PART XIII

ROUTING: CORES, PEERS, AND ALGORITHMS

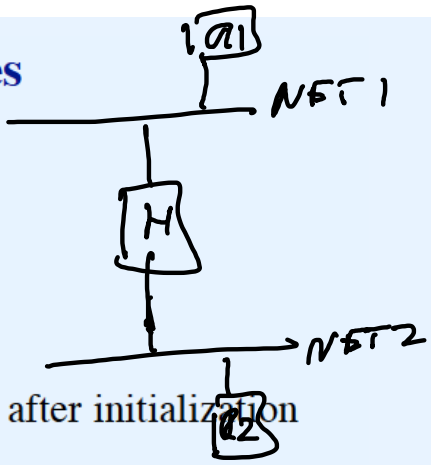
Internet Routing
(review)

- IP implements datagram forwarding
- Both hosts and routers
  - Have an IP module
  - Forward datagrams
- IP forwarding is table-driven
- Table known as *routing table*

How / When Are IP Routing Tables Built?

- Depends on size / complexity of internet
- Static routing
  - Fixes routes at boot time
  - Useful only for simplest cases
- Dynamic routing
  - Table initialized at boot time
  - Values inserted / updated by protocols that propagate route information
  - Necessary in large internets

Routing Tables



- Two sources of information
  - Initialization (e.g., from disk)
  - Update (e.g., from protocols)
- Hosts tend to freeze the routing table after initialization
- Routers use protocols to learn new information and update their routing table dynamically

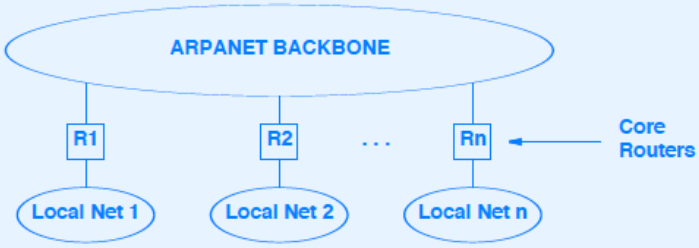
Routing With Partial Information

A host can forward datagrams successfully even if it only has partial routing information because it can rely on a router.

Routing With Partial Information
(continued)

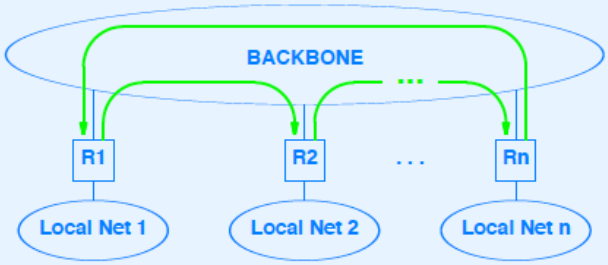
The routing table in a given router contains partial information about possible destinations. Routing that uses partial information allows sites autonomy in making local routing changes, but introduces the possibility of inconsistencies that may make some destinations unreachable from some sources.

Original Internet



- Backbone network plus routers each connecting a local network

Worst Case If All Routers Contain A Default Route

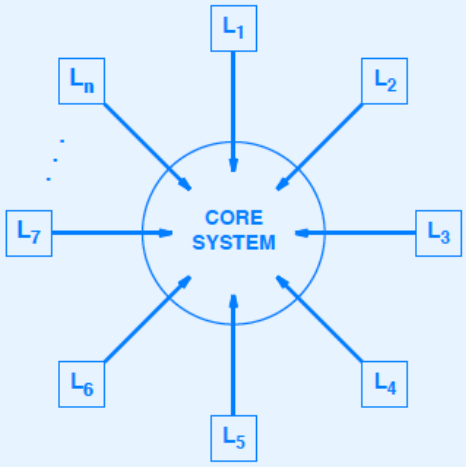


- Datagram sent to nonexistent destination loops until TTL expires

Original Routing Architecture

- Small set of ‘‘core’’ routers with complete information about all destinations
- Other routers know local destinations and use the core as central router

Illustration Of Default Routes In The Original Internet Core



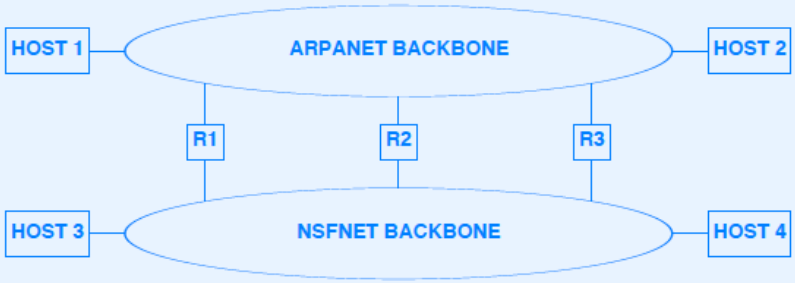
Disadvantage Of Original Core

- Central bottleneck for all traffic
- No shortcut routes possible
- Does not scale

Beyond A Core Architecture

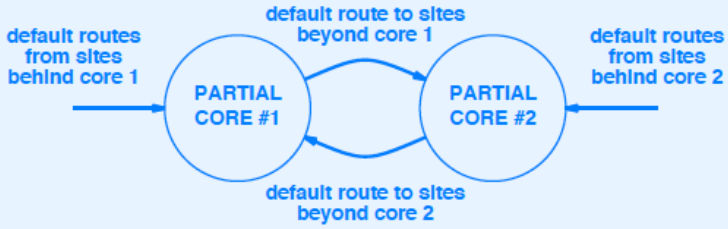
- Single core insufficient in world where multiple ISPs each have a wide-area backbone
- Two backbones first appeared when NSF and ARPA funded separate backbone networks
- Known as *peer backbones*

Illustration Of Peer Backbones



Partial Core

- Cannot have ‘‘partial core’’ scheme
- Proof:



- Datagram destined for nonexistent destination loops until TTL expires

When A Core Routing Architecture Works

A core routing architecture assumes a centralized set of routers serves as the repository of information about all possible destinations in an internet. Core systems work best for internets that have a single, centrally managed backbone. Expanding the topology to multiple backbones makes routing complex; attempting to partition the core architecture so that all routers use default routes introduces potential routing loops.

General Idea

- Have a set of core routers know routes to all locations
- Devise a mechanism that allows other routers to contact the core to learn routes (spread necessary routing information automatically)
- Continually update routing information

Automatic Route Propagation

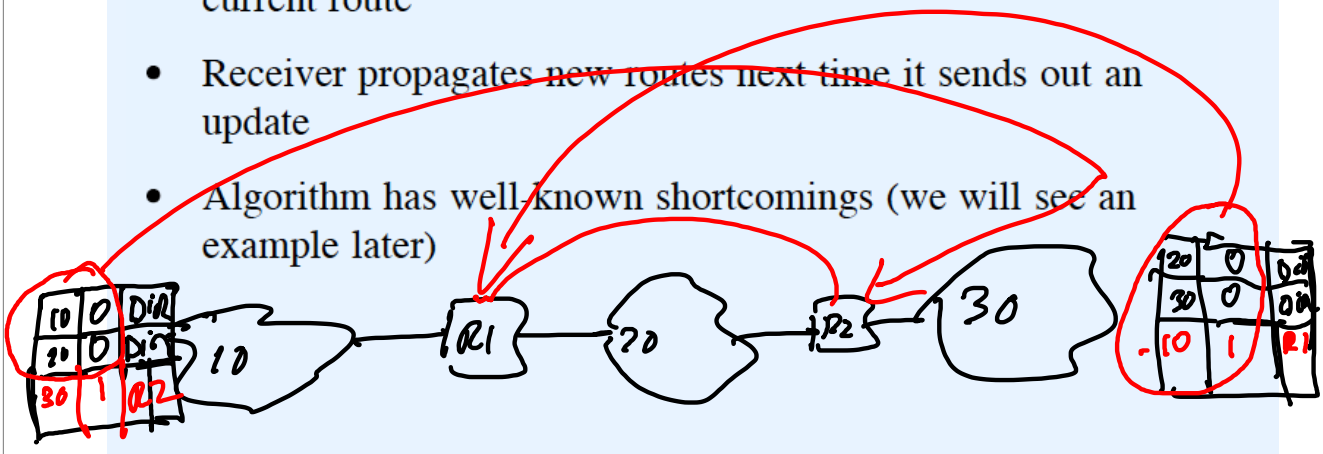
- Two basic algorithms used by routing update protocols
  - Distance-vector
  - Link-state
- Many variations in implementation details

Distance-Vector Algorithm

- Initialize routing table with one entry for each directly-connected network
- Periodically run a distance-vector update to exchange information with routers that are reachable over directly-connected networks

Dynamic Update With Distance-Vector

- One router sends list of its routes to another
- List contains pairs of destination network and distance
- Receiver replaces entries in its table by routes to the sender if routing through the sender is less expensive than the current route
- Receiver propagates new routes next time it sends out an update
- Algorithm has well-known shortcomings (we will see an example later)



Example Of Distance-Vector Update

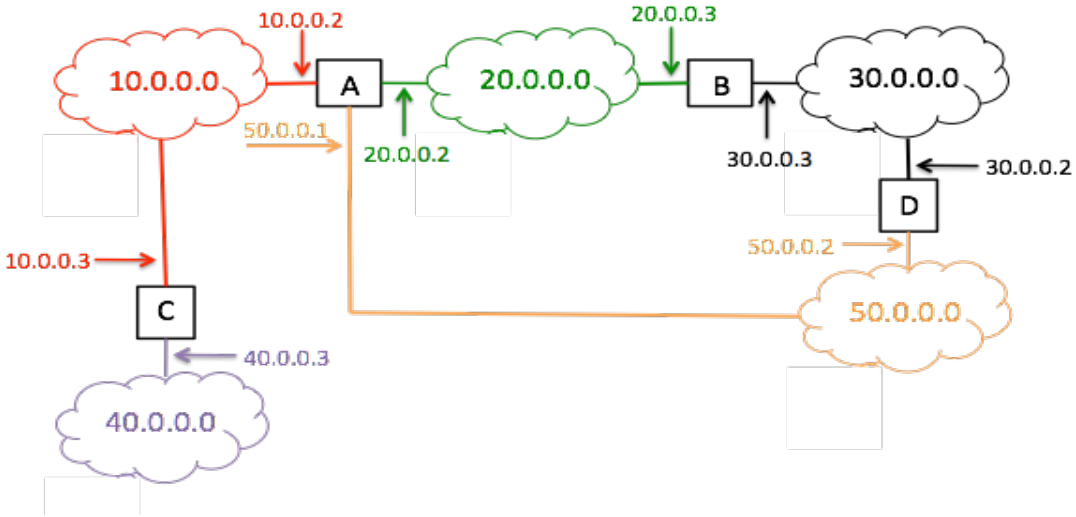
Destination	Distance	Route
Net 1	0	direct
Net 2	0	direct
Net 4	8	Router L
Net 17	5	Router M
Net 24	6	Router J
Net 30	2	Router Q
Net 42	2	Router J

(a)

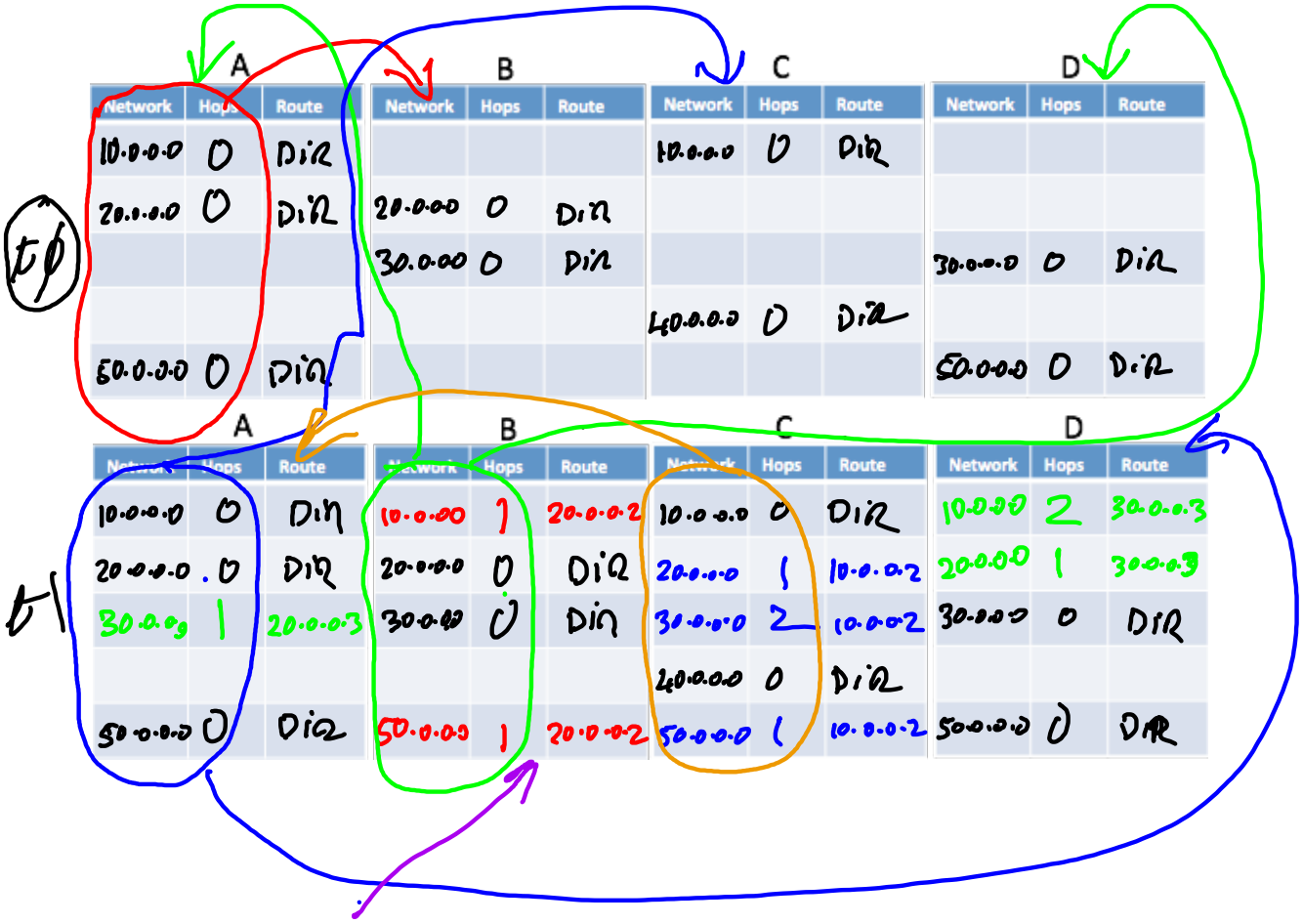
Destination	Distance
Net 1	2
Net 4	3
Net 17	6
Net 21	4
Net 24	5
Net 30	10
Net 42	3

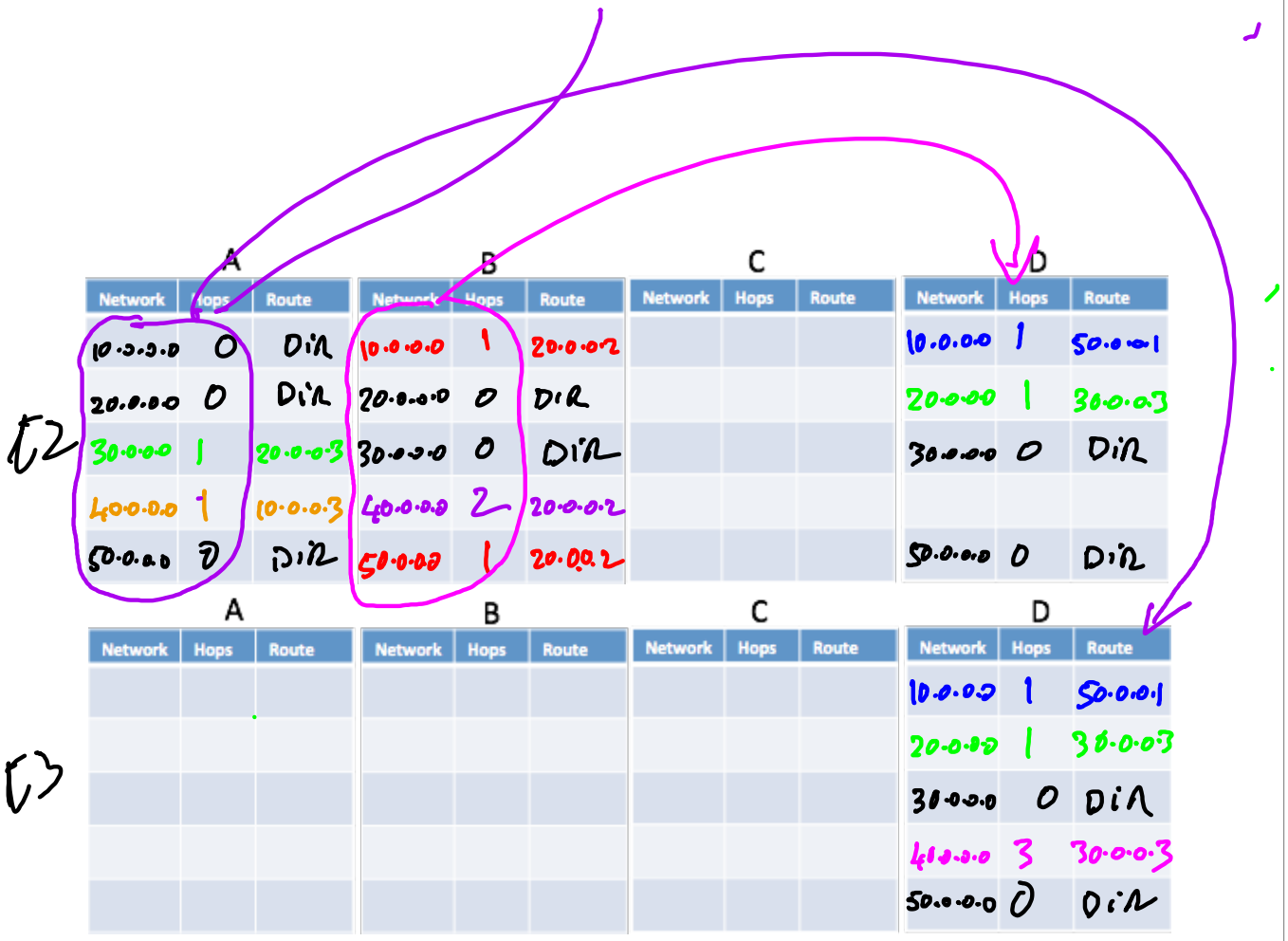
(b)

- (a) is existing routing table
- (b) incoming update (marked items cause change)



#	From Router/Tx – To Router/Tx	#	From Router/Tx – To Router/Tx	#	From Router/Tx – To Router/Tx
1	A/T0 – B/T0	7	A/T2 – B/T1	13	
2	B/T1 – A/T0	8	B/T2 – D/T2	14	
3	B/T1 – D/T0	9	A/T2 – D/T3	15	
4	A/T1 – C/T0	10		16	
5	A/T1 – D/T1	11		17	
6	C/T1 – A/T1	12		18	


A
B
C
D
Network
Hops
Route
10.0.0.0 0 DIR
20.0.0.0 0 DIR
50.0.0.0 0 DIR
20.0.0.0 0 DIR
30.0.0.0 0 DIR
10.0.0.0 0 DIR
40.0.0.0 0 DIR
30.0.0.0 0 DIR
50.0.0.0 0 DIR
t0
10.0.0.0 0 DIR
20.0.0.0 0 DIR
30.0.0.0 1 20.0.0.3
50.0.0.0 0 DIR
10.0.0.0 1 20.0.0.2
20.0.0.0 0 DIR
30.0.0.0 0 DIR
50.0.0.0 1 20.0.0.2
10.0.0.0 0 DIR
20.0.0.0 1 10.0.0.2
30.0.0.0 2 10.0.0.2
40.0.0.0 0 DIR
50.0.0.0 1 10.0.0.2
10.0.0.0 2 30.0.0.3
20.0.0.0 1 30.0.0.3
30.0.0.0 0 DIR
50.0.0.0 0 DIR
t1


A
Network Hops Route
10.0.0.0 0 Dir
20.0.0.0 0 Dir
30.0.0.0 1 20.0.0.3
40.0.0.0 1 10.0.0.3
50.0.0.0 0 Dir
B
Network Hops Route
10.0.0.0 1 20.0.0.2
20.0.0.0 0 Dir
30.0.0.0 0 Dir
40.0.0.0 2 20.0.0.2
50.0.0.0 1 20.0.0.2
C
Network Hops Route
D
Network Hops Route
10.0.0.0 1 50.0.0.1
20.0.0.0 1 30.0.0.3
30.0.0.0 0 Dir
50.0.0.0 0 Dir
A
Network Hops Route
B
Network Hops Route
C
Network Hops Route
D
Network Hops Route
10.0.0.0 1 50.0.0.1
20.0.0.0 1 30.0.0.3
30.0.0.0 0 Dir
40.0.0.0 3 30.0.0.3
50.0.0.0 0 Dir

T4

A

Network	Hops	Route

B

Network	Hops	Route

C

Network	Hops	Route

D

Network	Hops	Route
10.0.0.0	1	50.0.0.2
20.0.0.0	1	30.0.0.3
30.0.0.0	0	Dir
40.0.0.0	2	50.0.0.1
50.0.0.0	0	Dir

A

Network	Hops	Route

B

Network	Hops	Route

C

Network	Hops	Route

D

Network	Hops	Route

A

Network	Hops	Route

B

Network	Hops	Route

C

Network	Hops	Route

D

Network	Hops	Route

A

Network	Hops	Route

B

Network	Hops	Route

C

Network	Hops	Route

D

Network	Hops	Route

A

Network	Hops	Route

B

Network	Hops	Route

C

Network	Hops	Route

D

Network	Hops	Route

A

Network	Hops	Route

B

Network	Hops	Route

C

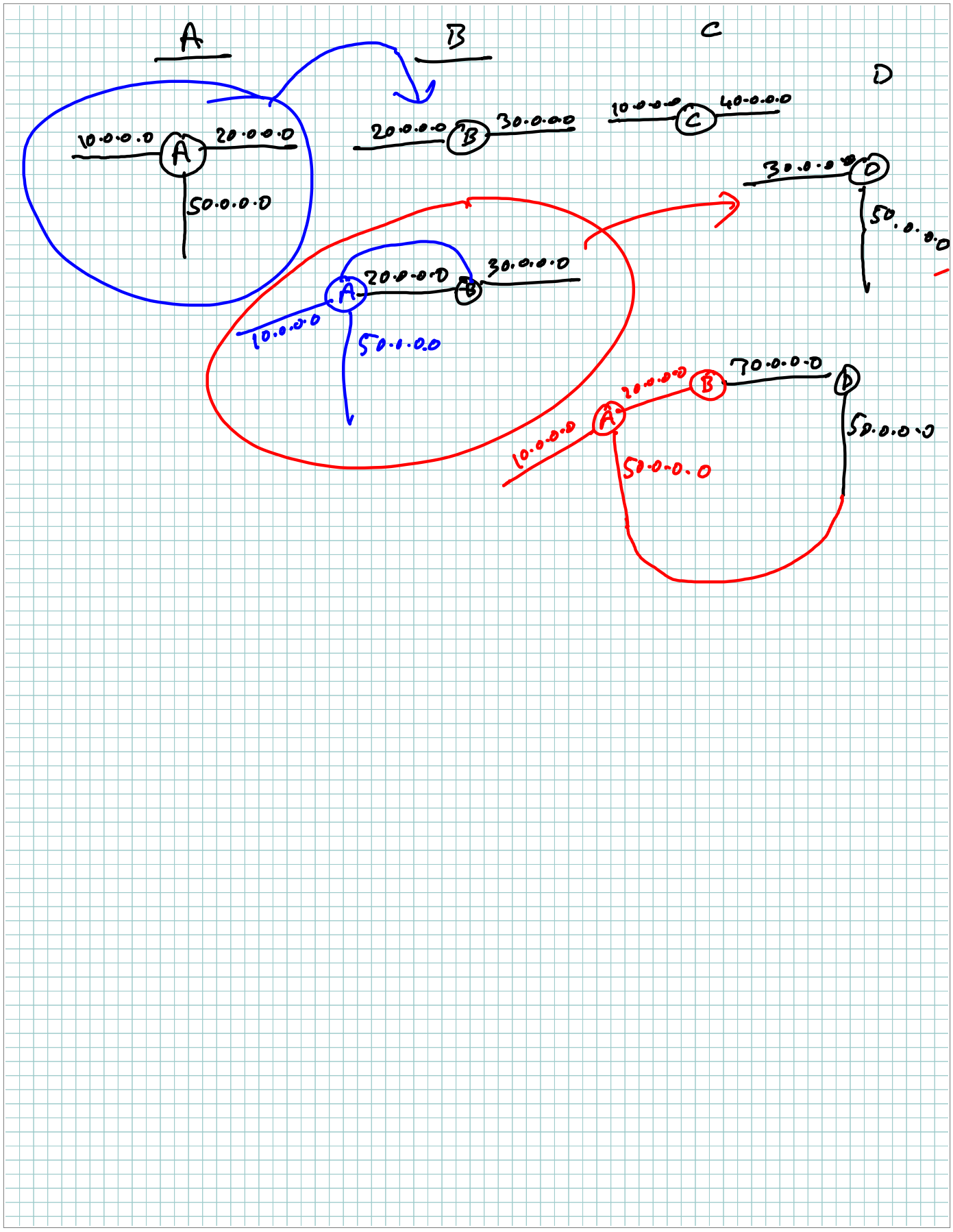
Network	Hops	Route

D

Network	Hops	Route

Link-State Algorithm

- Alternative to distance-vector
- Distributed computation
  - Broadcast information
  - Allow each router to compute shortest paths
- Avoids problem where one router can damage the entire internet by passing incorrect information
- Also called *Shortest Path First* (SPF)


A
B
C
D
10.0.0.0
A
20.0.0.0
50.0.0.0
20.0.0.0
B
30.0.0.0
10.0.0.0
C
40.0.0.0
30.0.0.0
D
50.0.0.0
A
20.0.0.0
B
30.0.0.0
10.0.0.0
50.1.0.0
70.0.0.0
D
B
20.0.0.0
A
10.0.0.0
50.0.0.0
50.0.0.0

Link-State Update

- Participating routers learn internet topology
- Think of routers as nodes in a graph, and networks connecting them as edges or links
- Pairs of directly-connected routers periodically
  - Test link between them
  - Propagate (broadcast) status of link
- All routers
  - Receive link status messages
  - Recompute routes from their local copy of information

Summary

- Routing tables can be
  - Initialized at startup (host or router)
  - Updated dynamically (router)
- Original Internet used core routing architecture
- Current Internet accommodates peer backbones
- Two important routing algorithms
  - Distance-vector
  - Link state