

Design and Analysis of Algorithms: Homework #6

Due in class on April 24, 2018

Professor Kasturi Varadarajan

Alic Szecsei

Problem 1

For any flow network G and any vertices u and v let $bottleneck_G(u, v)$ denote the maximum, over all paths π in G from u to v , of the minimum-capacity edge along π .

- (a) Describe and analyze an algorithm to compute $bottleneck_G(s, t)$ in $O(E \log V)$ time.

Solution**Part A**

Solution

Problem 2

Suppose you have already computed a maximum flow f^* in a flow network G with *integer* edge capacities.

- (a) Describe and analyze an algorithm to update the maximum flow after the capacity of a single edge is increased by 1.
- (b) Describe and analyze an algorithm to update the maximum flow after the capacity of a single edge is decreased by 1.

Both algorithms should be significantly faster than recomputing the maximum flow from scratch.

Solution**Part A**

Solution

Part B

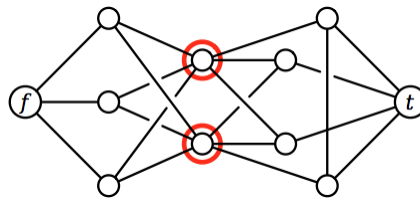
Solution

Problem 3

The Island of Sodor is home to a large number of towns and villages, connected by an extensive rail network. Recently, several cases of a deadly contagious disease (either swine flu or zombies; reports are unclear) have been reported in the village of Ffarquhar. The controller of the Sodor railway plans to close down certain railway stations to prevent the disease from spreading to Tidmouth, his home town. No trains can pass through a closed station. To minimize expense (and public notice), he wants to close down as few station as possible. However, he cannot close the Ffarquhar station, because that would expose him to the disease, and he cannot close the Tidmouth station, because then he couldn't visit his favorite pub.

Describe and analyze an algorithm to find the minimum number of stations that must be closed to block all rail travel from Ffarquhar to Tidmouth. The Sodor rail network is represented by an undirected graph, with a vertex for each station and an edge for each rail connection between two stations. Two special vertices f and t represent the stations in Ffarquhar and Tidmouth.

For example, given the following input graph, your algorithm should return the number 2.



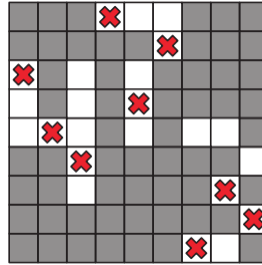
Solution

Solution

Problem 4

Suppose we are given an $n \times n$ square grid, some of whose squares are colored black and the rest white. Describe and analyze an algorithm to determine whether tokens can be placed on the grid so that

1. every token is on a white square;
2. every row of the grid contains exactly one token; and
3. every column of the grid contains exactly one token



Your input is a two dimensional array $IsWhite[1..n, 1..n]$ of booleans, indicated which squares are white. Your output is a single boolean. For example, given the grid above as input, your algorithm should return **TRUE**.

Solution

Solution

Problem 5

Ad-hoc networks are made up of low-powered wireless devices. In principle, these networks can be used on battlefields, in regions that have recently suffered from natural disasters, and in other hard-to-reach areas. The idea is that a large collection of cheap, simple devices could be distributed through the area of interest (for example, by dropping them from an airplane); the devices would then automatically configure themselves into a functioning wireless network.

These devices can communicate only within a limited range. We assume all the devices are identical; there is a distance D such that two devices can communicate if and only if the distance between them is at most D .

We would like our ad-hoc network to be reliable, but because the devices are cheap and low-powered, they frequently fail. If a device detects that it is likely to fail, it should transmit its information to some other *backup* device within its communication range. We require each device x to have k potential backup devices, all within distance D of x ; we call these k devices the **backup set** of x . Also, we do not want any device to be in the backup set of too many other devices; otherwise, a single failure might affect a large fraction of the network.

So suppose we are given the communication radius D , parameters b and k , and an array $d[1..n, 1..n]$ of distances, where $d[i, j]$ is the distance between device i and device j . Describe an algorithm that either computes a backup set of size k for each of the n devices, such that no device appears in more than b backup sets, or reports (correctly) that no good collection of backup sets exists.

Solution

Solution