Routing Algorithms

CS158a

Chris Pollett

Apr 4, 2007.

Outline

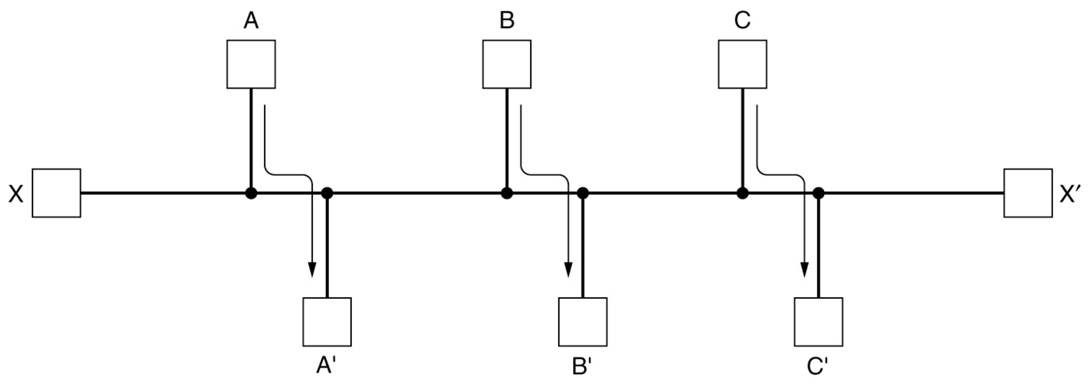
- Routing Algorithms
- Adaptive/non-adaptive algorithms
- The Optimality Principle
- Shortest Path Routing
- Flooding
- Distance Vector Routing

Routing Algorithms

- A **routing algorithm** is that part of the network layer responsible for deciding which output line an incoming packet should be transmitted on.
- If datagrams are being used this decision is made again for each packet coming from the same host.
- For virtual circuits you have **session routing**.
- There are two processes going on inside a router: (1) look up outgoing lines in a table and figure out which line to send them on. (**forwarding)** (2) update the routing table.
- Routing algorithms are responsible for (2).

Desirable Properties of Routing Algorithms

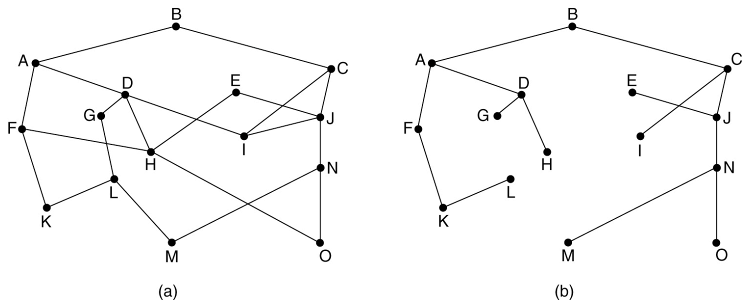
- **Correctness** -- should get packets eventually to the correct destination
- **Simplicity** -- this usually implies faster
- **Robustness** -- should be able to handle new routers coming online, as well as, handle other going off or malfunctioning.
- **Stability** -- under constant conditions should converge to some equilibrium.
- **Fairness and Optimality** -- these are hard to simultaneously satisfy. For example, in the situation below it might occur that to optimize flow we would not allow traffic between X and X´, a situation which is not fair.



Adaptive/nonadaptive algorithms

- **Nonadaptive Algorithms** -- do not base their routing decisions on measurements or estimates of the current traffic and topology. All decisions are made in advance and off-line. They are downloaded to the router when it is booted. This is sometimes called **static routing.**
- **Adaptive Algorithms** -- change their routing decisions to reflect changes in topology and traffic. Usually, routers in such an algorithm use local information gleaned by looking at data from adjacent routers.

The Optimality Principle

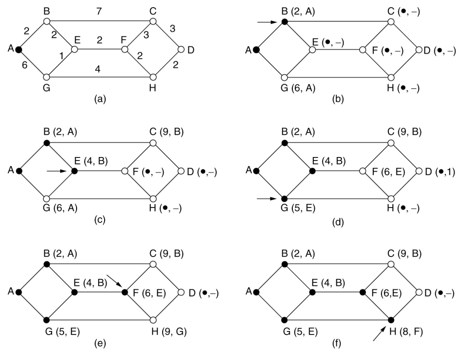


A sink tree

for B

- This principle states that if router J is on the optimal path from I to K, then the optimal path from J to K lies on the same route.
- The proof is that, if not, we could find a better route from I to K by using the same path from I, J, but following the better path from J to K.
- It follows from the optimality principle that the optimal routes from all sources to a given destination form a tree rooted at the destination.
- This is called a **sink tree**. The distance is measure as number of hops.

Shortest Path Routing

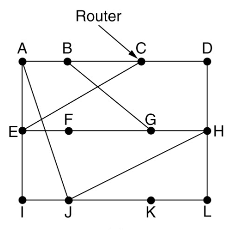


- We will measure **shortest paths** in terms of number of hops (not geographic distance).
- Dijkstra's algorithm (1959) can be used to find the shortest path between two points in a graph.
  - Each node is initially labeled with its distance from the source along the best known path. Node for which no path is known are labeled initially with INFINITY.
  - Nodes are also labeled temporary or permanent. Initially, the start node is labeled permanent. We also have an active node which is initially the start node.
  - In a round, we update the distances to each of the nodes adjacent to the active node, we then search the graph for the temporary node of least distance, mark it permanent and set it as active. Then iterate until no change.

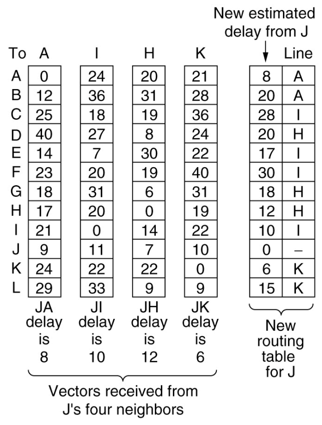
Flooding

- This is another static algorithm.
- Every incoming packet is sent out on every outgoing route except the one it arrived on.
- To prevent the algorithm from generating an infinite number of packets, a hop counter can be used.
- Ideally, would like the hop counter to be initialized to the distance from source to the destination. You can fake this by using the diameter of the subnet
- Another way to dam flooding is to have sequence numbers in the header. If a sequence number is seen twice by a router it is discarded
- In **selective flooding**, the router don’t send out on every line, only those in the approximate direction of the destination.

Distance Vector Routing



(a)



To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

(b)

- This is our first dynamic routing algorithm.
- It also goes by the names Bellman-Ford, Ford-Fulkerson, or RIP.
- Each router maintains a table containing one entry for each router in the subnet of the form (router, outgoing line)

More on Distance Vector Routing

- This entry contains the preferred outgoing line and an estimate of the time or distance to the destination.
- Each router is assumed to know the distance to its neighbors. (1 hop)
- To update its table, each router gets its neighbors’ tables and then recomputes its distances to destinations.