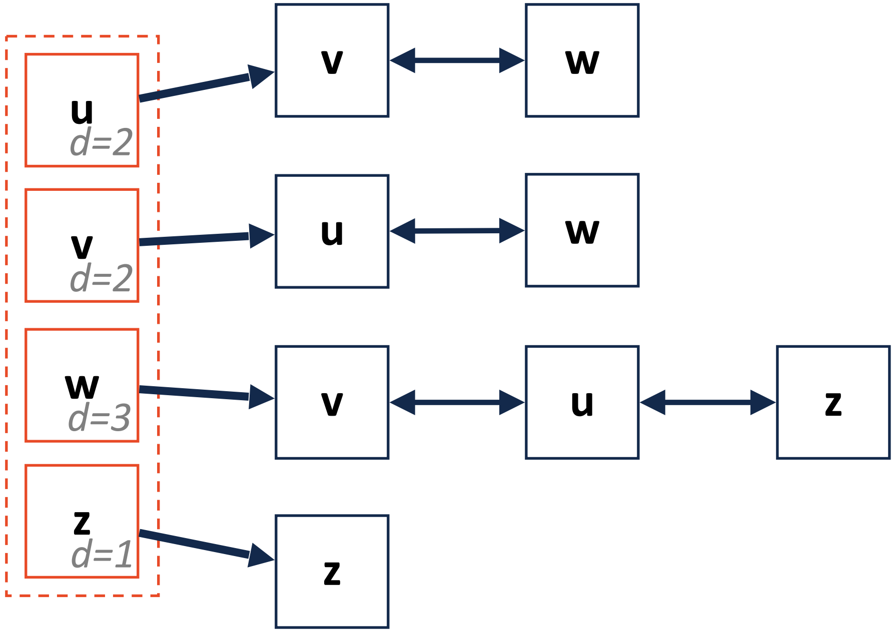


removeVertex(v):

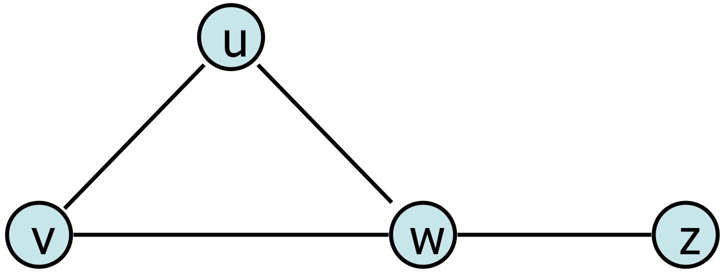
Adjacency List



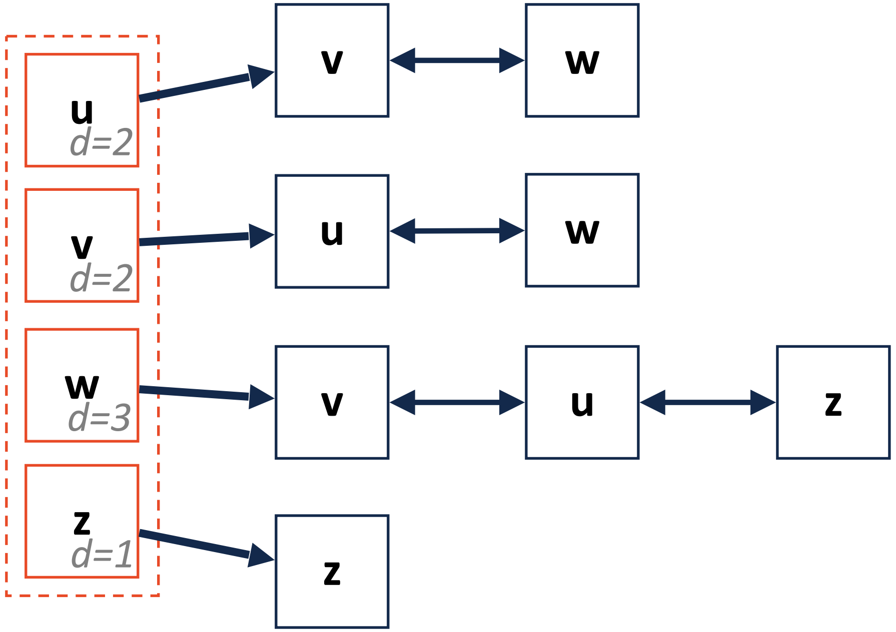
insertEdge(u, v):



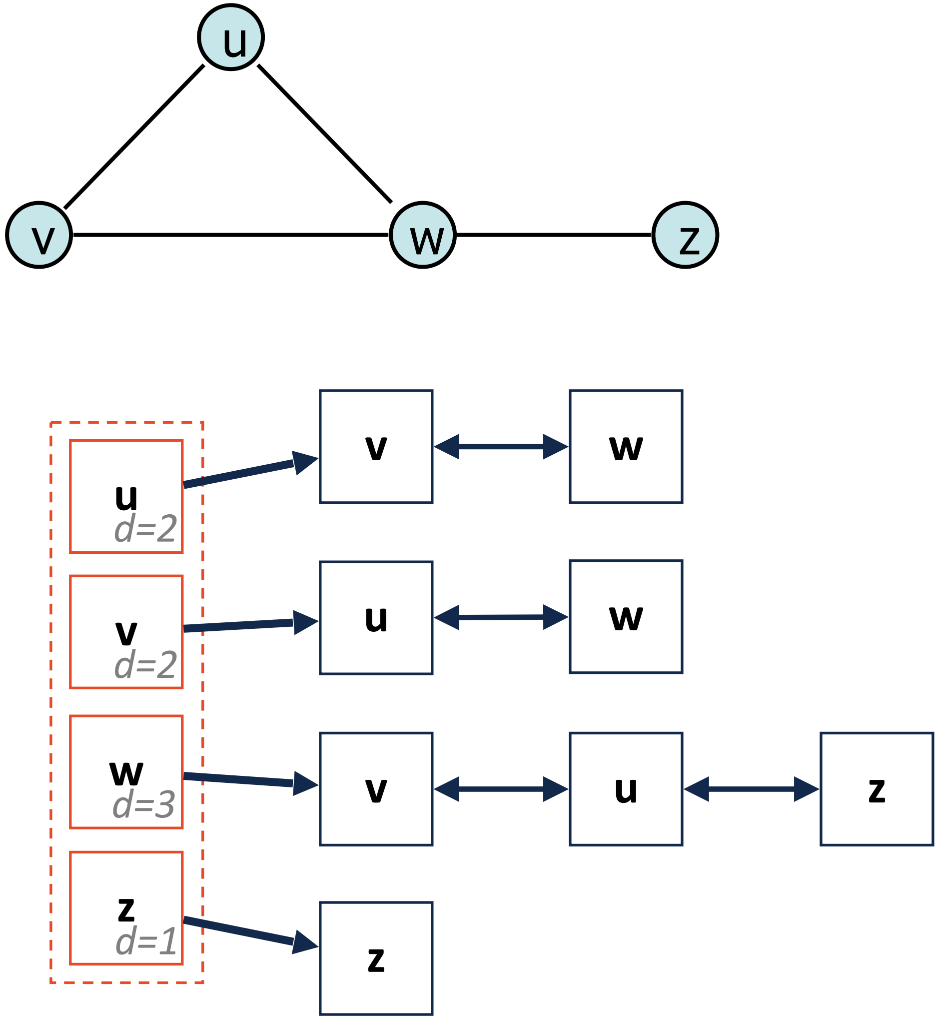
Adjacency List



removeEdge(u, v):



Adjacency List

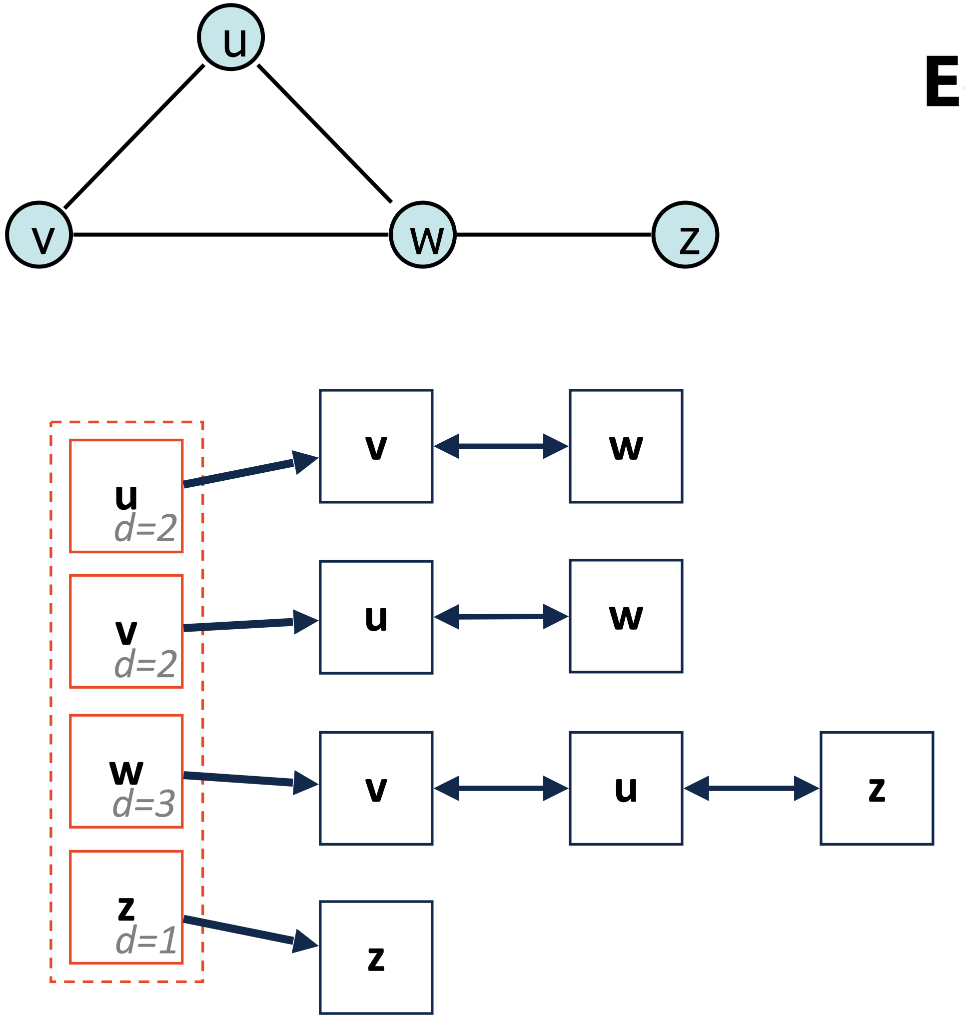


Pros:

Cons:

Adjacency List

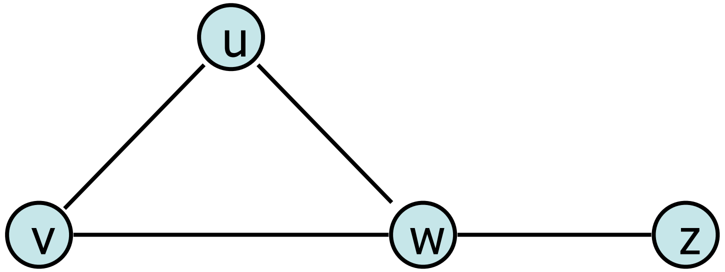
How would our data structure change if…



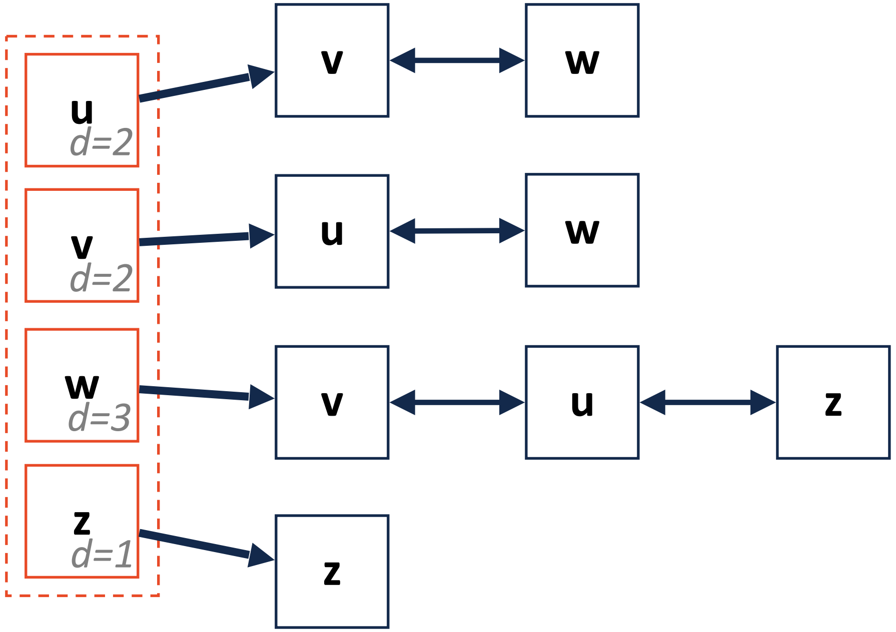
Edges are directed:

Adjacency List

How would our data structure change if…



Edges are weighted:



$|V| = n, |E| = m$

Expressed as O(f)	Edge List	Adjacency Matrix	Adjacency List
Space	n+m	n^2	n+m
insertVertex(v)	1*	n*	1*
removeVertex(v)	m**	n	deg(v)
insertEdge(u, v)	1	1	1*
removeEdge(u, v)	m	1	min(deg(u), deg(v))
getEdges(v)	m	n	deg(v)
areAdjacent(u, v)	m	1	min(deg(u), deg(v))

Next week: Traversals

There is no clear order in a graph (even less than a tree!)

How can we systematically go through a complex graph in the fewest steps?