ECE 435 – Network Engineering Lecture 11

Vince Weaver

http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

12 October 2016

Announcements

- Back from Memsys.
  Related things: fast memories for network routers. They can't use off the shelf?
  Memory is 100GB/s, starting to use similar to network protocols

- HW#4 was due.

- Will try to get midterms graded ASAP

The Network Layer

- Also “the internet protocol layer”
- Get packets from source to destination
- Critical, and much more complicated than Link Layer
- Connectivity, Scalability, and Resource Sharing problems
- Link any two machines around the world, connect billions of machines

- May require multiple hops
- Link layer: only between ends of wire
- Control Plane – control protocols on how packets processed
  Routing (find a path) and store (update routing info)
- Data Plane – how to process data packets
  forwarding – transfer from incoming network to outgoing

Network Layer Design Issues

- Should be independent of router tech, should hide topology and num, type of routers
- Network addreses should be uniform?
- Need to send packets between any two machines, globally:

1. How to identify a host globally
2. How to connect different networks together
3. How to find a path between two hosts

Internetworking

- Connecting various types of networks (ethernet, 802.11, etc)
- A group of LANs connected togeter is an inter-network, or "Internet"

Addressing

- Address at network layer needs to be global
- Needs to identify its network and the host itself. Hierarchical

Connection vs Connectionless

- Internet: Connectionless
  Network is unreliable. Connectionless. Send/Receive packet primitives. Packet ordering/flow control by higher level
  Each packet carry full destination address, as may travel independently of predecessors

- Telephony: connection
  reliable networks

Routing and Forwarding

- Routing: whch routes to use, find shortest path
- Forwarding: looking up which outgoing line to use
- Characteristics: simplicity/efficiency , robustness, stability, fairness, optimality
- Simplicity: packets stored on routers, efficient resource sharing
  maintain good performance (low delay and packet loss)

- Robustness: cope with changes w/o requiring all jobs stopped and rebooted
- Stability: routing eventually converges on an equilibirum
- Fairness and optimality often conflicting
- Fairness example?
- Unicast routing: point to point
- Multicast routing: one to many or many to many

Routing Algorithm Types

- Nonadaptive: not based on measurement, but computed in advance. Static routing. sysadmin sets them. Do not adapt well if routers fail.

- Adaptive: change routing decisions to reflect changes in topology and traffic
  centralized – require global information
  quasi-centralized (?)
  distributed – ?
  hop-by-hop (internet. source routing?)

Optimal Route?

- What do we optimize? Latency? Throughput? Number of hops?
- Something like ssh might want lowest latency
- Multimedia might want high bandwidth and low jitter
- Often a "cost" is defined based on the desired characteristics, and then this is optimized for

Connectionless

- Packets sometimes called Datagrams
- Packets injected into network with no prior setup
- Router responsible for picking how it gets there, routing algorithm
- Router makes "best-effort". Tries to get things there, but if packet gets lost, goes to wrong place, or arrives out of order it doesn't have to do anything about it.

Connection-Oriented

- Virtual circuit created
- Avoid creating a new route for every packet
- A route from source to destination created in all routers along the way
- Each packet carries an ID saying what route it belongs to

Tradeoffs

- Setup: none / required
- Addressing: full source + dest / short virt circuit num
- State: no router state / each virt circuit has state
- routing: each packet independent / routing done at startup
- router failure: can route around / all virt circuits terminated

- QoS: difficult / easy if resources allocated in advance
- congestion: difficult / easy of allocated in advance

Optimality Principle

- If J is on optimal path from I to K, then optimal J to K is on same route
- Set of all optimal routes from all sources to a destination form a tree rooted at desitnation, called a "sink tree". Not necessarily unique
- Tree and not a loop, so packets delivered in finite number of hops
- Though routers can come and go so things can go wrong

(static) Shortest Path Routing

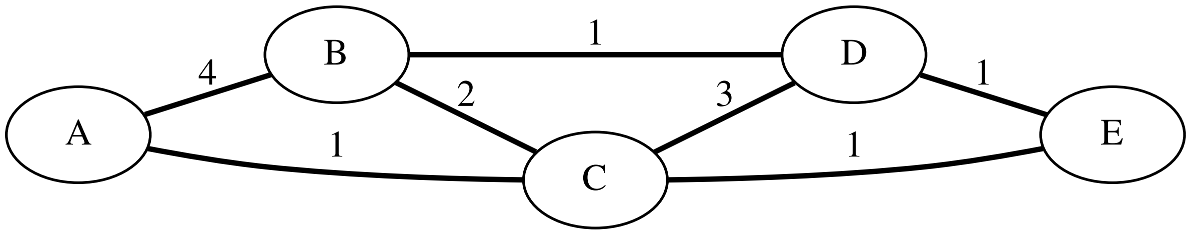
- Number of hops?
- Length (in meters?)
- Transmission delay?

(static) Link State Routing

- Requires global information, routers broadcast the info so all have consistent view

- Dijkstra Algorithm
  Form least spanning tree
  Find lowest cost iteratively

- Example based on one from Lin/Hwang/Baker



T=set of known machines, C(X)=cost of X, p(X)=previous hop

Iteration	T	C(B),p(B)	C(C),p(C)	C(D),p(D)	C(E),p(E)
0	A	4,A	1,A	∞	∞
1	AC	3,C		4,C	2,C
2	ACE	3,C		3,E	
3	ACEB			3,E	
4	ACEBD				

Iterative algorithm. Start not knowing anything but direct connections. Pick shortest cost and add to set. Update all the link costs. Repeat until all nodes added.

Final routing table for A.

Path	Cost	Next
A-B	3	C
A-C	1	C
A-D	3	C
A-E	2	C

(static) Flooding

- Every packet sent out on every outgoing line, with a counter (set to the distance) so after so many hops discarded

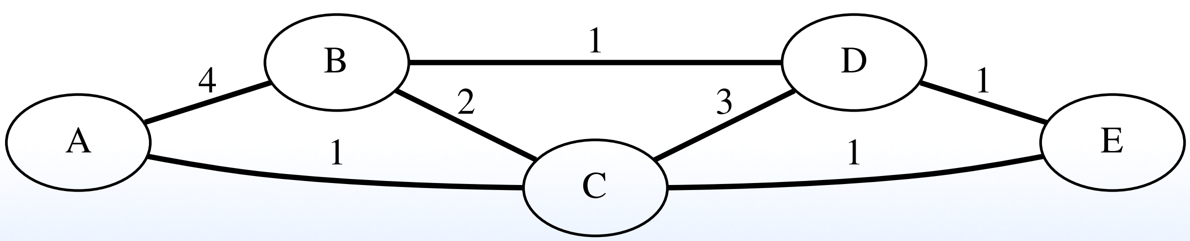
- Selective flooding, only floods out the connections going in vaguely the right direction

- Very robust (can handle if routers droppign out constantly)

- Flooding alwys chooses shortest path, as it finds all possible paths in parallel

(dynamic) Distance Vector Routing

- Used by ARPANET until 1979
- Each router maintains a table (vector) giving best known distance to each destination and line to use to get there
- Count to Infinity problem –



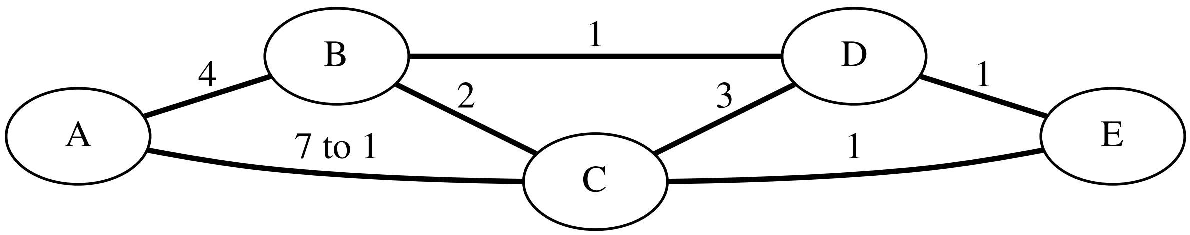
	A D	C	N	B D	C	N	C D	C	N	D D	C	N	E D	C	N
1	B C	4 1	B C	A C D	4 2 1	A C D	A B D E	1 2 3 1	A B D E	B C E	1 3 1	B C E	C D	1 1	C D
2	B C D E	3 1 4 2	C C C C	A C D E	3 2 1 2	C C D D	A B D E	1 2 2 1	A B E E	A B C E	4 1 2 1	C B E E	A B C D	2 2 1 1	C D C D
3	B C D E	3 1 3 2	C C C C							A B C E	3 1 2 1	E B E E			

1. Start with what you know

2. Send routing table to neighbor
3. Update if find shorter route. This is all happening simultaneously
4. Should convertge on Dijkstra.

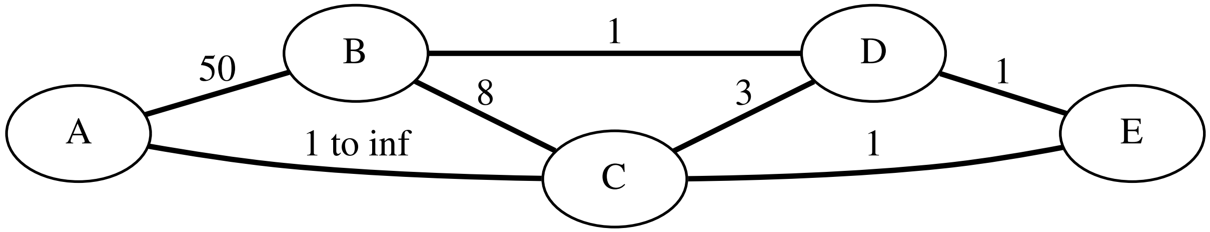
- Looping problems: packets can get stuck in loops.

- Good news travels fast.



Converges in two steps.

- Bad news travels slowly.



A to C line goes down. Have bad timing.
C thinks fastest to A is E, E thinks fastest to A is still C.
C tells B+E cost to A is inf. E (old) tells C cost to A still 2.
C updates with this info, path to A is 3 if go via E
E updates path to A is 4 if go via C
slowly loop, “counting to infinity”

- Solutions to count to infinity
  - Split horizon – a router should not tell neighbor back the least cost it just got from that neighbor
  - Poison Reverse – instead of not telling back, should say the cost back to itself is infinity
  - These only work for two hop loops. Other options to send additional “next hop” data, or have a “hold down timer” that lets things settle before updating info

(dynamic) Link State Routing

- Problems with DVR: did not take delay into account, took too long to converge

- Each router must:

1. Discover neighbors and learn network address
2. Measure delay or cost of each neighbor
3. Construct a packet telling all it learned
4. Send a packet to all other routers
5. Compute the shortest path to all other routers

- Learning about neighbors: sends HELLO packet at boot out all links
- Measure line cost: Send special ECHO paket and measure return. Take into account load?
- Building link-state packets
- Distrbiuting
- Computing new routes

Hierarchical Routing

- At some point not possible for every router to know about every other
- Split into regions
- Example?