Your Ultimate Guide To Landing Top AI roles

2.19.1

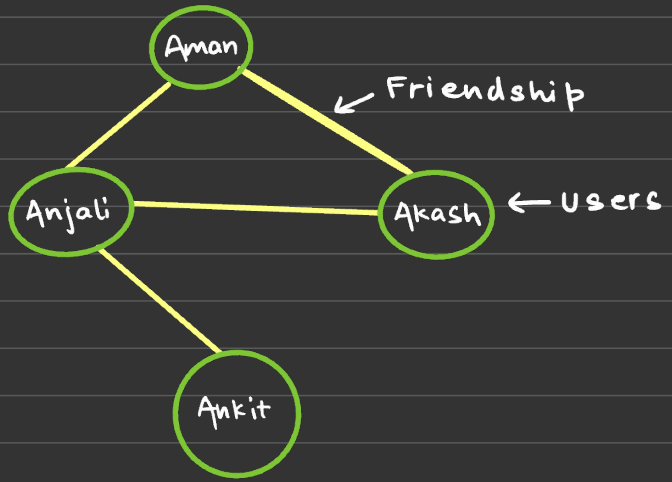
Introduction to Graph

DECODE AiML

→ A graph is a non-linear data structure.
→ Graph is a collection of nodes connected by edges.
→ Tree is a connected acyclic graph.

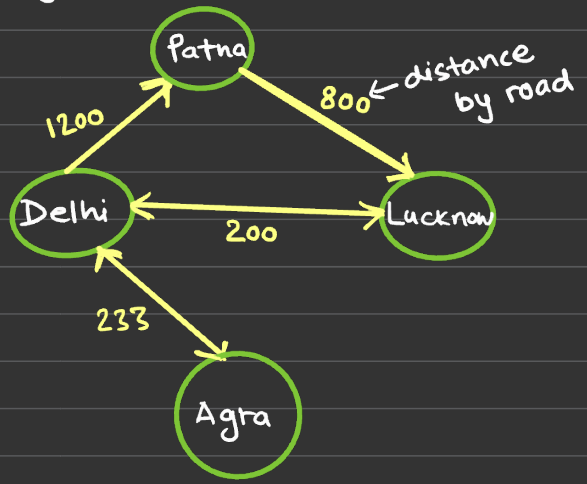
→ Real world usecase of Graph

① Social networks - facebook



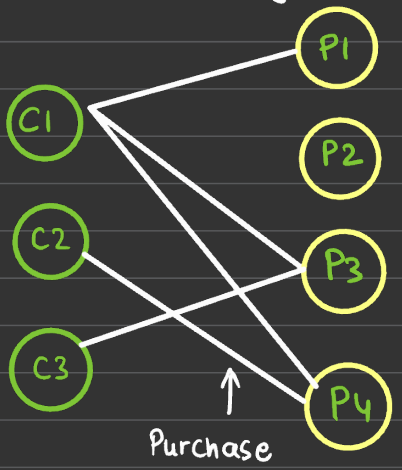
↳ Undirected graph.
↳ Directed for Twitter.

② Google Maps



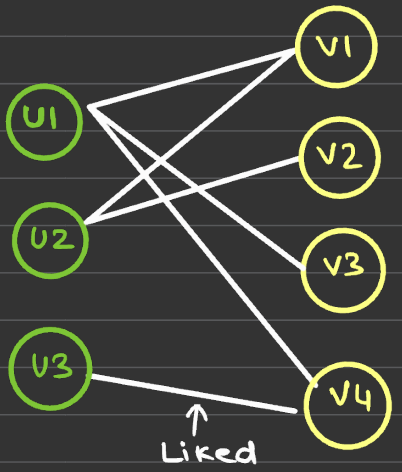
↳ Directed weighted Graph

③ Ecommerce - Customer Purchase history



↳ Undirected Graph

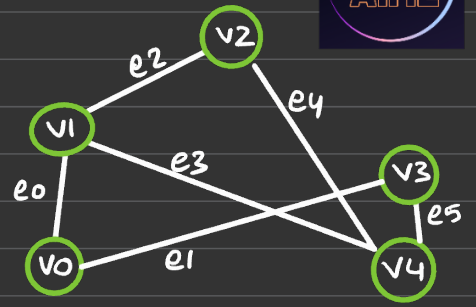
④ Youtube - User Interaction Graph



↳ Undirected Graph.

DECODE AiML

Components of Graphs



① Vertex(Node) : Fundamental unit representing an entity.

$V = \{V0, V1, V2, V3, V4\}$

② Edges : A connection between two vertices.

$E = \{e0, e1, e2, e3, e4, e5\}$

③ Weight: Some edges have weight representing cost, distance or time.

④ Degree of a vertex : No of edges connected to it.

- Indegree : No of incoming edges
- Outdegree: No of outgoing edges.

Types of Graphs

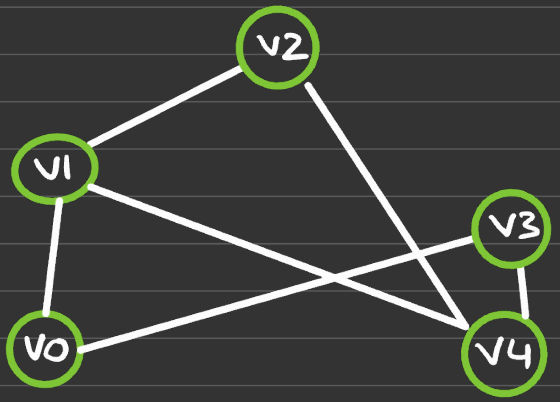
1. Directed Graph → edges have direction
2. Undirected Graph → edges have no direction.
3. Weighted Graph → Each edges has a weight.
4. Unweighted Graph → Edges don't have weight.
5. Cyclic Graph → Contains atleast one cyclic.
6. Connected Graph → Path exist between every pair of vertices. ← Undirected Graphs
7. Disconnected Graph → Not all vertices are connected.
8. Strongly Connected (Directed) → For every pair of vertex u and v, there exists a path $u \to v$ and $v \to u$
9. Weakly Connected (Directed) → Ignoring edge direction, graph is Connected.
10. Complete Graph → Edge for all pair of vertex.
11. Sparse Graph → $|E| \approx O(|V|)$
12. Dense Graph → $|E| \approx O(|V|^2)$

Representation of Graphs

① Adjacency Matrix

→ A 2d array where matrix[i][J] = 1 (or weight) if an edge exist between them, else 0.

→ Shape of matrix will be n x n



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

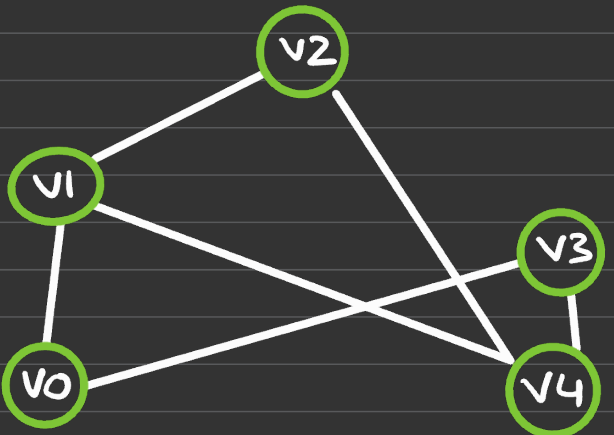
→ Advantage :- Easy to implement. Fast edge lookup.
Limitations :- Space inefficient for sparse graphs

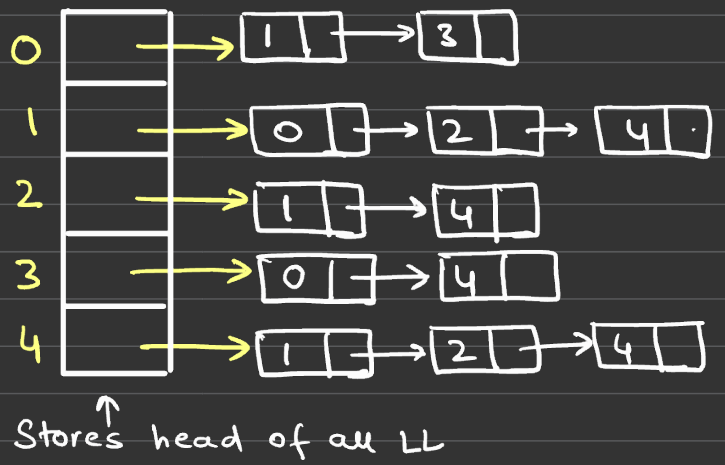
Representation of Graphs

① Adjacency List

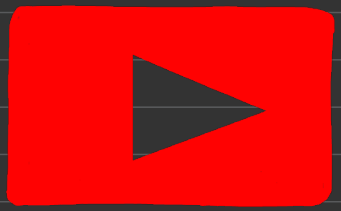
→ It is a representation where for each vertex of graph we maintain a list of its adjacent vertices in LL style.

→ Implement as a list of lists (array of Linked lists or dynamic arrays)





→ Advantage : space efficient for sparse graphs, faster traversal
Limitation : Edge lookup can be slower.

Like  Subscribe