Thought experiment: you own an ISP – what do you do?
Similar to UW network, corporate network.
What is your internal routing protocol?
  • OSPF?
What if you had to create your own protocol?

Intradomain routing (assume 100 – 1000) nodes):
Note that we distinguish between routing (addressing) and forwarding (packet traffic)
Centralized Arbiter?
  • Scaleability problems
    o Communication bottleneck?
      □ OSPF utilizes same bandwidth (~1 packet / minute / node)
    o Computation bottleneck? \(n^2 \log n\) – not a big deal for proper hardware
  • Single point of failure (fault tolerance)
    o Can address with redundant servers
  • Simpler
  • Up to date info?
    o Dynamic table maintenance is difficult:
      □ Flooding results in \(n^3\) effort
      □ Can be done, but the algorithms are tricky

Distributed Route Computation?
  • **Distance Vector** (pairwise communication/propogation/augmentation)
    o Exchange tables / keep best alternative
  • Why not Distance Vector?
    o What happens on a change?
      □ The source of information gets lost
      □ The source of change doesn’t have any assurance that the change has been propagated
Possible for deadlock to occur during propagation (poison reverse)

- Addressed by propagating full path, so that routers can debug path changes
  - Called **Path Vector**
  - Foundation of **BGP**

- Why does Distance (Path?) Vector do “periodic updates”?
  - pinging the network – “is my neighbor up?”
  - dynamic routing
  - discovery of changes
  - Soft State (auto-recovery) – note that BGP uses deltas to minimize data transfer
  - Self Correcting

- **Link State**: (send all info about neighbors)
  - Periodic update

Interesting Question: when you get an update, do you first update your tables – or do you notify your neighbors first?

- If you get a change, you don’t want your neighbor to update their tables before you update yours
  - Race conditions
  - Out of synchs

- Note that “old” routing table is used until table is updated (can’t stop routing!)
  - Time-to-live counter is available to keep errant packets from accidentally cycling forever

Interesting Question: how do we implement QoS over a dynamically changing network?

- With great difficulty! (but not impossible)
- Note that Telephone Network uses a separate “control” network, where the internet has only one network, that carries both data and control information
  - “One network is simpler” – simplified hardware requirements
  - “Two networks is simpler” – reliability simplified by separation of duties, but not overall

How do we choose a performance metric?

- Some combination of bandwidth and delay
• Queues
• Utilization

Cisco’s default algorithm uses bandwidth only
NP-Complete problem to “calculate” weighting algorithms

• Option 1 is to make your best guess (or use reasonable algorithms to approximate an “optimal” solution – e.g. simulated annealing)

• Option 2 is to “overlay” circuits onto the network, then determine policies that parse traffic between the “logical” circuits (which controls the physical routes)

• Use linear programming algorithms (constraint-based) to assign routing “logic” to the various gates