Trees

July 19, 2024

In computer science, trees are usually represented upside down. Trees can be used to represent an organization's structure showing the relationship between its different departments and their respective ranks and positions. Trees can also be used to represent file structures making it easy to locate the contents of the file. Trees can be used to represent a version control system where the repository of the project is stored on a centralized server with numerous programmers working on the copies of the project on their workstations.

Trees are specifically useful in computer science for their application in a wide range of algorithms.

- Trees can be used to find the shortest path in a graph.
- Trees can be used in games such as checkers and chess and can help determine winning strategies.
- Trees can be used to model procedures carried out using a sequence of decisions.
- The concept of a binary tree is a fundamental data structure in high-level programming.

1 Definition of a tree

Outlines

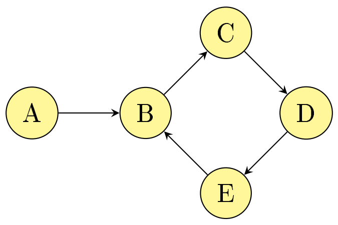
- Acyclic graphs
- Definition of a tree
- Definition of a forest
- Theorems on trees
- Definition of rooted trees

1.1 Acyclic graphs

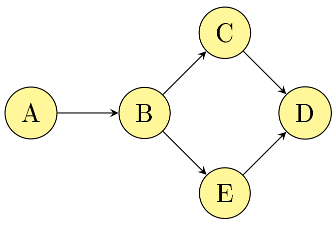
Definition

A graph G is called an acyclic graph if and only if G has no cycles.

G_1



G_2

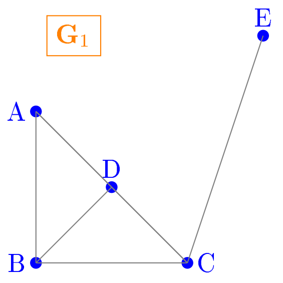


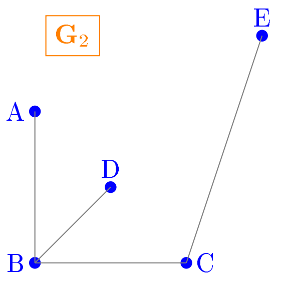
The directed graph G1 above, contains one cycle BCDEB, hence it is not an acyclic graph whereas G2 does not have any cycle in it. Thus G2 is an acyclic graph.

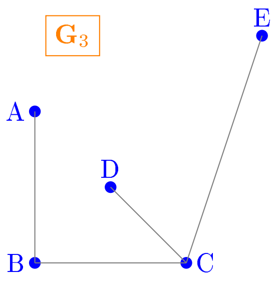
1.2 Definition of a tree

A tree is a connected acyclic undirected graph

An undirected graph G is a tree if and only if it is connected and acyclic. This means that there exists a path between any two certices of G and G is cycle-free. Hence, a tree can have neither loops nor multiple edges (parallel edges).



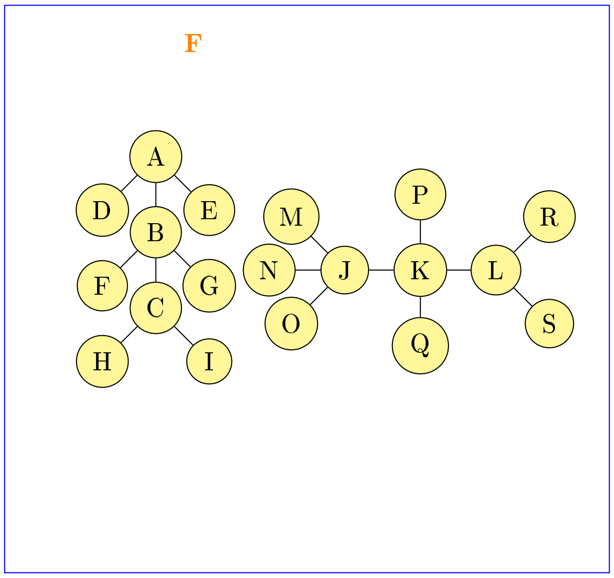




The graph G1 above is a tree since it is connected but contains a cycle ABDA. The graph G2 is a tree since it is connected and has no cycles. The graph G3 is also a tree since it is connected cycle free.

1.3 Definition of a forest

A forest is a collection of disjoint trees. A forest in nature contains many trees. A cycle-free disconnected graph is called a forest. The graph F below is a forest as it is disconnected and has no cycles.

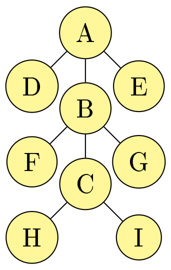


1.4 Theorems on trees

1.4.1 Theorem 1

An undirected graph is a tree if and only if there is unique and simple path betweeen any two vertices of the graph.

Proof by contradiction (also known as absurdum): Suppose there are two distinct paths between two vertices of a tree. Then there must be a cycle in the tree which contradicts the definition of a tree. Hence, there is a unique path between any two vertices of a tree.



Example: Given a graph G, let's assume that G is a tree, then G is connected and has no cycles. Let B and I be two vertices in G. We need to show there is a unique path between B and I. Let P_1 be a path from B to I. Let's assume there is another path P_2 between B and I. Hence we can combine the first part from B to I with the part from I to B obtained by reversing the order of the second path. The result will form a cycle which is a contradiction of the hyptothesis as the graph is a tree. Hence there exists a unique single path.

1.4.2 Theorem 2

A tree with n vertices has n-1 edges.

1.5 Definition of rooted trees

A rooted tree is a tree in which one vertex is designated as the root and every edge is directed away from the root.

2 Spanning trees of a graph

Outlines

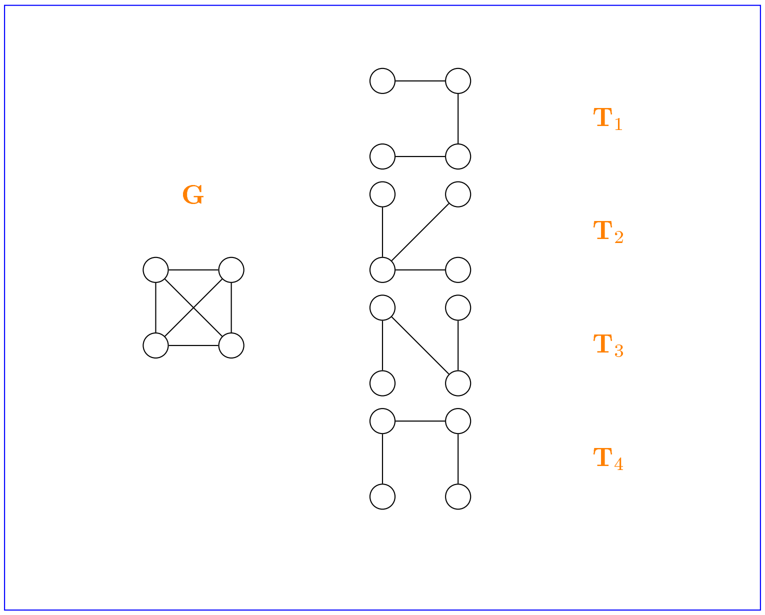
- Spanning trees of a graph
- Constructing a spanning tree
- Non-isomorphic spanning trees of a graph

2.1 Spanning trees of a graph

In many real life problems such as internet multicasting and network routing, it is important to find a tree that connects all the vertices of a graph. This tree is called a spanning tree.

2.1.1 Definition

A spanning tree of a graph G is a connected subgraph of G that contains all the vertices of G but carries no cycles.



The graph G above is a connected graph with no cycles. The graphs T_1, T_2, T_3 and T_4 are spanning trees of G.

2.2 Constructing a spanning tree

2.2.1 To get a spanning tree of a graph G

- Keep all vertices of G
- Break all the cycles but keep the tree connected

2.3 Non-isomorphic spanning trees of a graph

2.3.1 isomorphic Spanning trees

Two spanning trees are said to be isomorphic if there is a bijection preserving adjacency between the two trees.

Some of the spanning trees of a graph may be isomorphic which means they are the same. However, if we are asked to find all the spanning trees of a graph, we are only intersted in the non-isomorphic spanning trees. In the last example, the spanning trees T_4, T_3 and T_1 are isomorphic to each other, however, T_4, T_3 and T_1 are all non-isomorphic to T_2. So if we are asked to draw non-iomorphic trees of graph G, then we only need to draw T_2 and one of the rest of the the three non-isomorphic trees.

3 Minimum spanning tree

Outlines

- Example of a use
- Weight of a spanning tree
- Minimum spanning Trees
- Kruskal's algorithm
- Prim's algorithm

Example of a use case

Suppose we want to supply a set of houses with:

- electric power
- water pipes
- sewage lines
- telephone lines

To keep costs down, you could connect these houses with a spanning tree(power lines, for example). However, the houses are not all equal distances apart.

To reduce costs even further, you could connect the houses with a minimum-cost spanning tree.

Suppose you have a connected undirected graph with a weight (or cost) associated with each edge.

The cost of a spanning tree would be the sum of the cost of its edges.

3.1 Minimum cost spanning tree

A minimmum cost spanning tree is a spanning tree that has the lowest weight (lowest cost).

3.2 Finding spanning trees

There are two basic algortihms for finding minimum-cost spanning trees, and both are greedy algorithms:

- Kruskal's algorithm
- Prim's algorithm

3.3 Kruskal's algorithm

Start with the cheapest edges in the spanning tree

Repeatedly add the cheapest edge that does not create a cycle

3.4 Prim's algorithm

Start with any node in the spanning tree

Repeatedly add the cheapest edge, and the node it leads to, for which the node is not already in the spanning tree.