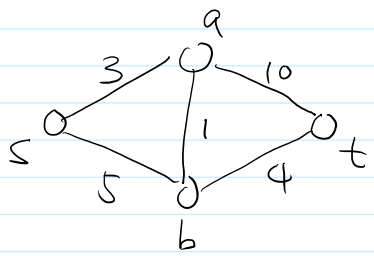
COMPSCI 330 Lecture 10 Shortest Paths

Wednesday, September 28, 2016 2:04 PM

- Structure of Shortest Paths
- Dijkstra's algorithm
- Bellman-Ford Algorithm

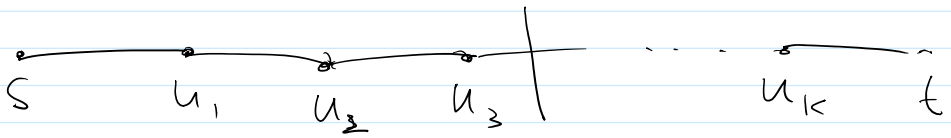
- Given a graph G, edge (u,v) has length $l(u,v)$
  find the shortest path from s to t.
  (length of path $= \sum_{(u,v)\in path} l(u,v)$)



shortest path:
$s \to a \to b \to t$, len: 8

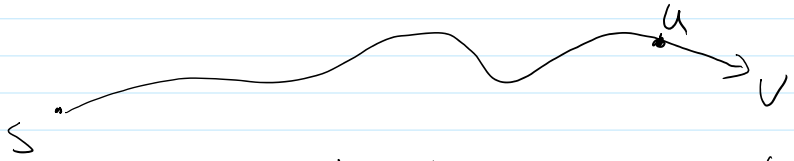
- Recall: Shortest path on DAG.
  shortest path using BFS

- basic property of shortest path
  Let $s, u_1, u_2, \cdots, u_k, t$ is a shortest path from s to t.
  Claim: $s, u_1, u_2, \cdots, u_i$ is also a shortest path from s to u_i.



Proof: If there is a shorter path $s, v_1, v_2, \cdots, v_l, u_i$
then $s, v_1, \cdots, v_l, u_i, u_{i+1}, \cdots, u_k, t$ will also be shorter than $s, u_1, u_2, \cdots, u_k, t$, but that is impossible □

- Dynamic Programming:
  $d(v)$: distance from s to v.
  length of the shortest path.



$$d(v) = \min_{(u,v)\in E} d(u) + l(u,v)$$ (same as the recursion for DAG)

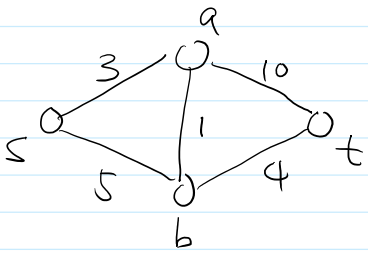
problem: if graph has cycles, it's hard to determine ordering.

- Dijkstra's Algorithm

main idea: compute $d(v)$ in increasing order.

$h(u)$: shortest path from s to u using only visited vertices

initialize $d(s) = 0$, mark s as visited
for any edge (s,u), $h(u) = l(s,u)$ $(prev(u) = s)$
for any other vertex $h(u) = +\infty$
for $i = 2$ to n
  let u to be the vertex with minimum $h(u)$ among unvisited vertices
  $d(u) = h(u)$, mark u visited
  for all edges (u,v)
    if $d(u) + l(u,v) < h(v)$
      $h(v) = d(u) + l(u,v)$ $(prev(v) = u)$



	s	a	b	t
h	0	3	5	$+\infty$
d	0	3		
h	0	3	4	13
d	0	3	4	
h	0	3	4	8
d	0	3	4	8

Correctness: use induction

Property: at any iteration; $d(u)$ for any visited u is the correct distance
$h(v)$ for any unvisited v is length of shortest path from s to v but last step uses a visited vertex.

Base case: only s is visited, property is clearly true.

induction: Suppose property is true at the beginning of iteration i, it is also true after the iteration.

need: ① for the vertex u we pick, $d(u)$ is the length of shortest path

② $h(v)$ updated correctly.

① assume shortest path from s to u is

$s, v_1, v_2, \cdots, v_r, u$

if v_r is visited, length $\geq h(u)$

if v_r is not visited, let v_j be the first unvisited vertex

$\text{length}(s, v_1, v_2, \cdots, v_{j-1}, v_j) \geq h(v_j) \geq h(u)$

(v_{j-1}: visited, v_j: not visited)

$\Rightarrow h(u) =$ length of shortest path

② after visit u

$$h(v) = \min_{\substack{u:\text{visited} \\ (u,v)\in E}} d(u) + l(u,v)$$

□

- Running time:

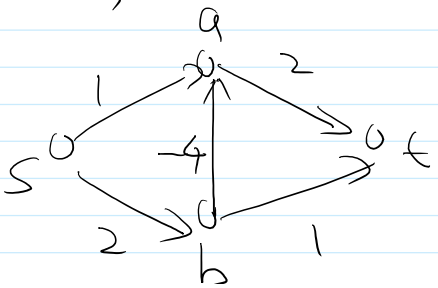
naive: $O(n^2)$

heap: $O((n+m)\log n)$

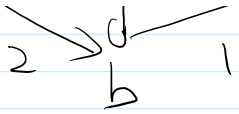
Fibonacci heap: $O(n\log n + m)$

- negative edges

- Dijkstra fails



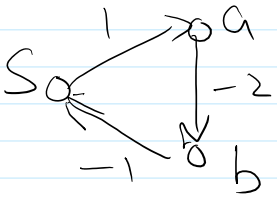
- negative cycle



- negative cycle

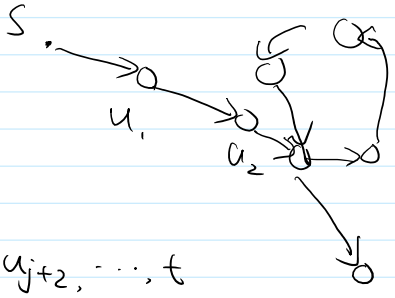
shortest path does not exist

$s \to a \to b \to s \to a \to b \to \cdots$



- Claim: if G does not have a negative cycle, shortest path will not visit any vertex twice.

Proof: consider path $s, u_1, u_2, \cdots, t$



if $u_i = u_j$

because $u_i, u_{i+1}, \cdots, u_j$ is a cycle

length $(u_i, u_{i+1}, \cdots, u_j) \geq 0$

remove the cycle and $s, \cdots, u_i, u_{j+1}, u_{j+2}, \cdots, t$

has length $\leq$ length of original path $\square$

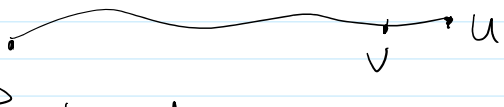
$\Longrightarrow$ number of steps of shortest path $\leq n-1$

- Bellman-Ford

$d(u,i)$ = shortest path from s to u using at most i steps.

initialize $d(s,0) = 0$, $d(u,0) = +\infty$

$$d(u,i) = \min \left\{ d(u,i-1), \min_{(v,u) \in E} d(v,i-1) + l(v,u) \right\}$$



Claim: If the graph has no negative cycles, $d(u,n-1)$ is the distance to u.

running time: $O(nm)$

Claim: $d(u,n) \neq d(u,n-1)$ for some u, there is a negative cycle.

if there is a negative cycle, then $\exists\, u$, $d(u,n) \neq d(u,n-1)$