More on Graphs

Minimum Spanning Tree and Dijkstra's Algorithm

Giulia Carocari, Giacomo Fabris, Francesco Lotito

Università degli Studi di Trento

ICPC Training @ UniTN
Day 3 - March 27, 2019

Table of Contents

Table of Contents

A - Counting Stars

Problem definition

- The sky is a grid of $N \cdot M$ cells, each containing a piece of sky ('#') or star ('-').
- A star is a section of the sky made of one or more adjacent (up/down or left/right) starry pixels.
- How many stars are there in this portion of sky?

Solution

Count connected components on a graph where adjacent star cells in the grid are neighbouring nodes.

Remarkable applications of such visits on graphs: You may want to consider ASD 2018/01/31 (exercise 3).

B - Dominos

Problem definition

- Consider a field covered in domino tiles
- We are given precedence rules on what piece makes what other peace fall.
- What is the minimum number of times we have to knock over a piece in order to make them all fall?

Solution

Find strongly connected components (Kosaraju or Tarjan) and count those that do not have any entering edge from other SCC.

C - Erdos Numbers

Problem definition

Find the Erdos number of the listed authors, given their set of co-authors. The Erdos number of a person is the minimum length of a co-authoring chain that connects him/her to Paul Erdos.

Solution

Run a BFS on the input graph, assigning to each author his distance from the starting node (PAUL_ERDOS).

Programming tips: Implement the graph as a hash map (`unordered_map` in `C++`), where the ID of a node is the author's name. The name of each node in the graph is hashed to a vector of adjacent people.

D - Arctic Network

We will speak about this problem and its solution in a few slides.

E - Family DAG

As no one solved it, we will leave you one more week to work on it.

F - Getting Gold

Problem definition

Given the map of a dungeon, featuring walls (#), floor tiles (.), gold (G) and traps (T), find the maximum amount of gold a person can collect without ever risking to fall into a trap.
NOTE: The player can detect traps if they are in an adjacent cell, but he will not know in what cells they are located.

Solution

If the player detects a trap, the only safe move he can perform is to go back to where he came from. To find all places safely reachable, start a flood fill from the initial position (P), ending the visit whenever the player hits a wall or detects a trap. Keep a gold counter to track the loot you collected.

Table of Contents

Minimum Spanning Tree

Problem definition

Given a graph $G = (V, E)$, with a weight function $w : E \to \mathbb{R}^+$, find a subset E' of E such that (V, E') is a **tree** and

$$W = \sum_{e' \in E'} w(e')$$

is minimal.

Greedy choice theorem.

- Base case: Every 1-vertex graph is a minimum spanning tree.
- Inductive step: Given a MST $(V', E') \subset (V, E)$, we can connect $v \in V \setminus V'$ by adding to E' the edge $e = (v, v')$ such that $w(e)$ is minimal with respect to all edges connecting v to a node in E'.

Two greedy approaches

Kruskal's Algorithm

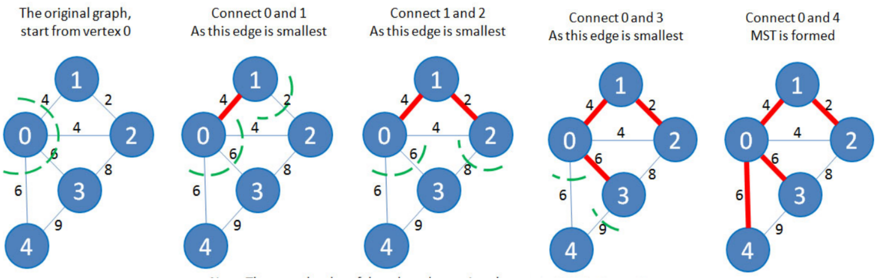
Keep all edges in a `MinPriorityQueue` to add them in incresing order to the minimum spanning tree.
Use Merge-Find Sets to detect cycles when adding edges to the MST: when adding an edge to the MST perform a merge of the two sets containing the endpoints of the edge itself. (if the two endpoints of the edge are in the same set, we will get a cycle).

Prim's Algorithm

Start from a random root of the minimum spaning tree and add the minimum weight edge that departs from the current MST.

Prim's Algorithm

- Select a root, add all its edges in a `PriorityQueue` sorted by increasing weight
- Pop from the queue, add edge to the MST if it connects a vertex which has not been connected yet


The original graph, start from vertex 0
Connect 0 and 1 As this edge is smallest
Connect 1 and 2 As this edge is smallest
Connect 0 and 3 As this edge is smallest
Connect 0 and 4 MST is formed
0
1
2
3
4
4
2
4
6
8
6
9

Arctic Network

Problem definition

Given a set of N outposts in the Arctic and a number of S available satellite channels, we want to compute the maximum power D (aka distance range) required by the radio transceivers that have to connect every outpost, directly or indirectly. NOTE: if an outpost is equipped with a satellite channel it does not need a transceiver.

Solution

Run Prim/Kruskal on the complete graph of the outposts (weight is given by the distance of every pair of nodes). After you have obtained the MST, sort its edges by increasing weight and return the edge with index $N - S - 2$.

Table of Contents

Dijkstra's Algorithm

Problem definition

Given a graph $G = (V, E)$, with a weight function $w : E \to \mathbb{R}^+$, find the length of the shortest path from a source s to all other nodes in V.

NOTE: Dijkstra's Algorithm works well only with positive weights on edges, for some reasons that you will see in class. Other algorithms (e.g. Bellman-Ford) have slight less strict constraints (the graph cannot contain negative-weight cycles), but we shall discuss them another time.

Dijkstra's Algorithm

Enough talking... Let's get to the code!