The Network Layer – Routing

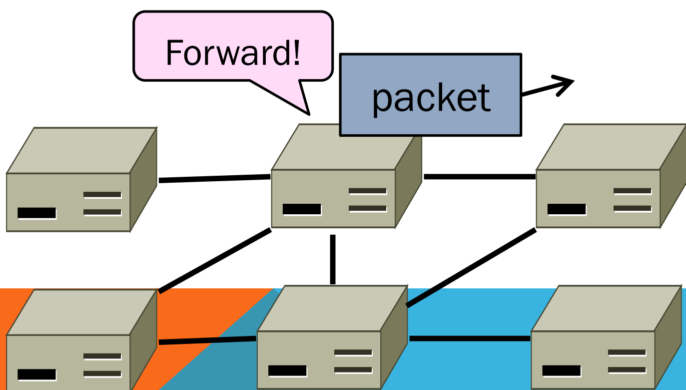
COMP 30650: Networks and Internet Systems

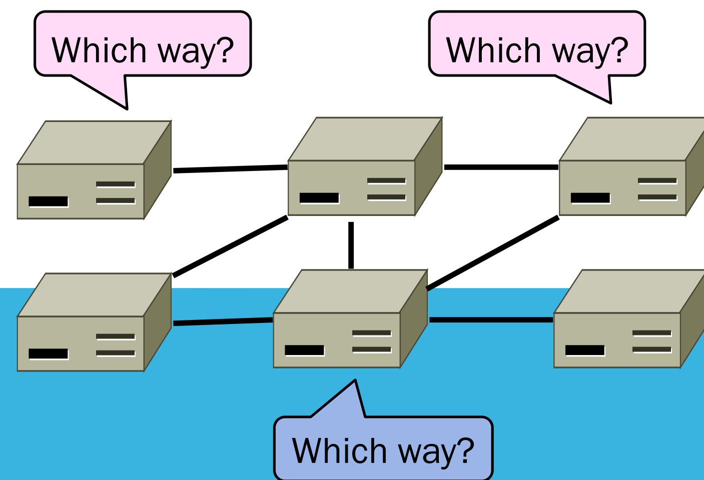
Dr. Gavin McArdle
Email: gavin.mcardle@ucd.ie
Office: A1.09 Computer Science

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





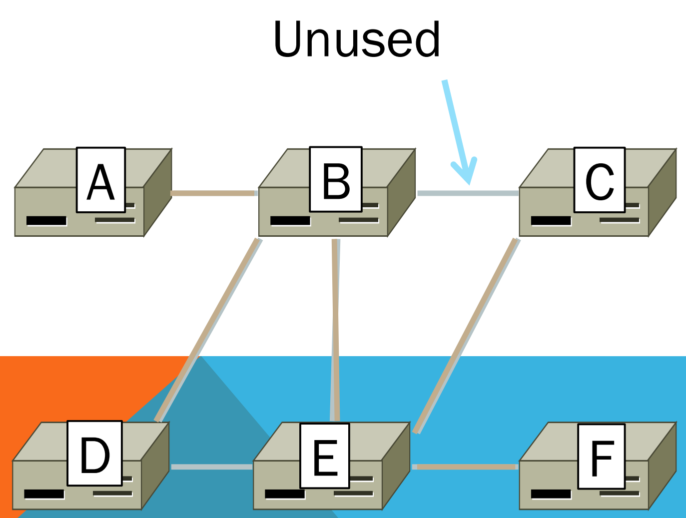
IMPROVING ON THE SPANNING TREE

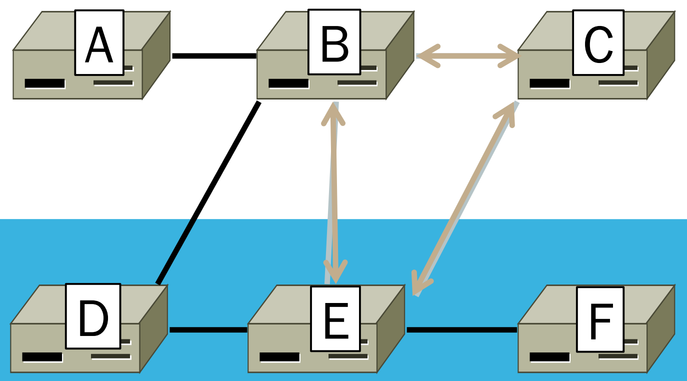
Spanning tree provides basic connectivity

- e.g., some path B→C

Routing uses all links to find "best" paths

- e.g., use BC, BE, and CE





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 ROUTING ALGORITHMS

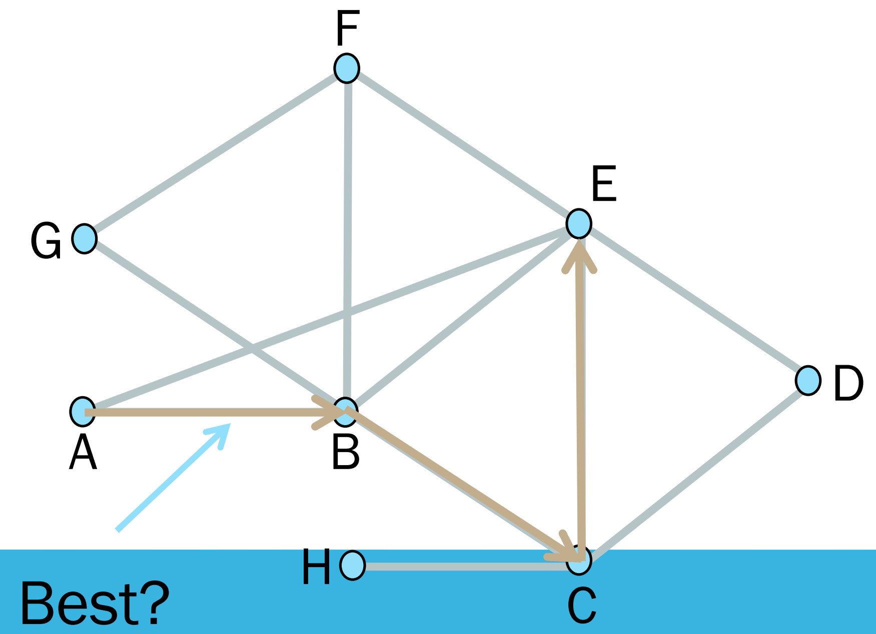
Decentralized, distributed setting

- All nodes are alike; no controller
- Nodes only know what they learn by exchanging messages with neighbours
- Nodes operate concurrently
- May be node/link/message failures

SHORTEST PATH

Defining "best" paths with link costs

- These are <u>shortest path</u> routes



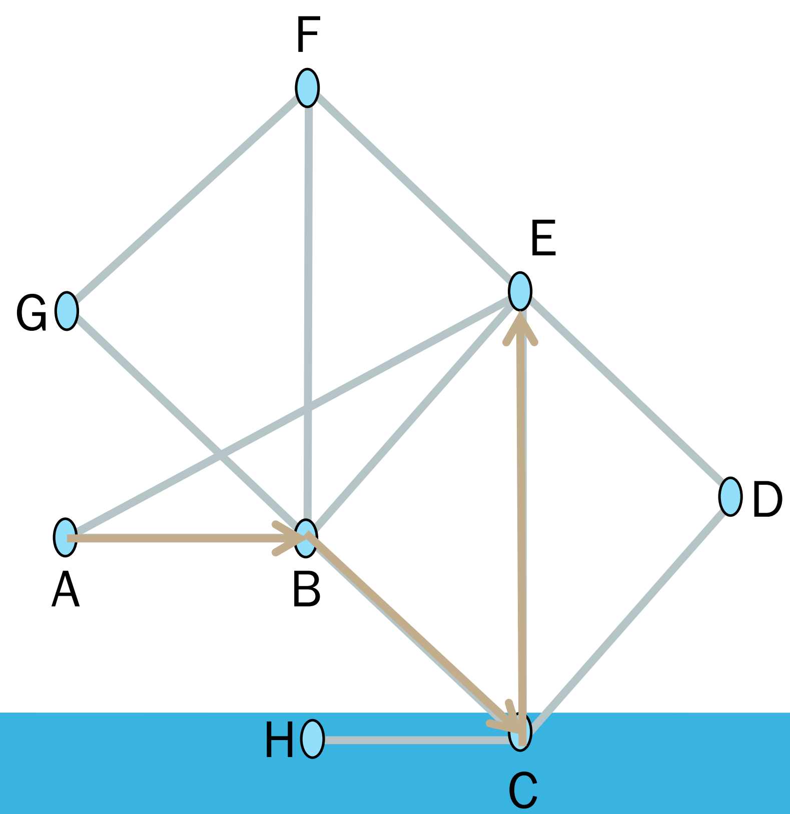
BEST PATHS

Multiple 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 call lowest "shortest"

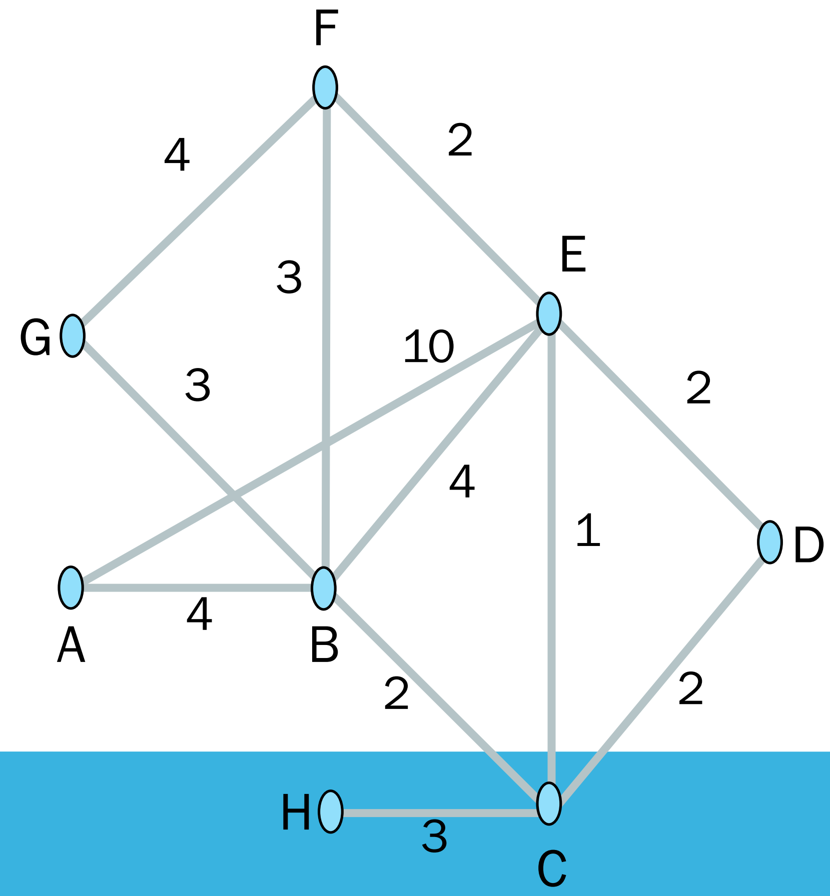
1. Assign each link a cost (distance)
2. Define best path between each pair of nodes as the path that has the lowest total cost (or is shortest)
3. Pick randomly to any break ties

SHORTEST PATHS

Find the shortest path A → E

All links are bidirectional, with equal costs in each direction

- Can extend model to unequal costs if needed



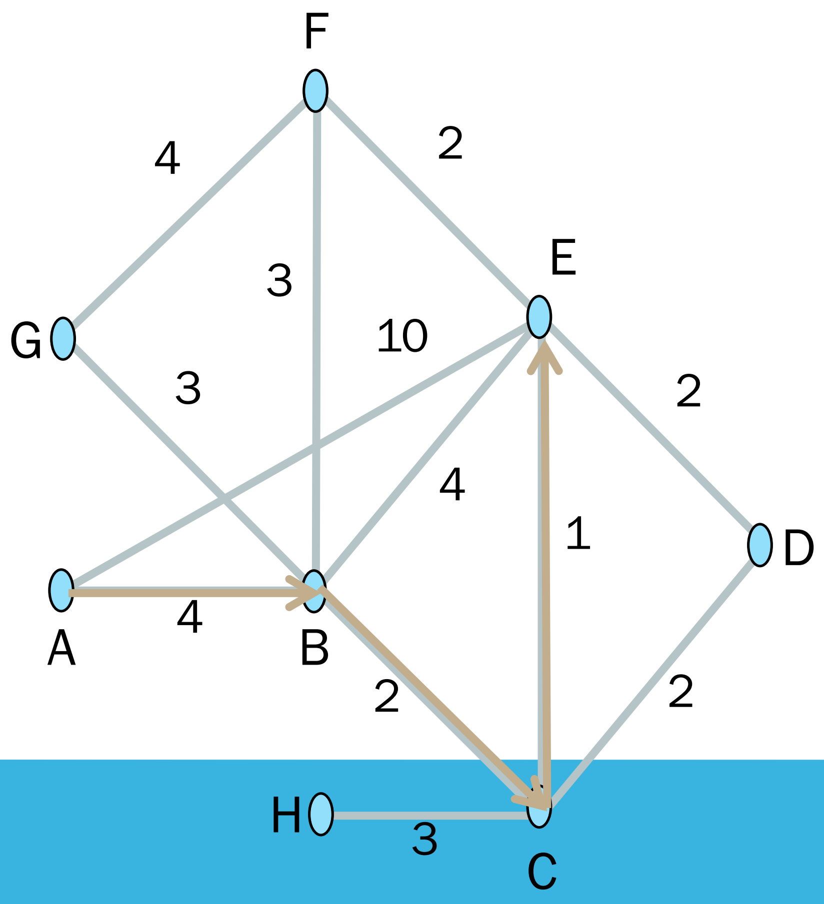
SHORTEST PATHS

ABCE is a shortest path

dist(ABCE) = 4 + 2 + 1 = 7

This is less than:

- dist(ABE) = 8
- dist(ABFE) = 9
- dist(AE) = 10
- dist(ABCDE) = 10



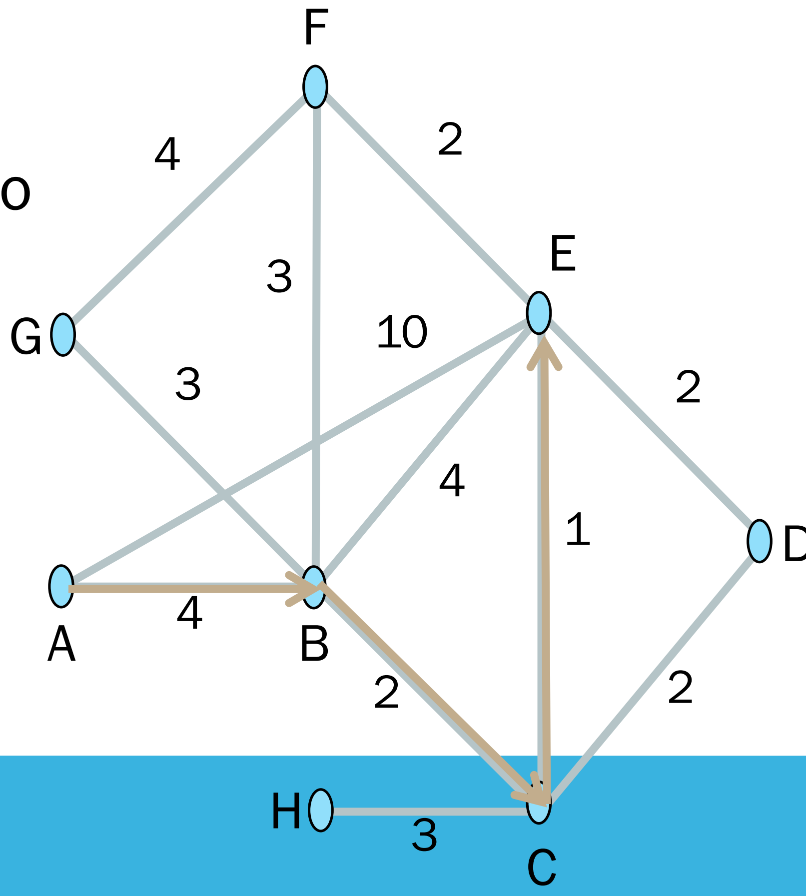
SHORTEST PATHS

Optimality property:

- Subpaths of shortest paths are also shortest paths

ABCE is a shortest path

→So are ABC, AB, BCE, BC, CE

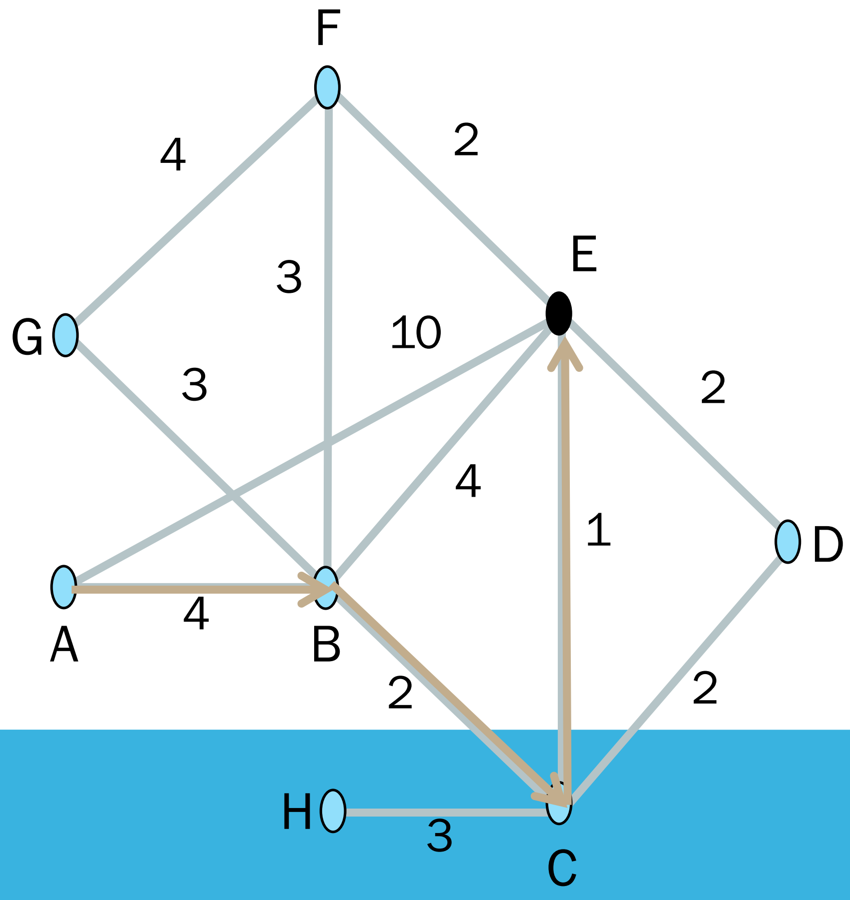


SINK TREES

<u>Sink tree</u> for a destination is the union of all shortest paths towards the destination

<u>Source tree</u> for an origin is the union of all shortest paths from a source

Find the sink tree for E



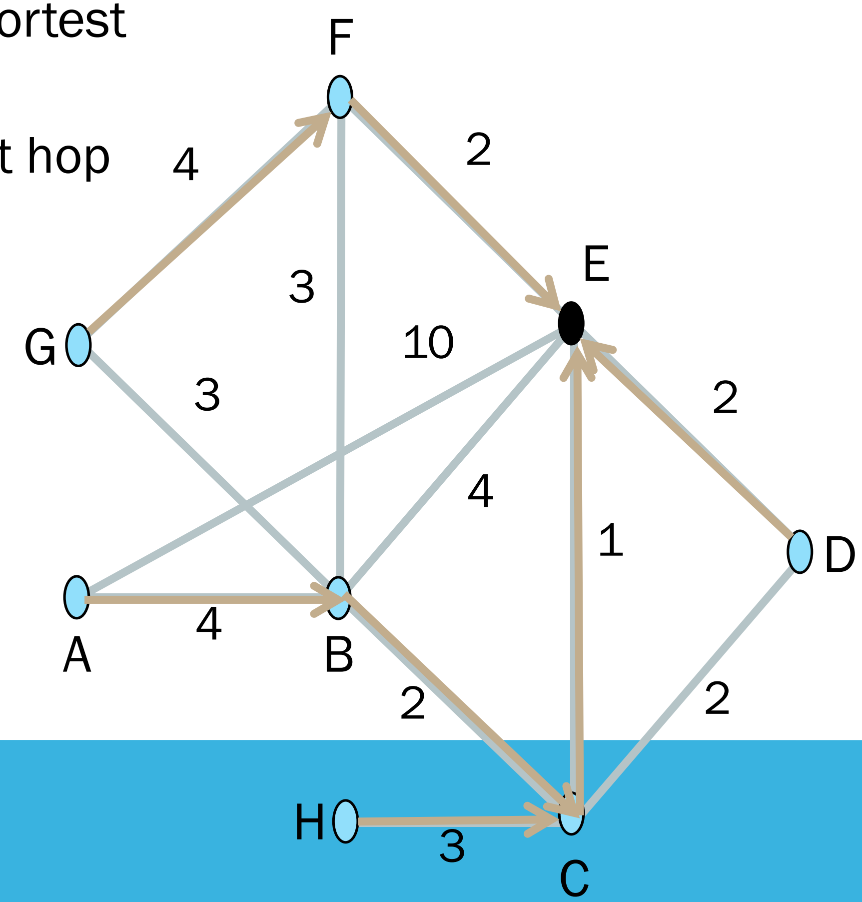
SINK TREES

Implications:

- Only need to use destination to follow shortest paths
- Each node only needs to send to the next hop

Forwarding table at a node

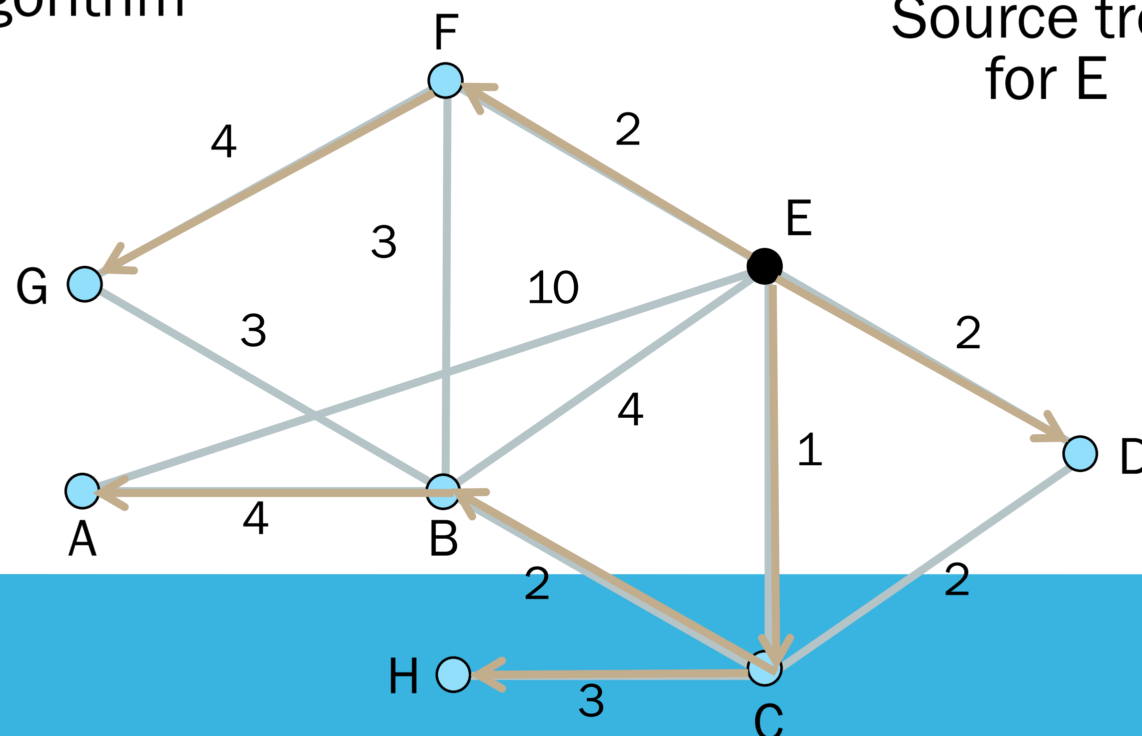
- Lists next hop for each destination
- Routing table may know more



DIJKSTRA'S ALGORITHM

How to compute shortest paths <u>given the network topology</u>

- With Dijkstra's algorithm



Source tree for E

DIJKSTRA'S ALGORITHM

Algorithm:

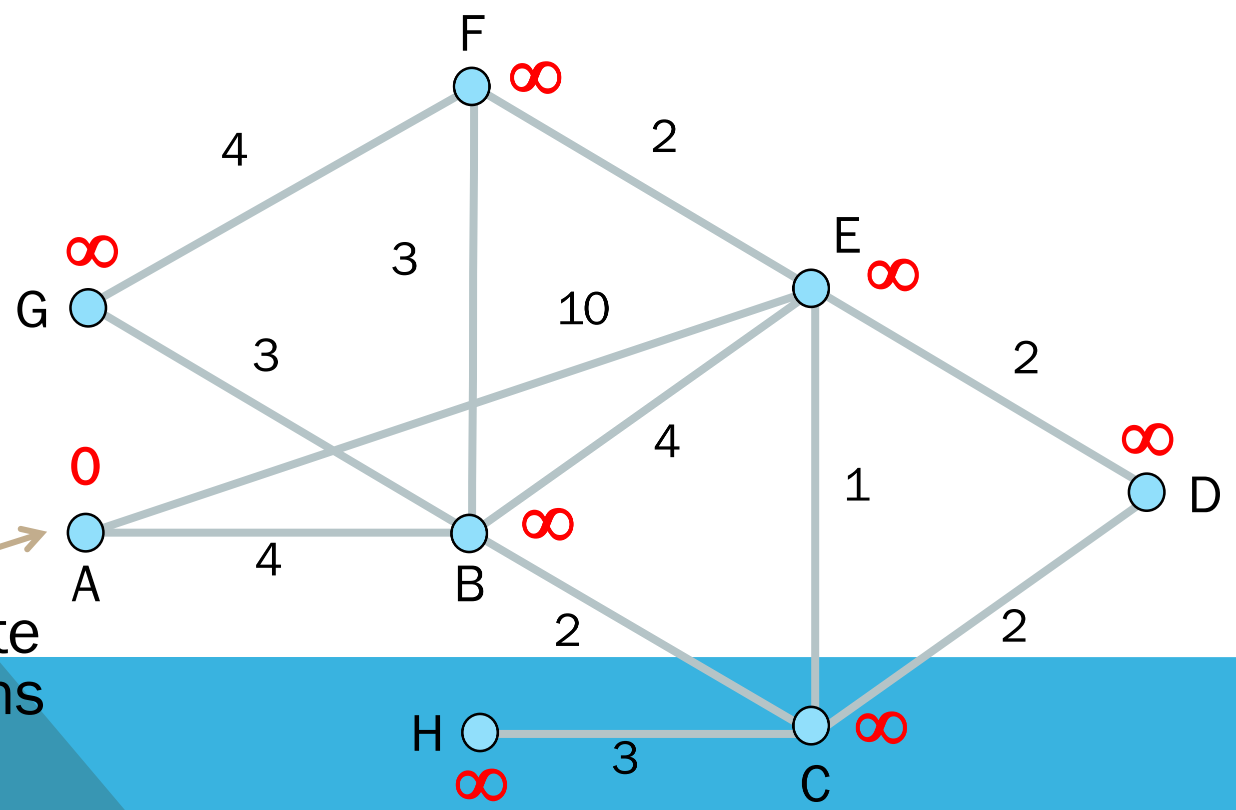
Mark all nodes tentative, set distances from source to 0 (zero) and ∞ (infinity) for all other nodes

While tentative nodes remain:

- Extract N, a node with lowest distance
- Add link to N to the shortest path tree
- Relax the distances of neighbours of N by lowering any better distance estimates

DIJKSTRA'S ALGORITHM (2)

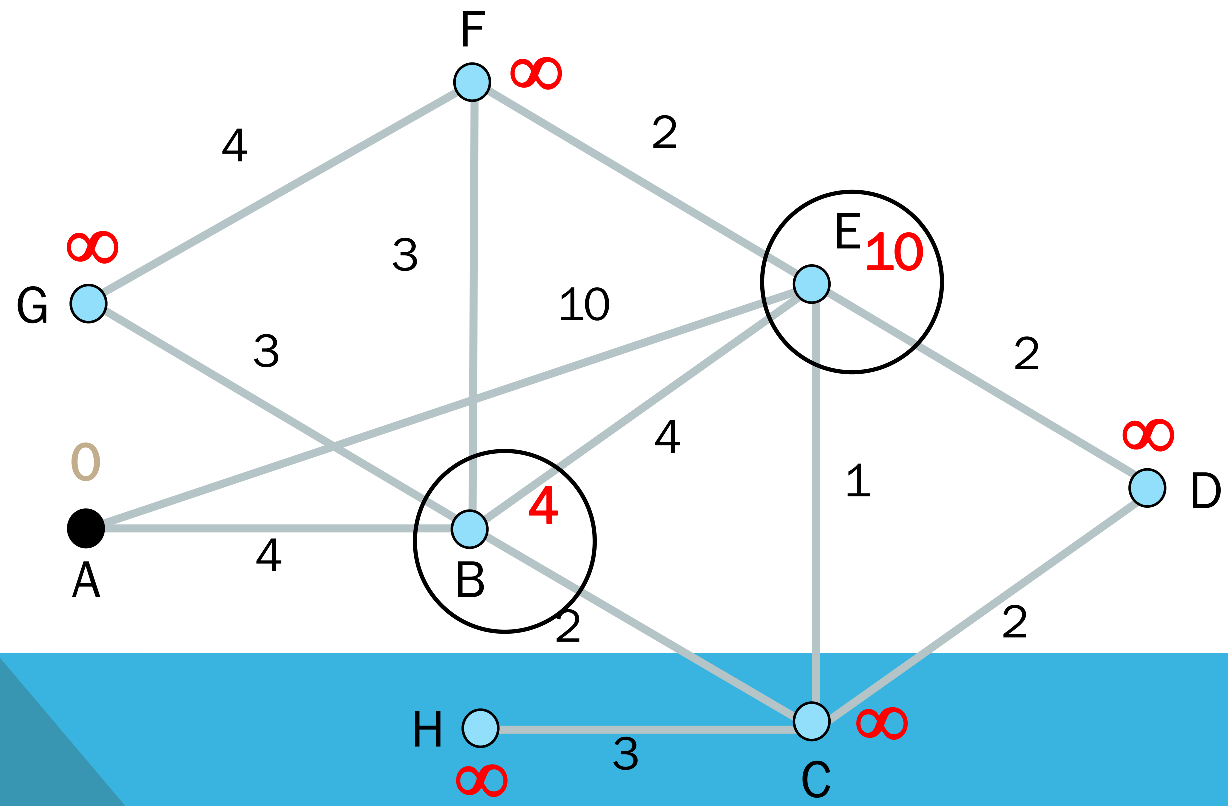
Initialization



We'll compute shortest paths from A

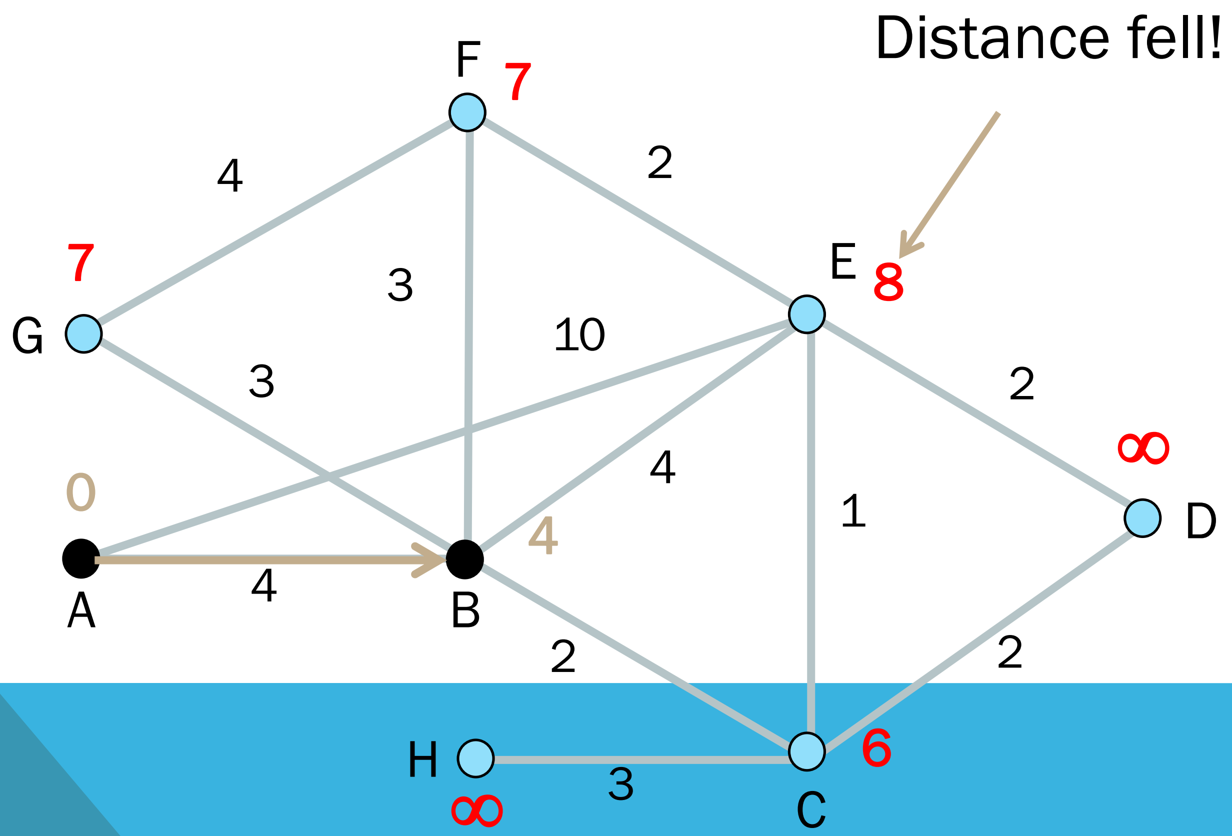
DIJKSTRA'S ALGORITHM (3)

Relax around A



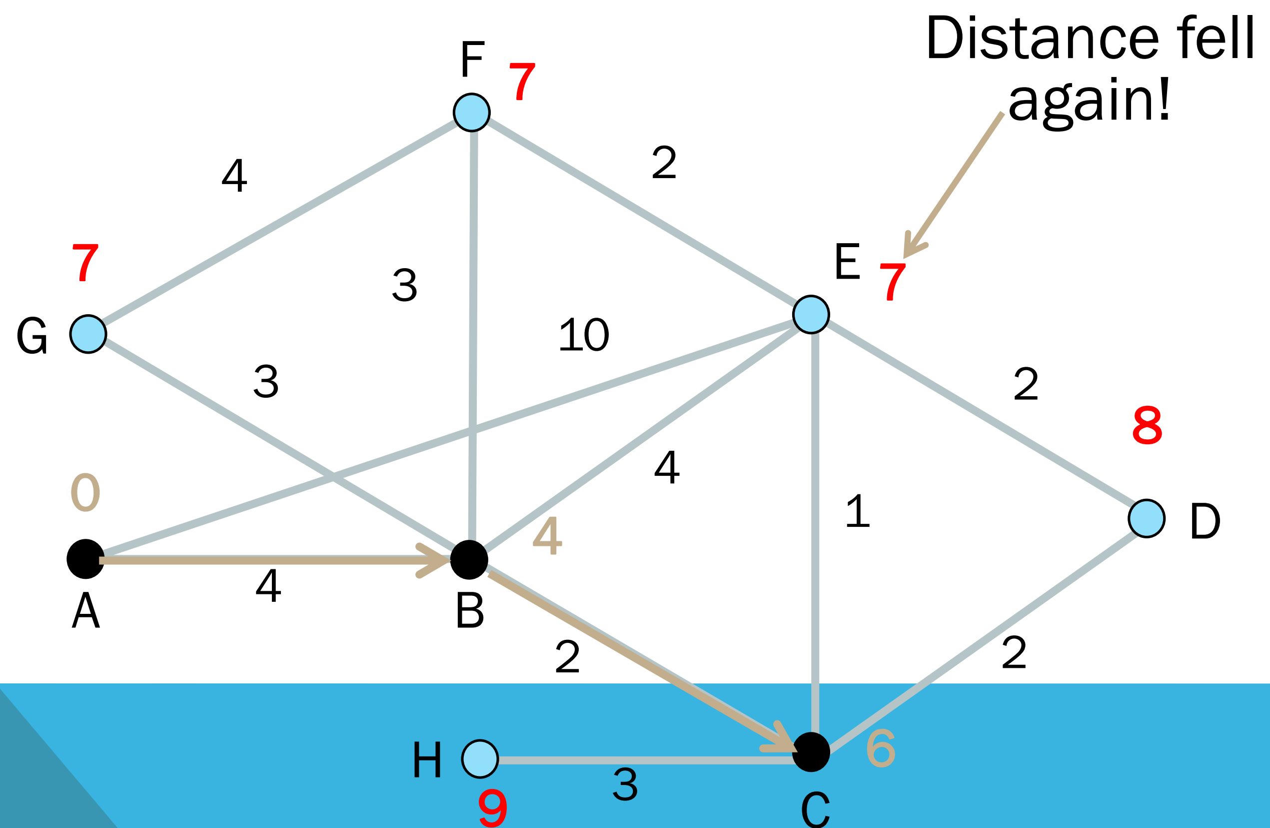
DIJKSTRA'S ALGORITHM (4)

Relax around B



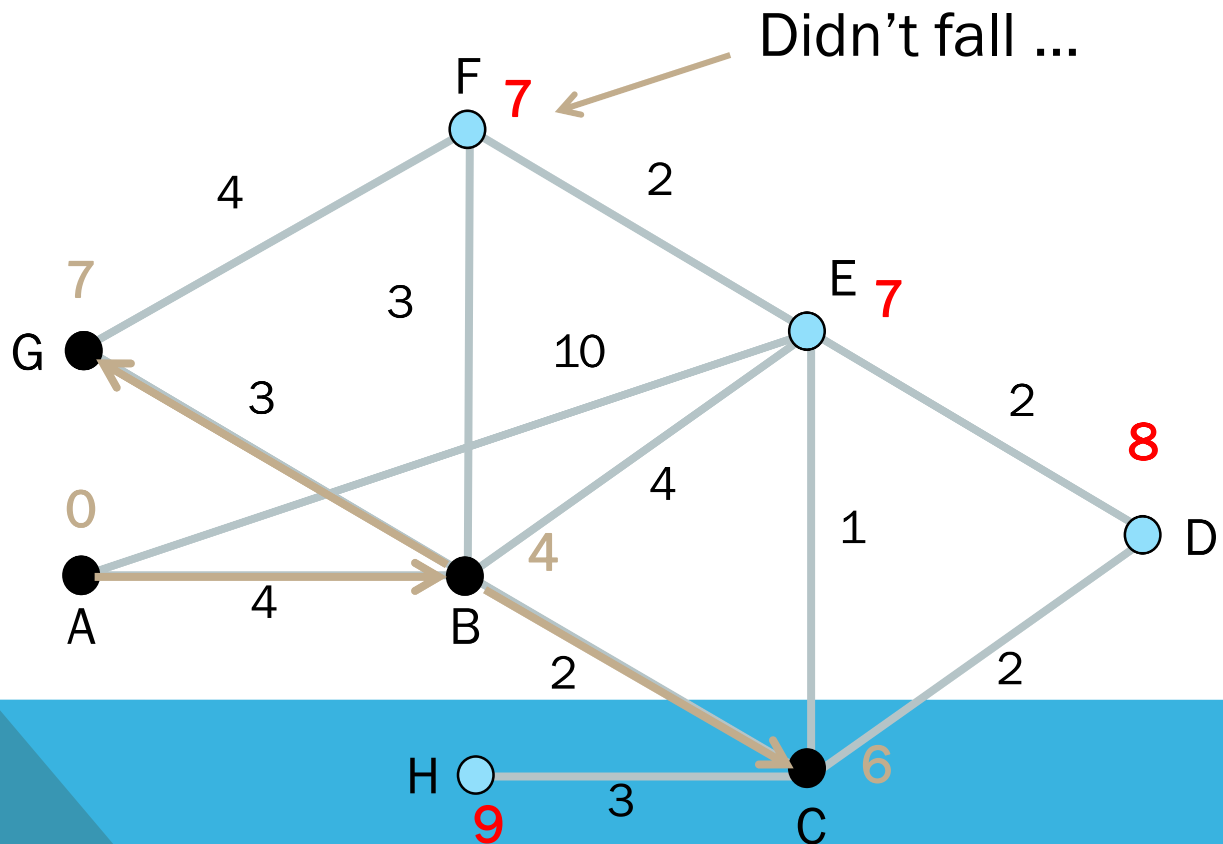
DIJKSTRA'S ALGORITHM (5)

Relax around C



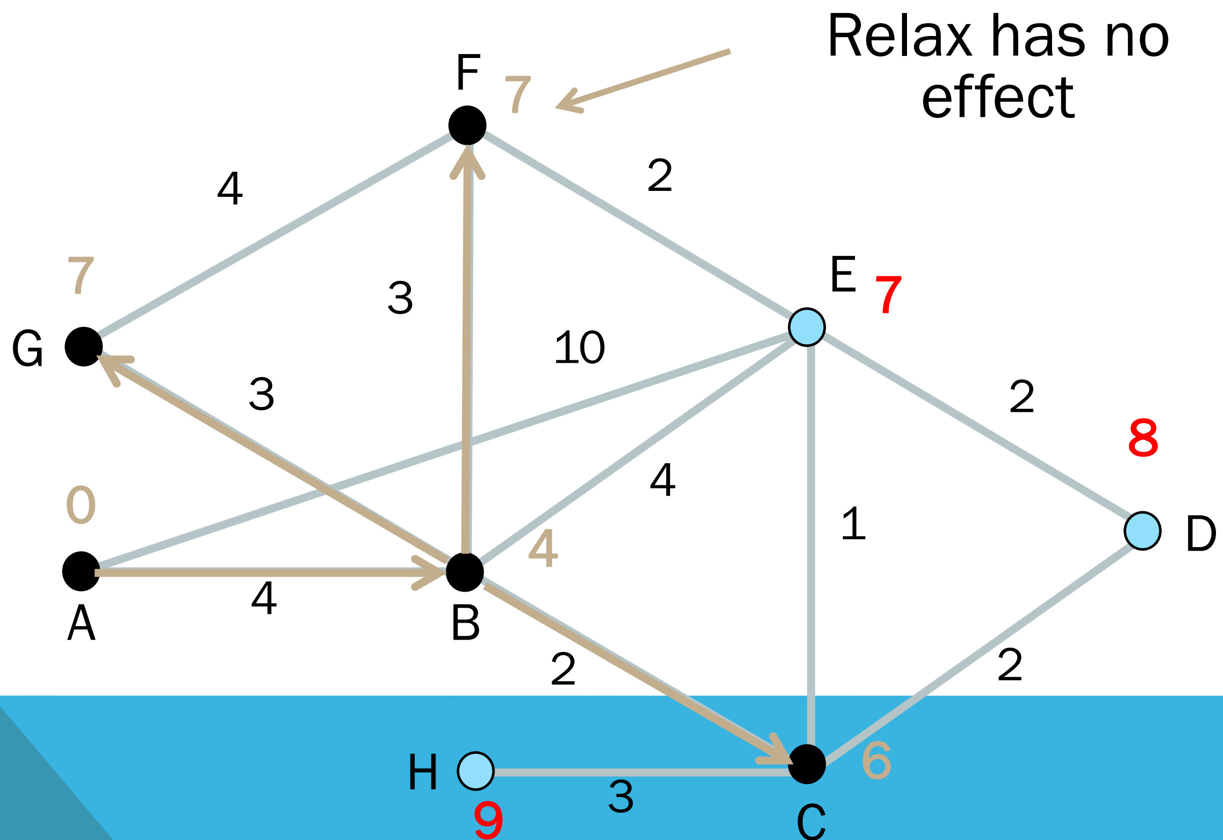
DIJKSTRA'S ALGORITHM (6)

Relax around G (say)



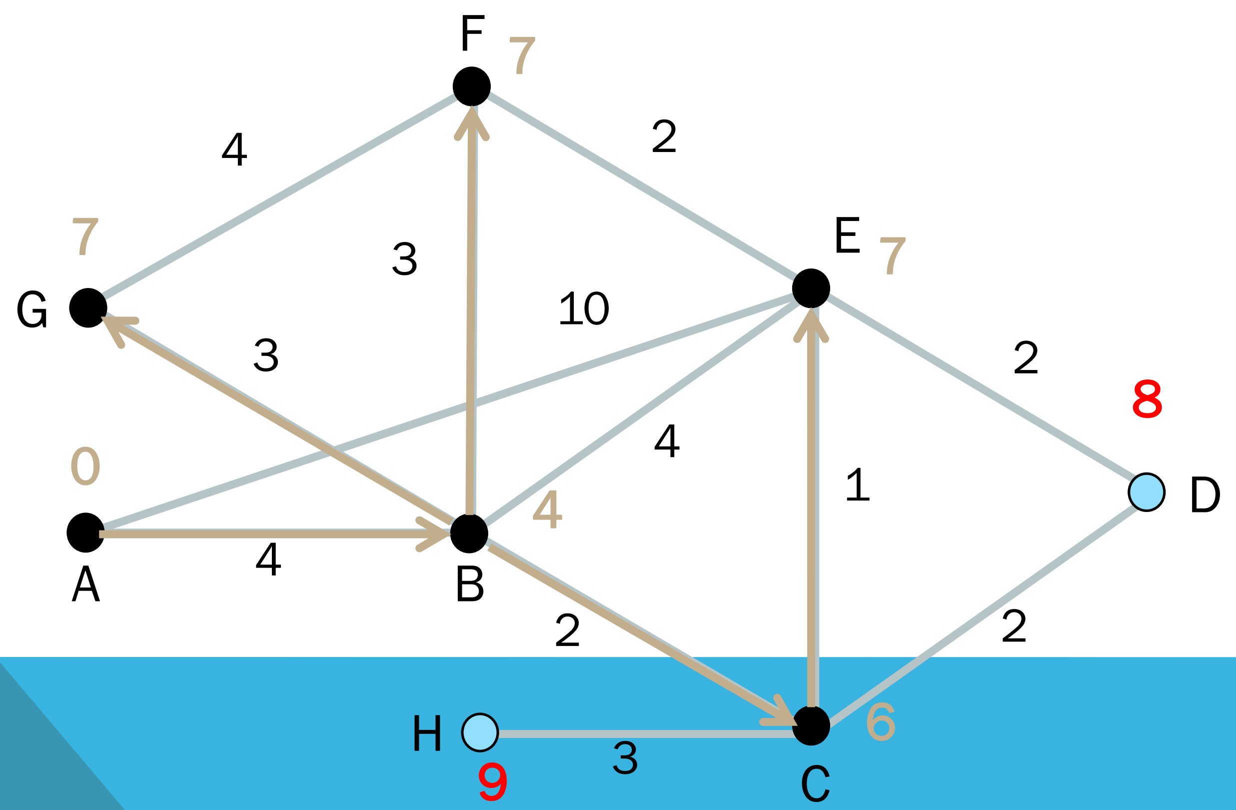
DIJKSTRA'S ALGORITHM (7)

Relax around F (say)



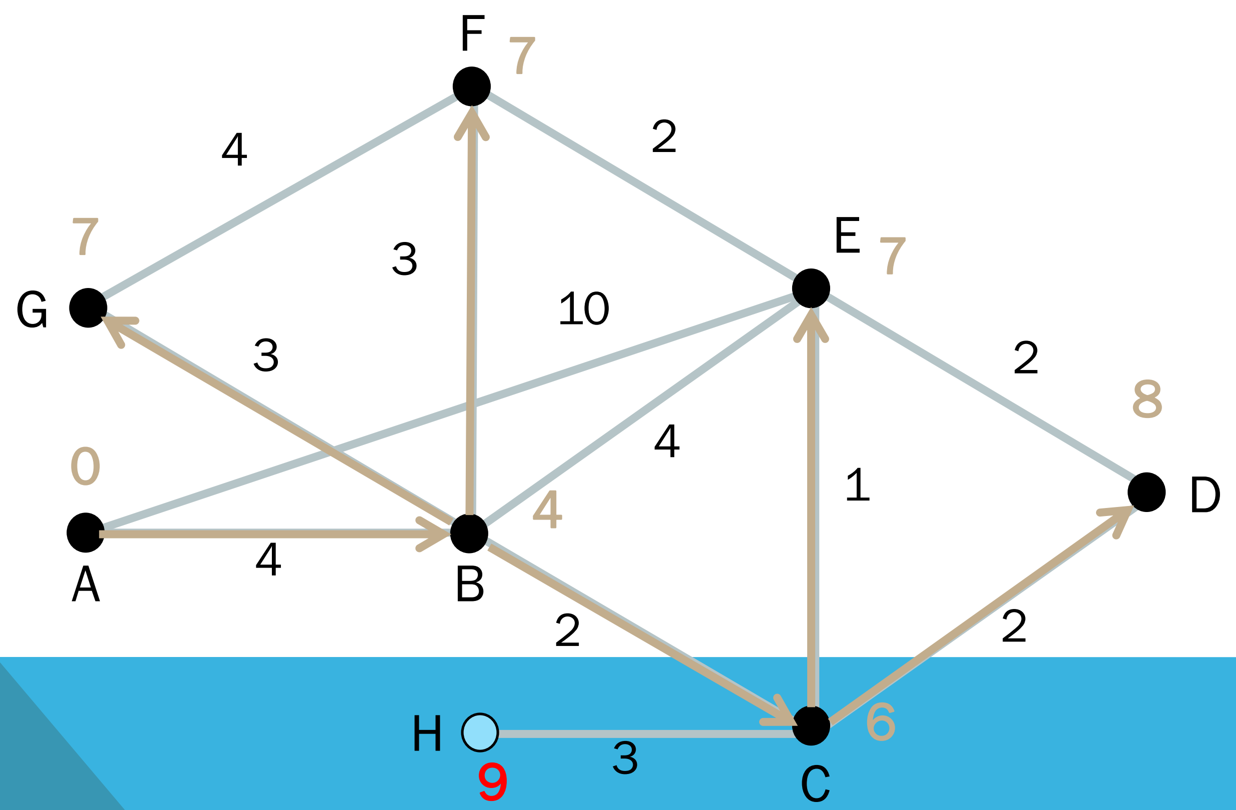
DIJKSTRA'S ALGORITHM (8)

Relax around E



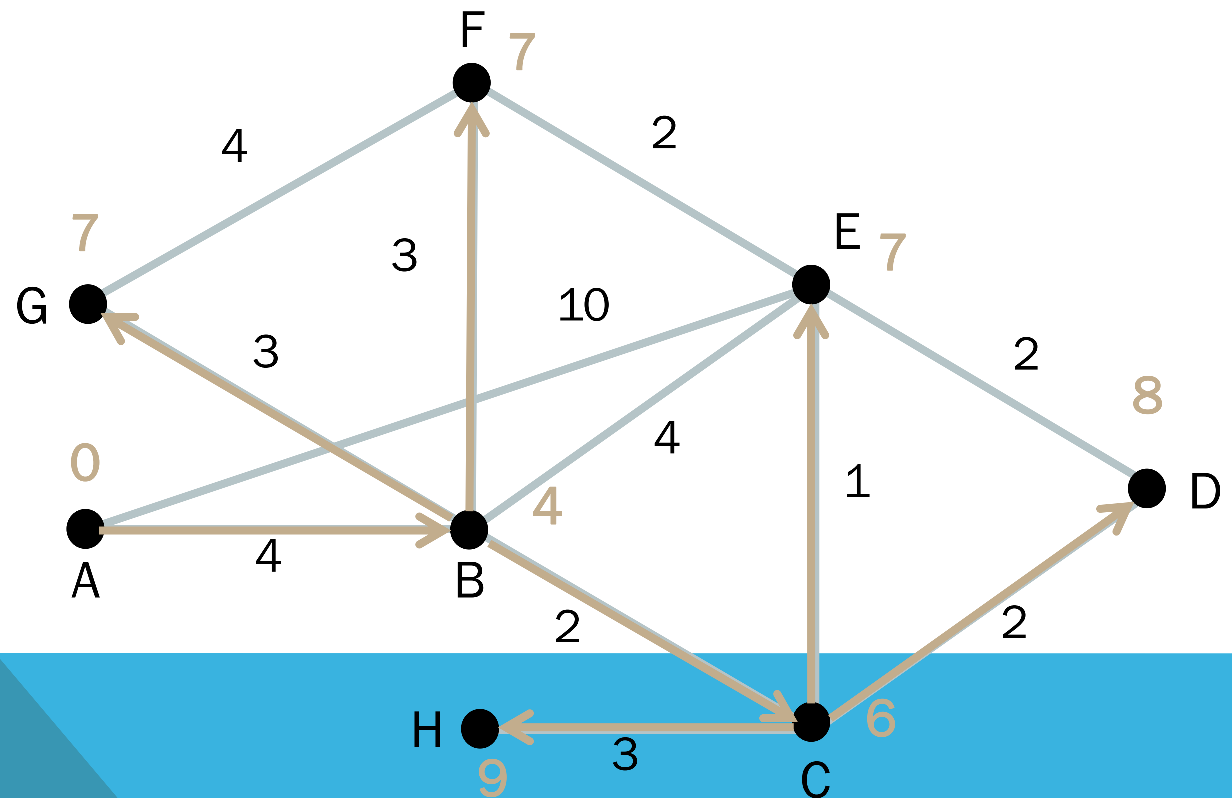
DIJKSTRA'S ALGORITHM (9)

Relax around D



DIJKSTRA'S ALGORITHM (10)

Finally, H … done



DIJKSTRA COMMENTS

Finds shortest paths in order of increasing distance from source

- Leverages optimality property

Runtime depends on efficiency of extracting min-cost node

- Superlinear in network size (grows fast)

Gives complete source/sink tree

- More than needed for forwarding!
- **But requires complete topology**

LINK-STATE ROUTING

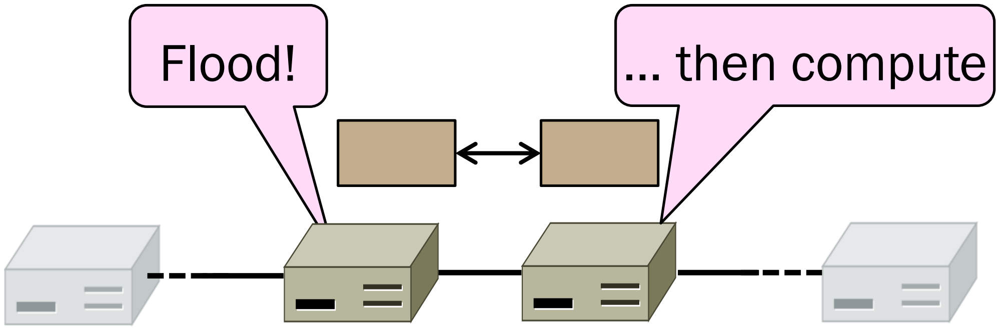
Widely used in practice

- Used in Internet/ARPANET from 1979
- Modern networks use OSPF and IS-IS
  - Open Shortest Path First (**OSPF**) is a routing protocol for Internet Protocol (IP) networks. It uses a link state routing
  - Intermediate System to Intermediate System (IS-IS) is neutral regarding the type of network addresses for which it can route. It uses a link state routing.

LINK STATE

How to compute shortest paths in a distributed network

- The Link-State (LS) approach



LINK-STATE SETTING

Each node computes their own forwarding table in a distributed setting:

1. Node knows only the cost to its neighbours; not the topology
2. Node can talk only to its neighbours using messages
3. Nodes run the same algorithm concurrently
4. Nodes/links may fail, messages may be lost

LINK-STATE ALGORITHM

Proceeds in two phases:

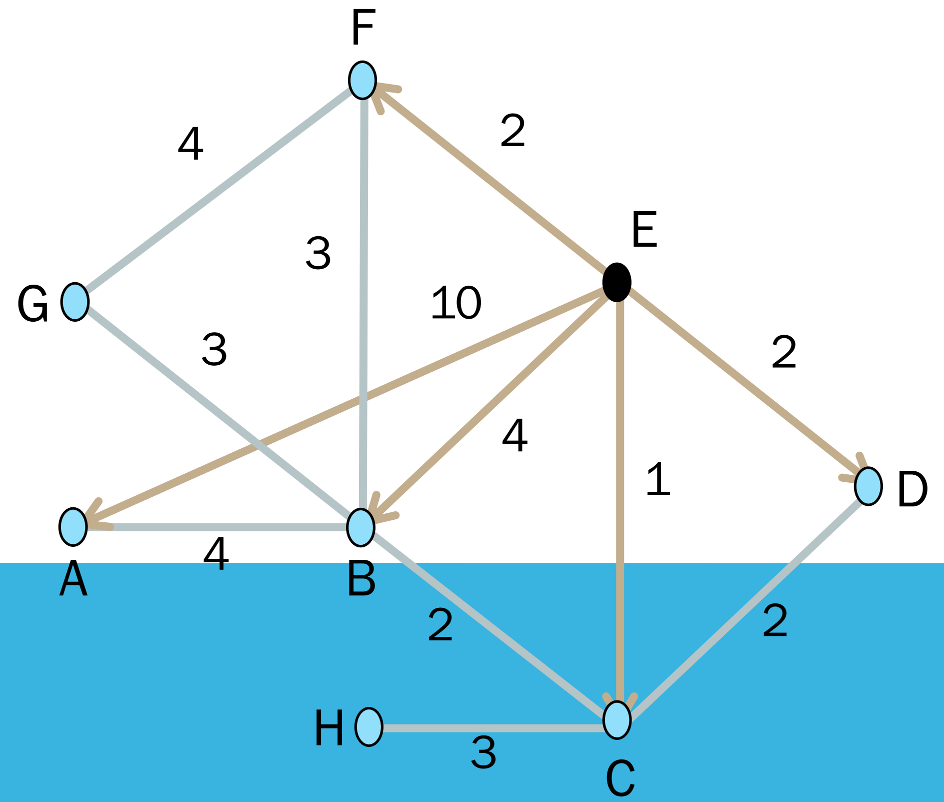
1. Nodes <u>flood</u> topology in the form of link state packets
   - Each node learns full topology
2. Each node computes its own forwarding table
   - By running Dijkstra (or equivalent)

PHASE 1: TOPOLOGY DISSEMINATION

Each node floods <u>link state packet</u> (LSP) that describes their portion of the topology

Node E's LSP flooded to A, B, C, D, and F

Seq. #	
A	10
B	4
C	1
D	2
F	2



PHASE 2: ROUTE COMPUTATION

Each node has full topology

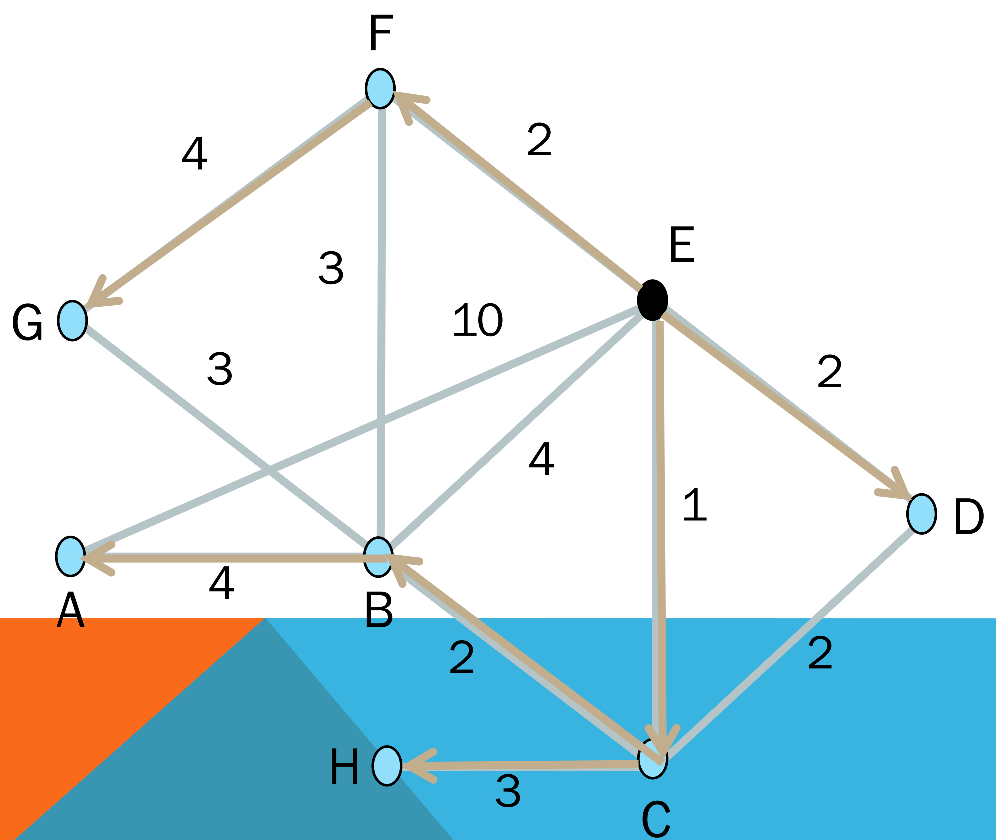
- By combining all LSPs

Each node simply runs Dijkstra

- Some replicated computation, but finds required routes directly
- Compile forwarding table from sink/source tree

FORWARDING TABLE

Source Tree for E (from Dijkstra)



E's Forwarding Table

To	Next
A	C
B	C
C	C
D	D
E	--
F	F
G	F
H	C

HANDLING CHANGES

On change, flood updated LSPs, and re-compute routes

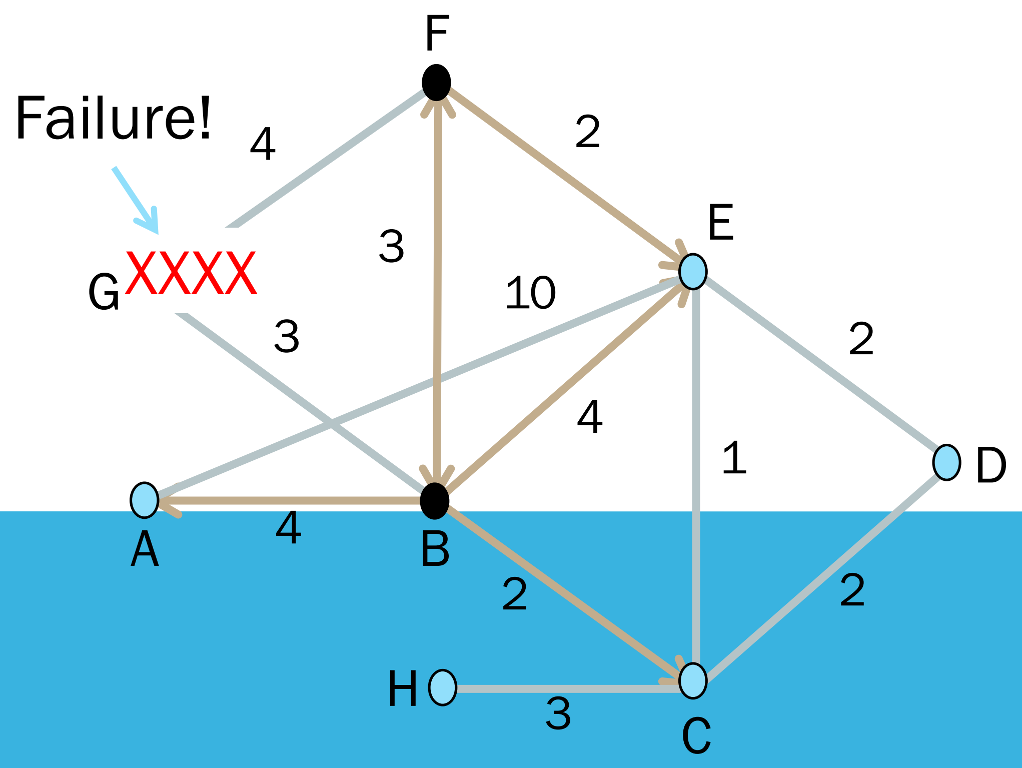
- E.g., nodes adjacent to failed link or node initiate

B's LSP

Seq. #	
A	4
C	2
E	4
F	3
G	∞

F's LSP

Seq. #	
B	3
E	2
G	∞



HANDLING CHANGES

Link failure

- Both nodes notice, send updated LSPs
- Link is removed from topology

Node failure

- All neighbors notice a link has failed
- Failed node can't update its own LSP
- But it is OK: all links to node removed

HANDLING CHANGES

Addition of a link or node

- Add LSP of new node to topology
- Old LSPs are updated with new link

LINK-STATE COMPLICATIONS

Things that can go wrong:

- Seq. number reaches max, or is corrupted
- Node crashes and loses seq. number

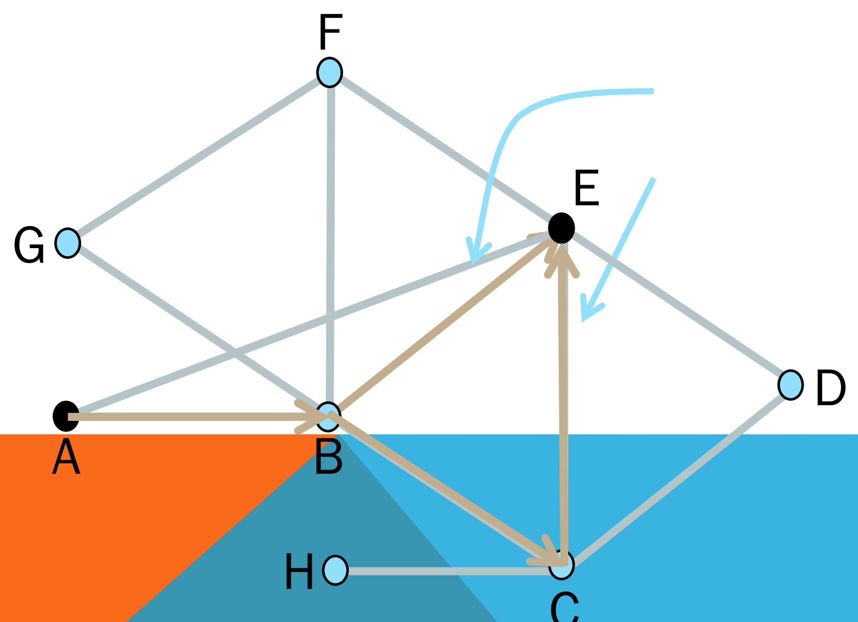
Strategy:

- Include age on LSPs and forget old information that is not refreshed

MULTIPATH ROUTING

More on shortest path routes

- Allow multiple shortest paths



Use ABE as well as ABCE from A→E

MULTIPATH ROUTING

Allow multiple routing paths from node to destination be used at once

- Topology has them for redundancy
- Using them can improve performance

Node	Next hops
A	B, C, D
B	B, C, D
C	C, D
D	D
E	--
F	F
G	F
H	C, D

IMPACT OF ROUTING GROWTH

1. **Forwarding tables grow**
   - Larger router memories, may increase lookup time

2. **Routing messages grow**
   - Need to keep all nodes informed of larger topology

3. **Routing computation grows**
   - Shortest path calculations grow

TECHNIQUES TO SCALE ROUTING

1. **IP prefixes/Aggregation**
   - Route to blocks of hosts
   - Combine, and split, prefixes

2. **Network hierarchy**
   - Route to network **regions**

HIERARCHICAL ROUTING

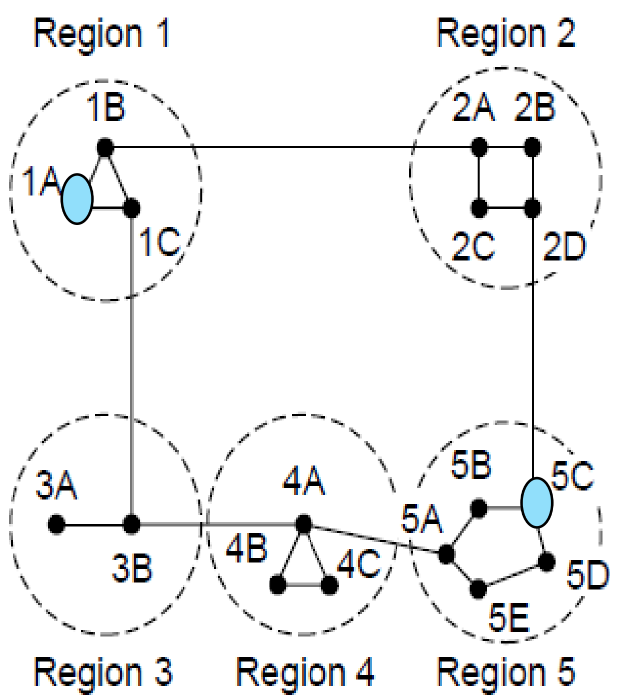
Introduce a larger routing unit

- 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



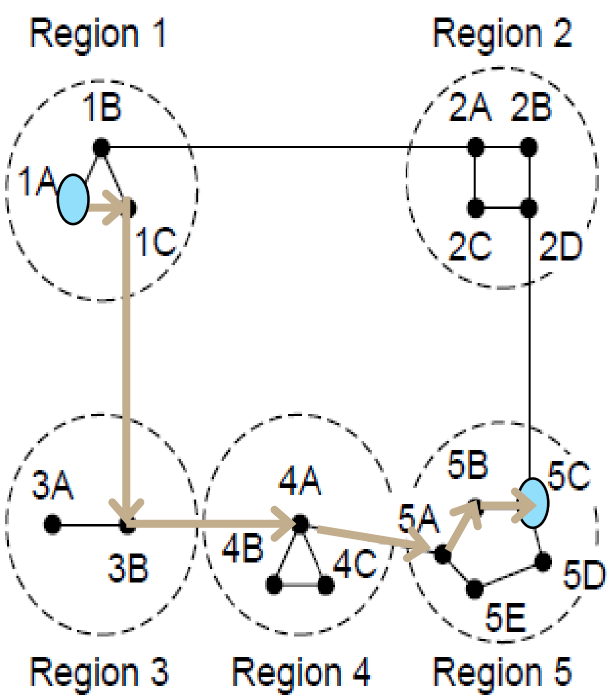
Full table for 1A

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



Full table for 1A

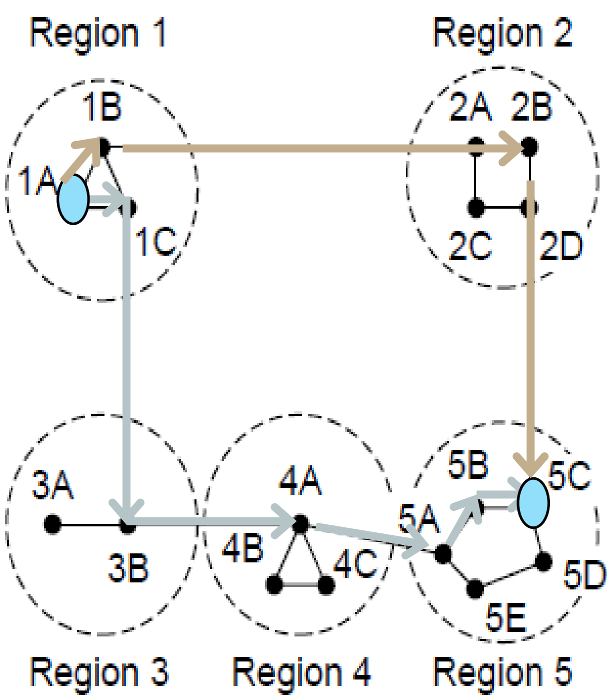
Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

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

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