CS3000
6/7 – Weds

Admin

- Short 3 are tom. 9pm
- Long HW5 out tom. → last Long HW !! :)
- Fun optional recitation tom.

Agenda

1. SSSP
2. Dijkstra's Algorithm
3. Dijkstra's proof

1. SSSP

- Single source shortest paths
- given starting vertex s, find the shortest path to every reachable vertex

  $\delta(s, v)$ for every $v \in G.V$

- Breadth First Search is a SSSP algorithm
  - unweighted undirected graph
    → requirement for BFS

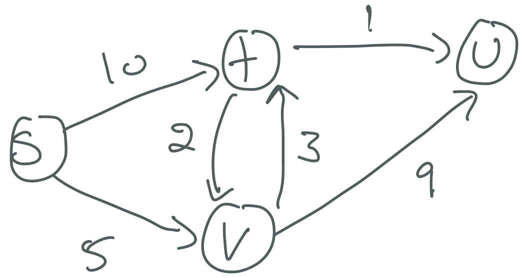
- Today: generalization
  - weighted graph   Func (G, w)
    → weight func.
  - directed
  - Adj. list

  → non-negative edge weights

  restriction, but a small one!

- SSSP algo: Dijkstra's
  (it is not the only one though!)

ex



- non-neg weights
- cycles are ok!

path from s to t

$\langle s,t \rangle$ is a valid path $w(s,t) = 10$

$\langle s,v,t \rangle$ is a valid path $w(\langle s,v,t \rangle) = 5+3=8$

but better!

Optimization problem

valid path with least total weight

Turning into an algorithm...

- greedy approach
- always want the lightest option at every step

$\boxed{\text{Dijkstra's works like Prim's}}$

- keep attributes for each vertex → greedy!
- can update attributes' values when we find a better option

on each vertex

$v.d$ ~~ distance of shortest path from s to v

$v.\pi$ ~~ predecessor vertex on shortest path

Initialization

$v.d = \infty$

$v.\pi = nil$

except the starting vertex, $s.d = 0$

Relaxation

- relax an edge (u, v)
- can we improve the distance to v by going through u?

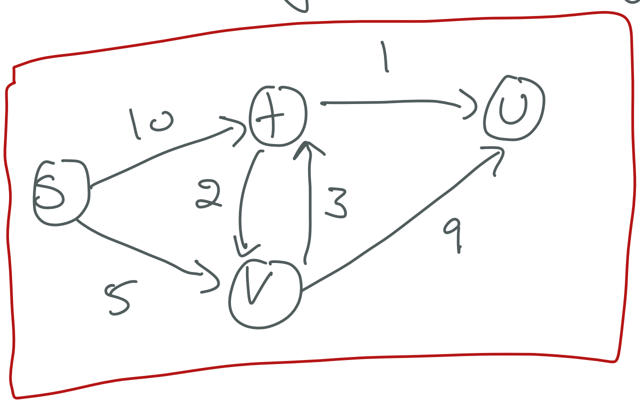
```
RELAX(u, v, w)
if v.d > u.d + w(u, v)
    v.d = u.d + w(u, v)
    v.π = u
```

Goal: $\boxed{v.d = \delta(s, v)}$

value assigned to $v.d$ is same as distance of actual shortest path

2. Dijkstra's Algorithm

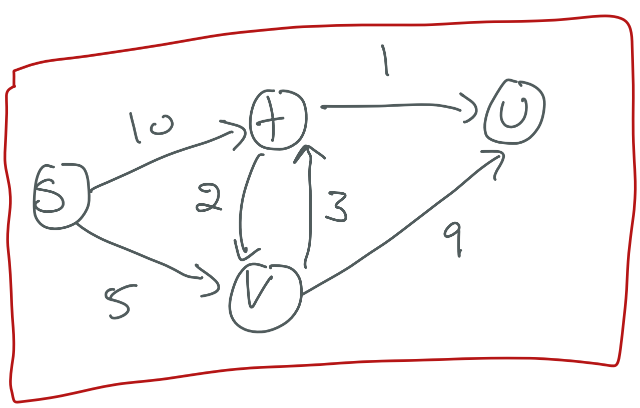
- SSSP
- directed
- weighted (non-neg weights)
- starting vertex
- goal: $v.d = \delta(s, v)$



Algo at high level

- maintain set S of vertices we're done with
- Heap, keyed on
  - v.d for each vertex
- Initialization step

  $\boxed{s.d = 0 \quad v.d = \infty \quad v.\pi = \text{nil}}$
- pull vertices out of heap, try to relax current vertex's neighbors
- after relaxing all neighbors, go in S



⟶ ex of Dijkstra's on this graph

Heap	vertex extracted	Set S	vertex attributes
s, t, u, v	s	$\{s\}$	$t.d = 10$ $t.\pi = s$ $v.d = 5$ $v.\pi = s$
t, u, v	v	$\{s, v\}$	$u.d = 14$ $u.\pi = v$ $t.d = 8$ $t.\pi = v$
t, u	t	$\{s, v, t\}$	$u.d = 9$ $u.\pi = t$ $v.d = 5$ $v.\pi = s$
u	u	$\{s, v, t, u\}$	

	s	u	v	t
d	0	9	5	8
π	nil	t	s	v

10:53

How do we know that when we add a vertex to S we never need to update it again?

3. Dijkstra's Proof

$v.d$ — distance assigned by algo

$\delta(s,v)$ — distance of actual shortest path

want: $v.d = \delta(s,v)$ for every reachable vertex from s

this is set before v is added to set S

#1 for any v, $v.d \geq \delta(s,v)$

#2 $S \xrightarrow{\text{path}} u \xrightarrow{\text{edge}} v$ actual shortest path

u,v gets relaxed

if $u.d = \delta(s,u)$, then $v.d = \delta(s,v)$ after relaxing

Proof by induction

↳ induction on $|S|$

Base case: $|S| = 1$

Source vertex $s \in S$

	algo	real	
	$s.d = 0$	$\delta(s,s) = 0$	✓

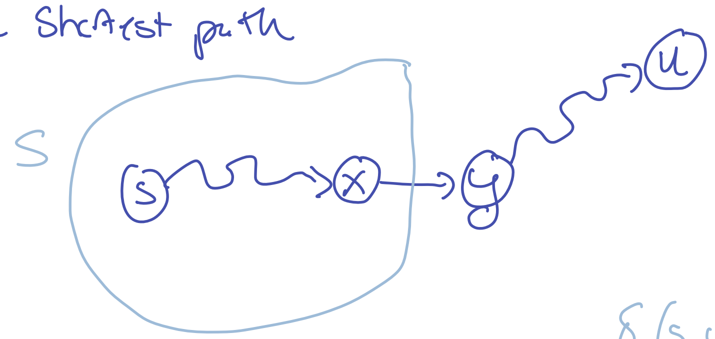
IH $v.d = \delta(s,v)$ for every $v \in S$ up to $|S| = k$

next, we add $k+1^{th}$ vertex! u is $k+1$

about to add u to S

u.d should already be fine

Actual shortest path



- $x \in S$
- (x, y) is an edge
- y precedes u on shortest path

$\delta(s, y) \le \delta(s, u)$

$x.d = \delta(s, x)$ IH

$u.d \le y.d$ because u got extracted first

$u.d \ge \delta(s, u)$ because of obs. #1

After relaxation... $y.d = \delta(s, y)$

all inequalities together

$$\delta(s,y) \le \delta(s,u) \le u.d \le y.d$$

But! $\boxed{\delta(s,y) = y.d}$

☆☆☆ :) done!

So... $\delta(s,y) = \boxed{\delta(s,u) = u.d} = y.d$

:) ♡ ♡

$5 \le a \le b \le 5$

$\hookrightarrow a = b = 5$!