Overview

• Routing
• Shortest Path
Objective

- Understand how routing in the Internet is performed
- Learn about two essential routing algorithms that operate on graphs

Routing – Intro

- Shortest path routing
- Centralized approach
  - Each node has full “view” of network
  - Each node calculates shortest path using routing algorithm
  - “Link state algorithm”
  - (Exchange of link information always decentralized)
Routing - Intro

• Distributed approach
  • Each node computes best path without full view
  • Shortest path computed as link information is exchanged
  • “Distance vector algorithm”

Link State Algorithm

• Link cost of all links is broadcast to all nodes
• Dijkstra’s algorithm to find shortest path to all nodes
  • Each node calculates its own tree
• Notation:
  • $D(v)$ is least cost to $v$ in current iteration
  • $p(v)$ is previous node along least cost path
  • $N'$ is subset of nodes with guaranteed least cost paths
Link State Algorithm

Algorithm:
Initialization:
- \( N' = \{ u \} \)
- For all nodes \( v \): if neighbor of \( u \) then \( D(v) = c(u, v) \), else \( D(v) = \infty \)
- Loop until \( N' = N \):
  - Find \( w \in N' \) with minimum \( D(w) \) and add \( w \) to \( N' \)
  - For each neighbor \( v \) of \( w \) (\( v \in N' \)):
    - \( D(v) = \min(D(v), D(w) + c(w, v)) \)

```
from Graph import Graph, Vertex, PriorityQueue
def dijkstra(aGraph, start):
    pq = PriorityQueue()
    start.setDistance(0)
    pq.buildHeap([(v.getDistance(), v) for v in aGraph])
    while not pq.isEmpty():
        currentVert = pq.delMin()
        for nextVert in currentVert.getConnections():
            newDist = currentVert.getDistance() + currentVert.getWeight(nextVert)
            if newDist < nextVert.getDistance():
                nextVert.setDistance(newDist)
                nextVert.setPred(currentVert)
                pq.decreaseKey(nextVert, newDist)
```
Link State Algorithm

PQ = x, v, w

Link State Algorithm

PQ = v, y, w
Link State Algorithm

![Diagram of a network with nodes labeled u, v, w, x, y, z and their respective distances d. Edges are labeled with weights.]

PQ = yw

---

Link State Algorithm

![Diagram of a network with nodes labeled u, v, w, x, y, z and their respective distances d. Edges are labeled with weights.]

PQ = wz
Link State Algorithm

PQ = z

Link State Algorithm

PQ = None
Link State Algorithm

<table>
<thead>
<tr>
<th>step</th>
<th>N'</th>
<th>D(b),p(b)</th>
<th>D(c),p(c)</th>
<th>D(d),p(d)</th>
<th>D(e),p(e)</th>
<th>D(f),p(f)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{a}</td>
<td>1,a</td>
<td>\infty</td>
<td>3,a</td>
<td>\infty</td>
<td>\infty</td>
</tr>
<tr>
<td>1</td>
<td>{a,b}</td>
<td>6,b</td>
<td>2,b</td>
<td>5,b</td>
<td>\infty</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>{a,b,d}</td>
<td>6,b</td>
<td>4,d</td>
<td>9,d</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>{a,b,d,e}</td>
<td>5,e</td>
<td></td>
<td>8,e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>{a,b,c,d,e}</td>
<td></td>
<td></td>
<td></td>
<td>7,c</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>{a,b,c,d,e,f}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Next Steps

• Next lecture on Tuesday
• Exam on 11/7