Graph Data Structure Representations

GRAPH DATA STRUCTURE REPRESENTATIONS (1/2)

SEDGEWICK 4.1

- Three approaches (there are others)
  - Edge List
  - Adjacency Matrix
  - Adjacency List
- There are pros and cons related to amount of memory for, and performance of various algorithms for a given data representation. Common issues:
  - How much space is used?
  - Performance of adding an edge between v and w?
  - Performance checking whether v is adjacent to w?
  - Performance of iterating adjacent vertices to v?

GRAPH DATA STRUCTURE REPRESENTATIONS (2/2)

SEDGEWICK 4.1

- Three approaches (there are others)

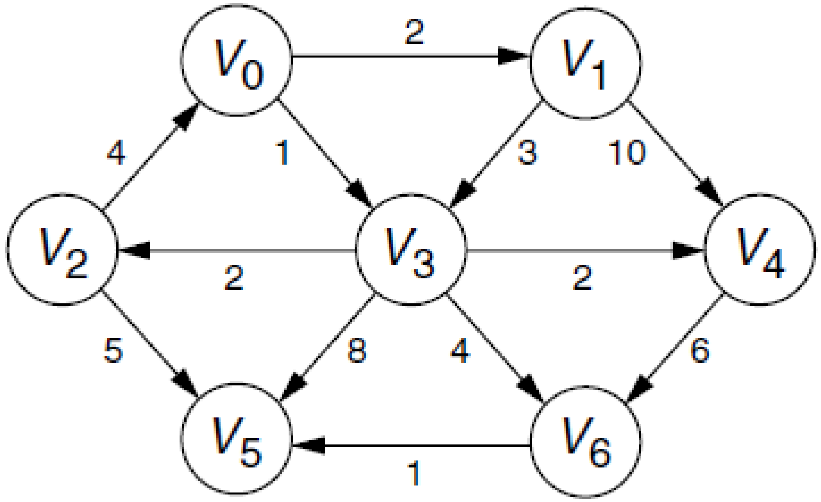
  1. Edge List – a list of edges
  2. Adjacency Matrix a two-dimensional array in which the value at the intersection of two vertices record the edge weight (1 if un-weighted), e.g. m[i][j] = 23
  3. Adjacency List an array of linked lists where the index represents the vertex and the list represents the adjacent vertices. The list node has the neighbor vertex id and edge weight.

- There are pros and cons related to amount of memory for, and performance of various algorithms for a given data representation.

EDGE LIST GRAPH REPRESENTATION

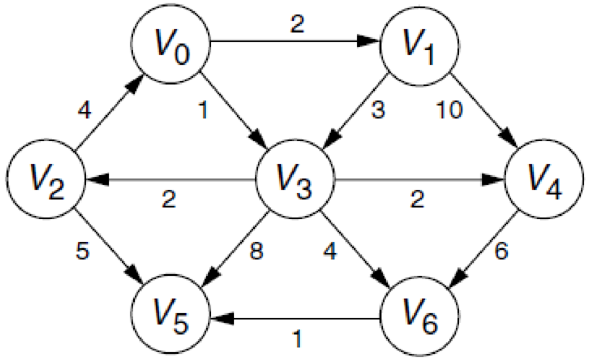
- Edge List is list of edges. For the previous graph (below):

$$E = \left\{\begin{array}{l}(V_0, V_1, 2), (V_0, V_3, 1), (V_1, V_3, 3), (V_1, V_4, 10) \\ (V_3, V_4, 2), (V_3, V_6, 4), (V_3, V_5, 8), (V_3, V_2, 2) \\ (V_2, V_0, 4), (V_2, V_5, 5), (V_4, V_6, 6), (V_6, V_5, 1)\end{array}\right\}$$



ADJACENCY MATRIX GRAPH REPRESENTATION

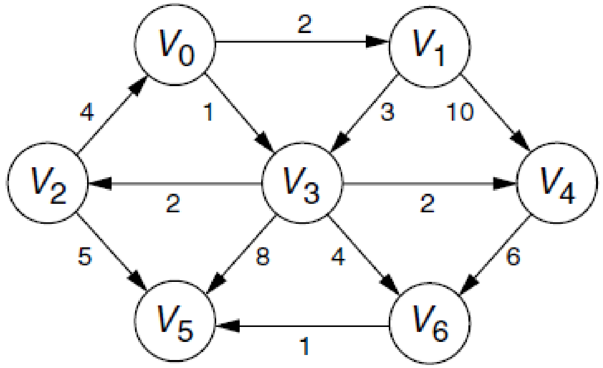
	V0	V1	V2	V3	V4	V5	V6
V0		2		1			
V1				3	10		
V2	4					5	
V3			2		2	8	4
V4							6
V5							
V6						1	



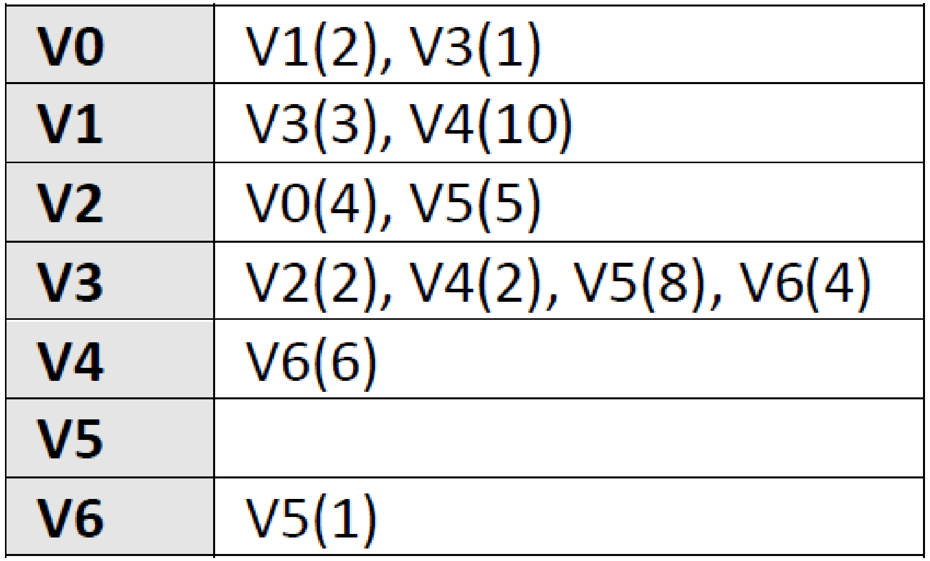
ADJACENCY MATRIX GRAPH REPRESENTATION

- Which is true? $|E| = \Theta(|V|)$ or $|E| = \Theta(|V|^2)$?

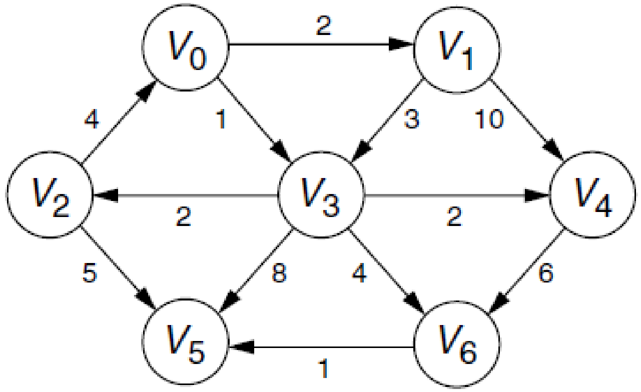
	V0	V1	V2	V3	V4	V5	V6
V0		2		1			
V1				3	10		
V2	4					5	
V3			2		2	8	4
V4							6
V5							
V6						1	



ADJACENCY LIST GRAPH REPRESENTATION



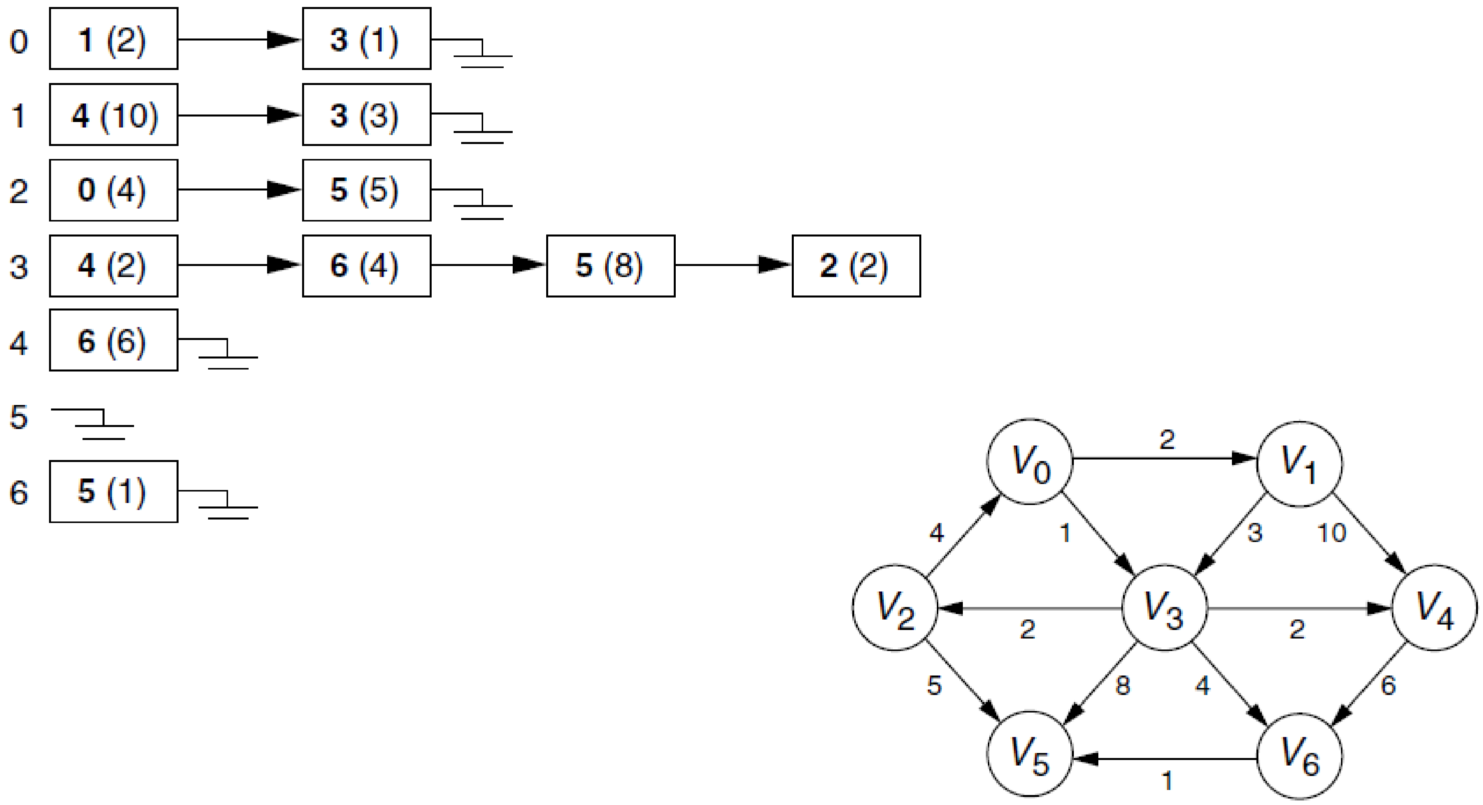
V0	V1(2), V3(1)
V1	V3(3), V4(10)
V2	V0(4), V5(5)
V3	V2(2), V4(2), V5(8), V6(4)
V4	V6(6)
V5	
V6	V5(1)



ADJACENCY LIST GRAPH REPRESENTATION

(WEISS, FIG. 14.2)

- Adjacency List – implemented as a linked list structures



0 1 (2) → 3 (1)

1 4 (10) → 3 (3)

2 0 (4) → 5 (5)

3 4 (2) → 6 (4) → 5 (8) → 2 (2)

4 6 (6)

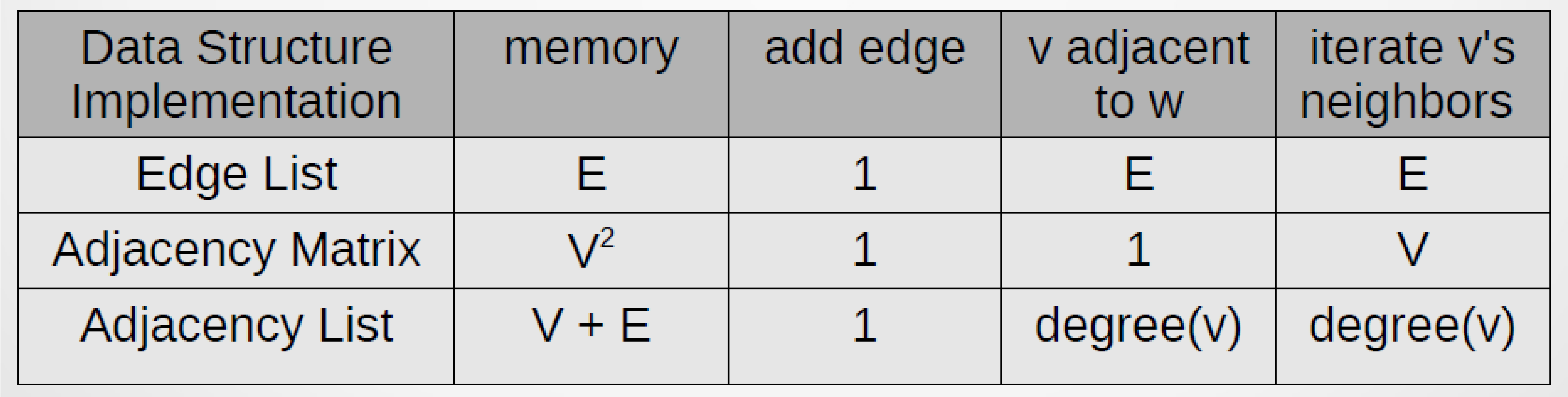
5

6 5 (1)

GRAPH DATA STRUCTURE ORDER OF GROWTH

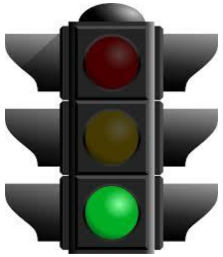
SEDGWICK 4.1

Rule of thumb: When dealing with a dense graph, use an adjacency **matrix**. Use adjacency **list** when dealing with sparse graphs. When you are not sure, use an adjacency list (most applications have sparse graphs).



Data Structure Implementation	memory	add edge	v adjacent to w	iterate v's neighbors
Edge List	E	1	E	E
Adjacency Matrix	V^2	1	1	V
Adjacency List	V + E	1	degree(v)	degree(v)

End of Graph Data Structure Representations



Please proceed to the next course activity now or at a later time