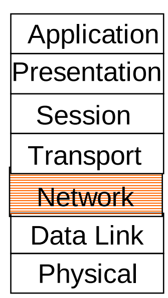
CSE/EE 461 – Lecture 10

Link State Routing

David Wetherall
djw@cs.washington.edu

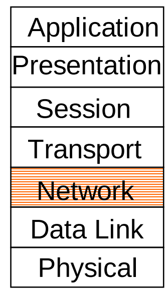
Last Time …

- Routing Algorithms
  - Introduction
  - Distance Vector routing (RIP)



This Lecture

- Routing Algorithms
  - Link State routing (OSPF)
  - Cost Metrics



Link State Routing

- Same assumptions/goals, but different idea than DV:
  - Tell all routers the topology and have each compute best paths
  - Two phases:
    1. Topology dissemination (flooding)
    2. Shortest-path calculation (Dijkstra's algorithm)

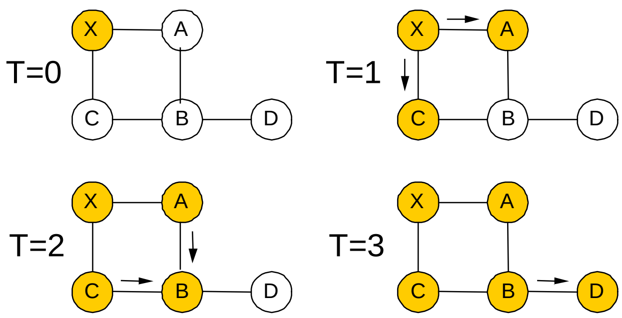
- Why?
  - In DV, routers hide their computation, making it difficult to decide what to use when there are changes
  - With LS, faster convergence and hopefully better stability
  - It is more complex though …

Flooding

- Each router maintains link state database and periodically sends link state packets (LSPs) to neighbor
  - LSPs contain [router, neighbors, costs]
- Each router forwards LSPs not already in its database on all ports except where received
  - Each LSP will travel over the same link at most once in each direction
- Flooding is fast, and can be made reliable with acknowledgments

Example

- LSP generated by X at T=0
- Nodes become yellow as they receive it



Complications

- When link / router fails need to remove old data. How?
  - LSPs carry sequence numbers to determine new data
  - Send a new LSP with cost infinity to signal a link down
- What happens when a router fails and restarts?
  - What sequence number should it use? Don’t want data ignored.
  - One option: age LSPs and send with “TTL 0” to purge
- What happens if the network is partitioned and heals?
  - Different LS databases must be synchronized
  - A version number is used!

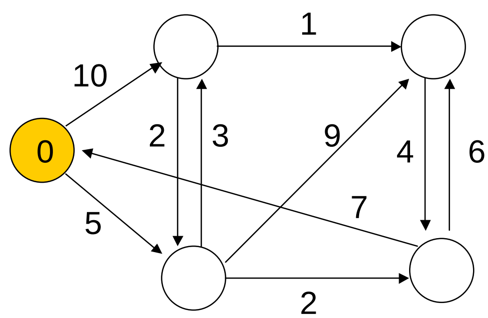
Shortest Paths: Dijkstra’s Algorithm

- Graph algorithm for single-source shortest path

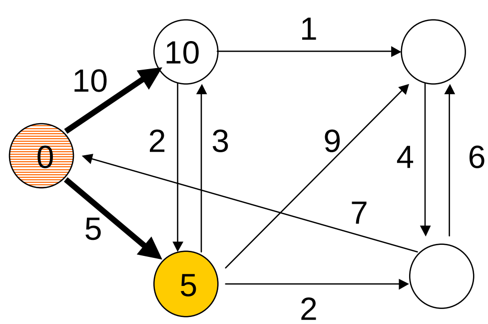
```
S     {}
Q     <all nodes keyed by distance>
While Q != {}
        u     extract-min(Q)
        S     S plus {u}
        for each node v adjacent to u
                "relax" the cost of v
```

u is done, add to shortest paths

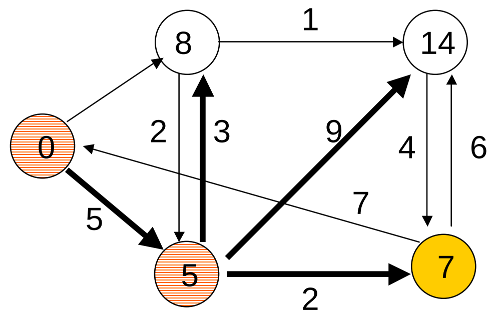
Dijkstra Example – Step 1



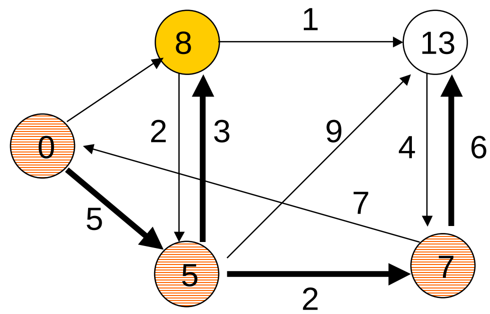
Dijkstra Example – Step 2



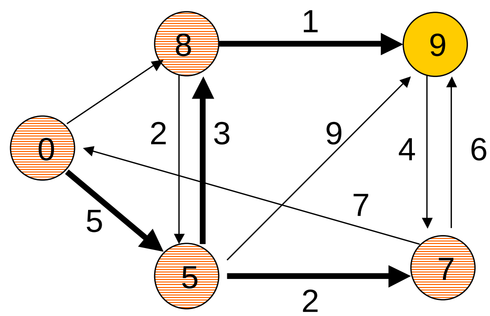
Dijkstra Example – Step 3



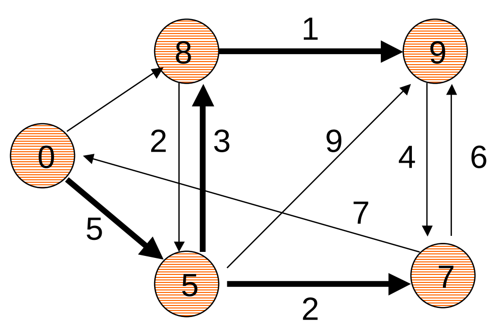
Dijkstra Example – Step 4



Dijkstra Example – Step 5



Dijkstra Example – Done



Open Shortest Path First (OSPF)

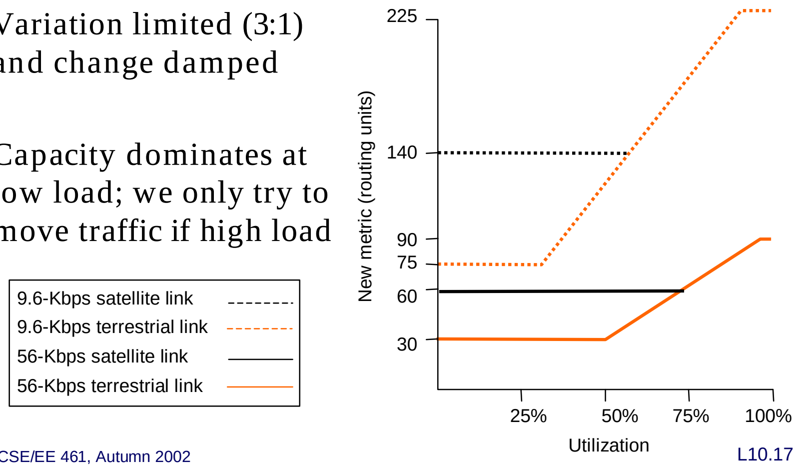
- Most widely-used Link State protocol today
- Basic link state algorithms plus many features:
  - Authentication of routing messages
  - Extra hierarchy: partition into routing areas
  - Load balancing: multiple equal cost routes

Cost Metrics

- How should we choose cost?
  - To get high bandwidth, low delay or low loss?
  - Do they depend on the load?
- Static Metrics
  - Hopcount is easy but treats OC3 (155 Mbps) and T1 (1.5 Mbps)
  - Can tweak result with manually assigned costs
- Dynamic Metrics
  - Depend on load; try to avoid hotspots (congestion)
  - But can lead to oscillations (damping needed)

Revised ARPANET Cost Metric

- Based on load and link
- Variation limited (3:1) and change damped
- Capacity dominates at low load; we only try to move traffic if high load



Key Concepts

- Routing uses global knowledge; forwarding is local
- Many different algorithms address the routing problem
  - We have looked at two classes: DV (RIP) and LS (OSPF)
- Challenges:
  - Handling failures/changes
  - Defining "best" paths
  - Scaling to millions of users