CS 303, Paul Cao Lecture 27

Plan: Dijkstra's algorithm

- MST construction with Prim's and Kruska's algorithms

Topic today: Dijkstra's algorithm

1. Dijkstra's algorithm

It is a single-source shortest path algorithm with wide range of applications.
e.g.
Given two hosts, one at Ashland and one in USC, how can they communicate with each other through internet?

- Fantasy: every node (a host, or a LAN) is directly connected together.
- Reality: nodes are connected indirectly through a subnet of routers

Theory on shortest Path

- Path: a digraph $G=(V,E)$ w/ edge weight given by function $w:(u,v)\to\mathbb{R}$ where $(u,v)\in E$, a path $p = v_1 \to v_2, \cdots \to v_k$
- weight of a path p: $w(p)=\sum_{i=1}^{k} w(v_i, v_{i+1})$

e.g.

$$v_1 \xrightarrow{4} v_2 \xrightarrow{-2} v_3 \xrightarrow{-1} v_4 \xrightarrow{-3} v_5 \qquad W(p) = 4-2-1-3 = -2$$

- shortest path (may not be unique) from u to v:

  a path of minimum path weight from u to v. We denote this shortest <u>weight</u> as $\delta(u,v)$

$$\delta(u,v) = \min\{\, w(p) : \text{path } p \text{ from } u \text{ to } v \,\}$$

Note: Q:/ when do shortest path not exist?

A:/ • if we allow negative weights, then some shortest path may not exist.

why? if a negative-weight cycle exist bet u & v

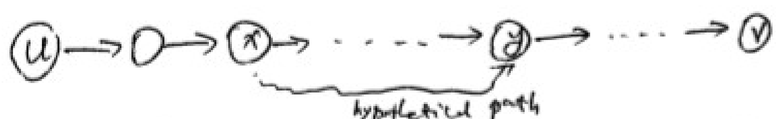
we may think it's $-\infty$.

• if there is no path from u to v, then it's ∞.

Q:/ If we wants to use DP to solve this problem, what's the key property?

A:/ optimal substructure.

i.e. A subpath of a shortest path is a shortest path.



Proof: by contradiction. the path $u \to v$ now is shortest.

if there is a subpath $x \to y$ that's not the shortest path from $x \to y$, then we can erase the path from $x \to y$ and $u \to v$ is now better than before. Contradiction. #

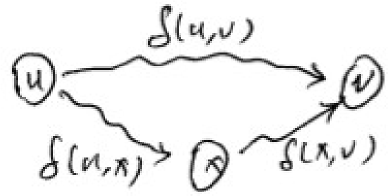
How to solve this problem?

Theorem: triangle inequality

for all vertices $u, v, x \in V$,

then $\delta(u,v) \le \delta(u,x) + \delta(x,v)$

Proof:



by definition of δ, it's proved.

this theorem is kind of the overlapping subproblem property.

One shortest path problem

Single-source shortest path problem

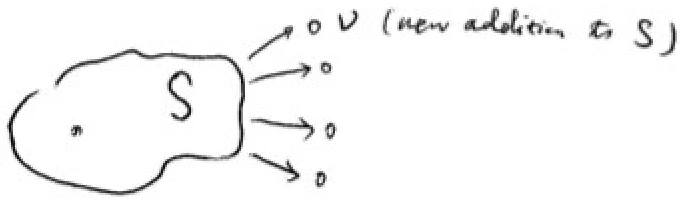
from a given source vertex $s \in V$, find shortest-path $\delta(s, u)$ for all $u \in V$

Assume $w(u, v) \geq 0$

Note: this problem is no harder than find $\delta(u, v)$ for just 2 vertices. :-)

Idea: greedy (means we won't go back and re-make our choice)

① maintain a set S of vertices whose shortest-path distance is known.



② at each step, add to S the vertex $v \in V - S$ whose estimated distance from s is min.

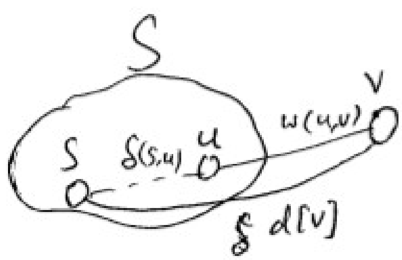
③ update distance estimate of vertices adjacent to v.

Dijkstra's algorithm

$d[s] \leftarrow 0$
for each vertex $v \in V - \{s\}$
  do $d[v] = \infty$
$S \leftarrow \phi$
$Q \leftarrow V$ (Q is a priority queue)
while $Q \neq \phi$
  do $u \leftarrow$ extract-min (Q)
    $S \leftarrow S \cup \{u\}$
    for each $v \in Adj[u]$ ← means outgoing from u to v
     do if $d[v] > d[u] + w(u,v)$
       $d[v] = d[u] + w(u,v)$

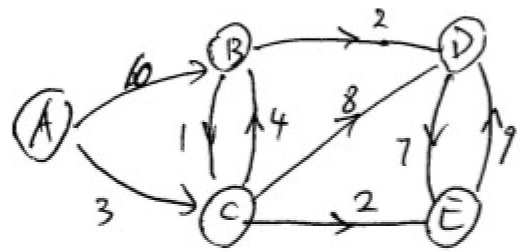
$d[x]$ = distance estimate from s to x → hopefully it's $\delta(s,x)$

$d[u]$ is hopefully $\delta(s,u)$, if $d[v]$



according to triangle inequality, $d[v]$ should be $<= \delta(s,u) + w(u,v)$
the last step in the algorithm basically fix the triangle inequality.

e.g.



Q: $\begin{matrix} A & B & C & D & E \\ 0 & \infty & \infty & \infty & \infty \end{matrix}$, A is source

① S: first $\begin{pmatrix} A \\ 0 \end{pmatrix}$ is added, Q: $\begin{matrix} B & C & D & E \\ 10 & 3 & \infty & \infty \end{matrix}$

② S: $\begin{pmatrix} A \\ 0 \end{pmatrix} \begin{pmatrix} C \\ 3 \end{pmatrix}$ Q: $\begin{matrix} B & D & E \\ 3+4=7 & 8+3=11 & 3+2=5 \end{matrix}$

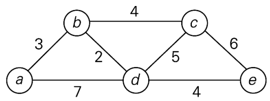
③ S: $\begin{pmatrix} A \\ 0 \end{pmatrix} \begin{pmatrix} C \\ 3 \end{pmatrix} \begin{pmatrix} E \\ 5 \end{pmatrix}$ Q: $\begin{matrix} B & D \\ 7 & 11 \end{matrix}$

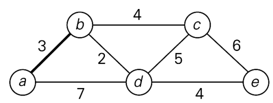
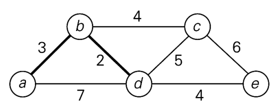
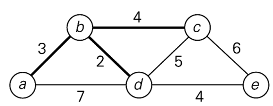
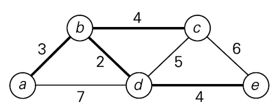
④ S: $\begin{pmatrix} A \\ 0 \end{pmatrix} \begin{pmatrix} C \\ 3 \end{pmatrix} \begin{pmatrix} E \\ 5 \end{pmatrix} \begin{pmatrix} B \\ 7 \end{pmatrix}$ Q: $\begin{matrix} D \\ 7+2=9 \end{matrix}$

⑤ S: $\begin{pmatrix} A \\ 0 \end{pmatrix} \begin{pmatrix} C \\ 3 \end{pmatrix} \begin{pmatrix} E \\ 5 \end{pmatrix} \begin{pmatrix} B \\ 7 \end{pmatrix} \begin{pmatrix} D \\ 9 \end{pmatrix}$

if we rember the decision, we can easily find the path too.

Exercise



Tree vertices	Remaining vertices	Illustration
a(−, 0)	**b(a, 3)** c(−, ∞) d(a, 7) e(−, ∞)	
b(a, 3)	c(b, 3 + 4) **d(b, 3 + 2)** e(−, ∞)	
d(b, 5)	**c(b, 7)** e(d, 5 + 4)	
c(b, 7)	**e(d, 9)**	
e(d, 9)		

The shortest paths (identified by following nonnumeric labels backward from a destination vertex in the left column to the source) and their lengths (given by numeric labels of the tree vertices) are

from a to b :	$a - b$	of length 3
from a to d :	$a - b - d$	of length 5
from a to c :	$a - b - c$	of length 7
from a to e :	$a - b - d - e$	of length 9

FIGURE 9.10 Application of Dijkstra's algorithm. The next closest vertex is shown in bold.