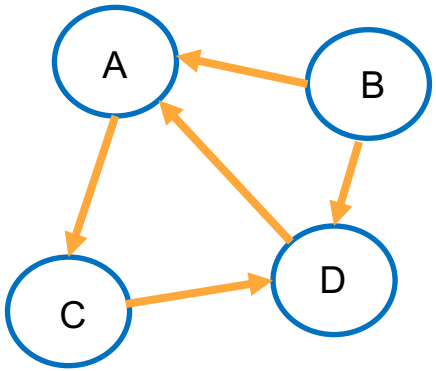
Graphs

Adjacency List

Prepared by Mahdi Ghamkhari

Directed Graph

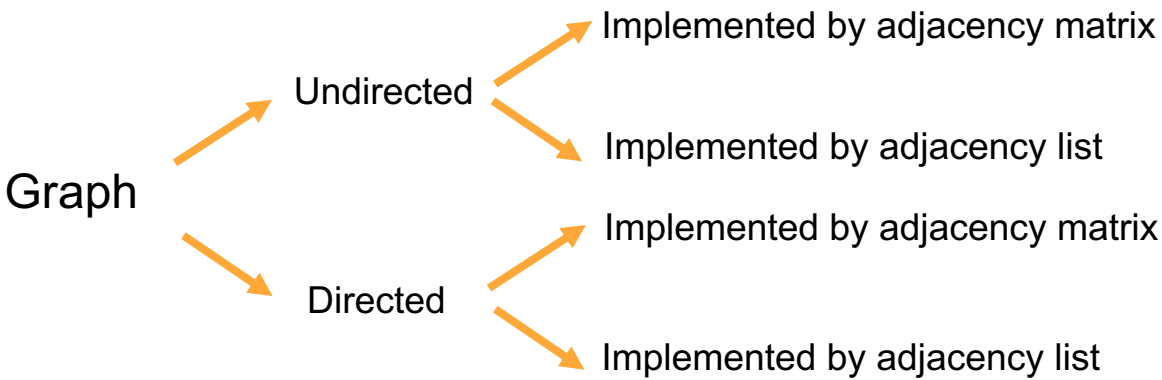


Edges in a directed graph have directions

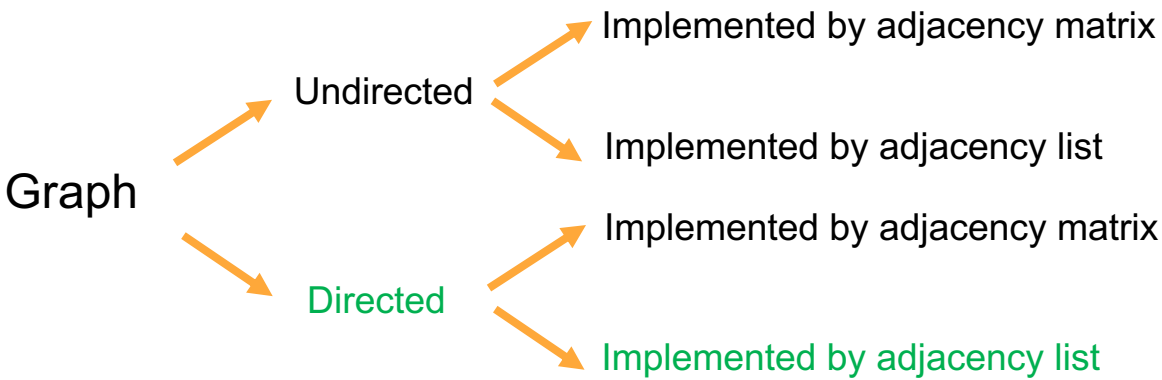
There is an edge from vertex B to Vertex A. Accordingly, A is adjacent to B.

There is no edge from vertex A to Vertex B. Accordingly, B is not adjacent to A

Graph implementation



Graph implementation



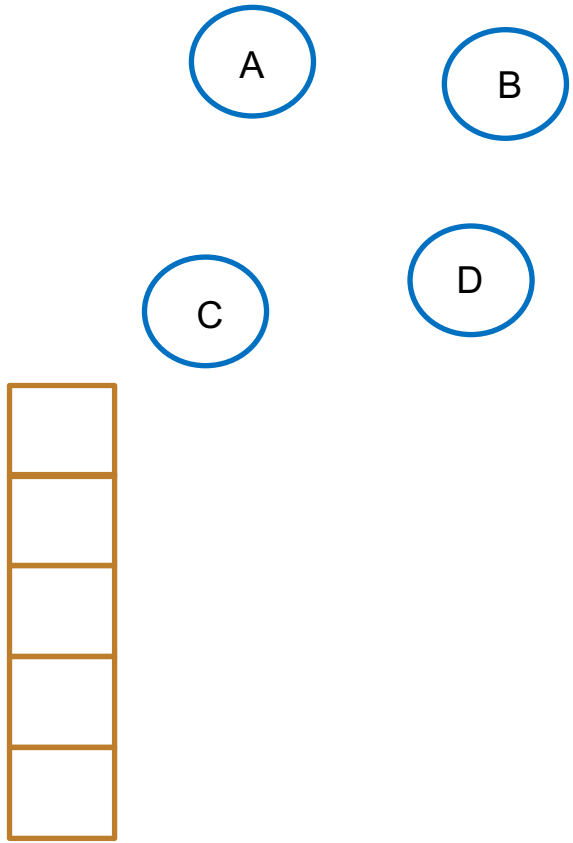
Adjacency list

AdjacencyList = new LinkedList[maxSize];

AdjacencyList is an array of LinkedLists

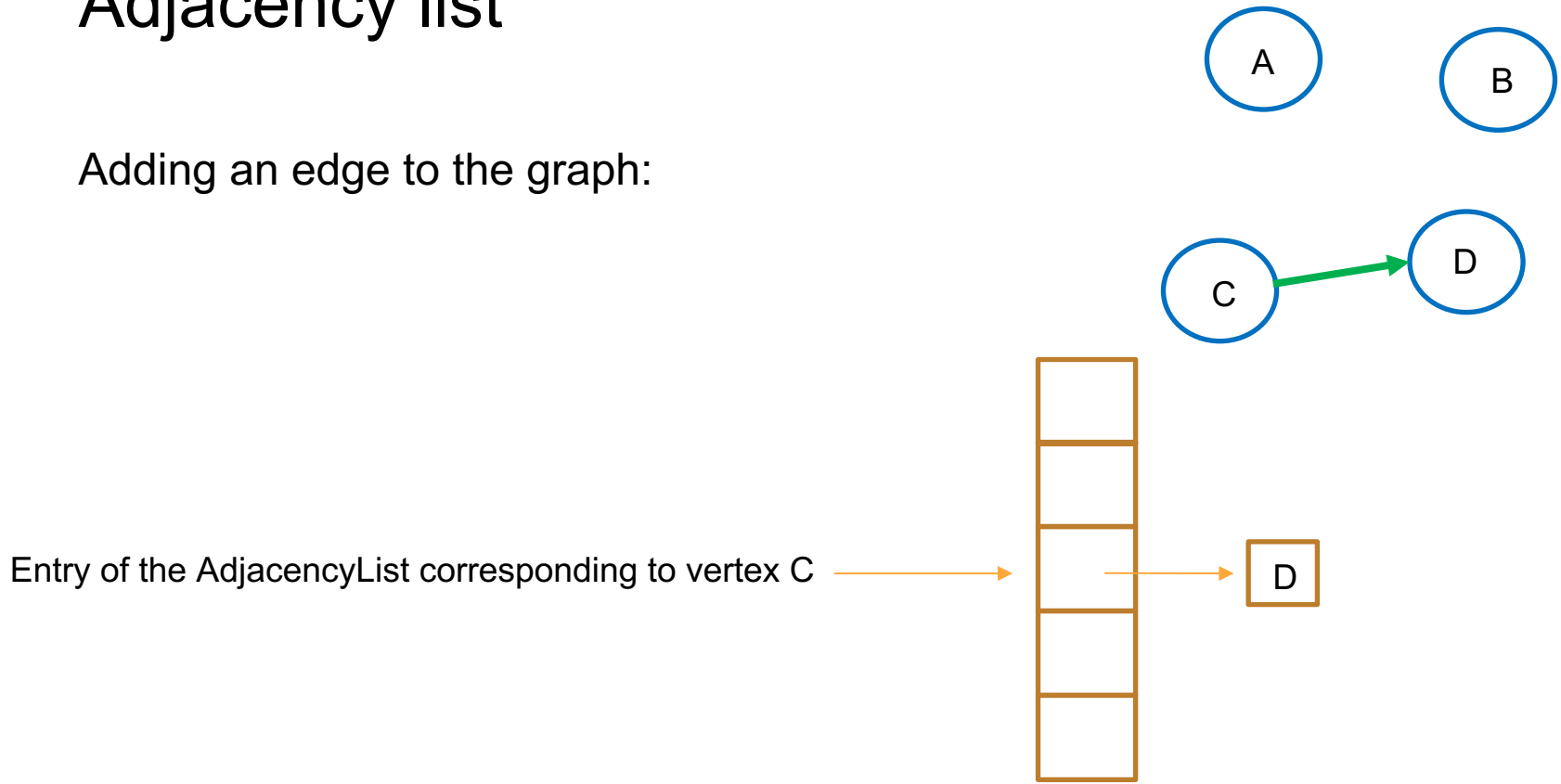
Each entry of AdjacencyList is a reference to a LinkedList

maxSize = 5



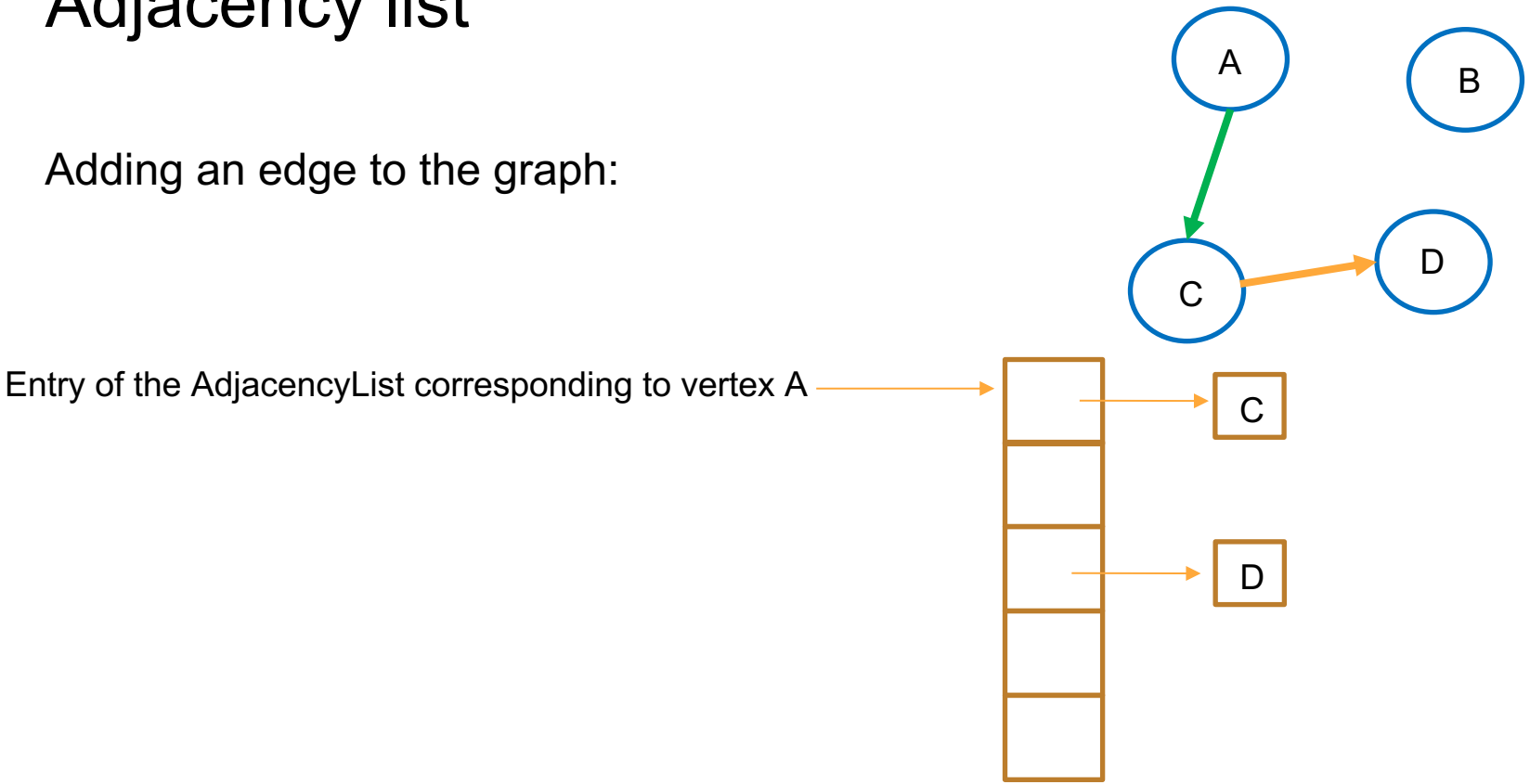
Adjacency list

Adding an edge to the graph:



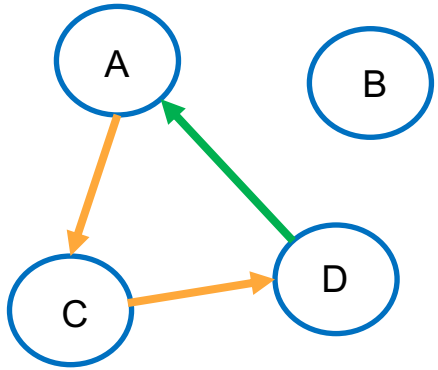
Adjacency list

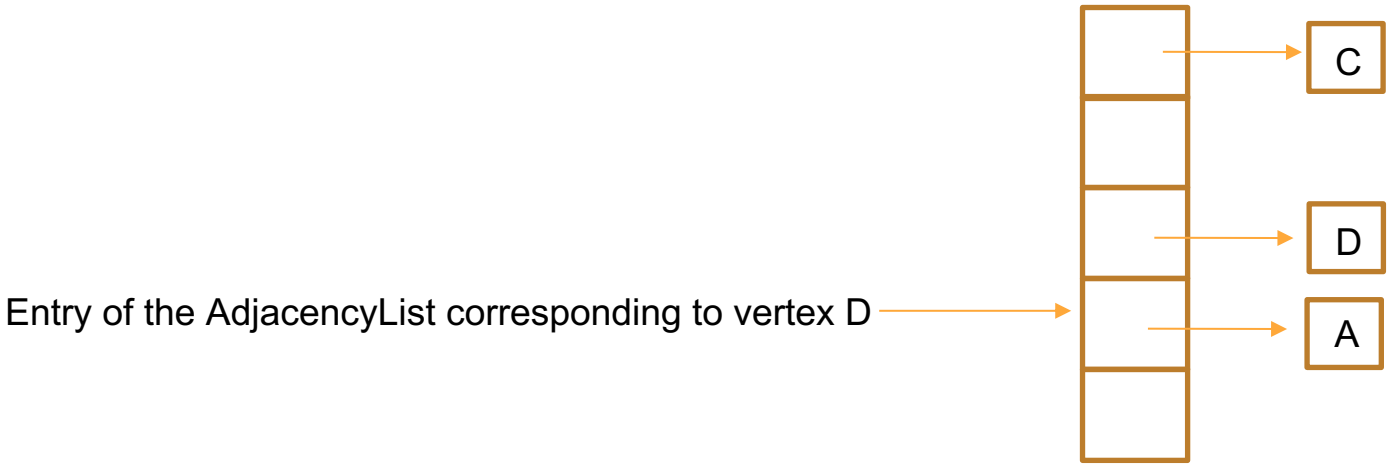
Adding an edge to the graph:



Adjacency list

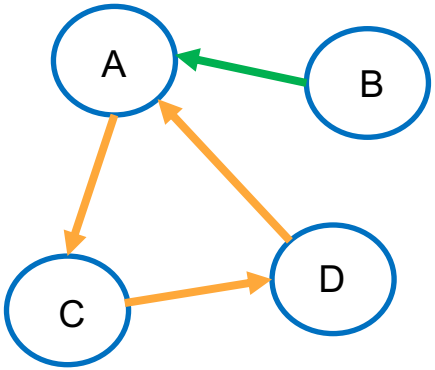
Adding an edge to the graph:



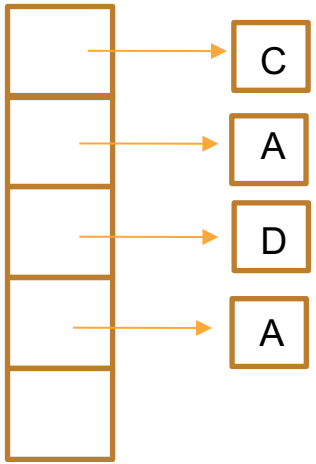


Adjacency list

Adding an edge to the graph:

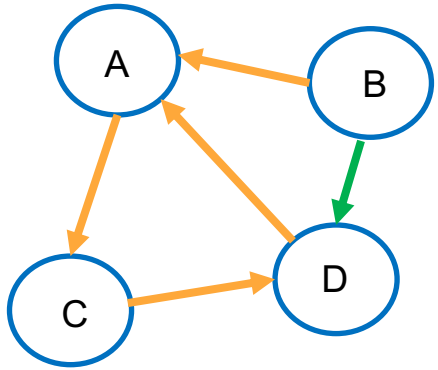


Entry of the AdjacencyList corresponding to vertex B

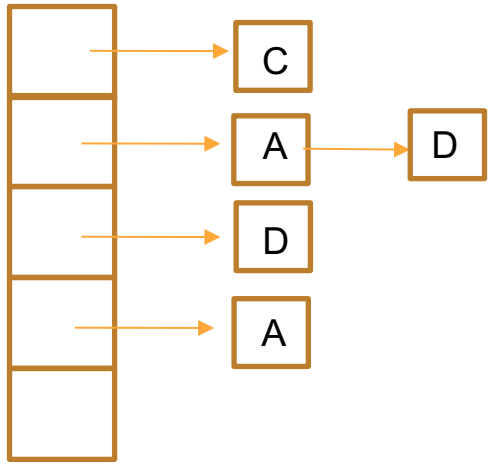


Adjacency list

Adding an edge to the graph:



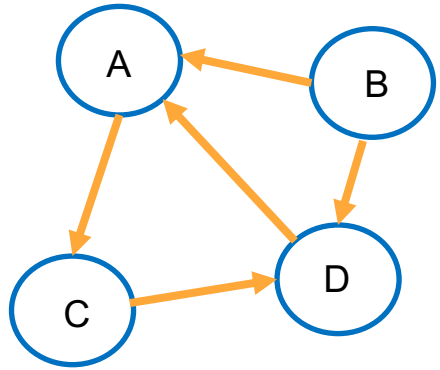
Entry of the AdjacencyList corresponding to vertex B

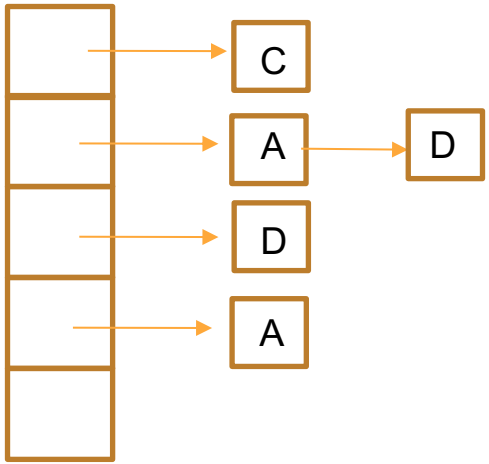


Efficiency

Adding a vertex to the graph has a time complexity of O(1). Because we only need to change one entry of the AdjacencyList array.

Removing a vertex from the graph has a time complexity of O(1). Because we only need to change one entry of the AdjacencyList array.





Efficiency

Adding an edge to the graph has a time complexity of O(n). Because we need to search a linked list to make sure that the edge does not already exist in the linked list.

Removing an edge from the graph has a time complexity of O(n). Because we need to first search a linked list.