Introduction to Computer Networks

COSC 4377

Lecture 13

Spring 2012

March 5, 2012

Announcements

- HW6 due this week
- No HW due next week

HW6: Rate Limits

- Client and Server negotiate rate
- Server does fair bandwidth allocation among clients, dynamic adjustments
- Rate throttle large downloads

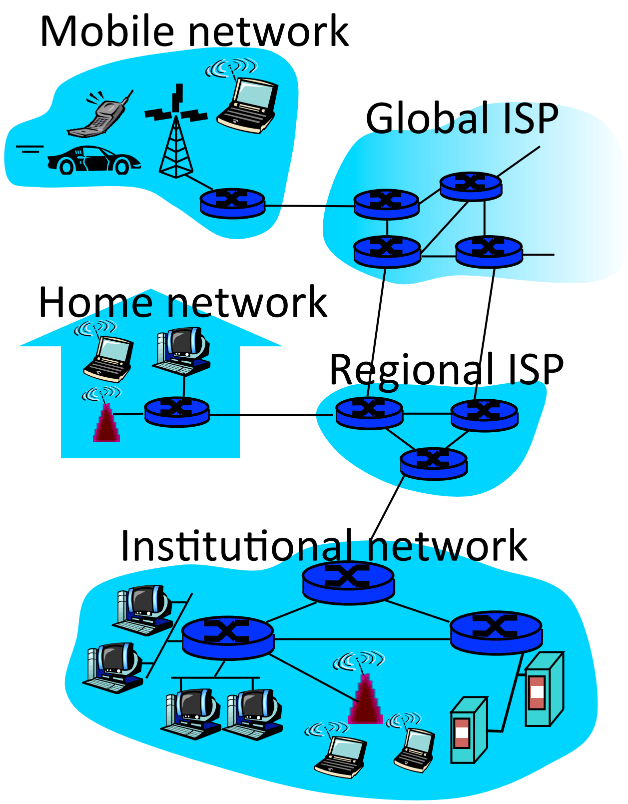
Today's Topics

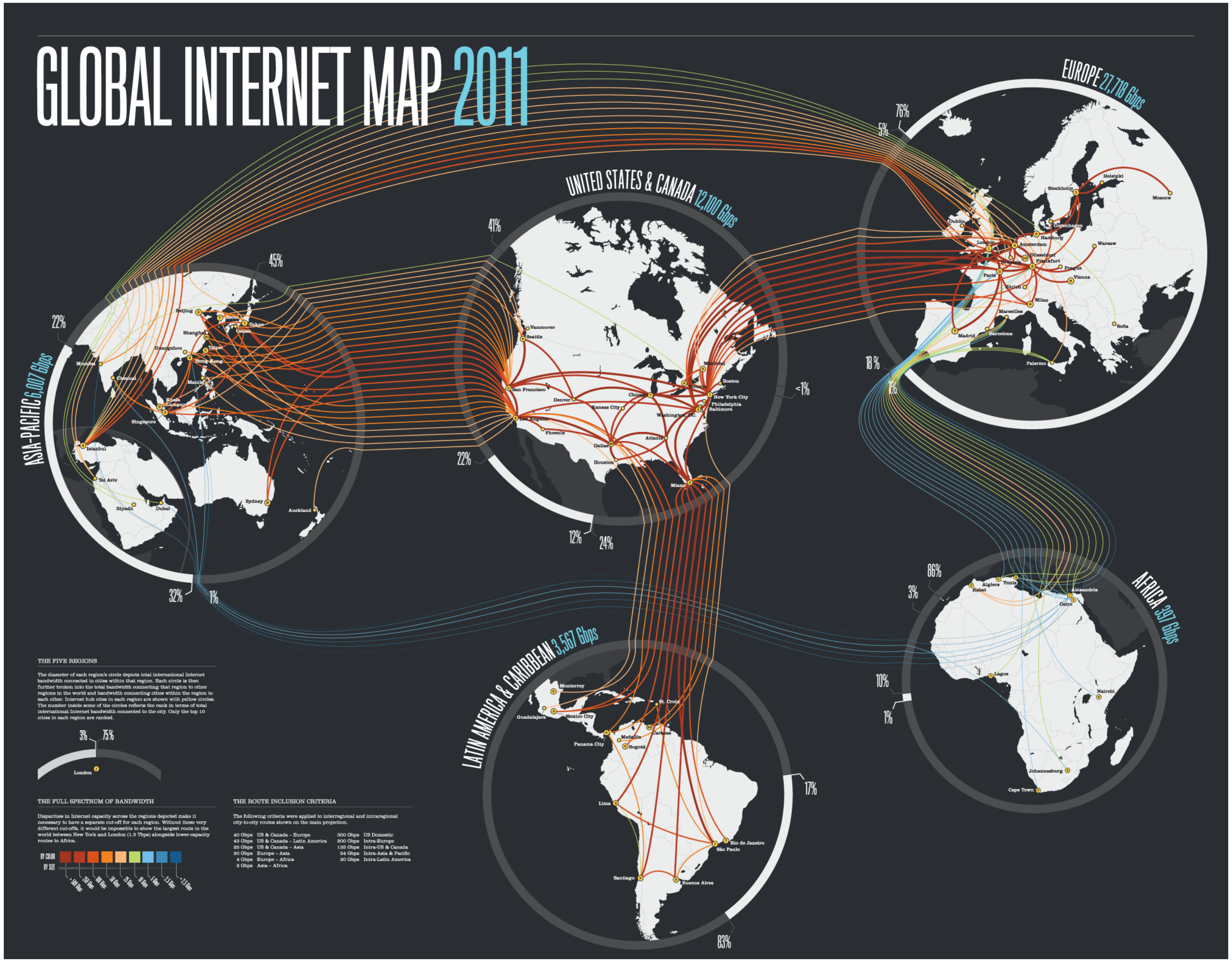
- Routing
- Distance Vector Routing

The Internet

- Collection of nodes, wire and wireless technology connecting those nodes, applications and services

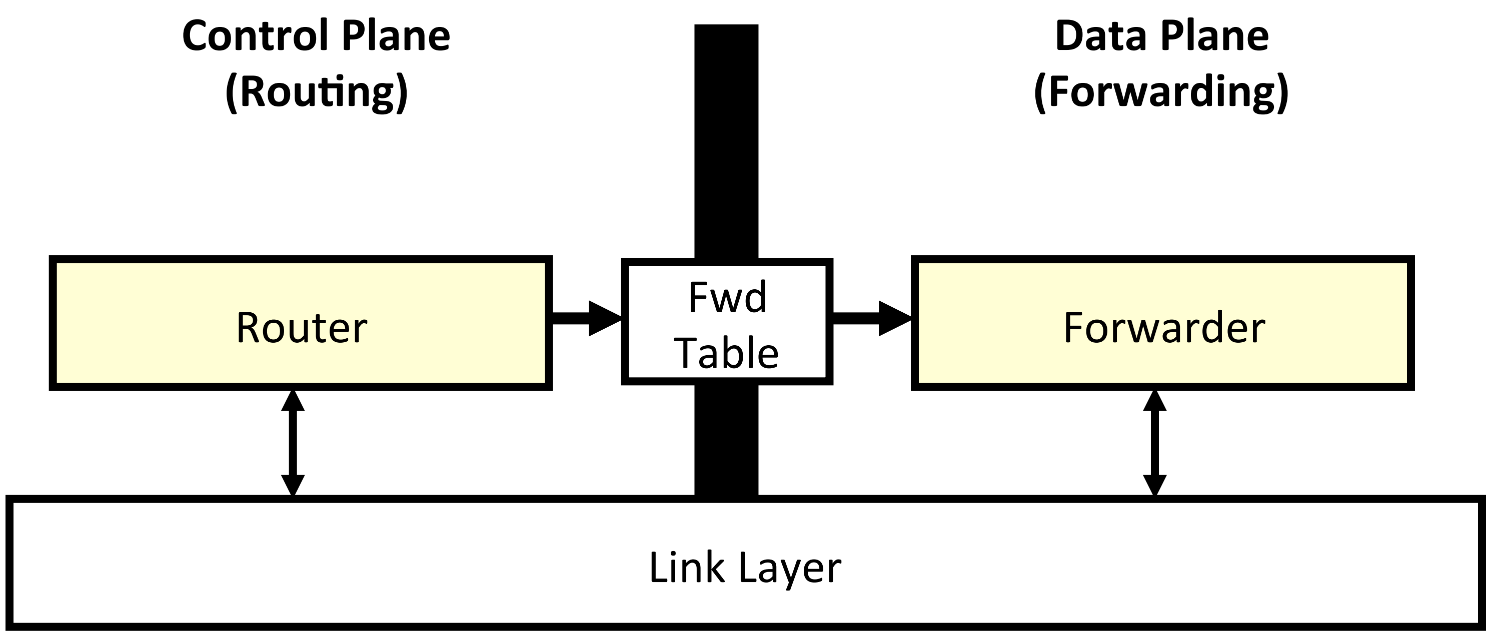
- Types of nodes
  – Desktops and Laptops
  – Servers
  – TV / Refrigerator
  – Cellphones





http://www.telegeography.com/telecom-resources/map-gallery/global-internet-map-2011/index.html

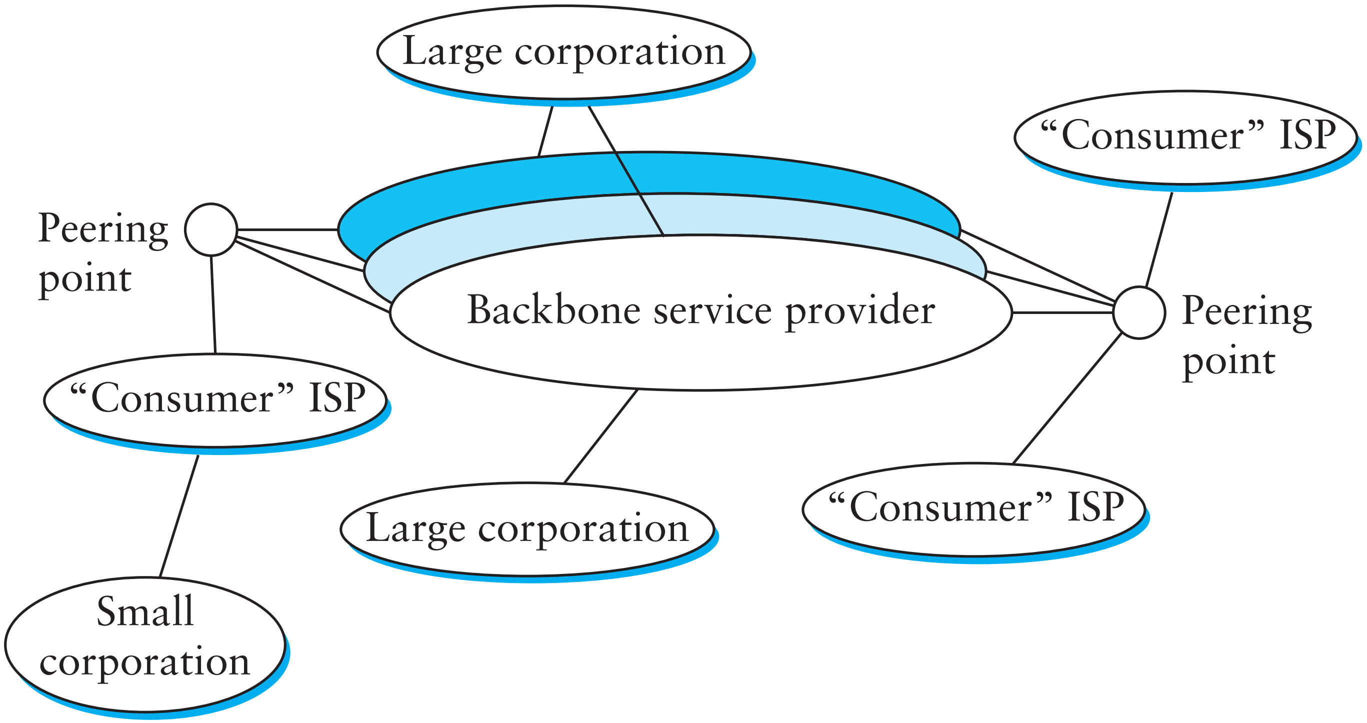
Router Architecture



Routing

- Routing is the process of updating forwarding tables
  - Routers exchange messages about routers or networks they can reach
  - Goal: find optimal route for every destination
  - … or maybe a good route, or *any* route (depending on scale)
- Challenges
  - Dynamic topology
  - Decentralized
  - Scale

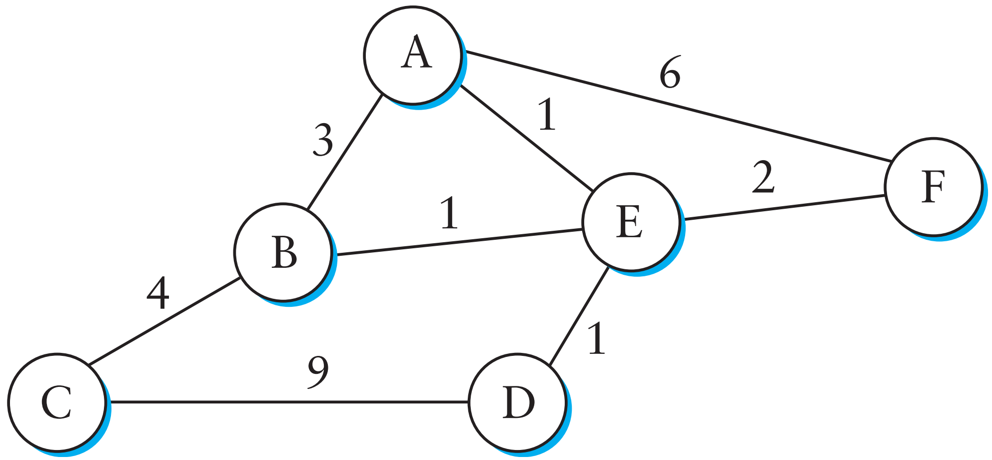
Internet structure



Inter and Intra-domain routing

- Routing organized in two levels
- Intra-domain routing
  - Complete knowledge, strive for *optimal* paths
  - Scale to ~100 networks
- Inter-domain routing
  - Aggregated knowledge, scale to Internet
  - Dominated by *policy*

Network as a graph

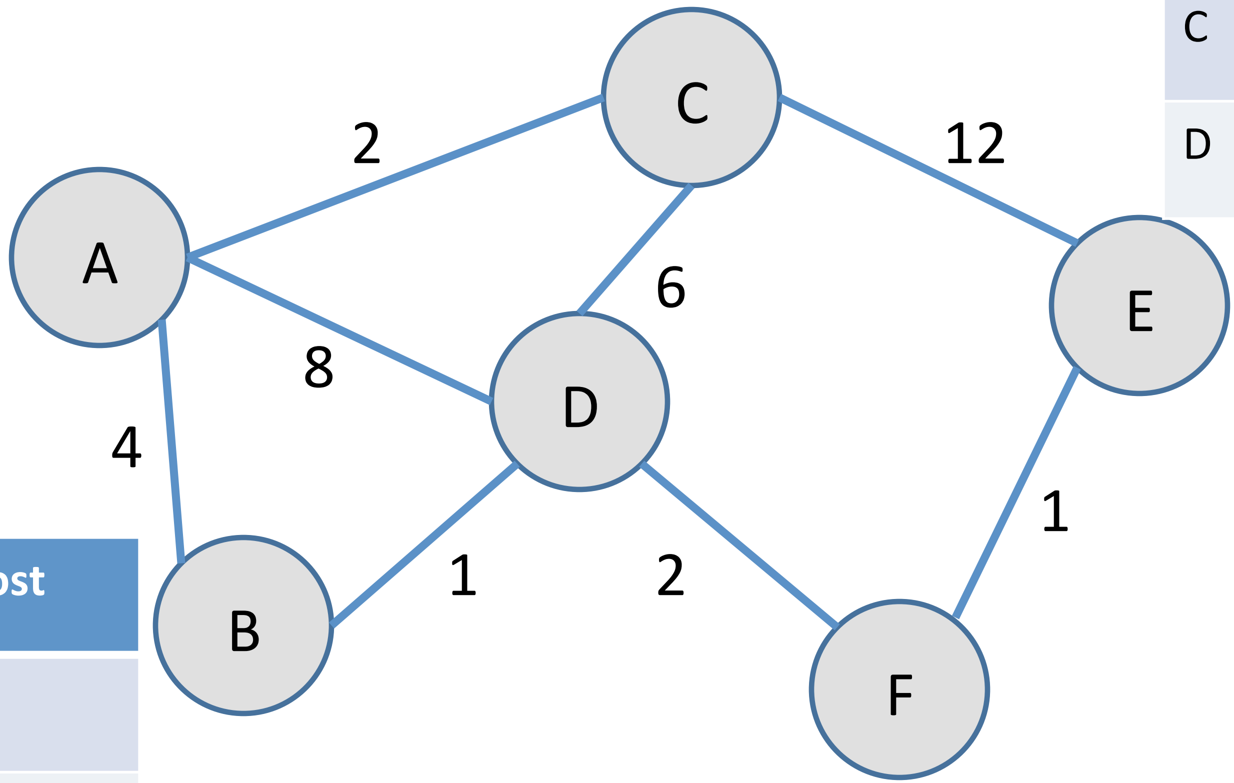


- Nodes are routers
- Assign *cost* to each edge
  - Can be based on latency, b/w, queue length, …
- Problem: find lowest-cost path between nodes
  - Each node individually computes routes

Basic Algorithms

- Two classes of intra-domain routing algorithms
- Distance Vector
  - Requires only local state
  - Harder to debug
  - Can suffer from loops
- Link State
  - Each node has global view of the network
  - Simpler to debug
  - Requires global state

Shortest Path Example



E’s table

ID	Nexthop
C	F
D	F

B’s table

ID	Cost
A	4
D	1

Distance Vector

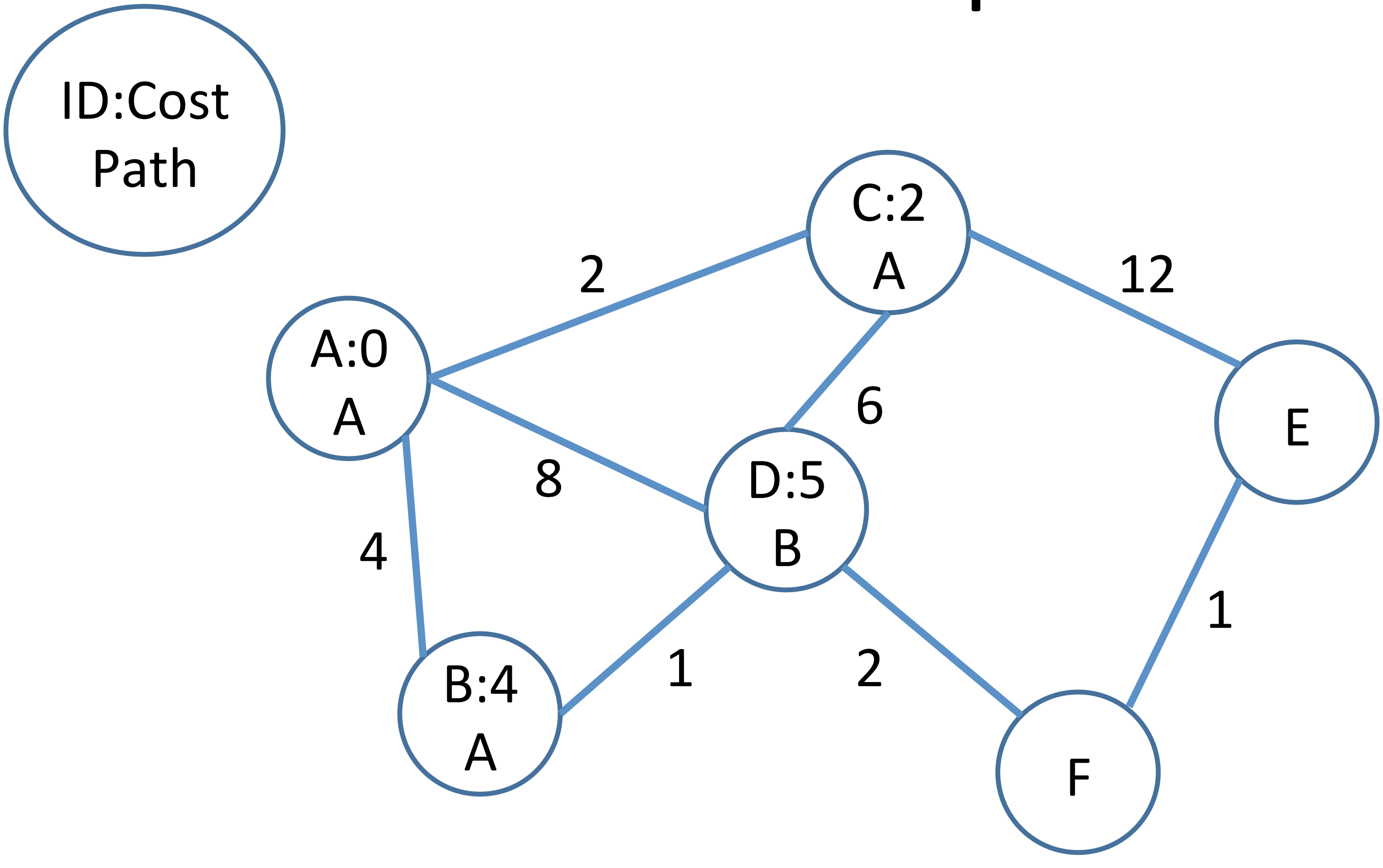
- Local routing algorithm
- Each node maintains a set of triples
  - <*Destination, Cost, NextHop*>
- Exchange updates with neighbors
  - Periodically (seconds to minutes)
  - Whenever table changes (*triggered* update)
- Each update is a list of pairs
  - <*Destination, Cost*>
- Update local table if receive a "better" route
  - Smaller cost
- Refresh existing routes, delete if time out

Calculating the best path

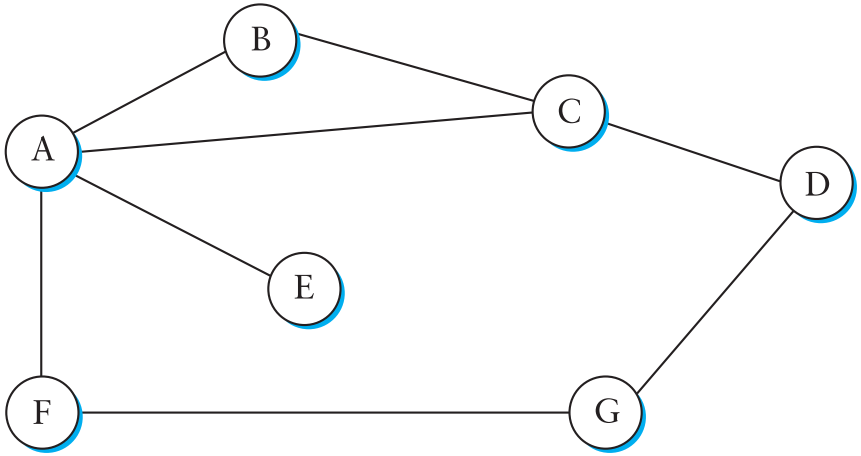
- Bellman-Ford equation
- Let:
  - $D_a(b)$ denote the current best distance from a to b
  - $c(a,b)$ denote the cost of a link from a to b
- Then $D_x(y) = \min_z(c(x,z) + D_z(y))$
- Routing messages contain D
- D is any additive metric
  - e.g, number of hops, queue length, delay
  - log can convert multiplicative metric into an additive one (e.g., probability of failure)

http://en.wikipedia.org/wiki/Bellman-Ford_algorithm

DVR Example



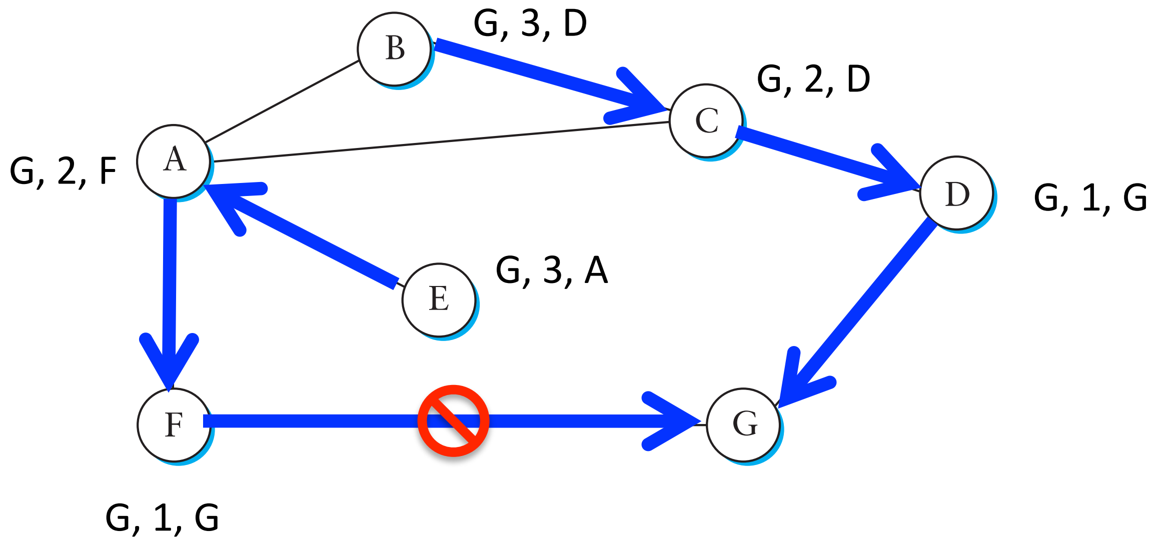
DV Example



B's routing table

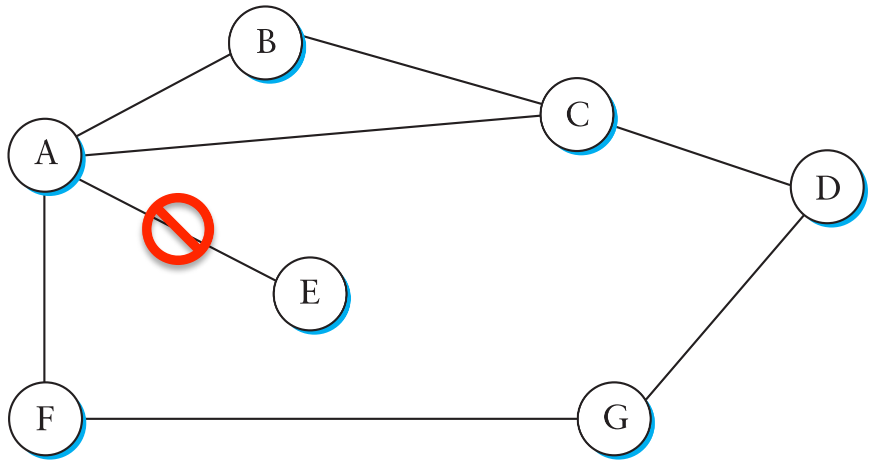
Destination	Cost	Next Hop
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	A

Adapting to Failures



- F-G fails
- F sets distance to G to infinity, propagates
- A sets distance to G to infinity
- A receives periodic update from C with 2-hop path to G
- A sets distance to G to 3 and propagates
- F sets distance to G to 4, through A

Count-to-Infinity



- Link from A to E fails
- A advertises distance of infinity to E
- B and C advertise a distance of 2 to E
- B decides it can reach E in 3 hops through C
- A decides it can reach E in 4 hops through B
- C decides it can reach E in 5 hops through A, …
- When does this stop?