Lecture 18

CSE 331

Oct 7, 2019

Quiz starts at 1pm and ends at 1:10pm

Lecture starts
at 1:15pm

Problem 1 on Coding project

note

stop following 30 views

Problem 1 on Coding Project is now live!

Apologies again for the delay in getting this done but Autolab is now accepting submissions for Problem 1 for the coding project. The coding project webpage has been updated with the required coding details:

http://www-student.cse.buffalo.edu/~atri/cse331/fall19/coding-project/index.html

Few things to keep in mind:

- This is group submission-- please see the webpage for instructions on how to do so. *Please follow the instructions EXACTLY. Not following the instructions might make the group submission on Autolab not behave as intended.*
- Problem 1 is **easy ONCE** you have understood what the problem is asking and you familiarize yourself with the template structure. *You literally need to add a couple of lines of code to get FULL points.*
- **If you are using C++**, please note that the provided code is a bit slow. Running the correct Solution takes a 1min to 1.5 mins, which is definitely slow. The issue is at our end and we are working on fixing it.
  - You can definitely submit your solutions in the meantime (it'll just take a bit to run).
  - We'll put out an update once this is fixed (this change will not affect your code at all).

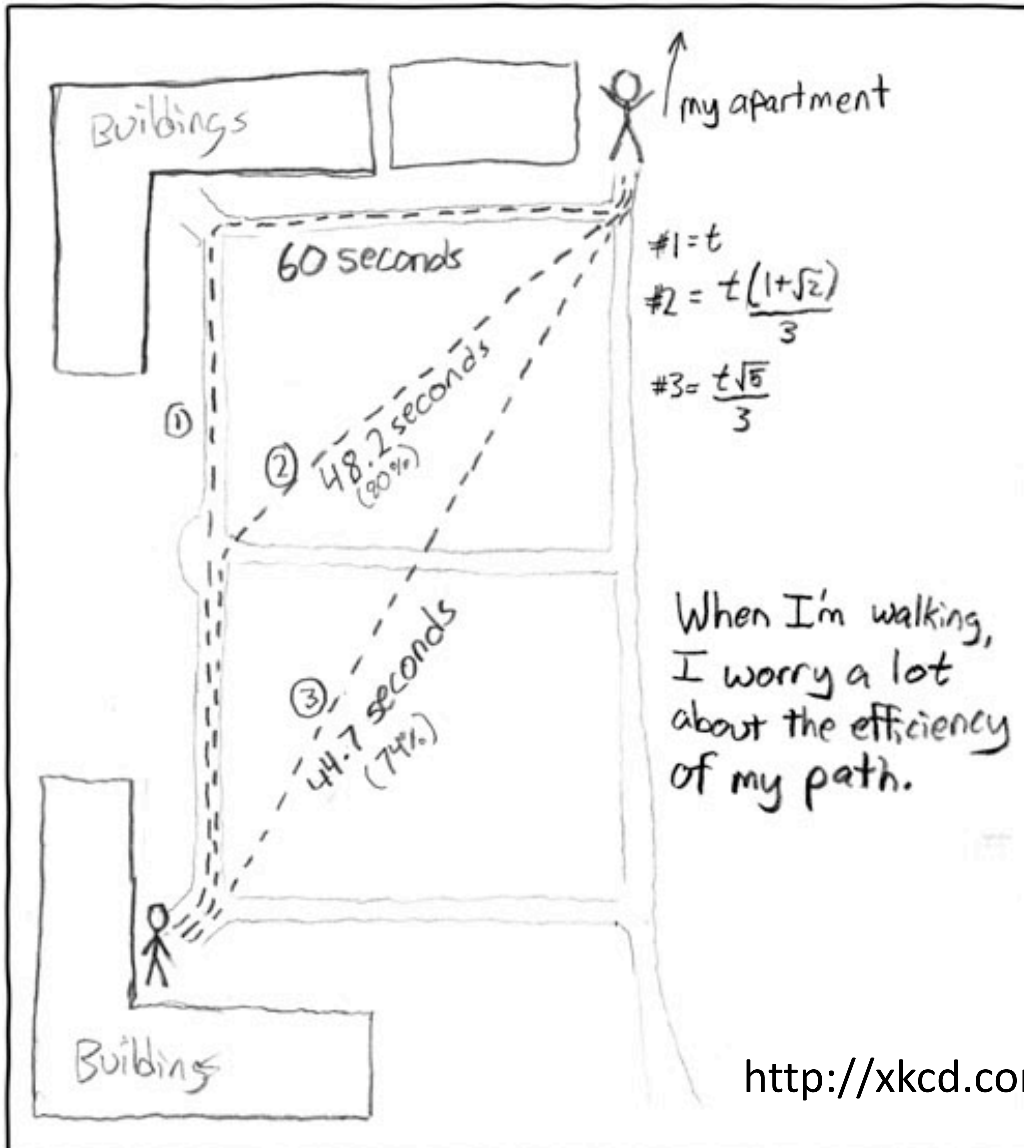
#pin

coding_mini_project

edit · good note | 0

Updated 41 minutes ago by Atri Rudra

Shortest Path Problem



http://xkcd.com/85/

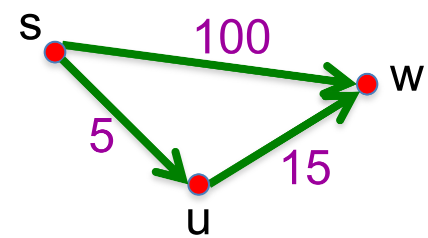
Another more important application

Is BGP a known acronym for you?

Routing uses shortest path algorithm

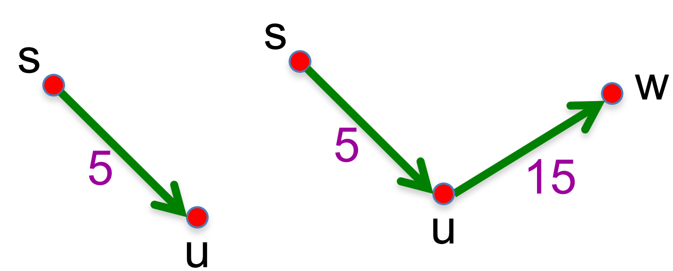
Shortest Path problem

Input: *Directed* graph $G=(V,E)$



Edge lengths, l_e for e in E

“start” vertex s in V



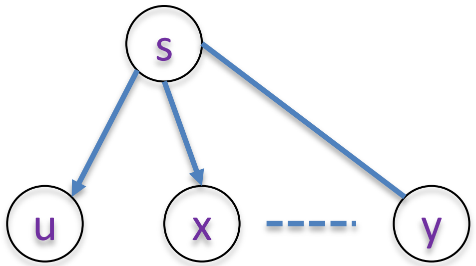
Output: Length of shortest paths from s to all nodes in V

Dijkstra’s shortest path algorithm

Towards Dijkstra's algo: part ek

Determine d(t) one by one

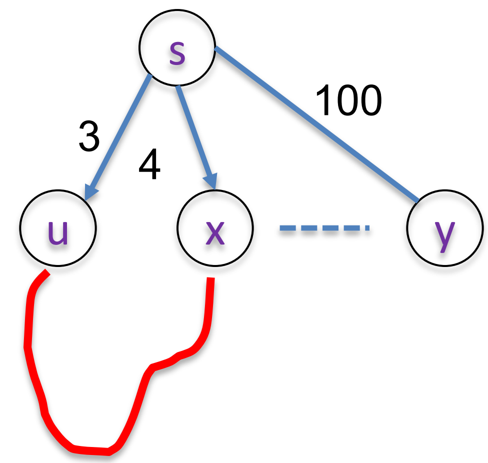
$$\boxed{d(s) = 0}$$



Towards Dijkstra's algo: part do

Determine $d(t)$ one by one

Let u be a neighbor of s with smallest $l_{(s,u)}$



$$\boxed{d(u) = l_{(s,u)}}$$

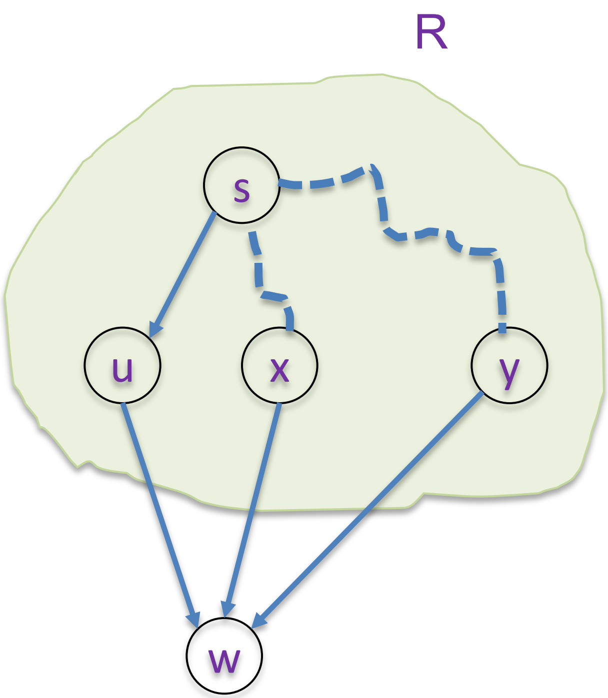
Length of is $\geq$ 0

Not making any claim
on other vertices

Towards Dijkstra's algo: part teen

Determine d(t) one by one

Assume we know d(v) for every v in R



Compute an upper bound d'(w) for every w not in R

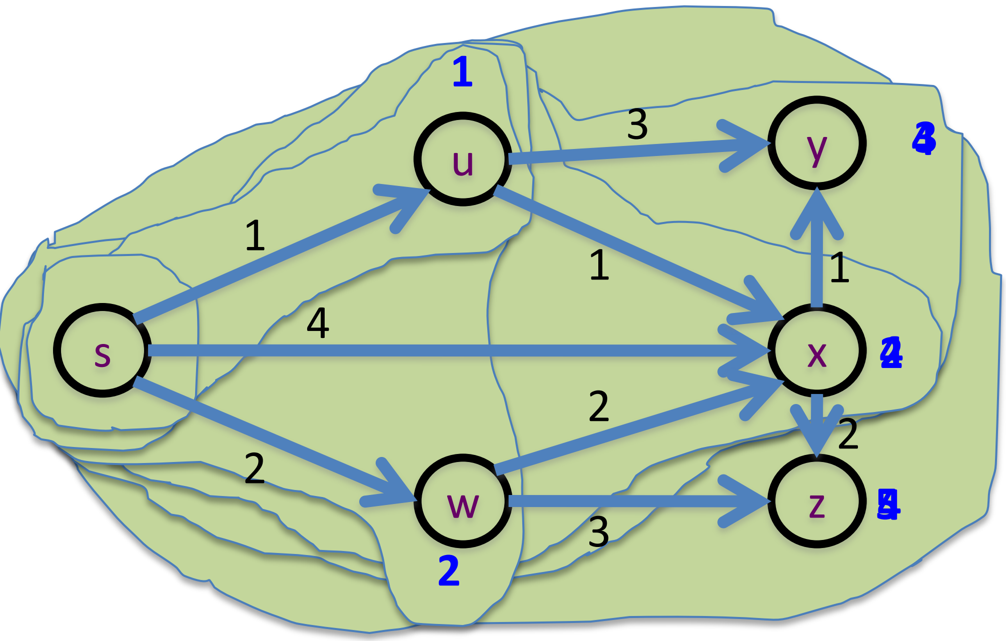
$$d(w) \leq d(u) + l_{(u,w)}$$

$$d(w) \leq d(x) + l_{(x,w)}$$

$$d(w) \leq d(y) + l_{(y,w)}$$

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

Dijkstra's shortest path algorithm

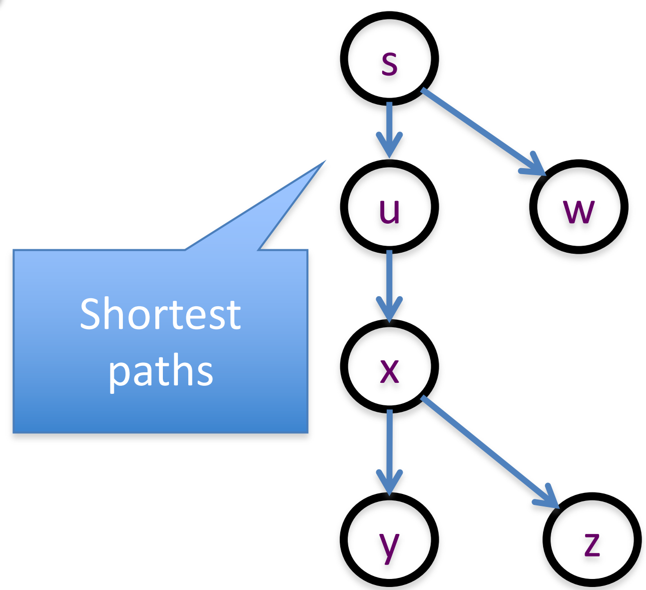


$d'(w) = \min_{e=(u,w) \text{ in } E,\ u \text{ in } R} d(u)+l_e$

$d(s) = 0$ $d(u) = 1$

$d(w) = 2$ $d(x) = 2$

$d(y) = 3$ $d(z) = 4$



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

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

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

- Pick w that minimizes $d'(w)$
- Add w to R
- $d(w) = d'(w)$

Couple of remarks

The Dijkstra's algo does not explicitly compute the shortest paths

Can maintain "shortest path tree" separately

Dijkstra's algorithm does not work with negative weights

Left as an exercise

Rest of Today's agenda

Prove the correctness of Dijkstra's Algorithm

Runtime analysis of Dijkstra's Algorithm