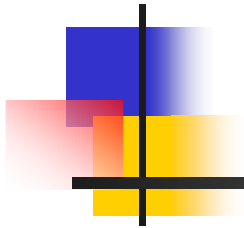


Network Layer Overview

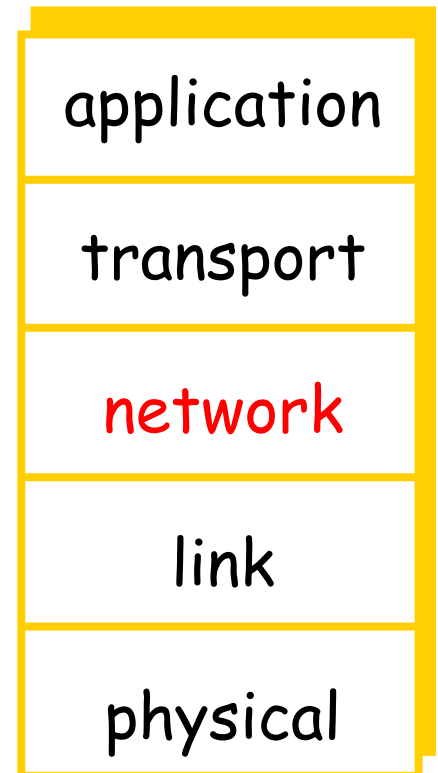


Dept. of Computer Science, University of Rochester

Internet Architecture

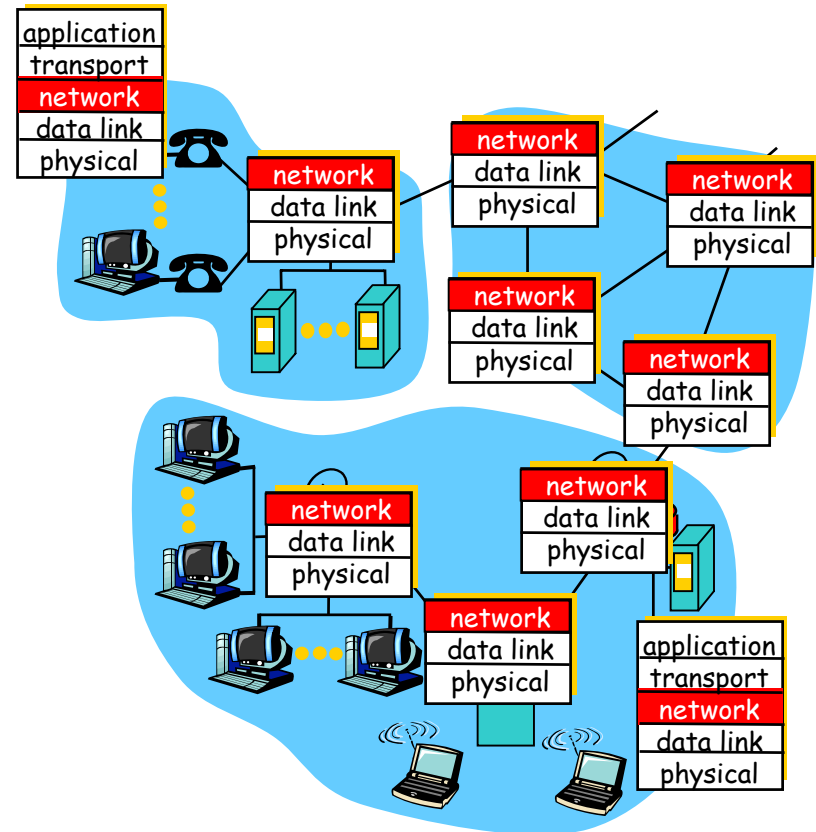
Bottom-up:

- **physical:** electromagnetic signals "on the wire"
- **link:** data transfer between neighboring network elements
 - encoding, framing, error correction, access control for shared links
- **network:** host-to-host connectivity
 - routing, addressing
- **transport:** host-host data transport
 - reliable data transport, congestion control, flow control
- **application:** anything you want to do on computer networks



Fundamental Network Layer Function

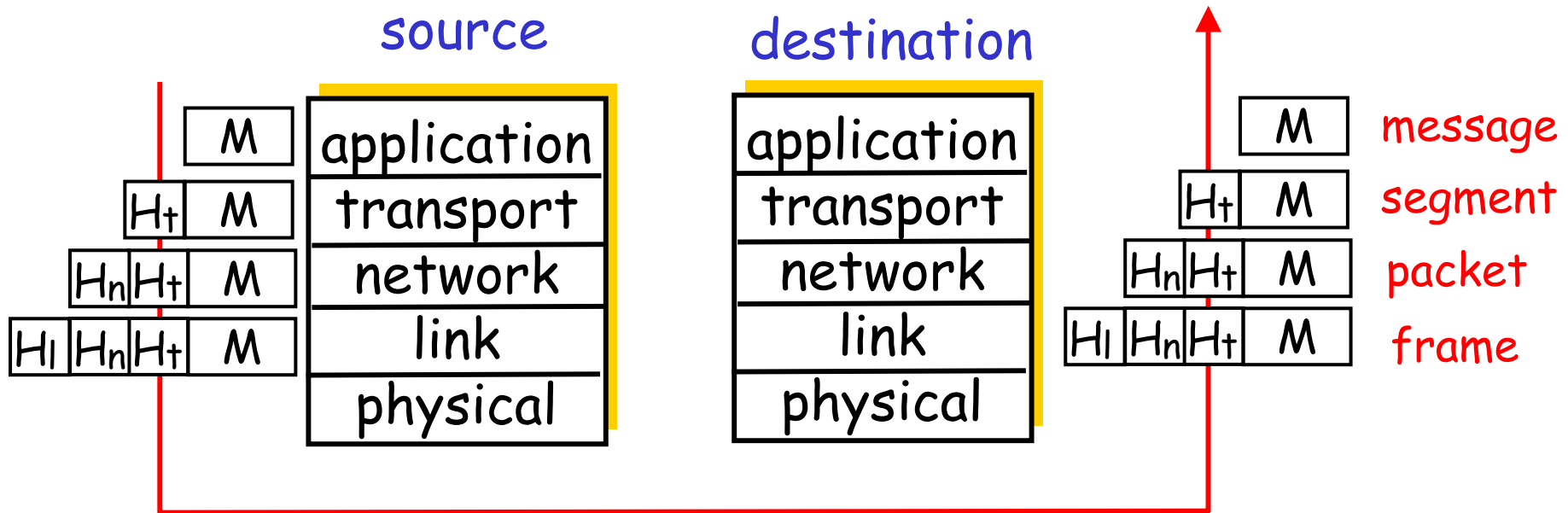
- **fundamental function:** transport packet from sending to receiving hosts
 - **addressing:** uniquely identify each node in the network
 - **routing:** determine a path from source to dest and route packets along the path
- network layer protocols in every host, router



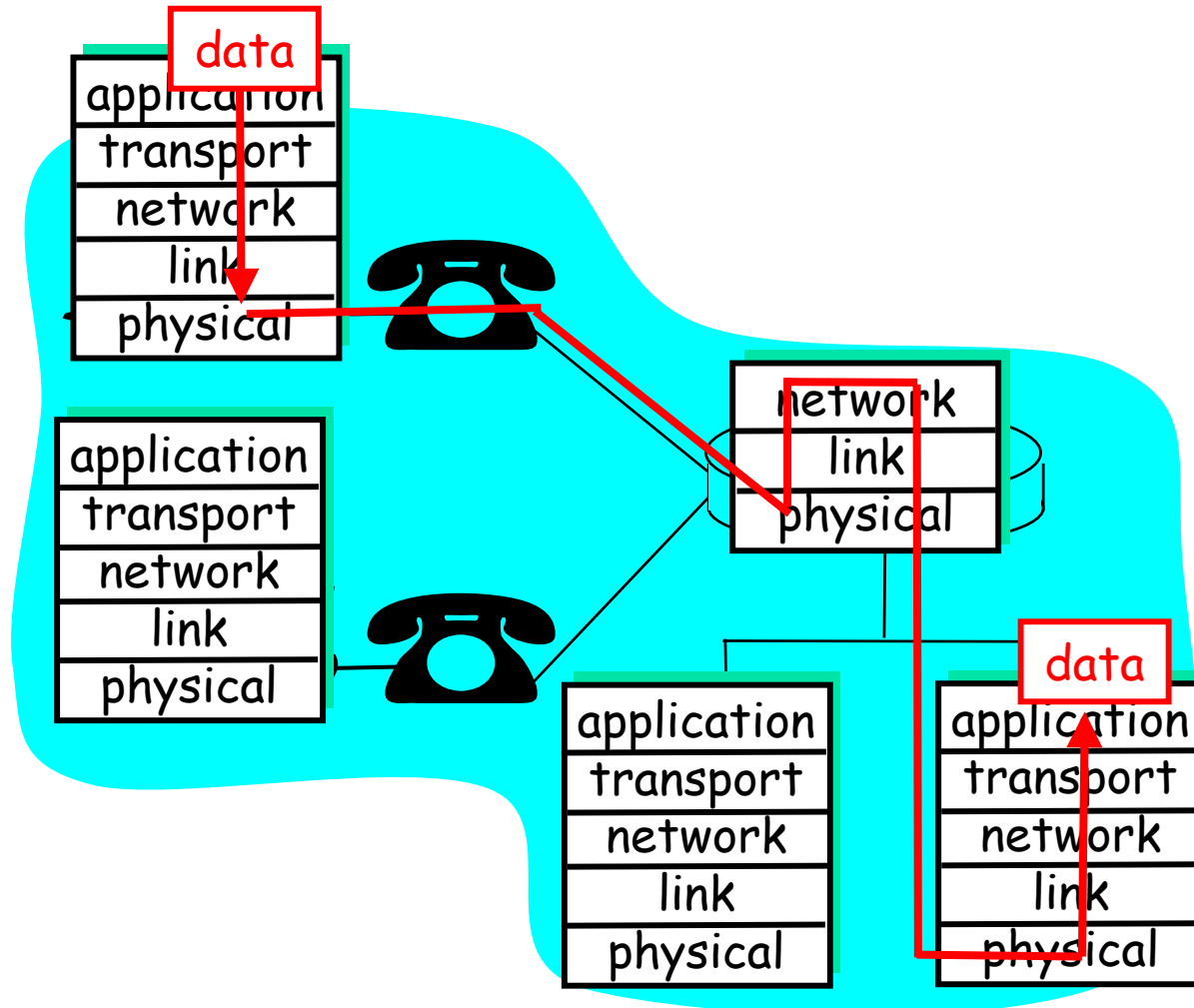
Protocol Layering and Data

Each layer takes data from above

- adds header information to create new data unit
- passes new data unit to layer below



Data Flow across Layers





Network Service Model

Q: What service model is needed in transporting packets from sender to receiver?

- guaranteed bandwidth?
- preservation of inter-packet delay (no jitter)?
- loss-free delivery?
- in-order delivery?
- congestion feedback to sender?

virtual circuit or
packet switching?

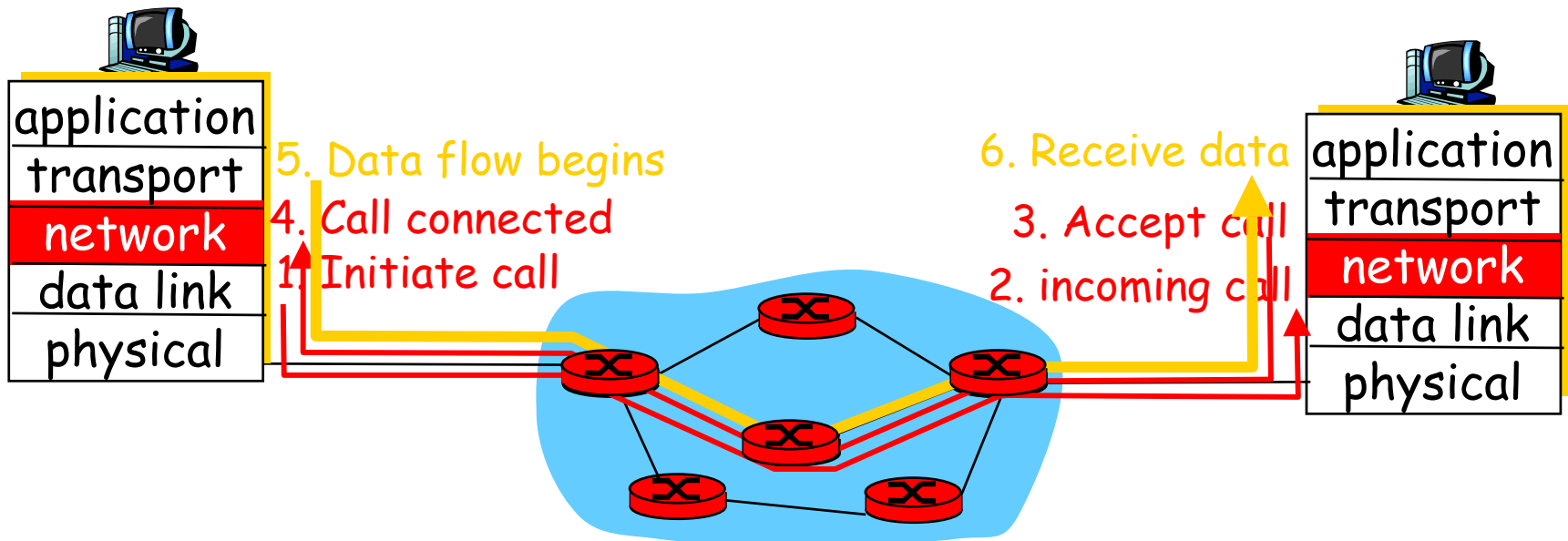
"Analogy"

Q: What service model is needed in air travel (transporting people from location to location)?

- on-time arrival?
- no lost baggage?
- may get bumped out of flights due to overbooking?

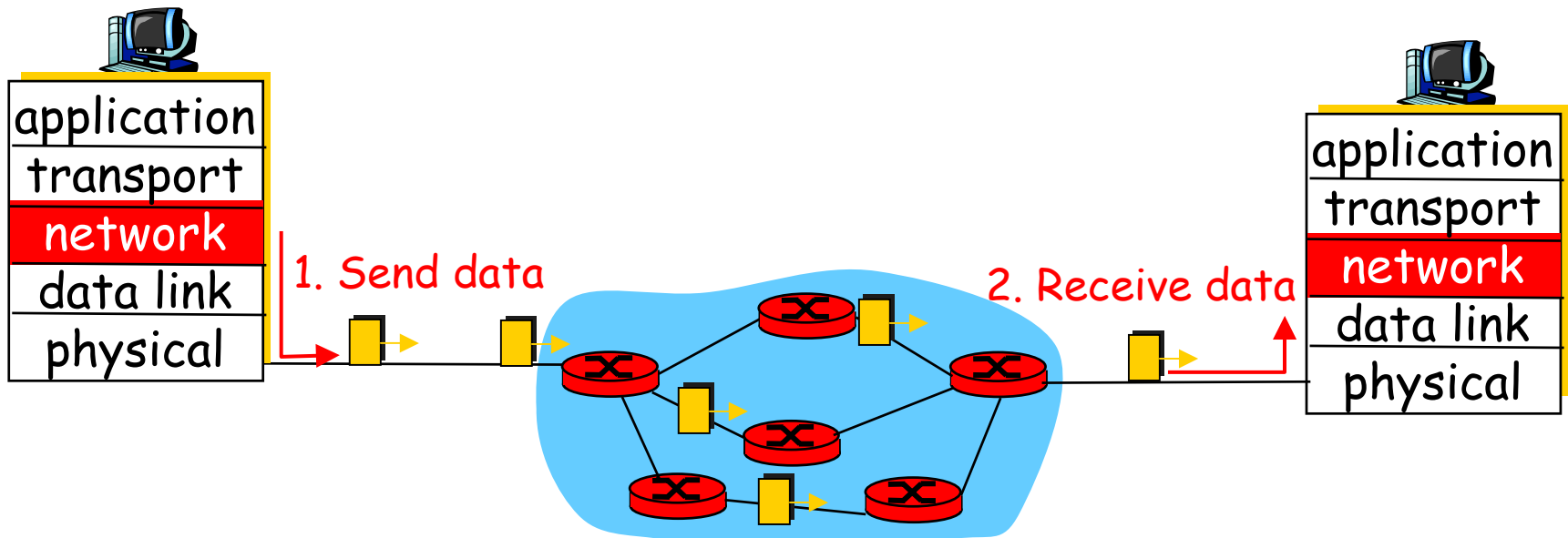
Virtual Circuits

- provide services such as guaranteed bandwidth, jitter-free, in-order delivery, ...
- require signaling protocols to setup, maintain, and teardown virtual circuit
- router maintains state about ongoing connections



Packet Switching

- no call setup at network layer
- routers: no state about end-to-end connections
- packets forwarded independently from each other
 - packets between same source-dest pair may take different paths



Packet Switching vs. Virtual Circuits

Packet switching

- poor service guarantee
 - “elastic” service
 - efficiency
 - robustness
 - flexibility
 - simple network core, complexity at “edge”
- ⇒ manage, control, adapt at “smart” end systems (computers)

Virtual circuits

- evolved from telephony
- guaranteed service
 - e.g., human conversation:
 - strict timing, reliability requirements
- hard to evolve
 - complex network core

Internet employs packet switching; virtual circuits are used for networks with “dumb” end terminals - telephones.

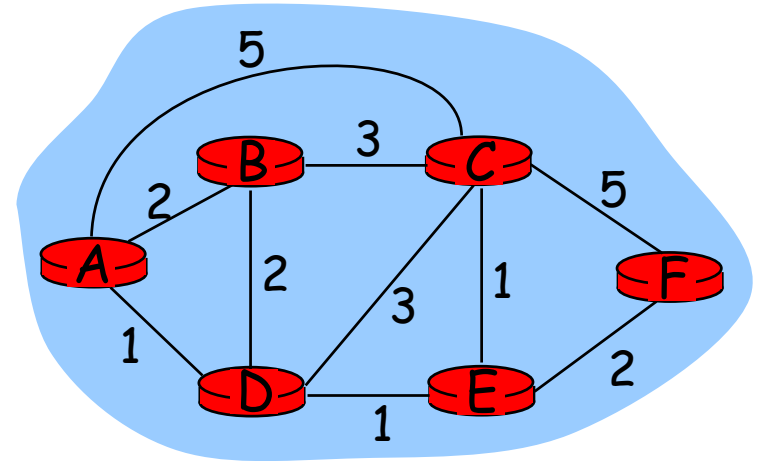
Routing Principles

Routing protocol

Goal: determine "good" path (sequence of routers) thru network from source to dest.

Graph abstraction for routing algorithms:

- graph nodes are hosts or routers
- graph edges are physical links
 - link cost: delay, \$ cost, or congestion level



- "good" path:
 - typically means minimum cost path



Routing Algorithm Classification

Global information:

- all routers have complete topology, link cost info
- "link state" algorithm

Decentralized:

- router knows physically-connected neighbors, link costs to neighbors
- exchange of info with neighbors, may take many rounds to converge
- "distance vector" algorithm



A Link-State Routing Algorithm

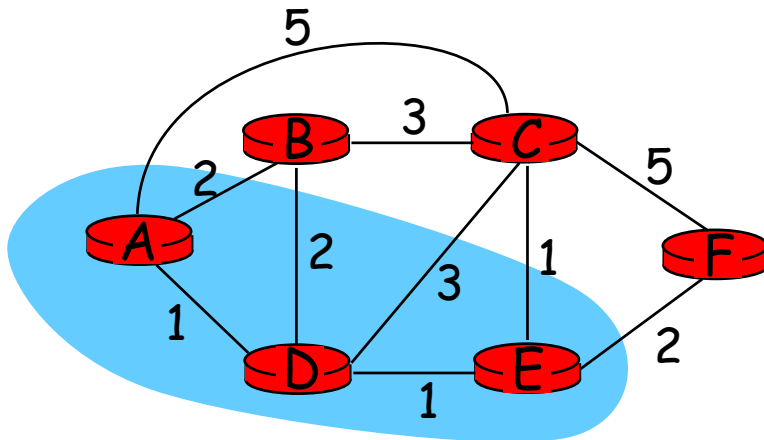
Dijkstra's algorithm

- Network topology, link costs/distances known to all nodes
 - accomplished via "link state broadcast"
 - all nodes have same info
- The algorithm calculates the shortest paths from one specific node ("source") to all other nodes
- Basic approach: gradually expands a set "N", containing nodes whose shortest paths from the source are known
 - initially "N" contains only the source itself
 - at every step one more node's shortest path from the source is learned, this node is then added to "N"
 - eventually "N" includes every node and the algorithm terminates

Dijkstra's Algorithm (cont.)

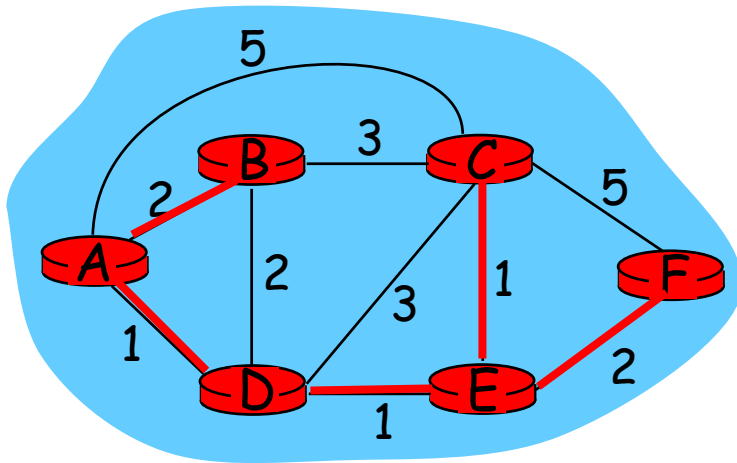
Key step: How to expand "N" to include one more node?

- For each v not in N
 - currently known paths are paths from source to v whose intermediaries are all in N
 - $D(v)$ as the distance of current known shortest path
- For a node with shortest $D(v)$ among all v 's not in N , its currently known shortest path is its globally shortest path



A is the source.
N currently contains
A, D, E.

Dijkstra's Algorithm: An Example



- N : set of nodes whose shortest paths are currently known
- $D(v)$: distance for current known shortest path from source to dest. v
- $p(v)$: predecessor node along path from source to dest. v

Step	N	$D(B), p(B)$	$D(C), p(C)$	$D(D), p(D)$	$D(E), p(E)$	$D(F), p(F)$
→ 0	A	2, A	5, A	1, A	infinity	infinity
→ 1	AD	2, A	4, D		2, D	infinity
→ 2	ADE	2, A	3, E			4, E
→ 3	ADEB		3, E			4, E
→ 4	ADEBC					4, E
5	ADEBCF					

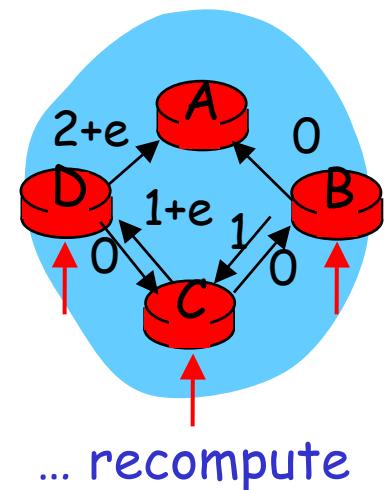
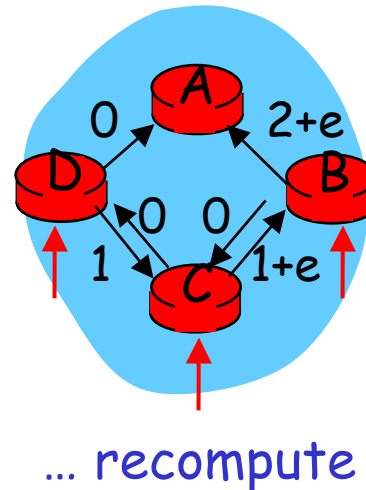
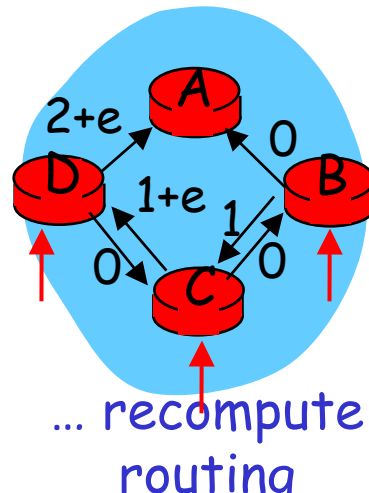
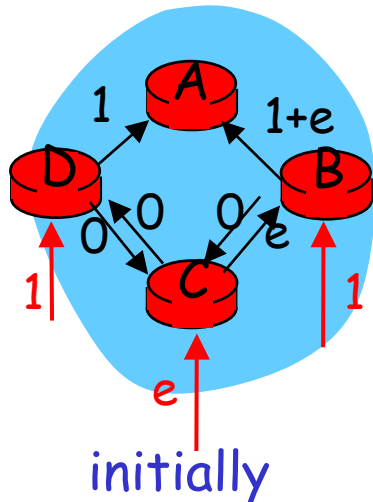
Dijkstra's Algorithm: Discussion

Algorithm complexity: n nodes

- each iteration: need to check all nodes, w , not in N
- $n*(n+1)/2$ comparisons: $O(n^2)$
- using Fibonacci heap to find minimum distance node: $O(n \log n + e)$

Oscillations possible:

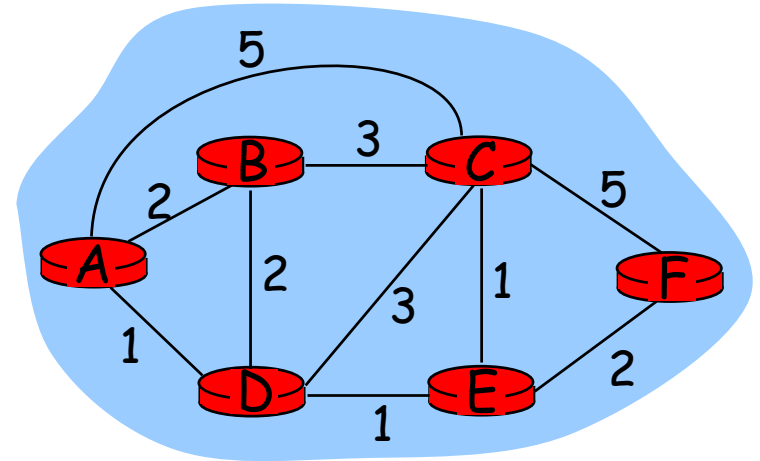
- e.g., link cost = amount of carried traffic
- solution: stable cost metric; asynchronous (random-time) link cost collection



Network Routing

Routing protocol

Goal: determine "good" path (sequence of routers) thru network from source to dest.



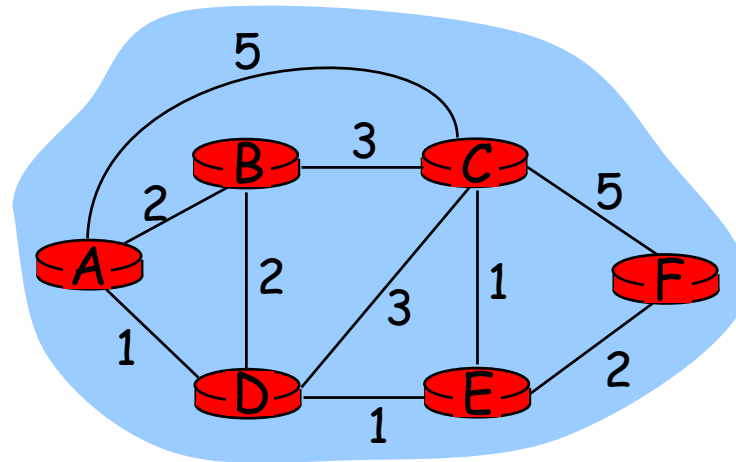
Global information:

- all routers have complete topology, link cost info
- "link state" algorithm

Decentralized:

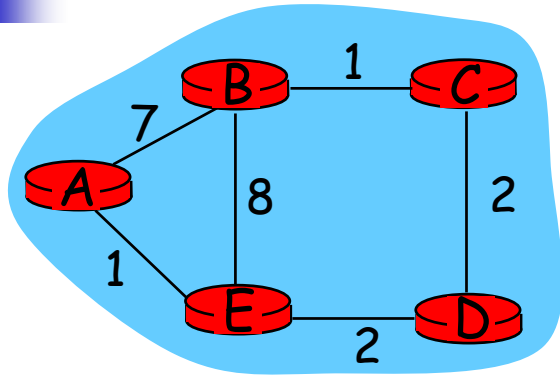
- router knows physically-connected neighbors, link costs to neighbors
- routers exchange of info with neighbors

Distance Vector Routing



- routing table (at each host): the next hop for each destination in the network
- distance vector routing: the routing table is constructed from a distance vector at each node
- distance vectors can be maintained in a decentralized fashion

Distance Vector: An Example



$$D(E, D, C) = c(E, D) + \text{shortest}(D, C) \\ = 2 + 2 = 4$$

$$D(E, D, A) = c(E, D) + \text{shortest}(D, A) \\ = 2 + 3 = 5 \text{ loop!}$$

$$D(E, B, A) = c(E, B) + \text{shortest}(B, A) \\ = 8 + 6 = 14 \text{ loop!}$$

Distance vector at node E

		via first hop x		
D (E,x,y)		A	B	D
destination y	A	1	14	5
	B	7	8	5
	C	6	9	4
	D	4	11	2

Distance Vector to Routing Table

		cost to destination via				
		A	B	D	Outgoing link to use, cost	
destination	D ^E ()					
	A	1	14	5	A	A,1
	B	7	8	5	B	D,5
	C	6	9	4	C	D,4
	D	4	11	2	D	D,4

Distance vector  Routing table



Disclaimer

- Parts of the lecture slides contain original work of James Kurose, Larry Peterson, and Keith Ross. The slides are intended for the sole purpose of instruction of computer networks at the University of Rochester. All copyrighted materials belong to their original owner(s).