Lecture 24

CSE 331

Oct 27, 2017

And back to our HW schedule..

Homework 6

Due by **11:00am, Friday, November 3, 2017**.

Make sure you follow all the homework policies.

All submissions should be done via Autolab.

Question 1 (Programming Assignment) [40 points]

Note

The text for Q1 is from last year. `Katie` : It needs to be updated. I did add the blurb about bonus points for fastest submissions though.

</> Note

This assignment can be solved in either Java, Python or C++ (you should pick the language you are most comfortable with). Please make sure to look at the supporting documentation and files for the language of your choosing.

Dijkstra's shortest path algorithm

$d'(v) = \min_{e=(u,v) \text{ in } E,\ u \text{ in } R} d(u)+l_e$

Input: Directed $G=(V,E)$, $l_e \geq 0$, s in V

$R = \{s\}$, $d(s) = 0$

While there is a v not in R with (u,v) in E, u in R

Pick w that minimizes $d'(w)$

Add w to S

$d(w) = d'(w)$

At most n iterations

$O(m)$ time

$O(mn)$ time bound is trivial

$O(m \log n)$ time implementation with priority Q

Reading Assignment

Sec 4.4 of [KT]

Building a fiber network

Lay down fibers to connect n locations

All n locations should be connected

Laying down a fiber costs money

What is the cheapest way to lay down the fibers?

Today's agenda

Minimum Spanning Tree (MST) Problem

Greedy algorithm(s) for MST problem

Kruskal's Algorithm

Joseph B. Kruskal

Input: $G=(V,E)$, $c_e > 0$ for every e in E

$T = \emptyset$

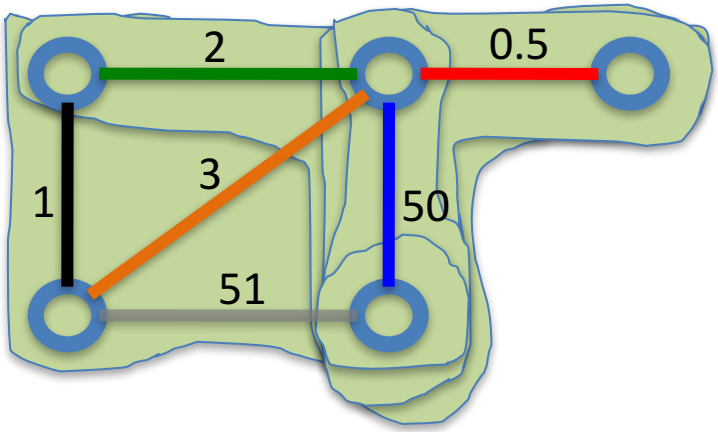
Sort edges in increasing order of their cost

Consider edges in sorted order

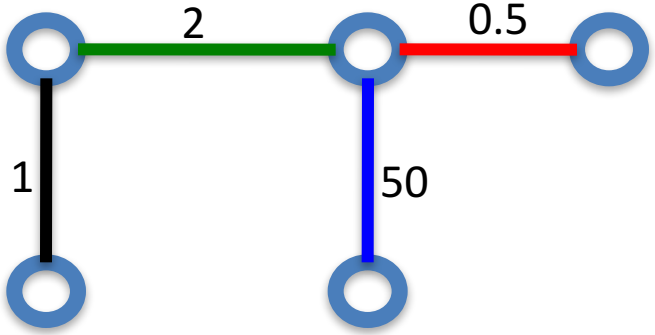
If an edge can be added to T without adding a cycle then add it to T

Prim's algorithm

Similar to Dijkstra's algorithm



Robert Prim



Input: G=(V,E), $c_e > 0$ for every e in E

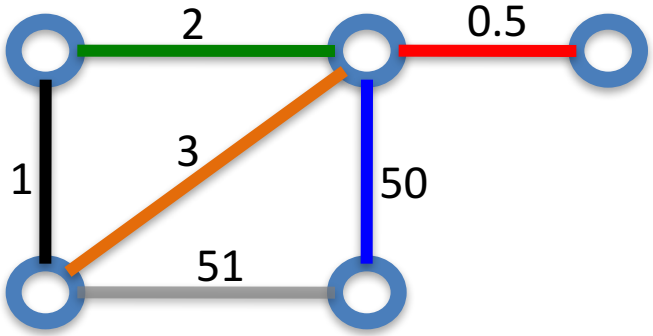
S = {s}, T = Ø

While S is not the same as V

Among edges e= (u,w) with u in S and w not in S, pick one with minimum cost

Add w to S, e to T

Reverse-Delete Algorithm



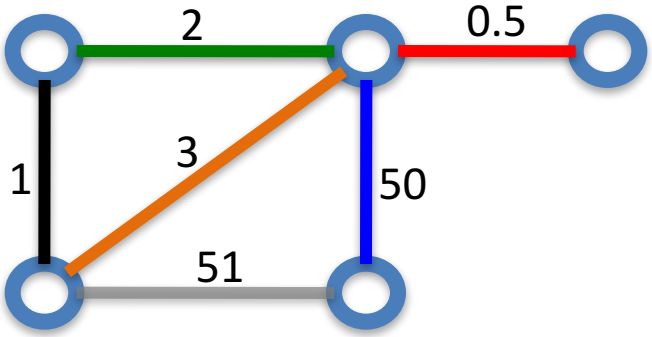
Input: G=(V,E), $c_e > 0$ for every e in E

T = E

Sort edges in decreasing order of their cost

Consider edges in sorted order

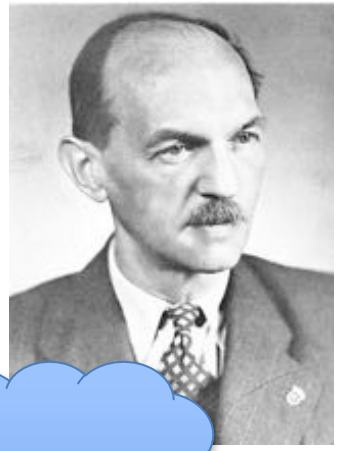
If an edge can be removed T without disconnecting T then remove it



(Old) History of MST algorithms

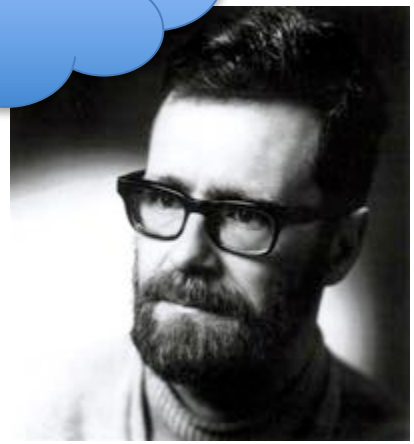
1920: Otakar Borůvka

1930: Vojtěch Jarník



Same algo!

1956: Kruskal



1957: Prim

1959: Dijkstra