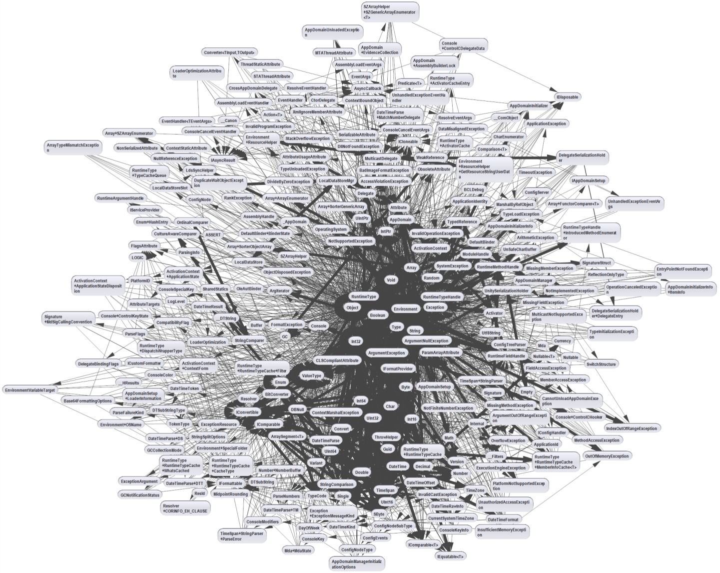
Design and Analysis of Algorithms

Mohammad GANJTABESH

mgtabesh@ut.ac.ir

School of Mathematics, Statistics and Computer Science,
University of Tehran,
Tehran, Iran.

An instance of Graph



Graph Theoretical Problems

- Basec Definitions
- Graph Representation
- Graph Traversal (BFS, DFS)
- Topological Sort
- Strongly Connected Components
- Shortest Paths
  - Single-Source All Destination (Bellman-Ford, Dijkestra)
  - All-Pairs (Floyd-Warshall, Johnson)
- Minimum Spanning Tree (Kruskal, Prim)

Graph: Basic Definitions

Definition

A graph G is a pair (V, E), where V is a finite set and E is a binary relation on V. The set V is called the vertex set of G and the set E is called the edge set of G.

- **Undirected:** if E consists of unordered pairs of vertices.
- **Directed:** if E consists of ordered pairs of vertices.

Graph: Basic Definitions

Definition

A graph G is a pair (V, E), where V is a finite set and E is a binary relation on V . The set V is called the vertex set of G and the set E is called the edge set of G.

- **Undirected:** if E consists of unordered pairs of vertices.
- **Directed:** if E consists of ordered pairs of vertices.

Definition

If (u, v) is an edge in a graph $G = (V, E)$, we say that vertex v is **adjacent to** vertex u.

Graph: Basic Definitions

Definition

A graph G is a pair (V, E), where V is a finite set and E is a binary relation on V . The set V is called the vertex set of G and the set E is called the edge set of G.

- **Undirected:** if E consists of unordered pairs of vertices.
- **Directed:** if E consists of ordered pairs of vertices.

Definition

If (u, v) is an edge in a graph $G = (V, E)$, we say that vertex v is **adjacent to** vertex u.

Definition

If (u, v) is an edge in an undirected graph $G = (V, E)$, we say that (u, v) is **incident on** vertices u and v. If G is directed graph, then we say that (u, v) leaves vertex u and enters vertex v.

Graph: Basic Definitions

Definition

The **degree** of a vertex in an undirected graph is the number of edges incident on it. In a directed graph, the **out-degree** of a vertex is the number of edges leaving it, and the **in-degree** of a vertex is the number of edges entering it. The **degree** of a vertex in a directed graph is its in-degree plus its out-degree.

Graph: Basic Definitions

Definition

The **degree** of a vertex in an undirected graph is the number of edges incident on it. In a directed graph, the **out-degree** of a vertex is the number of edges leaving it, and the **in-degree** of a vertex is the number of edges entering it. The **degree** of a vertex in a directed graph is its in-degree plus its out-degree.

Definition

A **path** of length k from a vertex u to a vertex u' in a graph $G = (V, E)$ is a sequence $\langle v_0, v_1, v_2, \cdots, v_k \rangle$ of vertices such that $u = v_0$, $u' = v_k$, and $(v_{i-1}, v_i) \in E$ for $i = 1, 2, \cdots, k$. The **length** of the path is the number of edges in the path. A path is **simple** if all vertices in the path are distinct. If there is a path p from u to u', we say that u' is reachable from u via p.

Graph: Basic Definitions

Definition

The **degree** of a vertex in an undirected graph is the number of edges incident on it. In a directed graph, the **out-degree** of a vertex is the number of edges leaving it, and the **in-degree** of a vertex is the number of edges entering it. The **degree** of a vertex in a directed graph is its in-degree plus its out-degree.

Definition

A **path** of length k from a vertex u to a vertex u' in a graph $G = (V, E)$ is a sequence $\langle v_0, v_1, v_2, \cdots, v_k \rangle$ of vertices such that $u = v_0$, $u' = v_k$, and $(v_{i-1}, v_i) \in E$ for $i = 1, 2, \cdots, k$. The **length** of the path is the number of edges in the path. A path is **simple** if all vertices in the path are distinct. If there is a path p from u to u', we say that u' is reachable from u via p.

Definition

In a directed graph, a path $\langle v_0, v_1, v_2, \cdots, v_k \rangle$ forms a **cycle** if $v_0 = v_k$ and the path contains at least one edge. The cycle is **simple** if $v_1, v_2, \cdots, v_k$ are distinct.

Graph: Basic Definitions

Definition

An undirected graph is **connected** if every pair of vertices is connected by a path. The **connected components** of a graph are the equivalence classes of vertices under the ”is reachable from” relation.

Graph: Basic Definitions

Definition

An undirected graph is **connected** if every pair of vertices is connected by a path. The **connected components** of a graph are the equivalence classes of vertices under the ”is reachable from” relation.

Definition

A directed graph is **strongly connected** if every two vertices are reachable from each other. The **strongly connected components** of a directed graph are the equivalence classes of vertices under the are mutually reachable relation.

Graph: Basic Definitions

Definition

An undirected graph is **connected** if every pair of vertices is connected by a path. The **connected components** of a graph are the equivalence classes of vertices under the ”is reachable from” relation.

Definition

A directed graph is **strongly connected** if every two vertices are reachable from each other. The **strongly connected components** of a directed graph are the equivalence classes of vertices under the are mutually reachable relation.

Definition

Two graphs $G = (V, E)$ and $G' = (V', E')$ are **isomorphic** if there exists a bijection $f : V \mapsto V'$ such that $(u, v) \in E$ if and only if $(f(u), f(v)) \in E'$.

Graph: Basic Definitions

Definition

We say that a graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if $V' \subset V$ and $E' \subset E$. Given a set $V' \subset V$, the subgraph of G **induced** by V' is the graph $G' = (V', E')$, where $E' = \{(u, v) \in E \ : \ u, v \in V'\}$.

Graph: Basic Definitions

Definition

We say that a graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if $V' \subset V$ and $E' \subset E$. Given a set $V' \subset V$, the subgraph of G **induced** by V' is the graph $G' = (V', E')$, where $E' = \{(u, v) \in E \,:\, u, v \in V'\}$.

Definition

A **complete graph** is an undirected graph in which every pair of vertices is adjacent.

Graph: Basic Definitions

Definition

We say that a graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if $V' \subset V$ and $E' \subset E$. Given a set $V' \subset V$, the subgraph of G **induced** by V' is the graph $G' = (V', E')$, where $E' = \{(u, v) \in E \,:\, u, v \in V'\}$.

Definition

A **complete graph** is an undirected graph in which every pair of vertices is adjacent.

Definition

A **bipartite graph** is an undirected graph $G = (V, E)$ in which V can be partitioned into two sets V_1 and V_2 such that $(u, v) \in E$ implies either $u \in V_1$ and $v \in V_2$ or $u \in V_2$ and $v \in V_1$.

Graph: Basic Definitions

Definition

We say that a graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if $V' \subset V$ and $E' \subset E$. Given a set $V' \subset V$, the subgraph of G **induced** by V' is the graph $G' = (V', E')$, where $E' = \{(u, v) \in E \ : \ u, v \in V'\}$.

Definition

A **complete graph** is an undirected graph in which every pair of vertices is adjacent.

Definition

A **bipartite graph** is an undirected graph $G = (V, E)$ in which V can be partitioned into two sets V_1 and V_2 such that $(u, v) \in E$ implies either $u \in V_1$ and $v \in V_2$ or $u \in V_2$ and $v \in V_1$.

Definition

An acyclic undirected graph is a **forest**, and a connected acyclic undirected graph is a **tree**. Also, **DAG** is directed Acyclinc Graph.

Representations of graphs

- **adjacency matrix:** this representation of a graph G consists of a $|V| \times |V|$ matrix $A = (a_{ij})$ such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Representations of graphs

- **adjacency matrix:** this representation of a graph G consists of a $|V| \times |V|$ matrix $A = (a_{ij})$ such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

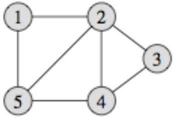
- **adjacency list:** this representation of a graph $G = (V, E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V. For each $u \in V$, the adjacency list $Adj[u]$ contains all the vertices adjacent to u in G.

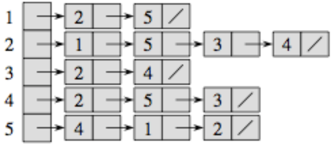
Representations of graphs

- **adjacency matrix:** this representation of a graph G consists of a $|V| \times |V|$ matrix $A = (a_{ij})$ such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

- **adjacency list:** this representation of a graph $G = (V,E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V. For each $u \in V$, the adjacency list $Adj[u]$ contains all the vertices adjacent to u in G.



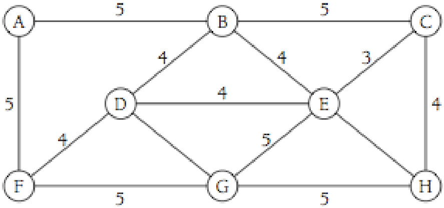


$$\begin{array}{c|ccccc|} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 1 & 0 & 0 & 1 \\ 2 & 1 & 0 & 1 & 1 & 1 \\ 3 & 0 & 1 & 0 & 1 & 0 \\ 4 & 0 & 1 & 1 & 0 & 1 \\ 5 & 1 & 1 & 0 & 1 & 0 \\ \hline \end{array}$$

Representations of graphs

Definition

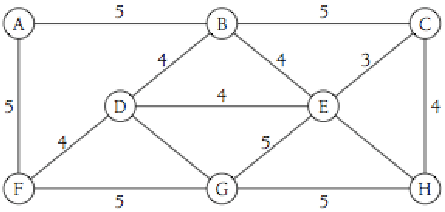
A graph is weighted if each edge has an associated weight, typically given by a weight function $w : E \mapsto R$.



Representations of graphs

Definition

A graph is weighted if each edge has an associated weight, typically given by a weight function $w : E \mapsto R$.



- Adjacency matrix can be adapted to represent weighted graphs (How?).
- Adjacency list can also be adapted to represent weighted graphs (How?).

Graph Traversal: Breadth First Search

Breadth First Search

Given a graph $G = (V, E)$ and a distinguished source vertex s, **breadth-first search** systematically explores the edges of G to discover every vertex that is reachable from s.

- It computes the distance (smallest number of edges) from s to each reachable vertex.
- It also produces a **breadth-first tree** with root s that contains all reachable vertices.

Graph Traversal: Breadth First Search

Breadth First Search

Given a graph $G = (V, E)$ and a distinguished source vertex s, **breadth-first search** systematically explores the edges of G to discover every vertex that is reachable from s.

- It computes the distance (smallest number of edges) from s to each reachable vertex.
- It also produces a **breadth-first tree** with root s that contains all reachable vertices.

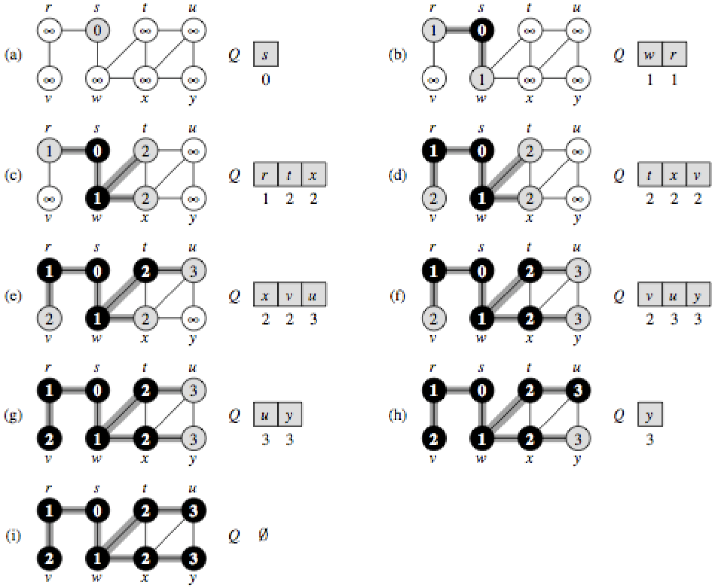
For each vertex $v \in V$, we store the following information during the execution of BFS:

- $color[v] \in \{white, gray, black\}$
  - white: if the vertex v is not yet discovered.
  - gray: if the vertex v is discovered but its neighbors are not.
  - black: if the vertex v and all its neighbors are discovered.
- $d[v]$: the number of edges from s to v in search tree.
- $p[v]$: the parent of v in search tree.

Graph Traversal: Breadth First Search

```
BFS(G, s)
for each vertex u ∈ V[G] − {s}
       do color[u] ← WHITE
          d[u] ← ∞
          π[u] ← NIL
color[s] ← GRAY
d[s] ← 0
π[s] ← NIL
Q ← ∅
ENQUEUE(Q, s)
while Q ≠ ∅
       do u ← DEQUEUE(Q)
          for each v ∈ Adj[u]
              do if color[v] = WHITE
                    then color[v] ← GRAY
                         d[v] ← d[u] + 1
                         π[v] ← u
                         ENQUEUE(Q, v)
          color[u] ← BLACK
```

Graph Traversal: Breadth First Search



Exercises

1. Describe the Adjacency Multi-list representation of a graph.
2. The square of a directed graph $G = (V, E)$ is the graph $G^2 = (V, E^2)$ such that $(u, w) \in E^2$ if and only if for some $v \in V$, both $(u, v) \in E$ and $(v, w) \in E$. That is, G^2 contains an edge between u and w whenever G contains a path with exactly two edges between u and w. Describe efficient algorithms for computing G^2 from G for both the adjacency-list and adjacency-matrix representations of G. Analyze the running times of your algorithms.
3. The incidence matrix of a directed graph $G = (V, E)$ is a $|V| \times |E|$ matrix $B = (b_{ij})$ such that

$$b_{ij} = \begin{cases} -1 & \text{if edge } j \text{ leaves vertex } i, \\ 1 & \text{if edge } j \text{ enters vertex } i, \\ 0 & \text{otherwise.} \end{cases}$$

   Describe what the entries of the matrix product $B \times B^T$ represent, where B^T is the transpose of B.
4. The diameter of a tree $T = (V, E)$ is given by

$$\max \delta(u, v)\ ;\ u, v \in V$$

   that is, the diameter is the largest of all shortest-path distances in the tree. Give an efficient algorithm to compute the diameter of a tree, and analyze the running time of your algorithm.