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

Dijkstra’s Algorithm

\[
S = \{ \}; \quad d[s] = 0; \quad d[v] = \infty \text{ for } v \neq s
\]

While \( S \neq V \)

Choose \( v \) in \( V - S \) with minimum \( d[v] \)

Add \( v \) to \( S \)

For each \( w \) in the neighborhood of \( v \)

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

Assume all edges have non-negative cost

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?
Simulate Dijkstra’s algorithm (starting from s) on the graph

<table>
<thead>
<tr>
<th>Round</th>
<th>Vertex Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>d</td>
</tr>
<tr>
<td>5</td>
<td>s</td>
</tr>
</tbody>
</table>

Who was Dijkstra?

- What were his major contributions?

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

- Edsger Wybe Dijkstra was one of the most influential members of computing science’s founding generation. Among the domains in which his scientific contributions are fundamental are
  - algorithm design
  - programming languages
  - program design
  - operating systems
  - distributed processing
  - formal specification and verification
  - design of mathematical arguments

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 v be a vertex in V-S with minimum d[v]
- Let P_v be a 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_u + c(x,y) + P_v
  - Len(P_u) + c(x,y) >= d[y]
  - Len(P_v) >= 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?