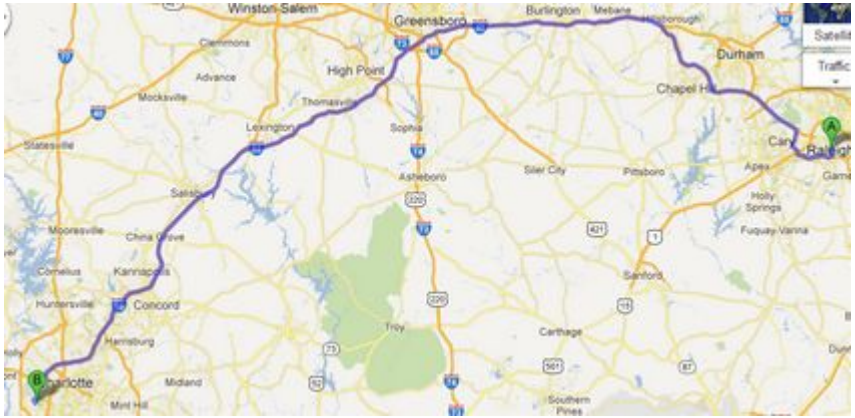
Shortest Paths

Week 3 AI Inspire Winter 2020

Problem

* Find shortest path between source **s** and sink **t** in edge-weighted digraph **G**
* No cycles



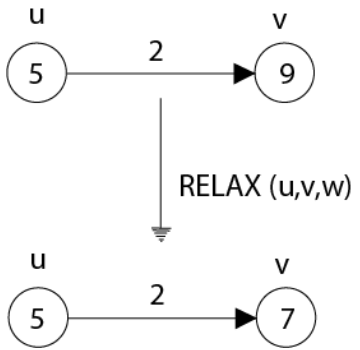
What does GPS use for shortest paths?

Data Structures

* distTo[ ] arr
* edgeTo[ ] arr
* Both will help find shortest path
  - Representing links ⇒ can trace back to source

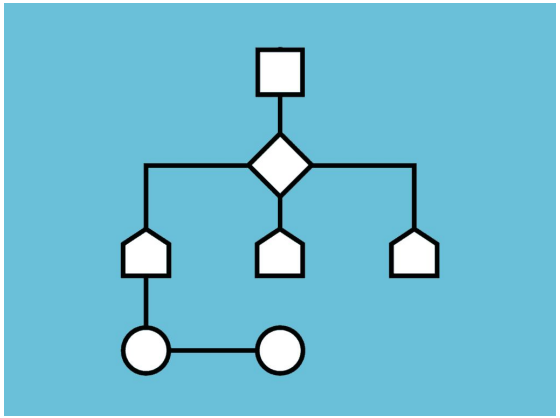
Edge Relaxation

* Given an edge **e** from **v** ⇒ **w**
* If **e** results in shorter dist to **w** ⇒ distTo[v] + e < distTo[w]
  * distTo[w] and edgeTo[w] will be changed so that we have min val



Edge Relaxations in Shortest Paths

* Used to get shortest paths ⇒ keep on updating shortest path
* # of edge relaxations and ordering are impo
* Two main algos
  - Bellman-ford
  - Dijkstra's



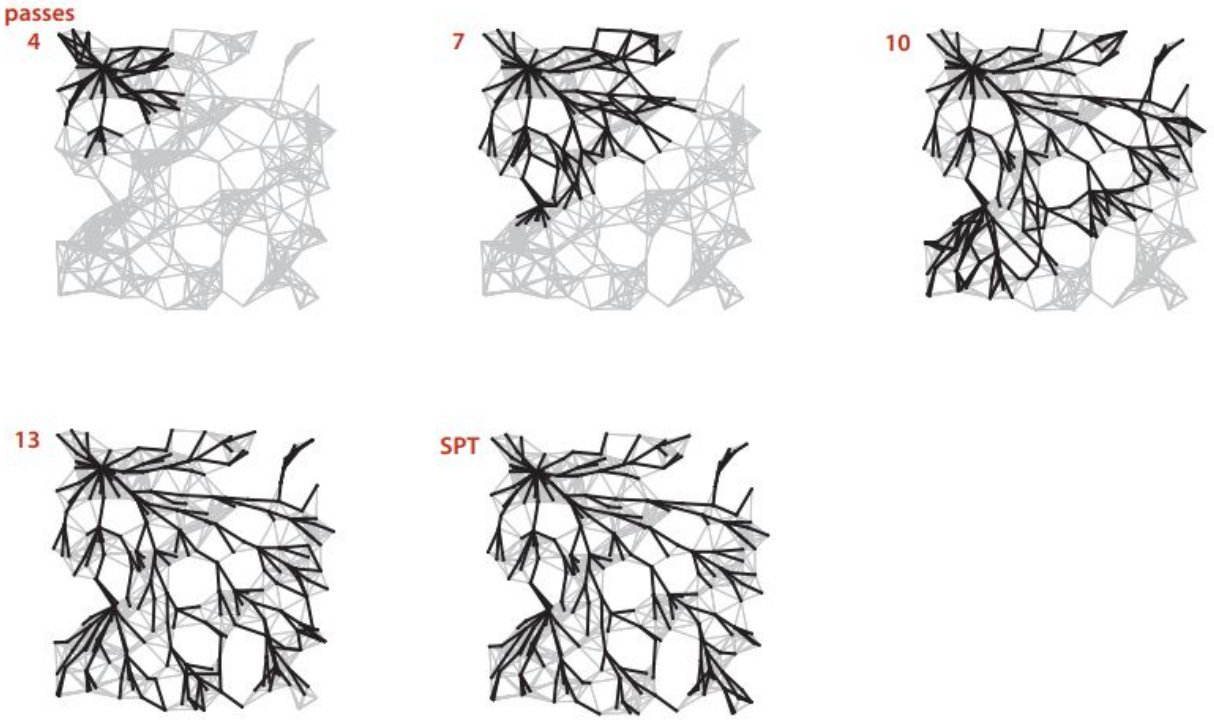
Bellman Ford Algorithm

Pseudocode

- Initialization stage
  - distTo[v] for all nodes
  - distTo[s] for source
  - edgeTo[v] for all nodes
- **Repeat (V-1) times ⇒ relax each edge**

Runtime analysis

* Best and worst case determined by type of input
* Best case ⇒ O(E * V)
* Worst case ⇒ O(E * V)



Credits = Princeton University (Fall 2019 COS 226)

How can we improve worst-case of Bellman Ford algorithm? Different optimizations to algorithm?

How can we use Bellman-Ford to find longest path?

Negative weights?
Negative cycles?

Dijkstra's Algorithm

About Edsger Dijkstra

* Very famous computer scientist
* His algo- Dijkstra's algo is very popular
  - Most efficient
  - Accurate
* Contributed to many other CS areas
  - Operating systems
  - Compiler construction
  - Distributed programming

Pseudocode

1) Look at distances of all nodes from the source **s** and explore them in ascending order of dist
    a) Look at distTo[ ]
2) Add node **u** to tree & relax all edges that touch that node **u**

Worst Case

* Worst case of Dijkstra's = E * log (V) b/c of its implementation
* Works with directed cycles but NOT negative weights
  * Bellman-ford ⇒ BOTH directed cycles and negative weights BUT NOT negative directed cycles


How would we implement Dijkstra's algorithm in Java? Data structures to use?


Quick Crash Course on Heaps and Min-heaps

How would we implement decreaseKey() operation of Dijkstra's algorithm using binary heap?

What algorithm we learned last class reminds you of Dijkstra's algorithm?

Video

Go through videos from previous sessions first!

https://www.youtube.com/watch?v=pVfj6mxhdMw

https://www.youtube.com/watch?v=qx9sJ3O3JM0