Introduction to Computer Networks

CS640

Distance Vector Routing

https://pages.cs.wisc.edu/~mgliu/CS640/F22/

Ming Liu
mgliu@cs.wisc.edu

Today

Last lecture

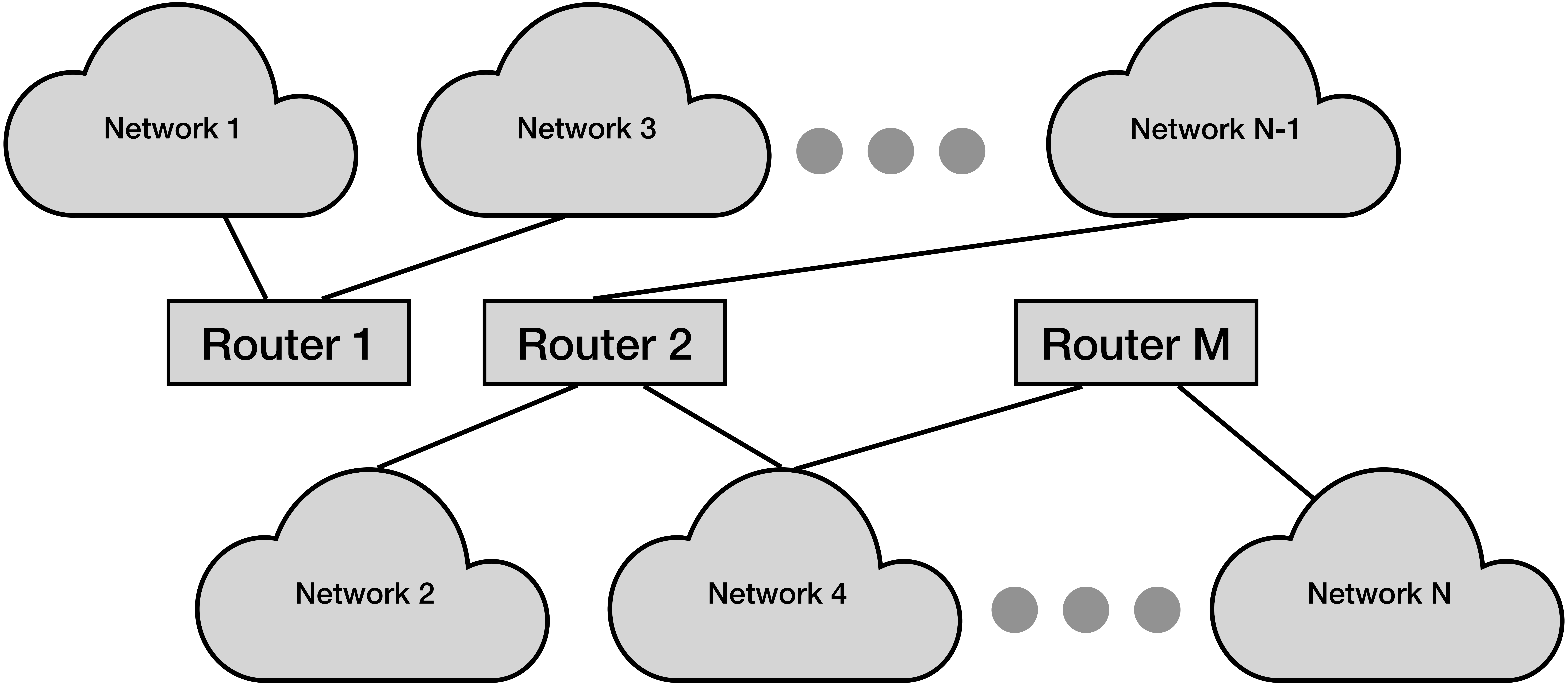
- How to assign an IP address?

Today

- How to decide the forwarding path among routers?

Announcements

- Lab3 is due 11/04/2022, 11:59 PM


Network 1
Network 3
Network N-1
Router 1
Router 2
Router M
Network 2
Network 4
Network N

The Goal of Routing

Build forwarding tables at a router to achieve both network connectivity and some design goals

Routing v.s. Forwarding

Routing

- Process by which routing table is built

Forwarding

- To select an output port based on destination address and routing table

Routing Table v.s. Forwarding Table

Routing table

- Built by the routing algorithm as a precursor to building the forwarding table
- Contain the mapping from network numbers to next hops

Forwarding table

- Used when a packet is being forwarded among physical ports/interfaces
- Ethernet switch: <MAC address, outgoing port>
- IP router: <Network information (of the next hop), outgoing port>

An Example

Routing Table

Subnet	Subnet Mask	NextHop
128.1.0.0	255.255.128.0	Router 1
128.1.128.0	255.255.128.0	Router 2
2.0.0.0	255.255.255.0	Router 3

Forwarding Table

NextHop	Forwarding Port
Router 1	Port 1
Router 2	Port 2
Router 3	Port 3

An Example

Destination

Routing Table

Subnet	Subnet Mask	NextHop
128.1.0.0	255.255.128.0	Router 1
128.1.128.0	255.255.128.0	Router 2
2.0.0.0	255.255.255.0	Router 3

Forwarding Table

NextHop	Forwarding Port
Router 1	Port 1
Router 2	Port 2
Router 3	Port 3

Q: How to decide the forwarding path among routers?

OR

Q: How to build the routing table?

Q: How to decide the forwarding path among routers?

OR

Q: How to build the routing table?

A: Routing Algorithm/Protocol.

- Represent connected networks as a graph
- Vertices in the graph are routers
- Edges in the graph are links
- Links have communication cost, which can be quantized!

Routing is hard!

#1: Network hardware fabric is dynamic

- Links and routers are failed or added

#2: Network traffic is dynamic

- A routing or link can be overloaded

#3: Cost is dynamic

- The value depends on the physical properties, ongoing traffic load, …

#4: No centralized view

- Protocols should work in a distributed fashion

Technique #1: Static Configuration

For a simple network, we can calculate all shortest (preferred) paths and load them into the non-volatile storage of each router

Drawbacks

- No adaptation
- Unable to scale

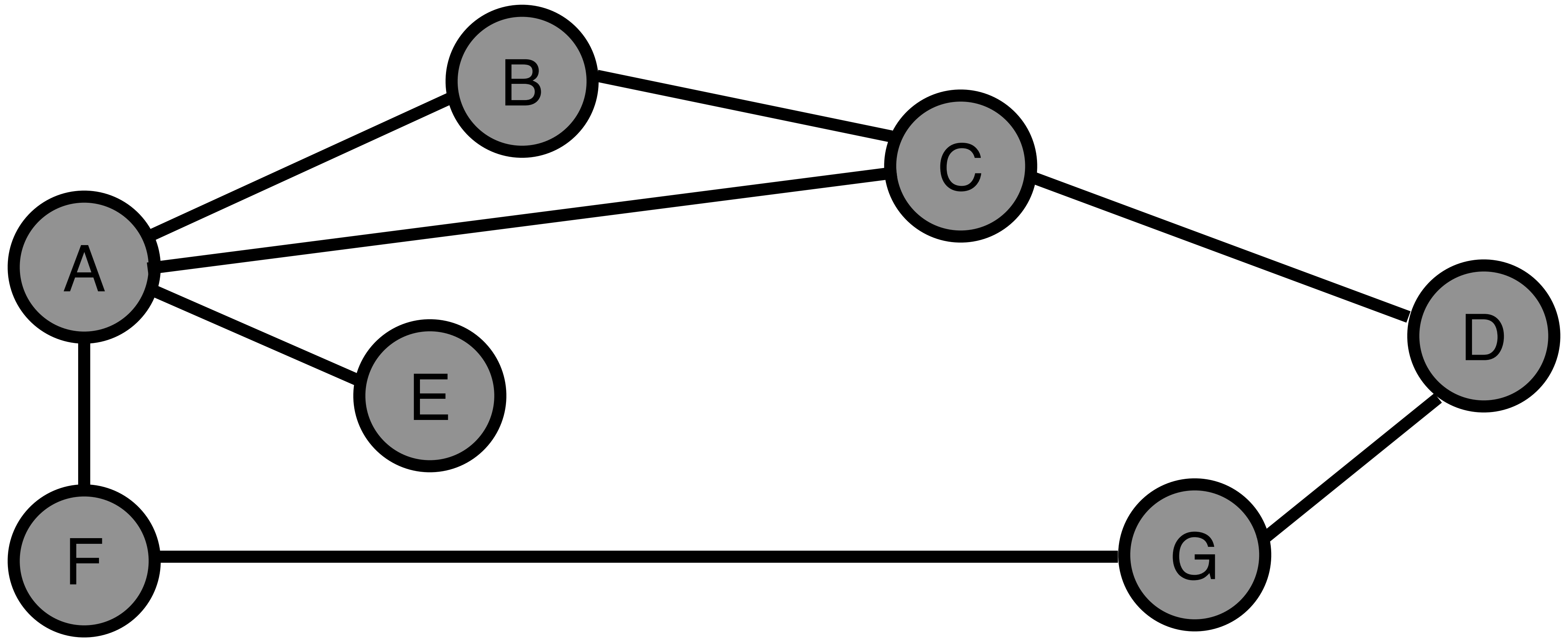
Technique #2: Distance Vector Routing

Key idea: Each node constructs a one-dimensional array (vector) that contains the "distance" (cost) to all other nodes, and distributes that vector to its immediate neighbors

Assumption

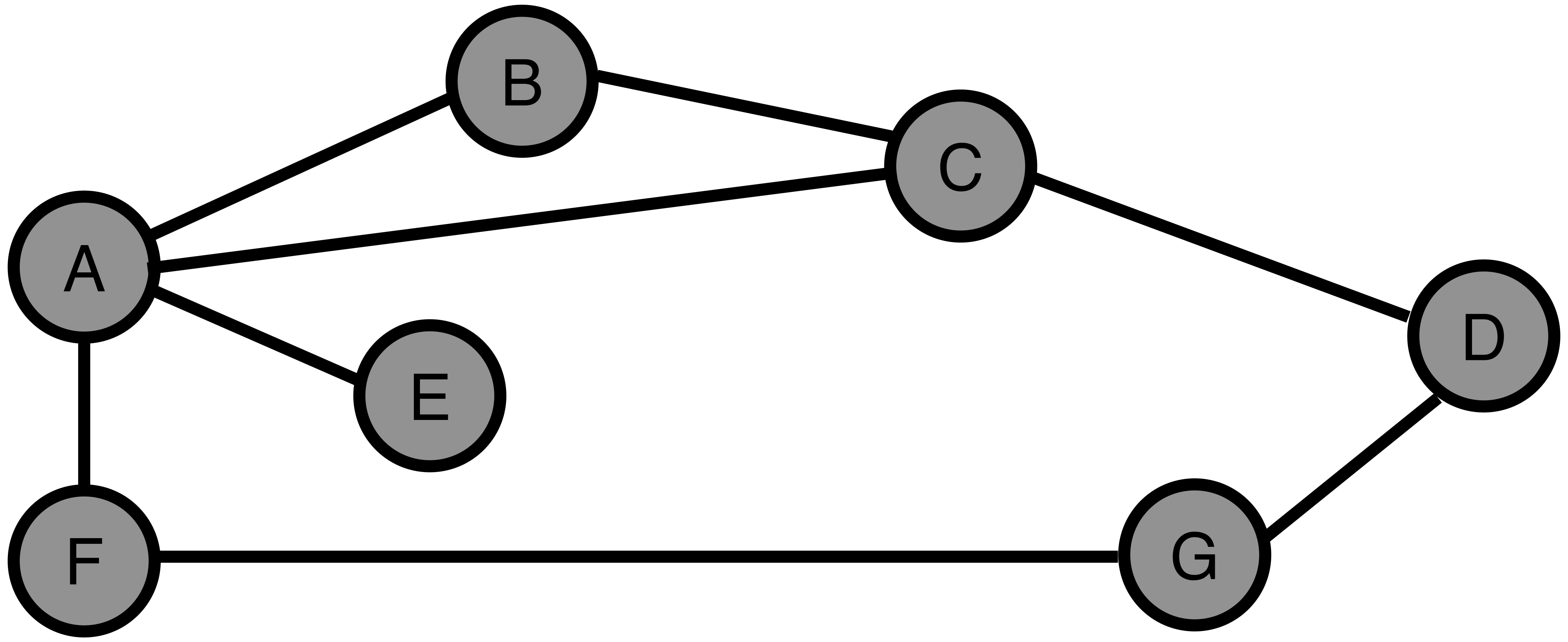
- Each node knows the cost of the link to each of its directly connected neighbors

Distance Vector Protocol



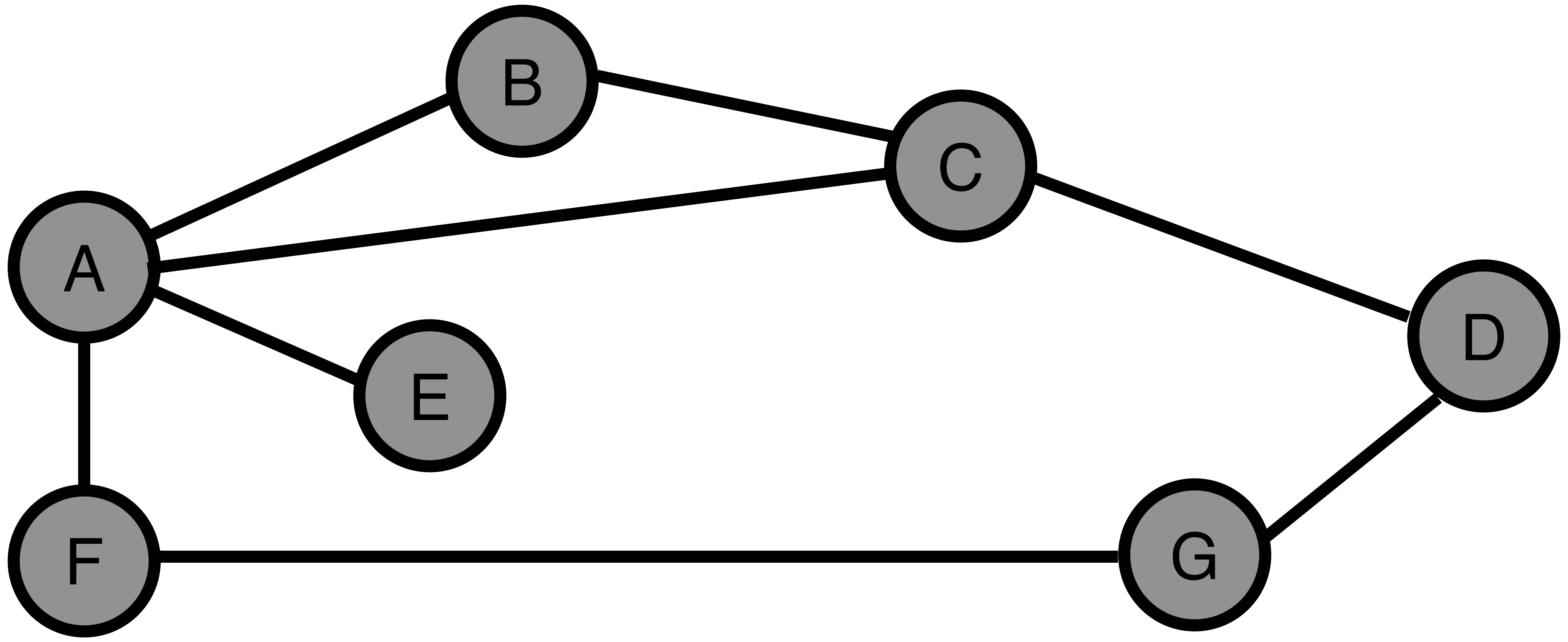
Distance is defined as the number of hops

Step 1: figure out initial distance



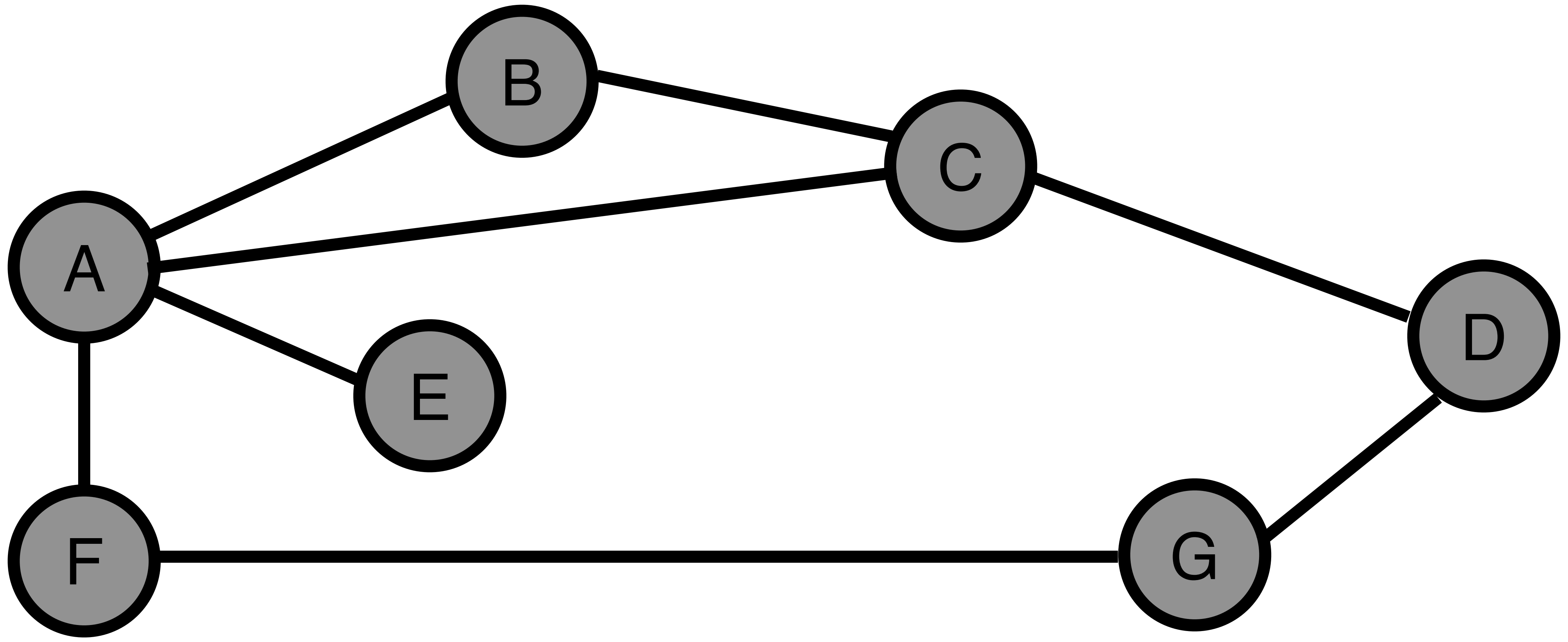
	Distance to Reach Node (Global View)						
	A	B	C	D	E	F	G
A							
B							
C							
D							
E							
F							
G							

Step 1: figure out initial distance



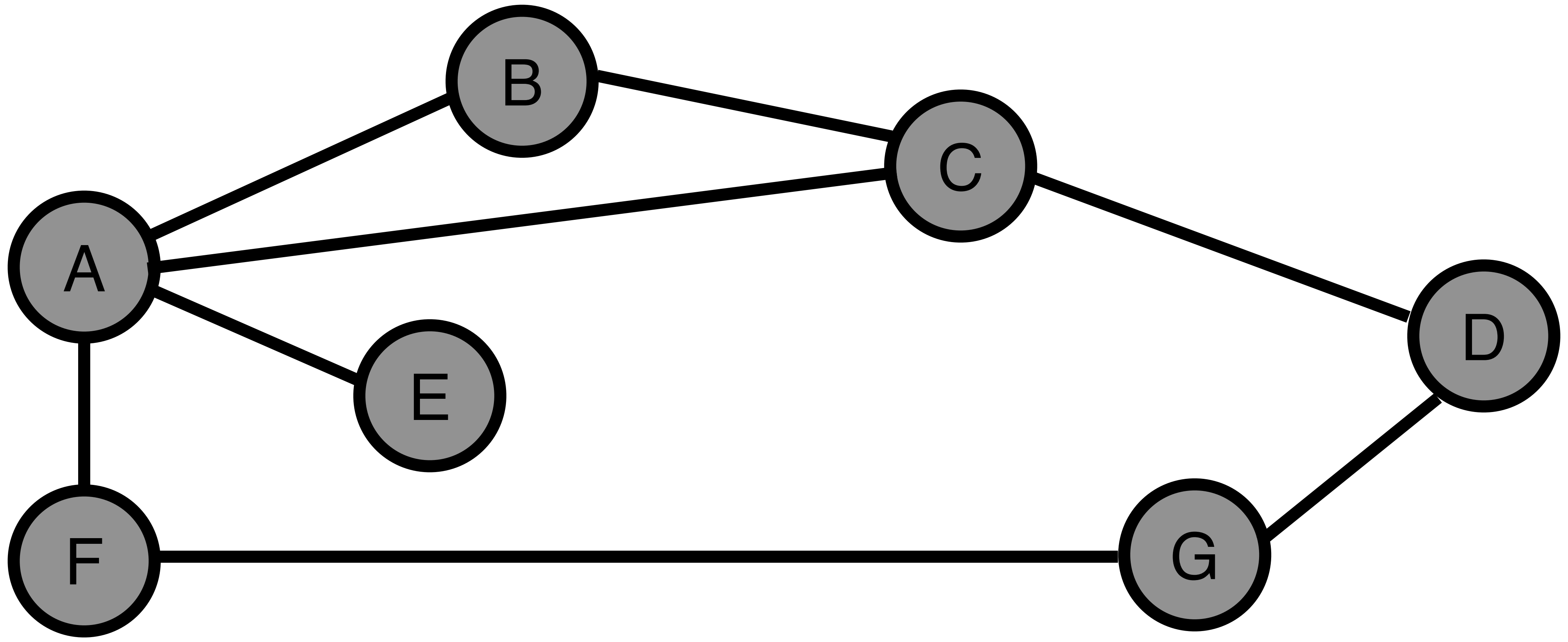
	Distance to Reach Node (Global View)						
	A	**B**	**C**	**D**	**E**	**F**	**G**
A	0	1	1	∞	1	1	∞
B							
C							
D							
E							
F							
G							

Step 1: figure out initial distance



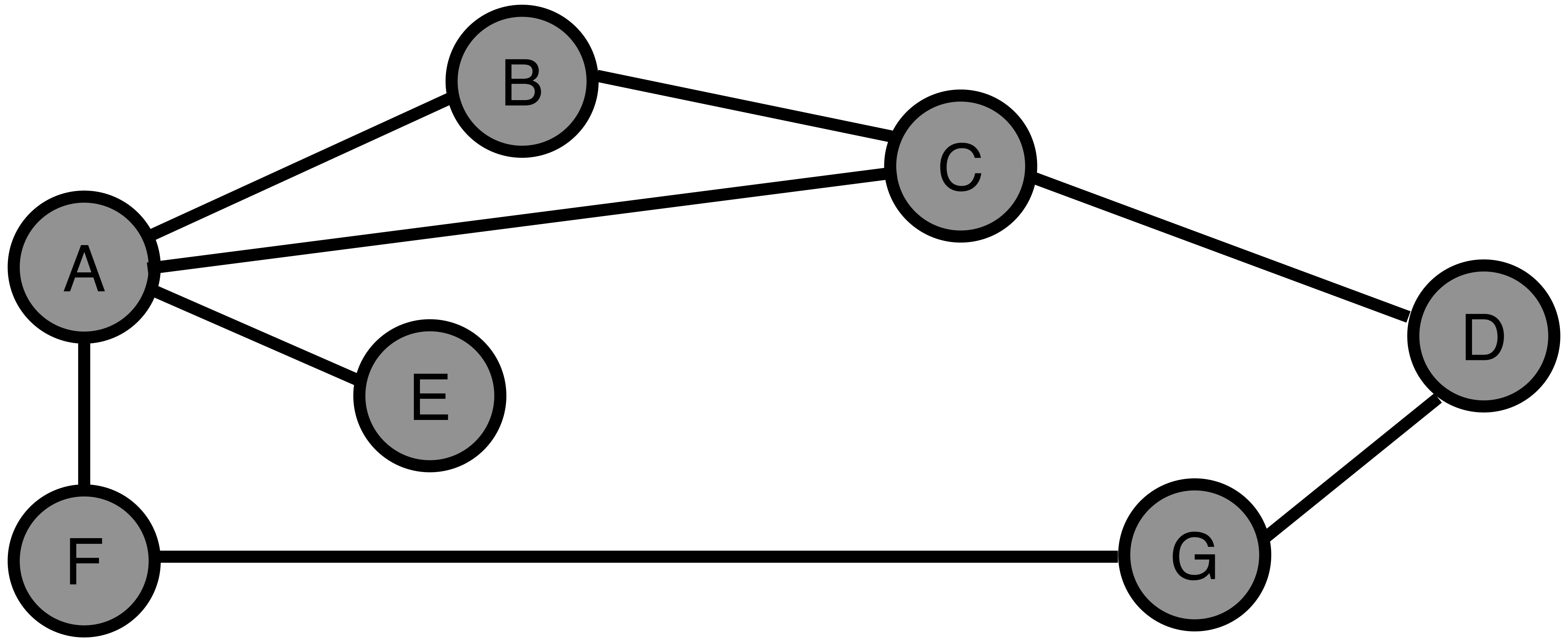
	Distance to Reach Node (Global View)						
	A	**B**	**C**	**D**	**E**	**F**	**G**
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C							
D							
E							
F							
G							

Step 1: figure out initial distance



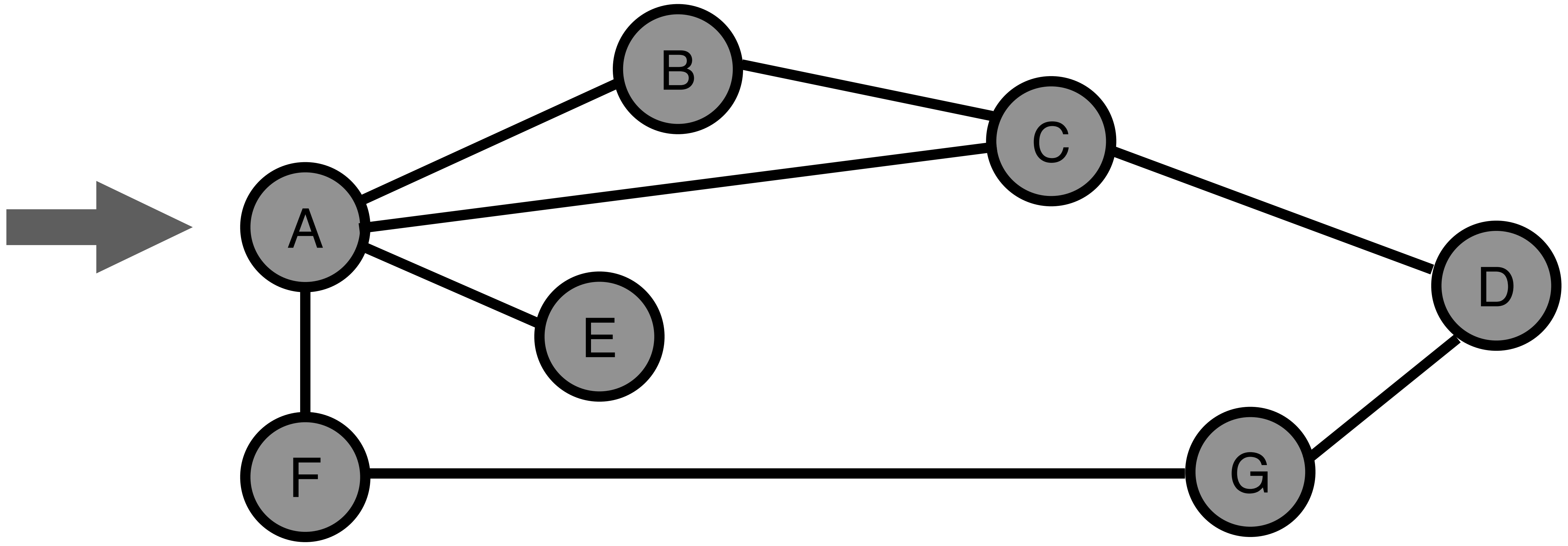
	Distance to Reach Node (Global View)						
	A	**B**	**C**	**D**	**E**	**F**	**G**
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D							
E							
F							
G							

Step 1: figure out initial distance



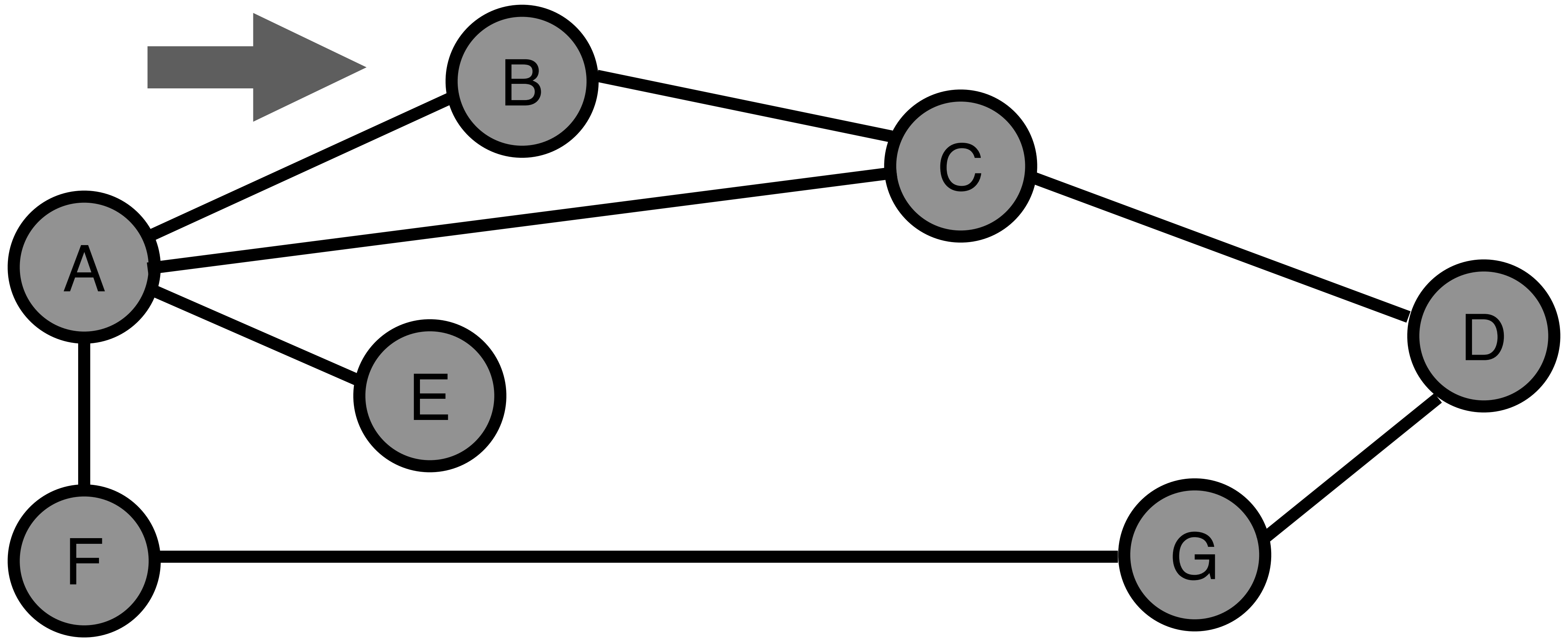
	Distance to Reach Node (Global View)						
	A	**B**	**C**	**D**	**E**	**F**	**G**
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	∞	∞	1	0

Initial Routing Table



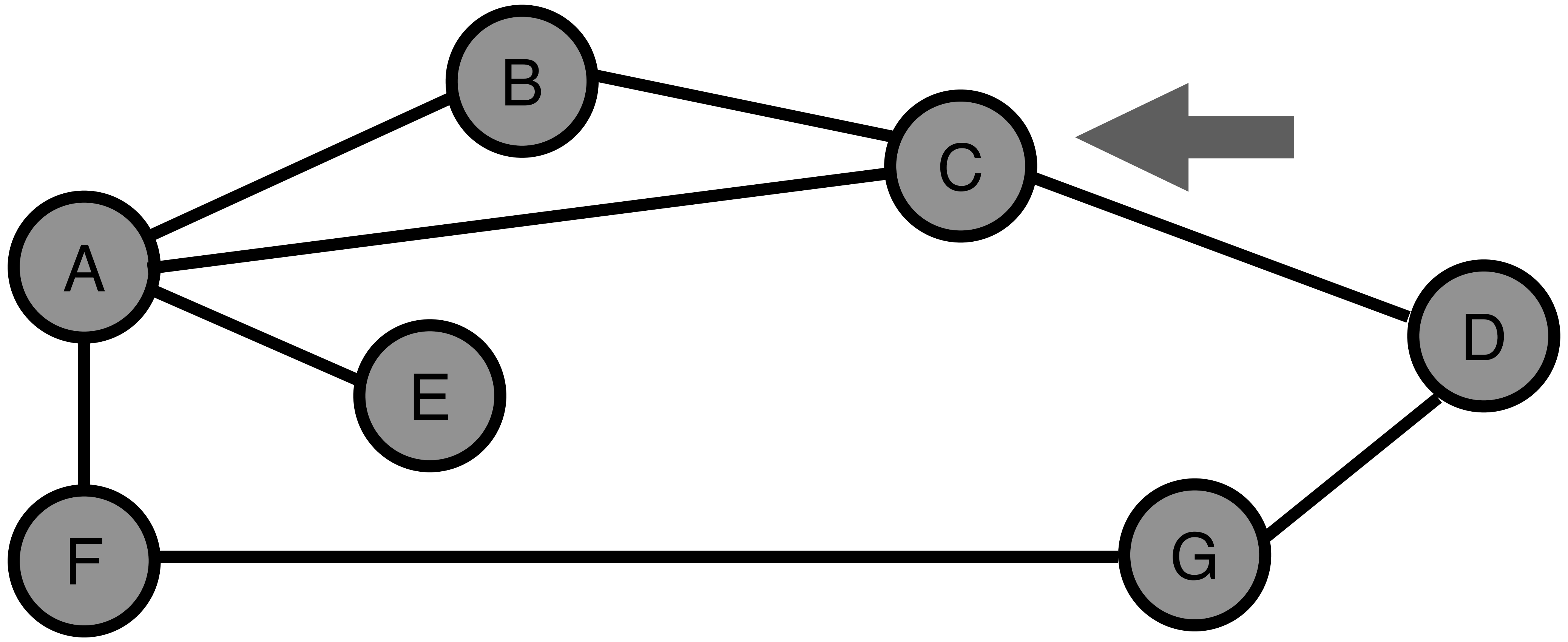
Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

Initial Routing Table



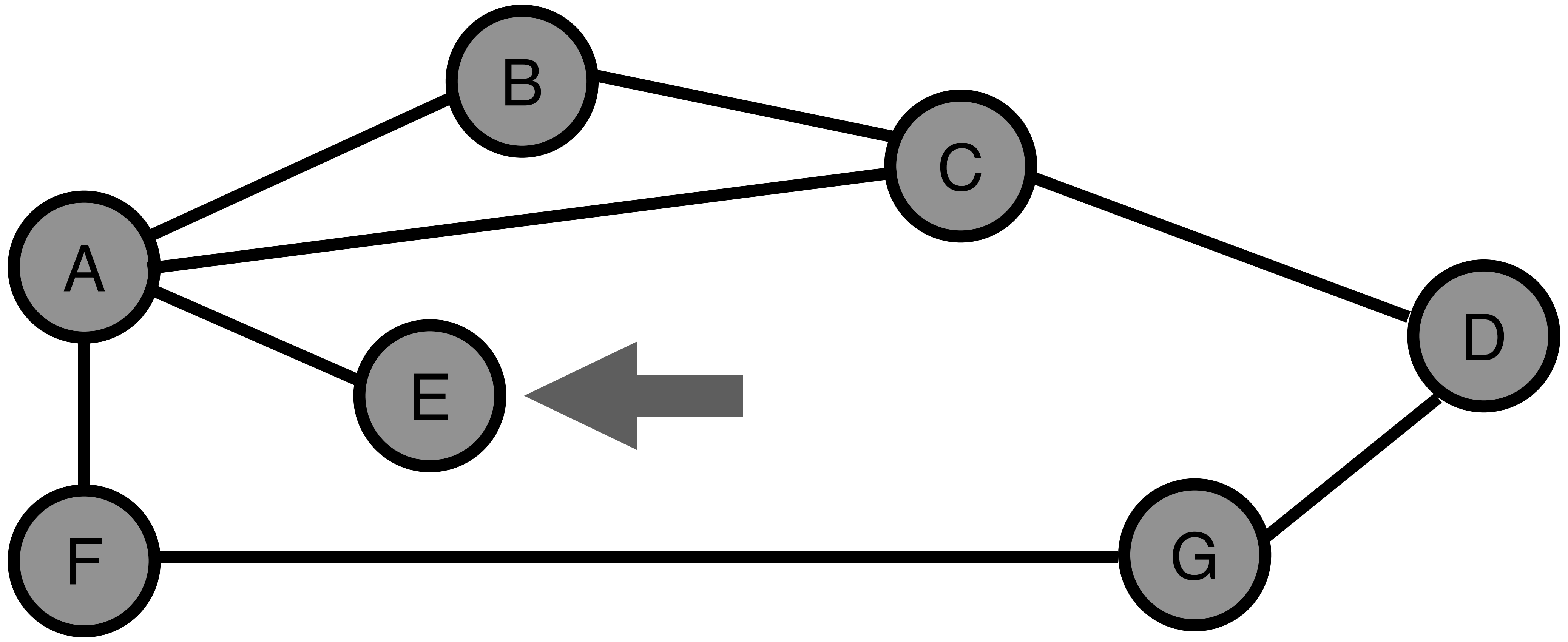
Destination	Cost	NextHop
A	1	A
C	1	C
D	∞	–
E	∞	–
F	∞	–
G	∞	–

Initial Routing Table



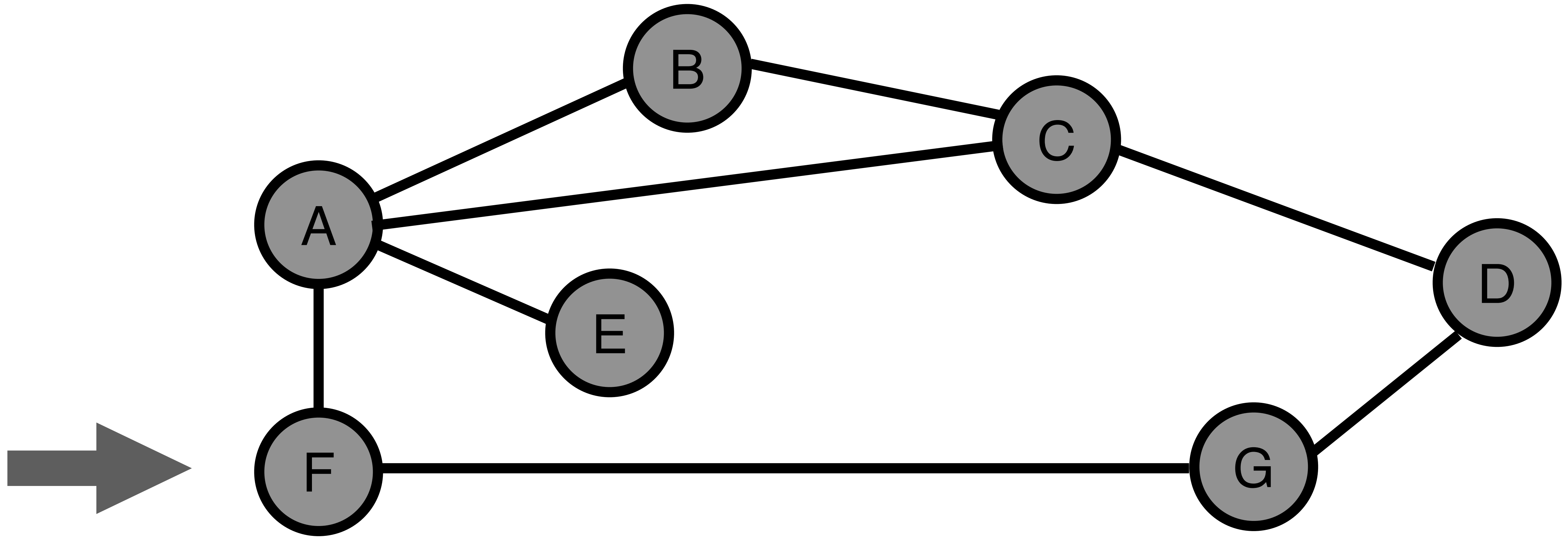
Destination	Cost	NextHop
A	1	A
B	1	B
D	1	D
E	∞	–
F	∞	–
G	∞	–

Initial Routing Table



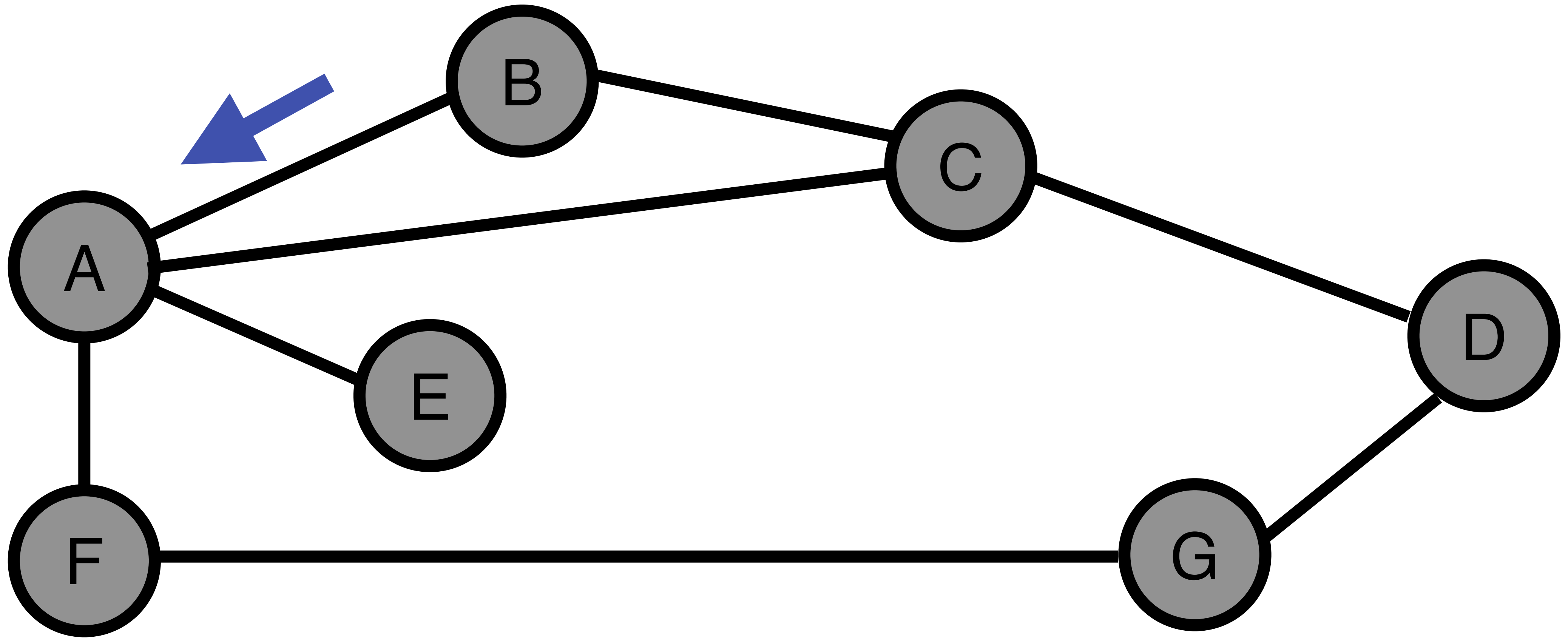
Destination	Cost	NextHop
A	1	A
B	∞	–
C	∞	–
D	∞	–
F	∞	–
G	∞	–

Initial Routing Table



Destination	Cost	NextHop
A	1	A
B	∞	–
C	∞	–
D	∞	–
E	∞	–
G	1	G

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

+

B

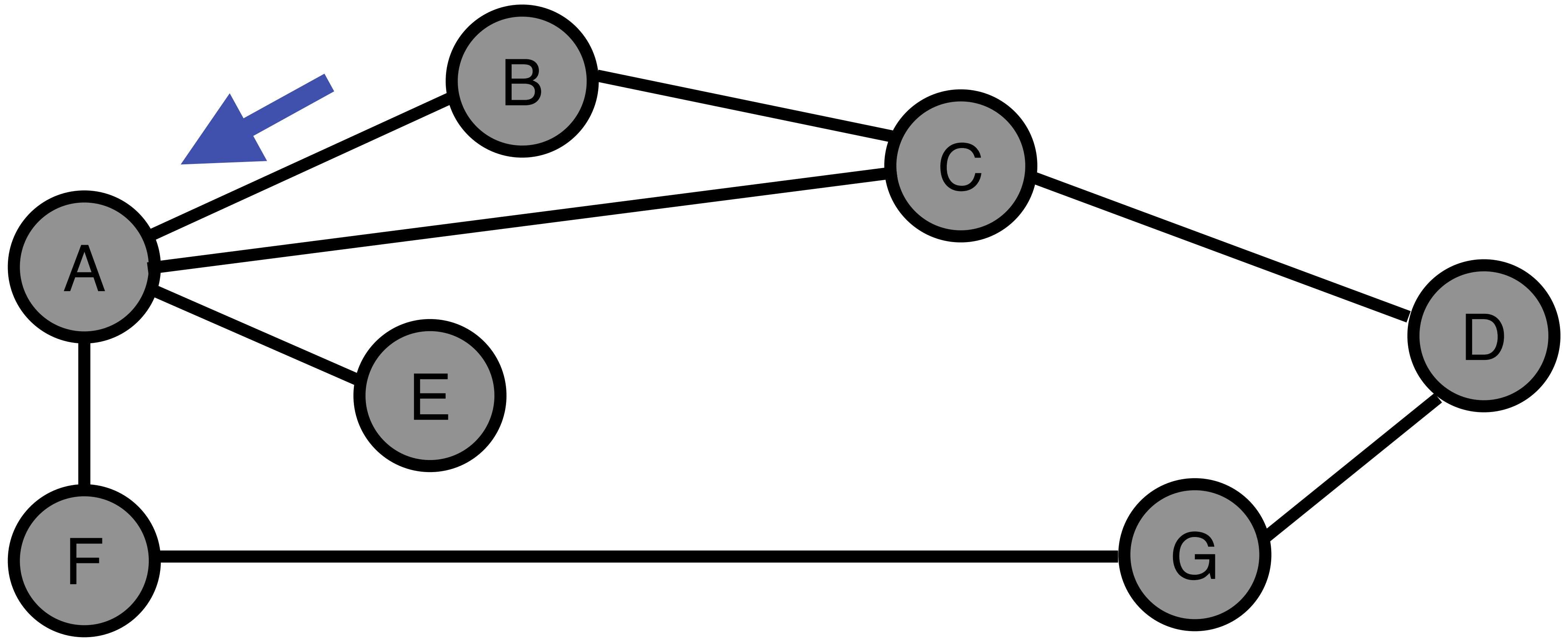
Dest.	Cost	NextHop
A	1	A
C	1	C
D	∞	-
E	∞	-
F	∞	-
G	∞	-

=

A

Dest.	Cost	NextHop
B		
C		
D		
E		
F		
G		

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

+

B

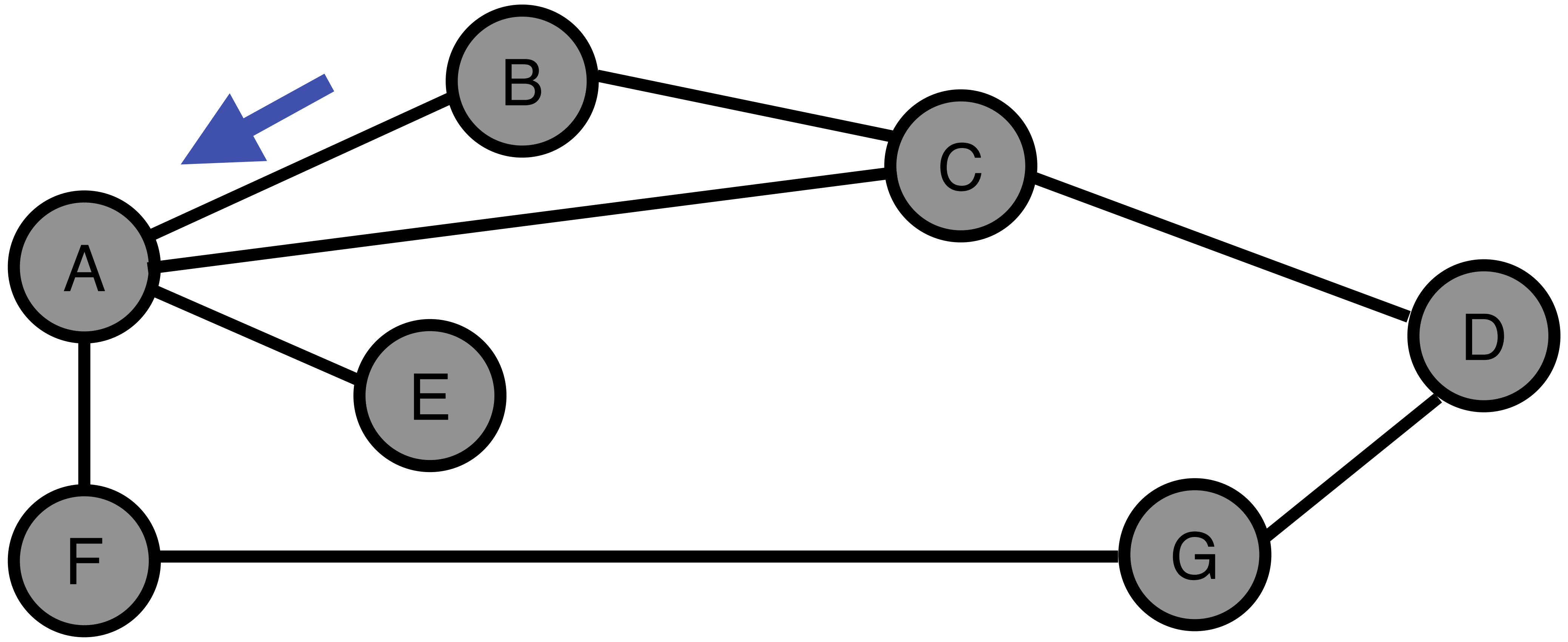
Dest.	Cost	NextHop
A	1	A
C	1	C
D	∞	-
E	∞	-
F	∞	-
G	∞	-

=

A

Dest.	Cost	NextHop
B	1	B
C		
D		
E		
F		
G		

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

+

B

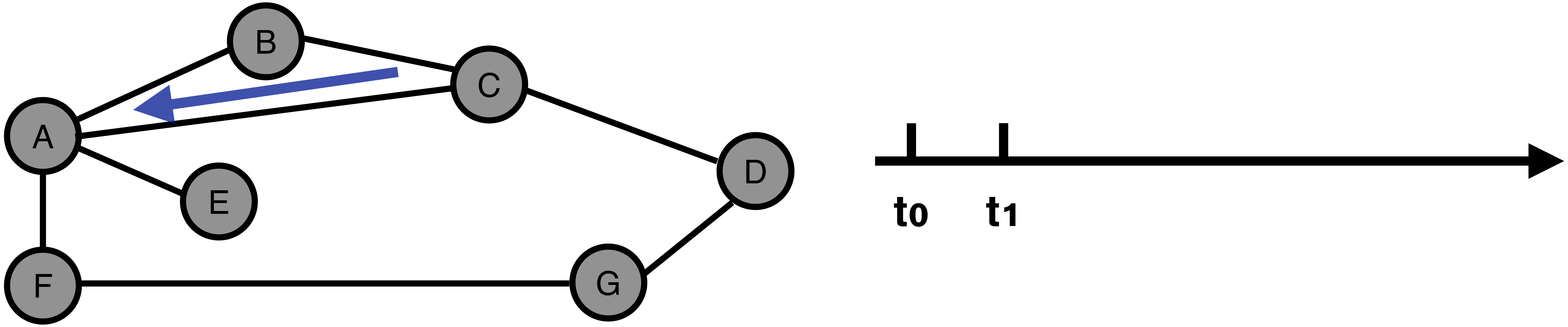
Dest.	Cost	NextHop
A	1	A
C	1	C
D	∞	-
E	∞	-
F	∞	-
G	∞	-

=

A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

+

C

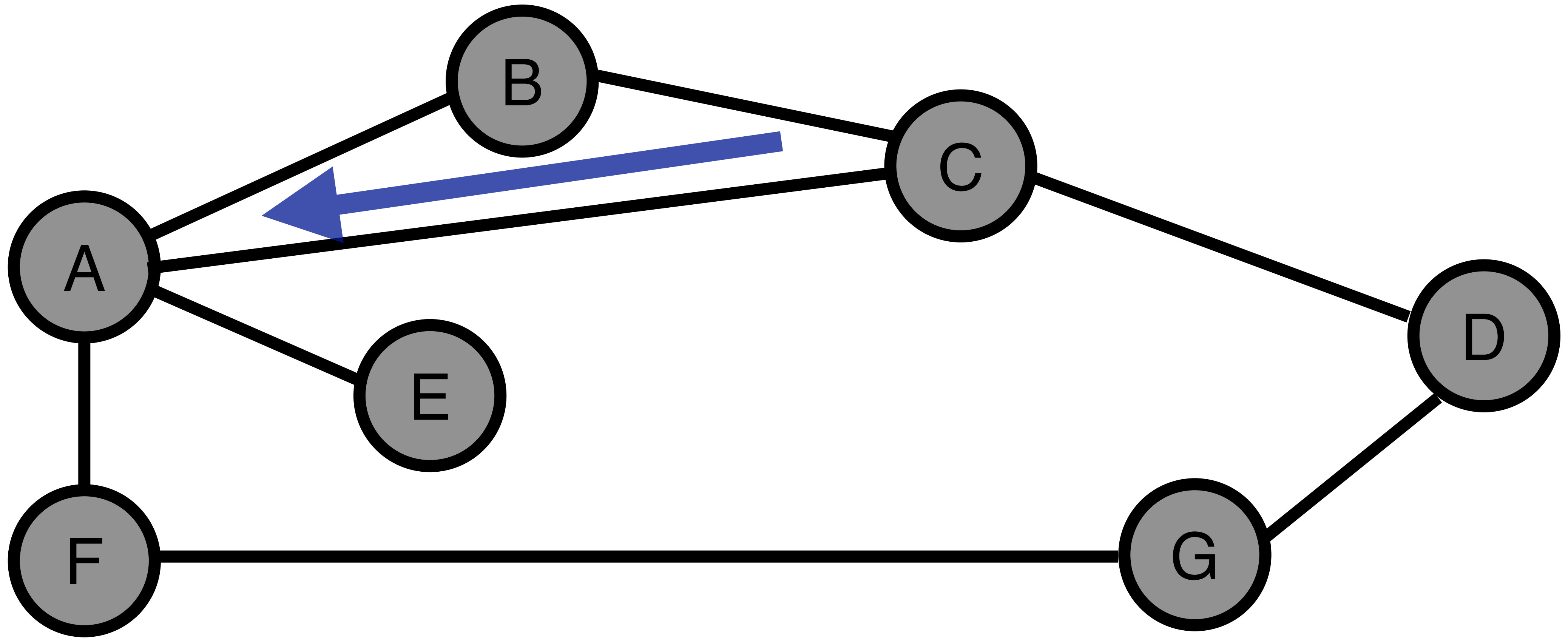
Dest.	Cost	NextHop
A	1	A
B	1	B
D	1	D
E	∞	-
F	∞	-
G	∞	-

=

A

Dest.	Cost	NextHop
B		
C		
D		
E		
F		
G		

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	∞	–
E	1	E
F	1	F
G	∞	–

\+

C

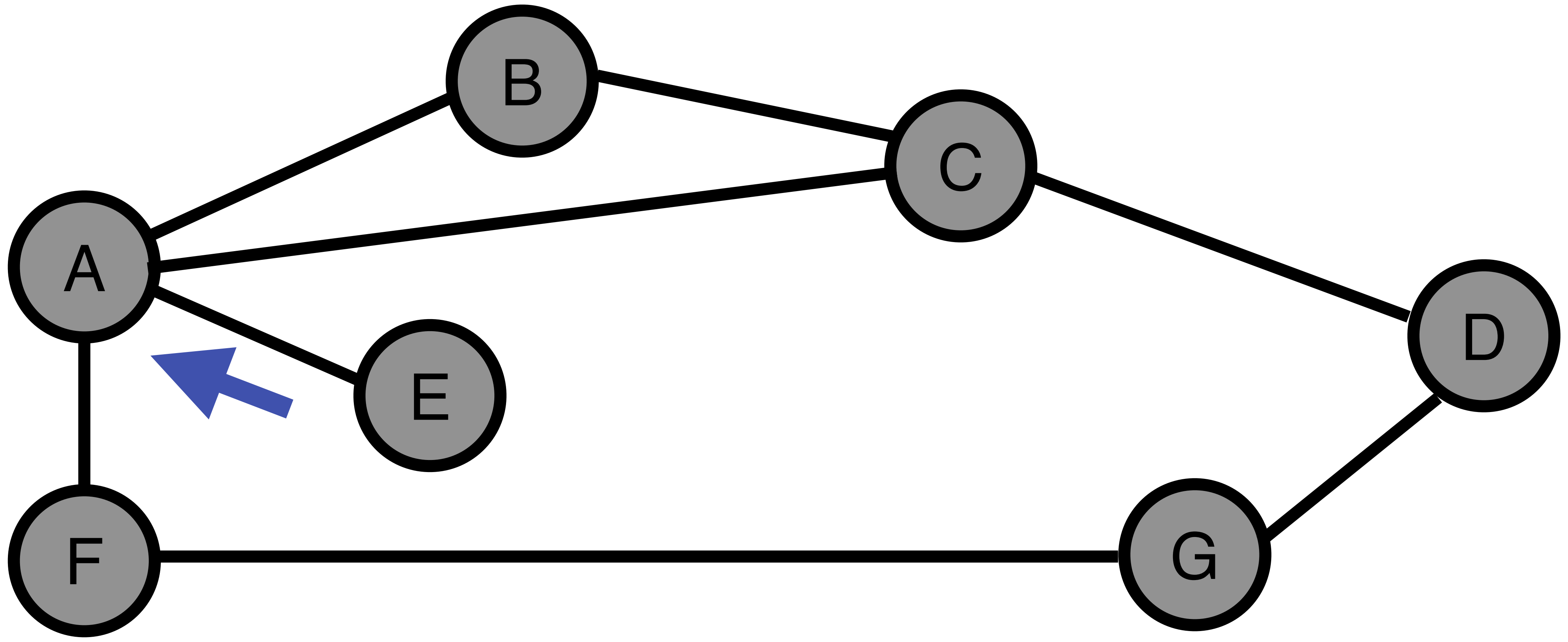
Dest.	Cost	NextHop
A	1	A
B	1	B
D	1	D
E	∞	–
F	∞	–
G	∞	–

=

A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	–

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	–

+

E

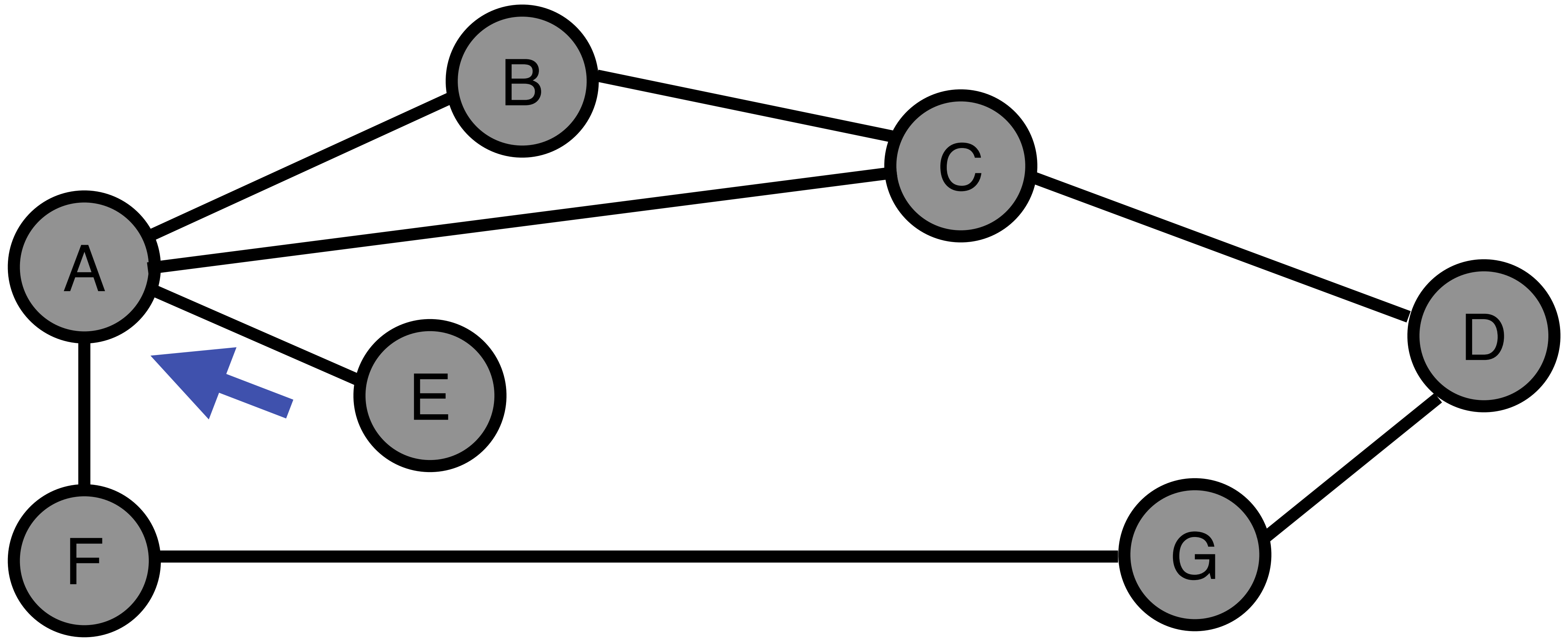
Dest.	Cost	NextHop
A	1	A
B	∞	–
C	∞	–
D	∞	–
F	∞	–
G	∞	–

=

A

Dest.	Cost	NextHop
B		
C		
D		
E		
F		
G		

Step 2: exchange the distance vector



A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	-

+

E

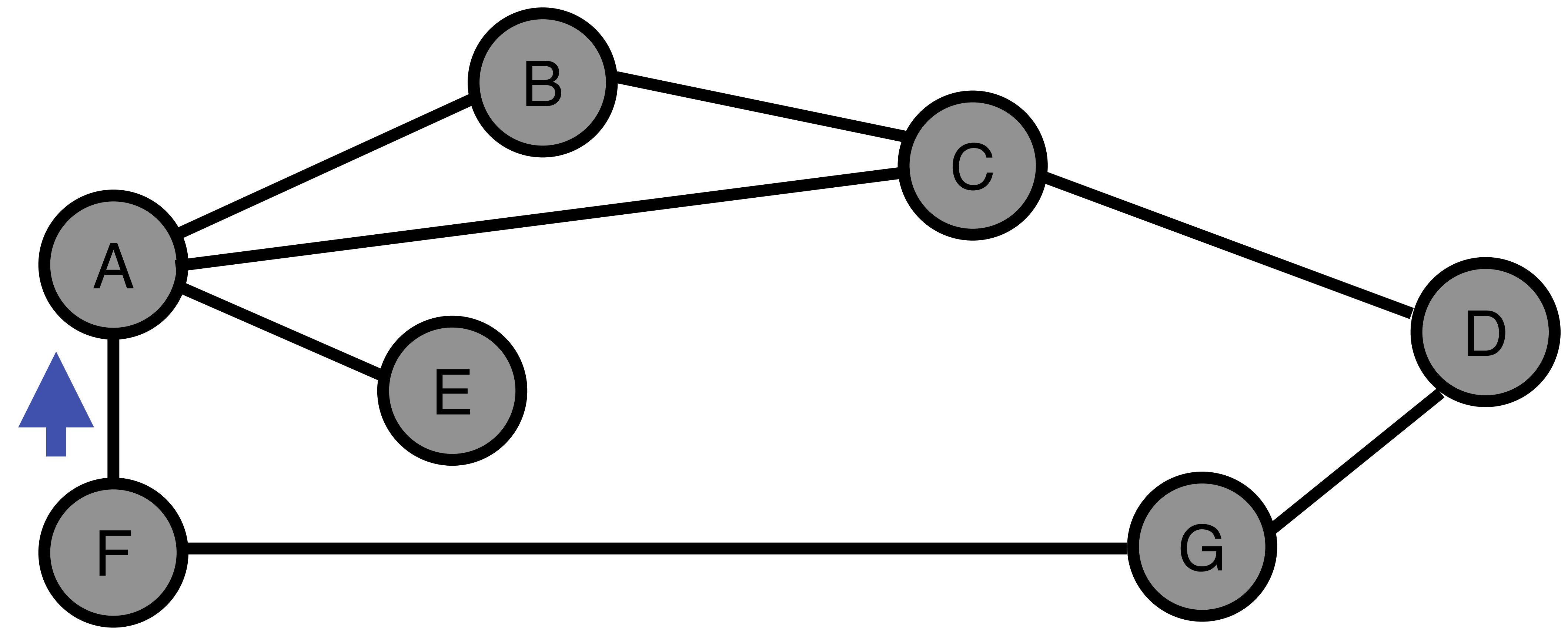
Dest.	Cost	NextHop
A	1	A
B	∞	-
C	∞	-
D	∞	-
F	∞	-
G	∞	-

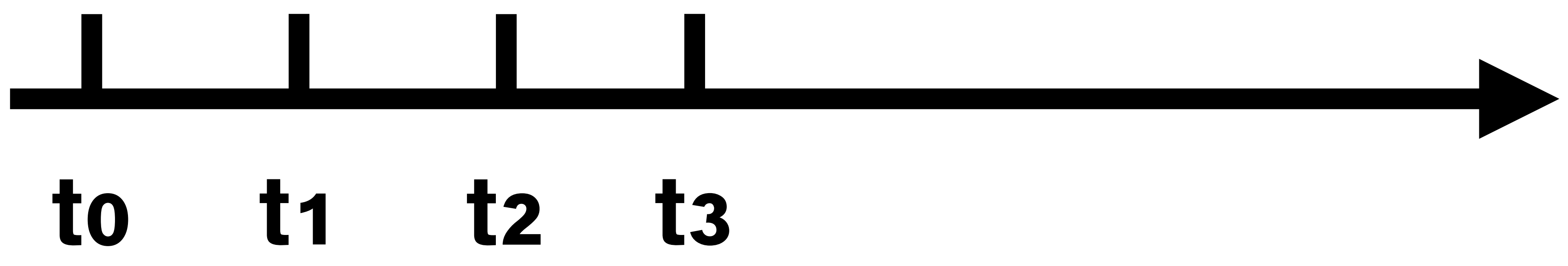
=

A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	-

Step 2: exchange the distance vector





A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	-

+

F

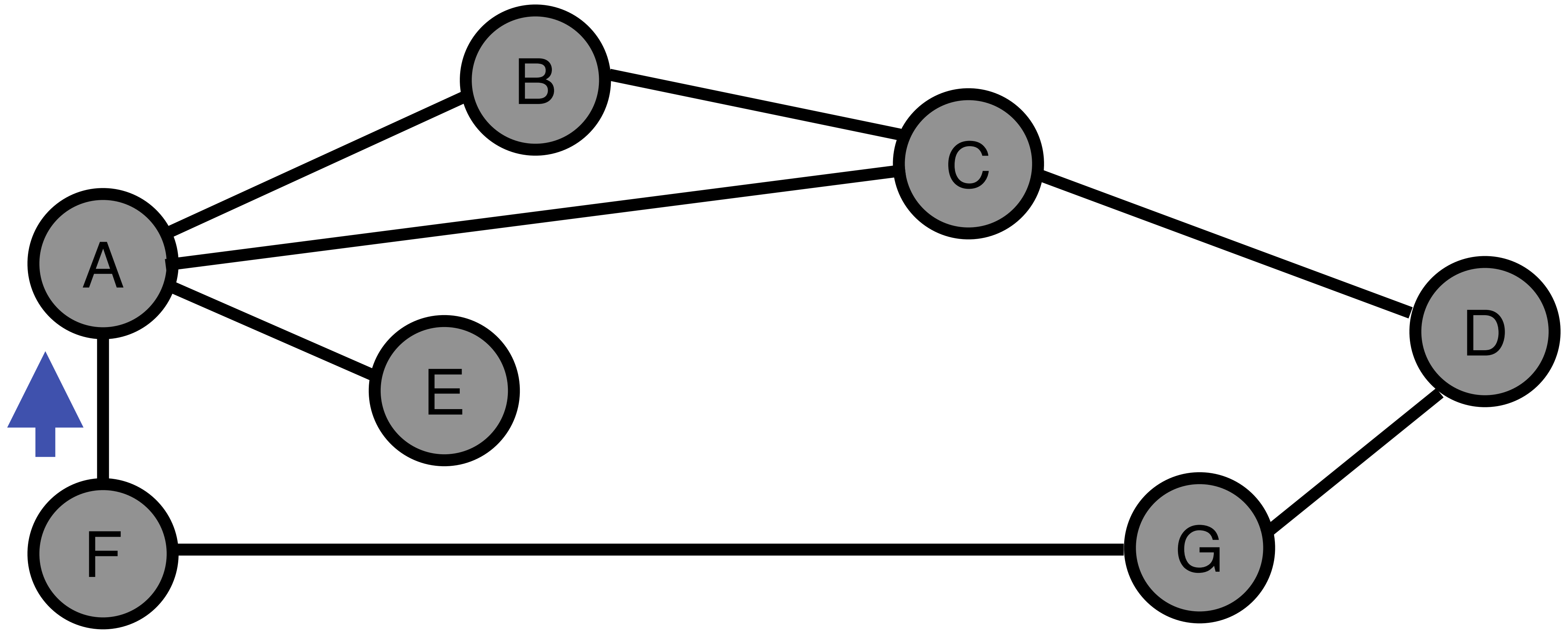
Dest.	Cost	NextHop
A	1	A
B	∞	-
C	∞	-
D	∞	-
F	0	F
G	1	G

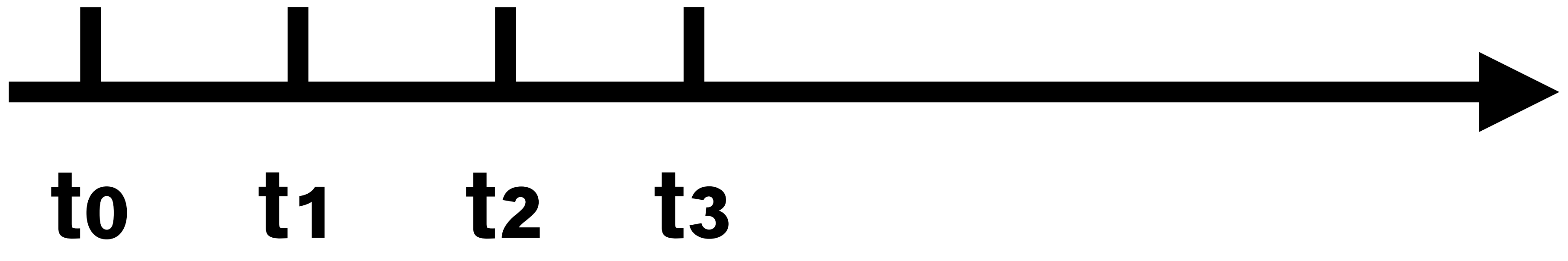
=

A

Dest.	Cost	NextHop
B		
C		
D		
E		
F		
G		

Step 2: exchange the distance vector





A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	–

+

F

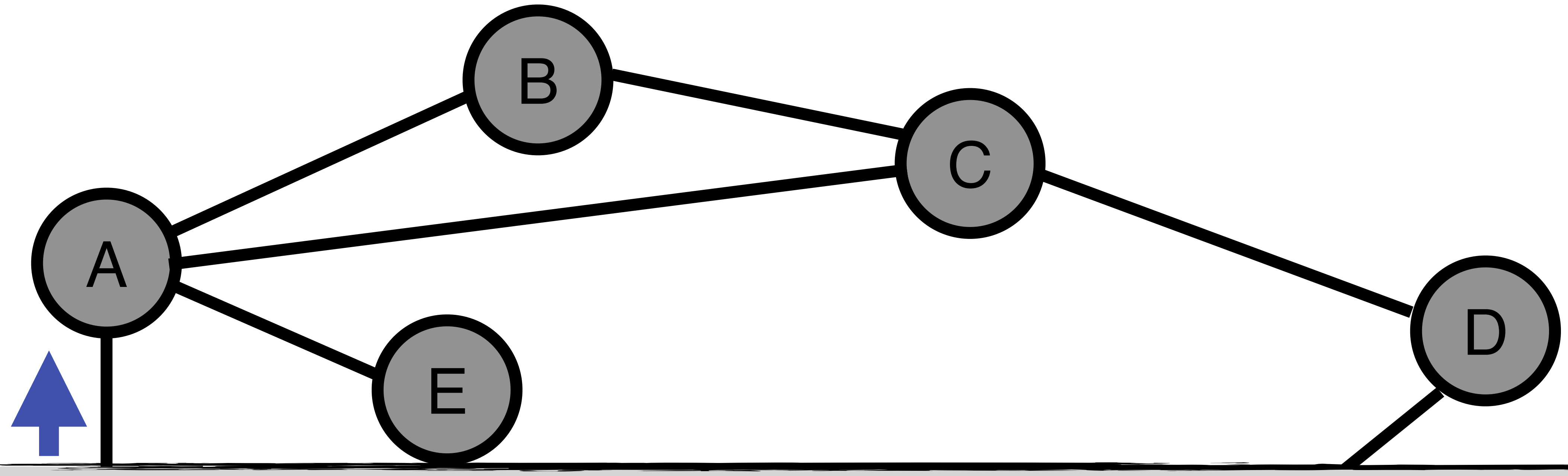
Dest.	Cost	NextHop
A	1	A
B	∞	–
C	∞	–
D	∞	–
F	0	F
G	1	G

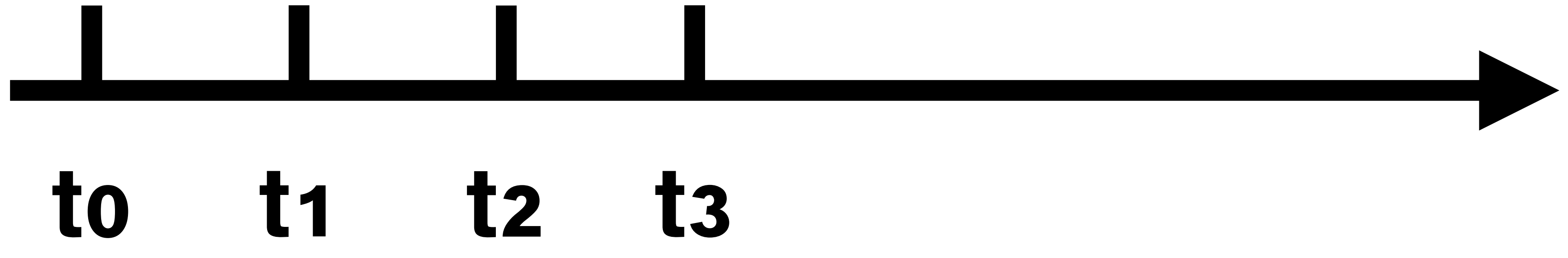
=

A

Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Step 2: exchange the distance vector





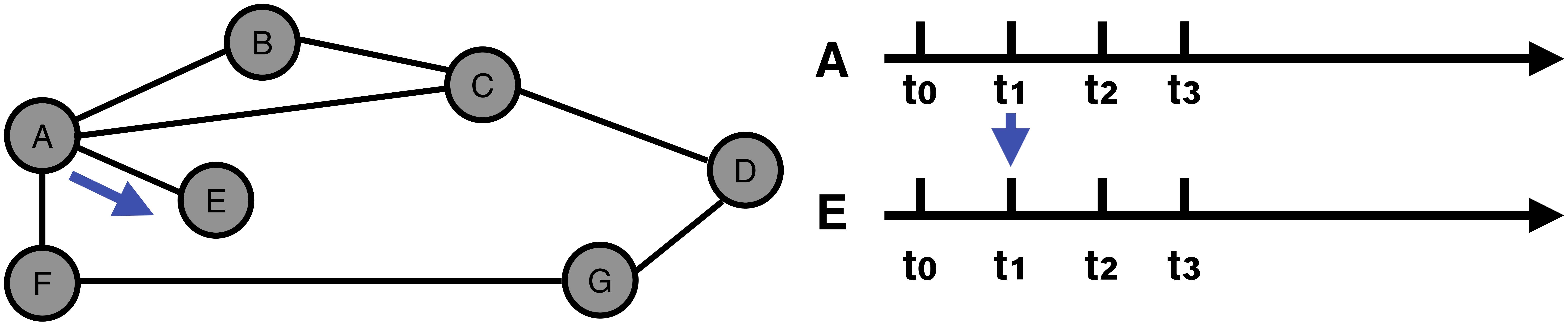
New_Cost (node) = Cost(node) (from neighbor) + Cost (node-neighbor)

- Cost (node-neighbor) = 1 in the above discussion
- Cost = Min (New_Cost, Old_Cost)
- If New_Cost is chosen, update the next hop to the neighbor node

The routing table is evolving

- Based on the event sequence

Step 2: exchange the distance vector



E

Dest.	Cost	NextHop
A	1	A
B	∞	–
C	∞	–
D	∞	–
F	∞	–
G	∞	–

\+

A (t_1)

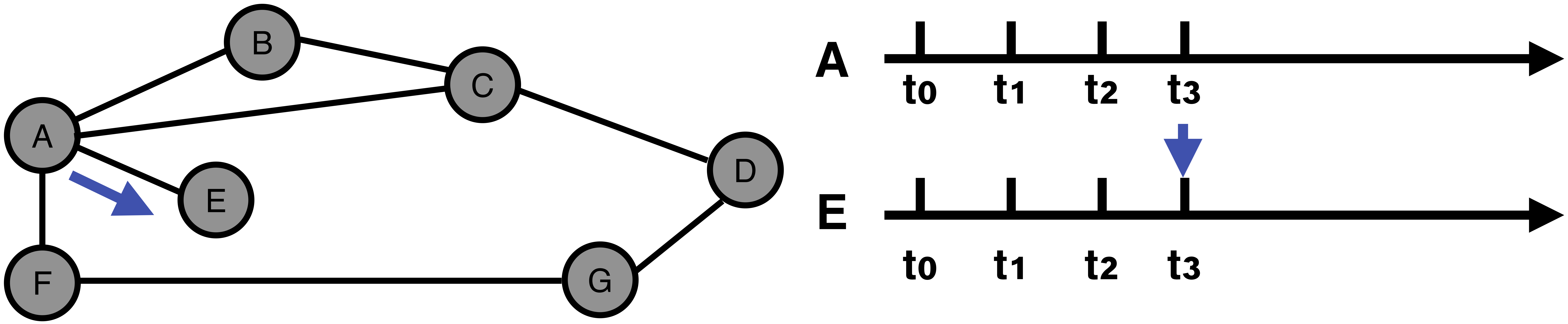
Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	∞	–

=

E

Dest.	Cost	NextHop
A	1	A
B	2	A
C	2	A
D	3	A
F	2	A
G	∞	–

Step 2: exchange the distance vector



E

Dest.	Cost	NextHop
A	1	A
B	∞	–
C	∞	–
D	∞	–
F	∞	–
G	∞	–

+

A (t3)

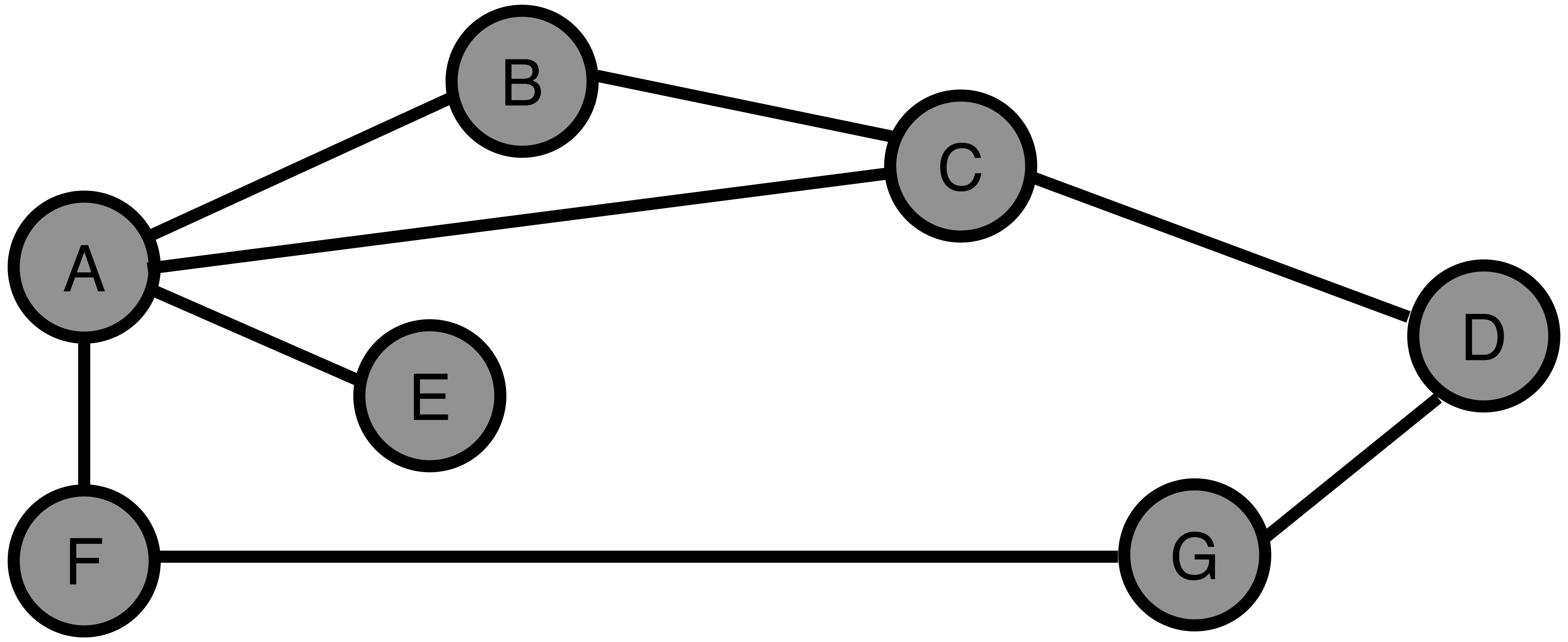
Dest.	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	E

=

E

Dest.	Cost	NextHop
A	1	A
B	2	A
C	2	A
D	3	A
F	2	A
G	2	A

Distance Table (stable)



	Distance to Reach Node (Global View)						
	A	**B**	**C**	**D**	**E**	**F**	**G**
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Distance Vector Discussion

#1: The distance vector routing is based on the Bellman-Ford algorithm

#2: Every T seconds each router sends a list of distance to all the routers to its neighbor

#3: Each router then updates its table based on the new information

Distance Vector Discussion

#1: The distance vector routing is based on the

Advantage

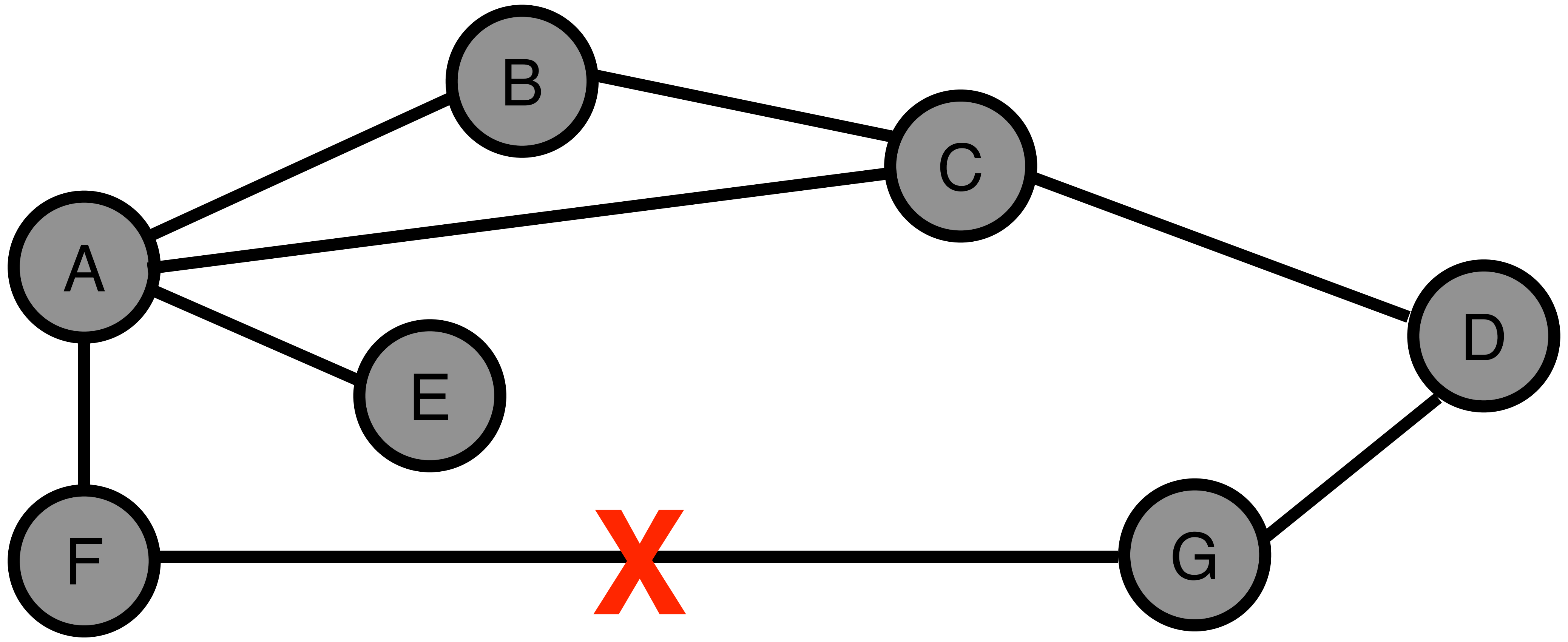
- Fast response to the good news

Disadvantage

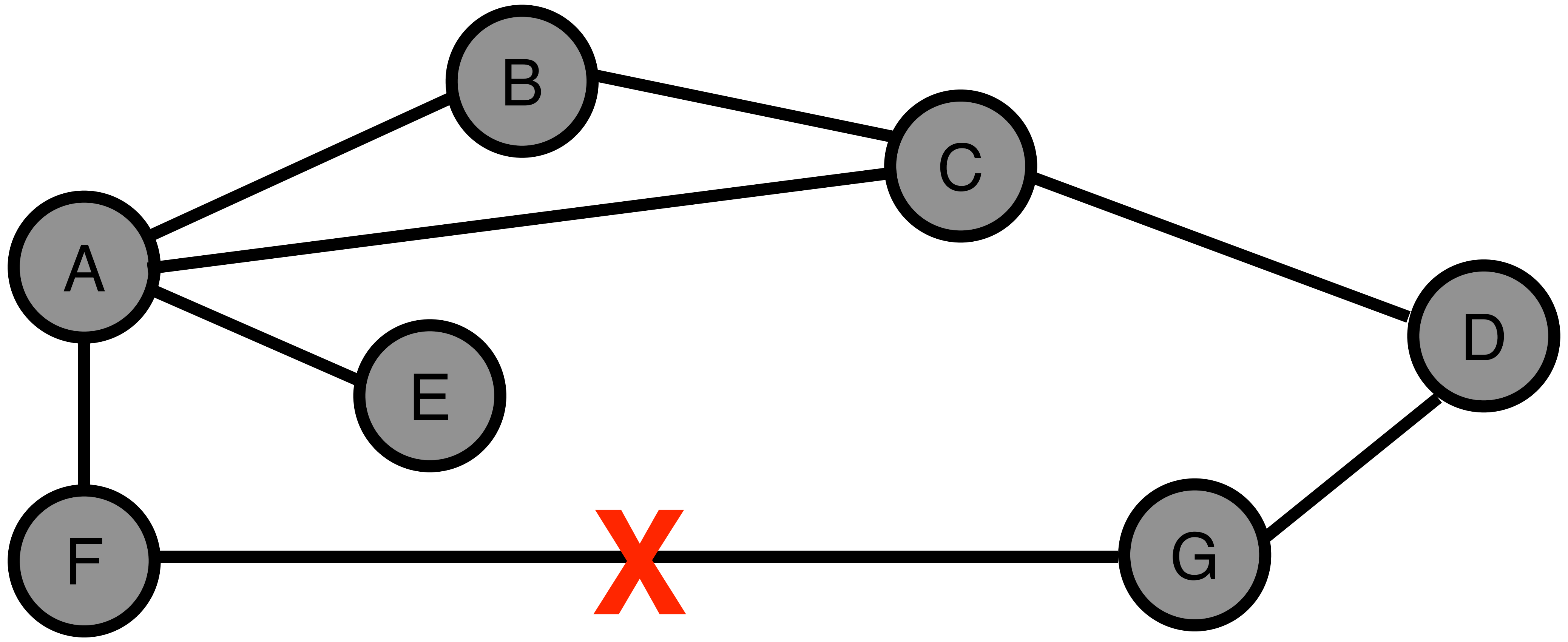
- Slow response to the bad news

new information

Distance Vector under Link Failure

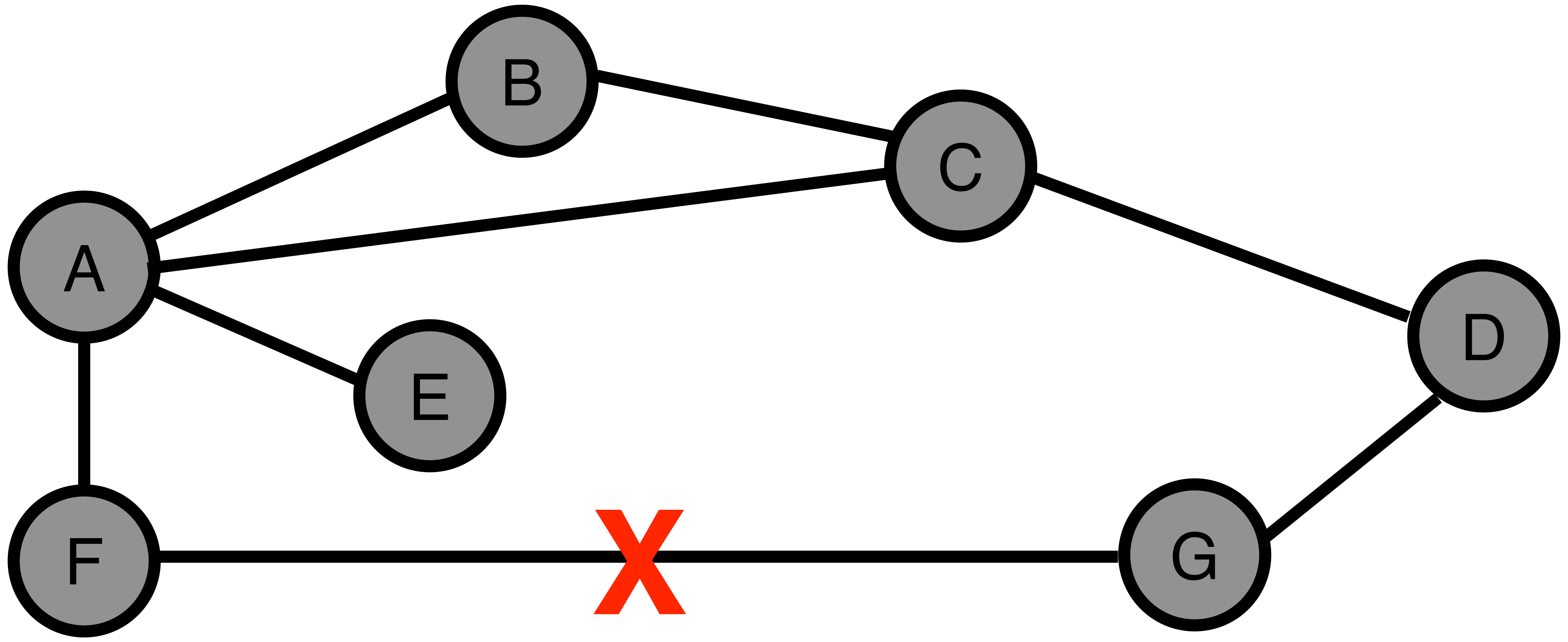


Distance Vector under Link Failure



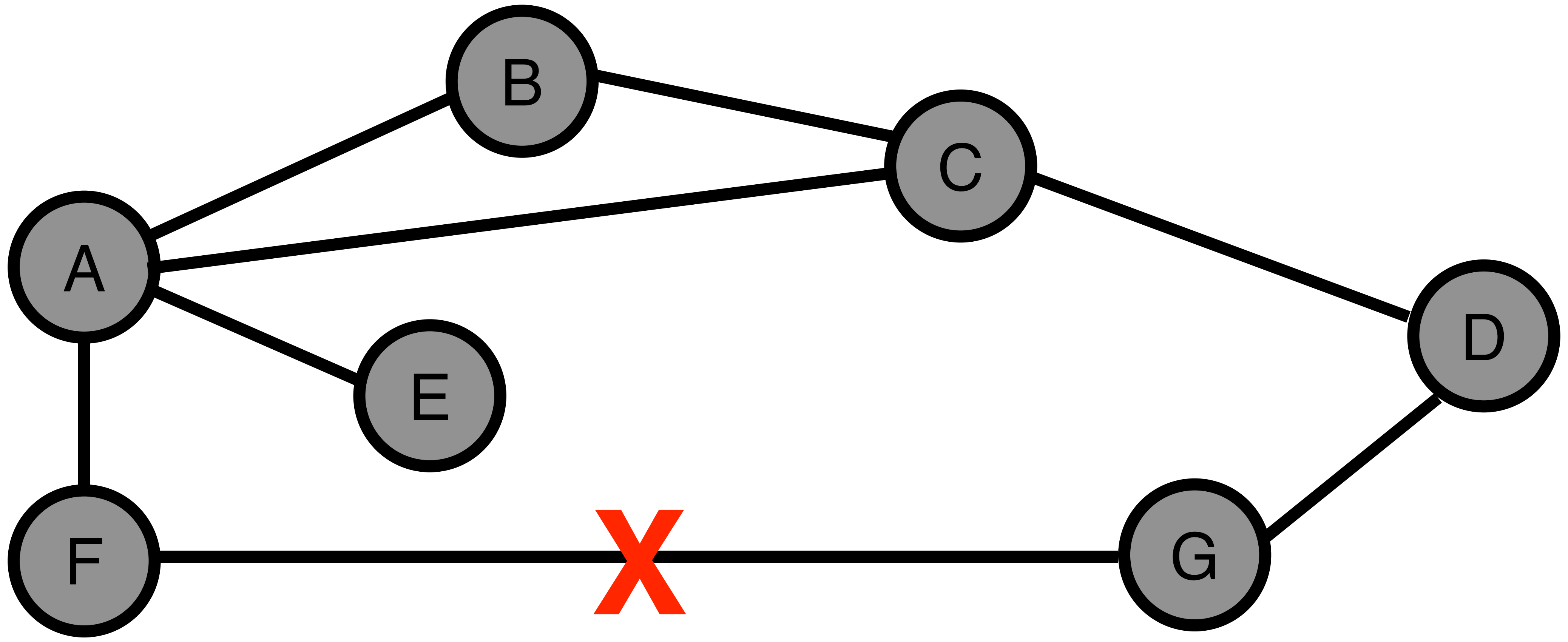
- F detects that the link to G has failed
- F sets the distance to G as infinity and sends updates to A
- A sets the distance to G to infinity since it uses F to reach G

Distance Vector under Link Failure



- F detects that the link to G has failed
- F sets the distance to G as infinity and sends updates to A
- A sets the distance to G to infinity since it uses F to reach G
- **A receives a periodic update from C with a 2-hop path to G**
- **A sets the distance to G to 3 and sends an update to F**
- **F decides it can reach G in 4 hops via A**

Distance Vector under Link Failure



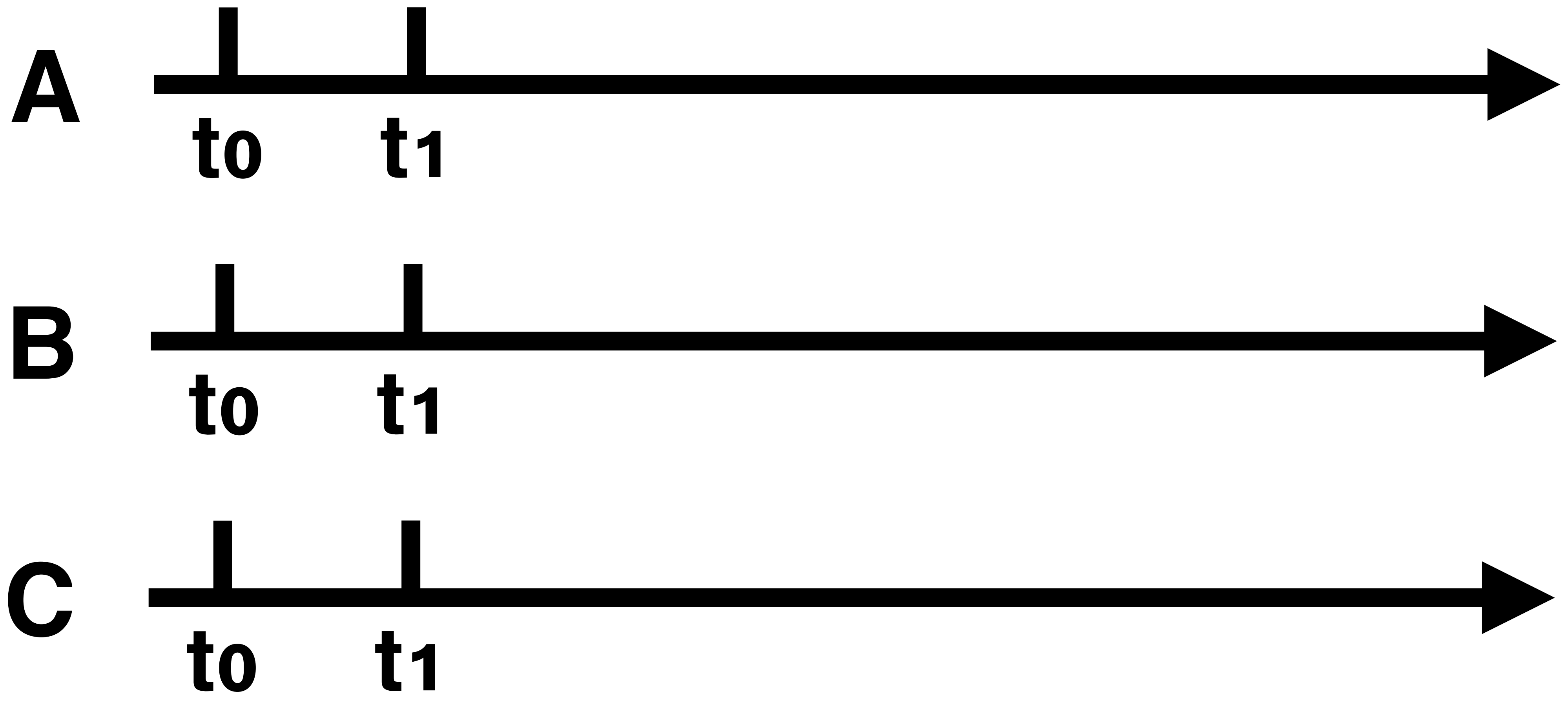
- F detects that the link to G has failed
- F sets the distance to G as infinity and sends updates to A
- A sets the distance to G to infinity since it uses F to reach G
- **A receives a periodic update from C with a 2-hop path to G**
- **A sets the distance to G to 3 and sends an update to F**
- **F decides it can reach G in 4 hops via A**

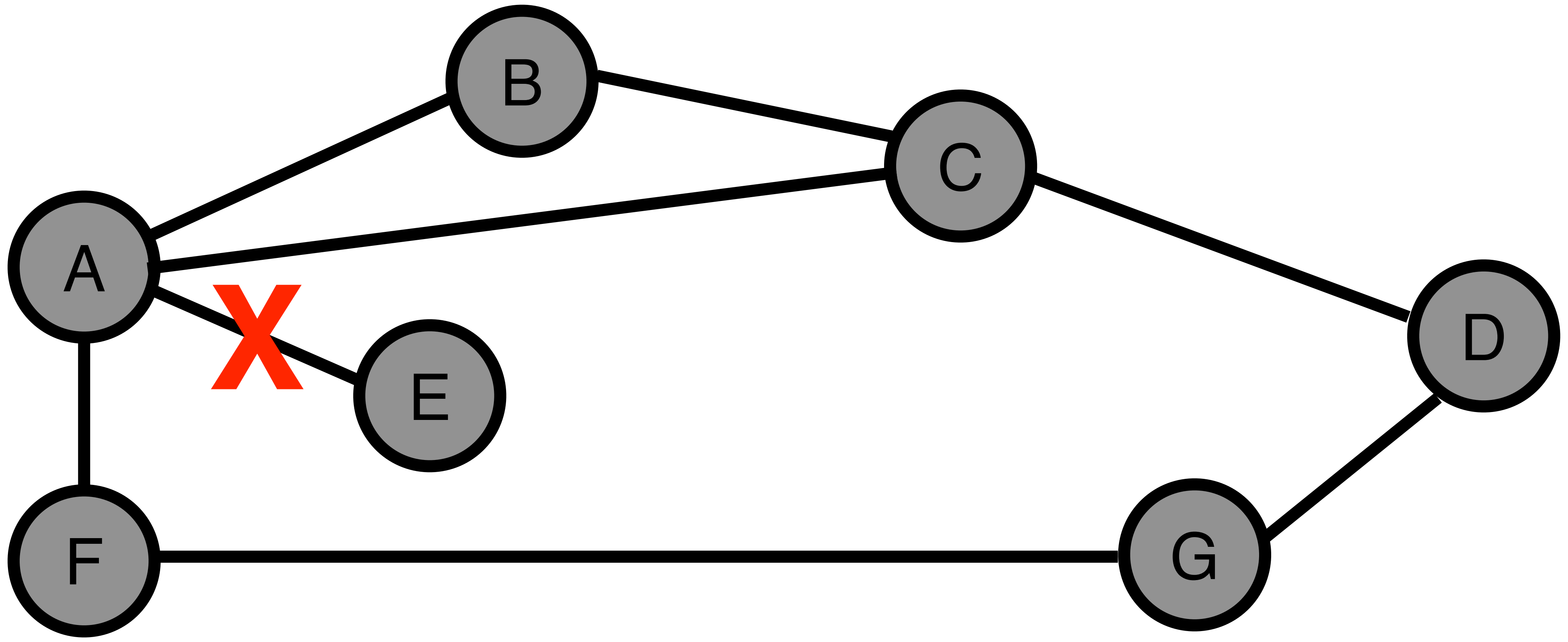
Distance Vector Converges Slowly

Converge: the process of getting consistent routing information to all the nodes

Slightly different circumstances can prevent the network from stabilizing

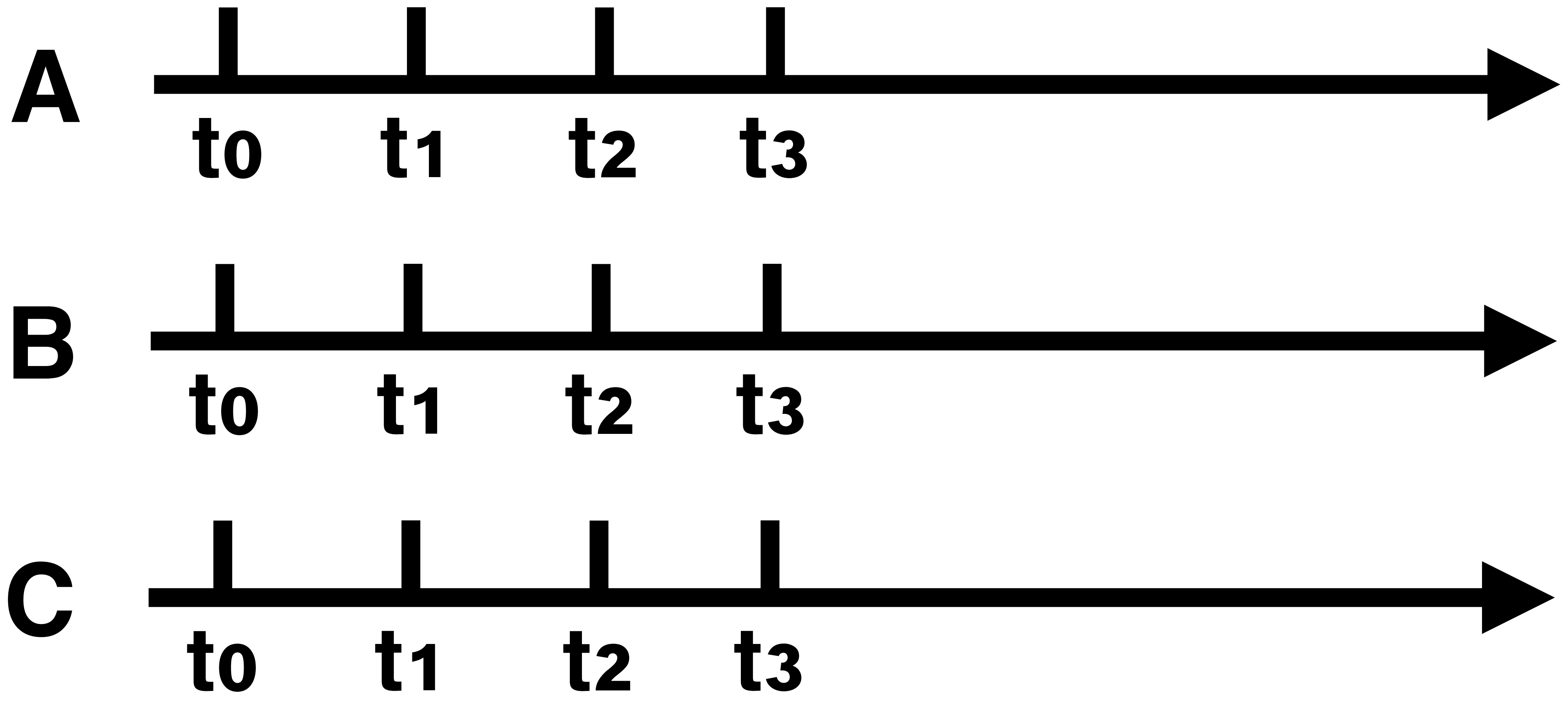
Distance Vector Converges Slowly

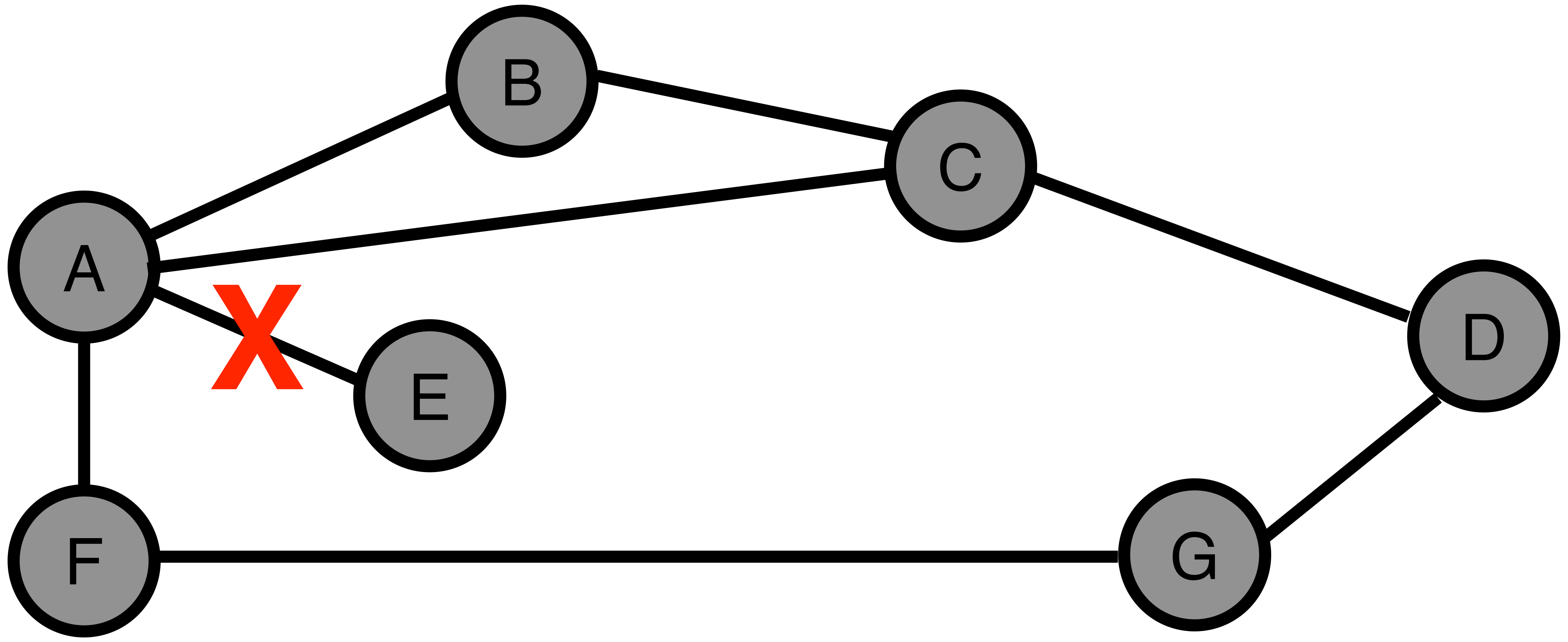




- At t_0, A detects the link failure and advertises a distance of infinity to E
- At t_1, B and C receive the message, and update the routing table accordingly

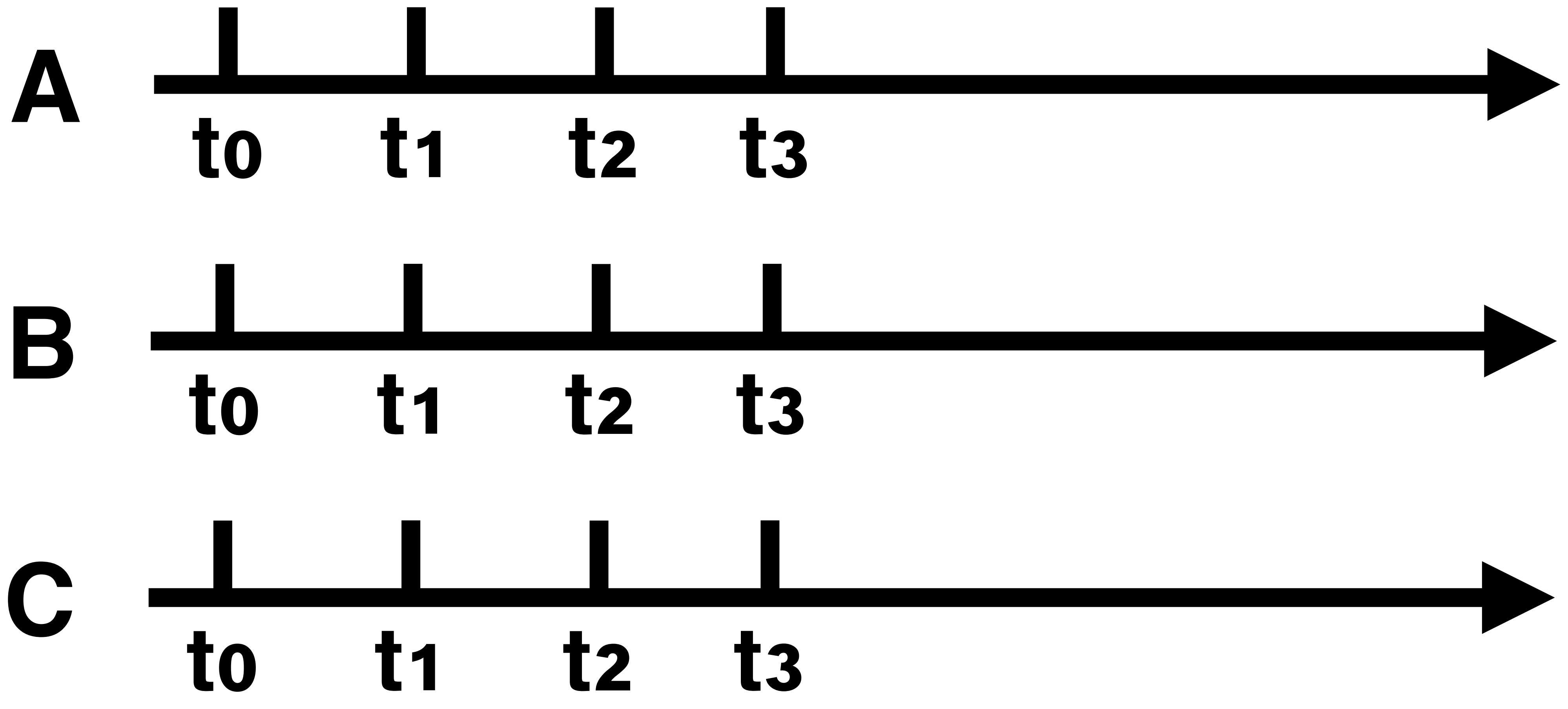
Distance Vector Converges Slowly

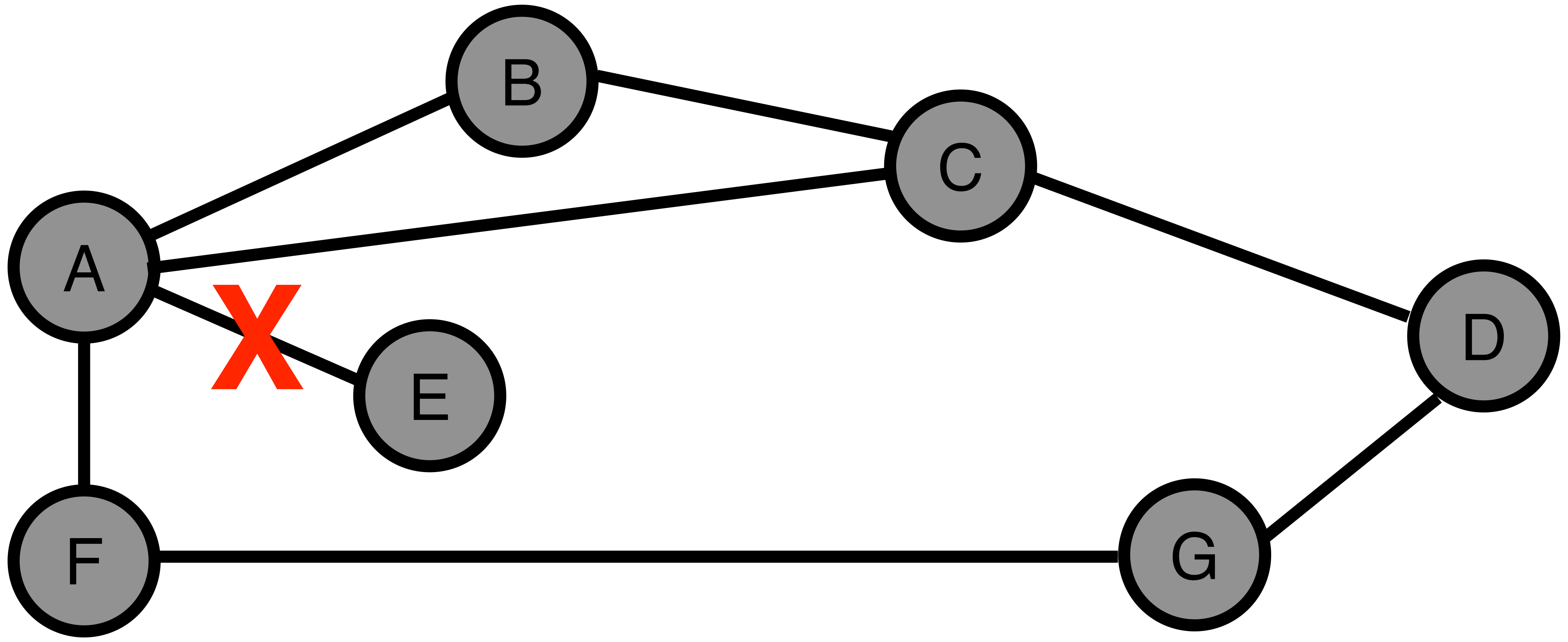




- At t_0, A detects the link failure and advertises **a distance of infinity to E**
- At t_1, B receives the message from A and updates the routing table as **<E, Infinity>**
- At t_2, B receives the message from C (saying the distance to E is 2), and updates the routing table as **<E, 3>**

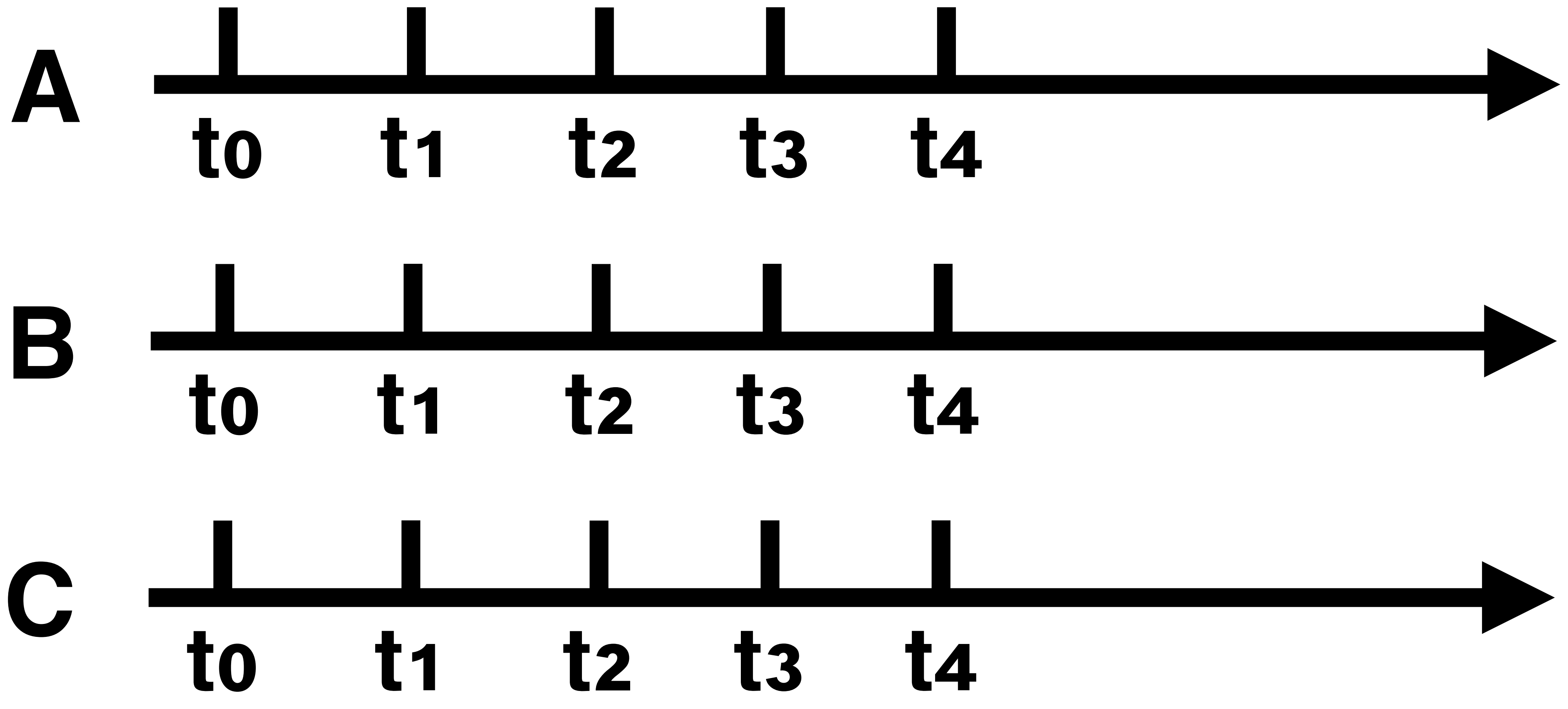
Distance Vector Converges Slowly

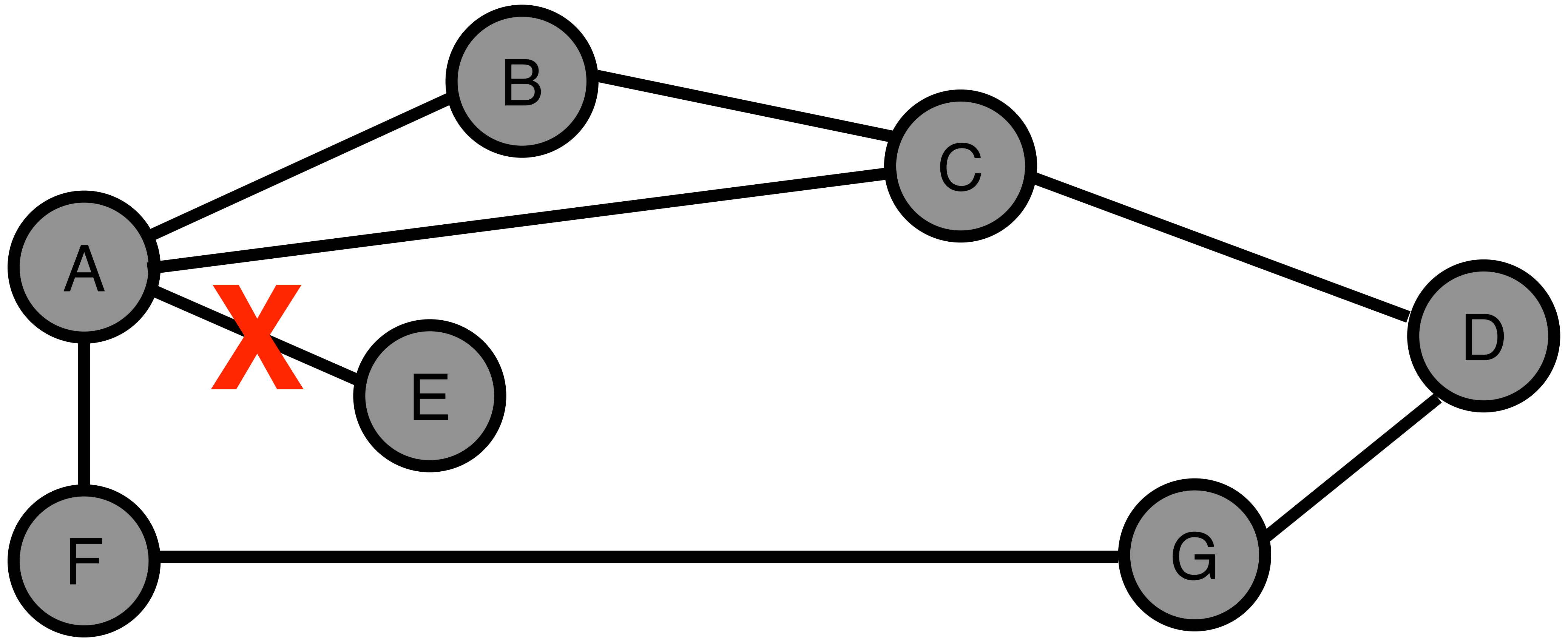




- At **t_0**, A detects the link failure and advertises **a distance of infinity to E**
- At **t_1**, B receives the message from A and updates the routing table as **<E, Infinity>**
- At **t_2**, B receives the message from C (saying the distance to E is 2), and updates the routing table as **<E, 3>**
- At **t_3**, C receives the message from A and updates the routing table as **<E, Infinity>**

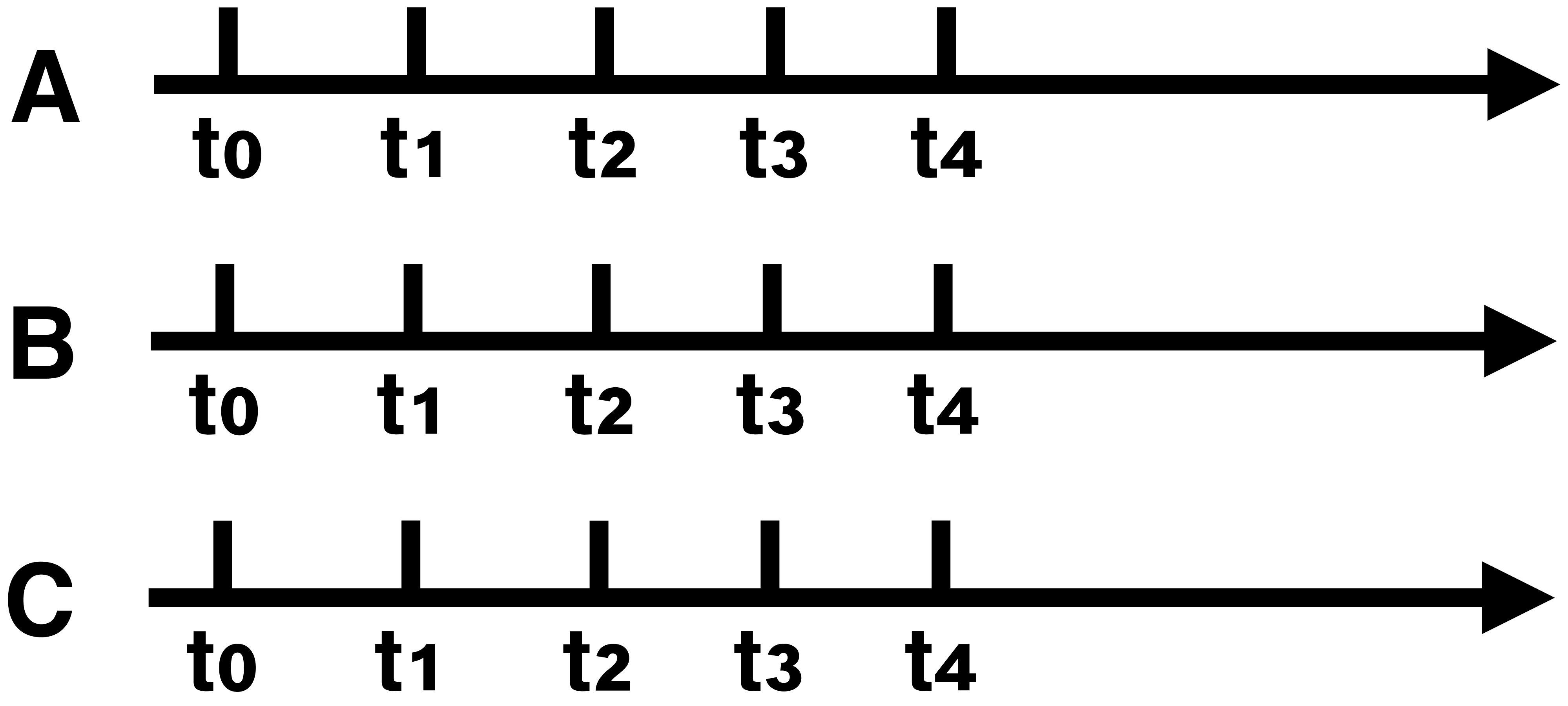
Distance Vector Converges Slowly

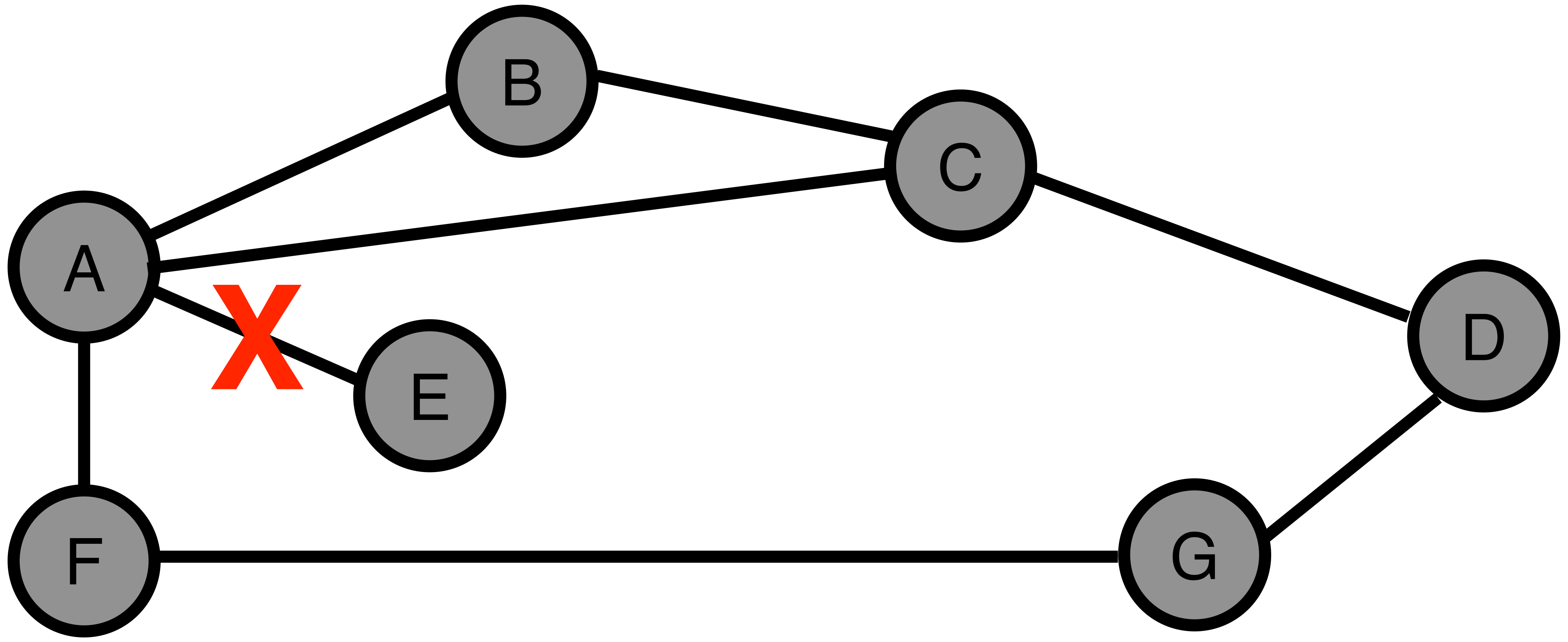




- At **t4**, C receives the message from B (saying the distance to E is 3), and updates the routing table as **<E, 4>**
- At **t4**, A receives the message from B (saying the distance to E is 3), and updates the routing table as **<E, 4>**

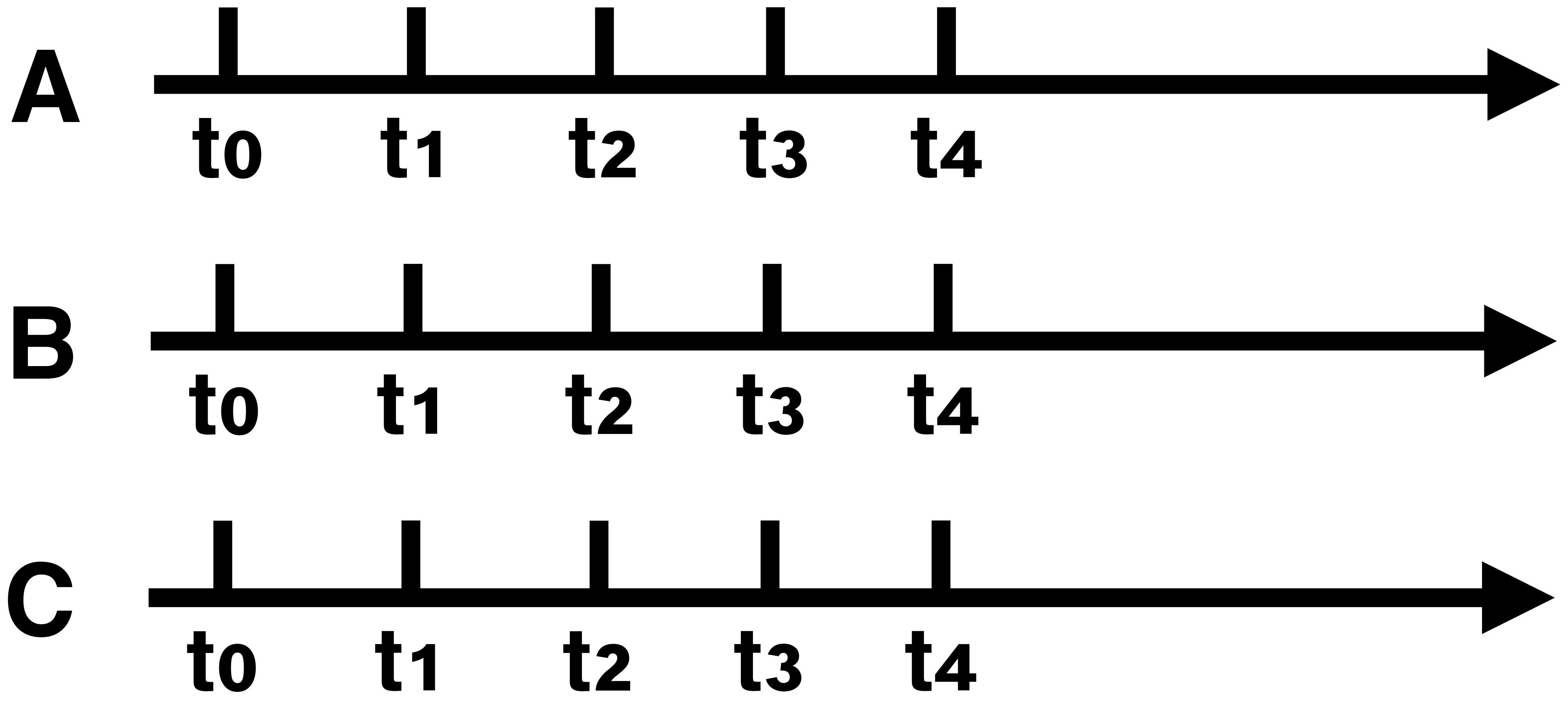
Distance Vector Converges Slowly

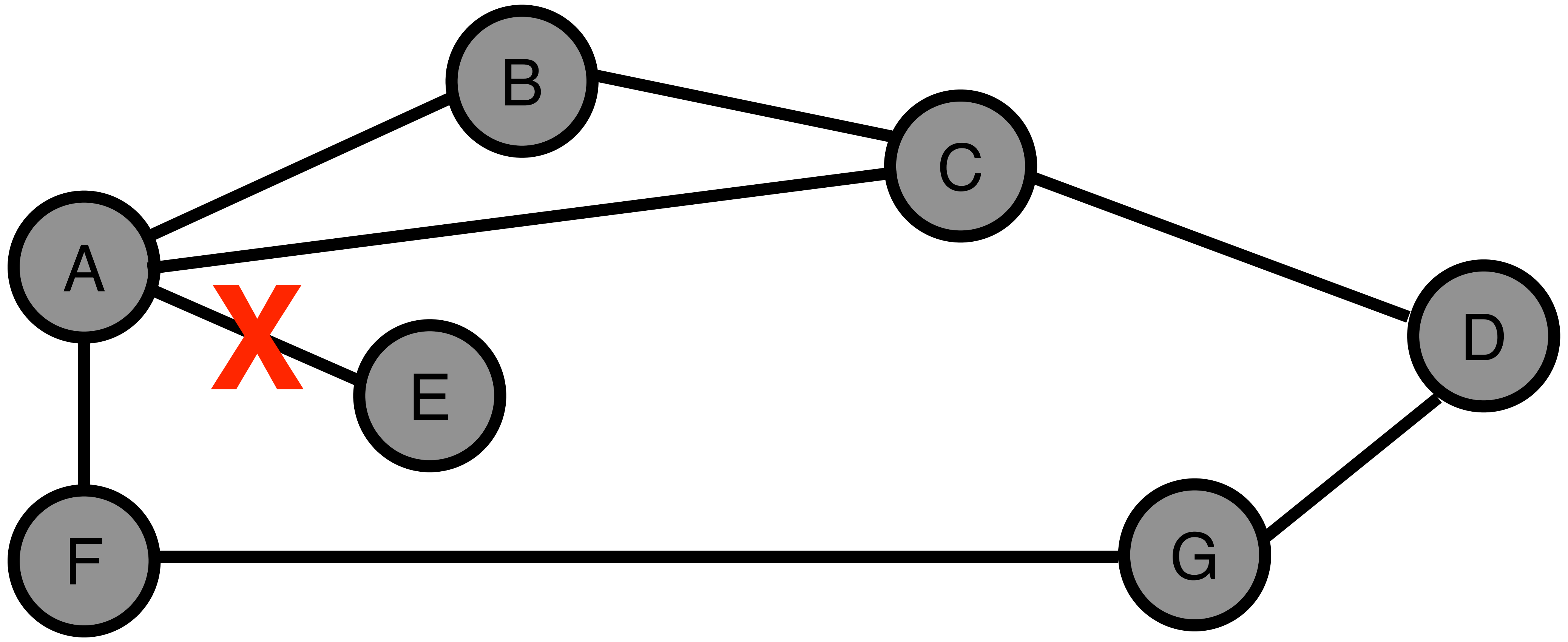




- At t_4, C receives the message from B (saying the distance to E is 3), and updates the routing table as **<E, 4>**
- At t_4, A receives the message from B (saying the distance to E is 3), and updates the routing table as **<E, 4>**
- **A will advertise this new changes to C, then C advertises B, B advertises A, …**

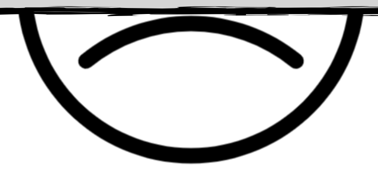
Distance Vector Converges Slowly





This cycle stops only when the distances reach some threshold that is large enough to be considered infinite

- **This is called the Count-to-infinity problem**



Count-to-infinity Problem: A Simple Fix

Use some relatively small number as an approximation of infinity

- The maximum number of hops to get across a network never exceeds 16

Routing Information Protocol (RIP)

Earliest IP routing protocol

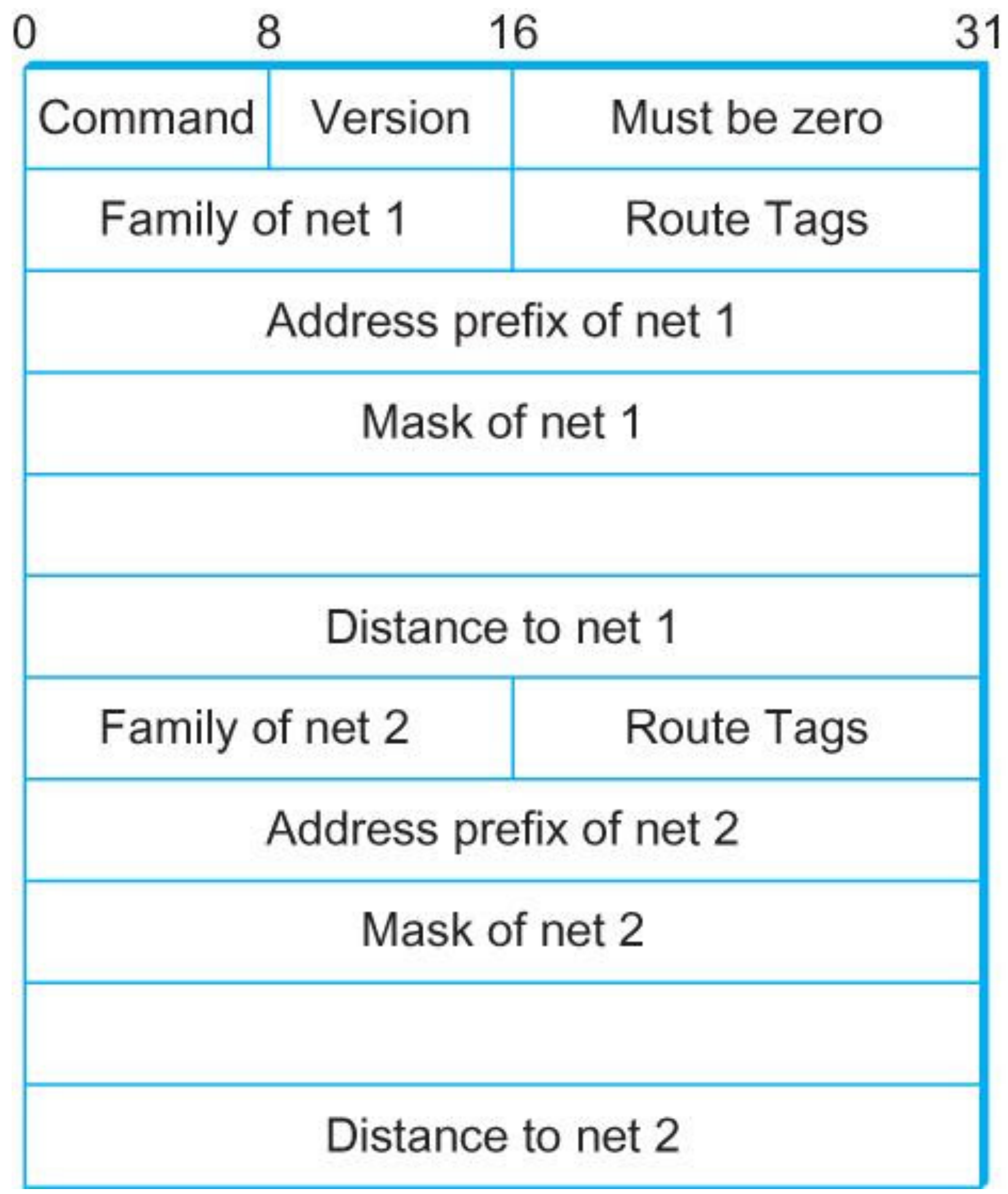
- 1982 BSD of Unix
- Current standard is version 2 (RFC 1723)

Features

- Cost: the number of hops
- “Infinity” = 16

Sending updates

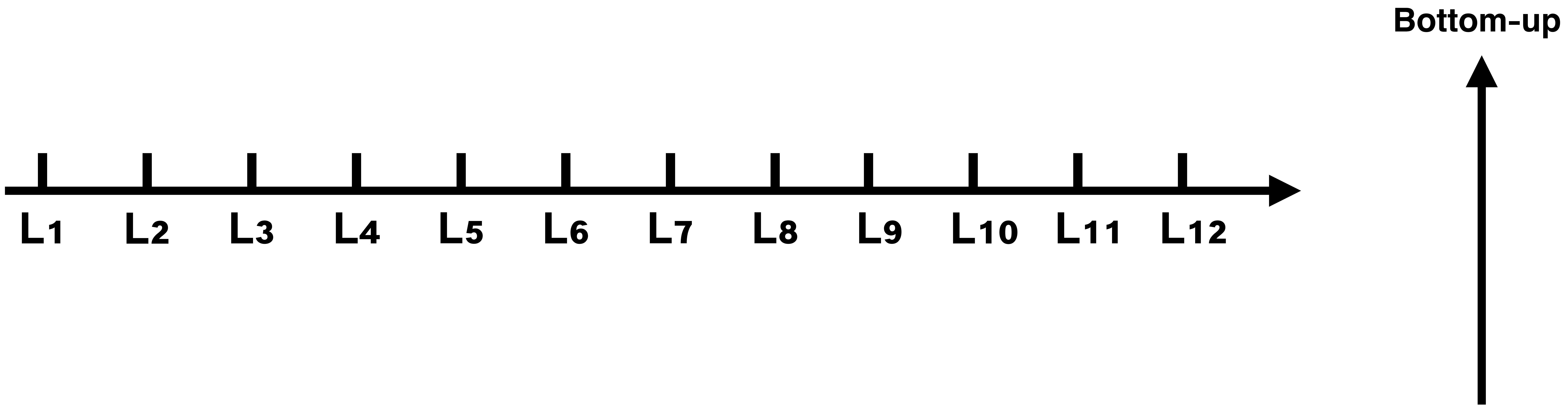
- Every router listens for updates on UDP port 520
- Frequency: 30 seconds
- Triggered when an entry is changed



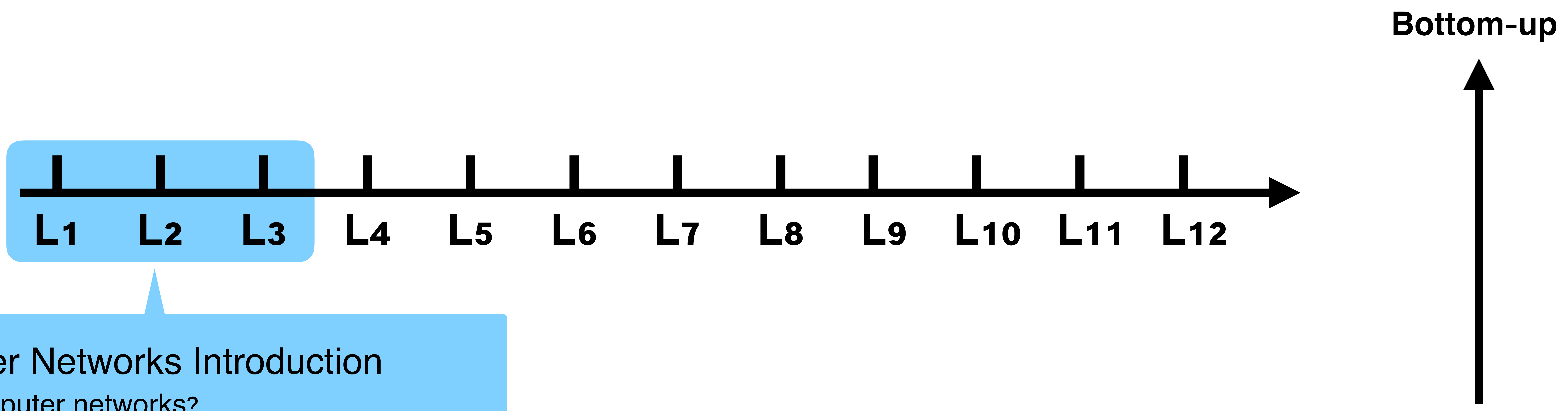
IP Router v.s. Ethernet Switch (Incomplete!)

	IP Router	Ethernet Switch
Layering	Layer 3	Layer 2
Packet Manipulation	Fragmentation and Reassembly; TTL update	N/A
Packet Forwarding	Based on the destination IP address	Based on destination Ethernet address; Run the spanning tree protocol to avoid forwarding loops
Routing	Based on the routing algorithm	N/A
Error Handling	Speak the ICMP protocol	N/A

Midterm 1 Review



Midterm 1 Review



Computer Networks Introduction

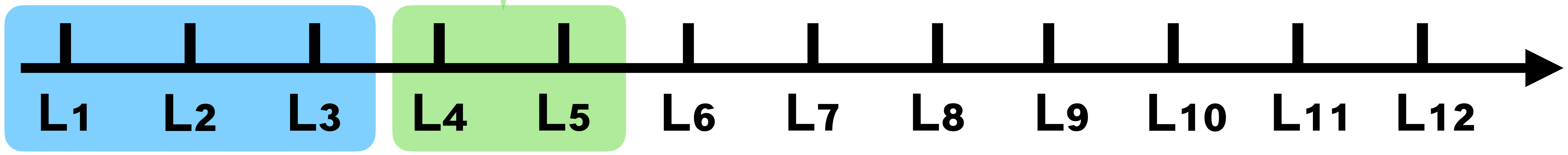
Q1: What are computer networks?
Q2: What are the requirements of computer networks?
Q3: What hardware elements are used?
Q4: What software components are needed?
Q5: How fast is the network?

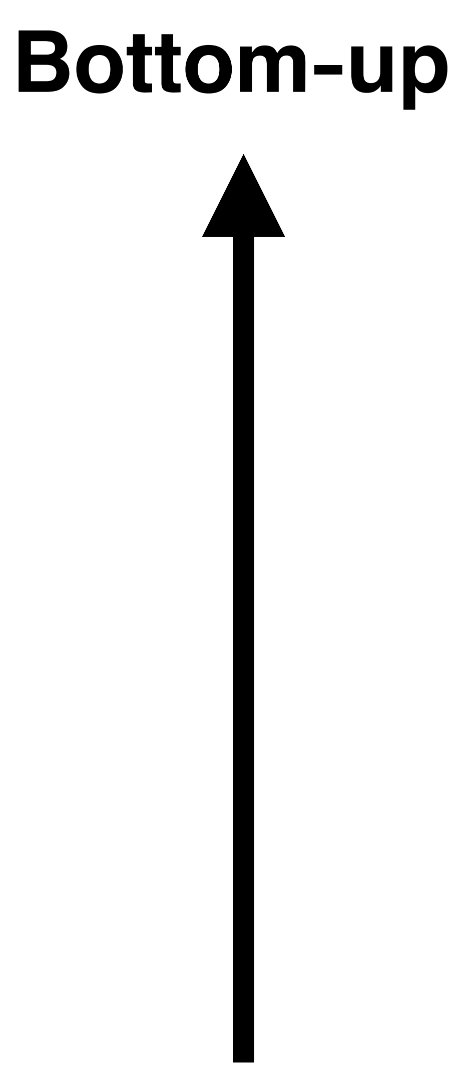
Midterm 1 Review

Physical Layer (L1): bits over wire

Q1: How to represent bits on the link?

Q2: How to propagate bits across the link reliably?





Computer Networks Introduction

Q1: What are computer networks?

Q2: What are the requirements of computer networks?

Q3: What hardware elements are used?

Q4: What software components are needed?

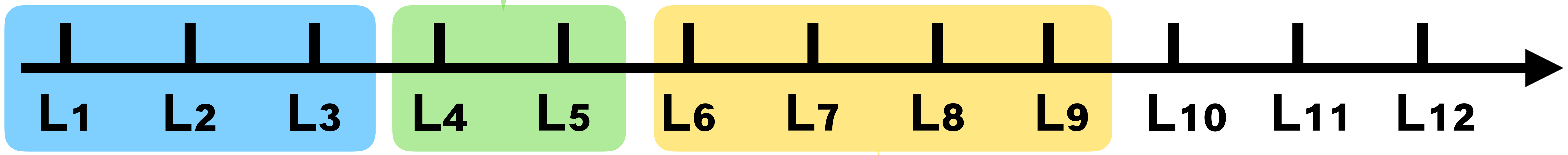
Q5: How fast is the network?

Midterm 1 Review

Physical Layer (L1): bits over wire

Q1: How to represent bits on the link?

Q2: How to propagate bits across the link reliably?



Bottom-up

Computer Networks Introduction

Q1: What are computer networks?

Q2: What are the requirements of computer networks?

Q3: What hardware elements are used?

Q4: What software components are needed?

Q5: How fast is the network?

Data Link Layer (L2): frames between NICs

Q1: How to identify a frame from bit streams?

Q2: How to handle transmission errors?

Q3: How do frames traverse NICs and switches?

Q4: How to achieve concurrent transmission?

Q5: How to ensure reliable frame delivery?

Midterm 1 Review

Physical Layer (L1): bits over wire

Q1: How to represent bits on the link?

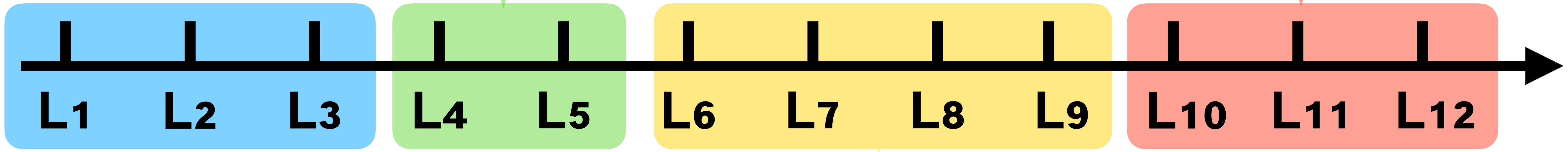
Q2: How to propagate bits across the link reliably?

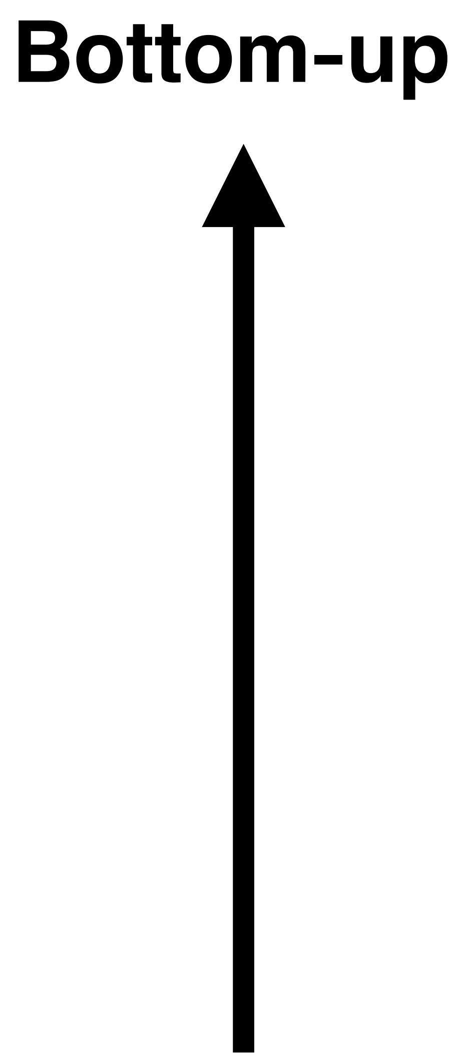
IP Layer (L3): Datagrams between hosts

Q1: How to address any hosts in any hosts?

(Or how to assign the IP address)?

Q2: How to decide the forwarding paths among routers?





Computer Networks Introduction

Q1: What are computer networks?

Q2: What are the requirements of computer networks?

Q3: What hardware elements are used?

Q4: What software components are needed?

Q5: How fast is the network?

Data Link Layer (L2): frames between NICs

Q1: How to identify a frame from bit streams?

Q2: How to handle transmission errors?

Q3: How do frames traverse NICs and switches?

Q4: How to achieve concurrent transmission?

Q5: How to ensure reliable frame delivery?

Terminology

1. Host
2. NIC
3. Multi-port I/O bridge
4. Protocol
5. RTT
6. Packet
7. Header
8. Payload
9. BDP
10. Baud rate
11. Frame/Framing
12. Parity bit
13. Checksum
14. Ethernet
15. MAC
16. (L2) Switch
17. Broadcast
18. Acknowledgement
19. Timeout
20. Datagram
21. TTL
22. MTU
23. Best effort
24. (L3) Router
25. Subnet mask
26. CIDR
27. Converge
28. Count-to-infinity

Principle

1. Layering
2. Minimal States
3. Hierarchy

Technique

1. NRZ Encoding
2. NRZI Encoding
3. Manchester Encoding
4. 4B/5B Encoding
5. Byte Stuffing
6. Byte Counting
7. Bit Stuffing
8. 2-D Parity
9. CRC
10. MAC Learning
11. Store-and-Forward
12. Cut-through
13. Spanning Tree
14. CSMA/CD
15. Stop-and-Wait
16. Sliding Window
16. Fragmentation and Reassembly
17. Path MTU discovery
18. DHCP
19. Subnetting
20. Supernetting
21. Longest prefix match
22. Distance vector routing (RIP)

Summary

Today's takeaways

#1: Routing is the process of building the routing table to instruct the forwarding logic

#2: Efficient routing mechanism should be adapted to the infrastructure and network variation

#3: The distance vector routing protocol decides the preferred communication path by exchanging the distance vector among neighboring routers

Next lecture

- In-class Midterm1