CSE 421
Algorithms

Richard Anderson

Lecture 9

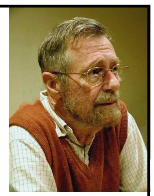
Dijkstra's algorithm

Who was Dijkstra?

- What were his major contributions?

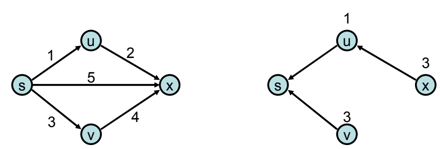
Edsger Wybe Dijkstra

http://www.cs.utexas.edu/users/EWD/

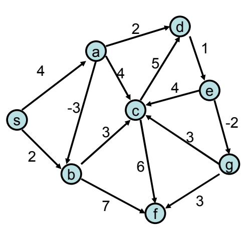


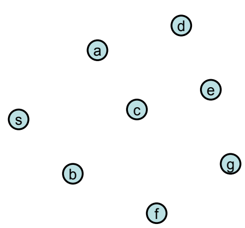
Single Source Shortest Path Problem

- Given a graph and a start vertex s
  - Determine distance of every vertex from s
  - Identify shortest paths to each vertex
    - Express concisely as a "shortest paths tree"
    - Each vertex has a pointer to a predecessor on shortest path



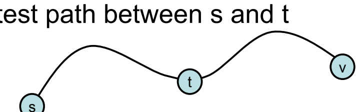
Construct Shortest Path Tree from s





Warmup

- If P is a shortest path from s to v, and if t is on the path P, the segment from s to t is a shortest path between s and t

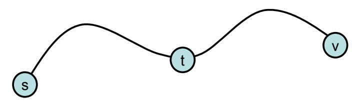


- WHY? (submit an answer)

Careful Proof

- Suppose s-v is a shortest path
- Suppose s-t is not a shortest path
- Therefore s-v is not a shortest path
- Therefore s-t is a shortest path

Prove if s-t not a shortest path then s-v is not a shortest path



Dijkstra's Algorithm

S = {}; d[s] = 0; d[v] = inf for v != s

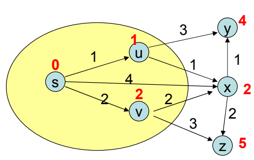
While S != V

Choose v in V-S with minimum d[v]

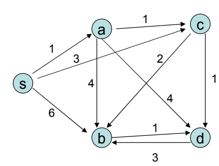
Add v to S

For each w in the neighborhood of v

d[w] = min(d[w], d[v] + c(v, w))



Simulate Dijkstra's algorithm (strarting from s) on the graph



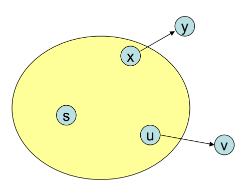
Round	Vertex Added	s	a	b	c	d
1						
2						
3						
4						
5						

Dijkstra's Algorithm as a greedy algorithm

- Elements committed to the solution by order of minimum distance

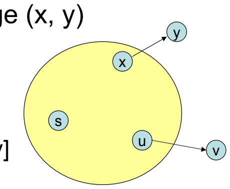
Correctness Proof

- Elements in S have the correct label
- Key to proof: when v is added to S, it has the correct distance label.



Proof

- Let P_v be the path of length d[v], with an edge (u,v)
- Let P be some other path to v. Suppose P first leaves S on the edge (x, y)
  - $P = P_{sx} + c(x,y) + P_{yv}$
  - $Len(P_{sx}) + c(x,y) >= d[y]$
  - $Len(P_{yv}) >= 0$
  - $Len(P) >= d[y] + 0 >= d[v]$

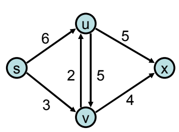


Negative Cost Edges

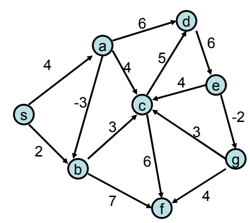
- Draw a small example a negative cost edge and show that Dijkstra's algorithm fails on this example

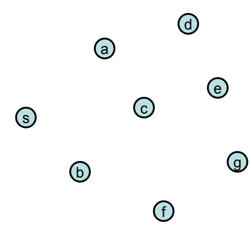
Bottleneck Shortest Path

- Define the bottleneck distance for a path to be the maximum cost edge along the path



Compute the bottleneck shortest paths





How do you adapt Dijkstra's algorithm to handle bottleneck distances

- Does the correctness proof still apply?