WAN TECHNOLOGIES AND ROUTING

Large Networks and Wide Areas

- In general, a network technology is classified into one of three broad categories
  - A Local Area Network (LAN)
    * Can span a single building or campus (distance limitation)
    * Can not handle an arbitrary number of computers (size limitation)
  - A Metropolitan Area Network (MAN)
    * Can span a single city
  - A Wide Area Network (WAN)
    * Can span sites in multiple cities, countries, or continents
    * Can handle an arbitrary number of computers
- Although a bridged LAN can extend the distance of a LAN over an arbitrary distance, it is not considered a WAN, because bandwidth limitations prevent a bridged LAN from serving arbitrarily many computers at arbitrarily many sites

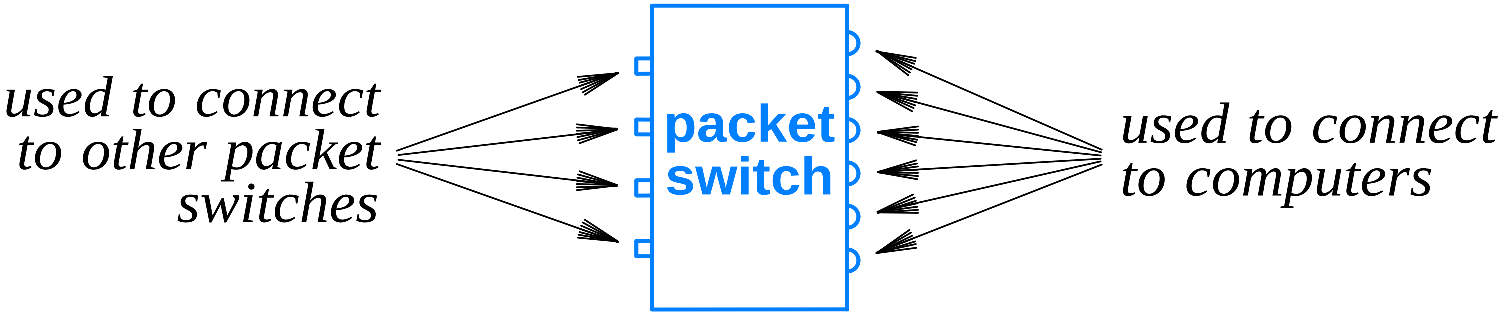
- A WAN not only connects many computers at many sites, but also it must provide sufficient capacity to permit these computers to communicate simultaneously
- WANs consist of two fundamental building blocks
  - Point-to-point long-distance connections
  - Switches to connect and route frames from source to destination, using these point-to-point connections

Point-to-point Long-distance Connections

- In most countries, laws prevent an organization from running a cable, or digging to lay a cable, between sites, unless the organization owns all the properties between the sites
- This problem may be overcome by
  - Using the existing telephone infrastructure
    - Leased lines
  - Using the existing CATV infrastructure
    - Sends/receives over CATV wiring
    - Group of subscribers in neighborhood share bandwidth using both FDM and TDM as well

Packet Switches

- A packet switch moves complete packets from one connection to another
- It is a special-purpose hardware that has a CPU, ROM, RAM, as well as I/O connectors
- These I/O connectors are used to send and receive packets
- A packet switch contains two types of I/O connectors
  - High-speed connectors: to connect the switch to another packet switch, via a point-to-point connection
  - Low-speed connectors: to connect the switch to an individual computer

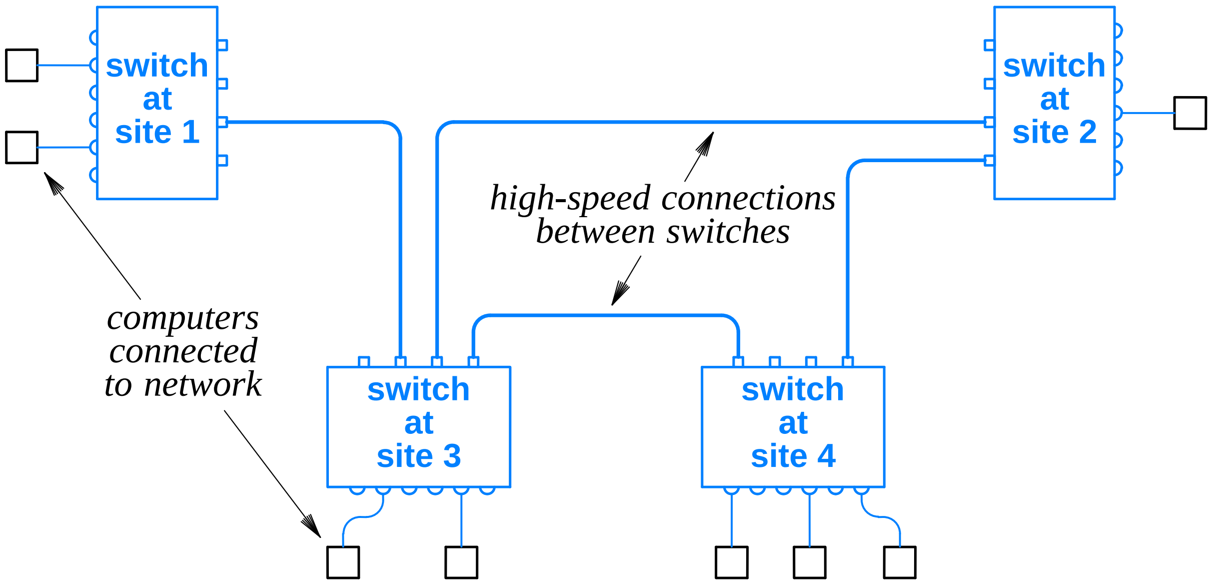


A packet switch

- Note that: LAN switches and WAN switches are not the same (think of them as two different technologies)
  - In a switched-LAN, all connections are short (a few hundred meters); while in a WAN switch, some long-distance connections must also exits
  - A switch in a LAN usually uses flat addressing, while a switch in a WAN usually use hierarchical addressing
  - Both of their hardware and software are not the same

Building a WAN

- Place one or more packet switches at each site
- Interconnect these switches, where
  - Connection capacities are chosen to accommodate the expected traffic
  - The interconnections among switches are chosen to provide redundancy
- Additional switches or interconnections can be added as needed to increase the capacity of the WAN



A small WAN formed by interconnecting 4 packet switches and 8 computers

How Can a WAN Grow Up?

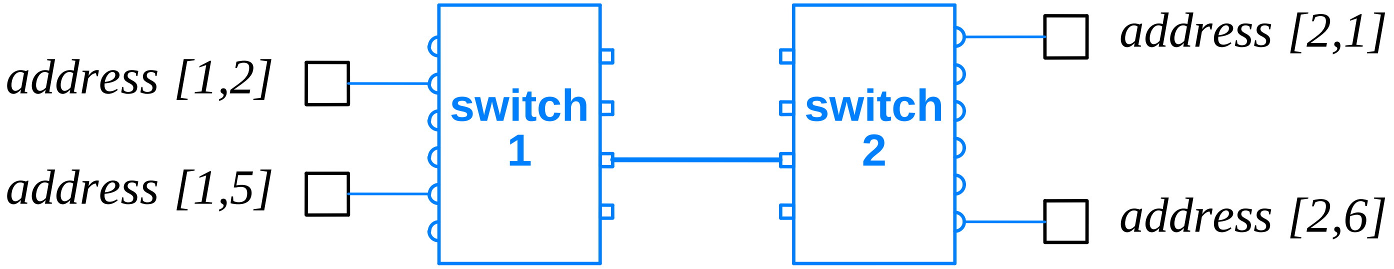
- By adding more computers to the existing switches
  - May lead to a small increase in network utilization
- By adding more switches
  - Interior switches to handle load
    - Only connects switches together and has no computers attached to any of its ports
    - Leads to better service and more reliability
  - exterior switches
    - Connects switches together and has computers attached to its ports
    - Leads to better service, more reliability, and a large increase in the network capacity
- By adding more connections
  - Leads to better service and more reliability

Store and Forward

- A basic paradigm used with WAN packet switching systems
  - A packet is sent from a source computer
  - It travels from switch to switch, where
    - The switch's I/O hardware
      - Stores the arrived packets in the switch's memory
      - Informs the switch's processor that a packet has arrived
    - The switch's processor
      - Examines packet's destination address
      - Places the packet in a queue associated with the I/O hardware that leads to the destination
    - The switch's I/O hardware that leads to the destination
      - As soon as it becomes idle, it extracts and begins sending the next packet in its queue

WAN Physical Addressing

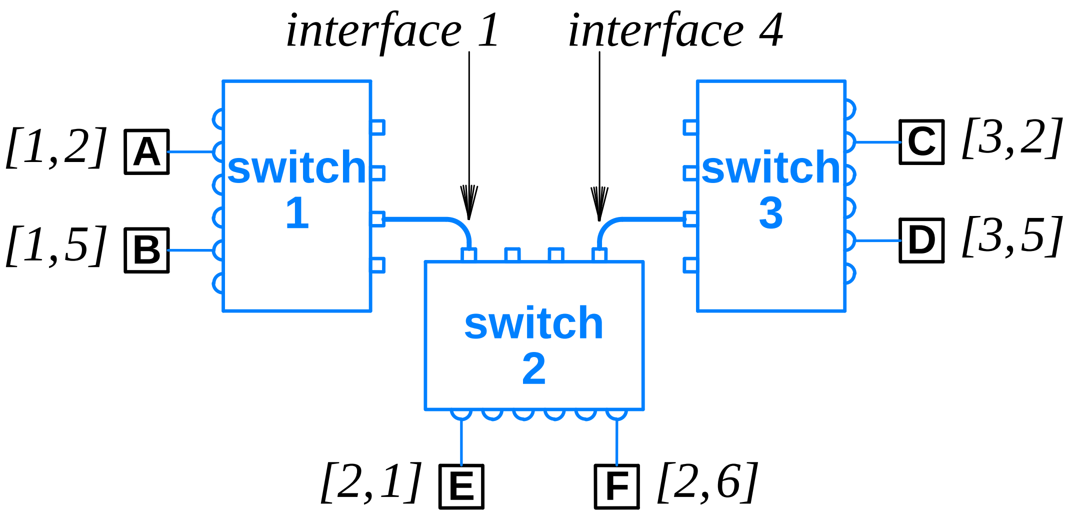
- Each computer attached to the WAN must have a unique physical address
- When sending a frame to another computer, the sender must supply the destination's address
- To make forwarding more efficient, many WAN use a hierarchical addressing scheme (e.g., two-part addressing)
- In the two-part addressing scheme, a physical address consists of two parts
  - The first part identifies a packet switch
  - The second part identifies a computer attached to that packet switch
- A two-part address is encoded as a single integer number



Example of hierarchical addresses in a WAN

Next-Hop Forwarding

- A packet switch must choose a path over which to forward each packet
- To make this choice, a switch uses the destination address stored in a packet, as well as next-hop forwarding information stored in the switch itself
- Next-hop forwarding information includes each possible destination and the next-hop used to reach that destination



(a)

destination	next hop
[1,2]	interface 1
[1,5]	interface 1
[3,2]	interface 4
[3,5]	interface 4
[2,1]	computer E
[2,6]	computer F

(b)

(a) A network consisting of 3 switches
(b) The next-hop forwarding information found in switch 2

- Next-hop forwarding information is organized in a table called a routing table
- The process of forwarding a packet to its next hop is known as routing
- Note that: only the first part of a hierarchical address is required to identify the next-hop; (shorter routing table)

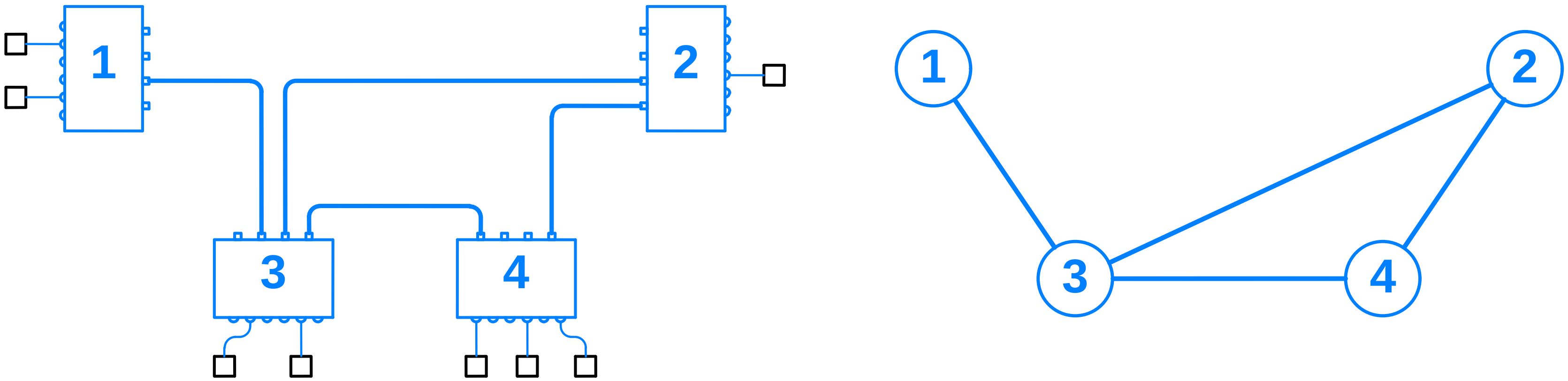
Destination	Next Hop
(1, anything)	interface 1
(3, anything)	interface 4
(2, anything)	local computer

An abbreviated version of switch 2 routing table

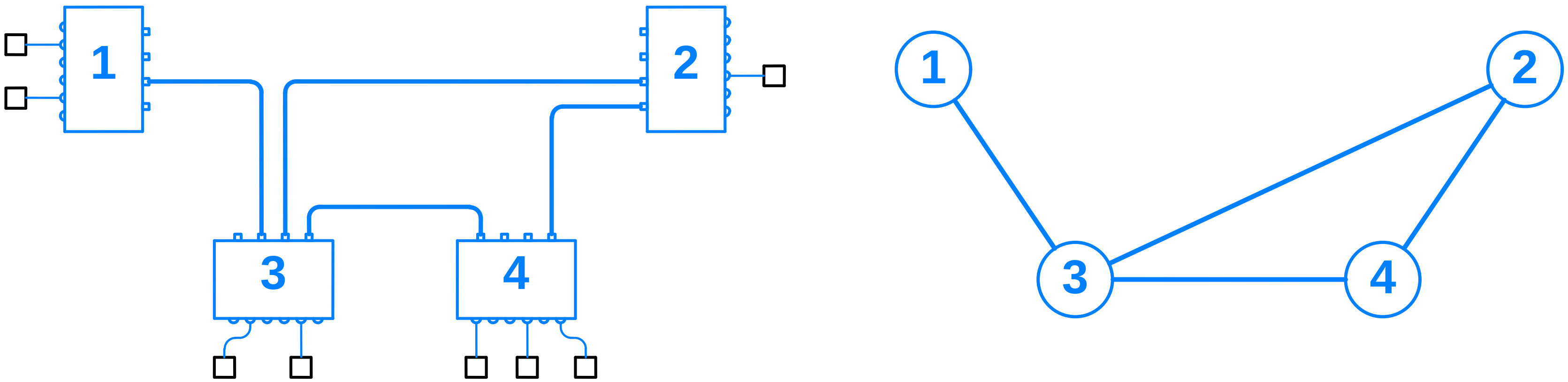
- The next-hop forwarding does not depend on the packet's original source or on the path the packet has taken before it arrives at a particular switch (source independence)
- When the packet reaches the switch to which the destination computer is attached, the switch examines the second part of the address to identify the appropriate local I/O port to use

From Routing to Graph Theory

- The easiest way to think about routing in a WAN is to model a network as a graph
  - Each switch is represented by a node in the graph
  - Each direct connection between two switches is represented by an edge, a.k.a. link, in the graph
- Note that: attached computers have no presence in the graph



An example of modeling a network as a graph



An example of modeling a network as a graph

node 1		node 2		node 3		node 4	
Destination	Next Hop	Destination	Next Hop	Destination	Next Hop	Destination	Next Hop
1	-	1	(2,3)	1	(3,1)	1	(4,3)
2	(1,3)	2	-	2	(3,2)	2	(4,2)
3	(1,3)	3	(2,3)	3	-	3	(4,3)
4	(1,3)	4	(2,4)	4	(3,4)	4	-

The routing table for each node

Default Routes

- A default entry is present only if more than one destination has the same next-hop value
- If the routing scheme does not find an explicit entry for a given destination, it uses the default entry

node 1		node 2		node 3		node 4	
Destination	Next Hop	Destination	Next Hop	Destination	Next Hop	Destination	Next Hop
1	-	1	(2,3)	1	(3,1)	1	(4,3)
2	(1,3)	2	-	2	(3,2)	2	(4,2)
3	(1,3)	3	(2,3)	3	-	3	(4,3)
4	(1,3)	4	(2,4)	4	(3,4)	4	-

The routing table for each node

node 1		node 2		node 3		node 4	
Destination	Next Hop	Destination	Next Hop	Destination	Next Hop	Destination	Next Hop
1	-	2	-	1	(3,1)	2	(4,2)
*	(1,3)	4	(2,4)	2	(3,2)	4	-
		*	(2,3)	3	-	*	(4,3)
				4	(3,4)		

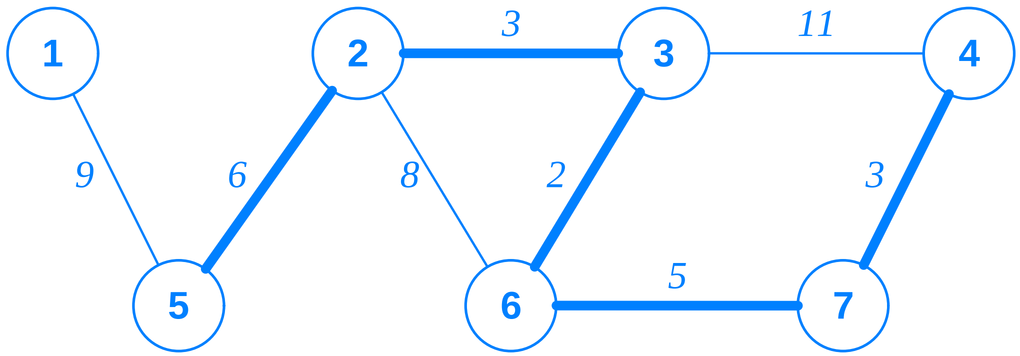
The routing table for each node

Routing Table Computation

- How is a routing table constructed?
  - Using manual computation
    - Table created by hand
    - Useful in small networks
    - Useful if routes never change
    - Impractical in large networks
  - Using automatic computation
    - Static routing
      - A routing table is built and installed once a packet switch boots
      - The routing table is not changeable
    - Dynamic routing
      - An initial routing table is built and installed once a packet switch boots
      - The routing table can be changed, if network conditions are changed, e.g., a failure occur

Shortest Path Between nodes

- A weight is assigned to each edge
- This weight, a.k.a. distance, may represent
  - Geographic distance
  - Economic cost
  - Inverse of capacity
- It is required to find the shortest path between each pair of nodes, where the distance along the path is the sum of the weights on the edges
- Note that: a path with the fewest number of edges may not be the path with least weight



A graph with weights assigned to edges

Dijkstra's Shortest Path Algorithm

- Basic idea
  - Start with the source node
  - Move outward
  - At each step
    - Find node u such that it
      - Is the "*closest*" to source
      - Has not been considered before
    - For each direct neighbor to u
      - Find the distance from u to its direct neighbor v
      - If this distance is shorter, make a path from the source node to v through u

Input

- A graph with weighted edges, $Weight(u, v)$
- A designated source node, n

Output

- A set of shortest paths from node n to each other node
- The cost of each path

Method

- Initialize set $Source$ to contain all nodes except the source node
- Initialize array $Distance$ so that $Distance[v]$ is the weight of the edge from source node n to node v, i.e., $Weight(u, v)$, if such a direct edge between n and v exist, o.w. infinity
- Initialize array $Route$ so that $Route[v]$ is assigned v if an edge between n and v exist, o.w. zero

- While (set $Source$ is not empty)
  - Choose a node u from $Source$ such that $Distance[u]$ is minimum
  - If($Distance[u]$ is infinity)
    - Error: no path exists to nodes in $Source$
    - Quit
  - Delete u from set $Source$
  - For each node v such that (u,v) is an edge
    - If(v is still in $Source$)
      - $c = Distance[u] + Weight(u, v)$
      - If ($c < Distance[v]$)

        $Route[v] = Route[u]$

        $Distance[v] = c$

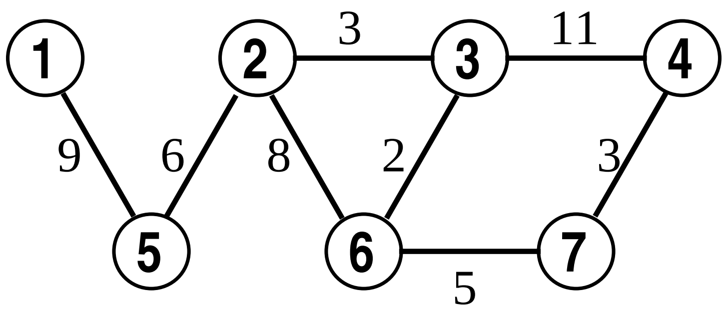
Source $= \{1, 2, 4, 5, 6, 7\}$

Distance =

1	2	3	4	5	6	7
∞	3	∞	11	∞	2	∞

Route =

1	2	3	4	5	6	7
—	2	—	4	—	6	—



$u = 6$

Source $= \{1, 2, 4, 5, 7\}$

$v \in \{2, 3, 7\}$

$v = 2 \rightsquigarrow c = 2 + 8 > 3$

$v = 7 \rightsquigarrow c = 2 + 5 < \infty$

Distance =

1	2	3	4	5	6	7
∞	3	∞	11	∞	2	7

Route =

1	2	3	4	5	6	7
—	2	—	4	—	6	6

$u = 2$

Source $= \{1, 4, 5, 7\}$

$v \in \{3, 5, 6\}$

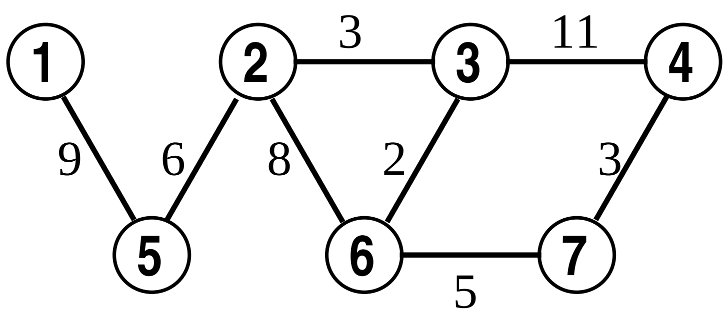
$v = 5 \rightsquigarrow c = 3 + 6 < \infty$

Distance =

1	2	3	4	5	6	7
∞	3	∞	11	9	2	7

Route =

1	2	3	4	5	6	7
—	2	—	4	2	6	6



$u = 7$

Source $= \{1, 4, 5\}$

$v \in \{4, 6\}$

$v = 4 \rightsquigarrow c = 7 + 3 < 11$

	1	2	3	4	5	6	7
Distance =	∞	3	∞	10	9	2	7
Route =	—	2	—	6	2	6	6

$u = 5$

Source $= \{1, 4\}$

$v \in \{1, 2\}$

$v = 1 \rightsquigarrow c = 9 + 9 < \infty$

	1	2	3	4	5	6	7
Distance =	18	3	∞	10	9	2	7
Route =	2	2	—	6	2	6	6

$u = 4$

Source $= \{1\}$

$v \in \{3, 7\}$

$u = 1$

Source $=$ NULL

Shortest Path First (SPF)

- A.k.a. Link-status routing
  - Each switch multicast to all other switches the status of the links connected to it
  - Switches collect incoming status information and use them to build an updated graph of the network
  - Using the updated network graph, each switch re-apply Dijkstra algorithm to produce a routing table for itself.
- SPF can adapt itself to hardware failures
- Note that, all computation can be carried out simultaneously;
  i.e., after the status of a link changes, all packet switches receive a status message, and each switch begins computing its routing table

Distance Vector Routing

- One of the best-known algorithms for distributed route computation
- Switches only communicated with direct attached neighbor switches; They exchange information in their routing tables, i.e., (destination, distance) pairs
- When a switch receives this information, it
  - Compares each item in list to local routes
  - Changes routes, iff better path exists

$\boxed{\text{Given}}$

- A local routing table
- A weight for each link that directly connects to another switch
- An incoming routing message

$\boxed{\text{Output}}$

- An updated routing table

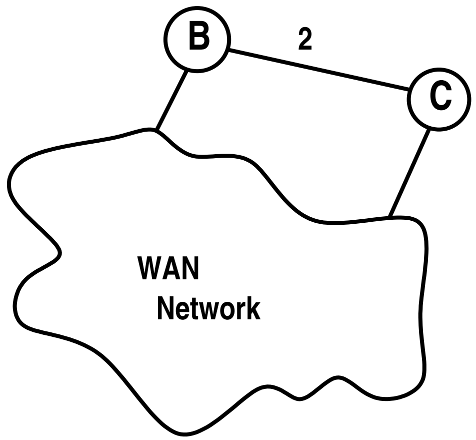
$\boxed{\text{Method}}$

- If the router is just rebooted:
  - Initialize routing table with a single entry that has
    - $Distination =$ local switch
    - $Distance = 0$
    - Next-hop unused

- Repeat forever
  - Wait for the next routing message to arrive over the network from a direct neighbor; let us call the sender switch N
  - For each entry in the message
    - Let V be the destination in the entry and
      Let D be the distance
    - Let $C = D +$ the weight assigned to the link over which the message arrived
    - If (no route exists to V)
      - Add an entry to the local routing table for destination V with next-hop N and distance C
    - Else
      - If (a route exists that has next-hop N)
        Replace the distance in existing route with C
      - Else
      - If (a route exists with distance greater than C)
        Change the next-hop to N and $distance$ to C

An advertised message from C

Destination	Distance
G	3
B	2
H	7
F	3
C	0
L	4



The advertised message from C After incrementing the distance

Destination	Distance
G	5
B	4
H	9
F	5
C	2
L	6

The B's old routing table

Destination	Distance	Next Hop
A	7	A
G	2	C
B	0	-
F	8	F
C	2	C
L	4	F

Update Algorithm

The B's new routing table

Destination	Distance	Next Hop
A	7	A
G	5	C
B	0	-
H	9	C
F	5	C
C	2	C
L	4	F

A Distance Vector Example

WAN Examples

- *Advanced Research Projects Agency Network* (ARPANET)
- X.25
- Frame Relay
- *Switched Multi-megabit Data Service* (SMDS)
- *Asynchronous Transfer Mode* (ATM)