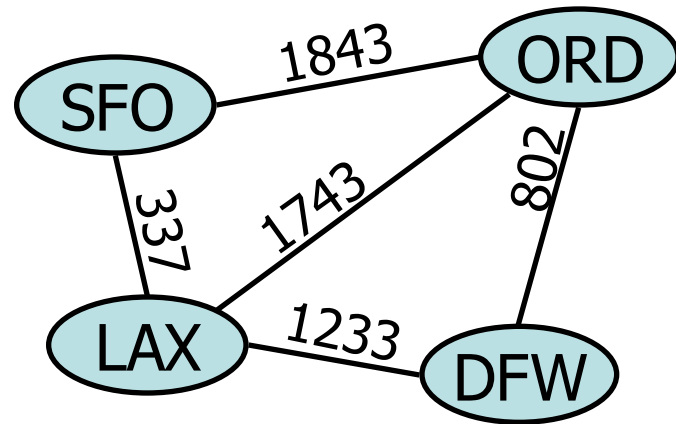


Graphs

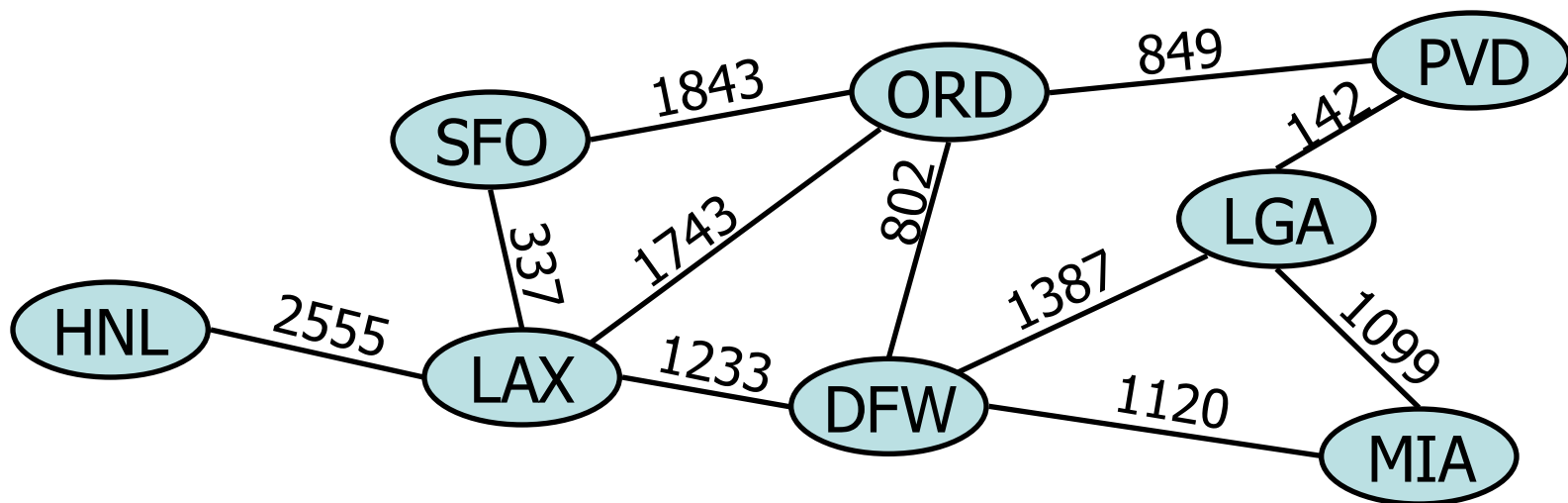


Outline and Reading

- Graphs
 - Definition
 - Applications
 - Terminology
 - Properties
 - ADT
- Data structures for graphs
 - Edge list structure
 - Adjacency list structure
 - Adjacency matrix structure

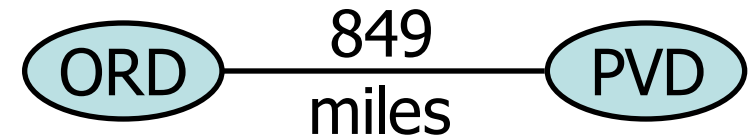
Graph

- A graph is a pair (V, E) , where
 - V is a set of nodes, called vertices
 - E is a collection of pairs of vertices, called edges
 - Vertices and edges are positions and store elements
- Example:
 - A vertex represents an airport and stores the three-letter airport code
 - An edge represents a flight route between two airports and stores the mileage of the route



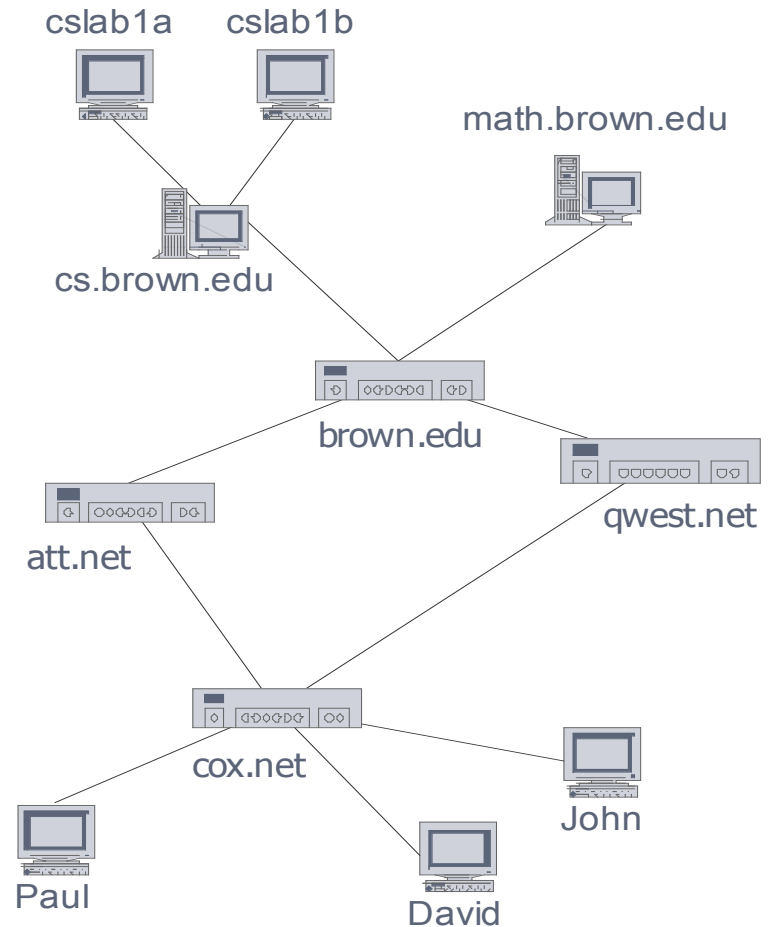
Edge Types

- Directed edge
 - ordered pair of vertices (u, v)
 - first vertex u is the origin
 - second vertex v is the destination
 - e.g., a flight
- Undirected edge
 - unordered pair of vertices (u, v)
 - e.g., a flight route
- Directed graph
 - all the edges are directed
 - e.g., flight network
- Undirected graph
 - all the edges are undirected
 - e.g., route network



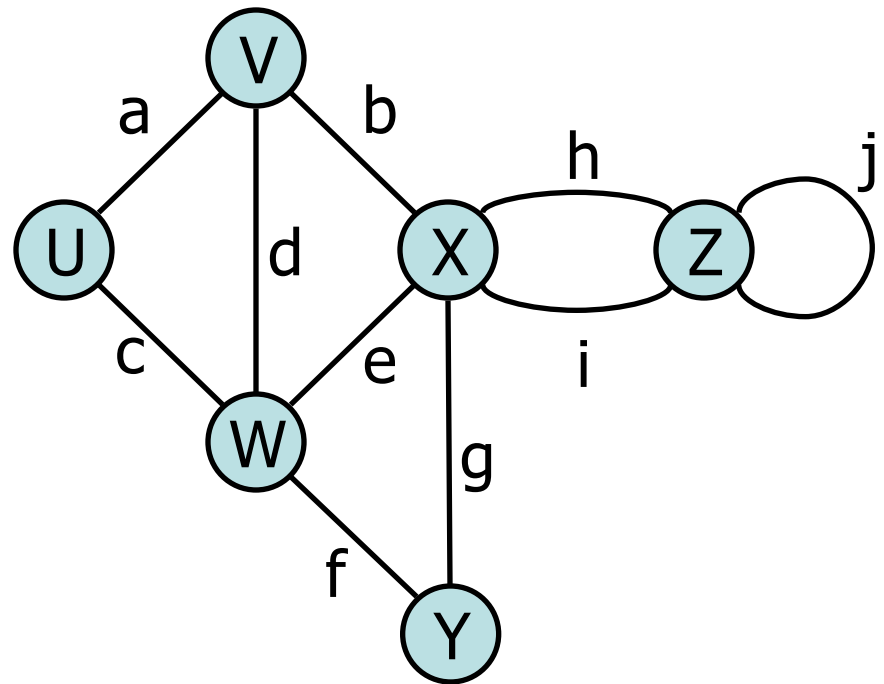
Applications

- Electronic circuits
 - Printed circuit board
 - Integrated circuit
- Transportation networks
 - Highway network
 - Flight network
- Computer networks
 - Local area network
 - Internet
 - Web
- Databases
 - Entity-relationship diagram



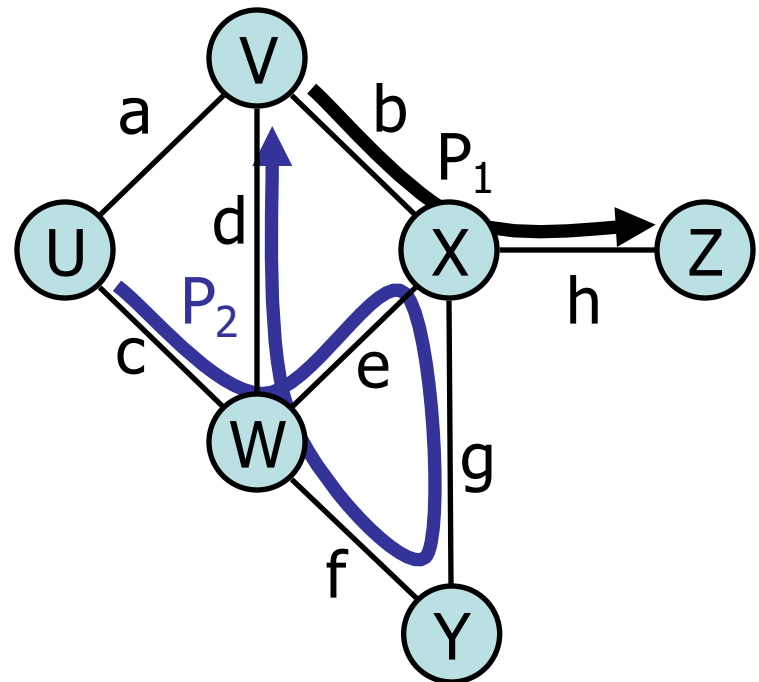
Terminology

- 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



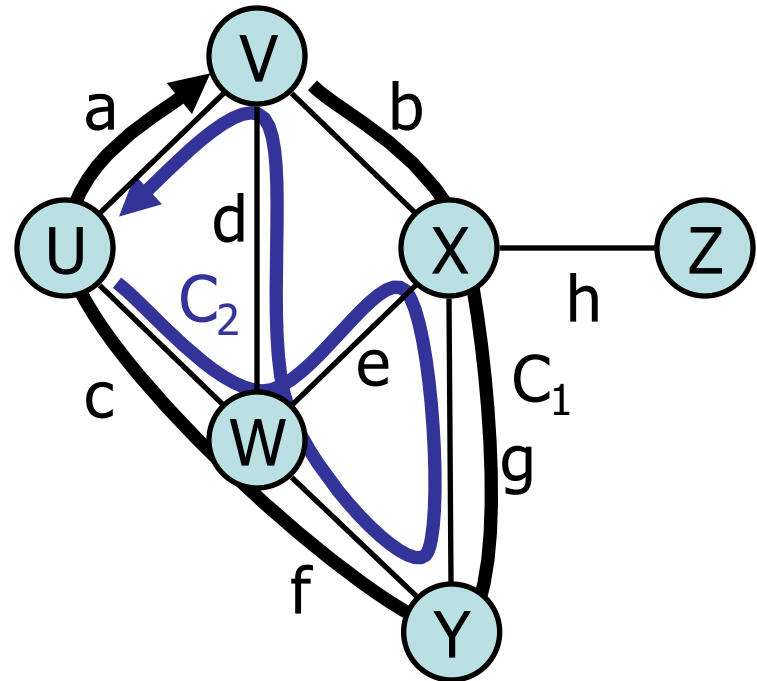
Terminology (cont.)

- Path
 - sequence of alternating vertices and edges
 - begins with a vertex
 - ends with a vertex
 - each edge is preceded and followed by its endpoints
- Simple path
 - path such that all its vertices and edges are distinct
- Examples
 - $P_1 = (V, b, X, h, Z)$ is a simple path
 - $P_2 = (U, c, W, e, X, g, Y, f, W, d, V)$ is a path that is not simple



Terminology (cont.)

- Cycle
 - circular sequence of alternating vertices and edges
 - each edge is preceded and followed by its endpoints
- Simple cycle
 - cycle such that all its vertices and edges are distinct
- Examples
 - $C_1 = (V, b, X, g, Y, f, W, c, U, a, \hookrightarrow)$ is a simple cycle
 - $C_2 = (U, c, W, e, X, g, Y, f, W, d, V, a, \hookrightarrow)$ is a cycle that is not simple



Properties

Property 1

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

Proof: each endpoint is counted twice

Notation

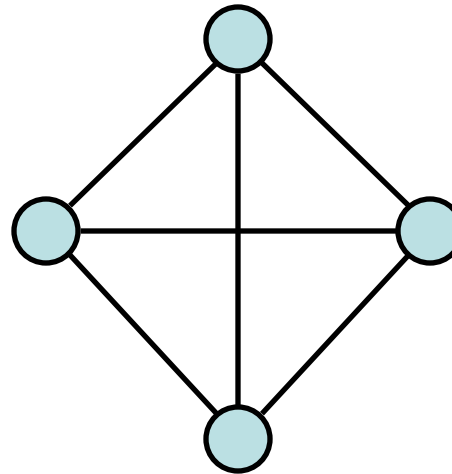
n	number of vertices
m	number of edges
$\deg(v)$	degree of vertex v

Property 2

In an undirected graph with no self-loops and no multiple edges

$$m \leq n(n-1)/2$$

Proof: each vertex has degree at most $(n-1)$

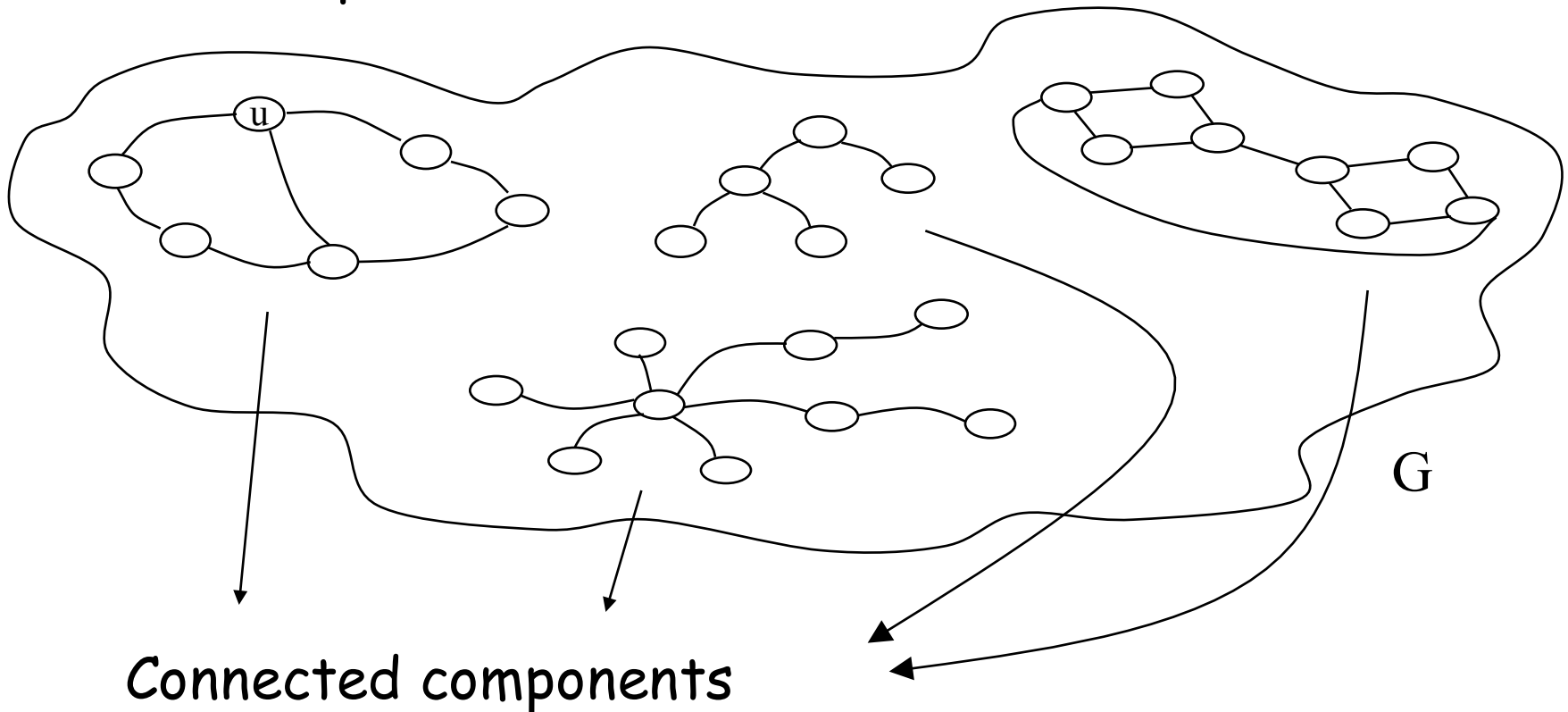


Example

- $n = 4$
- $m = 6$
- $\deg(v) = 3$

Connected Graphs

A (non-directed) graph is connected if there exists a path $\forall u, v \in V$.



Main Methods of the Graph ADT

- Vertices and edges
 - are positions
 - store elements
- Accessor methods
 - aVertex()
 - incidentEdges(v)
 - endVertices(e)
 - isDirected(e)
 - origin(e)
 - destination(e)
 - opposite(v, e)
 - areAdjacent(v, w)
- Update methods
 - insertVertex(o)
 - insertEdge(v, w, o)
 - insertDirectedEdge(v, w, o)
 - removeVertex(v)
 - removeEdge(e)
- Generic methods
 - numVertices()
 - numEdges()
 - vertices()
 - edges()

There could be other methods

Representations

Edge List

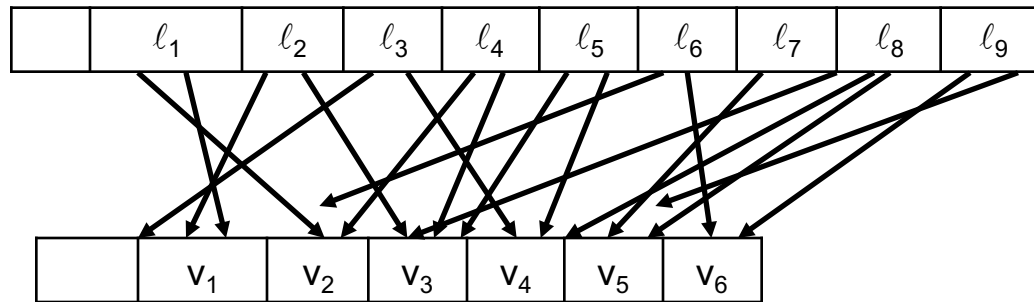
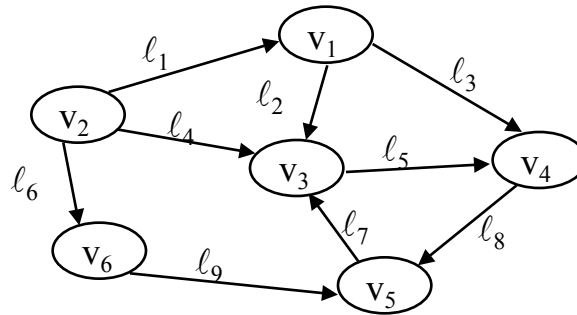
Adjacency List

Adjacency Matrix

Incidence Matrix

n = number of nodes
 m = number of edges

Edge List Structure (example)

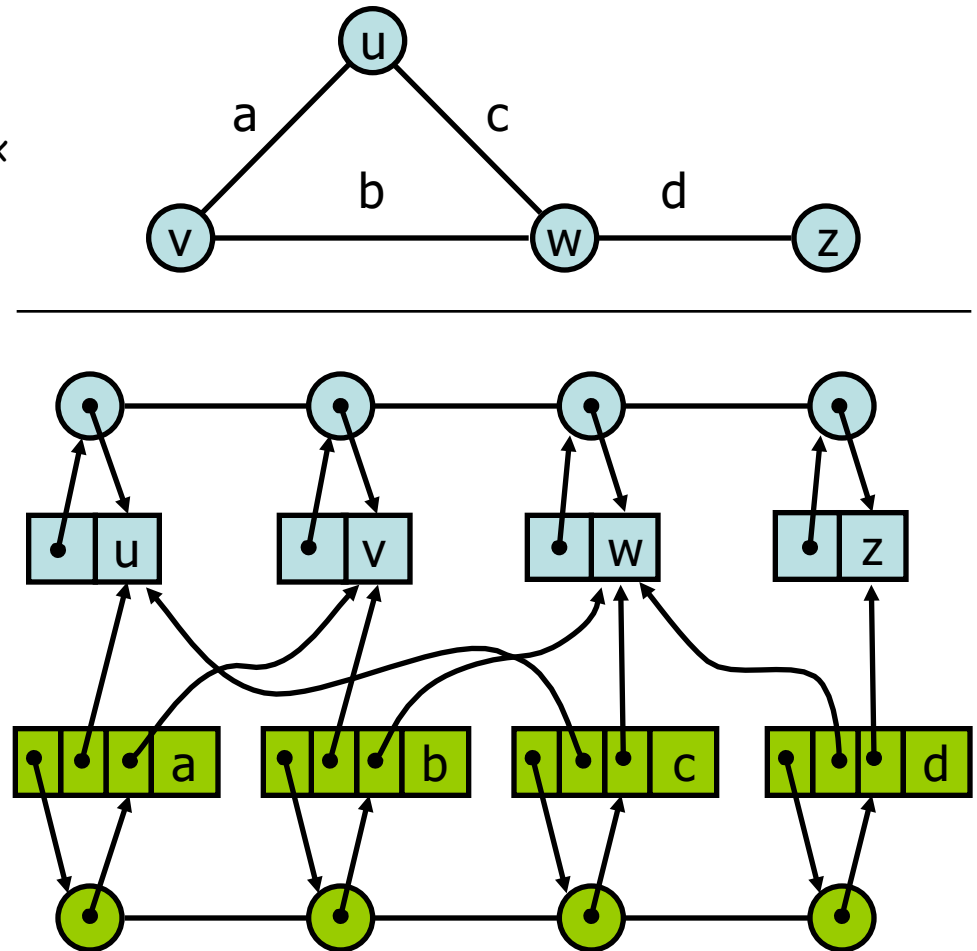


Space:

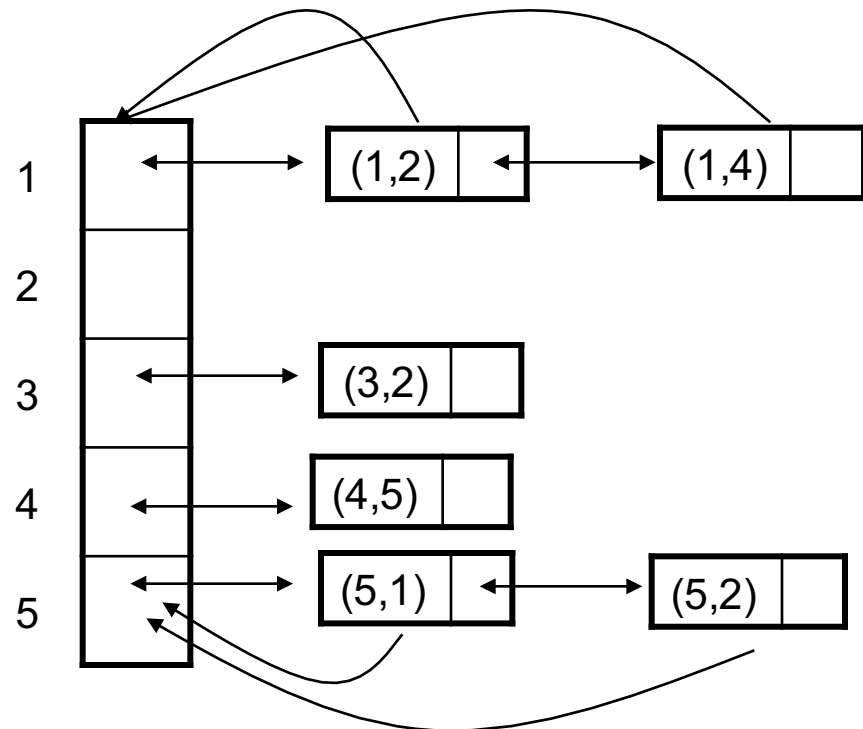
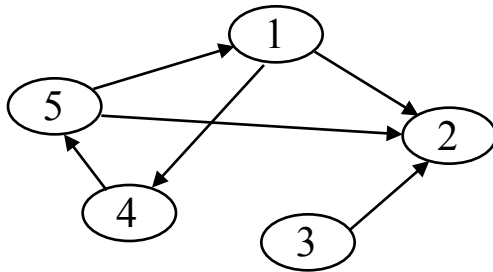
$$n + m$$

Edge List Structure

- Vertex object
 - element
 - reference to position in vertex sequence
- Edge object
 - element
 - origin vertex object
 - destination vertex object
 - reference to position in edge sequence
- Vertex sequence
 - sequence of vertex objects
- Edge sequence
 - sequence of edge objects

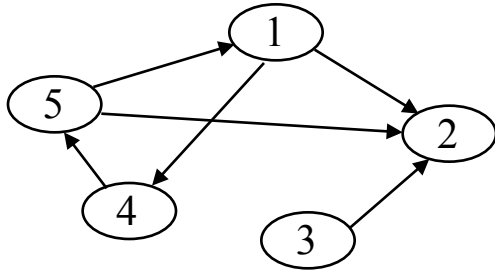


Adjacency List (example)

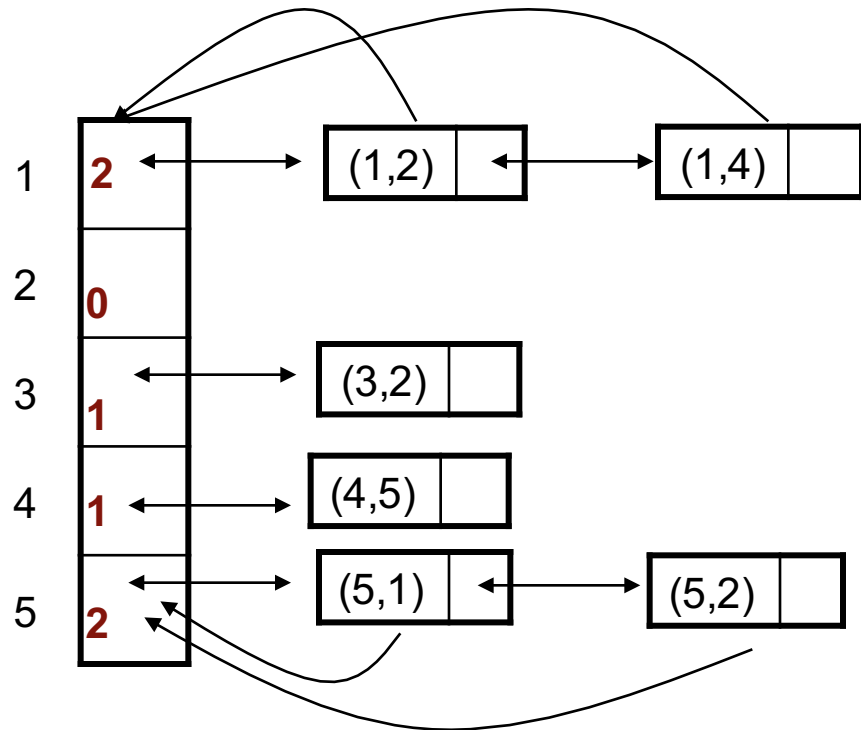


Edges augmented by link to node

Adjacency List (example)



Often, the node out-degree is kept at the node.

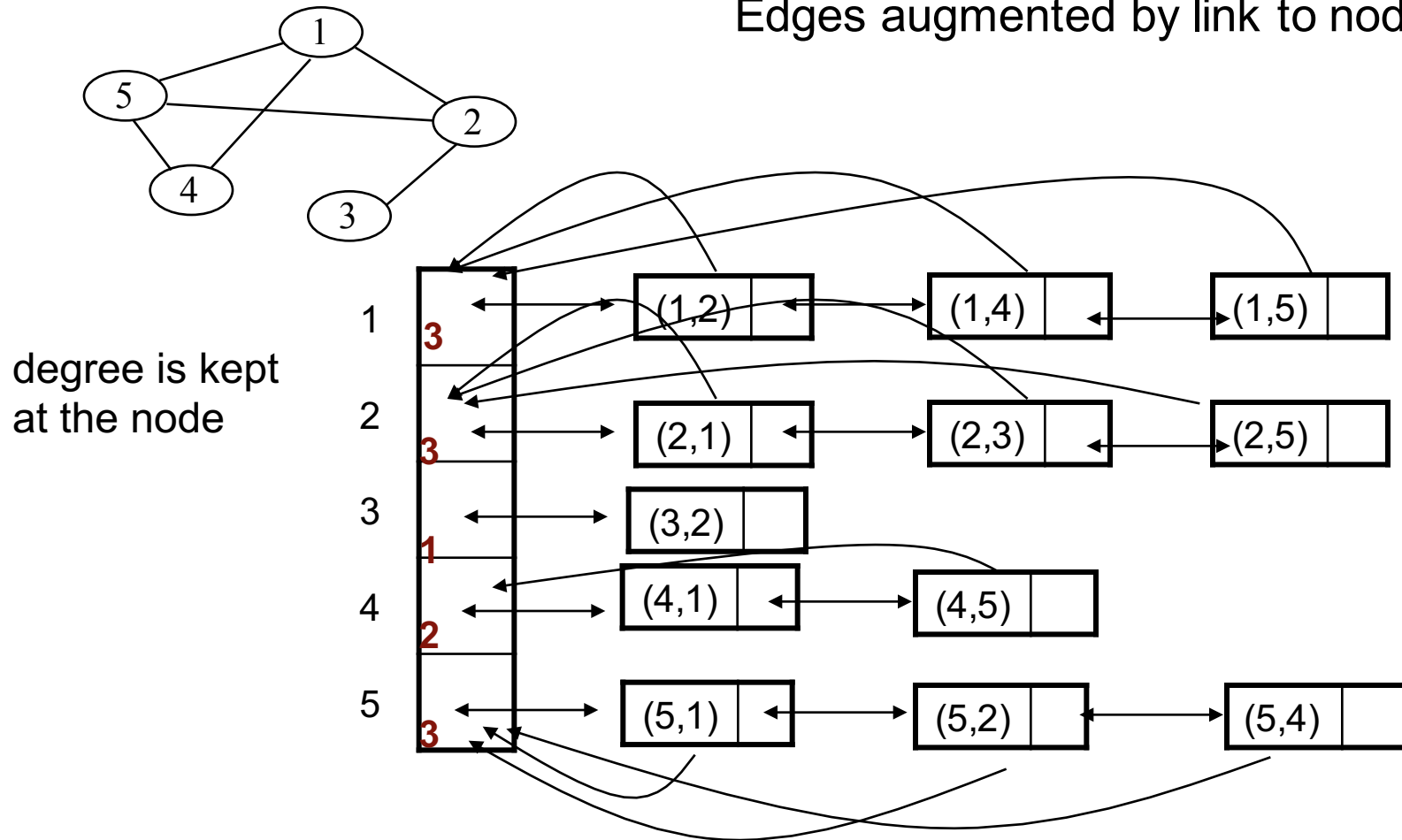


Edges augmented by link to node

Adjacency List

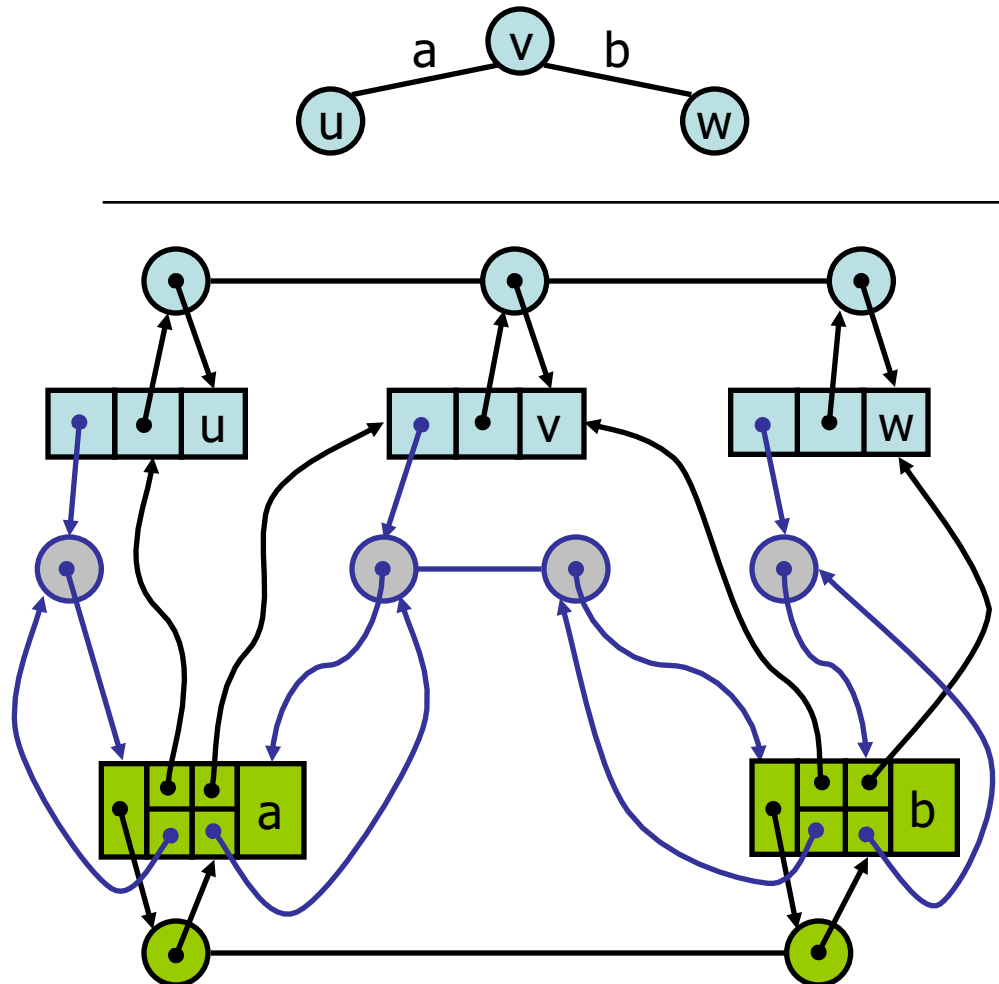
(another example -with undirected edges)

Edges augmented by link to node

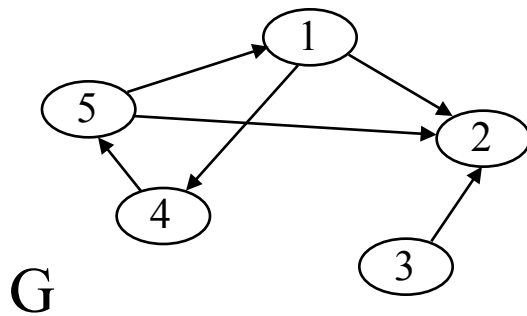


Adjacency List Structure

- Edge list structure
- Incidence sequence for each vertex
 - sequence of references to edge objects of incident edges
- Augmented edge objects
 - references to associated positions in incidence sequences of end vertices

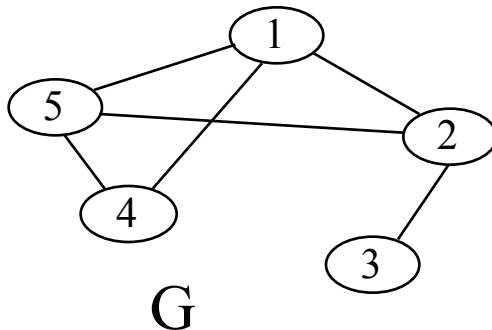


Adjacency Matrix (examples)



	1	2	3	4	5
1	0	1	0	1	0
2	0	0	0	0	0
3	0	1	0	0	0
4	0	0	0	0	1
5	1	1	0	0	0

If G is not-directed



	1	2	3	4	5
1	0	1	0	1	1
2	1	0	1	0	1
3	0	1	0	0	0
4	1	0	0	0	1
5	1	1	0	1	0

symmetric matrix

Adjacency Matrix (observation)

Space:

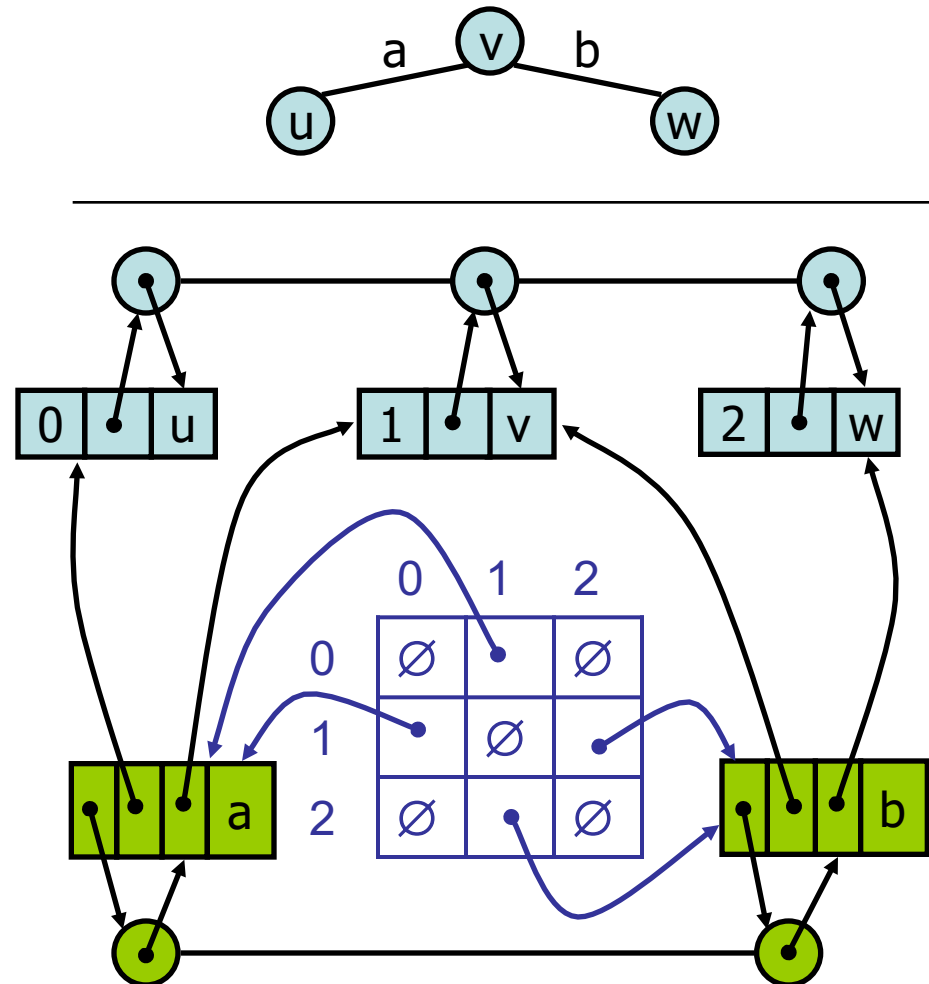
$n \times n$

Lots of waste space if the matrix is SPARSE ...

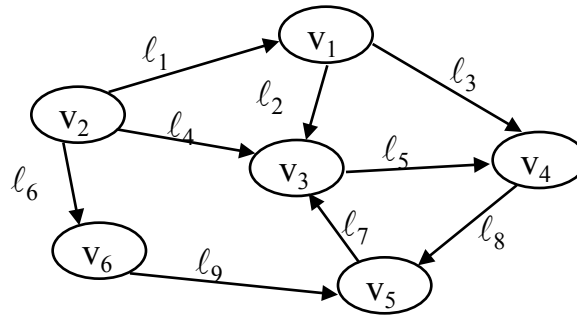
1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	0	1	0
0	0	0	0	1	0	0	0	0	0
1	0	1	1	0	1	0	0	0	0
0	0	0	0	1	0	0	1	0	0
0	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0	0	0
0	1	1	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0

Adjacency Matrix Structure

- Edge list structure
- Augmented vertex objects
 - Integer key (index) associated with vertex
- 2D-array adjacency array
 - Reference to edge object for adjacent vertices
 - Null for non adjacent vertices



Incidence Matrix (1)



	ℓ_1	ℓ_2	ℓ_3	ℓ_4	ℓ_5	ℓ_6	ℓ_7	ℓ_8	ℓ_9
v_1	-1	1	1	0	0	0	0	0	0
v_2	1	0	0	1	0	1	0	0	0
v_3	0	-1	0	-1	1	0	-1	0	0
v_4	0	0	-1	0	-1	0	0	1	0
v_5	0	0	0	0	0	0	1	-1	-1
v_6	0	0	0	0	0	-1	0	0	1

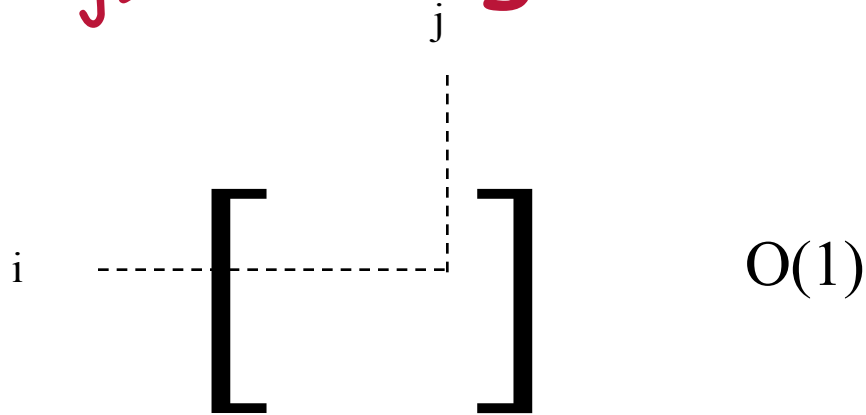
Space:

$n \times m$

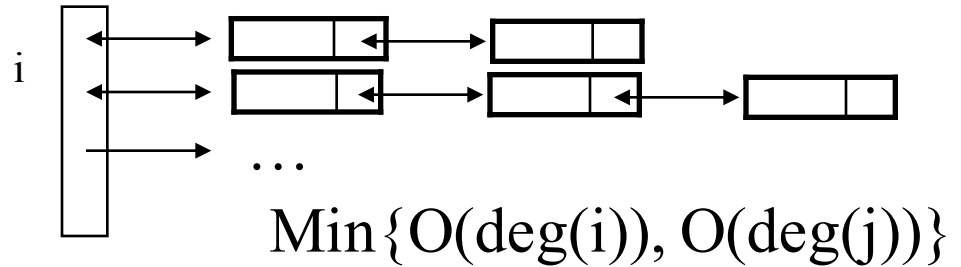
Example in non-directed graphs

Is (v_i, v_j) an edge?

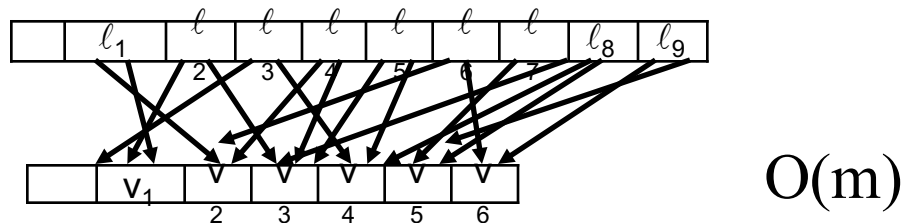
Adjacency
Matrix:



Adjacency
List:

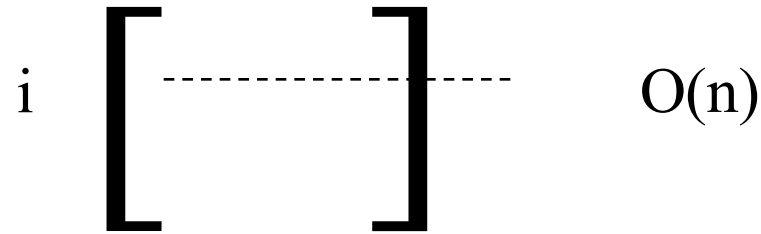


Edge
List:

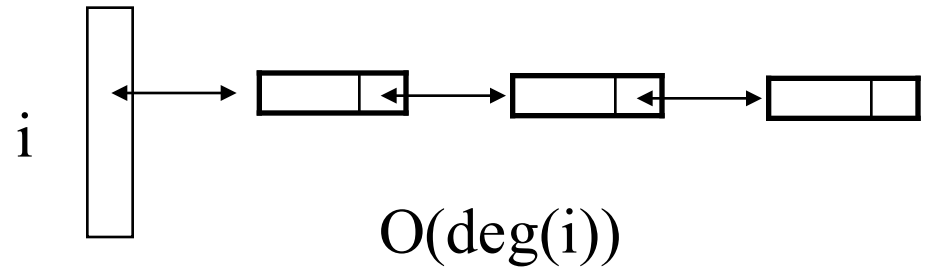


Which nodes are adjacent to v_i ?

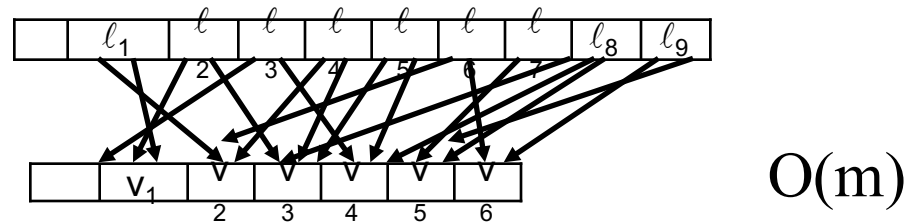
Adjacency
Matrix:



Adjacency
List:

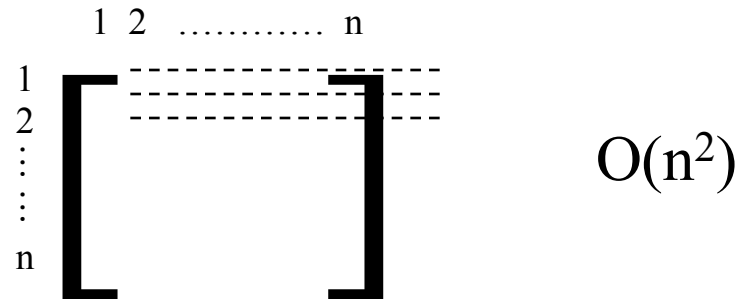


Edge
List:

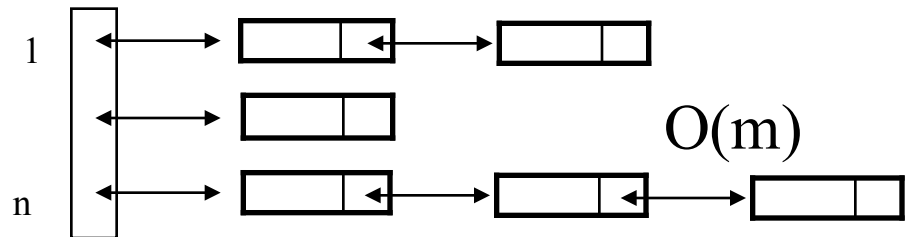


Mark all Edges

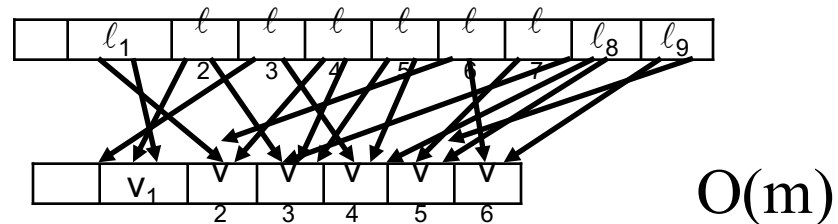
Adjacency
Matrix:



Adjacency
List:

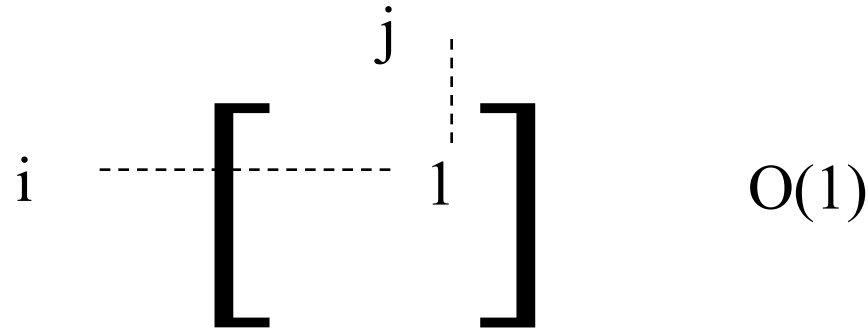


Edge
List:

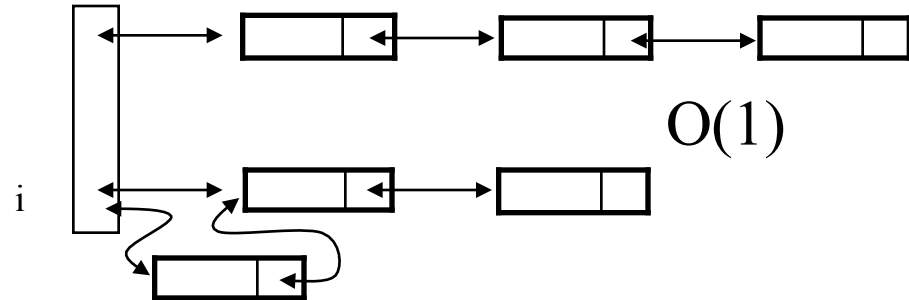


Add an Edge (v_i, v_j)

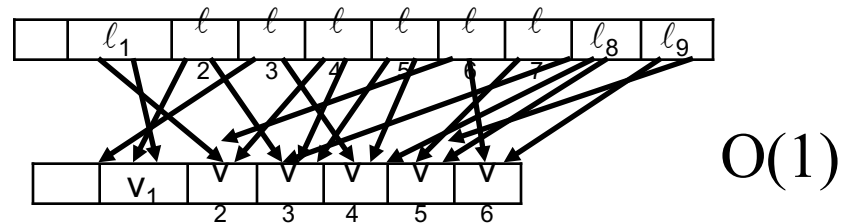
Adjacency
Matrix:



Adjacency
List (linked):

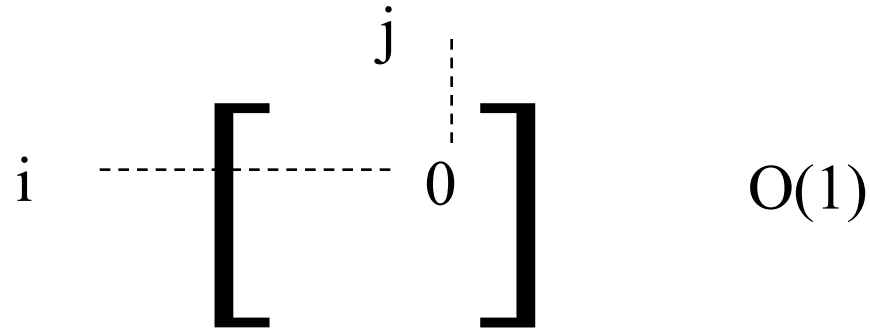


Edge
List:

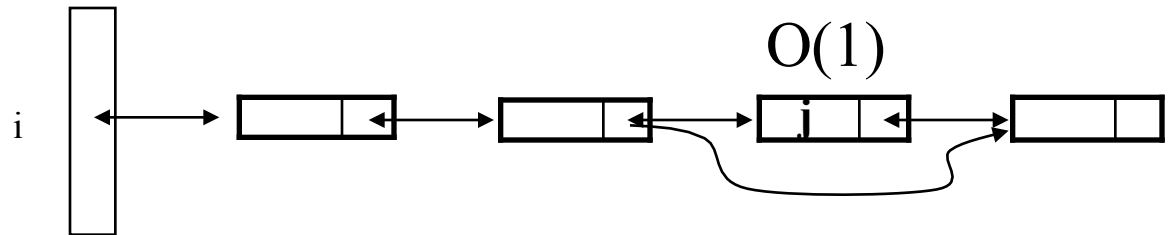


Remove a given Edge $e=(v_i, v_j)$

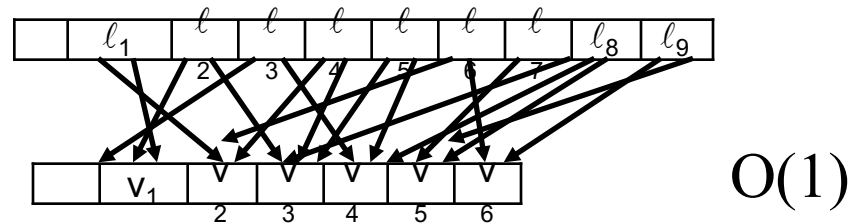
Adjacency
Matrix:



Adjacency
List :



Edge
List:



Adjacency
Matrix

Adjacency
List

Is (v_i, v_j) an
edge?

$O(1)$

$O(\deg(i))$

Which nodes
are adjacent to
 v_i ?

$O(n)$

$O(\deg(i))$

Mark all edges

$O(n^2)$

$O(m)$

Add edge
 (v_i, v_j)

$O(1)$

$O(1)$

Remove edge
 (v_i, v_j)

$O(1)$

$O(1)$

$O(\deg(i))$ = OUT-degree of node v_i

G is directed

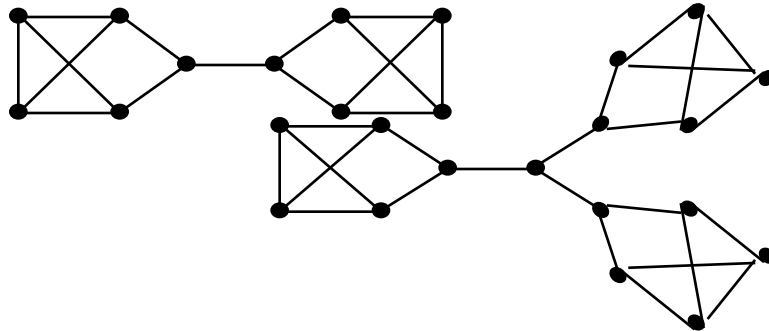
What are the
predecessors of v_i ?

Performance

<ul style="list-style-type: none"> • n vertices • m edges • no parallel edges • no self-loops 	Edge List	Adjacency List	Adjacency Matrix
Space	$n + m$	$n + m$	n^2
incidentEdges(v)	m	deg(v)	n
areAdjacent (v, w)	m	min(deg(v), deg(w))	1
insertVertex(o)	1	1	n^2
insertEdge(v, w, o)	1	1	1
removeVertex(v)	m	deg(v)	n^2
removeEdge(e)	1	1	1

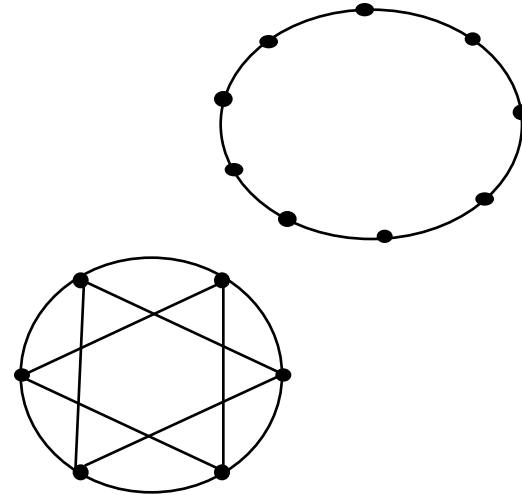
Special Graphs

Regular Graphs

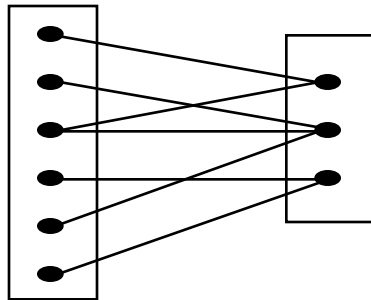


$$\forall v_i, v_j \in V$$

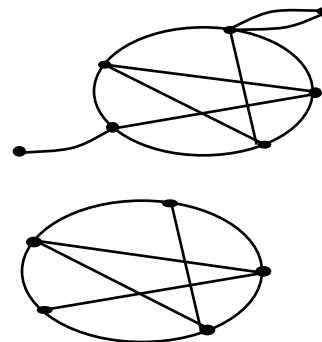
$$d(v_i) = d(v_j)$$



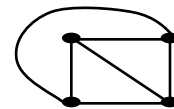
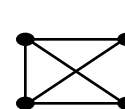
Bipartite Graphs



Planar Graphs



Cannot have



$$n - 1 \leq m \leq \frac{n(n-1)}{2}$$

$$1 \leq \deg(i) \leq n - 1$$

connected,
non-directed



degree

$$n = |V| \quad m = |E|$$

$$n - 1 \leq m \leq n(n - 1)$$

$$1 \leq \deg(i) \leq n - 1$$

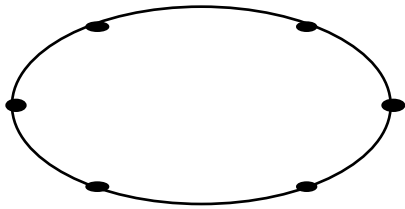
connected,
directed



OUT-degree

Some Regular Graphs

— Ring —

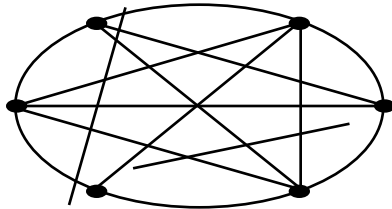


$$m = n$$
$$d_i = 2 \quad \forall i$$

Tree

$$m = O(n)$$

— Complete Graph —

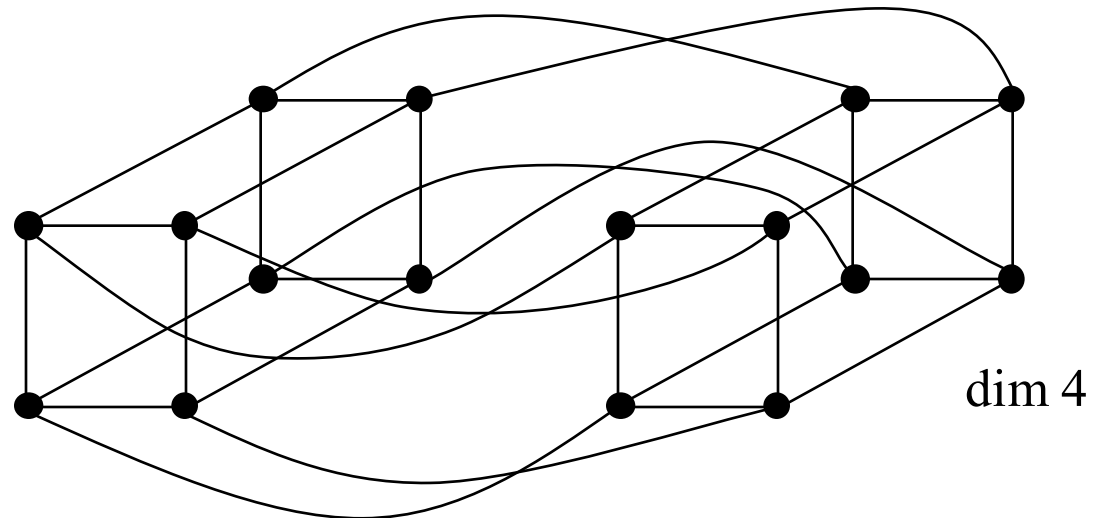
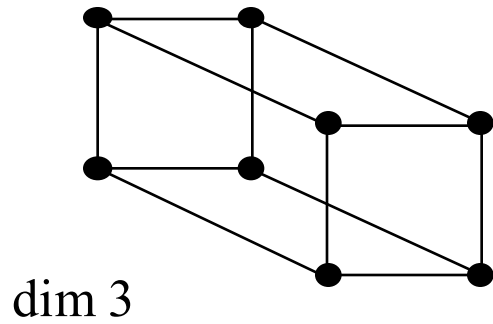
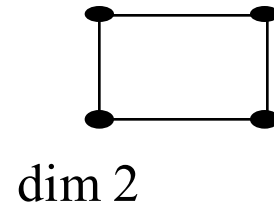


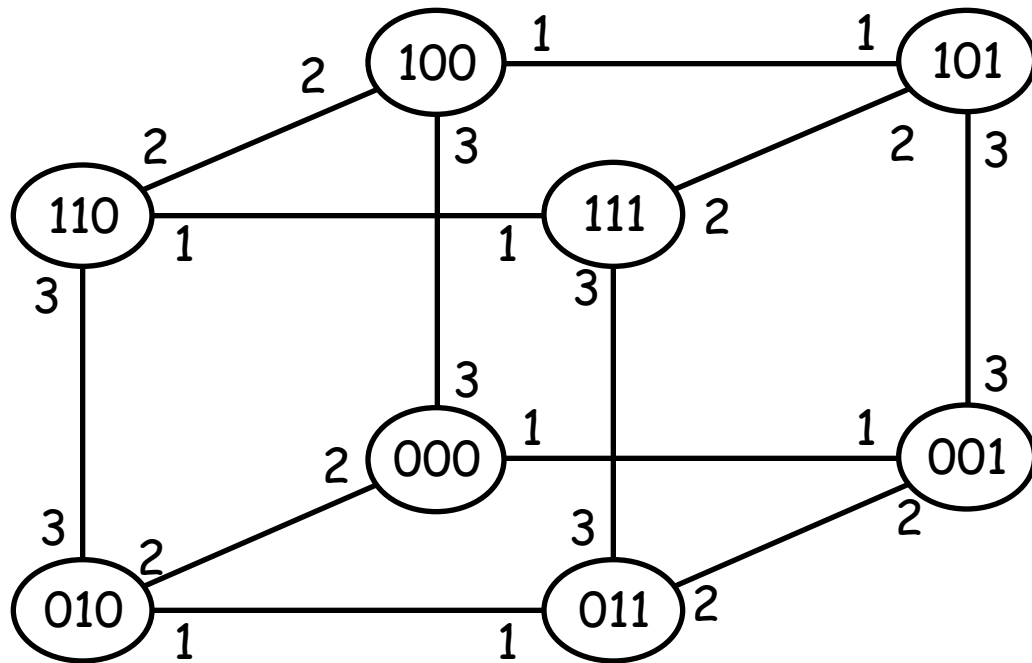
$$m = \frac{n(n-1)}{2}$$

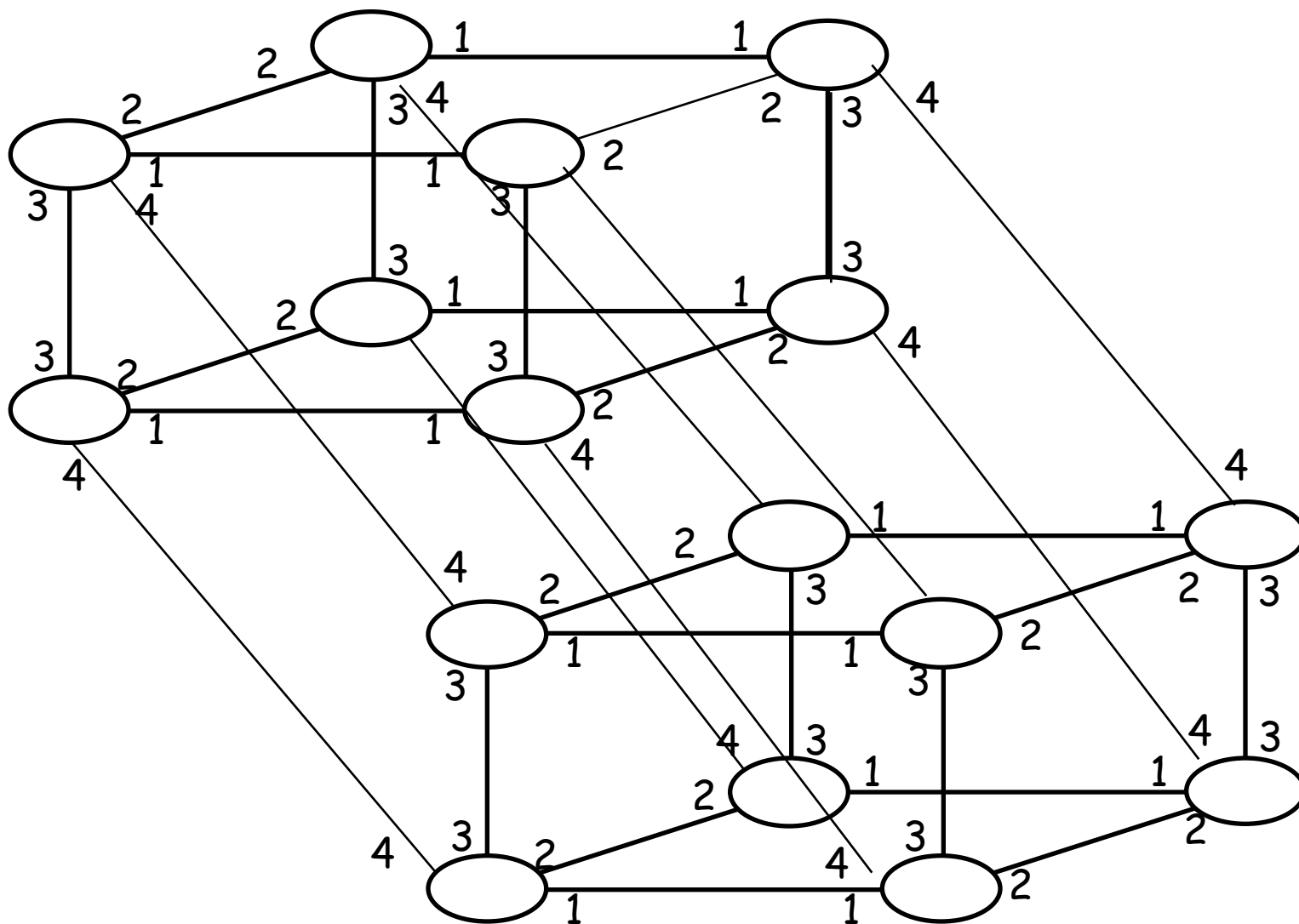
$$d_i = n - 1 \quad \forall i$$

$$m = O(n^2)$$

— Hypercube —







Hypercube

	n	m	
h_0	1	0	
h_1	2	1	
h_2	4	$1 \times 2 + 2 = 4$	
h_3	8	$4 \times 2 + 4 = 12$	
h_4	16	$12 \times 2 + 8 = 32$	$m_i = i \cdot 2^{i-1}$

$h_i:$

$$n_0 = 1$$

$$n_i = 2 n_{i-1} \longrightarrow n_i = 2^i$$

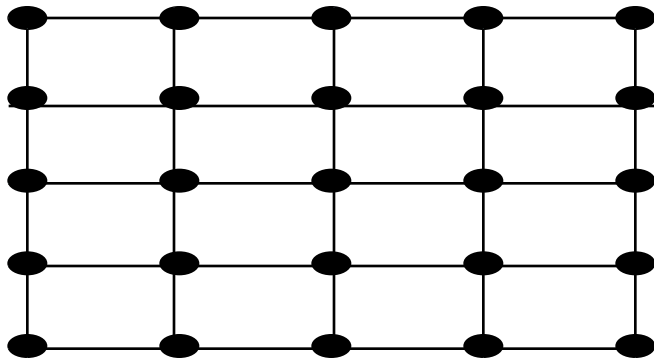
$$\left\{ \begin{array}{l} n_0 = 1 \\ n_i = 2 n_{i-1} \end{array} \right. \longrightarrow n_i = 2^i$$

$$m_i = i \cdot 2^{i-1}$$

$$\begin{array}{l} m = \frac{n \log n}{2} \\ \text{degree} = \log n \end{array}$$

$$m = O(n \log n)$$

— Grid —



Not regular

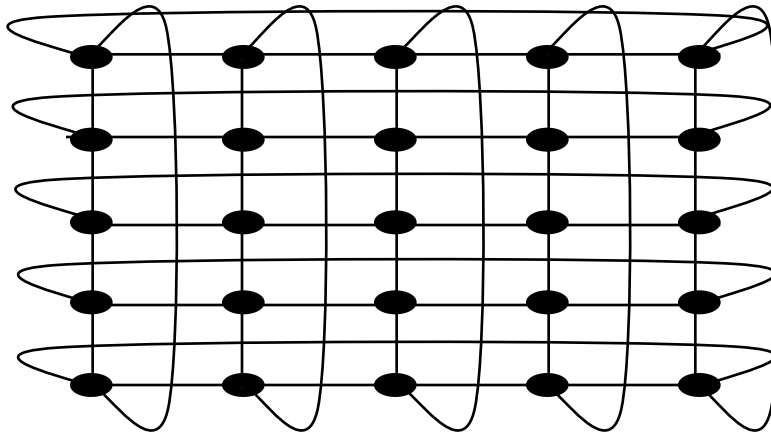
$$m = O(n)$$

$\deg(i) = 4$ $v_i = \text{internal}$

$\deg(i) = 3$ $v_i = \text{border}$

$\deg(i) = 2$ $v_i = \text{corner}$

— Torus —



$$m = O(n)$$

$$\deg(i) = 4 \quad \forall i$$