CSE 461: Computer networks

Spring 2021

Ratul Mahajan

Network Layer (Routing)

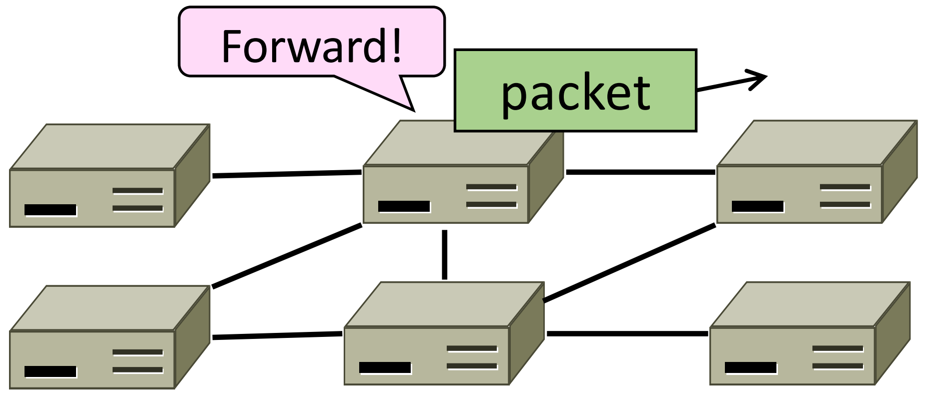
Recap: Why do we need a Network layer?

- Internetworking
    - Need to connect different link layer networks
- Addressing
    - Need a globally unique way to "address" hosts
- Routing and forwarding
    - Need to find and traverse paths between hosts

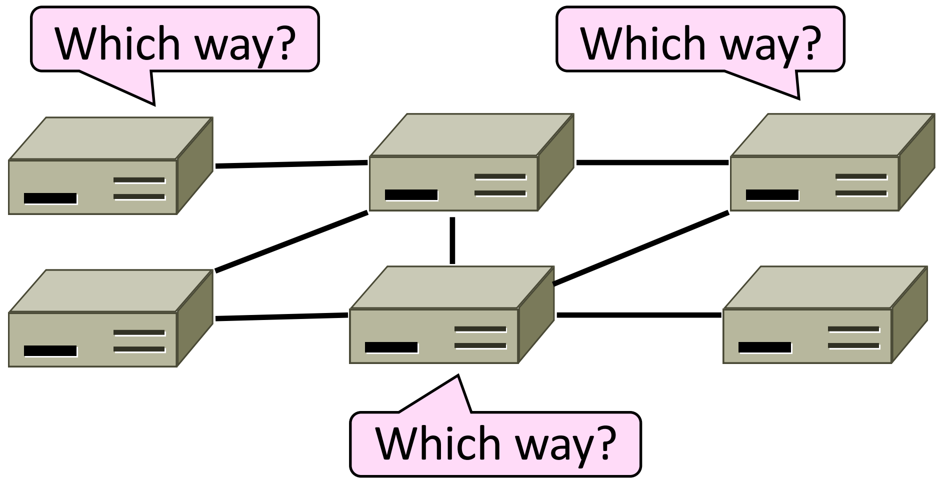
$\left.\begin{matrix} \\ \\ \end{matrix}\right\}$ Now this

Recap: Routing versus Forwarding

- Forwarding is the process of sending a packet on its way



- Routing is the process of deciding in which direction to send traffic



Overview of Internet Routing and Forwarding

- Hosts on same network have IPs in the same IP prefix
- Hosts send off-network traffic to the gateway router

- Routers discover routes to different prefixes (<u>routing</u>)
- Routers use <u>longest prefix matching</u> to send packets to the right next hop (forwarding)

Longest Prefix Matching

- Prefixes in the forwarding table can overlap

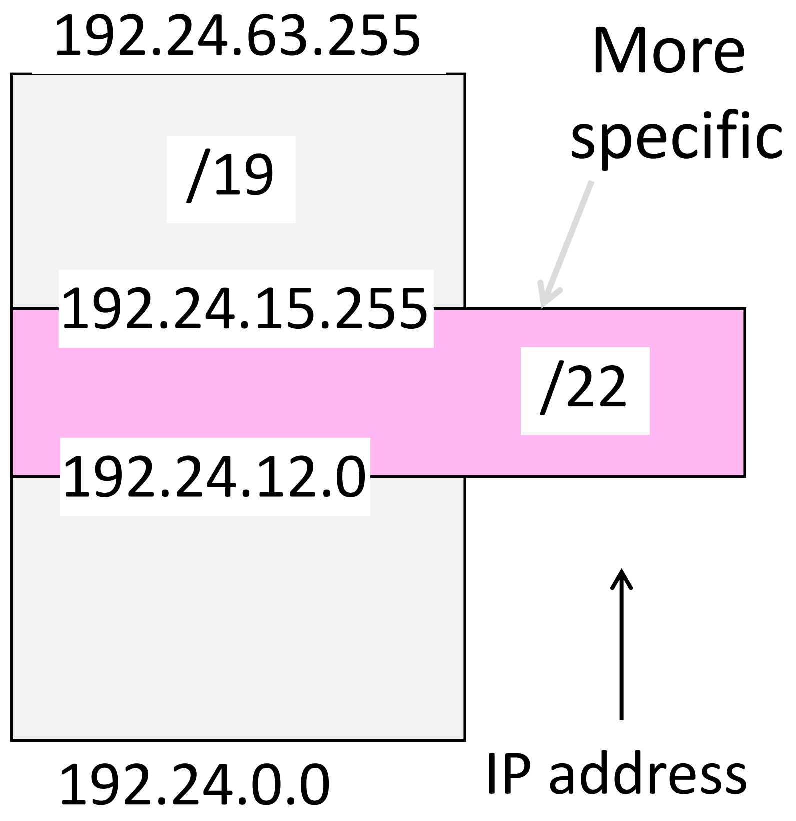
Prefix	Next Hop
0.0.0.0/0	A
192.24.0.0/19	B
192.24.12.0/22	C

- <u>Longest prefix matching</u> forwarding rule:
  - For each packet, find the longest prefix that contains the destination address, i.e., the most specific entry
  - Forward the packet to the next hop router for that prefix

Longest Prefix Matching (2)

Prefix	Next Hop
192.24.0.0/19	D
192.24.12.0/22	B

192.24.6.0 → ?
192.24.14.32 → ?
192.24.54.0 → ?



Flexibility of Longest Prefix Matching

- Can provide default behavior, with less specifics
  - Send traffic going outside an organization to a border router (gateway)
- Can special case behavior, with more specifics
  - For performance, economics, security, …

Performance of Longest Prefix Matching

- Uses hierarchy for a compact table
  - Relies on use of large prefixes
- Lookup more complex than table
  - Used to be a concern for fast routers
  - Not an issue in practice these days

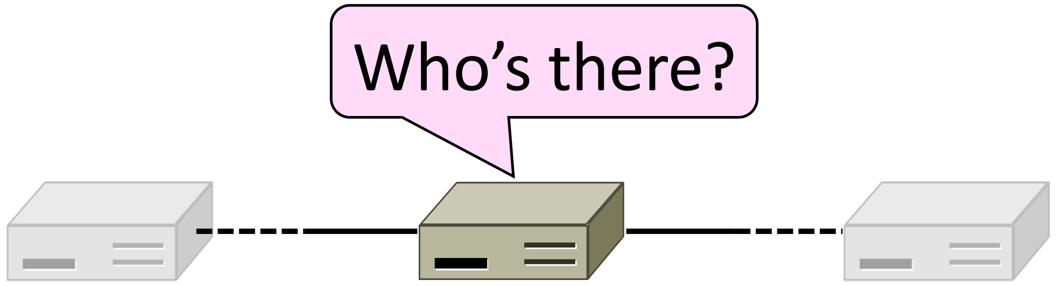
Goals of Routing Algorithms

- We want several properties of any routing scheme:

Property	Meaning
Correctness	Finds paths that work
Efficient paths	Uses network bandwidth well
Fair paths	Doesn’t starve any nodes
Fast convergence	Recovers quickly after changes
Scalability	Works well as network grows large

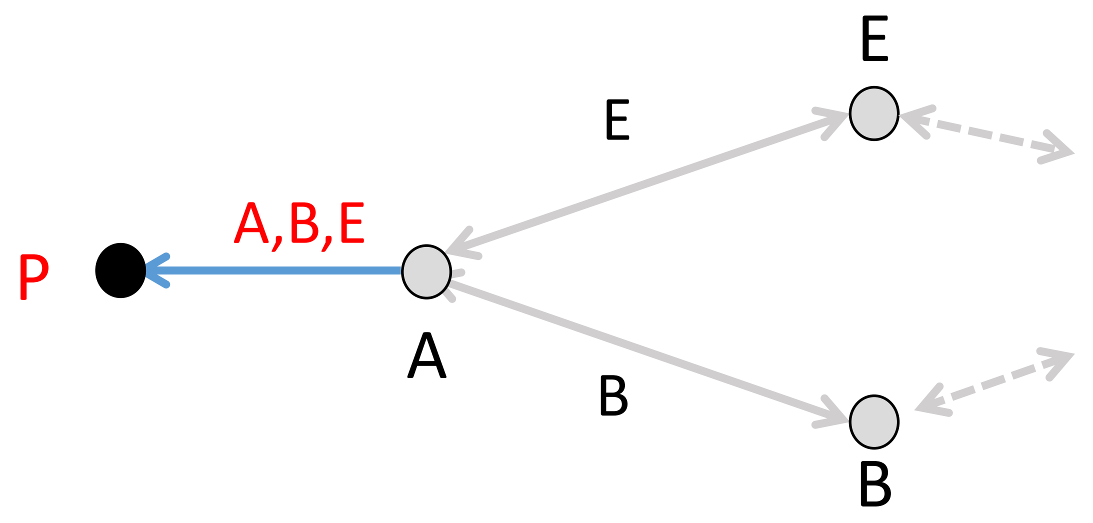
Rules of Fully Distributed Routing

- All nodes are alike; no controller
- Nodes learn by exchanging messages with neighbors
- Nodes operate concurrently
- There may be node/link/message failures



Simple routing that obeys the rules

- Send out routes for hosts you have paths to
  - And the routes they've sent you



- This works
  - All routers find a path to all hosts
- But scales poorly!

Recall: Internet Size

- Over 4 billion people
- 50B devices connect

Impact of Network Growth

1. Forwarding tables grow
   - Larger router memories, may increase lookup time
2. Routing messages grow
   - Need to keeps all nodes informed of larger topology
3. Routing computation grows
   - Shortest path calculations grow faster than the network

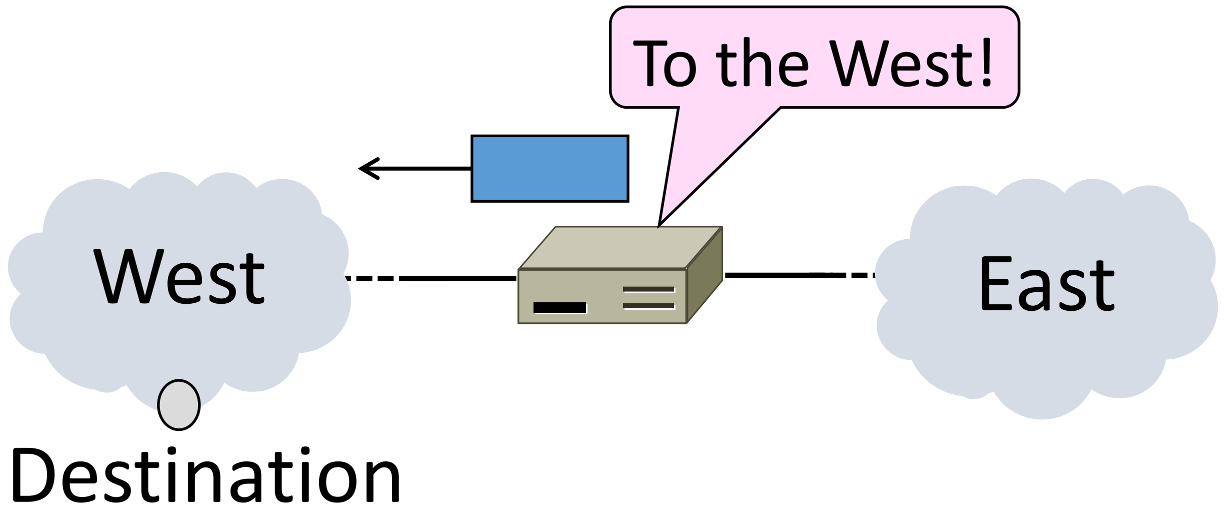
Techniques to Scale Routing

- First: Network hierarchy
  - Route to network regions
- Next: IP prefix aggregation
  - Combine, and split, prefixes

Scaling Idea 1: Hierarchical Routing

Idea

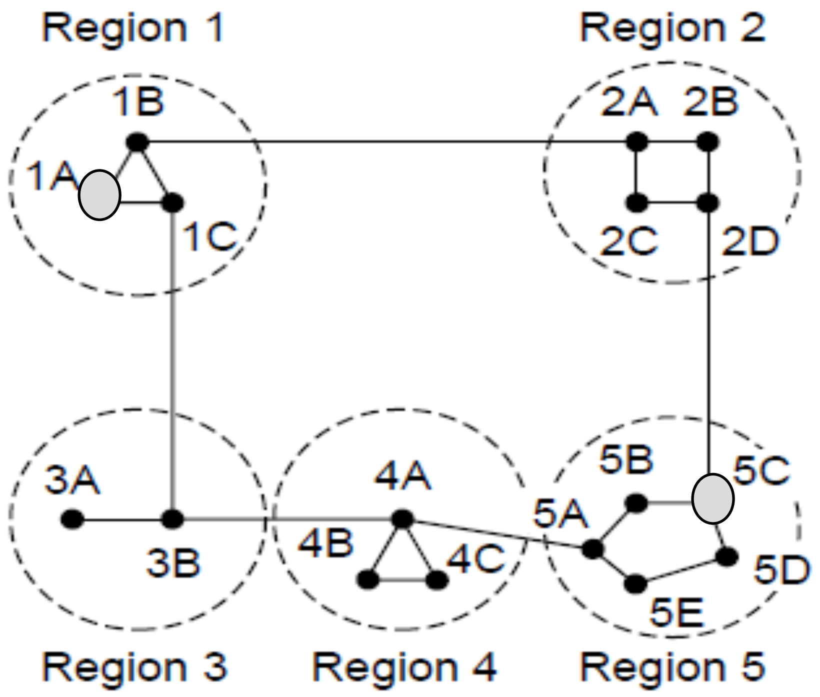
- Scale routing using hierarchy with regions
  - Route to regions, not individual nodes



Hierarchical Routing

- Introduce a larger routing unit
  - IP prefix (hosts) ← from one host
  - Region, e.g., ISP network
- Route first to the region, then to the IP prefix within the region
  - Hide details within a region from outside of the region

Hierarchical Routing (2)



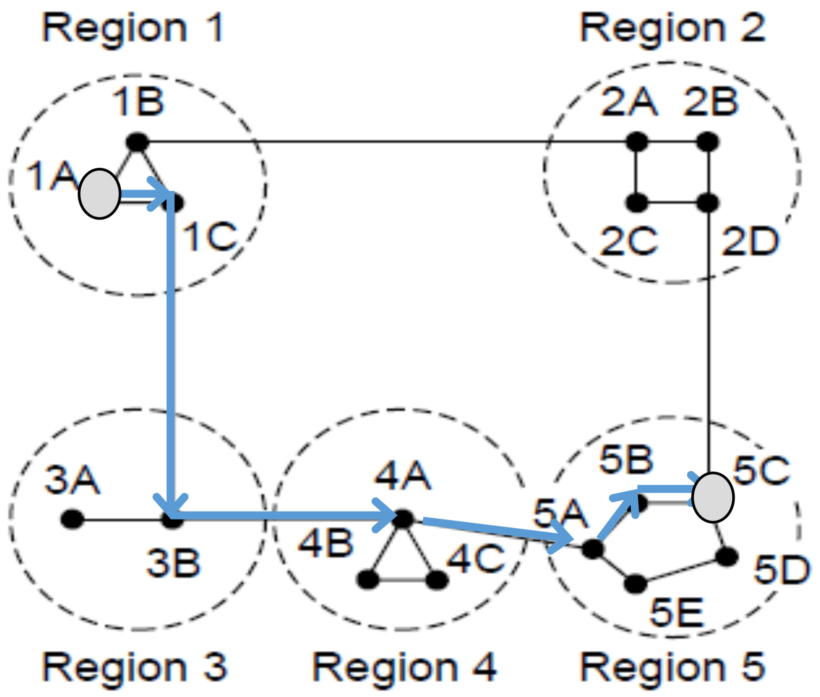
Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

Hierarchical Routing (3)



Full table for 1A

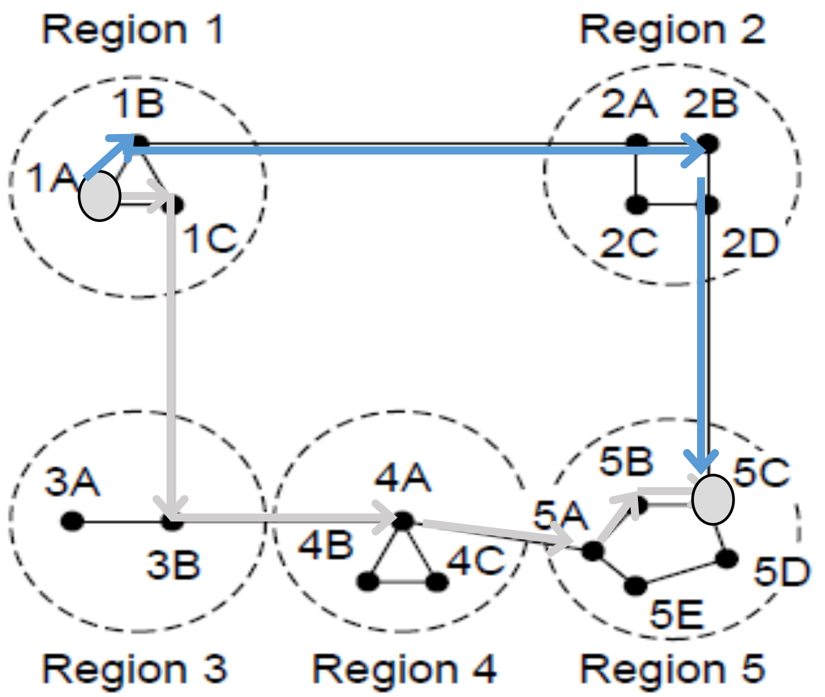
Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

Hierarchical Routing (4)

- Penalty is longer paths



Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

1C is best route to region 5, except for destination 5C

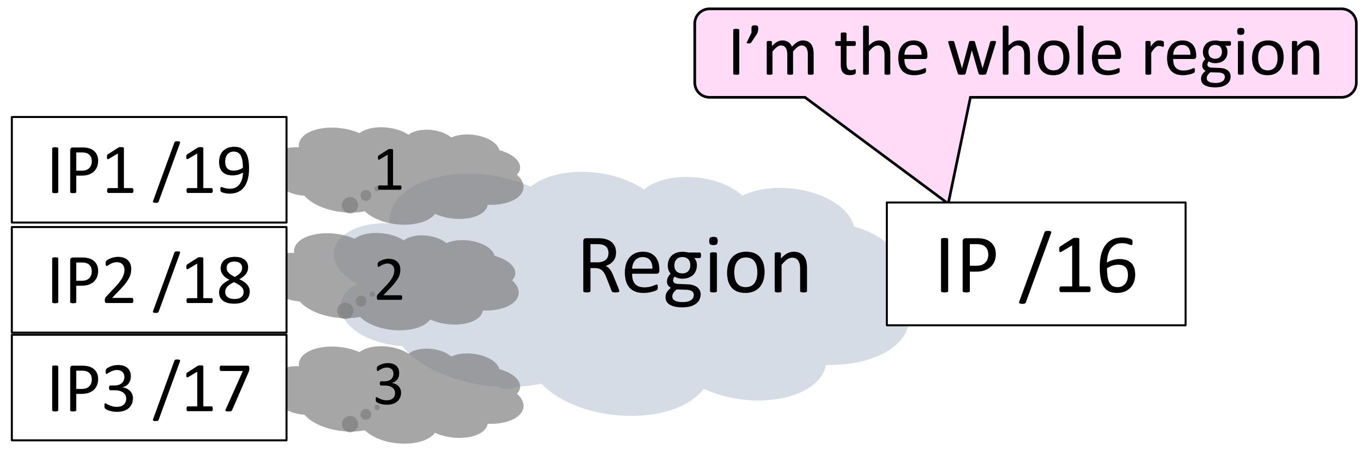
Observations

- Outside a region, nodes have <u>one route</u> to all hosts within the region
    - This gives savings in table size, messages and computation
- However, each node may have a <u>different route</u> to an outside region
    - Routing decisions are still made by individual nodes; there is no single decision made by a region

Scaling Idea 2:
IP Prefix Aggregation and Subnets

Idea

- Scale routing by adjusting the size of IP prefixes
  - Split (subnets) and join (aggregation)



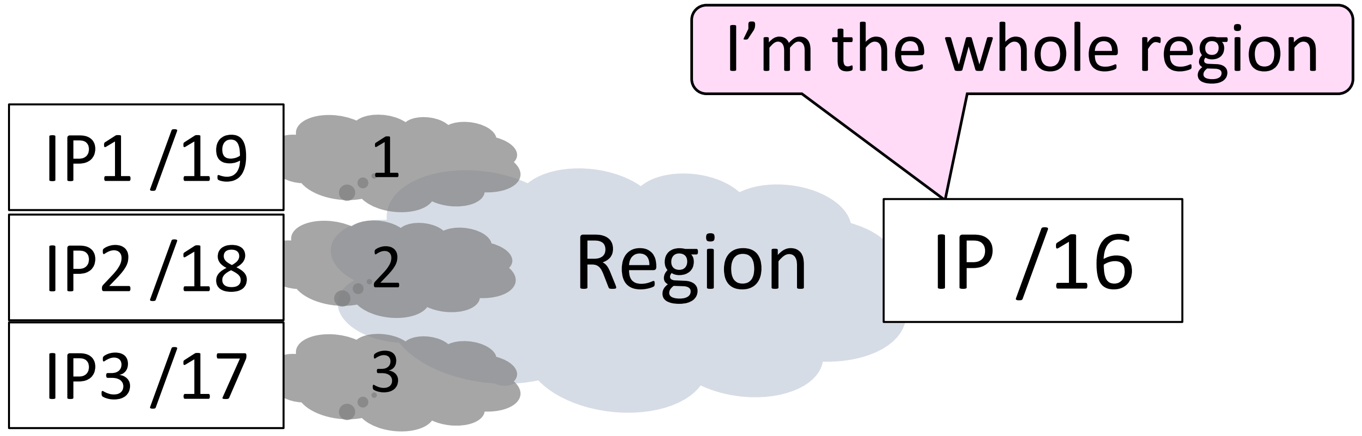
Recall

- IP addresses are allocated in blocks called IP prefixes, e.g., 18.31.0.0/16
  - Hosts on one network in same prefix
- "/N" prefix has the first N bits fixed and contains 2^{32-N} addresses
  - E.g., a "/24" has 256 addresses
- Routers keep track of prefix lengths
  - Use it as part of longest prefix matching

Routers can change prefix lengths without affecting hosts

Prefixes and Hierarchy

- IP prefixes help to scale routing, but can go further
  - Use a less specific (larger) IP prefix as a name for a region



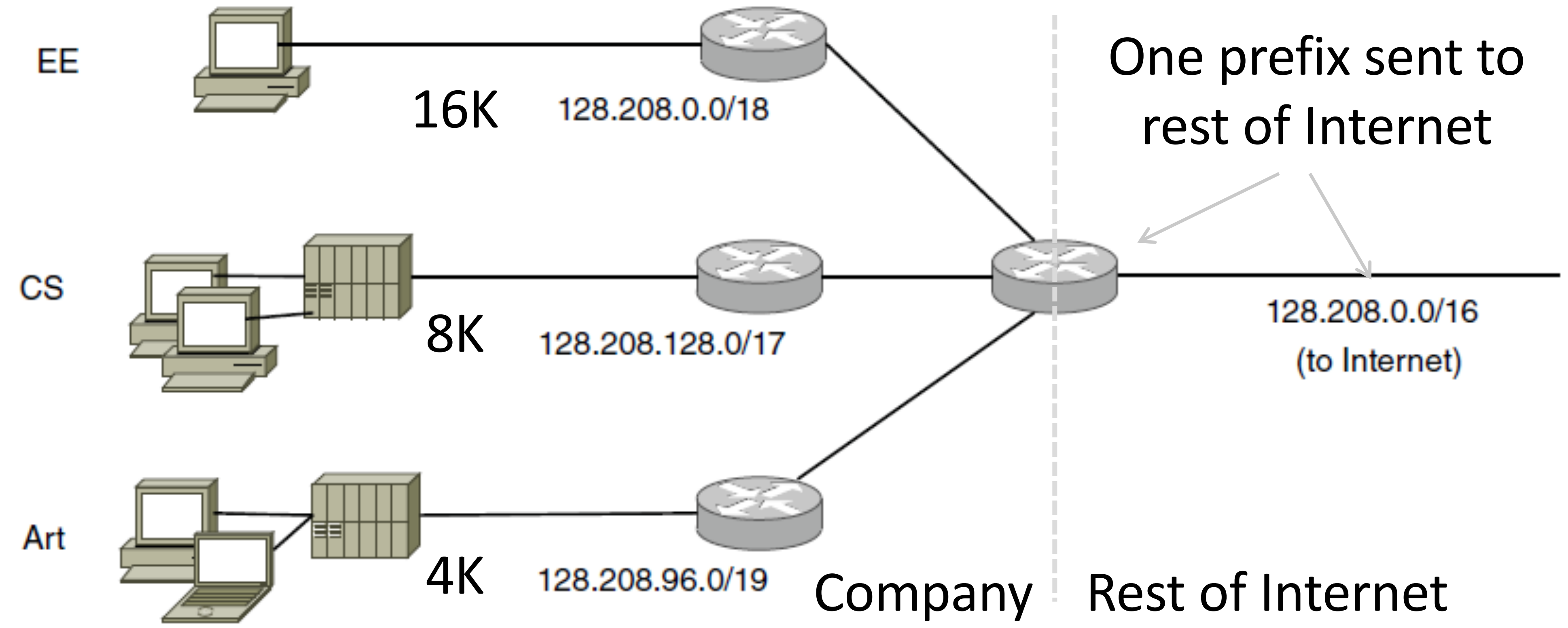
Subnets and Aggregation

- Two use cases for adjusting the size of IP prefixes; both reduce routing table

1. Subnets
   - Internally split one large prefix into multiple smaller ones
2. Aggregation
   - Join multiple smaller prefixes into one large prefix

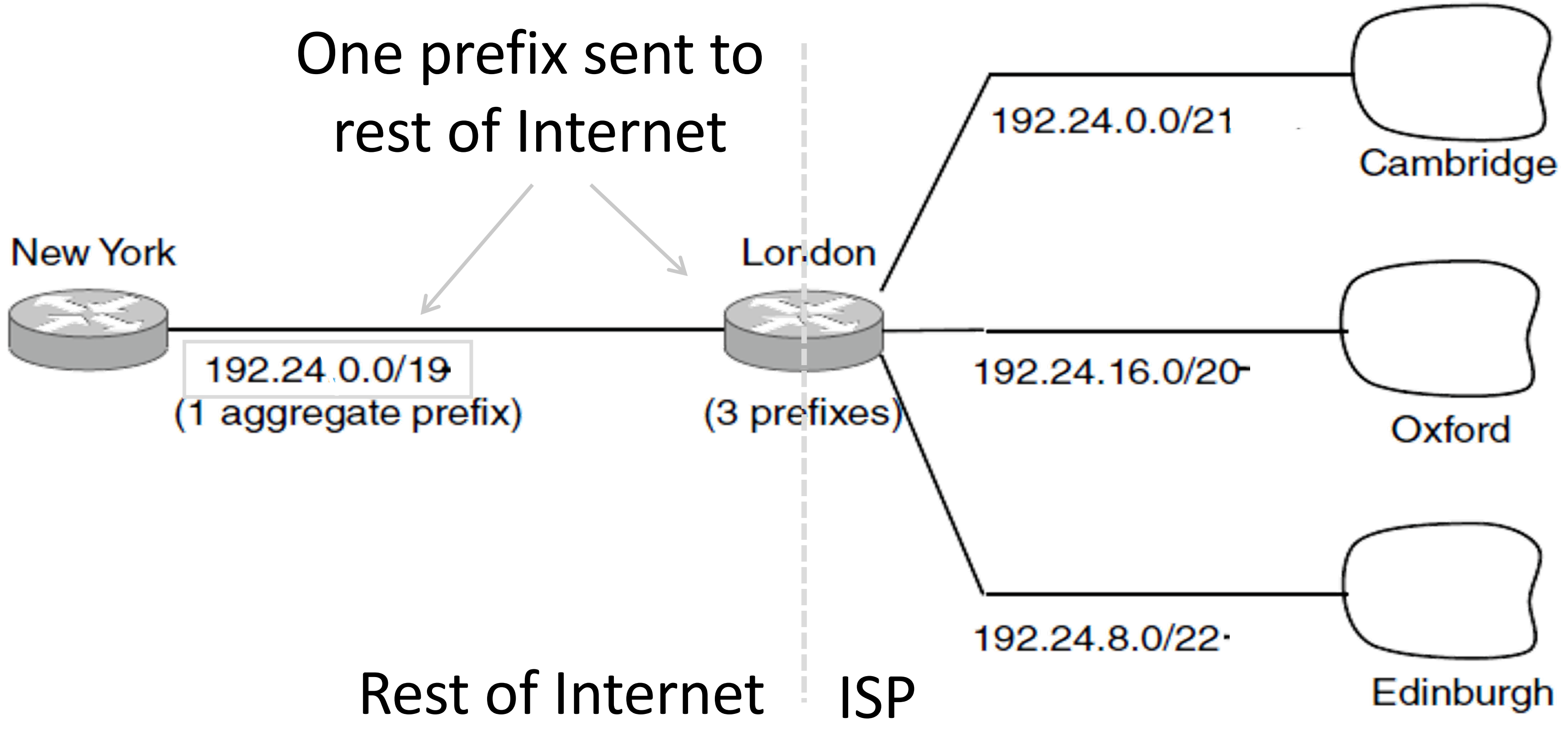
Subnets

- Internally split up one IP prefix



Aggregation

- Externally join multiple separate IP prefixes

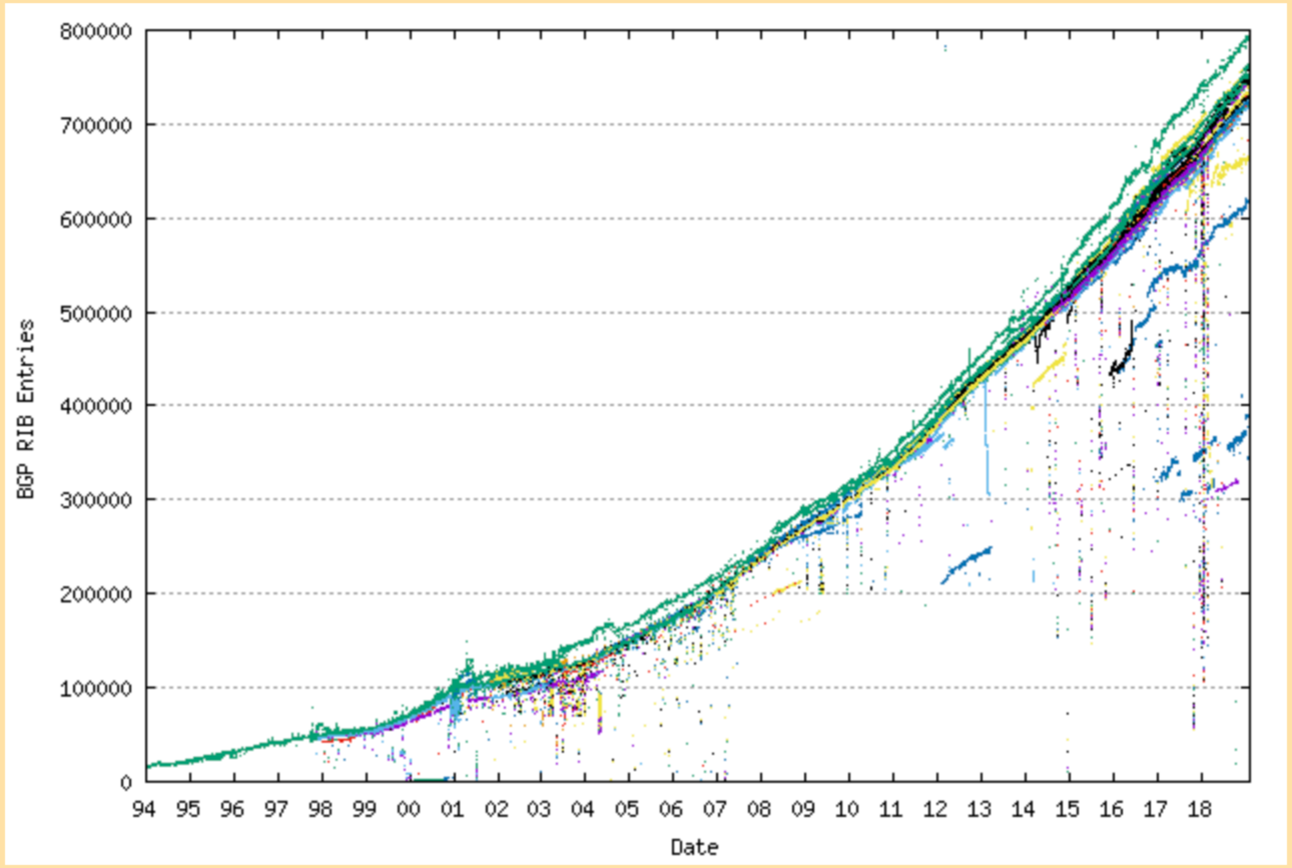


Routing Process

1. Ship these prefixes or regions around to nearby routers
2. Receive multiple prefixes and the paths of how you got them
3. Build a global routing table

Internet Routing Growth

Growth of the BGP Table - 1994 to Present

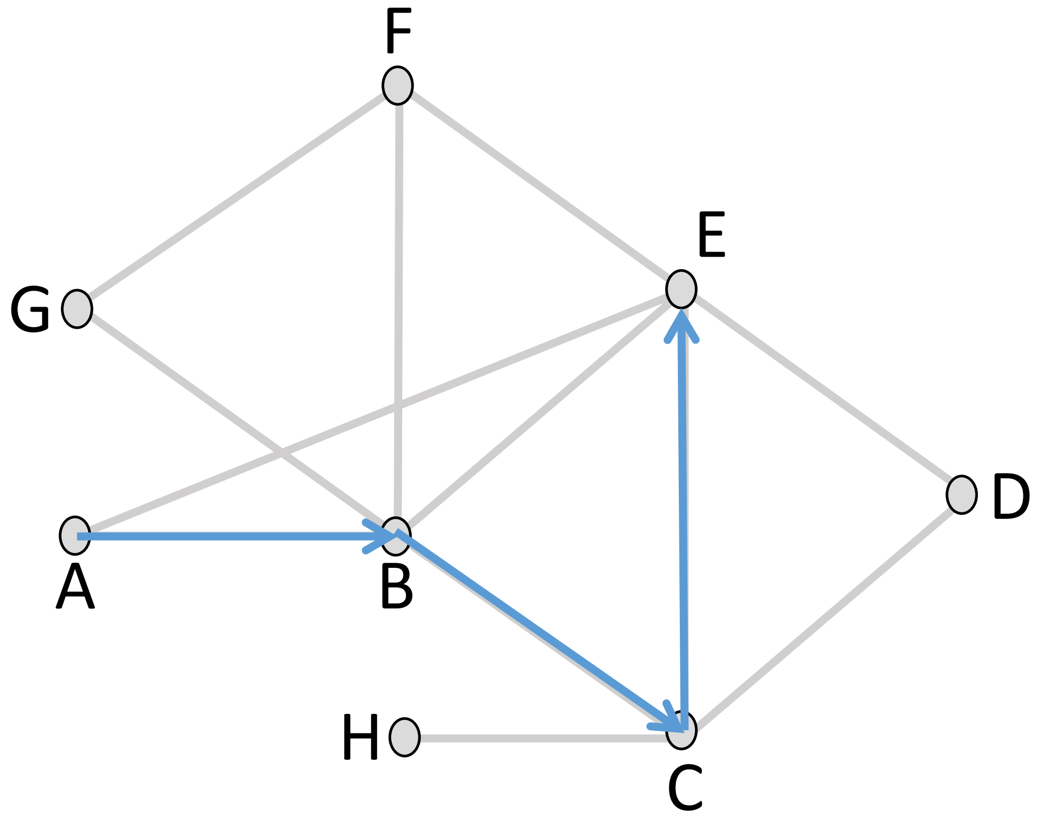


Source: bgp.potaroo.net

Finding “Best” Paths

What are "Best" paths anyhow?

- Many possibilities:
  - Latency, avoid circuitous paths
  - Bandwidth, avoid slow links
  - Money, avoid expensive links
  - Hops, to reduce switching
- But only consider topology
  - Ignore workload, e.g., hotspots



Shortest Paths

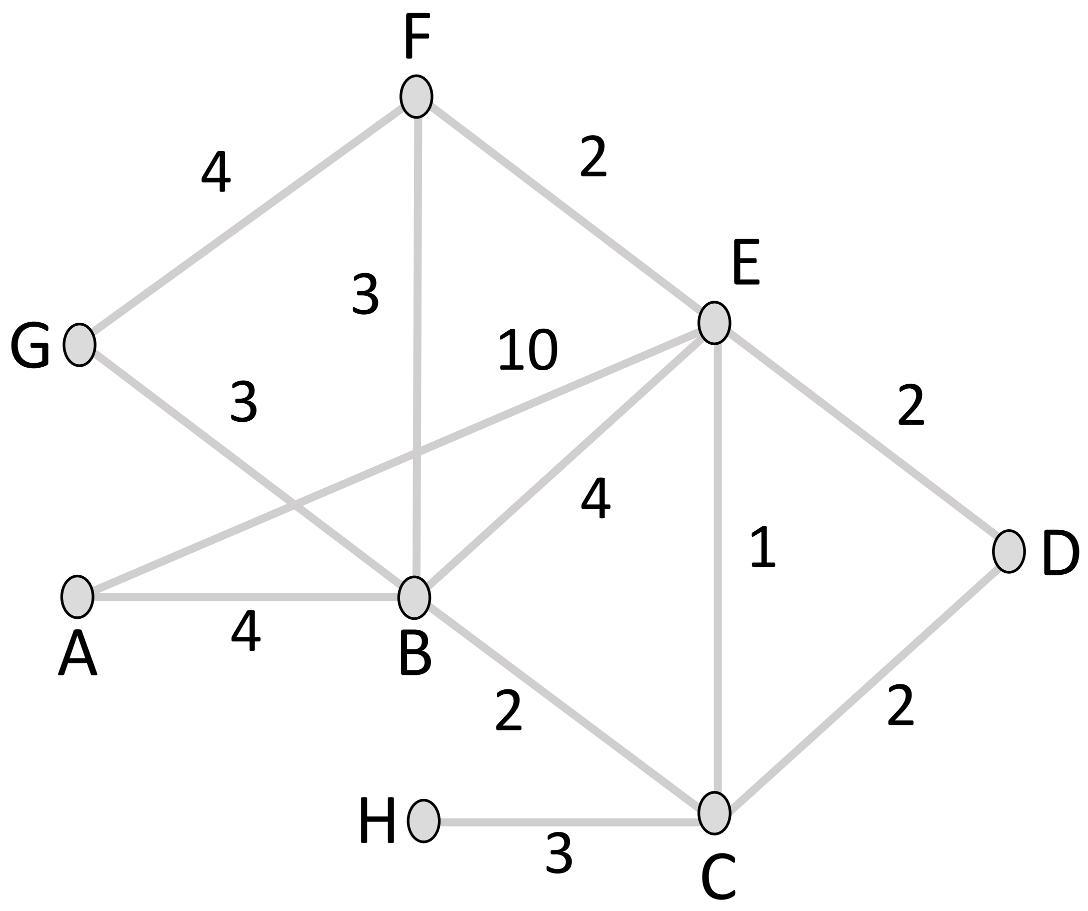
We'll approximate "best" by a cost function that captures the factors

- Often called "least cost" or "shortest"

1. Assign each link a cost (distance)
2. Define best path between each pair of nodes as the path that has the least total cost
3. Pick randomly to any break ties

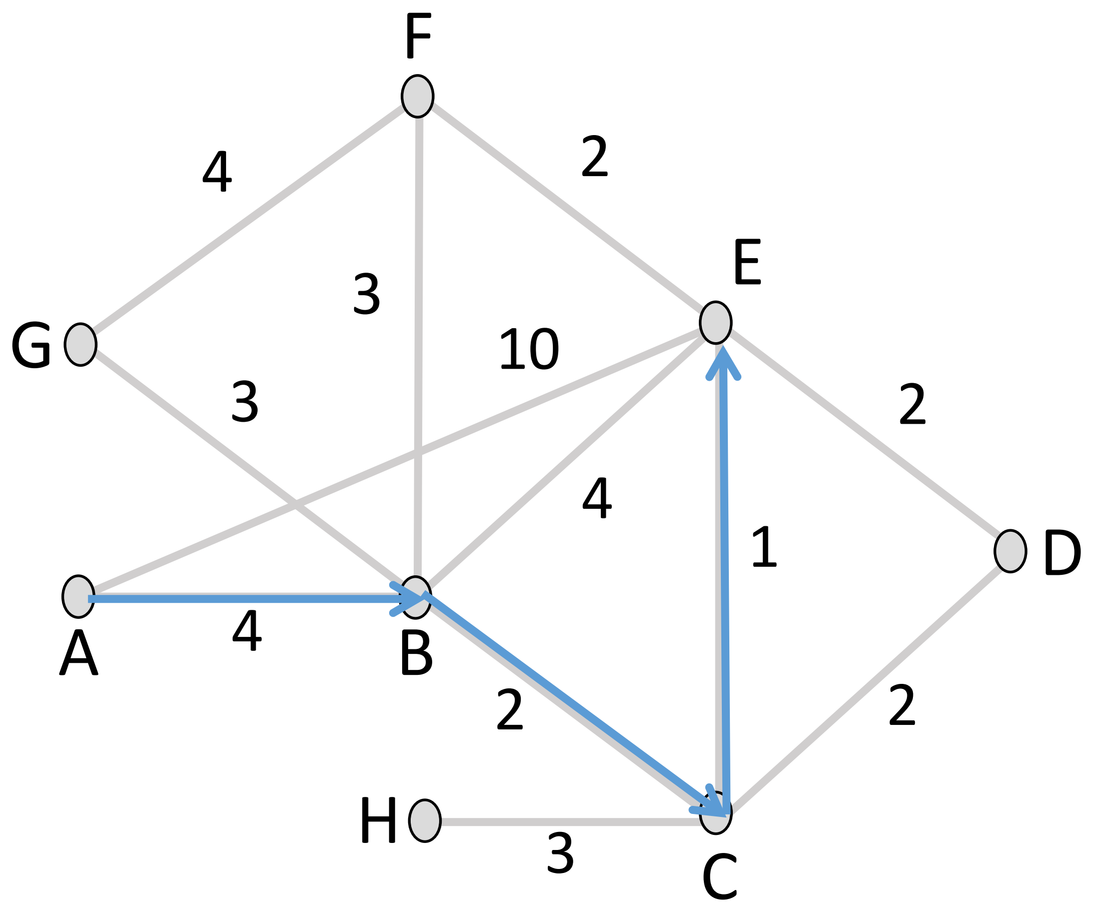
Shortest Paths (2)

- Find the shortest path A → E
- All links are bidirectional, with equal costs in each direction
  - Can extend model to unequal costs if needed



Shortest Paths (3)

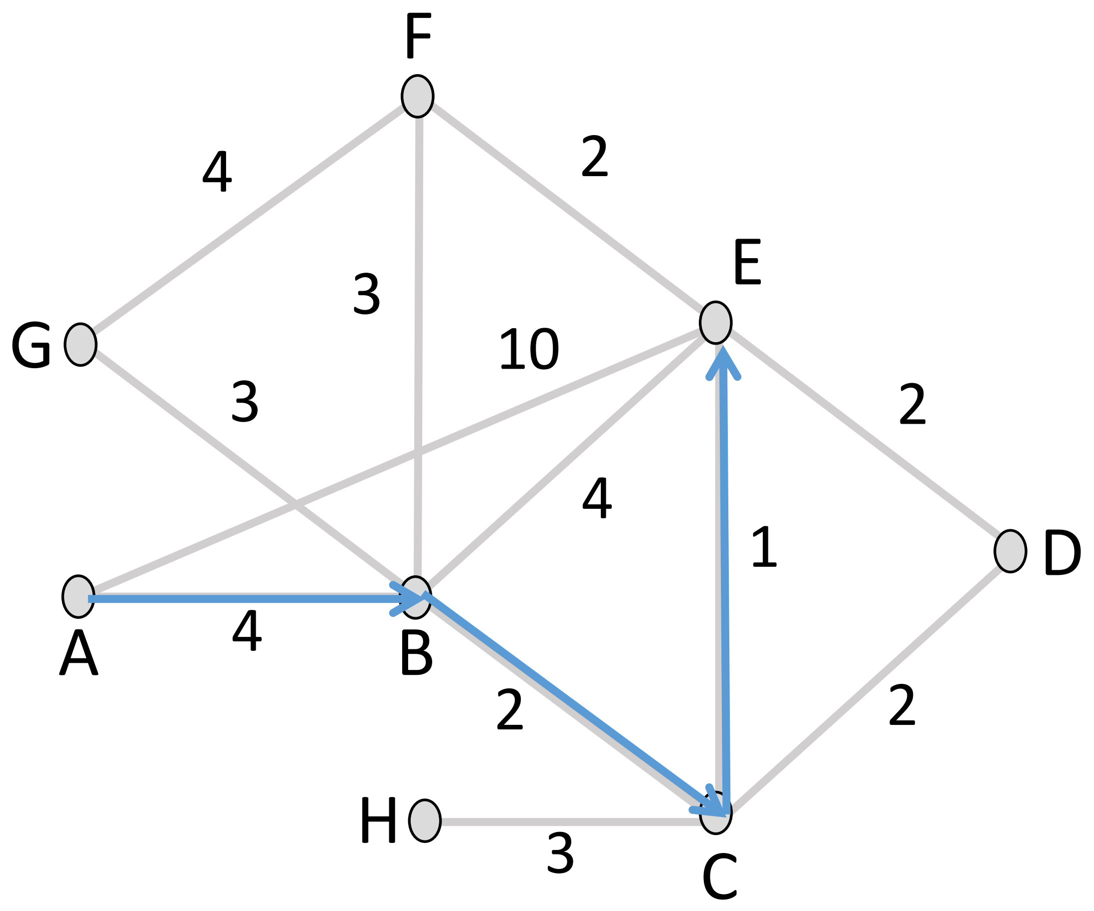
- ABCE is a shortest path
  - cost(ABCE) = 4 + 2 + 1 = 7
- It is shorter than:
  - cost(ABE) = 8
  - cost(ABFE) = 9
  - cost(AE) = 10
  - cost(ABCDE) = 10



Shortest Paths (4)

- Optimality property:
  - Subpaths of shortest paths are also shortest paths
- ABCE is a shortest path

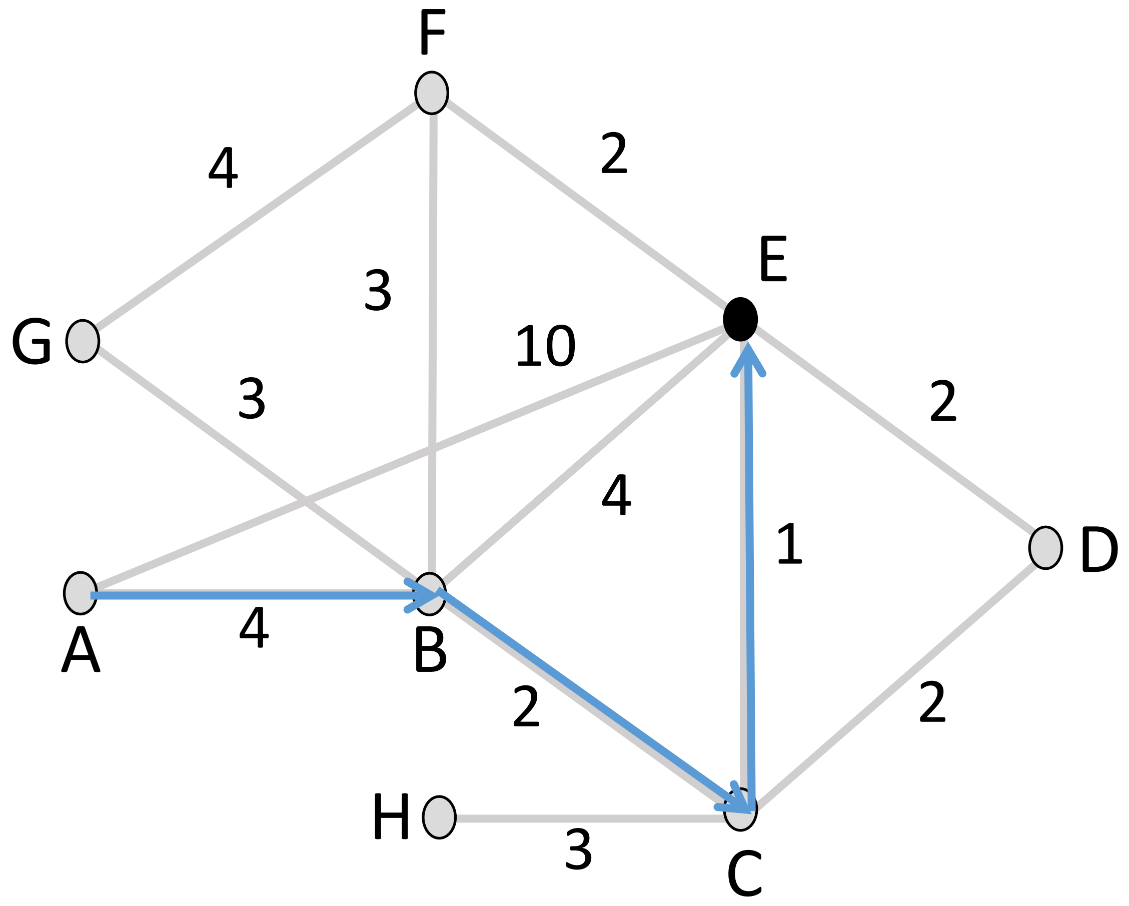
  →So are ABC, AB, BCE, BC, CE



Sink Trees

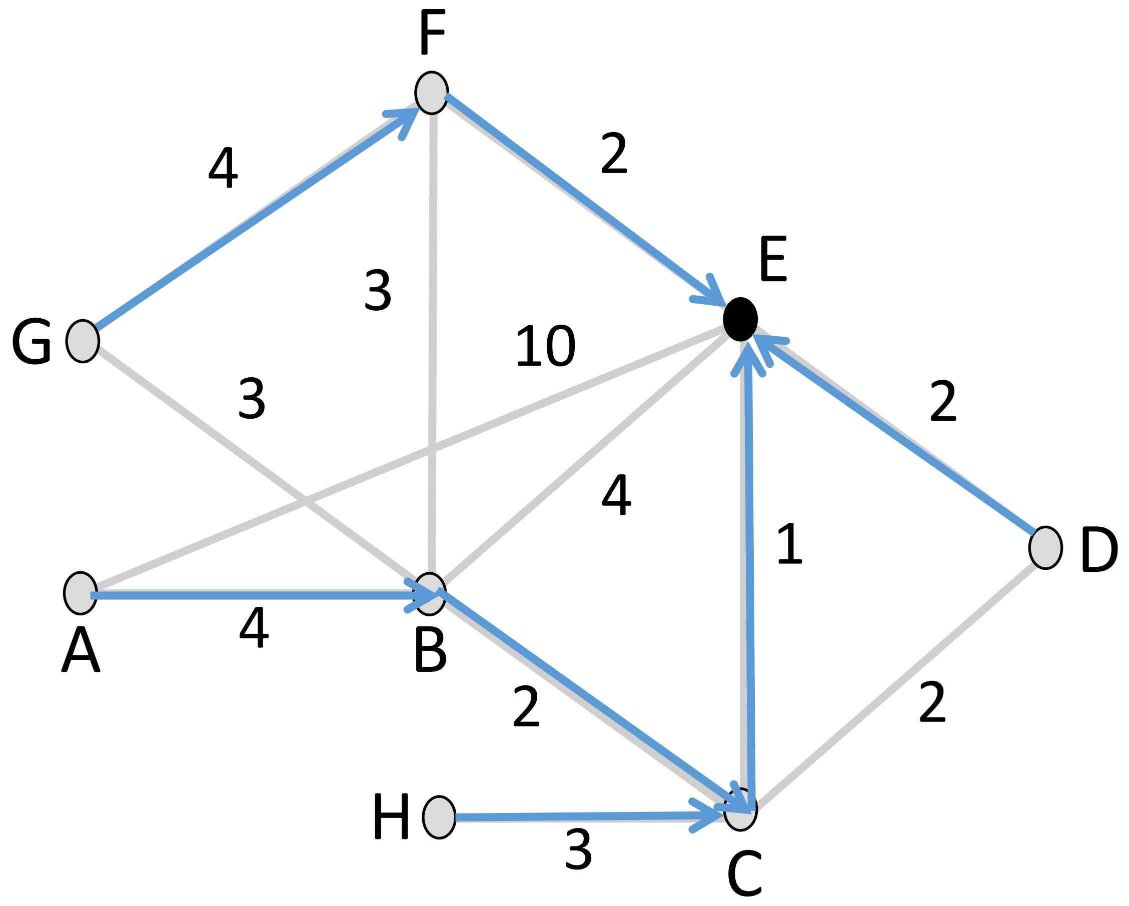
- Sink tree for a destination is the union of all shortest paths towards the destination
  - Similarly source tree

- Find the sink tree for E



Sink Trees (2)

- Implications:
  - Only need to use destination to follow shortest paths
  - Each node only need to send to the next hop
- Forwarding table at a node
  - Lists next hop for each destination
  - Routing table may know more



F
4
2
E
3
G
10
3
2
4
1
D
A
4
B
2
2
H
3
C