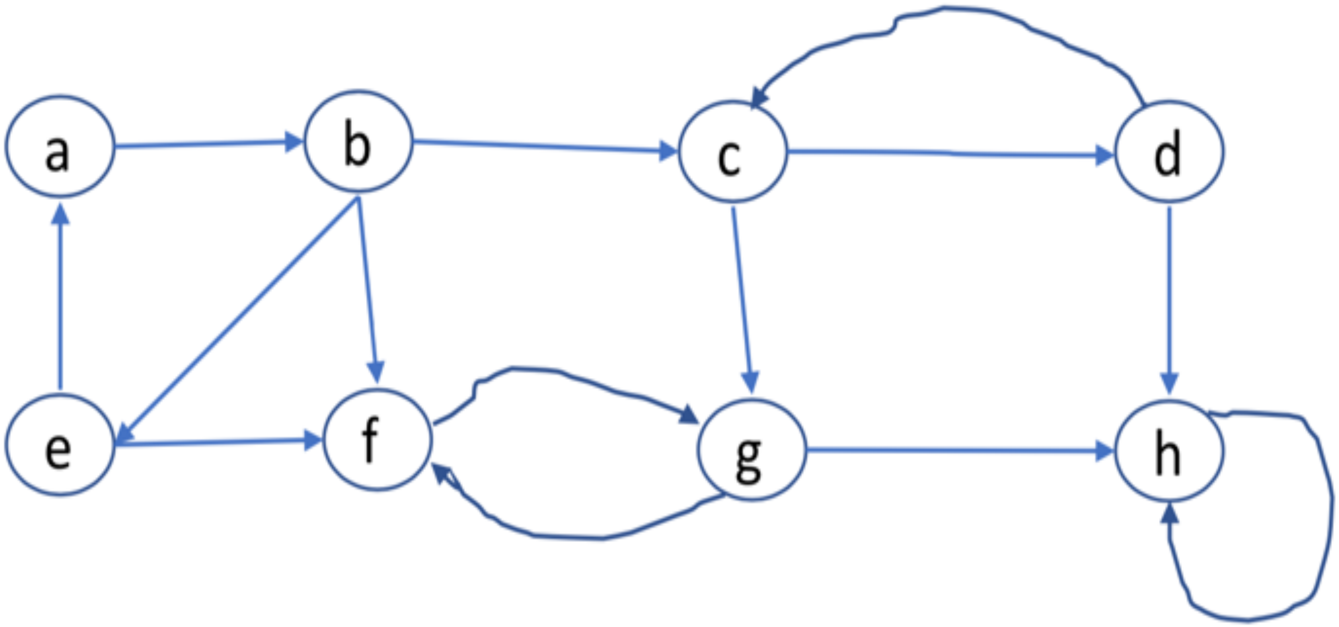
Graphs

References: "Data Structures and Algorithm Analysis", C. Shaffer, pp. 371–388.

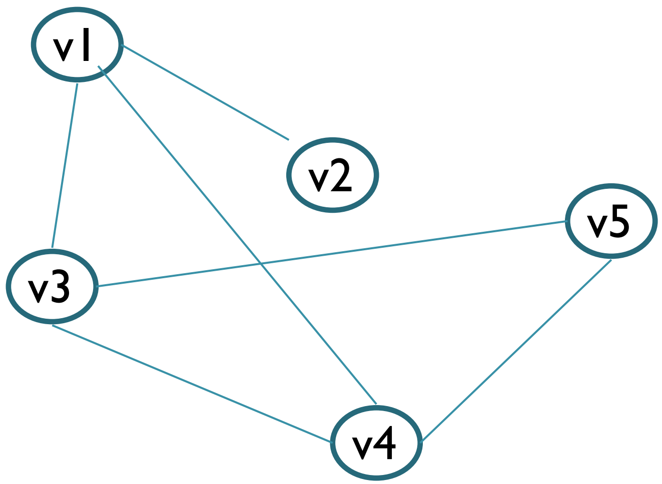
Graphs

- A directed graph G = (V,E), V is set of vertices and E is set of edges, i.e. subset of V x V
- A digraph defines a binary relation on V.
- Useful representation in many applications (e.g. web page links, transportation routes, semantic, social networks)
- InEdges(v) = { (u,v)| (u,v) ϵ E } and outEdges(v) = { (v,u) | (v,u) ϵ E }
- indeg(v) = |InEdges(v)| outDeg(v) = |outEdges(v)|
- $\sum_{v \in V} indeg(v) = \sum_{v \in V} outdeg(v) = |E|$
- |E| ≤ n^2 where n is number of vertices and |E| ≤ $n(n-1)$ if there are no self-cycles.


a
b
c
d
e
f
g
h

Undirected graphs

- In undirected graph, no edge orientation.
  (u,v) ϵ E → (v,u) ϵ E
- Defines a symmetric relation on V. Not reflexive i.e. (u,u) not in E.
- incidentEdges(v) = { (u,v)| (u,v) ϵ E or (v,u) ϵ E}
- deg(v) = |incidentEdges(v)|
- $\sum_{v \in V} deg(v)$ = 2 |E|
- |E| ≤ $n(n-1)/2$


v1
v2
v5
v3
v4

Graph data Structures

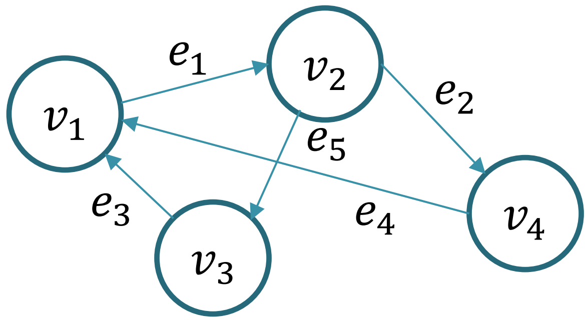
- Adjacency matrix A – n x n boolean matrix where A(i,j) = 1 iff (v_i, v_j) ϵ E where V = $\{v_1, v_2, \dots v_n\}$

- Useful for matrix operations to solve all-pair problems

- Adjacency List – Array L$[0..n-1]$ of lists where L[i] contains the list of indices of vertices v_j such that (v_i, v_j) ϵ E

Space is O(n+m) as opposed to O(n^2) for adjacency matrix. Useful for many efficient graph algorithms.

Graph representations (example)



$$A = \begin{array}{c|cccc} & v_1 & v_2 & v_3 & v_4 \\ \hline v_1 & 0 & 1 & 0 & 0 \\ v_2 & 0 & 0 & 1 & 1 \\ v_3 & 1 & 0 & 0 & 0 \\ v_4 & 1 & 0 & 0 & 0 \end{array}$$

Adjacency Lists