SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-IOT Including CS&BCT

COURSE NAME : **19CS307**- DATA STRUCTURES
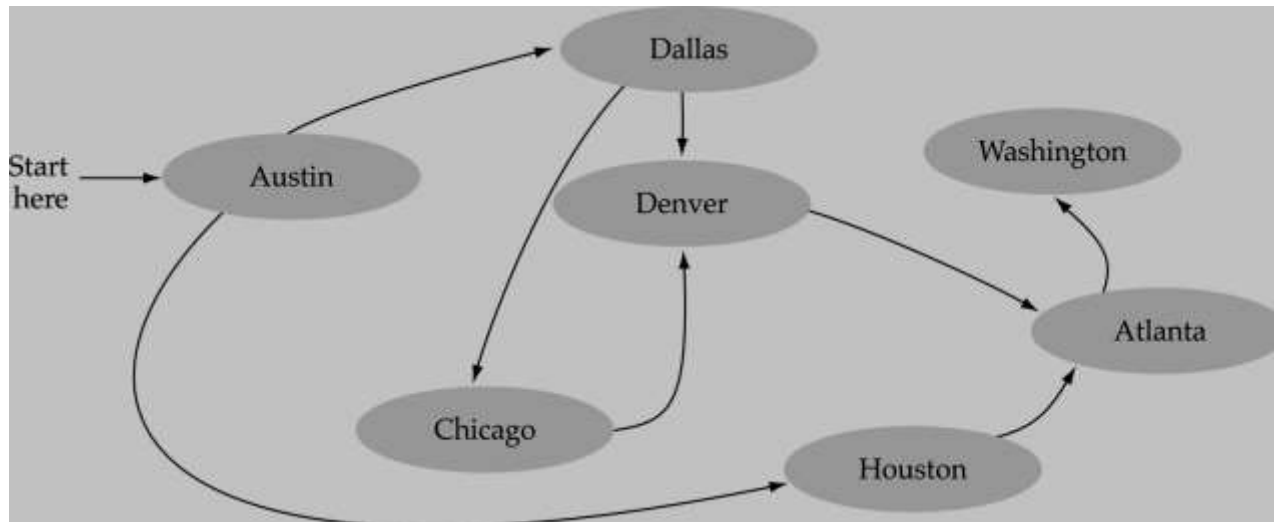
II YEAR / III SEMESTER

Unit IV- **NON LINEAR DATA STRUCTURES - GRAPH**

TOPIC: Graphs

What is a graph?

- A data structure that consists of a set of nodes (*vertices*) and a set of edges that relate the nodes to each other

- The set of edges describes relationships among the vertices

Formal definition of graphs

- A graph *G* is defined as follows:

$$G=(V,E)$$

V(G): a finite, nonempty set of vertices

E(G): a set of edges (pairs of vertices)

Graphs Cont..

- Graph and its representations

- Graph is a data structure that consists of following two components:
  1. A finite set of vertices also called as nodes.
  2. A finite set of ordered pair of the form (u, v) called as edge.

- The pair is ordered because (u, v) is not same as (v, u) in case of a directed graph(di-graph).

- The pair of the form (u, v) indicates that there is an edge from vertex u to vertex v. The edges may contain weight/value/cost.
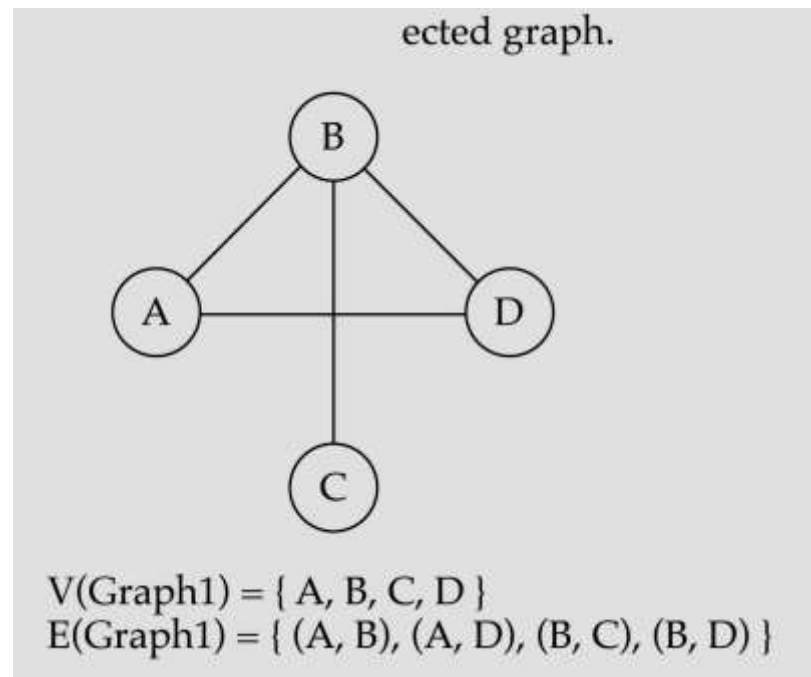
Cont..

- Graphs are used to represent many real-life applications

- Graphs are used to represent networks.

- The networks may include paths in a city or telephone network or circuit network

- Graphs are also used in social networks like linkedIn, Facebook.

- For example, in Facebook, each person is represented with a vertex(or node).

- Each node is a structure and contains information like person id, name, gender and locale
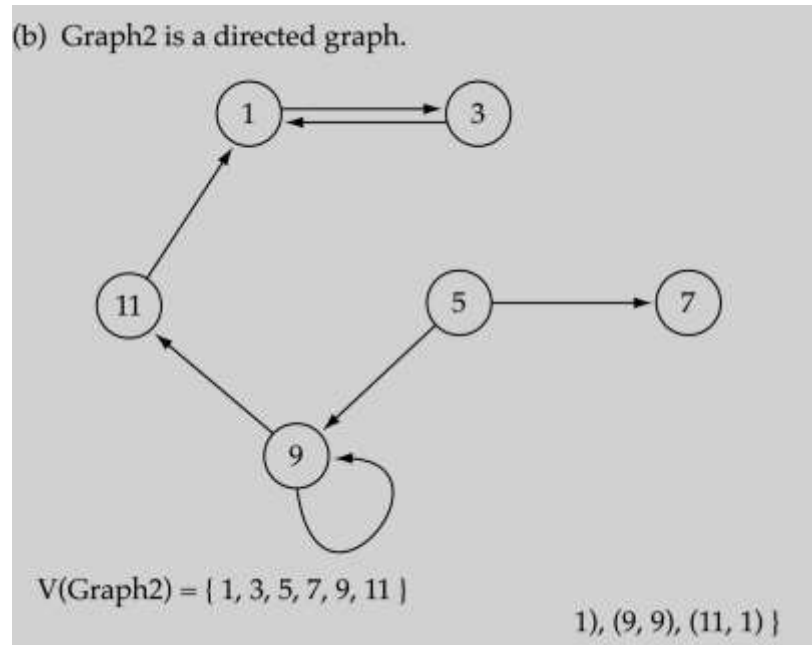
Directed vs. undirected graphs

- When the edges in a graph have no direction, the graph is called *undirected*

ected graph.

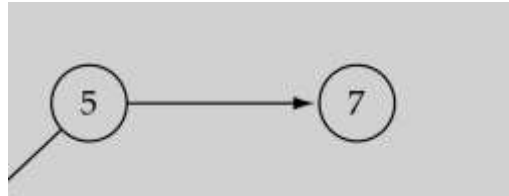V(Graph1) = { A, B, C, D }
E(Graph1) = { (A, B), (A, D), (B, C), (B, D) }

Directed vs. undirected graphs (cont.)

- When the edges in a graph have a direction, the graph is called *directed* (or *digraph*)

(b) Graph2 is a directed graph.

V(Graph2) = { 1, 3, 5, 7, 9, 11 }

1), (9, 9), (11, 1) }

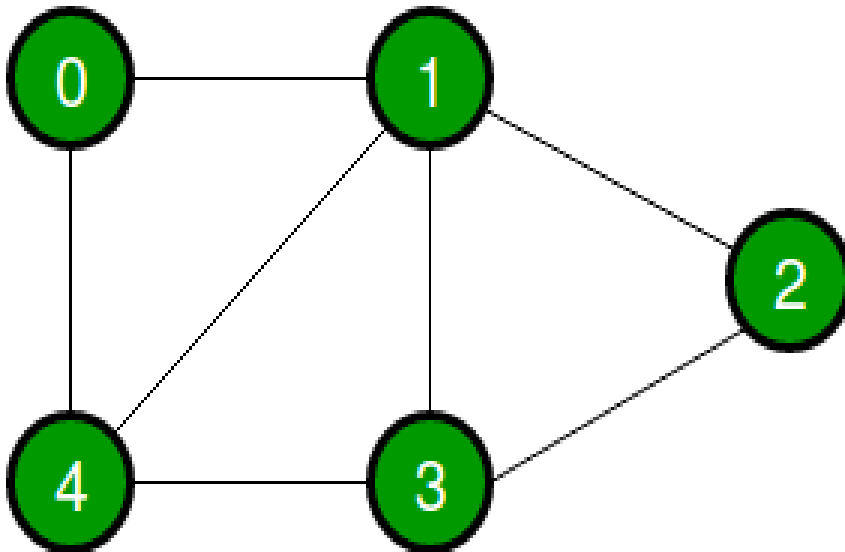# Graph terminology

- <u>Adjacent nodes</u>: two nodes are adjacent if they are connected by an edge

adjacent to 7
7 is adjacent from 5

- <u>Path</u>: a sequence of vertices that connect two nodes in a graph

- <u>Complete graph</u>: a graph in which every vertex is directly connected to every other vertex

Adjacency Matrix

Following two are the most commonly used representations of a graph.

1. Adjacency Matrix

2. Adjacency List

The adjacency matrix for the above example graph is:

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

Adjacency List

NON LINEAR DS-GRAPH /19CS307- DATA STRUCTURES /Mr.R.Kamalakkannan/CSE-IOT/SNSCE
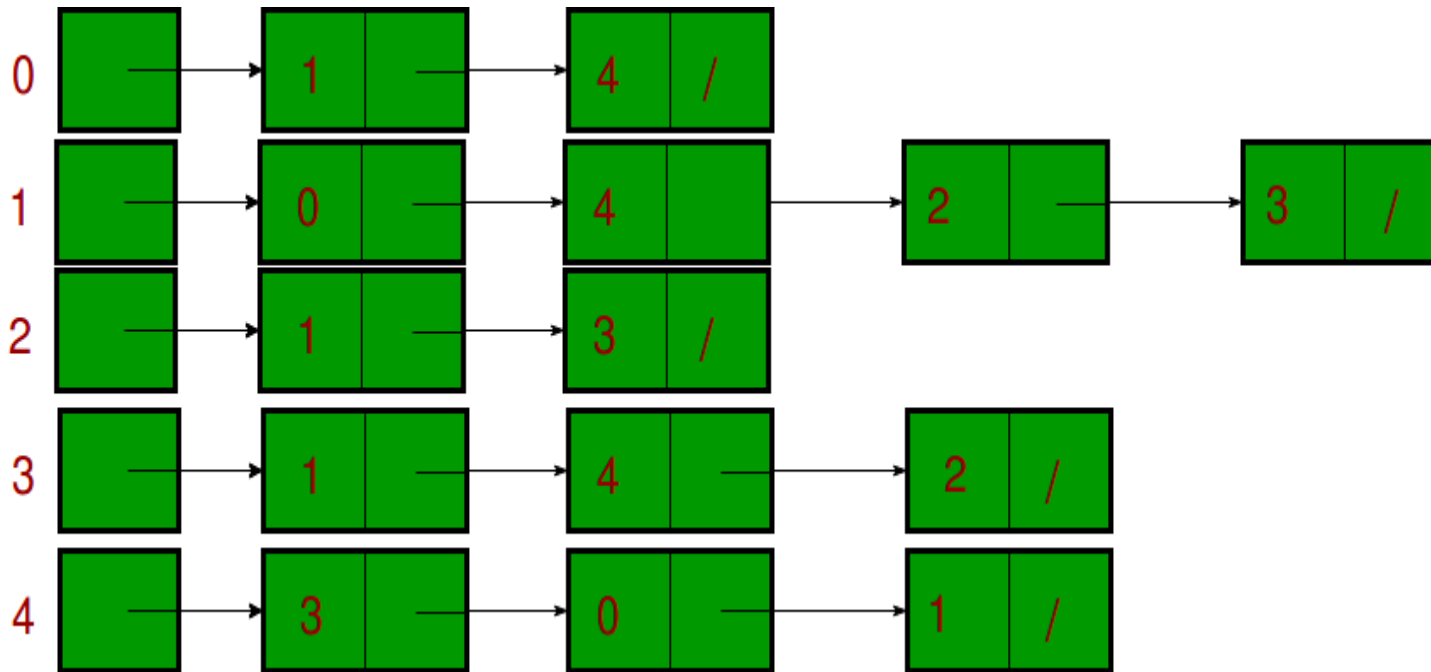
Graph implementation

- ## Array-based implementation
  - A 1D array is used to represent the vertices
  - A 2D array (adjacency matrix) is used to represent the edges
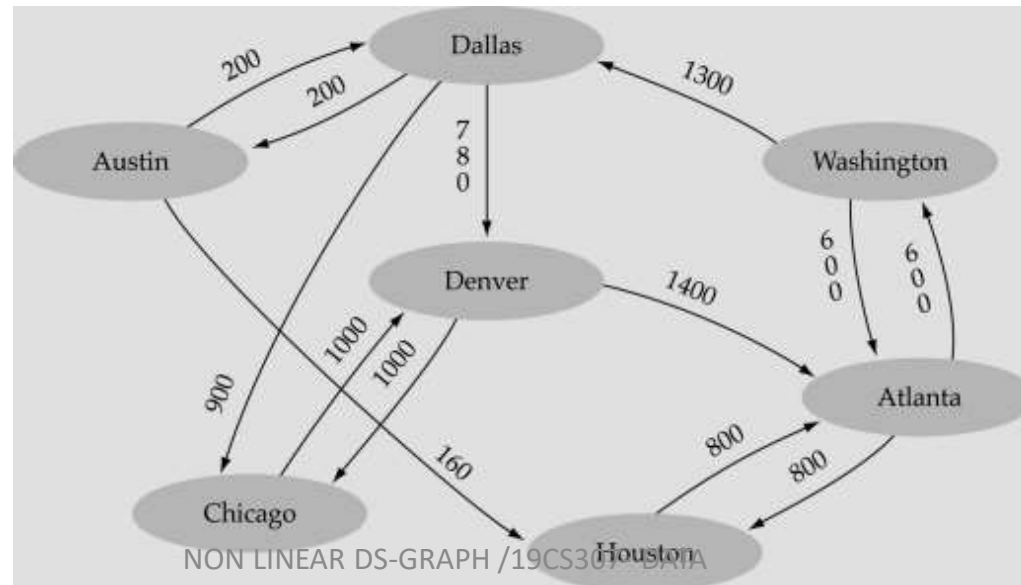
graph

.numVertices 7

.vertices   .edges

			[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
[0]	"Atlanta "	[0]	0	0	0	0	0	800	600	•	•	•
[1]	"Austin "	[1]	0	0	0	200	0	160	0	•	•	•
[2]	"Chicago "	[2]	0	0	0	0	1000	0	0	•	•	•
[3]	"Dallas "	[3]	0	200	900	0	780	0	0	•	•	•
[4]	"Denver "	[4]	1400	0	1000	0	0	0	0	•	•	•
[5]	"Houston "	[5]	800	0	0	0	0	0	0	•	•	•
[6]	"Washington"	[6]	600	0	0	1300	0	0	0	•	•	•
[7]		[7]	•	•	•	•	•	•	•	•	•	•
[8]		[8]	•	•	•	•	•	•	•	•	•	•
[9]		[9]	•	•	•	•	•	•	•	•	•	•

(Array positions marked '•' are undefined)
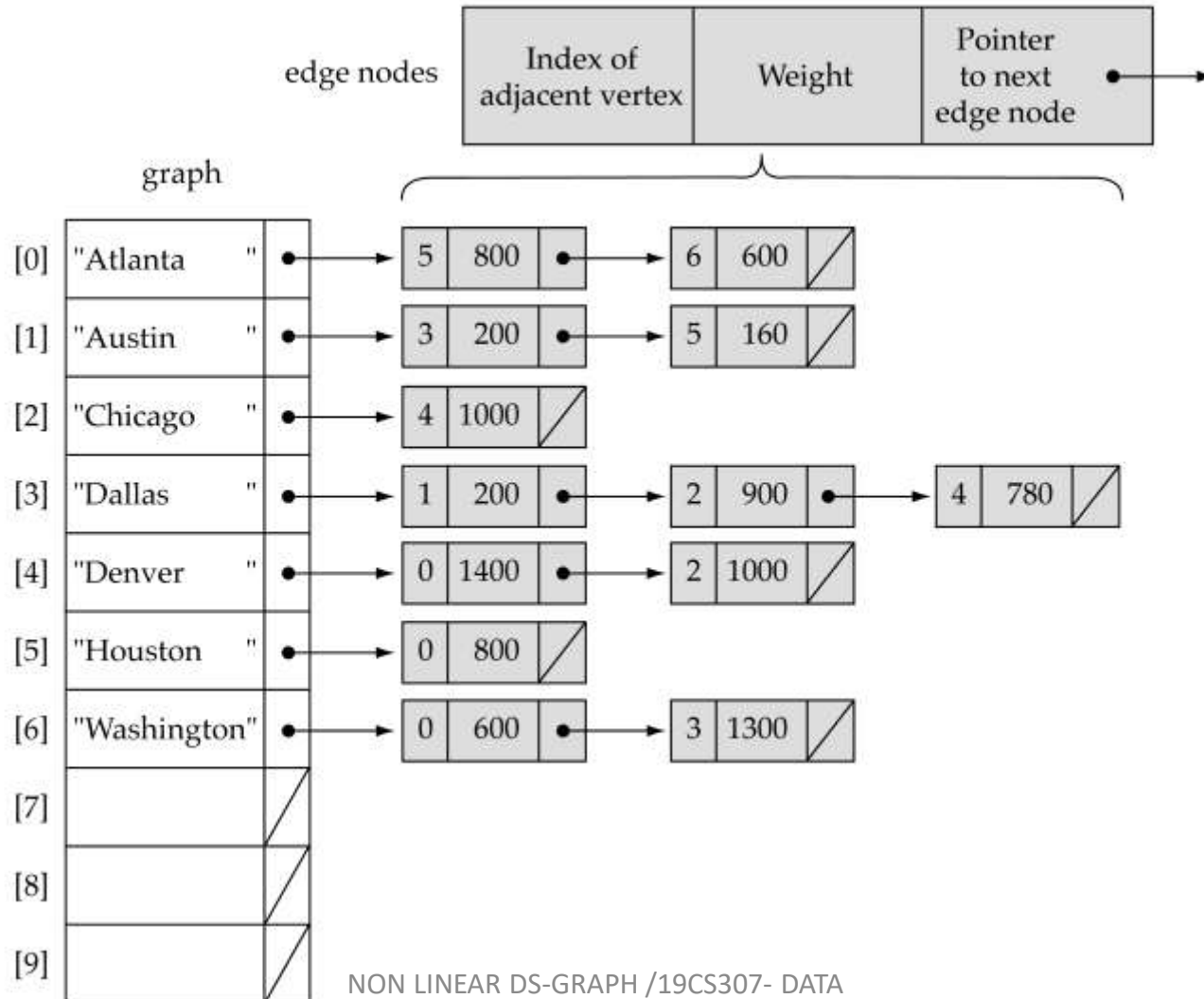
Graph implementation (cont.)

- ## Linked-list implementation

  - A 1D array is used to represent the vertices

  - A list is used for each vertex *v* which contains the vertices which are adjacent from *v* (adjacency list)

(a)

edge nodes

Index of adjacent vertex	Weight	Pointer to next edge node

graph

[0] "Atlanta   " ● → | 5 | 800 | ● | → | 6 | 600 | / |

[1] "Austin   " ● → | 3 | 200 | ● | → | 5 | 160 | / |

[2] "Chicago   " ● → | 4 | 1000 | / |

[3] "Dallas   " ● → | 1 | 200 | ● | → | 2 | 900 | ● | → | 4 | 780 | / |

[4] "Denver   " ● → | 0 | 1400 | ● | → | 2 | 1000 | / |

[5] "Houston   " ● → | 0 | 800 | / |

[6] "Washington" ● → | 0 | 600 | ● | → | 3 | 1300 | / |

[7]

[8]

[9]
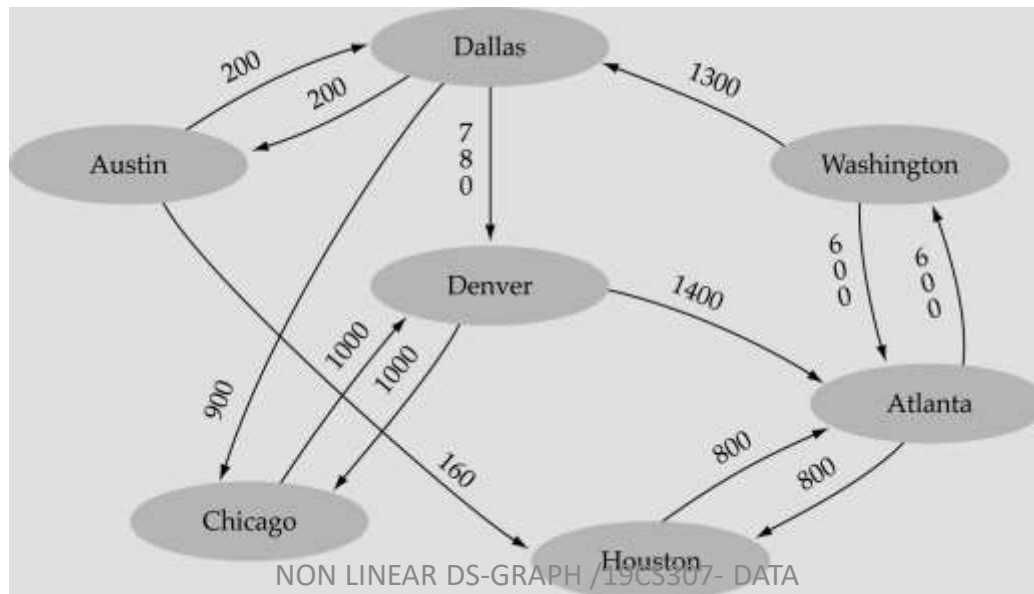
Adjacency matrix vs. adjacency list representation

- **Adjacency matrix**
  - Good for dense graphs
  - Memory requirements
  - Connectivity between two vertices can be tested quickly

- **Adjacency list**
  - Good for sparse graphs
  - Memory requirements
  - Vertices adjacent to another vertex can be found quickly

NON LINEAR DS-GRAPH /19CS307- DATA STRUCTURES /Mr.R.Kamalakkannan/CSE-IOT/SNSCE

ACTIVITY

MCQ

1. Which of the following is true?

   a) A graph may contain no edges and many vertices

   b) A graph may contain many edges and no vertices

   c) A graph may contain no edges and no vertices

   d) A graph may contain no vertices and many edges

2. A graph with all vertices having equal degree is known as a _____

   a) Multi Graph

   b) Regular Graph

   c) Simple Graph

   d) Complete Graph

NON LINEAR DS-GRAPH /19CS307- DATA STRUCTURES /Mr.R.Kamalakkannan/CSE-IOT/SNSCE

Thank you