

Lecture 14:

Distance-vector Routing

CSE 123: Computer Networks
Alex C. Snoeren



Lecture 14 Overview



- Link-state convergence
- Distance vector
 - Assume each router knows its own address and cost to reach each of its directly connected neighbors
- Bellman-Ford algorithm
 - Distributed route computation using only neighbor's info

Making Something Disappear



- Need to remove failed/old links from topology
 - ◆ LSPs carry sequence numbers to distinguish new from old
 - ◆ Routers only accept (and forward) the “newest” LSP
 - ◆ Send a new LSP with link cost infinity to signal a link down
- But also need to remove entire routers
 - ◆ TTL in every LSP, decremented periodically by each router
 - ◆ When TTL = 0, purge the LSP and flood the network with an LSP with TTL 0 to tell everyone else to do the same



When to Flood?

- Triggered by a topology change
 - ◆ Link or node failure/recovery or
 - ◆ Configuration change like an updated link metric
 - ◆ Converges quickly, but can cause flood of updates
- Periodically
 - ◆ Typically (say) every 30 minutes
 - ◆ Corrects for possible corruption of the data
 - ◆ Limits the rate of updates, but also failure recovery

Convergence

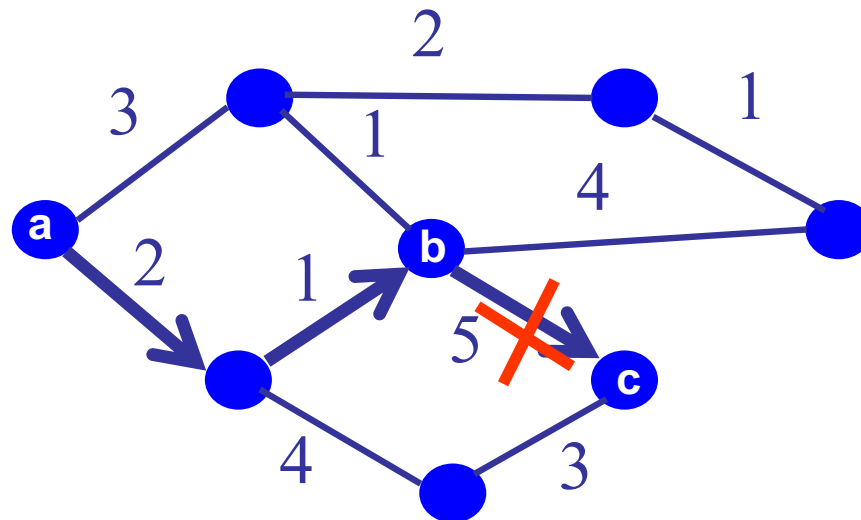


- Getting consistent routing information to all nodes
 - ◆ E.g., all nodes having the same link-state database
 - ◆ Until routing protocol converges, strange things happen...
- Consistent forwarding after convergence
 - ◆ All nodes have the same link-state database
 - ◆ All nodes forward packets on shortest paths
 - ◆ The next router on the path forwards to the next hop



Transient Disruptions

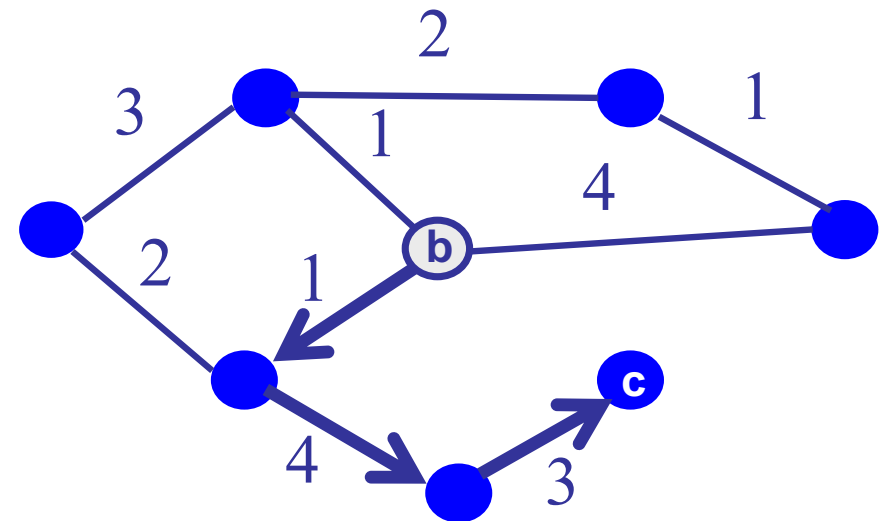
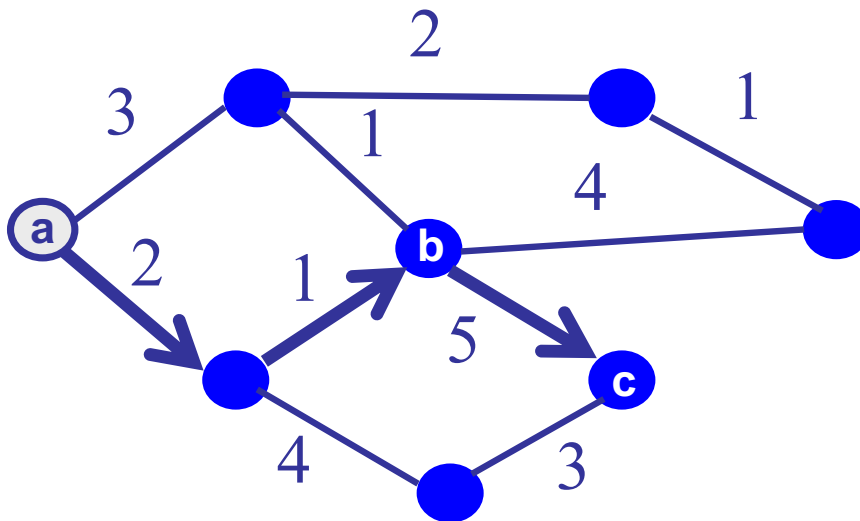
- Detection delay
 - ◆ A node does not detect a failed link immediately
 - ◆ ... and forwards data packets into a **black hole**
 - ◆ Depends on timeout for detecting lost hellos





Transient Disruptions

- Inconsistent link-state database
 - ◆ Some routers know about failure before others
 - ◆ The shortest paths are no longer consistent
 - ◆ Can cause transient forwarding loops



Convergence Delay



- Sources of convergence delay
 - ◆ Detection latency
 - ◆ Flooding of link-state information
 - ◆ Shortest-path computation
 - ◆ Creating the forwarding table
- Performance during convergence period
 - ◆ Lost packets due to black holes and loops (IP TTL will expire)
 - ◆ Looping packets consuming resources
 - ◆ Out-of-order packets reaching the destination
- Very bad for VoIP, online gaming, and video

Reducing Delay



- Faster detection
 - ◆ Smaller hello timers
 - ◆ Link-layer technologies that can detect failures
- Faster flooding
 - ◆ Flooding immediately
 - ◆ Sending link-state packets with high-priority
- Faster computation
 - ◆ Faster processors on the routers
 - ◆ Incremental Dijkstra's algorithm
- Faster forwarding-table update
 - ◆ Data structures supporting incremental updates



Real Link-state Protocols

- OSPF (Open Shortest Path First) and IS-IS
 - ◆ Most widely used intra-domain routing protocols
 - ◆ Run by almost all ISPs and many large organizations
- Basic link state algorithm plus many features:
 - ◆ Authentication of routing messages
 - ◆ Extra hierarchy: Partition into routing areas
 - » “Border” router pretends to be directly connected to all routers in an area (answers for them)
 - ◆ Load balancing: Multiple equal cost routes

Link State evaluation



- Strengths
 - ◆ Loop free as long as LS database's are consistent
 - » Can have transient routing loops – shouldn't last long
 - ◆ Messages are small
 - ◆ Converges quickly
 - ◆ Guaranteed to converge

- Weaknesses
 - ◆ Must flood data across entire network (scalability?)
 - ◆ Must maintain state for entire topology (database)

Distance vector algorithm

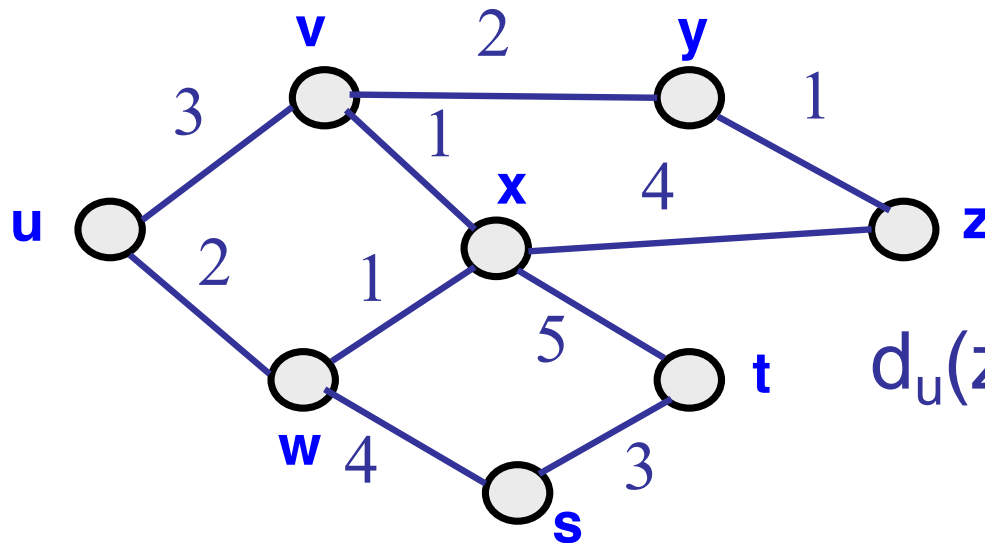


- Base assumption
 - ◆ Each router knows its **own address** and the cost to reach each of its **directly connected neighbors**
- Bellman-Ford algorithm
 - ◆ Distributed route computation using **only neighbor's info**
- Mitigating loops
 - ◆ Split horizon and poison reverse



Bellman-Ford Algorithm

- Define distances at each node X
 - ◆ $d_x(y)$ = cost of least-cost *path* from X to Y
- Update distances based on neighbors
 - ◆ $d_x(y) = \min \{c(x,v) + d_v(y)\}$ over all neighbors V



$$d_u(z) = \min\{c(u,v) + d_v(z), \\ c(u,w) + d_w(z)\}$$



Distance Vector Algorithm

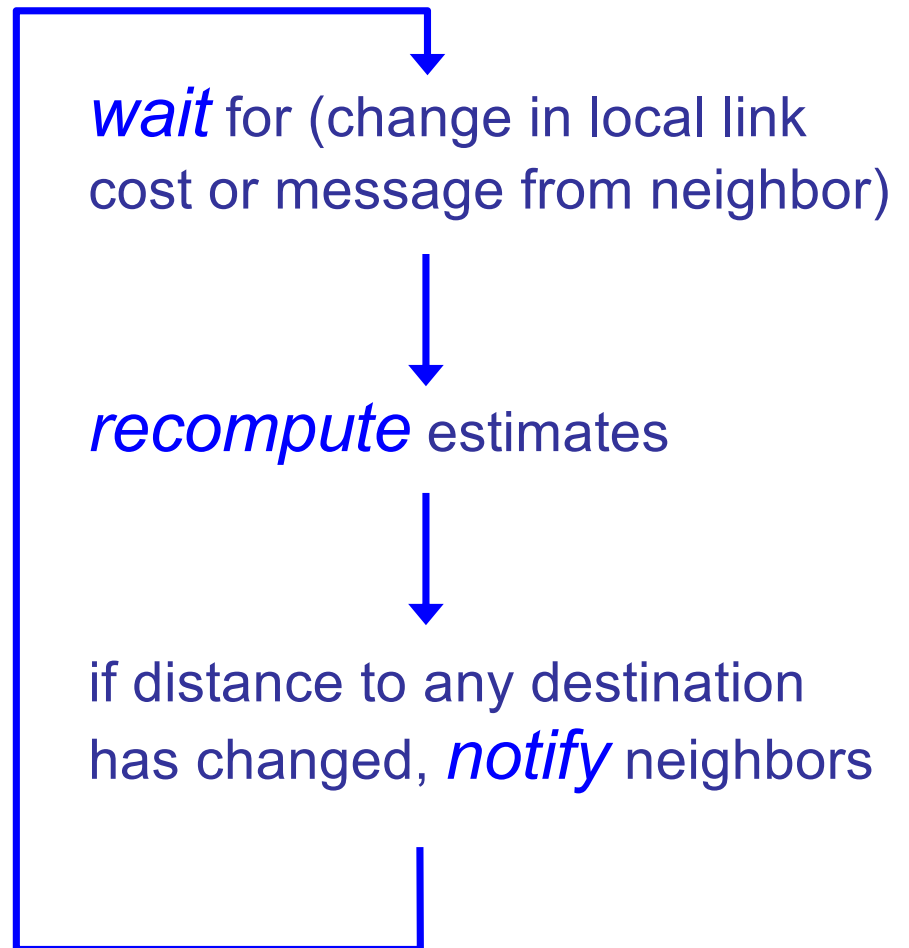
Iterative, asynchronous: each local iteration caused by:

- Local link cost change
- Distance vector update message from neighbor

Distributed:

- Each node notifies neighbors when its DV changes
- Neighbors then notify *their* neighbors if necessary

Each node:



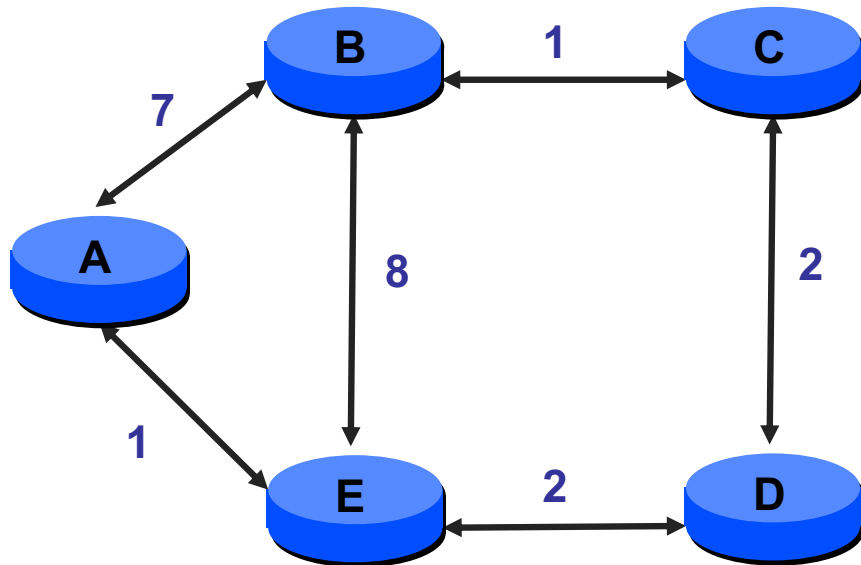
Step-by-Step



- $c(x, v)$ = cost for direct link from x to v
 - ◆ Node x maintains costs of direct links $c(x, v)$
- $D_x(y)$ = estimate of least cost from x to y
 - ◆ Node x maintains distance vector $\mathbf{D}_x = [D_x(y): y \in N]$
- Node x maintains its neighbors' distance vectors
 - ◆ For each neighbor v , x maintains $\mathbf{D}_v = [D_v(y): y \in N]$
- Each node v periodically sends D_v to its neighbors
 - ◆ And neighbors update their own distance vectors
 - ◆ $D_x(y) \leftarrow \min_v \{c(x, v) + D_v(y)\}$ for each node $y \in N$



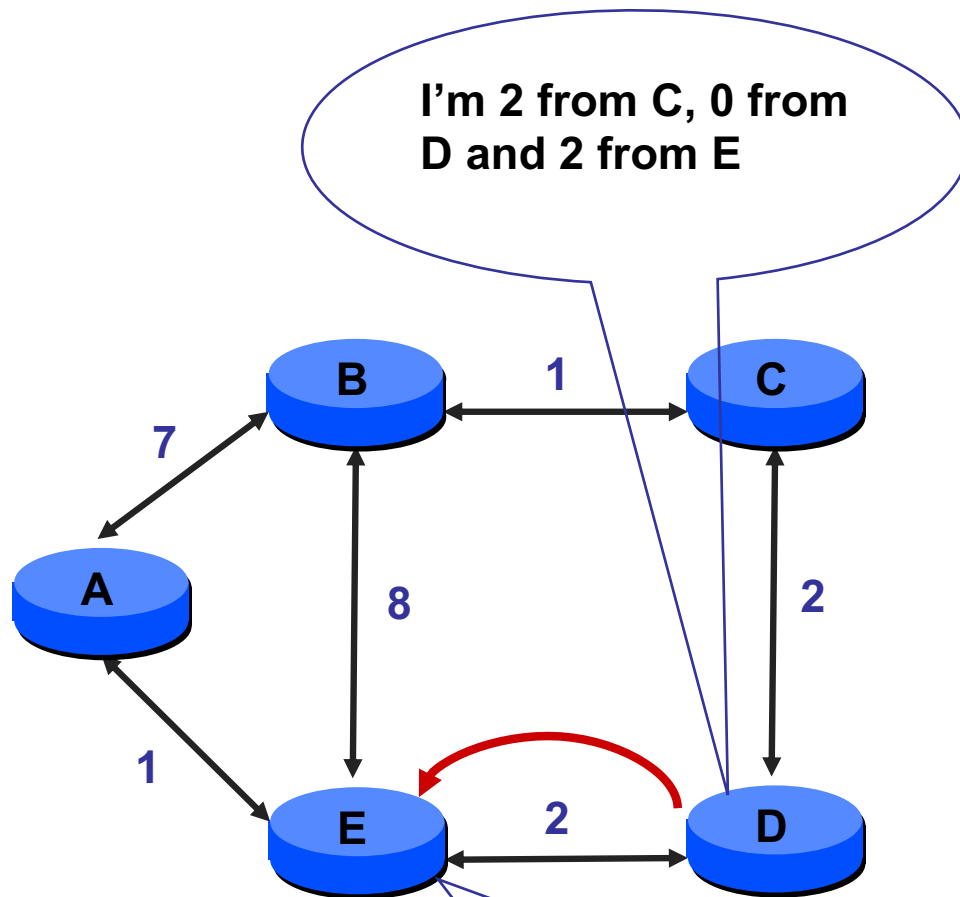
Example: Initial State



Info at node	Distance to Node				
	A	B	C	D	E
A	0	7	∞	∞	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	∞	2	0



D sends vector to *E*



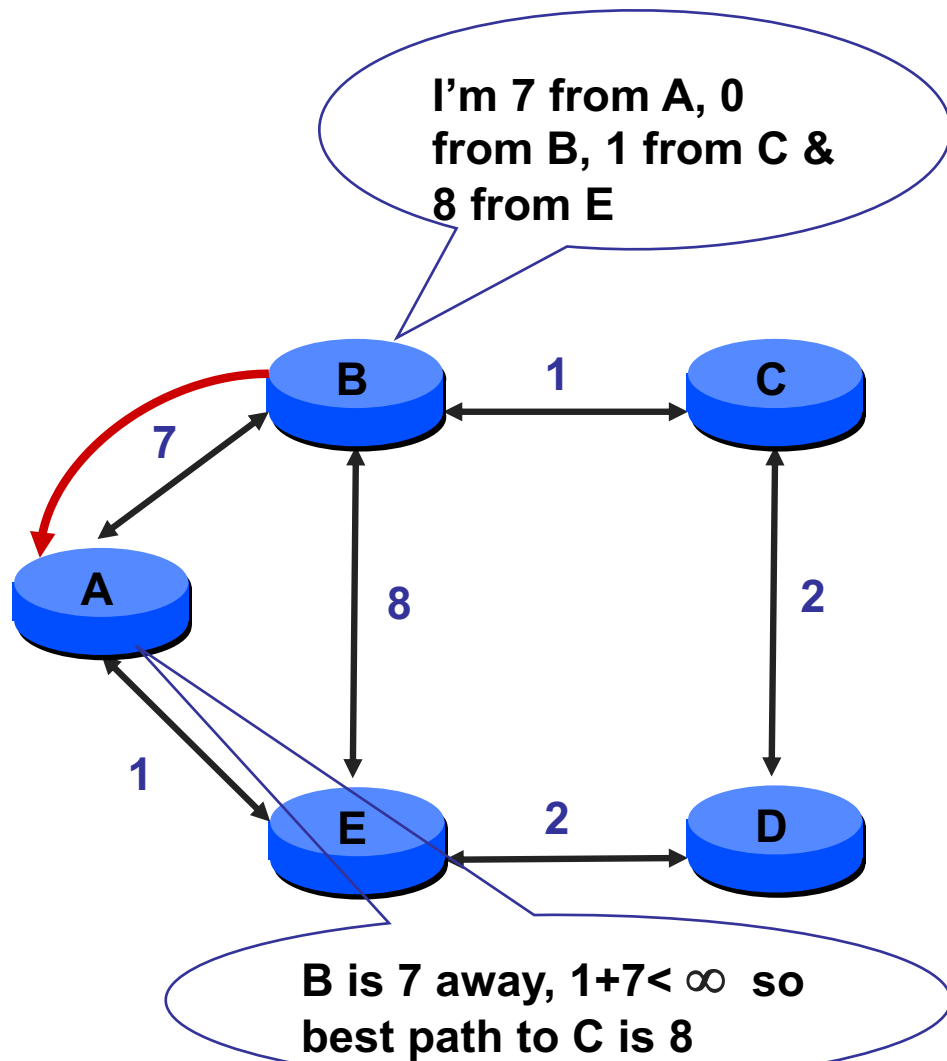
I'm 2 from C, 0 from D and 2 from E

D is 2 away, $2+2 < \infty$,
so best path to C is 4

Info at node	Distance to Node				
	A	B	C	D	E
A	0	7	∞	∞	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	4	2	0



B sends vector to A

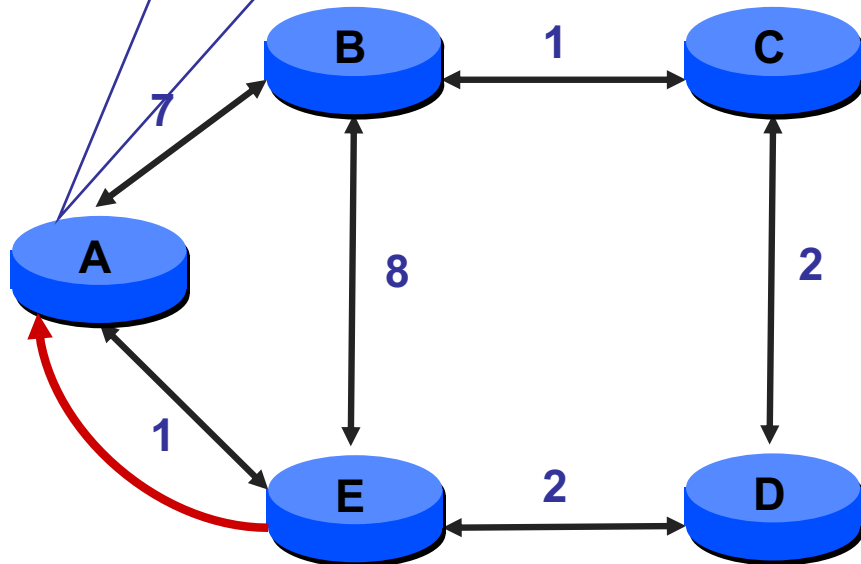


Info at node	Distance to Node				
	A	B	C	D	E
A	0	7	8	∞	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	4	2	0



E sends vector to A

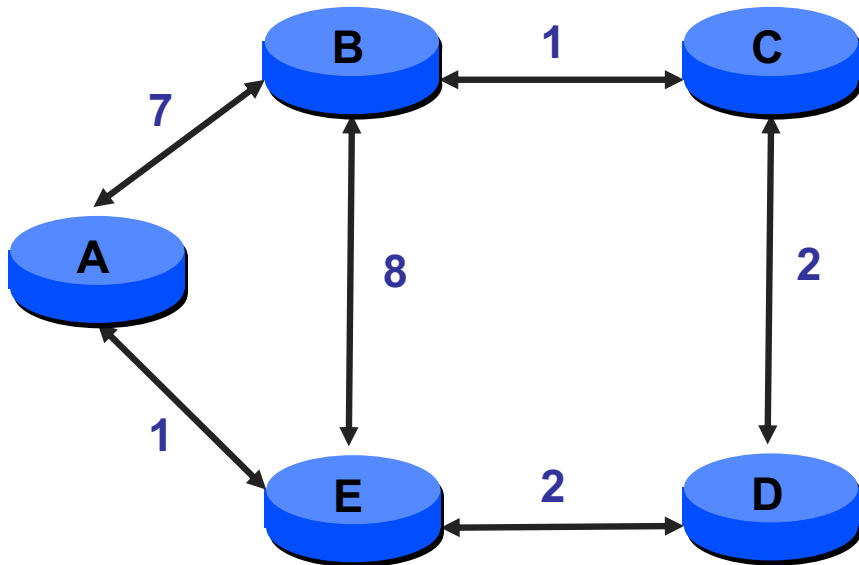
E is 1 away, $4+1 < 8$
so C is 5 away, $1+2 < \infty$
so D is 3 away



I'm 1 from A, 8 from B, 4 from C, 2 from D & 0 from E

Info at node	Distance to Node				
	A	B	C	D	E
A	0	7	5	3	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	4	2	0

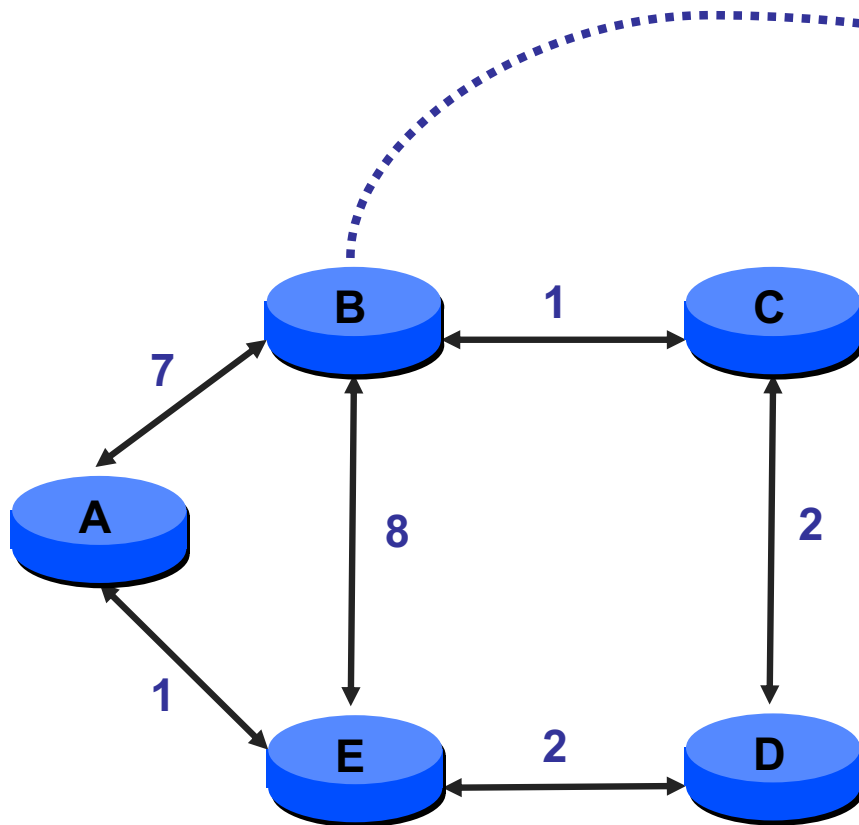
...until Convergence



Info at node	Distance to Node				
	A	B	C	D	E
A	0	6	5	3	1
B	6	0	1	3	5
C	5	1	0	2	4
D	3	3	2	0	2
E	1	5	4	2	0



Node *B*'s distance vectors



Dest	Next hop		
	A	E	C
A	7	9	6
C	12	12	1
D	10	10	3
E	8	8	5

For next time...



- Project due TONIGHT
- Midterm WEDNESDAY
 - ◆ 1 8.5x11" sheet of paper with whatever you want
 - ◆ Must turn it in WITH your exam
- Read Ch. 3.4 in P&D for Friday