CSE221

Lecture 18:
Graphs

Hyungon Moon

Acknowledgment: The content of this file is based on the slides of
the textbook as well as the slides provided by Prof. Won-Ki Jeong.

ULSAN NATIONAL INSTITUTE OF
SCIENCE AND TECHNOLOGY

Outline

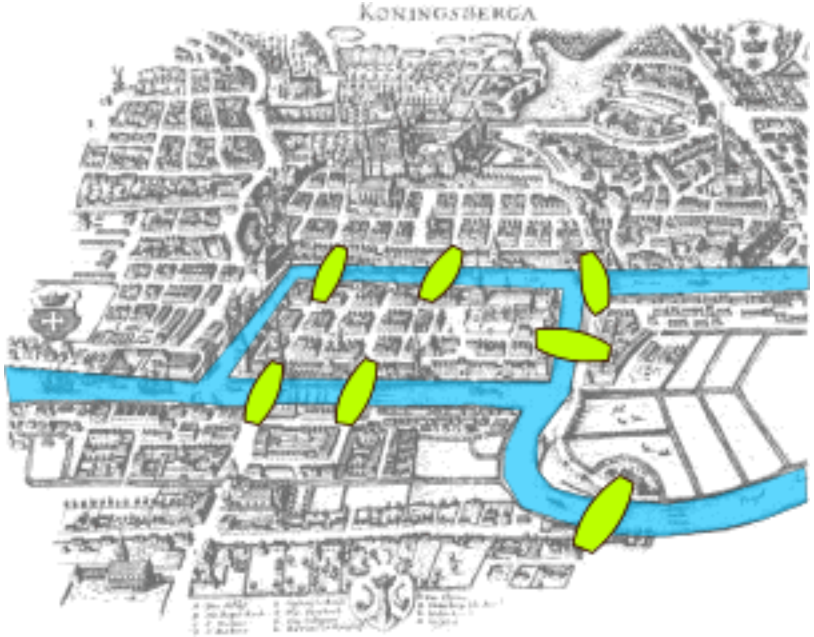
- Graph definitions
- Graph representations
  - Adjacency matrix
  - Adjacency list
  - Adjacency multilist

Outline

- Graph definitions
- Graph representations
  - Adjacency matrix
  - Adjacency list
  - Adjacency multilist

The Seven Bridges of Königsberg

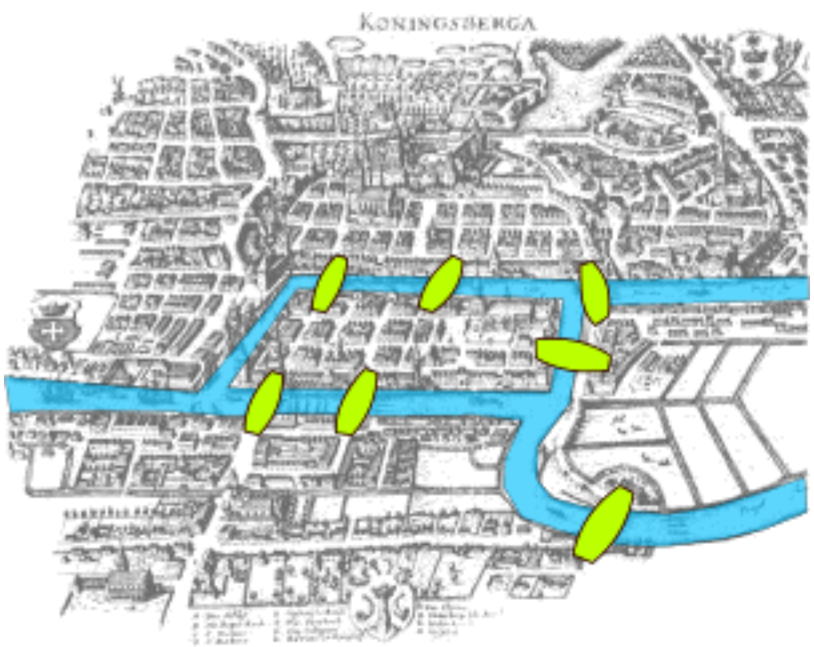
- Is it possible to start from and return to one land after crossing all the bridges only once?
  - Eulerian walk



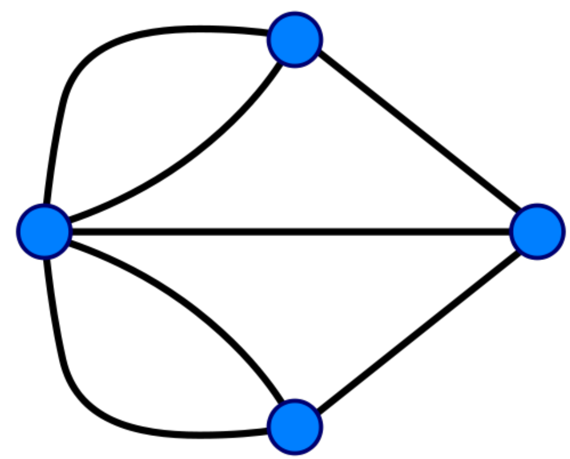
Wikipedia

The Seven Bridges of Königsberg

- Graph problem
  - Degree of a node : # of edges incident to it
  - If a node is not start/end, it must have even edges (one enter, one exit)
  - There must be <u>zero or two odd degree nodes</u>



Wikipedia



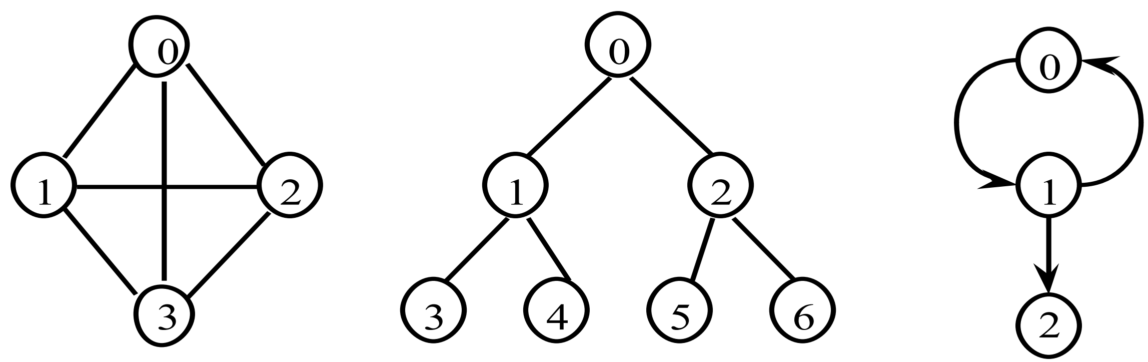
4 odd degree nodes : no path!

Definitions

- Graph G consists of two sets, V and E
  - G=(V,E)
  - V : finite, nonempty set of vertices
  - E : set of pairs of vertices (edges)
- Undirected graph
  - Pair of vertices representing any edge is unordered
  - (u, v) = (v, u)

Definitions

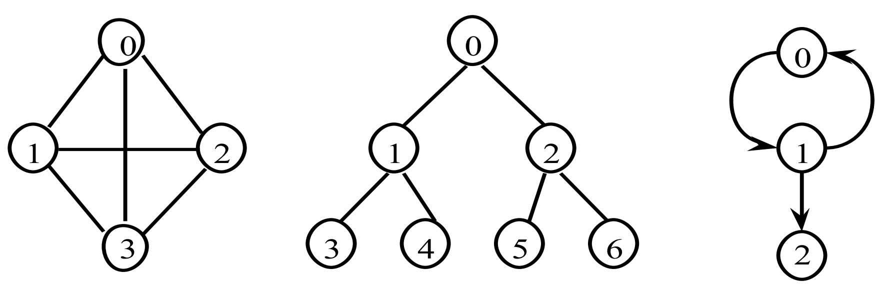
- Directed graph
  - Each edge represents directed pair <u, v>
  - u : tail, v : head
  - <u, v> != <v, u>



(a) G_1 (b) G_2 (c) G_3

Definitions

- # unordered edges in a graph with n vertices
  - $(n-1) + (n-2) + .. + 2 + 1 = {}_nC_2 = n(n-1)/2$
- Complete graph
  - Graph that has maximum number of edges
  - For n vertices, $n(n-1)/2$ edges
  - G_1 is complete



(a) G_1 (b) G_2 (c) G_3

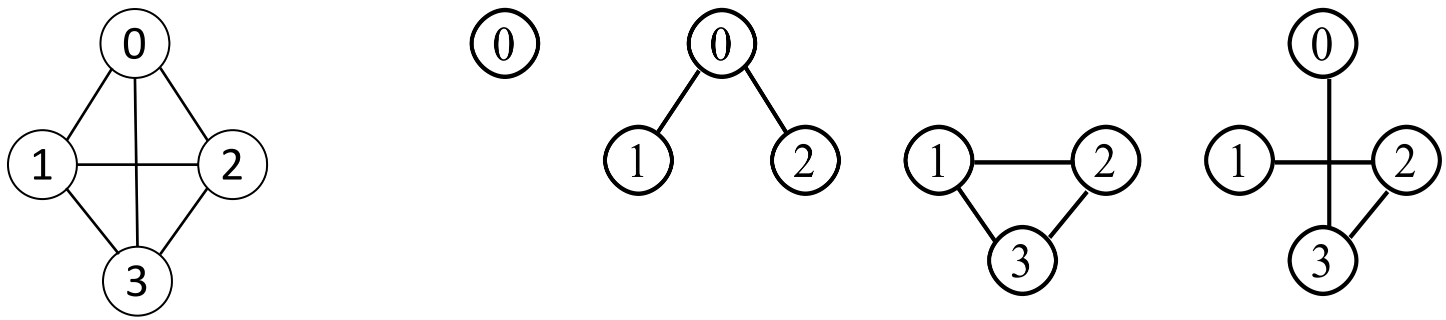
Definitions

- If (u,v) is an edge
  - u and v are *adjacent*
  - Edge (u,v) are *incident* on vertices u and v

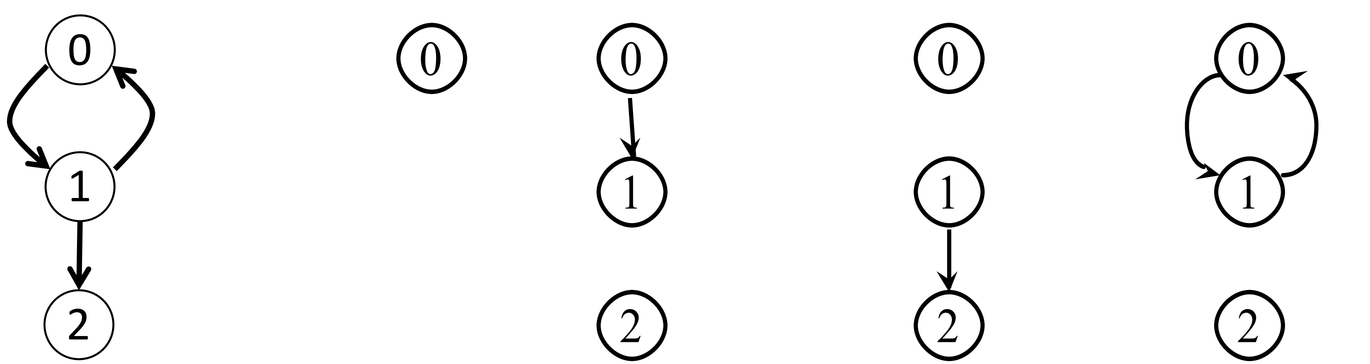
- Subgraph of G
  - G'=(V',E'), $V' \subseteq V$ and $E' \subseteq E$

Definitions

Subgraphs



(i) (ii) (iii) (iv)



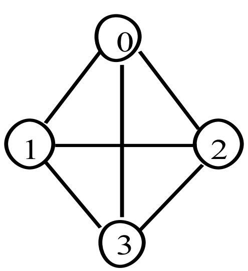
(i) (ii) (iii) (iv)

Definitions

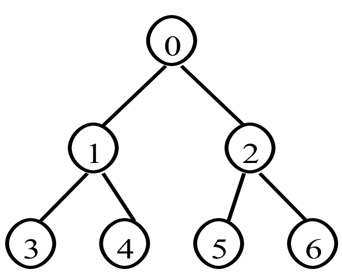
- Path
  - From u to v in G: sequence of vertices, u, i1, i2, ..., ik, v such that (u,i1), (i1,i2), ..., (ik,v) are edges in E(G)
- Length
  - Number of <u>edges</u> in the path
- Simple path
  - Path that as all distinct vertices except first and last vertices (i.e., first and last vertices can be same)

Definitions

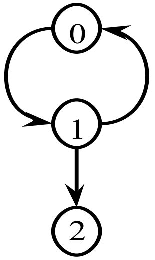
- Examples
  - Path 0,1,3,2,4 : length 4, simple
  - Path 0,1,3,1,4 : length 4, not simple
- Cycle
  - Simple path that has the same first and last vertices



(a) G_1



(b) G_2



(c) G_3

any cycle?

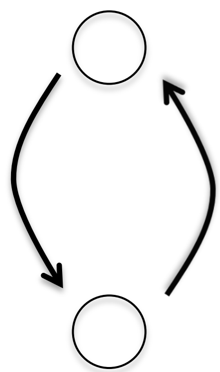
ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

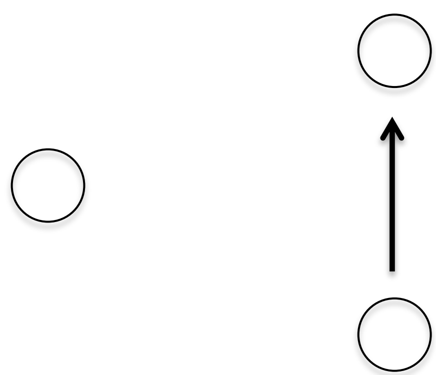
Definitions

- Vertices u and v are *Connected* iff
  - There is a path from u to v
- Undirected graph G is connected iff
  - There is a path from u to v for every pair u, v in G
- Connected component H of undirected graph G
  - Maximally connected subgraph of G
  - i.e., there is no subgraph of G that is connected and properly contains H

Definitions

- Tree
  - An acyclic (no cycle) connected graph
- Directed graph is strongly connected iff
  - For every pair u and v there is a directed path from u to v and v to u





Definitions

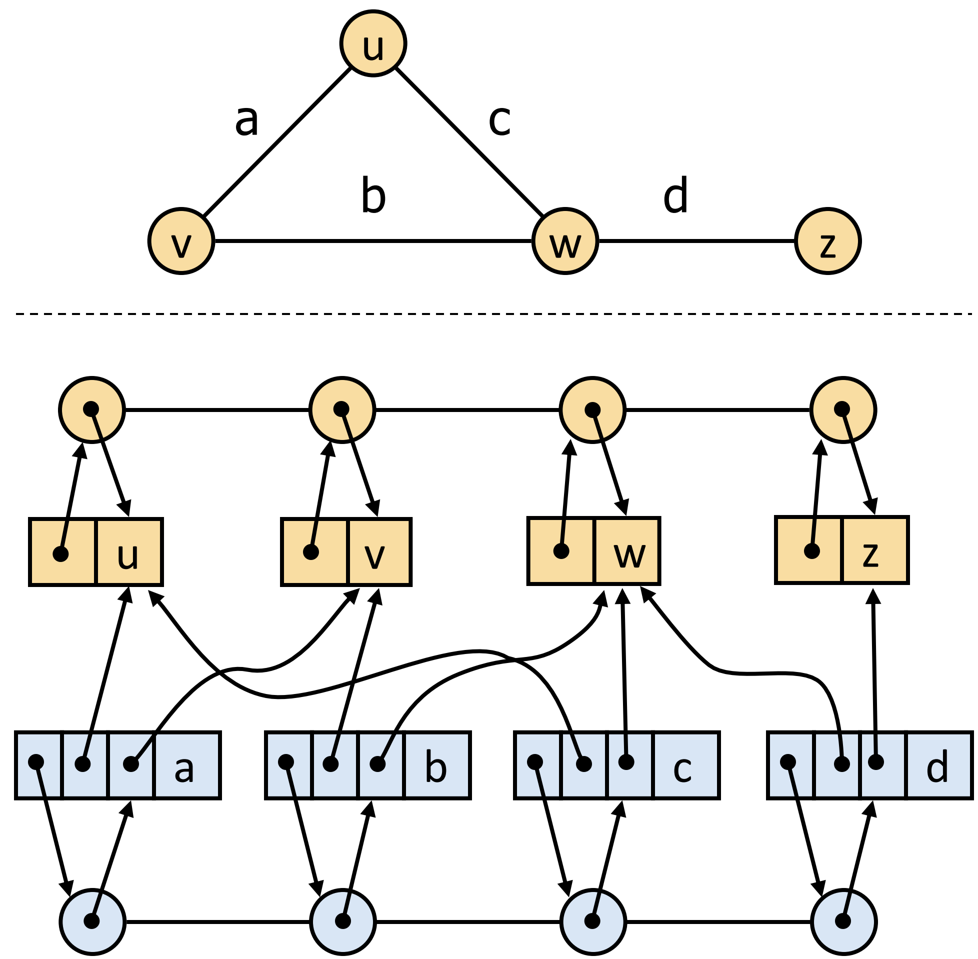
- Degree of a vertex
  - Number of edges incident to that vertex
- In-degree of a vertex in a directed graph
  - Number of edges for which v is the head
- Out-degree of a vertex in a directed graph
  - Number of edges for which v is the tail
- Digraph
  - Directed graph

Outline

- Graph definitions
- Graph representations
  - Adjacency matrix
  - Adjacency list
  - Adjacency multilist

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

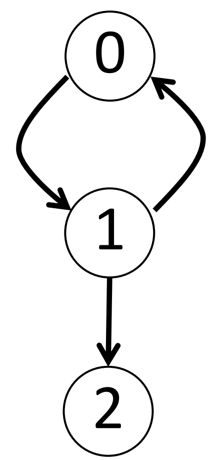


Adjacency Matrix

- n x n matrix a for a graph G having n vertices
  - a[i][j] = 1 if there is an edge (i,j) (or <i,j>)
  - a[i][j] = 0 otherwise

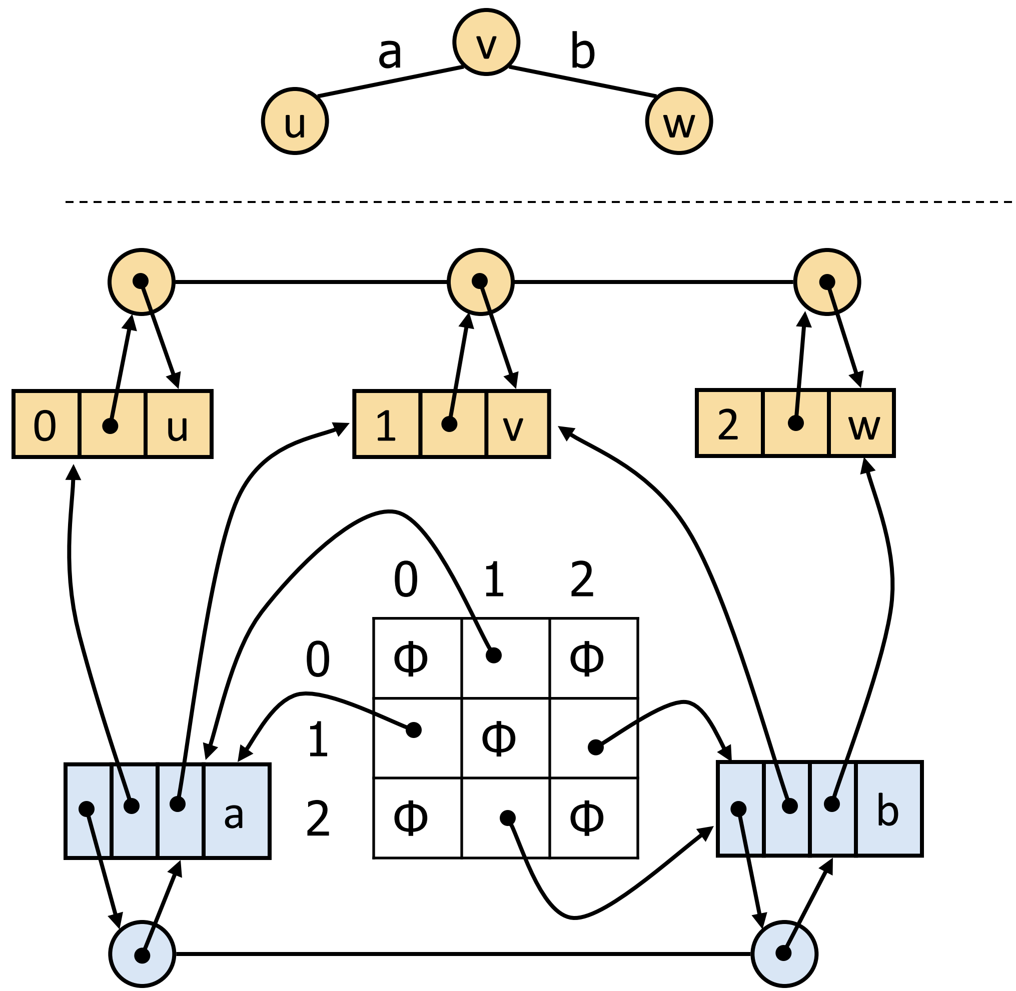
$$\begin{array}{cc} & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \end{array} & \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

<0,1>
<1,0>
<1,2>



Adjacency Matrix (Pointer Version)

- 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 nonadjacent vertices



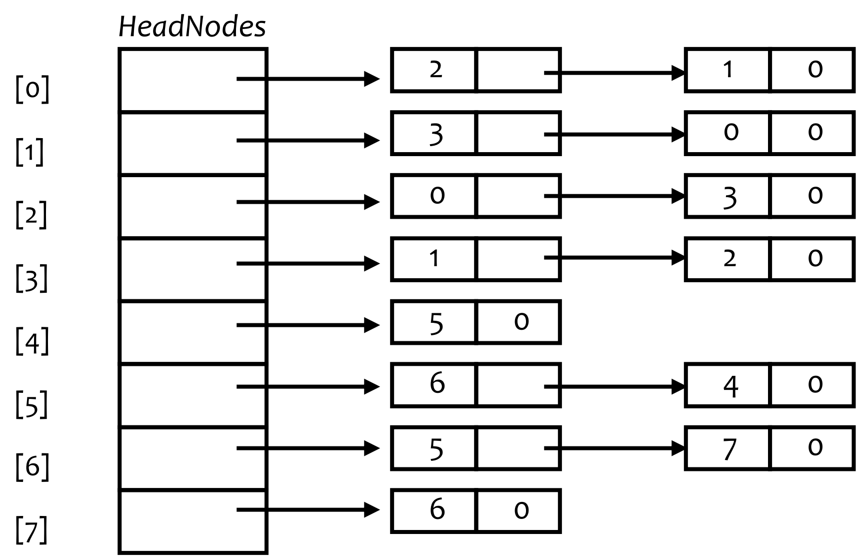
Adjacency Matrix

- Properties
  - Matrix for undirected graph is symmetric
  - Diagonal entries are zero
  - Digraph : row is tail, column is head
    - Sum of row i : out-degree of i
    - Sum of column j : in-degree of j
- Problems
  - Require n^2 bits
  - $O(n^2)$ runtime for algorithms
    - ex) count the number of edges in a graph

Adjacency List

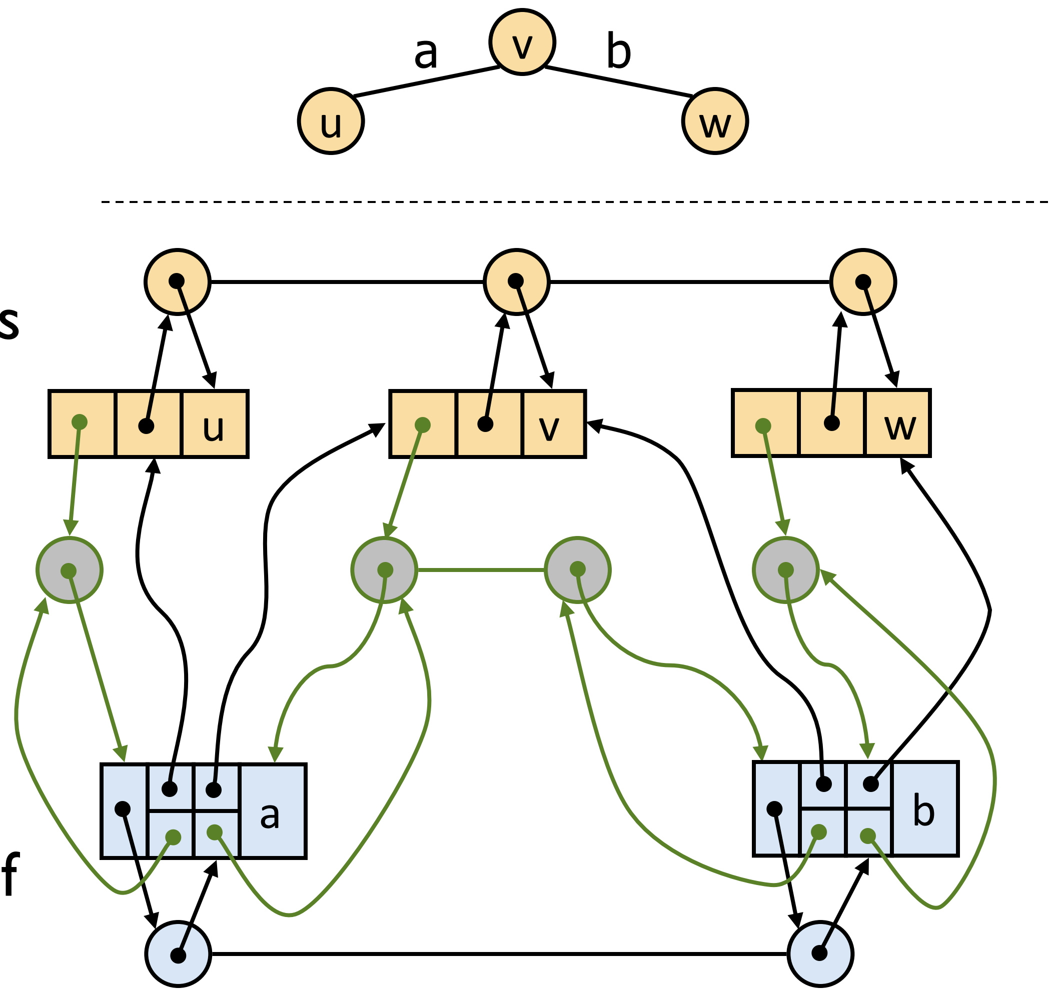
- *n* rows are represented as *n* chains
- Vertices in each row does not need to be ordered

$$
\begin{array}{c}
0\\1\\2\\3\\4\\5\\6\\7
\end{array}
\begin{pmatrix}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0\\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1\\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
$$



Adjacency List (Pointer Version)

- 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

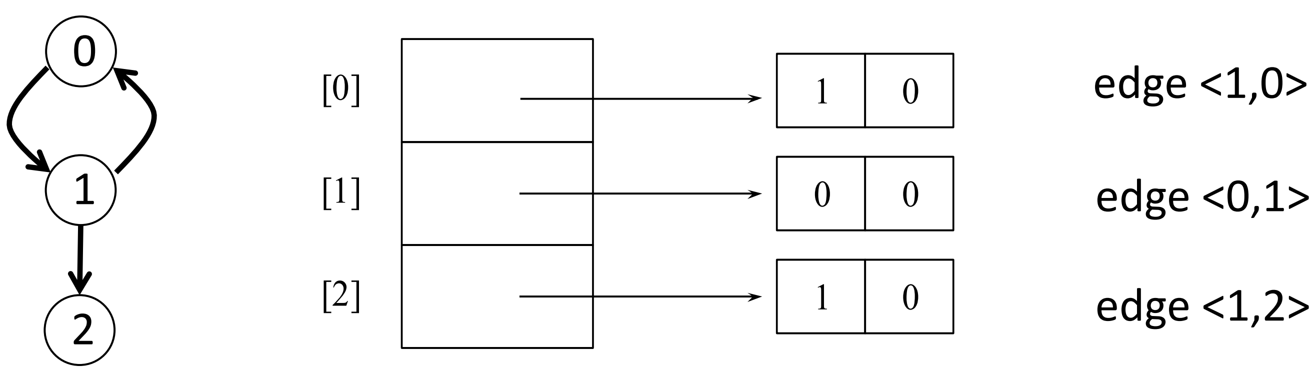


Performance

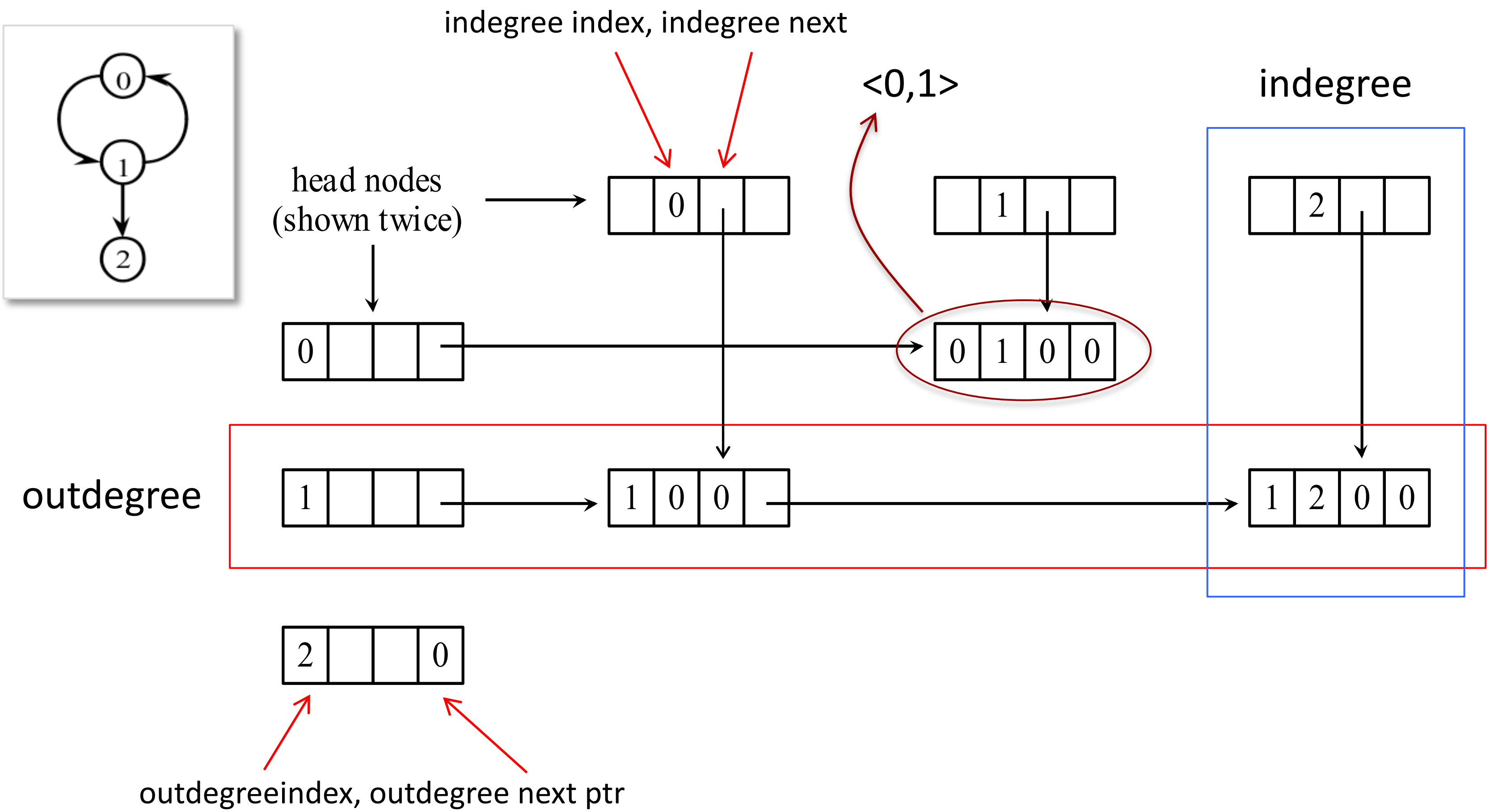
▪ n vertices, m edges ▪ no parallel edges ▪ no self-loops	Edge List	Adjacency List	Adjacency Matrix
Space	$n + m$	$n + m$	n^2
v.incidentEdges()	m	$\deg(v)$	n
u.isAdjacentTo (v)	m	$\min(\deg(v), \deg(w))$	1
insertVertex(o)	1	1	n^2
insertEdge(v, w, o)	1	1	1
eraseVertex(v)	m	$\deg(v)$	n^2
eraseEdge(e)	1	1	1

Inverse Adjacency List

- Inverse adjacency list
  - Adjacency list can be used for calculating <u>out-degree</u> of a vertex
  - In-degree vertex is not easy
  - A row of inverse adjacency list stores incoming vertex list



Orthogonal List Representation



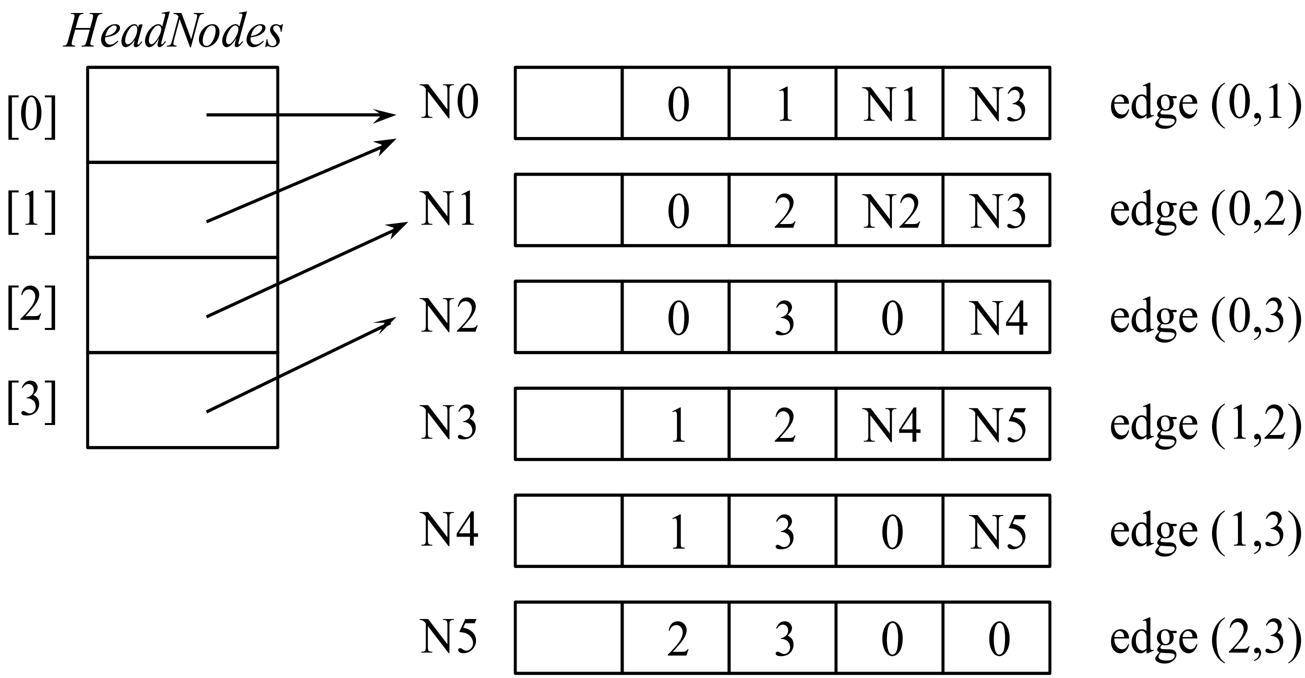
Adjacency Multilist

- Each edge in an undirected graph shows twice in adjacency list – waste of memory!
- Edge node can be in two lists

link for the edge containing vertex 1

m	vertex 1	vertex 2	list 1	list 2

link for the edge containing vertex 2



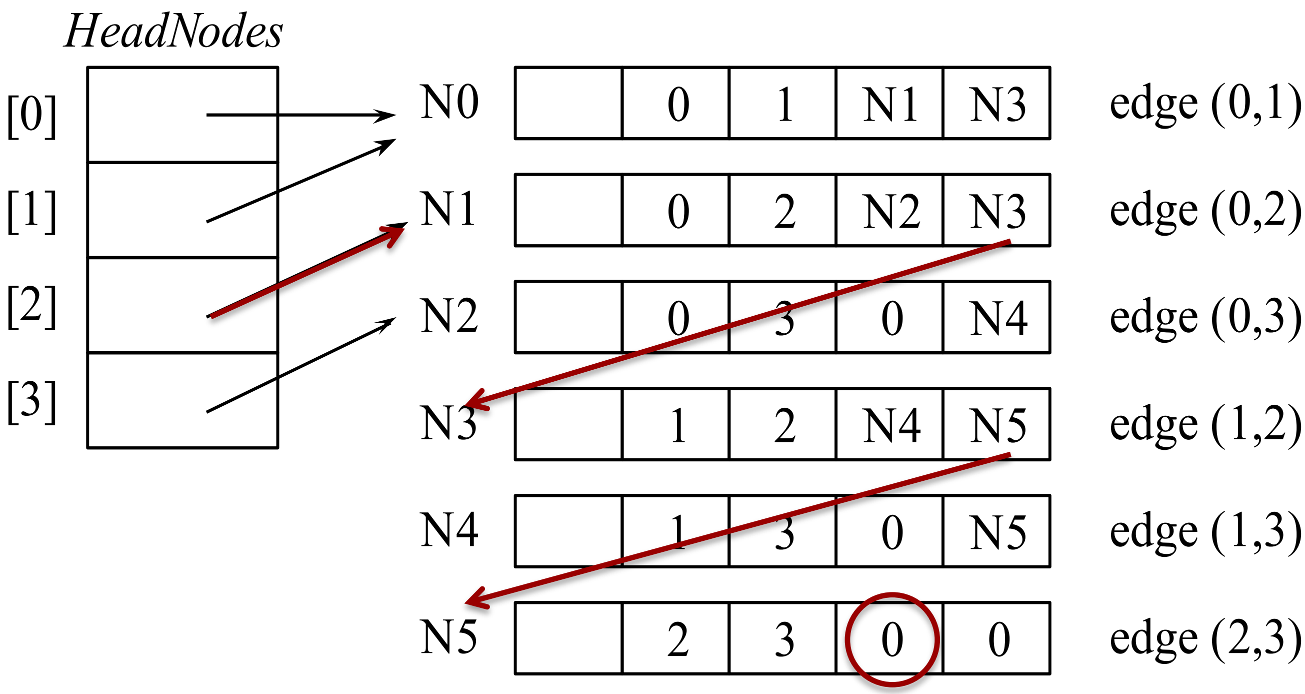
The lists are

vertex 0: N0 -> N1 -> N2

vertex 1: N0 -> N3 -> N4

vertex 2: N1 -> N3 -> N5

vertex 3: N2 -> N4 -> N5



The lists are

vertex 0: N0 -> N1 -> N2

vertex 1: N0 -> N3 -> N4

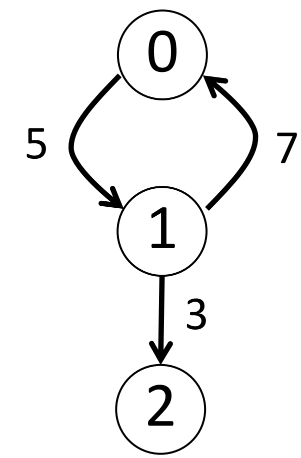
vertex 2: N1 -> N3 -> N5

vertex 3: N2 -> N4 -> N5

Weighted Edges

- Edges of a graph may have weights assigned
- Adjacency matrix
  - Each entry is weight
- Adjacency list
  - Extra field required

$$
\begin{array}{c c}
 & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \end{array} & \begin{bmatrix} 0 & 5 & 0 \\ 7 & 0 & 3 \\ 0 & 0 & 0 \end{bmatrix}
\end{array}
$$



- Network
  - A graph with weighted edges

Questions?