Computer Communication Networks

Routing algorithms

Review

- IP
  - addressing and *routing*
    - address class, classless, NAT
  - fragmentation and reassembly

Routing

- Routing algorithms
  - flooding
    - receive from one interface and send to other ifs
    - reduce duplicate packets
      - TTL
      - if received before, drop
      - shortest reverse path
  - link state
  - distance vector

Link state routing

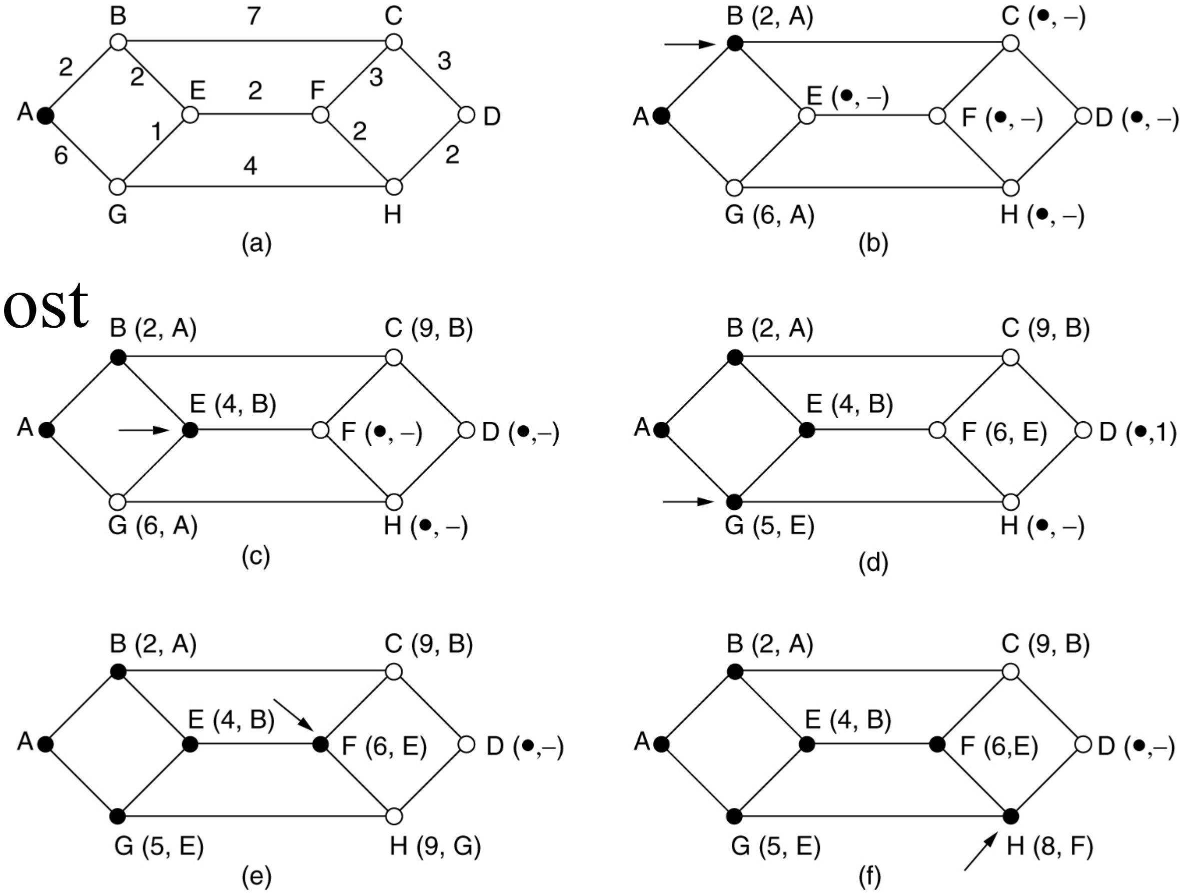
- Neighbor discovery
  - “hello-hello” between directly connected nodes
- Link-state broadcast
  - link state: cost, delay, or other metrics
- Topology generation
  - node/link graph
- Shortest-path calculation
  - from one node to all other nodes

Dijkstra algorithm

```
Initialization:
  N' = {u}
  for all nodes v
    if v adjacent to u
        then $D(v) = c(u,v)$
    else $D(v) = \infty$

Loop
  find w not in N' such that D(w) is a minimum
  add w to N'
  update D(v) for all v adjacent to w and not in N' :
      $D(v) = \min( D(v), D(w) + c(w,v) )$
  /* new cost to v is either old cost to v or known
   shortest path cost to w plus cost from w to v */
until all nodes in N'
```

Dijkstra's algorithm: example

- Condition
  - – nonnegative link cost



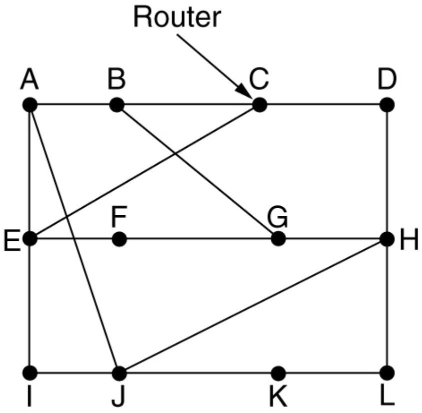
Distance vector routing

- Neighbor discovery
  - “hello-hello” between directly connected nodes
- Route exchange
  - A: “I can reach X at cost Path (A,X).”
  - B: “I can reach X at cost Path (B,X).”
  - A: “I am Link (A,B) away from B.”
- Shortest-path calculation
  - A: min{Path (A,X), Link (A,B) + Path (B,X)}

Bellman-Ford algorithm

```
Initialization:
  for all adjacent nodes v:
     D  (*,v) = infinity        /* the * operator means "for all rows" */
     D  (v,v) = c(X,v)          /* direct neighbors */
  for all destinations, y
     send min  D  (y,w) to each neighbor  /* w over all X's neighbors */

loop
  wait (until I receive update from neighbor V)

  if (update received from V wrt destination Y)
     /* shortest path from V to some Y has changed  */
     /* V has sent a new value for its  min   DV(Y,w) */
     /* call this received new value is "newval"     */
     for the single destination y: D  (Y,V) = c(X,V) + newval

  if we have a new min   D  (Y,w) for any destination Y
     send new value of min   D  (Y,w) to all neighbors

forever
```

Bellman-Ford algorithm: example



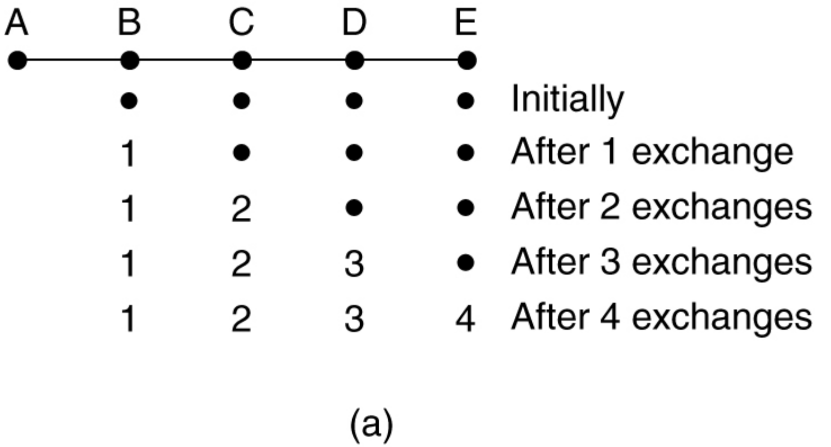
(a)

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

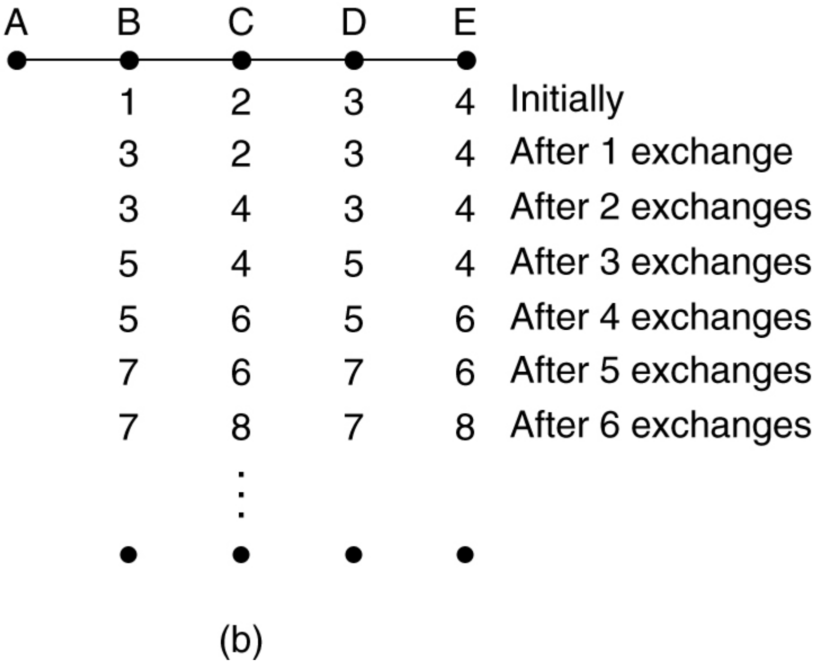
(b)

Count-to-infinity problem



A	B	C	D	E	
	•	•	•	•	Initially
	1	•	•	•	After 1 exchange
	1	2	•	•	After 2 exchanges
	1	2	3	•	After 3 exchanges
	1	2	3	4	After 4 exchanges

(a)



A	B	C	D	E	
	1	2	3	4	Initially
	3	2	3	4	After 1 exchange
	3	4	3	4	After 2 exchanges
	5	4	5	4	After 3 exchanges
	5	6	5	6	After 4 exchanges
	7	6	7	6	After 5 exchanges
	7	8	7	8	After 6 exchanges
		⋮			
	•	•	•	•	

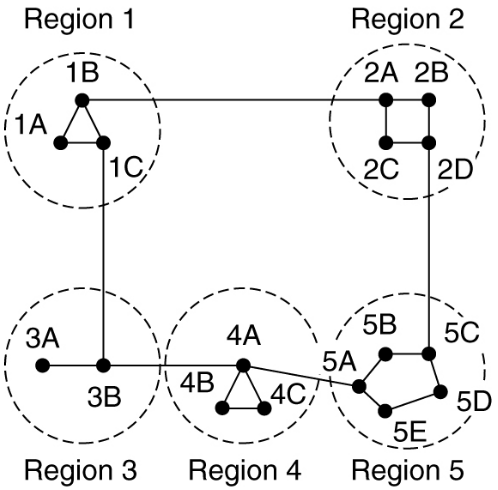
(b)

- Choose a small "infinity"
- Poisoned reverse
  - A: I can reach X through B for cost T
  - A tells B
    - I can reach X for infinity cost, since I reach X through you!
- When "poisoned reverse" fails

Hierarchical routing

- Why hierarchical
  - scalability
- Internet
  - autonomous system (AS)
  - Inter-domain routing
    - distance vector
  - Intra-domain routing
    - distance vector or link state

Hierarchical routing: example



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

Summary

- Routing algorithms
  - Dijkstra algorithm
  - Bellman-Ford algorithm
- Explore further
  - /bin/netstat -r

Next lecture

- Internet routing