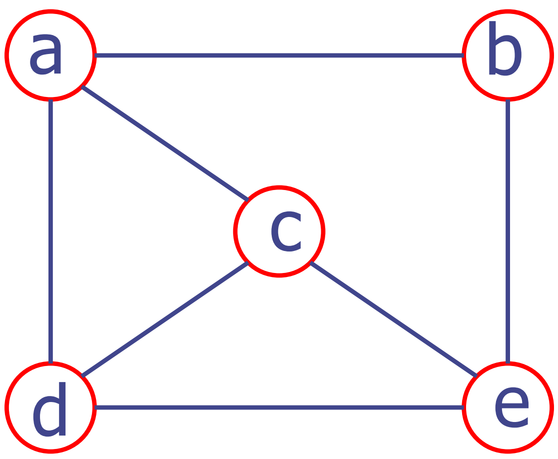
Graphs

A graph is a pair (V, E), where

- V is a set of nodes or vertices
- E is a collection of pairs of vertices (u,v), called edges, links, or arcs



V = {a,b,c,d,e}
E = {(a,b),(a,c),(a,d),
(b,e),(c,d),(c,e),(d,e)}

Edge Types

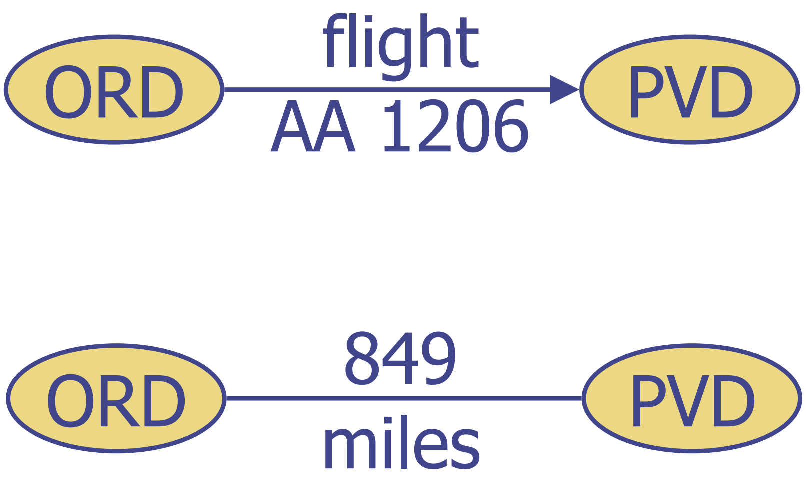
- Directed edge
  - ordered pair of vertices (u,v)
  - first vertex u is the origin
  - second vertex v is the destination

Edge Types

- Directed edge
  - ordered pair of vertices (u,v)
  - first vertex u is the origin
  - second vertex v is the destination
- Undirected edge
  - unordered pair of vertices (u,v)

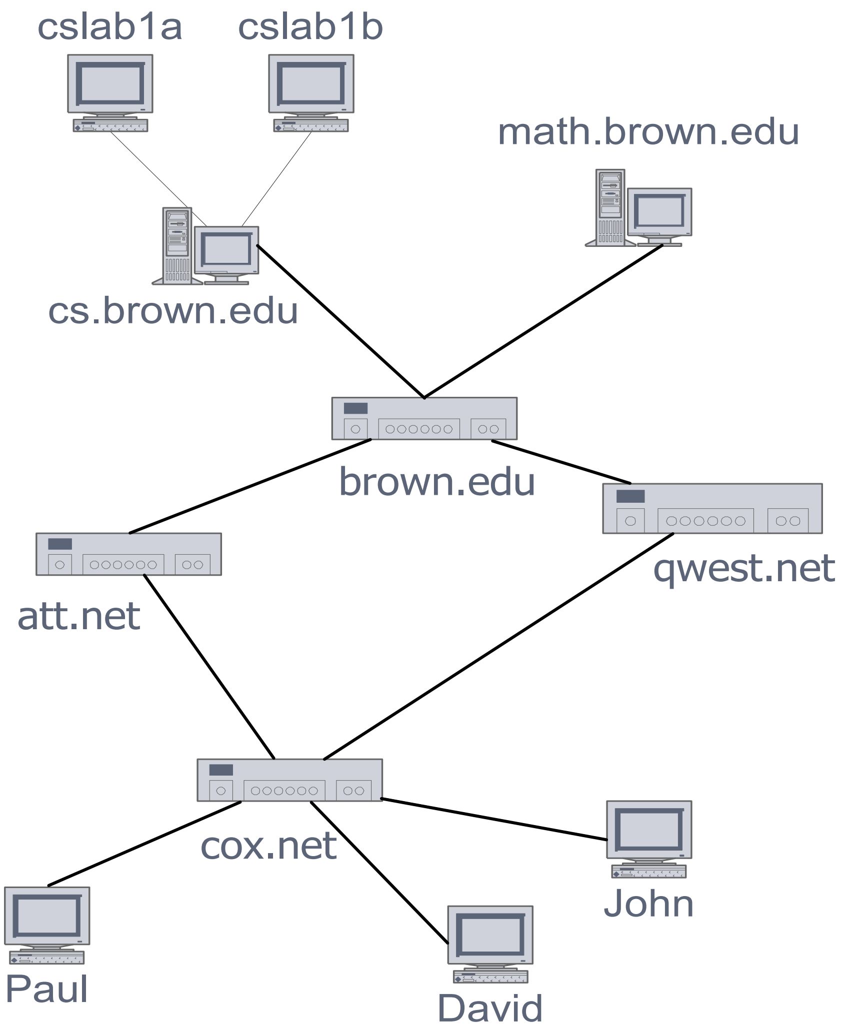
Edge Types

- Directed edge
  - ordered pair of vertices (u,v)
  - first vertex u is the origin
  - second vertex v is the destination
- Undirected edge
  - unordered pair of vertices (u,v)
- Directed graph or digraph
  - all the edges are directed
- Undirected graph
  - all the edges are undirected
- Mixed graph
  - directed and undirected edges



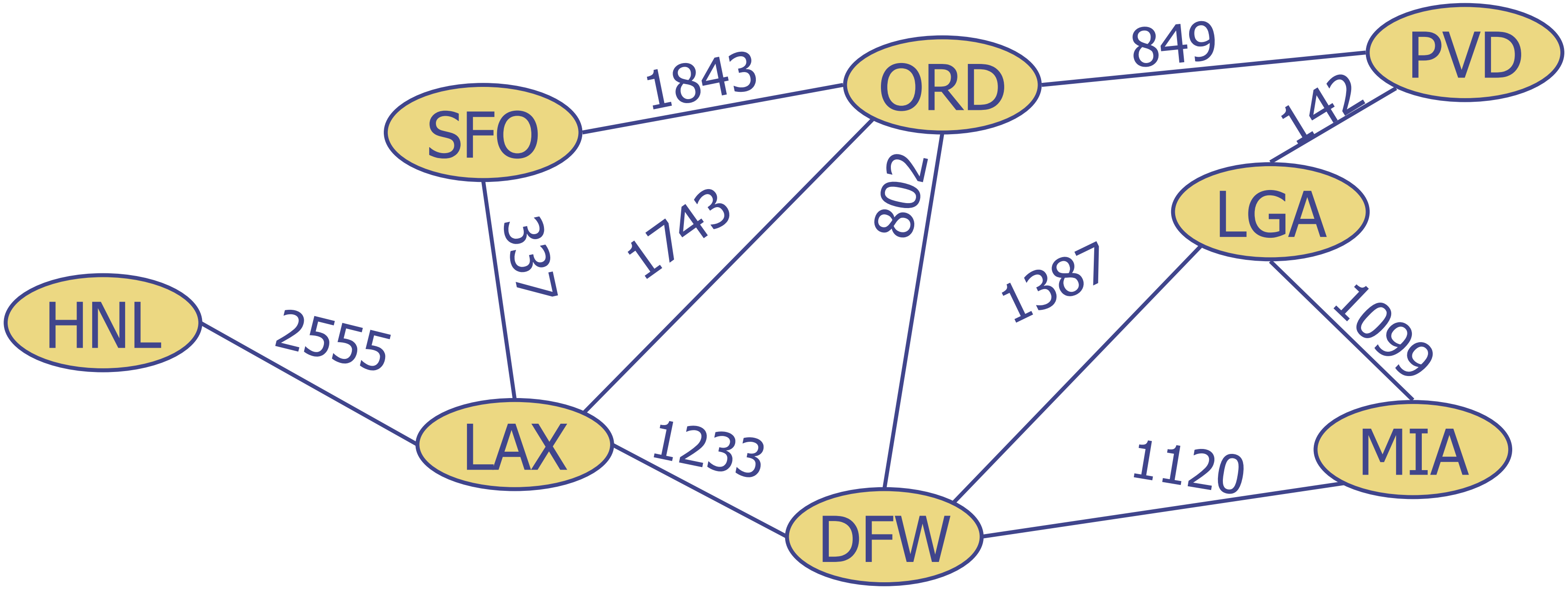
Applications

- Computer networks



Applications

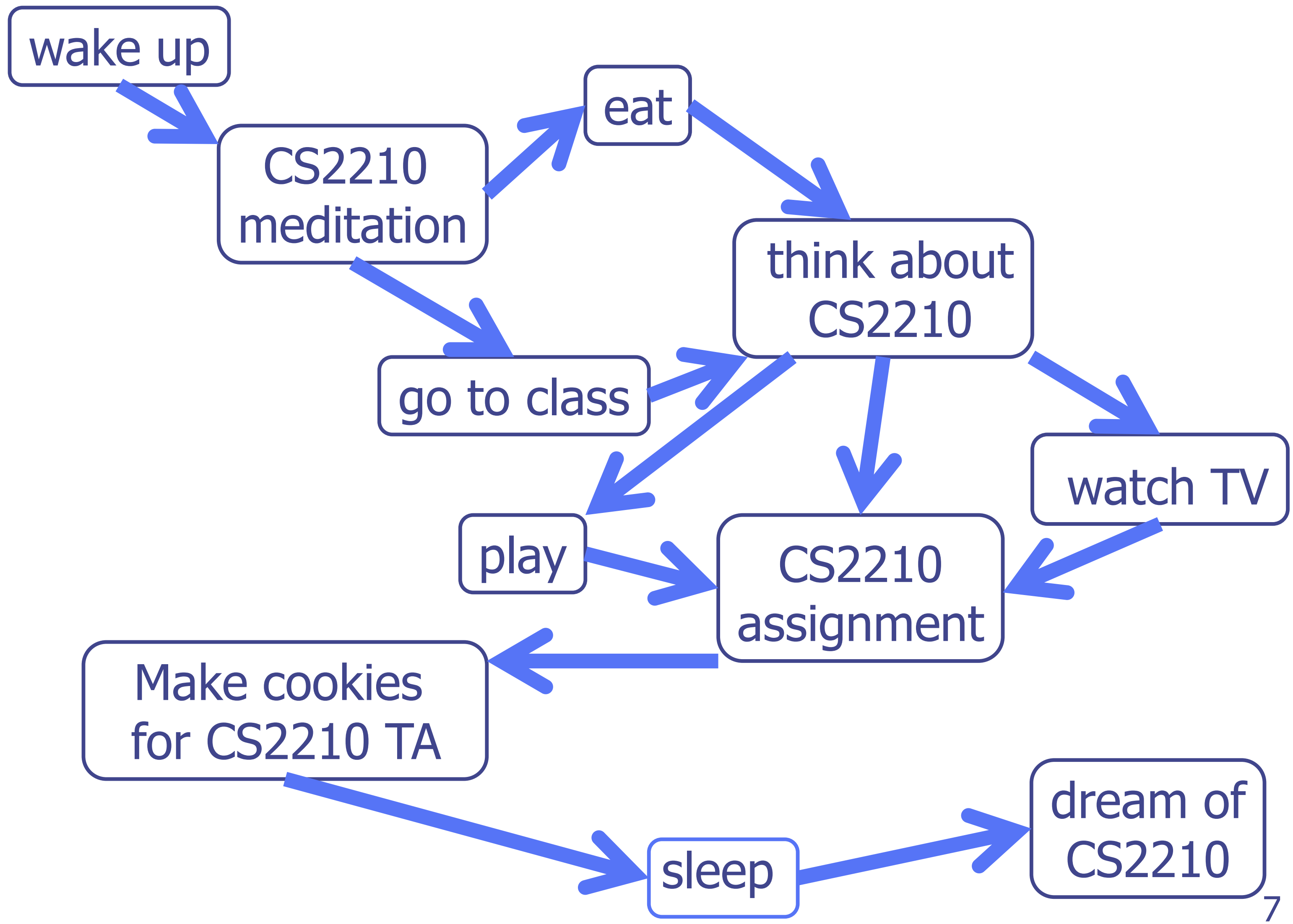
- Transportation networks



Applications

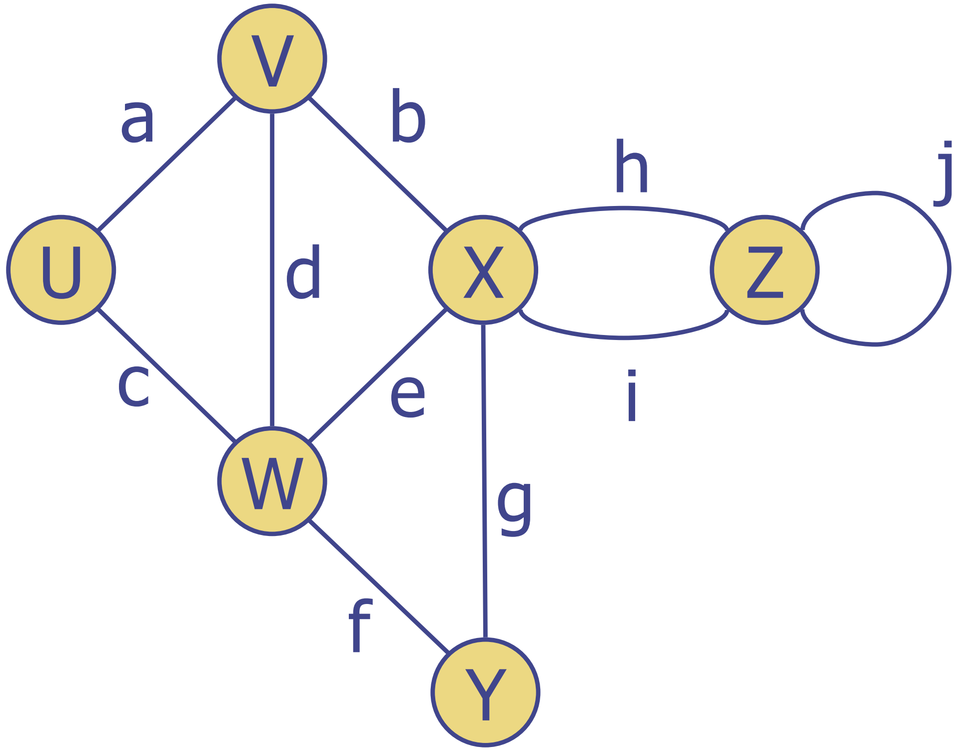
- Scheduling tasks

A typical student day



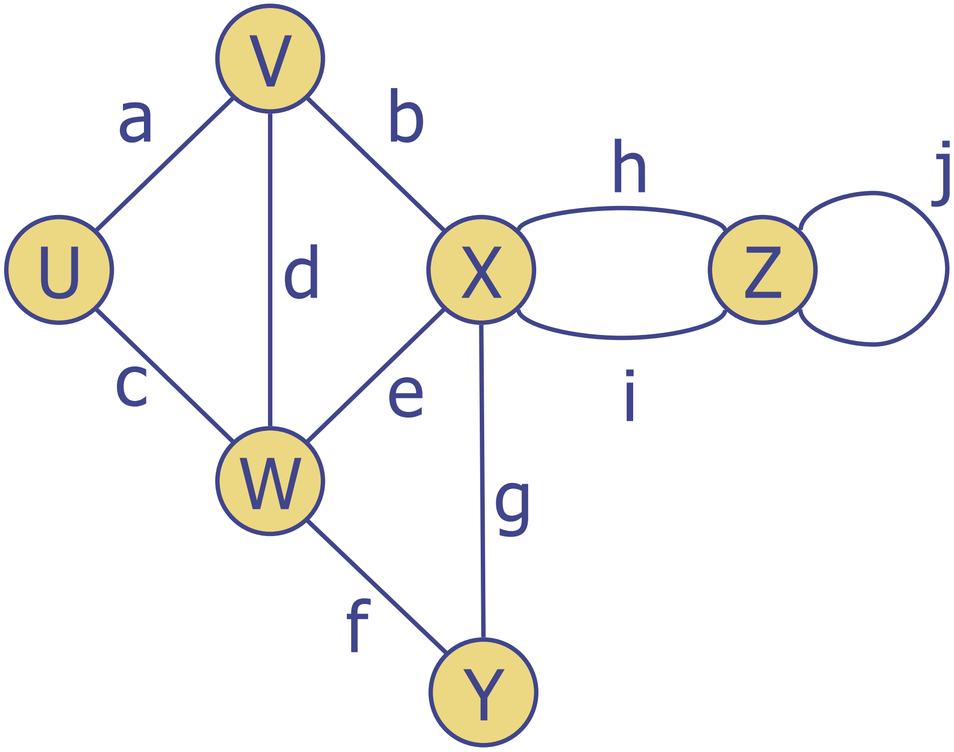
Terminology

- End vertices (or endpoints) of an edge
  - U and V are the endpoints of a



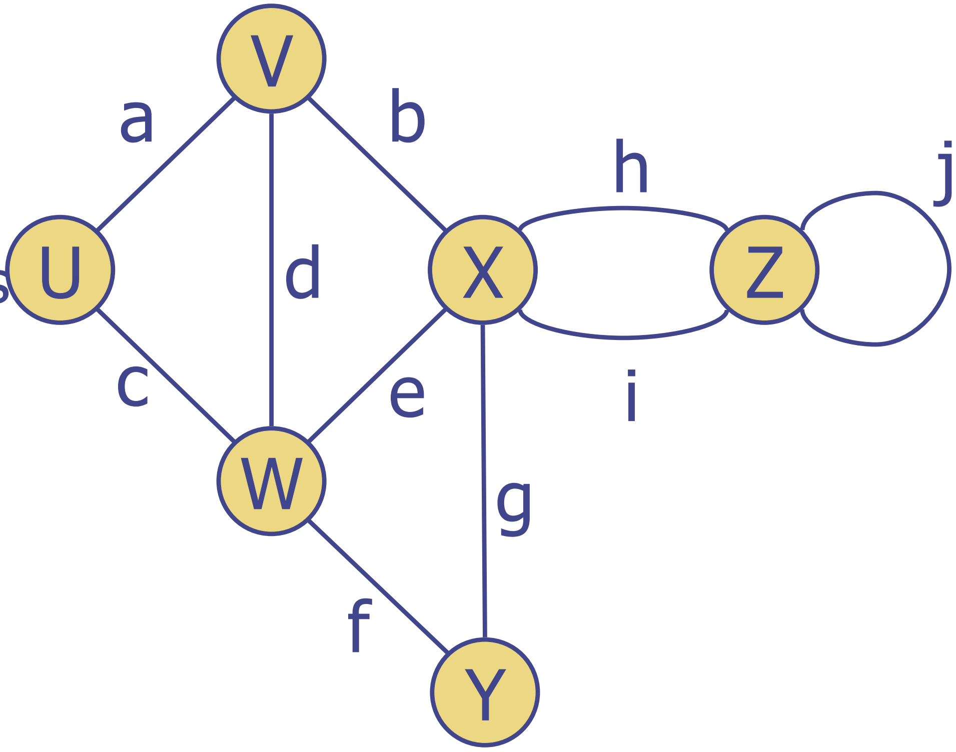
Terminology

- End vertices (or endpoints) of an edge
  - U and V are the endpoints of a
- Edges incident on a vertex
  - a, d, and b are incident on V



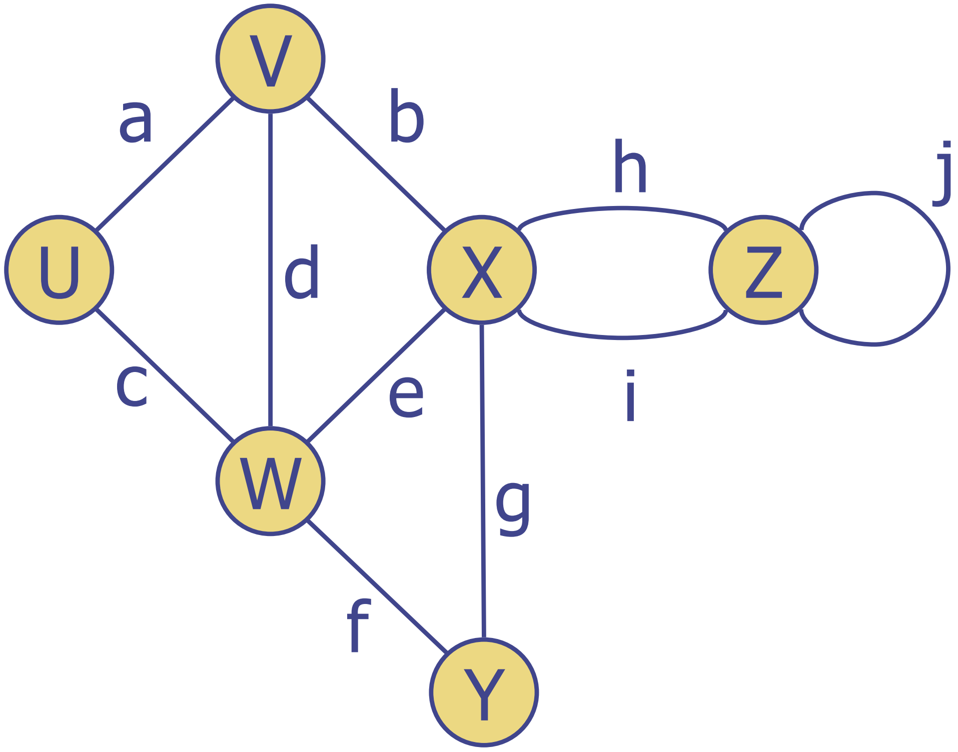
Terminology

- End vertices (or endpoints) of an edge
  - U and V are the endpoints of a
- Edges incident on a vertex
  - a, d, and b are incident on V
- Adjacent vertices or neighbours
  - U and V are adjacent



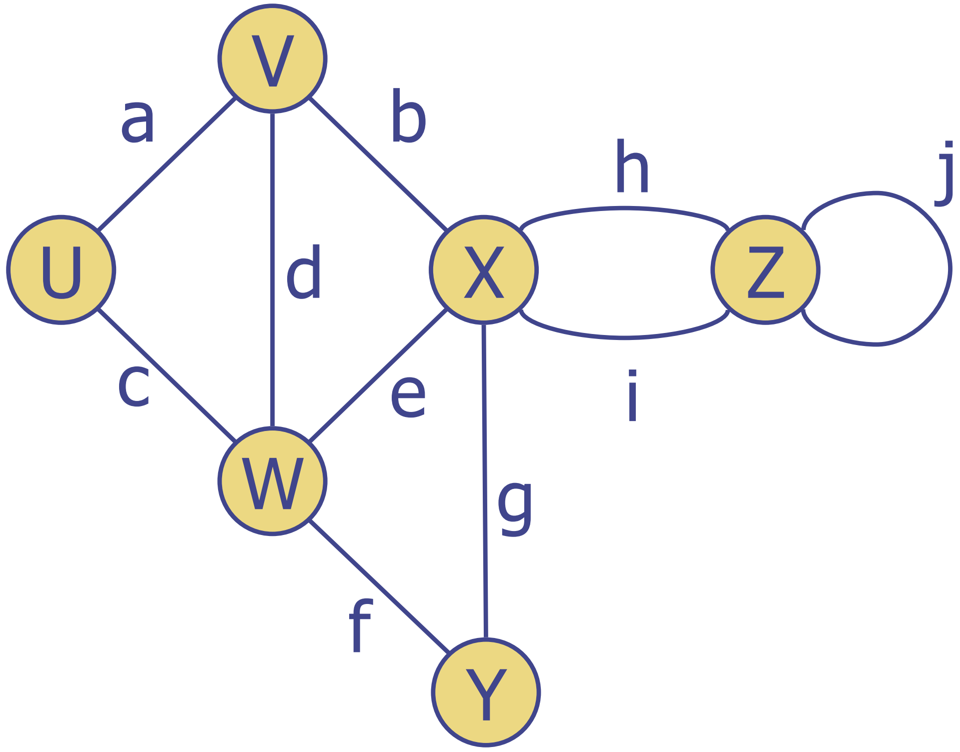
Terminology

- End vertices (or endpoints) of an edge
  - U and V are the endpoints of a
- Edges incident on a vertex
  - a, d, and b are incident on V
- Adjacent vertices
  - U and V are adjacent
- Degree of a vertex
  - X has degree 5



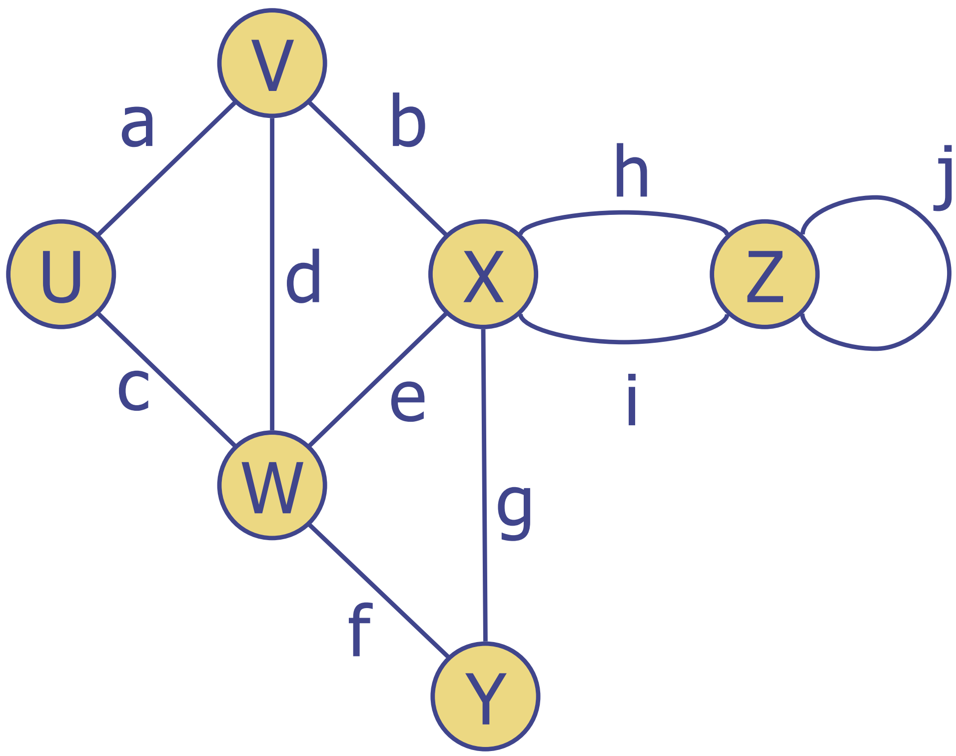
Terminology

- End vertices (or endpoints) of an edge
  - U and V are the endpoints of a
- Edges incident on a vertex
  - a, d, and b are incident on V
- Adjacent vertices
  - U and V are adjacent
- Degree of a vertex
  - X has degree 5
- Parallel edges
  - h and i are parallel edges
- Self-loop
  - j is a self-loop



Terminology

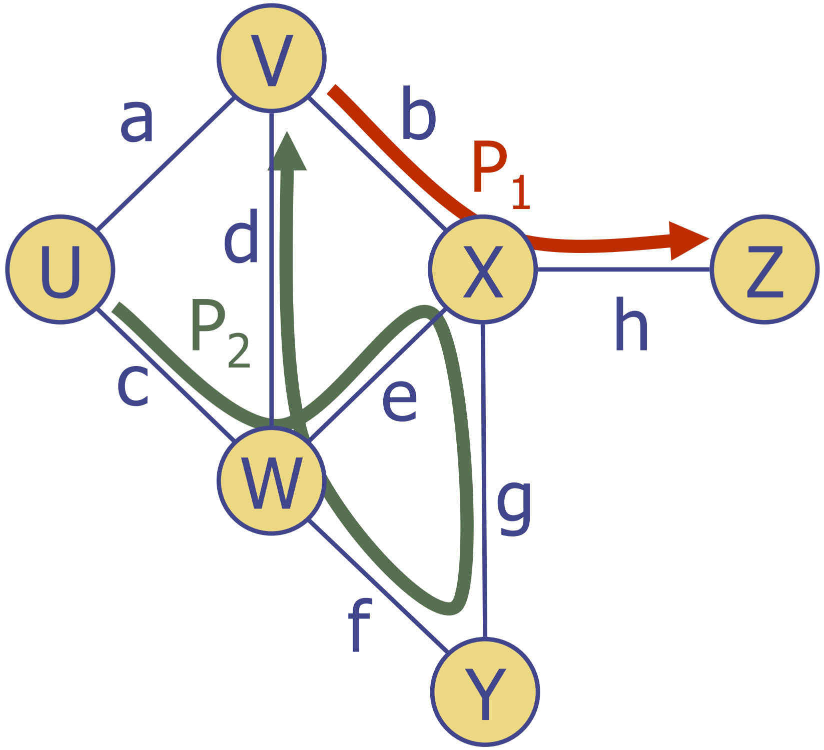
- End vertices (or endpoints) of an edge
  - U and V are the endpoints of a
- Edges incident on a vertex
  - a, d, and b are incident on V
- Adjacent vertices
  - U and V are adjacent
- Degree of a vertex
  - X has degree 5
- Parallel edges
  - h and i are parallel edges
- Self-loop
  - j is a self-loop



A graph without parallel edges or self-loops is called a simple graph

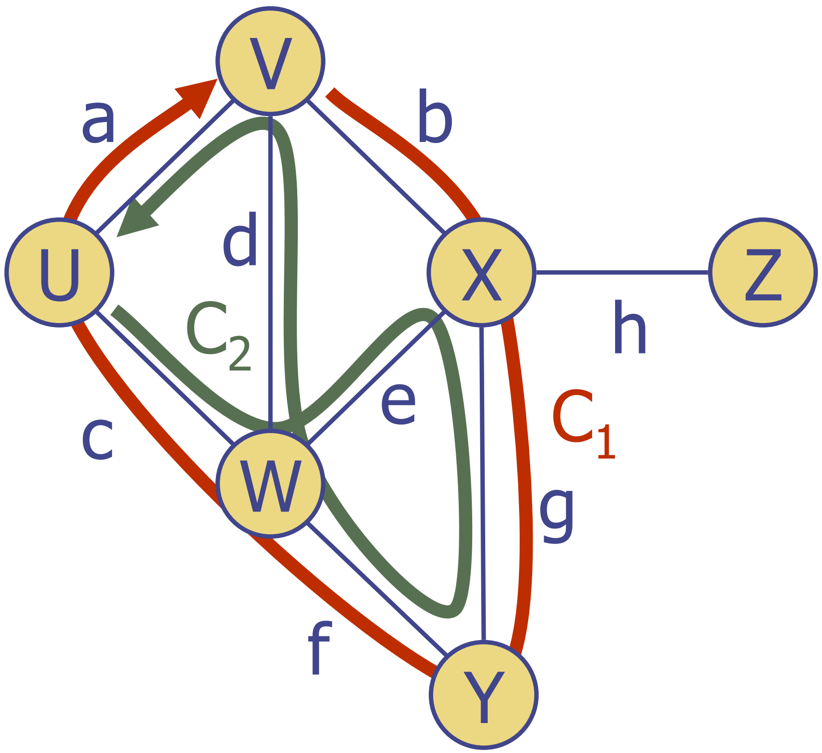
Terminology

- Path
  - sequence of adjacent vertices
- Simple path
  - path such that all its vertices and edges are distinct
- Examples
  - P_1=(V,X,Z) is a simple path
  - P_2=(U,W,X,Y,W,V) is a path that is not simple



Terminology

- Cycle
  - circular sequence of adjacent vertices
- Simple cycle
  - cycle such that all its vertices are distinct (except first and last)
- Examples
  - C_1=(V,X,Y,W,U,V) is a simple cycle
  - C_2=(U,W,X,Y,W,V,U) is a cycle that is not simple



Properties

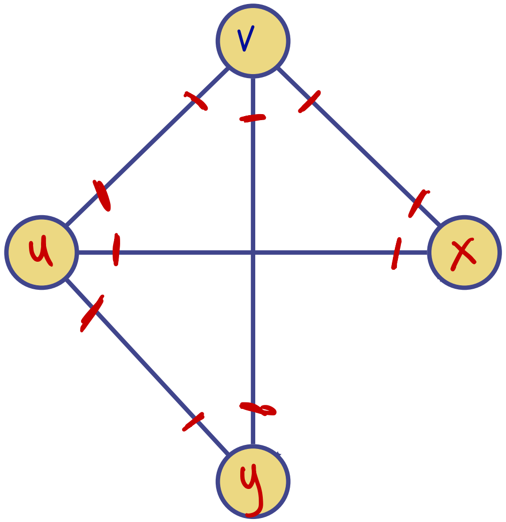
Notation

n	number of vertices
m	number of edges
$\deg(v)$	degree of vertex v

Property 1

$\Sigma_v \deg(v) = 2m$

$m = \frac{1}{2} \sum_v \deg(v)$

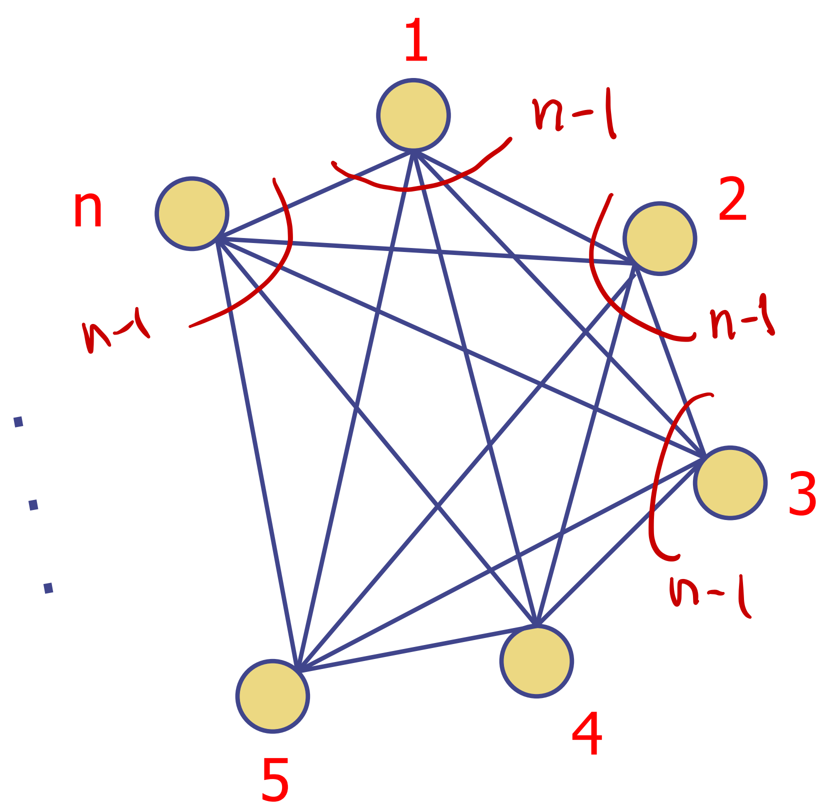


Example

- $n = 4$
- $m = 5$
- $\deg(v) = 3$
- $\Sigma_v \deg(v) = 10$

Complete Graph or Clique

Each vertex is connected to every other vertex.

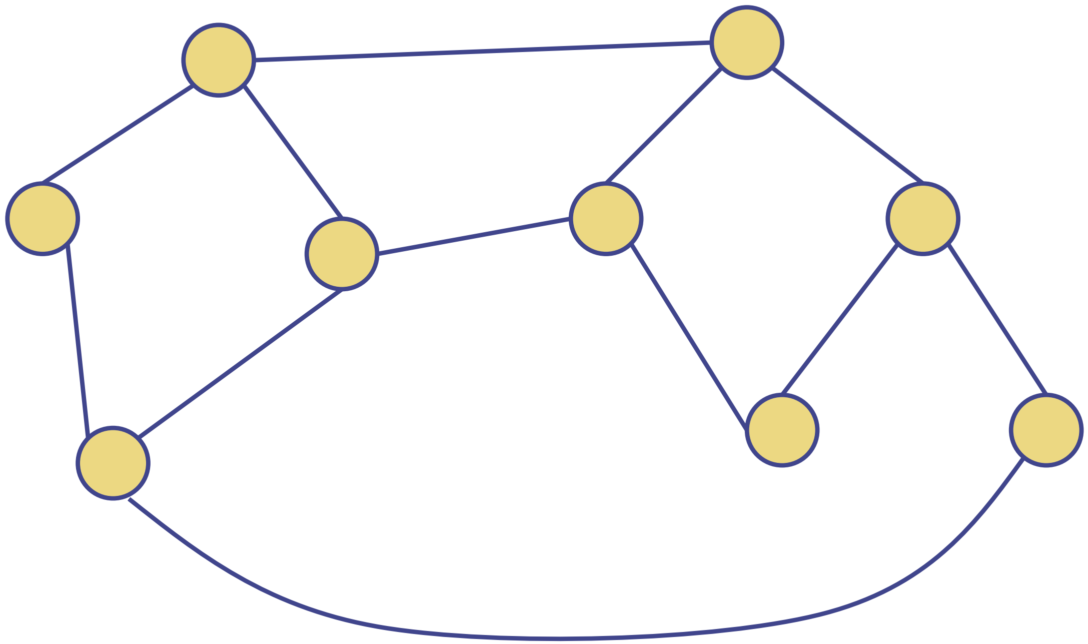


$$\sum_u \underbrace{degree(u)}_{n-1} = n(n-1) = 2m$$

m = n(n-1)/2

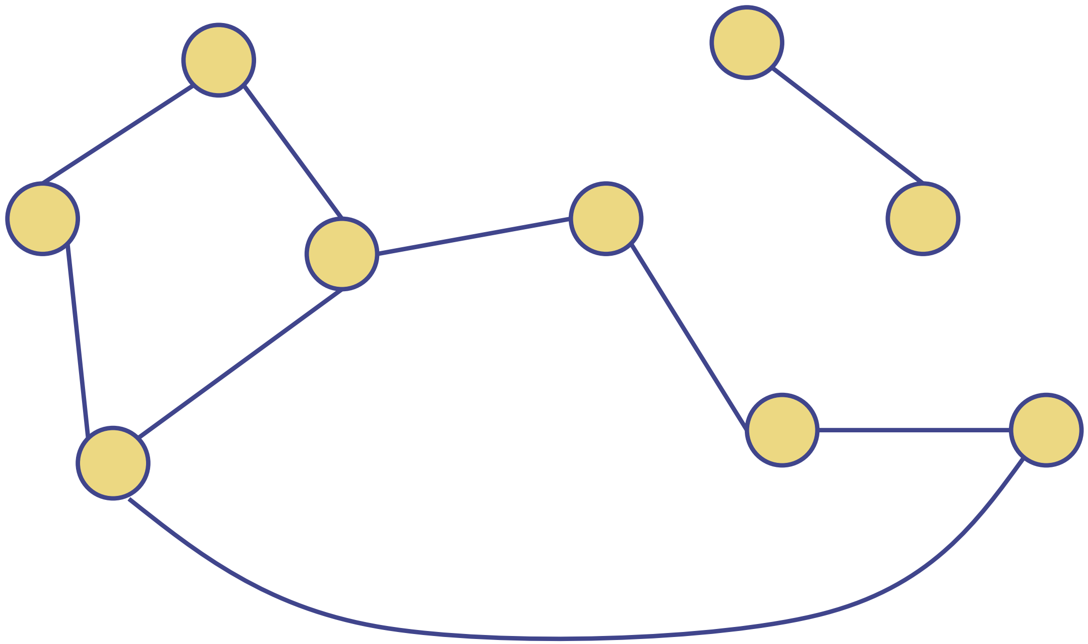
Connected Graph

A graph is connected if there is a path from each vertex to every other vertex



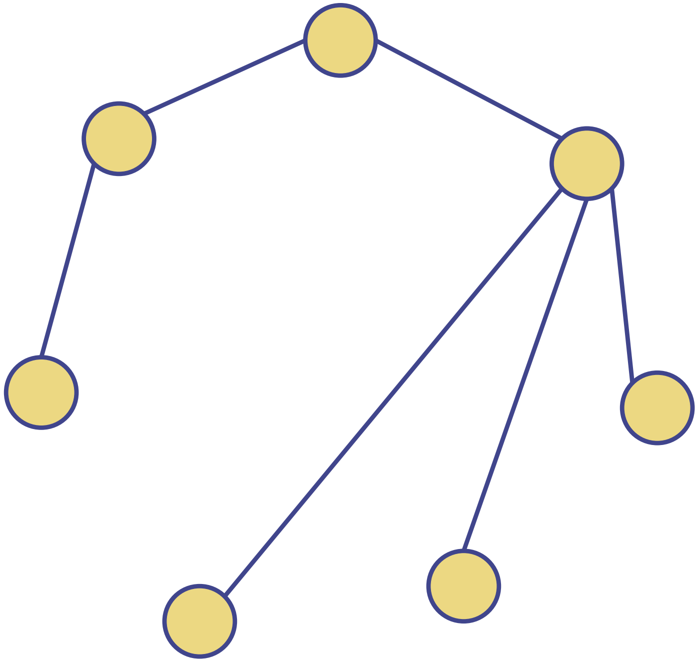
Connected Graph

A graph is connected if there is a path from each vertex to every other vertex



Trees

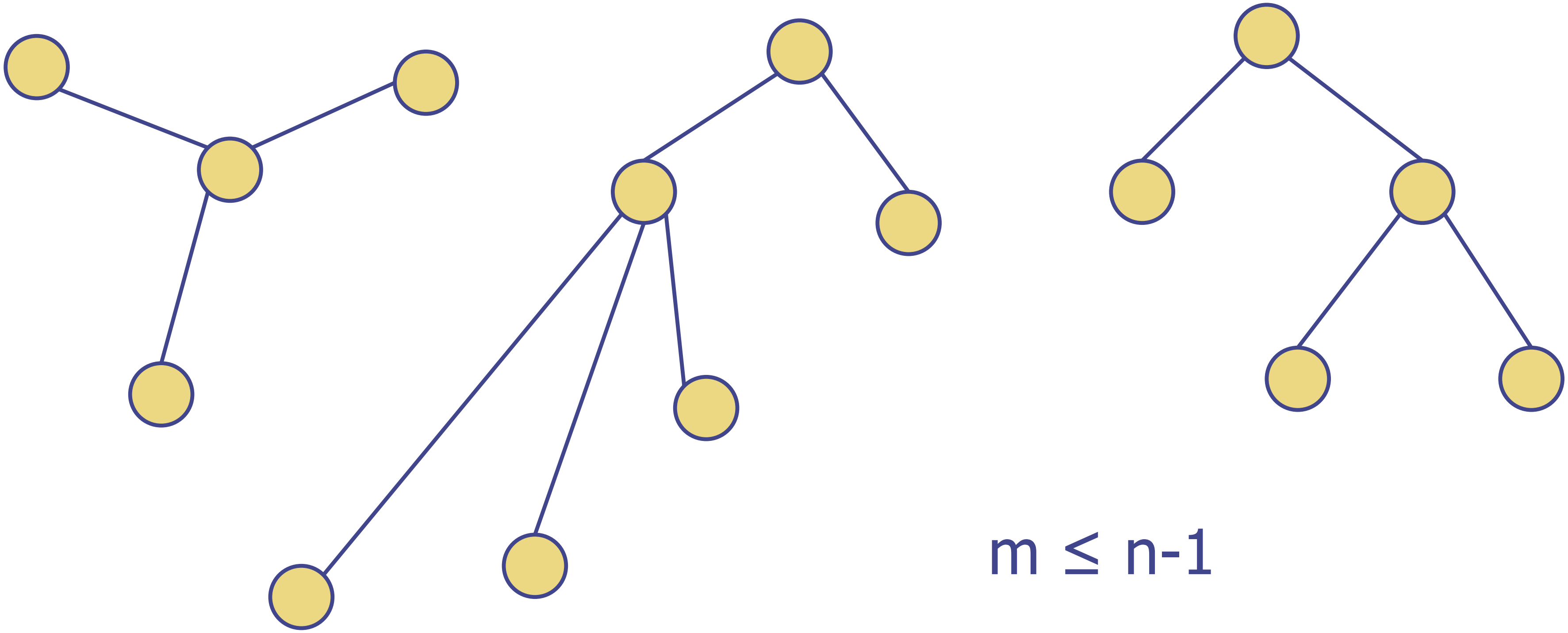
A tree is a graph without cycles.



$m = n-1$

Forest

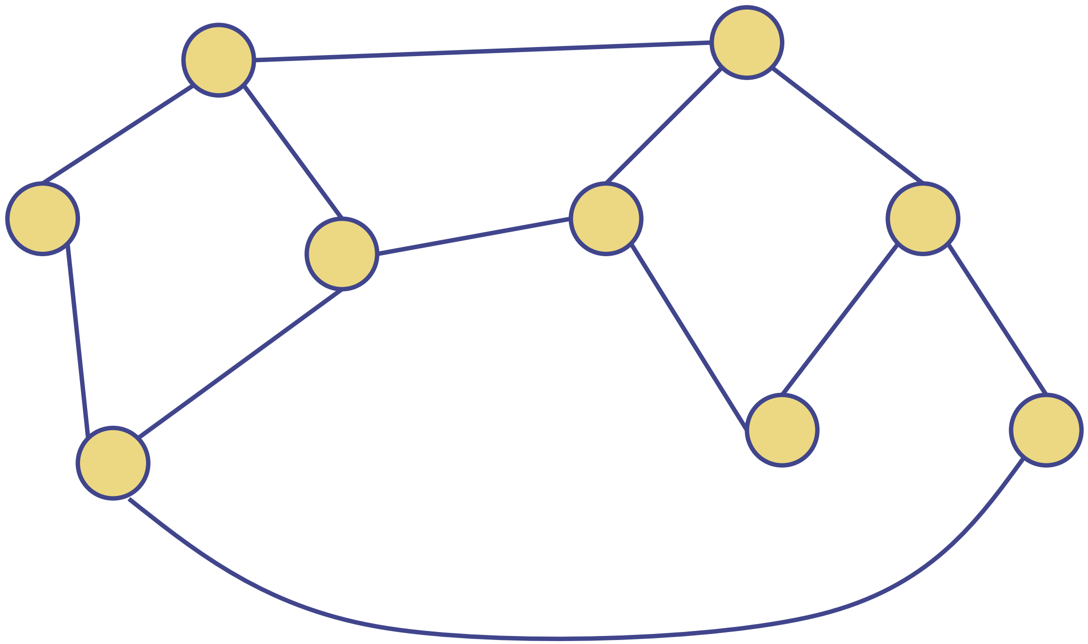
A forest is a set of trees.



$m \leq n-1$

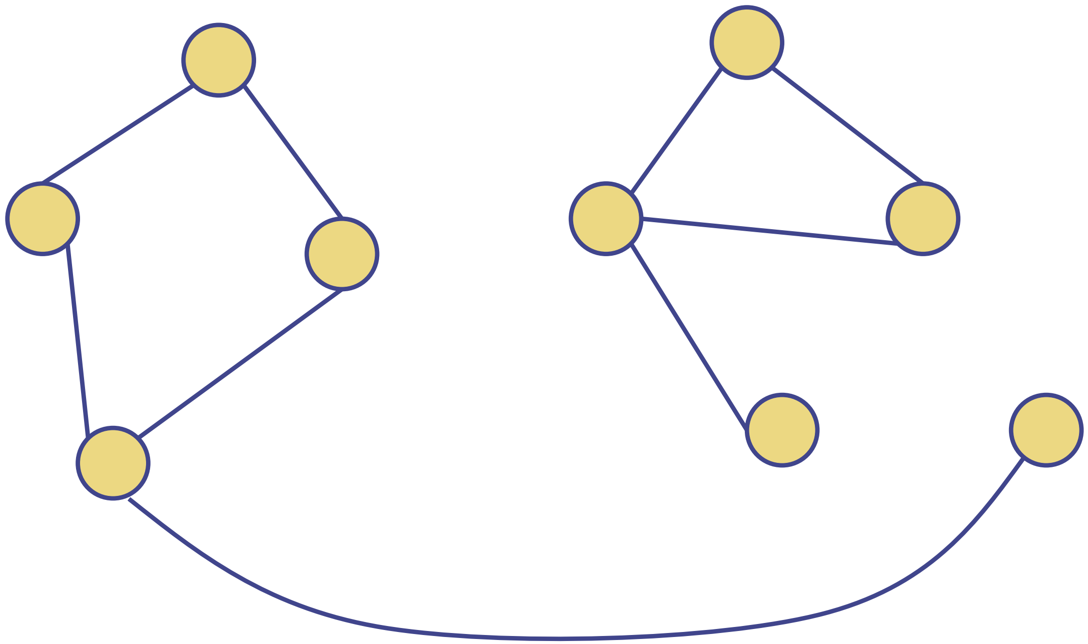
Subgraph

A subgraph is a subset of vertices and edges that forms a graph.



Connected Component

A connected component is a maximal connected subgraph.



How many connected components does this graph have?

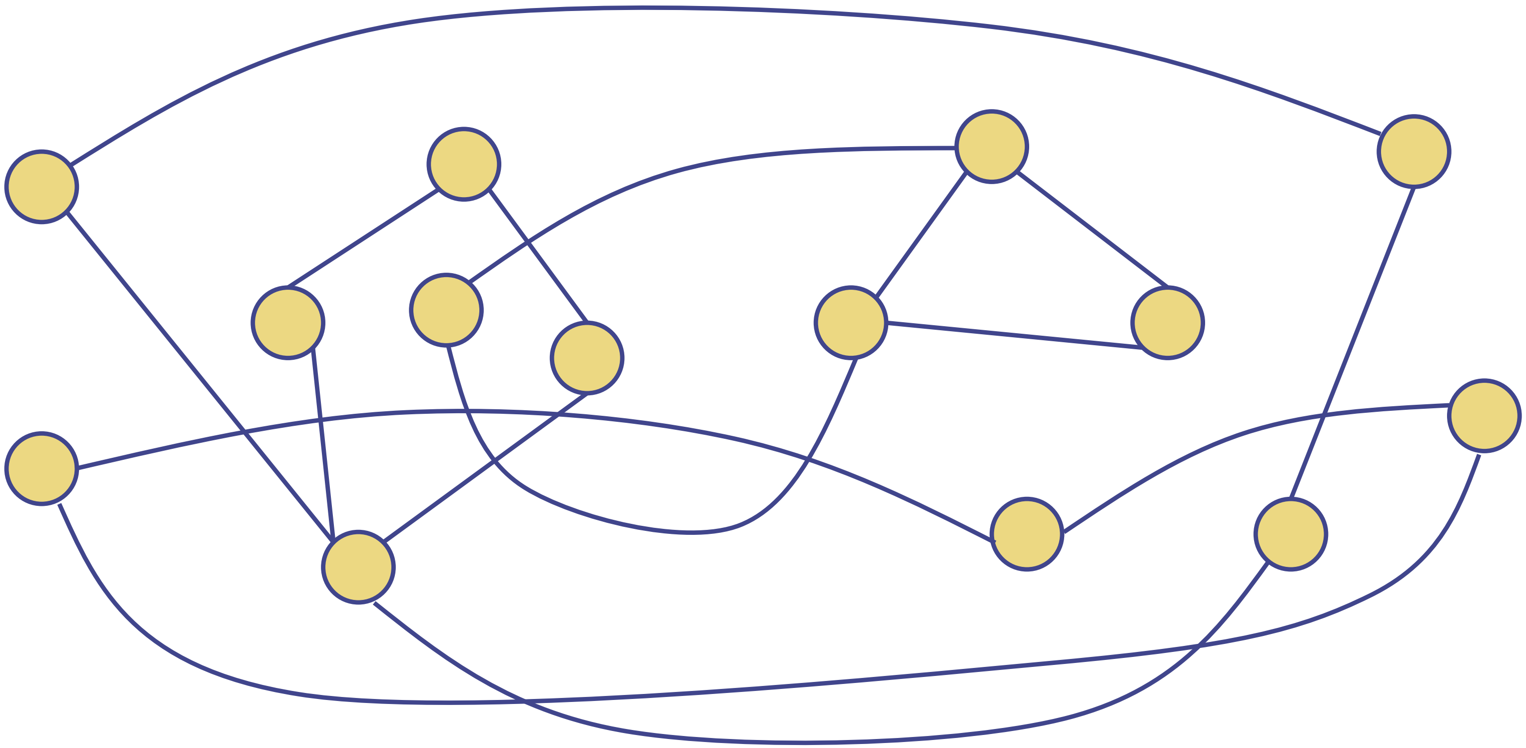
$G = (V, E)$

$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$

$E = \{(1,6), (1,12), (2,11), (3,4), (3,7), (4,6), (5,8),$
$(5,9), (6,7), (6,13), (8,9), (8,10), (9,10), (11,14), (12,13)\}$

Connected Component

A connected component is a maximal connected subgraph.



How many connected components?

Graph ADT

numVertices(): number of vertices of the graph

getEdge(u,v): returns the edge between vertices u and v

opposite(v,e): returns the vertex other than v that is incident on e

insertVertex(x): creates and returns a new vertex storing value x

insertEdge(u,v,x): creates an edge between u and v soring value x

removeVertex(v): removes vertex v and all edges incident on it

removeEdge(e): removes edge e

areAdjacent(u,v): returns true is u and v are adjacent; false otherwise

incidentEdges(u): returns an iterator of all edges incident on vertex u.

Data Structures to Store Graphs

Edge List

V

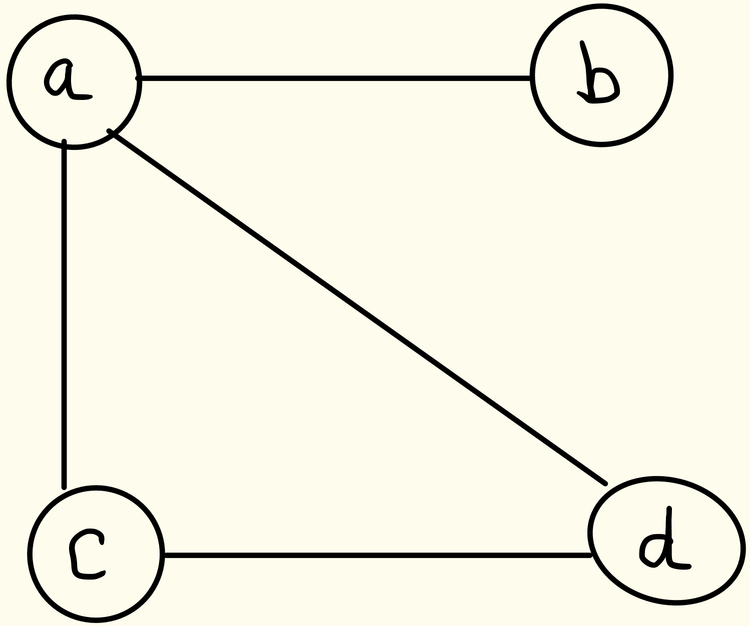
a	b	c	d

n

E

(a,b)	(a,c)	(a,d)	(c,d)

m



Space: $c_1 n + c_2 m$ is $O(n+m)$

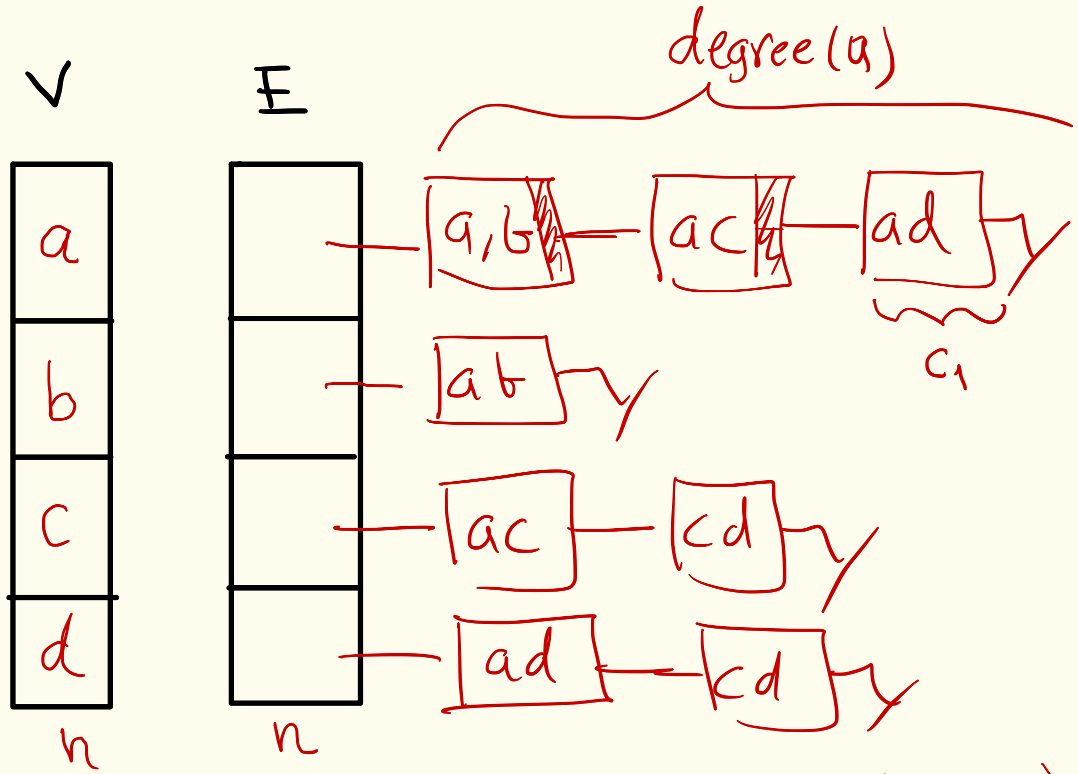
Are Adjacent(u,v): $O(m)$

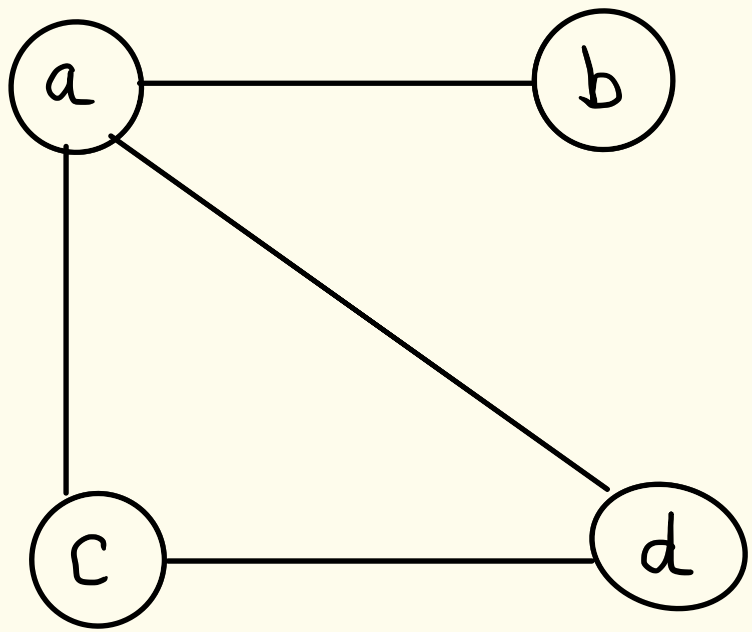
Incident Edges(u): $O(m)$

Data Structures to Store Graphs

Adjacency List

Sparse graphs
("few" edges)





Space: $c_2 n + c_1(2m)$ is $O(n+m)$

Are adjacent (u,v): $O(\min\{\text{degree}(u), \text{degree}(v)\})$

Incident Edges (u): $O(\text{degree}(u))$

Data Structures to Store Graphs

Adjacency Matrix

V

a
b
c
d

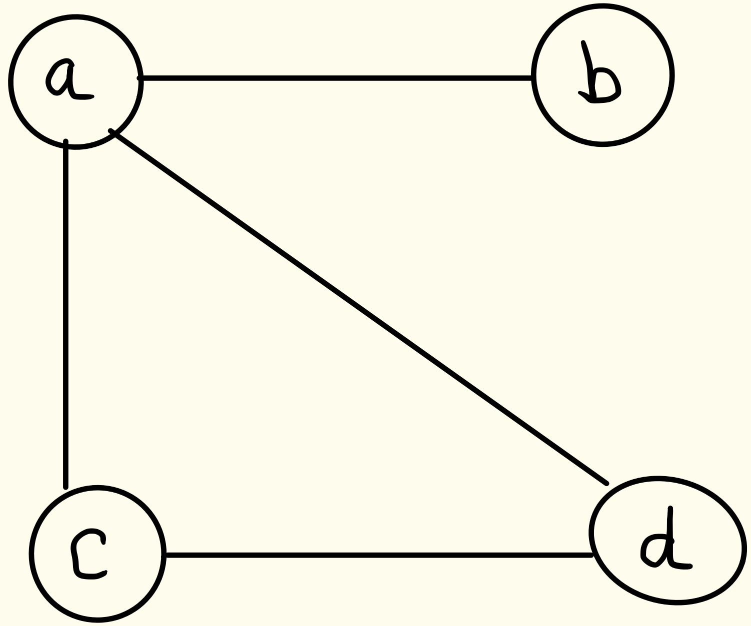
$O(n)$

E

	a	b	c	d
a		(a,b)	(a,c)	(a,d)
b	(b,a)			
c	(c,a)			(c,d)
d	(d,a)		d(c)	

$O(n \times n)$

Dense graphs
("Many edges")



Space: $O(n^2)$

Are Adjacent(u,v): $O(1)$

Incident Edges(u): $O(n)$

Performance

▪ n vertices, m edges	Edge List	Adjacency List	Adjacency Matrix
Space	$O(n + m)$	$O(n + m)$	$O(n^2)$
incidentEdges(v)	$O(m)$	$O(\deg(v))$	$O(n)$
areAdjacent (v, w)	$O(m)$	$O(\min\{\deg(v), \deg(w)\})$	$O(1)$
insertVertex(o)	$O(1)$	$O(1)$	$O(n^2)$
insertEdge(v, w, o)	$O(1)$	$O(1)$	$O(1)$
removeVertex(v)	$O(m)$	$O(\deg(v))$	$O(n^2)$
removeEdge(v,w)	O(m)	O(deg(u)+deg(v))	O(1)