GRAPHS

DATA STRUCTURES IN JAVA

Sercan Külcü | Data Structures In Java | 10.05.2023

Contents

Graphs 2
Directed Acyclic Graphs 3
Weighted Graphs 5
Multigraphs 7
Minimum Spanning Trees 9

Graphs

A graph is a data structure that represents a collection of objects, called vertices, and the relationships between them, called edges. Vertices are connected by edges, which can be directed or undirected. A directed edge is an edge that has direction, such as from vertex A to vertex B. An undirected edge does not have direction, such as between vertex A and vertex B.

Graphs can be used to model a wide variety of real-world objects and relationships, such as:

- Transportation networks, such as roads and railways, can be modeled as graphs, where vertices represent cities and edges represent roads or railways between cities.
- Social networks, such as Facebook or Twitter, can be modeled as graphs, where vertices represent people and edges represent friendships between people.
- Computer networks, such as the internet, can be modeled as graphs, where vertices represent computers and edges represent connections between computers.

Representing Graphs

There are two main ways to represent graphs:

- Adjacency lists represent graphs as a list of lists. Each list represents the vertices that are connected to a particular vertex.
- Adjacency matrices represent graphs as a matrix. Each entry in the matrix represents the connection between two vertices.

Operations on Graphs

There are several operations that can be performed on graphs, including:

- Searching is the process of finding a particular vertex in a graph.

- Traversing is the process of visiting all the vertices in a graph. There are a number of different ways to traverse a graph, including:
  - Breadth-first search visits all the vertices that are connected to a particular vertex before visiting any other vertices.
  - Depth-first search visits all the vertices that are connected to a particular vertex before visiting any vertices that are connected to those vertices.
- Adding vertices is the process of adding a new vertex to a graph.
- Deleting vertices is the process of removing a vertex from a graph.
- Adding edges is the process of adding a new edge to a graph.
- Deleting edges is the process of removing an edge from a graph.

Applications

Graphs are used in a wide variety of applications, including:

- Routing, such as finding the shortest path between two cities.
- Scheduling, such as finding a schedule for a set of tasks that minimizes the total amount of time required to complete the tasks.
- Social network analysis, such as finding influential people in a social network.
- Computer vision, such as finding objects in an image.
- Natural language processing, such as finding the relationships between words in a sentence.

Directed Acyclic Graphs

A directed acyclic graph (DAG) is a graph that does not contain any cycles. A cycle is a path that starts and ends at the same vertex. DAGs can be used to model a wide variety of real-world objects and relationships, such as:

- Computer programs, where vertices represent instructions and edges represent control flow.
- Manufacturing processes, where vertices represent steps in the process and edges represent the order in which the steps must be performed.
- Scheduling problems, where vertices represent tasks and edges represent dependencies between tasks.

Representing DAGs

There are two main ways to represent DAGs:

- Adjacency lists represent DAGs as a list of lists. Each list represents the vertices that are connected to a particular vertex.
- Adjacency matrices represent DAGs as a matrix. Each entry in the matrix represents the connection between two vertices.

Operations on DAGs

There are a number of operations that can be performed on DAGs, including:

- Topological sorting is the process of ordering the vertices in a DAG such that there are no directed edges from a vertex to a vertex that comes before it in the order.
- Finding the shortest path is the process of finding the path between two vertices that has the least number of edges.
- Finding the longest path is the process of finding the path between two vertices that has the greatest number of edges.
- Finding the critical path is the process of finding the longest path from the start vertex to the end vertex in a DAG.

Applications

DAGs are used in a wide variety of applications, including:

- Computer programming, where DAGs are used to represent the control flow of computer programs.

- Manufacturing processes, where DAGs are used to represent the steps in a manufacturing process.
- Scheduling problems, where DAGs are used to represent tasks and dependencies between tasks.
- Network analysis, where DAGs are used to represent networks of computers or other devices.
- Theorizing about algorithms, where DAGs are used to represent problems that can be solved using algorithms.

Weighted Graphs

A weighted graph is a graph where edges have a weight associated with them. The weight can represent the distance between two vertices, the cost of traveling between two vertices, or any other value.

Weighted graphs can be used to model a wide variety of real-world objects and relationships, such as:

- Transportation networks, such as roads and railways, can be modeled as weighted graphs, where vertices represent cities and edges represent roads or railways between cities. The weight of the edge can represent the distance between the cities.
- Social networks, such as Facebook or Twitter, can be modeled as weighted graphs, where vertices represent people and edges represent friendships between people. The weight of the edge can represent the strength of the friendship.
- Computer networks, such as the internet, can be modeled as weighted graphs, where vertices represent computers and edges represent connections between computers. The weight of the edge can represent the bandwidth of the connection.

Representing Weighted Graphs

There are two main ways to represent weighted graphs:

- Adjacency lists represent weighted graphs as a list of lists. Each list represents the vertices that are connected to a particular vertex. The weight of the edge is stored in the list.
- Adjacency matrices represent weighted graphs as a matrix. Each entry in the matrix represents the connection between two vertices. The weight of the edge is stored in the entry.

Operations on Weighted Graphs

There are a number of operations that can be performed on weighted graphs, including:

- Finding the shortest path is the process of finding the path between two vertices that has the least weight.
- Finding the longest path is the process of finding the path between two vertices that has the greatest weight.
- Finding the minimum spanning tree is the process of finding a subset of the edges in a graph that connects all the vertices and has the least total weight.
- Finding the maximum flow is the process of finding the maximum amount of flow that can be sent from one vertex to another vertex in a graph.

Applications

Weighted graphs are used in a wide variety of applications, including:

- Routing, such as finding the shortest path between two cities.
- Scheduling, such as finding a schedule for a set of tasks that minimizes the total amount of time required to complete the tasks.
- Social network analysis, such as finding influential people in a social network.
- Computer vision, such as finding objects in an image.
- Natural language processing, such as finding the relationships between words in a sentence.

Multigraphs

A multigraph is a graph that allows multiple edges to exist between two vertices. This means that there can be more than one path between two vertices in a multigraph.

Multigraphs can be used to model a wide variety of real-world objects and relationships, such as:

- Transportation networks, such as roads and railways, can be modeled as multigraphs, where vertices represent cities and edges represent roads or railways between cities. There can be multiple roads or railways between two cities, representing different routes that can be taken.
- Social networks, such as Facebook or Twitter, can be modeled as multigraphs, where vertices represent people and edges represent friendships between people. There can be multiple friendships between two people, representing different ways that the people are connected.
- Computer networks, such as the internet, can be modeled as multigraphs, where vertices represent computers and edges represent connections between computers. There can be multiple connections between two computers, representing different ways that the computers can be connected.

Representing Multigraphs

There are two main ways to represent multigraphs:

- Adjacency lists represent multigraphs as a list of lists. Each list represents the vertices that are connected to a particular vertex. The number of times a vertex appears in the list represents the number of edges between the vertex and the current vertex.
- Adjacency matrices represent multigraphs as a matrix. Each entry in the matrix represents the connection between two vertices. The

number of times a value appears in the entry represents the number of edges between the two vertices.

Operations on Multigraphs

There are a number of operations that can be performed on multigraphs, including:

- Finding the shortest path is the process of finding the path between two vertices that has the least number of edges.
- Finding the longest path is the process of finding the path between two vertices that has the greatest number of edges.
- Finding the minimum spanning tree is the process of finding a subset of the edges in a graph that connects all the vertices and has the least total weight.
- Finding the maximum flow is the process of finding the maximum amount of flow that can be sent from one vertex to another vertex in a graph.

Applications

Multigraphs are used in a wide variety of applications, including:

- Routing, such as finding the shortest path between two cities.
- Scheduling, such as finding a schedule for a set of tasks that minimizes the total amount of time required to complete the tasks.
- Social network analysis, such as finding influential people in a social network.
- Computer vision, such as finding objects in an image.
- Natural language processing, such as finding the relationships between words in a sentence.

Minimum Spanning Trees

A minimum spanning tree (MST) is a subset of the edges in a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight. That is, it is a spanning tree whose sum of edge weights is as small as possible. More generally, any edge-weighted undirected graph (not necessarily connected) has a minimum spanning forest, which is a union of the minimum spanning trees for its connected components.

The minimum spanning tree of a graph can be found using several algorithms, including:

- Kruskal's algorithm
- Prim's algorithm
- Boruvka's algorithm

Kruskal's algorithm works by iteratively adding edges to the MST, starting with the edge with the smallest weight. Each edge is added to the MST only if it does not create a cycle. Prim's algorithm works by iteratively adding edges to the MST, starting with any vertex. Each edge is added to the MST only if it connects the current vertex to a vertex that is not already in the MST. Boruvka's algorithm works by iteratively merging MSTs, starting with each vertex as its own MST. Each MST is merged with the MST that has the smallest edge connecting them.

Applications

Minimum spanning trees are used in a wide variety of applications, including:

- Routing, such as finding the shortest path between two cities.
- Scheduling, such as finding a schedule for a set of tasks that minimizes the total amount of time required to complete the tasks.
- Network design, such as finding the cheapest way to connect a set of points.

- Electrical engineering, such as finding the cheapest way to connect a set of power plants to a set of loads.
- Computer vision, such as finding the edges in an image.
- Natural language processing, such as finding the relationships between words in a sentence.