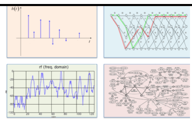


Network Routing - I (Without Failures)

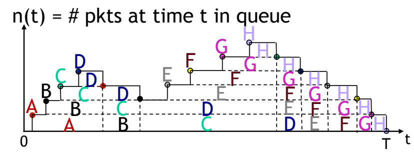
Lecture 18

6.02 Fall 2010

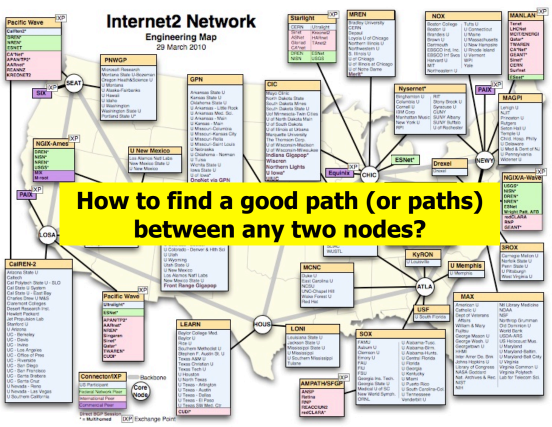
November 10, 2010

- Forwarding. Distributed routing.
- Distance-vector protocol with Bellman-Ford step
- Link-state protocol with Dijkstra's shortest-paths

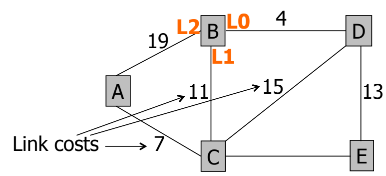
But First, Delays & Little's Law



- Suppose T is large and that P packets are forwarded in that time
- Let A = area under the n(t) curve from 0 to T
- Then, rate = P/T; and mean number of pkts in queue, E[n] = A/T
- How to calculate mean delay per packet?
  - A is aggregate delay weighted by each packet's time in queue (why?)
  - So, mean delay per packet sent = A/P
- Therefore, **E[n] = rate * (mean delay)**
- For a given link rate, increasing queue size increases delay

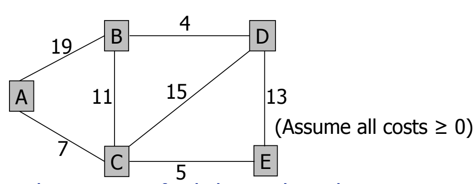


The Problem: Finding Paths



- **Addressing** (how to name nodes)
- **Forwarding** (how does a switch process a packet)
- **Routing** (building and updating data structures to ensure that forwarding works)
- Functions of the **network layer**

Shortest Path Routing



- Each node wants to find the path with *minimum total cost* to other nodes
  - We use the term "shortest path" even though we're interested in min cost (and not min #hops)
- Several possible **distributed** approaches
  - **Vector protocols, esp. distance vector (DV)**
  - **Link-state protocols (LS)**

Routing Table Structure

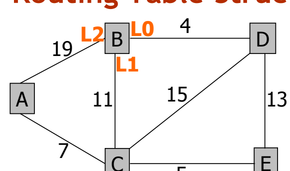


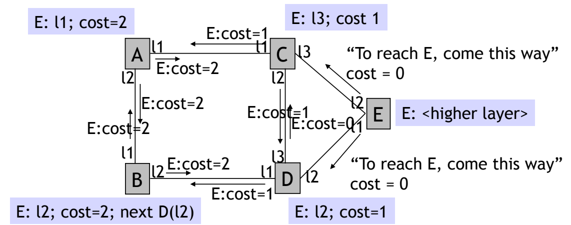
Table @ B

Destination	Link (next-hop)	Cost
A	ROUTE L1	18
B	'Self'	0
C	L1	11
D	L0	4
E	L1	16

Distributed Routing: A Common Plan

- **Determining live neighbors**
  - HELLO protocol (periodic) – next lecture
  - Common to both DV and LS protocols
- **Advertisement step (periodic)**
  - Send some information to all neighbors
- **Integration step**
  - Compute routing table using info from advertisements

Distance Vector Routing



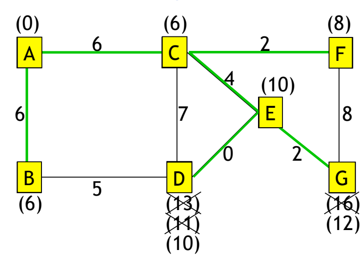
- Advertisement: Each node announces a vector of (destination:pathcost) tuples to all its neighbors
- Integration: On hearing advertisement, run Bellman-Ford step:
  if (current cost to dest > cost in advertisement + linkcost):
  update cost, route

Link-State Routing

- Advertisement step:
  - Information about its links to its neighbors
  - Neighbors re-send on *their* links → **flooding**
  - Result: Each node discovers map of the network
- Integration: Each node runs the same shortest path algorithm over its map
  - If each node implements computation correctly and each node has the same map, then routing tables will be correct
- Optimal substructure property:
  - ***Suppose shortest path from X to Y goes through Z. Then, the sub-path from X to Z must be a shortest path. [Why?]***

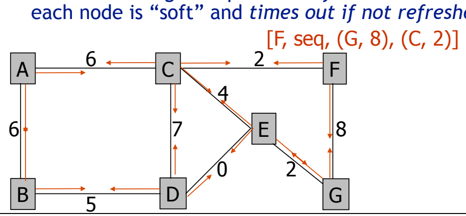
Integration Step: Dijkstra's Algorithm (Example)

Suppose we want to find paths from A to other nodes



Link-State Advertisements and Flooding

- Periodically send LSA (Link-State Advertisement) [seq#, [(nbhr1, linkcost1), (nbhr2, linkcost2), …] to all neighbors
- If seq > last_heard:
  - save seq, LSA; rebroadcast LSA to neighbors
- LSAs aren't reliable messages, so periodic
- Periodic messages help handle dynamism: state in each node is "soft" and *times out if not refreshed*



Summary

- The network layer implements the "glue" that achieves connectivity
  - Does addressing, forwarding, and routing
- Forwarding entails a routing table lookup; the table is built using routing protocol
- DV protocol: distributes route computation; each node advertises its best routes to neighbors
- LS protocol: distributes (floods) neighbor information; centralizes route computation using shortest-path algorithm