Lectures 18: Routing in Data Networks

Eytan Modiano
Routing

• Must choose routes for various origin destination pairs (O/D pairs) or for various sessions
  
  – Datagram routing: route chosen on a packet by packet basis

    Using datagram routing is an easy way to split paths

  – Virtual circuit routing: route chosen a session by session basis

  – Static routing: route chosen in a prearranged way based on O/D pairs
Routing is a global problem

- Static routing is not desirable
- Datagam routing is a natural way to split the traffic
  - How?
Shortest Path routing

• Each link has a cost that reflects
  – The length of the link
  – Delay on the link
  – Congestion
  – $$ cost

• Cost may change with time

• The length of the route is the sum of the costs along the route

• The shortest path is the path with minimum length

• Shortest Path algorithms
  – Bellman-Ford: centralized and distributed versions
  – Dijkstra’s algorithm
  – Many others
Directed graphs (digraphs)

- A directed graph (digraph) \( G = (N,A) \) is a finite nonempty set of nodes \( N \) and a set of ordered node pairs \( A \) called directed arcs.

- Directed walk: \( (4,2,1,4,3,2) \)

- Directed path: \( (4,2,1) \)

- Directed cycle: \( (4,2,1,4) \)

- Data networks are best represented with digraphs, although typically links tend to be bi-directional (cost may differ in each direction)
  - For simplicity we will use bi-directional links of equal costs in our examples
Bellman Ford algorithm

• Finds the shortest paths, from a given source node (1) to all other nodes

• General idea:
  – First find the shortest single arc path,
  – Then the shortest path of at most two arcs, etc.
  – Let $d_{ij}=\infty$ if (i,j) is not an arc.

• Let $D_i(h)$ be the shortest distance from 1 to i using at most h arcs
  – $D_i(1) = d_{1i}$ ; i\neq 1 \quad D_1(1) = 0$
  – $D_i(h+1) = \min \{j\} \ [D_j(h) + d_{ji}] ; i\neq 1 \quad D_1(h+1) = 0$

• If all weights are positive, algorithm terminates in N-1 steps
Bellman Ford - example
Distributed Bellman Ford

• Link costs may change over time
  – Changes in traffic conditions
  – Link failures
  – Mobility

• Each node maintains its own routing table
  – Need to update table regularly to reflect changes in network

• Let $D_i$ be the shortest distance from node i to the destination
  \[ D_i = \min \{j\} \left[ D_j + d_{ij} \right] \] : update equation

• Each node (i) regularly updates the values of $D_i$ using the update equation
  – Each node maintains the values of $d_{ij}$ to its neighbors, as well as values of $D_j$ received from its neighbors
  – Uses those to compute $D_i$ and send new value of $D_i$ to its neighbors
  – If no changes occur in the network, algorithm will converge to shortest paths in no more than N steps
Slow reaction to link failures

• Start with $D_3=1$ and $D_2=100$
  – After one iteration node 2 receives $D_3=1$ and $D_2 = \min [1+1, 100] = 2$

• In practice, link lengths occasionally change
  – Suppose link between 3 and 1 fails (i.e., $d_{31}=\infty$)
  – Node 3 will update $D_3 = d_{32} + D_2 = 3$
  – In the next step node 2 will update: $D_2 = d_{23} + D_3 = 4$
  – It will take nearly 100 iterations before node 2 converges on the correct route to node 1

• Possible solutions:
  – Propagate route information as well
  – Wait before rerouting along a path with increasing cost
    Node next to failed link should announce $D=\infty$ for some time to prevent loops
Assume $d_{ij}$ is equal to the flow on $(i,j)$ Note that $D_6 = D_5 + 0$

As routes change due to traffic conditions, they affect the loadings on the links, hence routes may oscillate
Instability

- Having a bias independent of flow in the arc distances helps to prevent this problem.

- Asynchronous updates also helps.
Dijkstra's algorithm

• Find the shortest path from a given source node to all other nodes
  – Requires non-negative arc weights

• Algorithm works in stages:
  – Stage k: the k closest nodes to the source have been found
  – Stage k+1: Given k closest nodes to the source node, find k+1st

• Key observation: the path to the k+1st closest nodes includes only nodes from among the k closest nodes

• Let M be the set of nodes already incorporated by the algorithm
  – Start with $D_n = d_{sn}$ for all n ($D_n =$ shortest path distance from node n to the source node
  – Repeat until $M = N$

  Find node $w \notin M$ which has the next least cost distance to the source node
  Add $w$ to $M$
  Update distances: $D_n = \min \{ D_n, D_w + d_{wn} \}$ (for all nodes $n \notin M$)

  – Notice that the update of $D_n$ need only be done for nodes not already in $M$ and that the update only requires the computation of a new distance by going through the newly added node $w$
Dijkstra example
Dijkstra’s algorithm implementation

- **Centralized version:** Single node gets topology information and computes the routes
  - Routes can then be broadcast to the rest of the network

- **Distributed version:** each node $i$ broadcasts $\{d_{ij} \text{ all } j\}$ to all nodes of the network; all nodes can then calculate shortest paths to each other node
  - Open Shortest Path First (OSPF) protocol used in the internet
Routing in the Internet

• Autonomous systems (AS)
  – Internet is divided into AS’s each under the control of a single authority

• Routing protocol can be classified in two categories
  – Interior protocols - operate within an AS
  – Exterior protocols - operate between AS’s

• Interior protocols
  – Typically use shortest path algorithms
    Distance vector - based on distributed Bellman-ford
    link state protocols - Based on “distributed” Dijkstra’s
Distance vector protocols

• Based on distributed Bellman-Ford
  – Nodes exchange routing table information with their neighbors

• Examples:
  – Routing information protocols (RIP)
    Metric used is hop-count ($d_{ij}=1$)
    Routing information exchanged every 30 seconds

  – Interior Gateway Routing Protocol (IGRP)
    CISCO proprietary
    Metric takes load into account
    $D_{ij} \sim 1/(\mu-\lambda)$ (estimate delay through link)

    Update every 90 seconds
    Multi-path routing capability
Link State Protocols

• Based on Dijkstra’s Shortest path algorithm
  – Avoids loops
  – Routers monitor the state of their outgoing links
  – Routers broadcast the state of their links within the AS
  – Every node knows the status of all links and can calculate all routes using Dijkstra’s algorithm
    Nonetheless, nodes only send packet to the next node along the route with the packets destination address. The next node will look-up the address in the routing table

• Example: Open Shortest Path First (OSPF) commonly used in the internet

• Link State protocols typically generate less “control” traffic than Distance-vector
Inter-Domain routing

• Used to route packets across different AS’s

• Options:
  – Static routing - manually configured routes
  – Distance-vector routing
    Exterior Gateway Protocol (EGP)
    Border Gateway Protocol (BGP)

• Issues
  – What cost “metric” to use for Distance-Vector routing
    Policy issues: Network provider A may not want B’s packets routed through its network or two network providers may have an agreement
    Cost issues: Network providers may charge each other for delivery of packets
Bridges, Routers and Gateways

• A Bridge is used to connect multiple LAN segments
  – Layer 2 routing (Ethernet)
  – Does not know IP address
  – Varying levels of sophistication
    Simple bridges just forward packets
    smart bridges start looking like routers

• A Router is used to route connect between different networks using network layer address
  – Within or between Autonomous Systems
  – Using same protocol (e.g., IP, ATM)

• A Gateway connects between networks using different protocols
  – Protocol conversion
  – Address resolution

• These definitions are often mixed and seem to evolve!
Bridges, routers and gateways

Small company

Ethernet A

Bridge

Ethernet B

IP

Router

Gateway

Another provider’s
Frame Relay
Backbone

Gateway

Service provider’s
ATM
backbone

ATM switches
(routers)