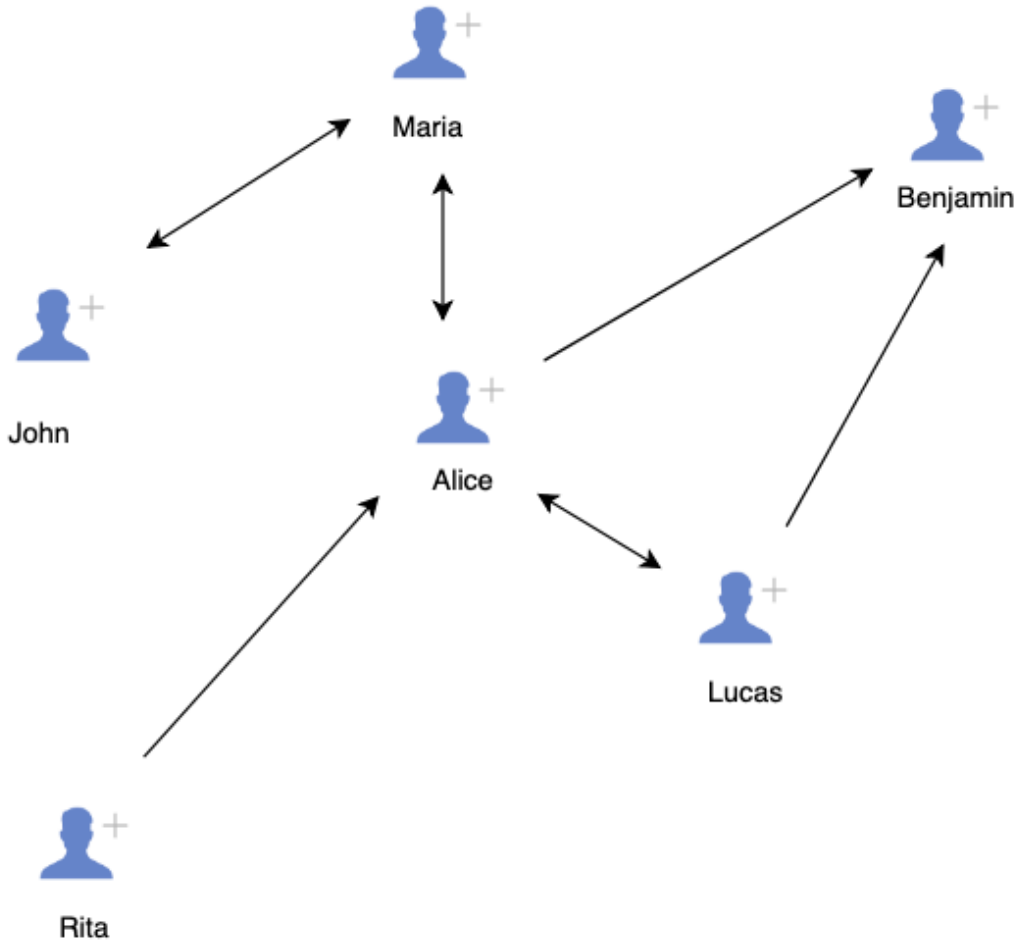
Graphs and Graph Terminologies

Background

We use graphs to represent many real-life entities. Consider a social network (as shown in Figure 1) where people can follow other people. This can be represented by a graph. Each people represents a vertex (or node) and the edge between two people tells the relationship between them in terms of *following*.

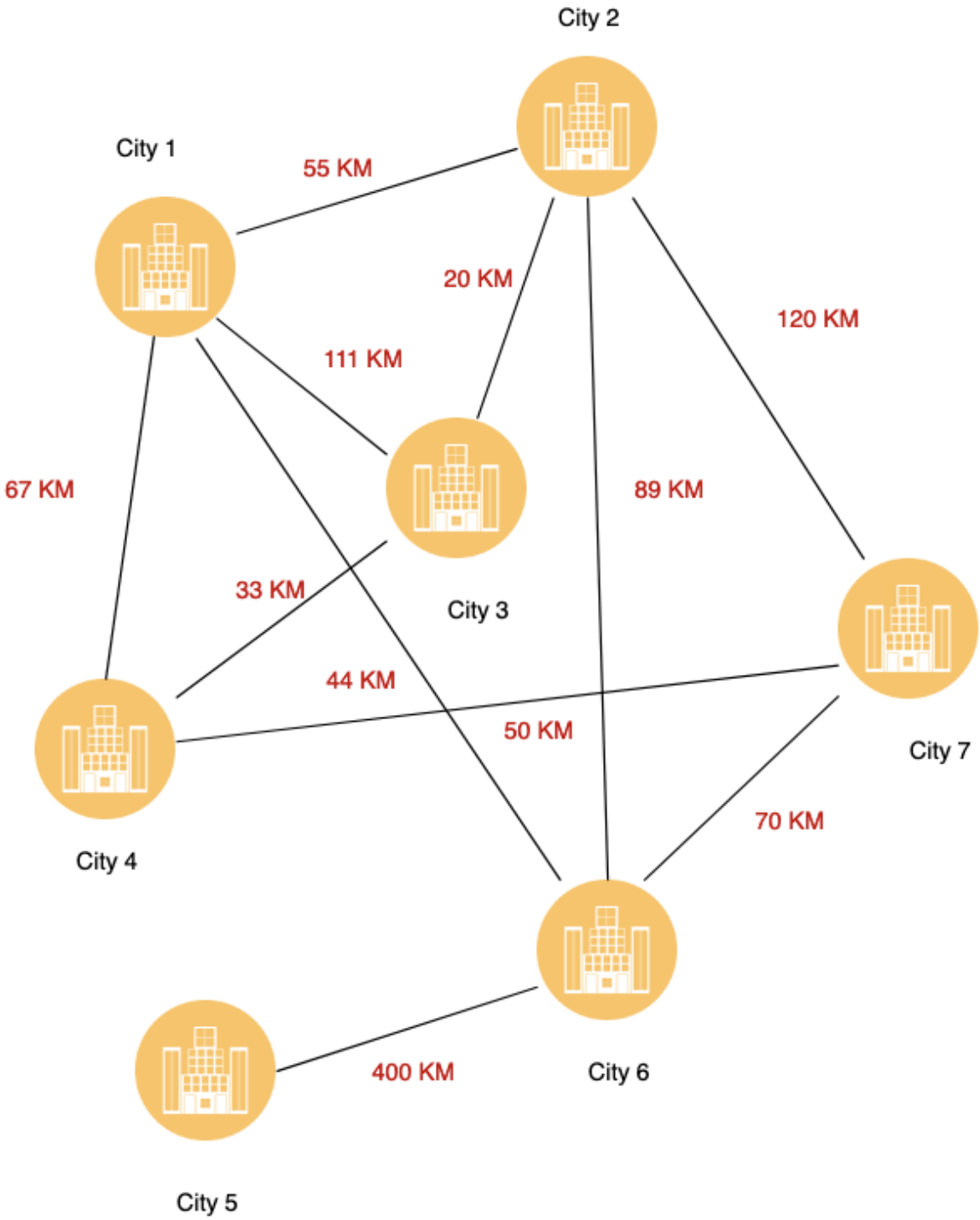


(

Figure 1: Graph Representing Social Network

As we see in Figure 1, each person acts as a node in the graph. If a person A has an outgoing edge to person B, that means A has followed B. In Figure 1, Rita has followed Alice, Alice has followed Benjamin, John has followed Maria, Maria has followed John and so on.

Similarly, a graph can represent cities linked by roads. Figure 2 depicts this.



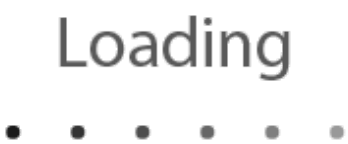
(

Figure 2: Graph Representing Cities

The nodes of the graph represent cities and an edge between two cities represent the road between them. If there is an edge between cities A and B that means they are connected by a road. Notice one extra information (length of the road) in the edge that was not present in the social network graph. This kind of graphs are called weighted graph and we will cover them later in the post.

Introduction

Formally, a graph $G = (V, E)$ is defined on a set of vertices V, and contains a set of edges E. An edge is a pair of vertices which can be ordered or unordered depending upon whether the edge is directed or undirected. Usually, a vertex is represented by a lower case u or v and an edge is represented by the pair of u and v. A directed edge is written as an ordered pair (u, v) while the undirected edge is written as an unordered pair $\{u, v\}$. Figure 3 depicts an example of a graph.



(

Figure 3: Illustrating Graph

Types of graph

There are many flavors of graphs we use in computer science. We discuss some of them here.

Undirected and directed

A graph $G = (V, E)$ is undirected if edge $(u, v) \in E$ implies that edge (v, u) is also in E. In simple English sentence, a graph is called undirected if the edge can be traversed from both of its endpoints. In the similar way, the graph G is directed if edge $(u, v) \in E$ and edge $(v, u) \notin E$. This is illustrated in Figure 4.

Loading

(

Figure 4: Illustrating Undirected and Directed Graph

In a visual representation, undirected edges are drawn as a line segment and directed edges are drawn as a line segment with an arrow on one of the endpoints.

Weighted and unweighted

A weighted graph G has a numeric value attached to its edges. We call this numeric value a *weight* of the edge. In Figure 2, the weight is the length of the road joining cities. The weight can represent varieties of things depending upon the application. In an electric circuit, weight can be the amount of current flowing through the wire. In a road network, weight can be the length of the road, speed limit or the difficulty level. Figure 5 illustrates this.

Loading

(

Figure 5: Illustrating weighted and unweighted graph

In computer science, a weighted graph is used heavily in the shorted path problems.

Simple and non-simple

A simple graph has no self-loops and no multi-edges. Self-loop is an edge going from a node to itself i.e. (u, u). Multi-edge is the edge occurring more than one time between the same endpoints. A graph containing one or more self-loops or multi-edges is a non-simple graph. Figure 6 shows examples of these graphs.

Loading

(

Figure 6: Illustrating simple and non-simple graph

Sparse and Dense

A graph can have a quadratic number of edges. If V is the number of vertices in a graph, it can have up to $O(V^2)$ edges. A graph having edges in this order is called a dense graph (Usually). On the other hand, a graph having a fewer number of edges is called a sparse graph. If a graph has an edge between every pair of nodes, we call this graph a *complete graph*. Figure 7 illustrates a sparse and dense graph.

Loading

(

Figure 7: Illustrating sparse and dense graph

Cyclic and acyclic

A graph having no cycles is an acyclic graph. A tree is a connected acyclic graph. A directed graph with no cycles is called a Direct Acyclic Graph (DAG) and has many use cases in computer science including the scheduling problems. Scheduling algorithm like topological sorting requires the graph to be a DAG. A graph with one or more cycles is called a cyclic graph. Figure 8 depicts examples of Cyclic and Acyclic graph.

Loading

(

Figure 8: Illustrating cyclic and acyclic graph

Terminologies

In this section, we discuss graph terminologies that you are most likely to encounter when studying about graphs.

1. The two vertices of an undirected graphs are called **endpoints**. In directed graph, we distinguish endpoints by calling them **tail** and **head**. For edge (u, v), we call u the tail and v the head.
2. If $\{u, v\}$ is an edge in an undirected edge, we call u the **neighbor** of v and vice versa. The number of neighbors of a node is called the **degree** of the node.
3. If (u, v) is an edge in a directed graph, we call u a **predecessor** or **in-neighbor** of v and v a **successor** or **out-neighbor** of u. The **in-degree** of a node is the number of predecessors; the **out-degree** is the number of successors.
4. A graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$.
5. In an undirected graph, a **walk** is a sequence of vertices, where each successive pair of vertices are adjacent. A walk is a **path** if it visits each vertex at most once.
6. For any two vertices u and v in a graph G, we say that v is **reachable** from u if G contains a walk (and therefore a path) between u and V.
7. An undirected graph is **connected** if every vertex is reachable from every other vertex.
8. A **component** is a maximal connected subgraph. Two vertices are in the same component if and only if there is a path between them.
9. A **directed walk** is a sequence of directed edges, where the head of each edge is the tell of the next.
10. A **directed path** is a directed walk without repeated vertices. Vertex v is **reachable** from vertex u in a directed graph G if and only if G contains a directed walk from u to v.
11. A directed graph is **strongly connected** if every vertex is reachable from every other vertex.

References

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (n.d.). Introduction to algorithms (3rd ed.). The MIT Press.
2. Jeff Erickson. Algorithms (Prepublication draft). http://algorithms.wtf (http://algorithms.wtf)
3. Steven S. Skiena. 2008. The Algorithm Design Manual (2nd ed.). Springer Publishing Company, Incorporated.

graphs (/tags/graphs/)

Treaps (/Data-Structures/Tree/Treaps/)

Graph Representation: Adjacency List and Matrix (/Data-Structures/Graph/Graph-Representation-Adjacency-List-and-Matrix/)