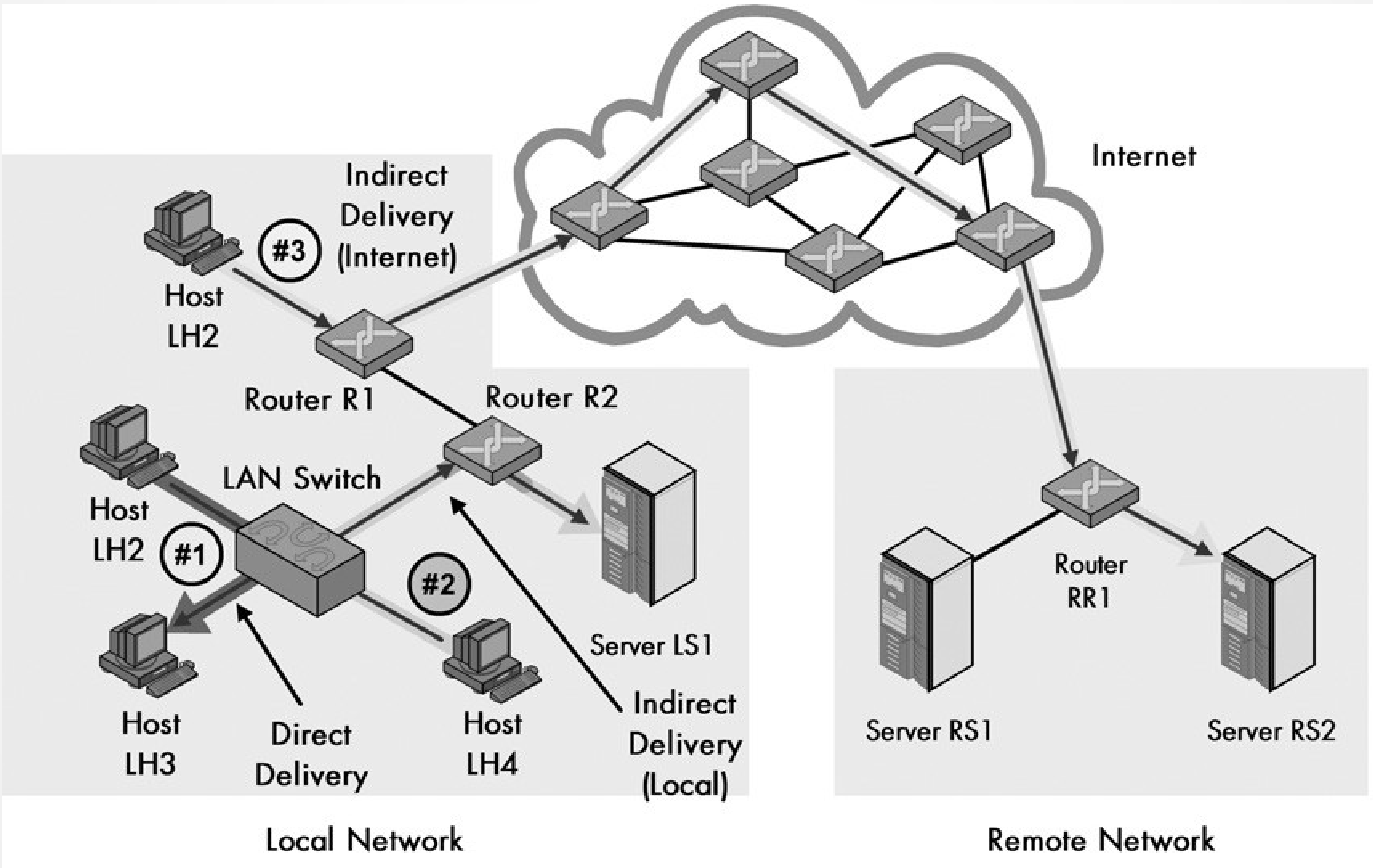
Routing -- Overview

Manas Ranjan Lenka
School of Computer Engineering,
KIIT University

How Routing Works

IP Datagram Delivery



How Routing Works

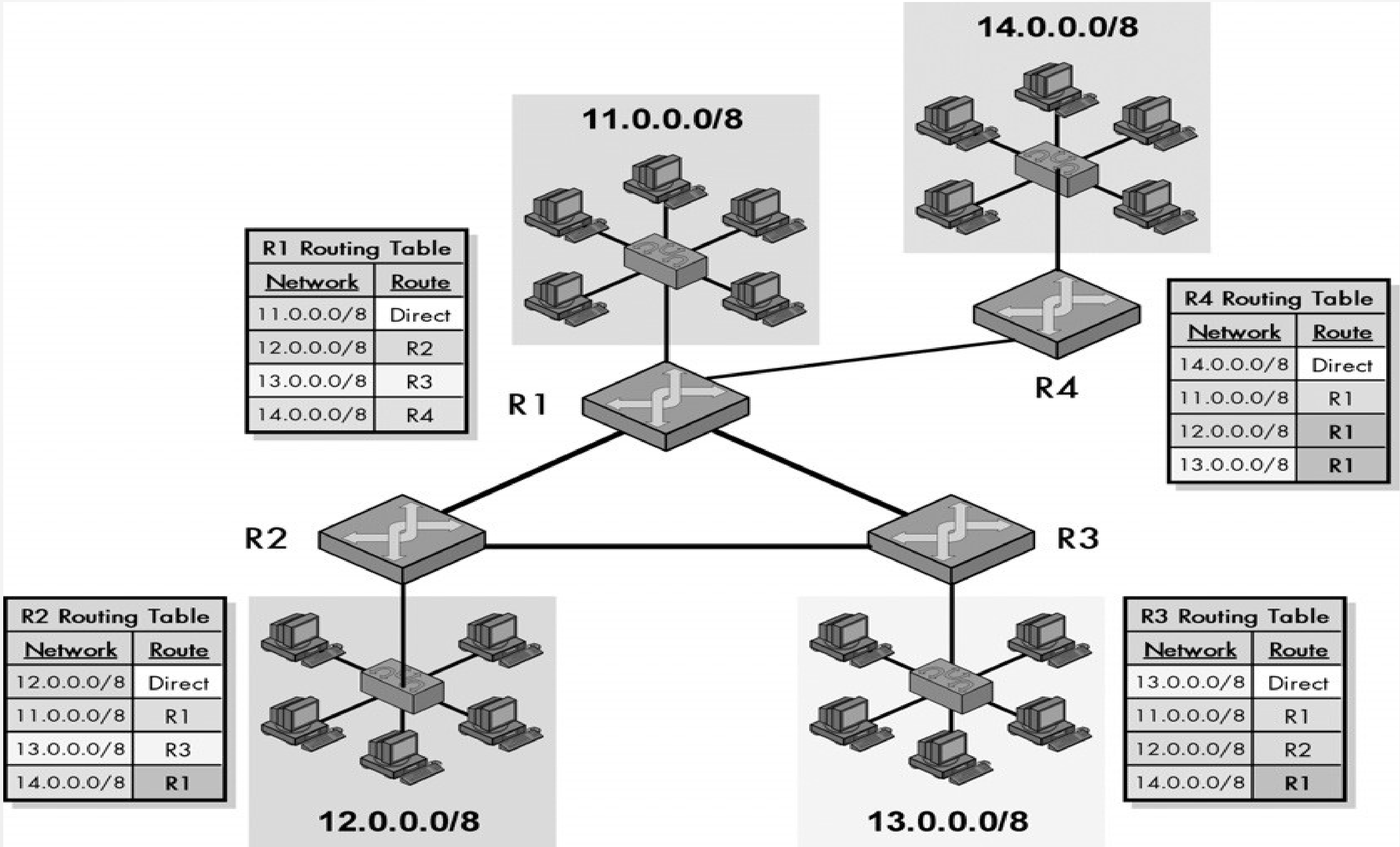
- First, the sender has to determine whether the data can be delivered directly or not, based on IP address of the destination.
- If the destination is another network, a host just sends the datagram to its router.
- If a host has a connection to more than one router, it needs to know which router to use for delivery to the destination network.
- Once a source sends a datagram to its local router, then the datagram's header is examined, and the router decides which device should get the datagram next.

  How does a router know where to send different datagrams?

  - router make decisions about how to route datagrams using its internal data structure called routing table

Routing Table

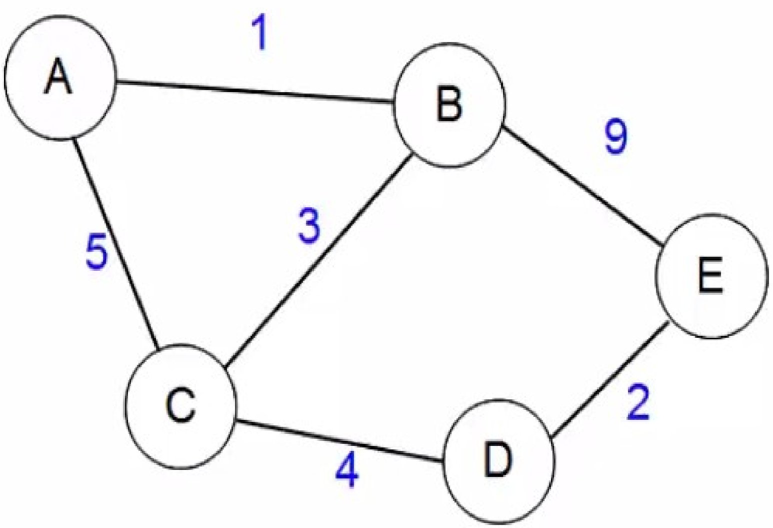
Routing Table, maintained by each node that provides a mapping between different network IDs and the routers to which it is connected.



Routing Algorithm

Goal of Routing

- Single-source 'shortest' path problem: Find least cost path from a source to all other nodes in the graph
- Refer to Dijkstra's algorithm



Static vs Dynamic Approach

– Static: Route computed in advance and downloaded in all routers. Typically used in hosts.

– Dynamic: Handles changes in the topology

- Nodes failure, addition of new nodes, variation in cost.
- Mostly used by routers

Dynamic preferable over static

Central vs Distributed Approach

- Central:
  - All nodes pass neighborhood information to a central node
  - Central node calculates routes and distributes to all
- Distributed:
  Each node determines routes by itself

Distributed preferable to Central

Global vs Local Approach

– Global : Node calculates routes based on full knowledge of entire topology

– Local : Node does not have global information, determine routes based on local message Exchange

Both approaches are used in practice

Popular Approaches

- Dynamic, distributed algorithms
  – Global knowledge: Link State Algorithm
  – Local knowledge: Distance Vector Algorithm

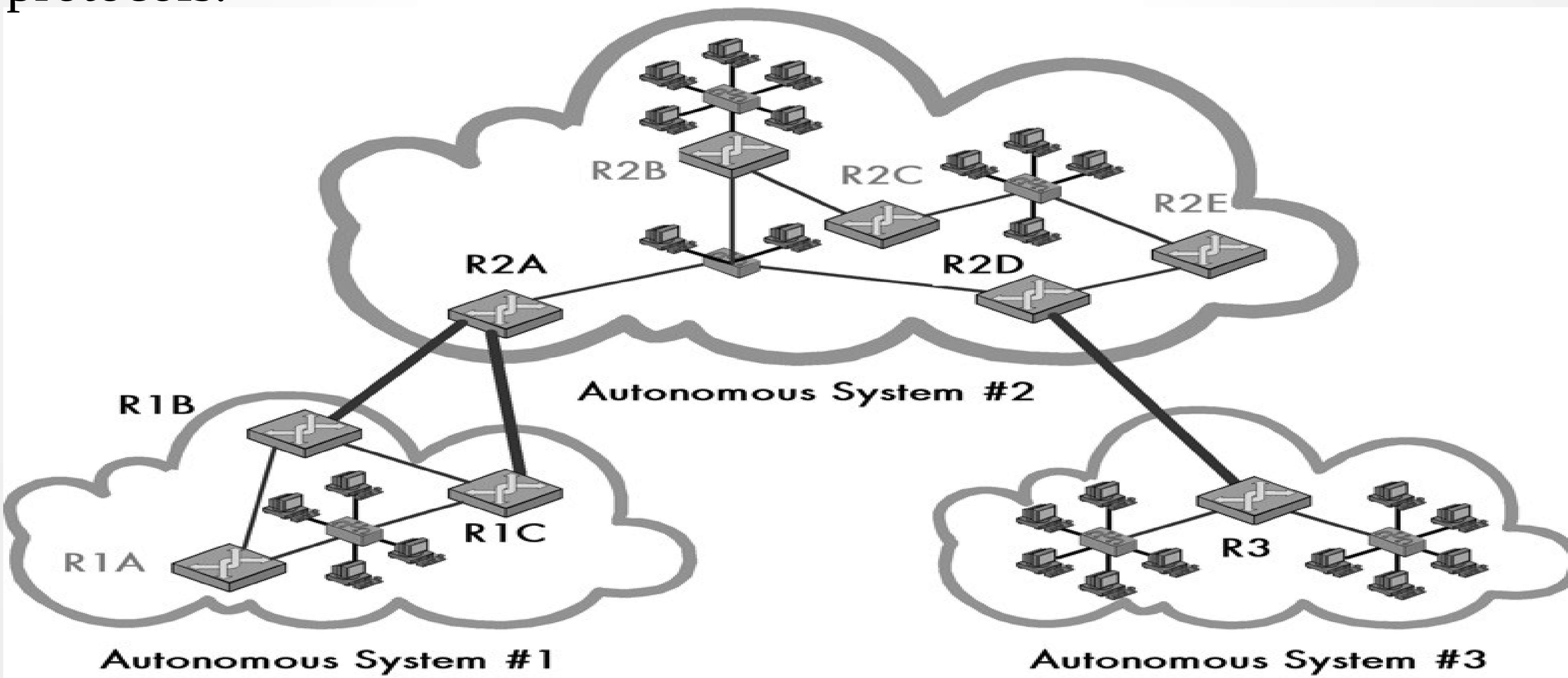
– The distributed architecture treats the internet as a set of independent groups, called an **autonomous system (AS)**.

– An AS consists of a set of routers and networks controlled by a particular organization or administrative entity, which uses a **single consistent policy** for **internal routing**.

AS routing architecture

Internal Routers: connect only to other routers in the same AS and run intraAS routing protocols.

Border Routers: connect both to routers within the AS and to routers in one or more other ASes. They run both intra and inter AS routing protocols.



Routing In Internet

- Routing Protocols used in Internet
  - Intra domain (Intra AS) routing
    - Link State/Shortest-Path First (Ex: OSPF)
    - Distance Vector/Bellman-Ford (Ex: RIP)
  - Inter domain (Inter AS) routing
    - Path Vector (Ex: BGP)

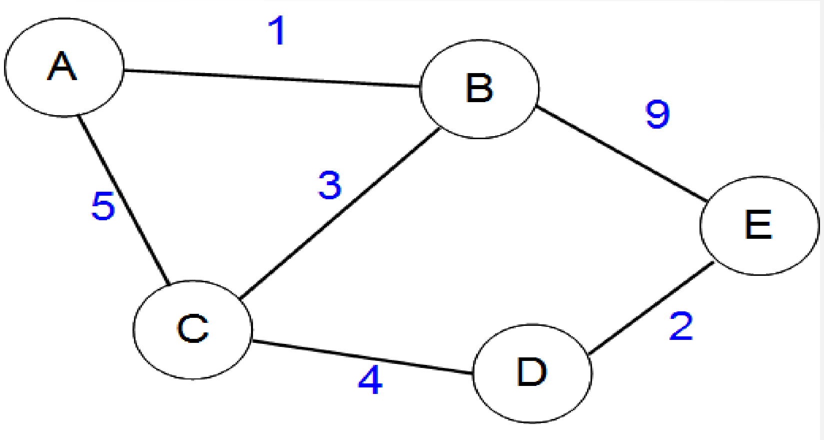
Distance Vector Routing Algorithm

- Also known as Bellman-Ford algorithm

- Implemented in Internet under the routing protocol standard RIP (Routing Information Protocol)

- Now, it is not much used

Basic Idea



- Each node maintains a routing table (distance vector)
  - Destination
  - Estimated cost to destination
  - Next hop via which to reach destination
- Initial state: Cost to neighbors
- Each node exchanges the “Routing Table” info with all its neighbors

Basic Idea

- On receiving a message from a neighbor v,
  - Update cost (estimate) to destinations based on above Bellman-ford equation to find least cost;
    - Bellman-Ford equation
      - $D_x(y) = \min\{D_x(y), (c(x,v) + D_v(y))\}$
      - $D_x(y)$ – least cost path from node x to y
      - $c(x,v)$ – cost of going from x to v
  - Estimated costs finally converge to optimal cost after series of message exchanges

Distance-Vector Algorithm

At each node, x:

Initialization:

for all destinations y in N:

{

if(y is a neighbor)

$D_x(y) = c(x,y)$

else

$D_x(y) = \infty$

}

send all the distance vectors to their neighors, $D_x(y)$: y in N

loop

wait (until I see a link cost change to some neighbor v or until I receive a distance vector from some neighbor v)

for each y in N:

$D_x(y) = \min\{D_x(y), (c(x,v) + D_v(y))\}$

if $D_x(y)$ changed for any destination y

send distance vector $D_x(y)$: y in N to all neighbors

forever

Distance-Vector Algorithm

Example

Reference Node C

initial routing @ A

D	C	H
A	5	A
B	3	B
D	4	D

Routing Table of C

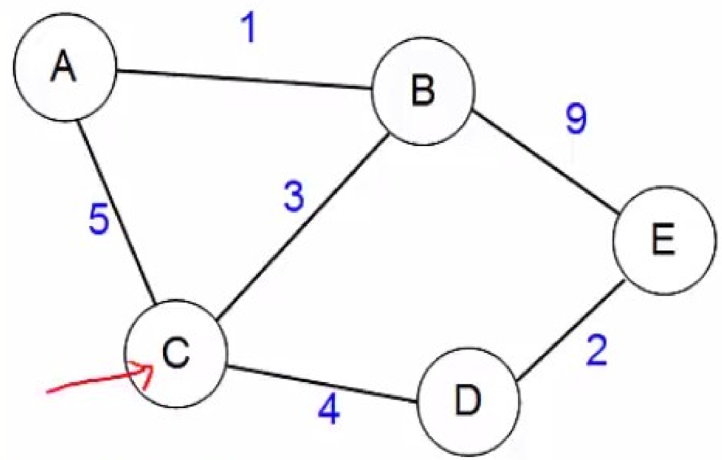
(1)

To	A
A	0
B	1 =6
C	5 x

Message from A
C to A: C = 5

D	C	H
A	5	A
B	3	B
D	4	D

Routing Table of C



D	C	H
A	5	A
B	3	B
D	4	D

Routing Table of C

(2)

To	B
A	1
B	0
C	3
E	9 B

Message from B
C to B: C = 3

D	C	H
A	4	B
B	3	B
D	4	D
E	12	B

Routing Table of C

D	C	H
A	4	B
B	3	B
D	4	D
E	12	B

Routing Table of C

(3)

To	D
C	4
D	0
E	2

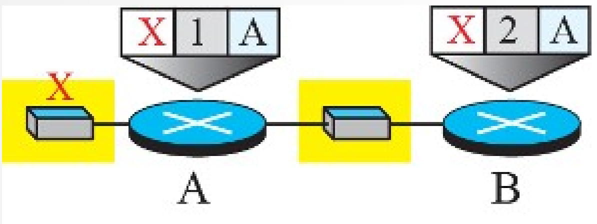
Message from D
C to D: C = 4

D	C	H
A	4	B
B	3	B
D	4	D
E	6	D

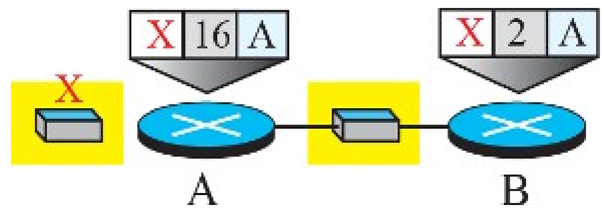
Routing Table of C

Problem with Distance Vector Algo

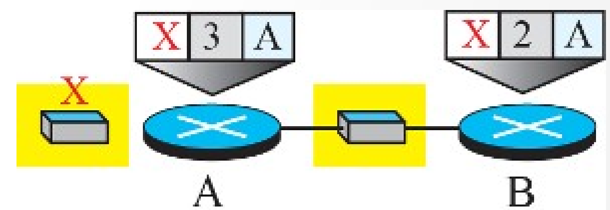
Two Node loop (Count to Infinity)



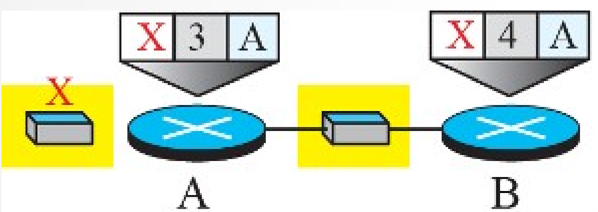
a. Before failure



b. After link failure

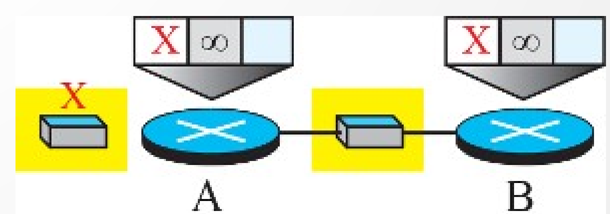


c. After A is updated by B



d. After B is updated by A

• • •



c. Finally

Link State Routing Algorithm

Idea:

- Two Phases

- Phase 1: Reliable flooding
  - Initial State: Each node knows the cost to its neighbors
  - Final State: Each node knows the entire graph (network topology)

- Phase 2: Route calculation
  - Each node uses Dijkstra's algorithm on the graph to calculate optimal routes to all nodes

Reliable Flooding

- Each node sends its link-state (neighborhood information) to all nodes in the topology reliably.

- What message to send?

  Link-state packet (LSP)

  - The id of node sending the packet
  - The link-state of the node: neighborhood information (list of neighbors and cost to each)
  - Sequence number
  - Time-To-Live (TTL)
- What to do when you receive an LSP?
- When to send LSPs?

Reliable Flooding

- What to do when you receive an LSP?
  - Suppose a node X receives an LSP generated by node Y (Y need not be X's neighbor)
  - Did I (i.e. X) hear from Y before?
    - No: Store the link-state information. Start an ageing timer.
    - Yes: Compare sequence number of this packet (Seq_new) with stored information (Seq_old).
      - If Seq_new > Seq_old, overwrite old link-state information, refresh ageing timer, forward to 'required' neighbors
      - If Seq_old >= Seq_new, discard received packet

Reliable Flooding

- When to send LSPs?
  - Flooding leads to lot of traffic
    - – Avoid to the extent possible
  - Triggered updates
    - – A node floods the network whenever its link-state information changes
  - Periodic updates
    - – Need not be sent often, use long timers (order of hours)

Route Calculation

- Once a node has a LSP packet from every node, it has complete graph information

- Use Dijkstra’s algorithm to calculate shortest paths to nodes

Shortest Path using Dijkstra's algorithm

Notation:

- c(i,j): link cost from node i to j. cost infinite if not direct neighbors
- D(v): current value of cost of path from source to dest v.
- n(v): next hop from this source to v along the least cost path.
- N: set of nodes whose least cost path definitively known.

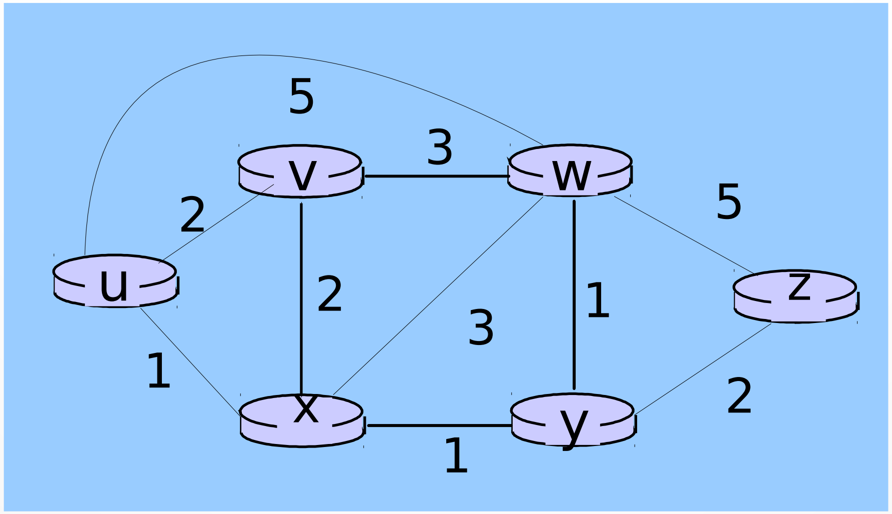
Shortest Path using Dijkstra's algorithm

```
Initialization:
  N = {u}
  for all nodes v
    if v adjacent to u
        then D(v) = c(u,v)
    else D(v) = ∞

Loop
  find w not in N such that D(w) is a minimum
  add w to N
  update D(v) for all v adjacent to w and not in N :
     D(v) = min( D(v), D(w) + c(w,v) )
  /* new cost to v is either old cost to v or known
   shortest path cost to w plus cost from w to v */
until all nodes in N
```

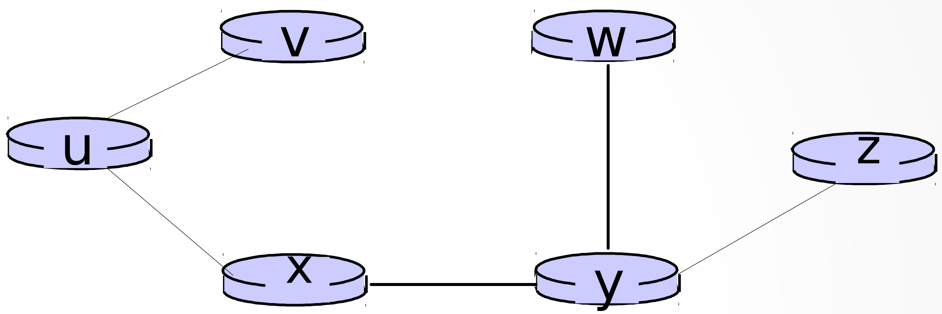
Dijkstra’s algorithm: example

Step	N	D(v),n(v)	D(w),n(w)	D(x),n(x)	D(y),n(y)	D(z),n(z)
0	u	2,v	5,w	1,x	∞	∞
1	ux	2,v	4,x		2,x	∞
2	uxy	2,v	3,x			4,x
3	uxyv		3,x			4,x
4	uxyvw					4,x
5	uxyvwz					



Dijkstra’s algorithm: example

Resulting shortest-path tree from u:



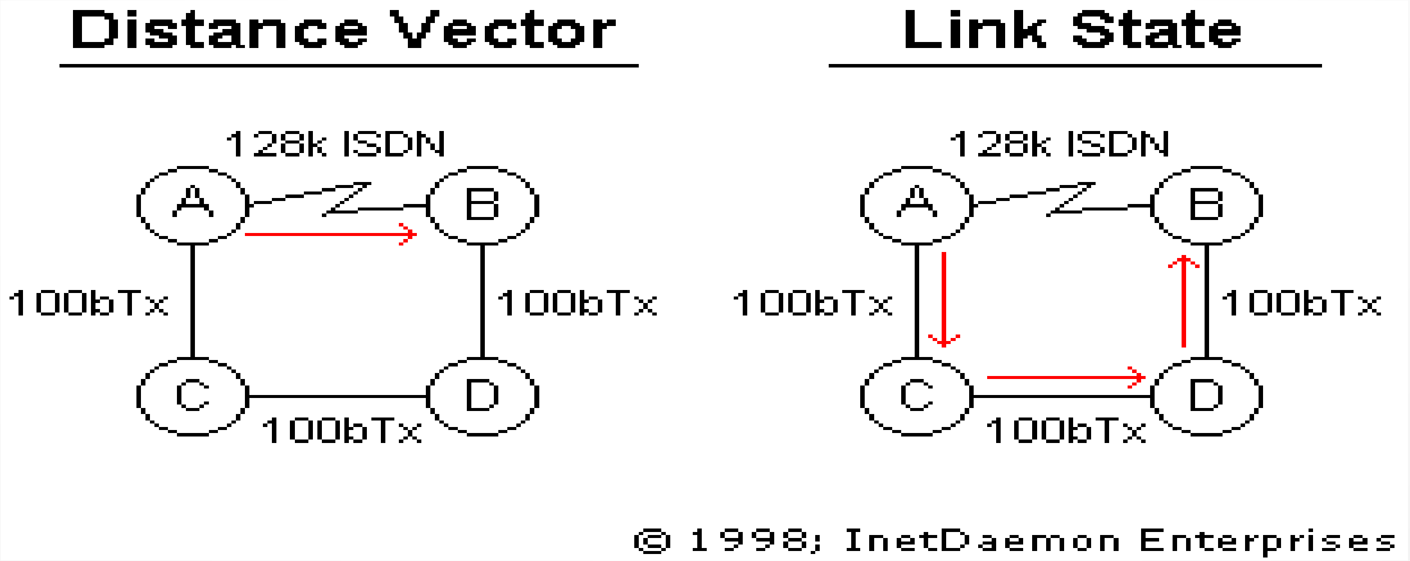
Resulting forwarding table in u:

destination	Outgoing link
v	n(v) = v
x	n(x) = x
y	n(y) = x
w	n(w) = x
z	n(z) = x

Link State vs. Distance Vector

If all routers were running a Distance Vector protocol, the path or 'route' chosen would be from A B directly over the ISDN serial link, even though that link is about 10 times slower than the indirect route from A C D B.

A Link State protocol would choose the A C D B path because it's using a faster medium (100 Mb Ethernet). In this example, it would be better to run a Link State routing protocol, but if all the links in the network are the same speed, then a Distance Vector protocol is better.



Hierarchical Routing

- Our routing study thus far - idealization
  - all routers identical
  - network “flat”

But in practice, this is not the reality because of the following reasons.

- scale: with 200 million destinations:
  - can’t store all dest’s in routing tables!
  - routing table exchange would exhaust links!

- administrative autonomy
  - internet = network of networks
  - each network admin may want to control routing in its own network

Hierarchical Routing

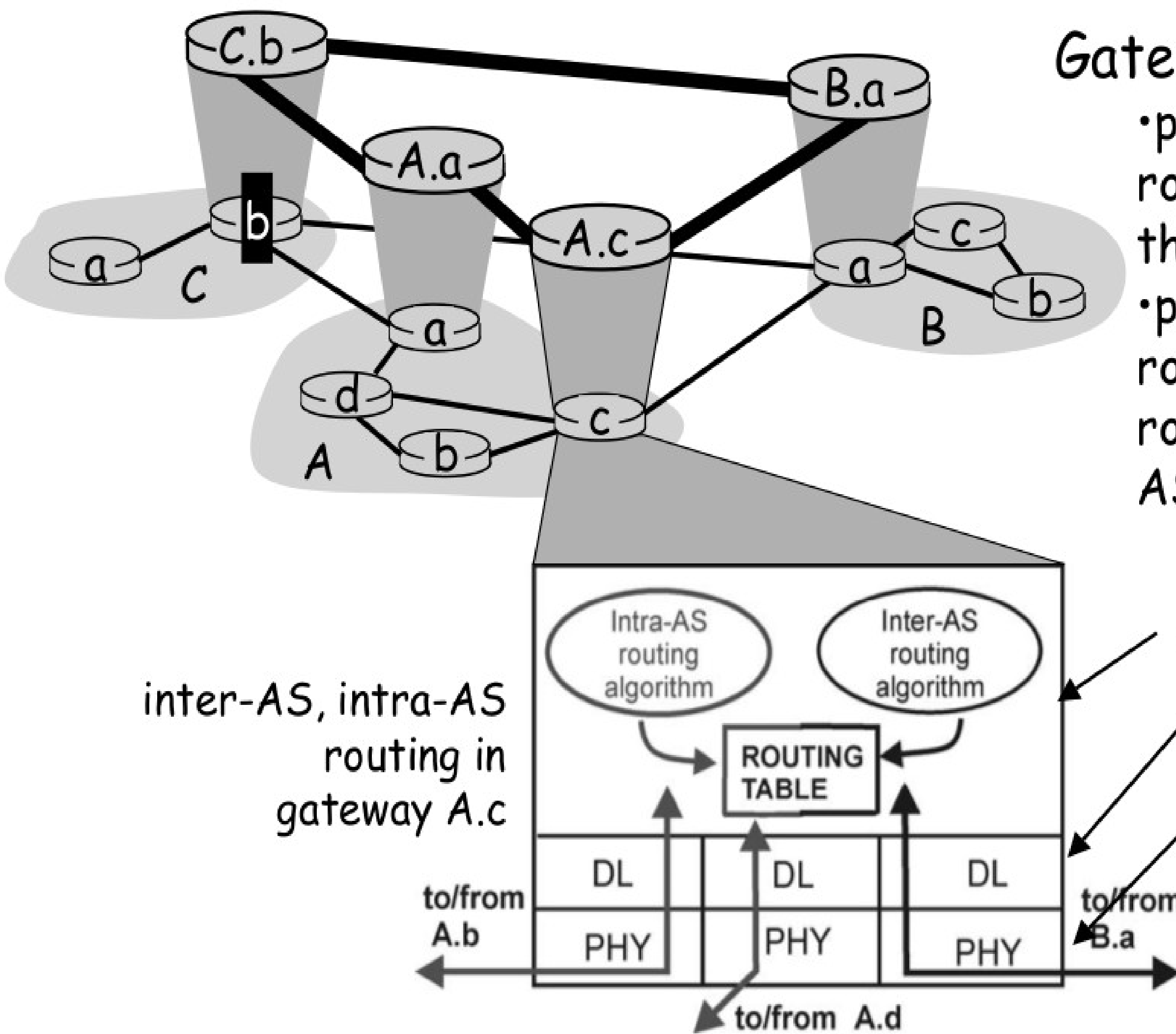
- aggregate routers into regions, “autonomous systems” (AS)

- routers (Internal routers) in same AS run same routing protocol
  – “intra-AS” routing protocol
  – routers in different AS can run different intra-AS routing protocol

- gateway routers (Border routers) at “edge” of its own AS
  – special routers in AS
  – run intra-AS routing protocol with all other routers in AS
  – run inter-AS routing protocol with other gateway routers

Hierarchical Routing

Intra-AS and Inter-AS routing

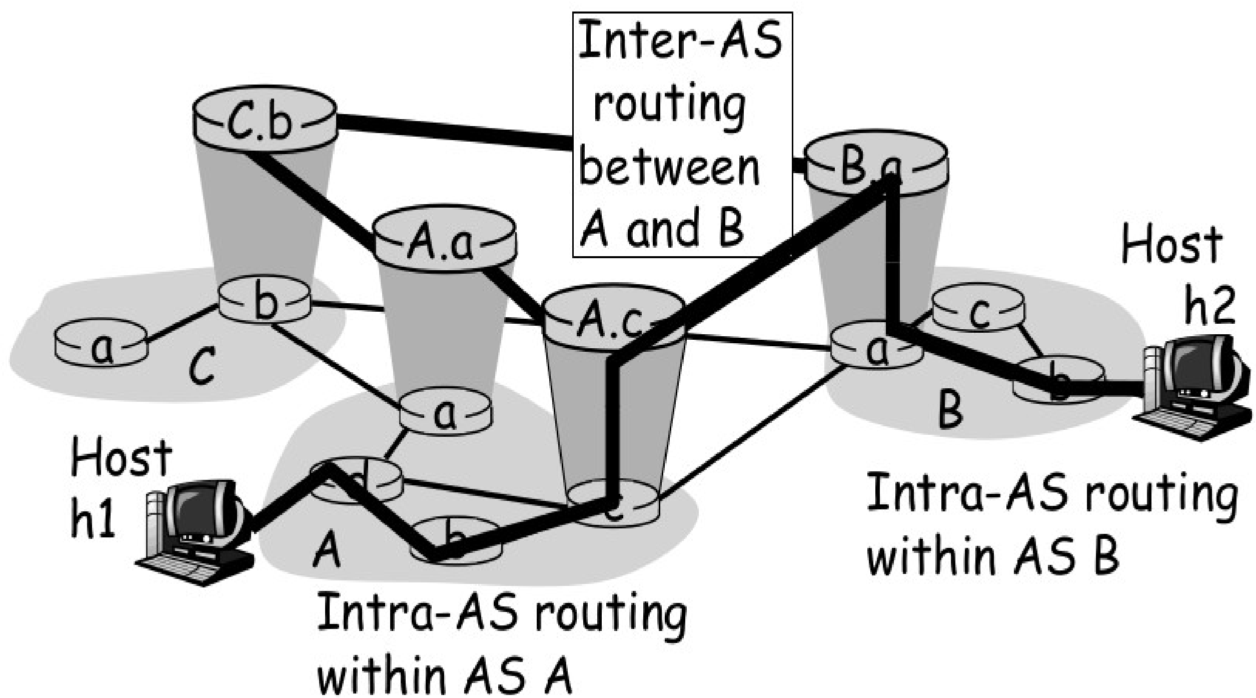


Gateways:

- perform inter-AS routing among themselves
- perform intra-AS routers with other routers in their AS

Hierarchical Routing

Intra-AS and Inter-AS routing



Hierarchical Routing Problems

- source AS has one link that leads outside the AS then its easy.

- suppose source AS has 2 or more links that lead outside the AS?
  - where to forward the packet?

Solution:
  - AS needs to learn which destination are reachable through which gateway routers.
  - Propagate this reachability information to all other routers within the AS.
  - These 2 tasks are handled by inter-AS routing protocol.

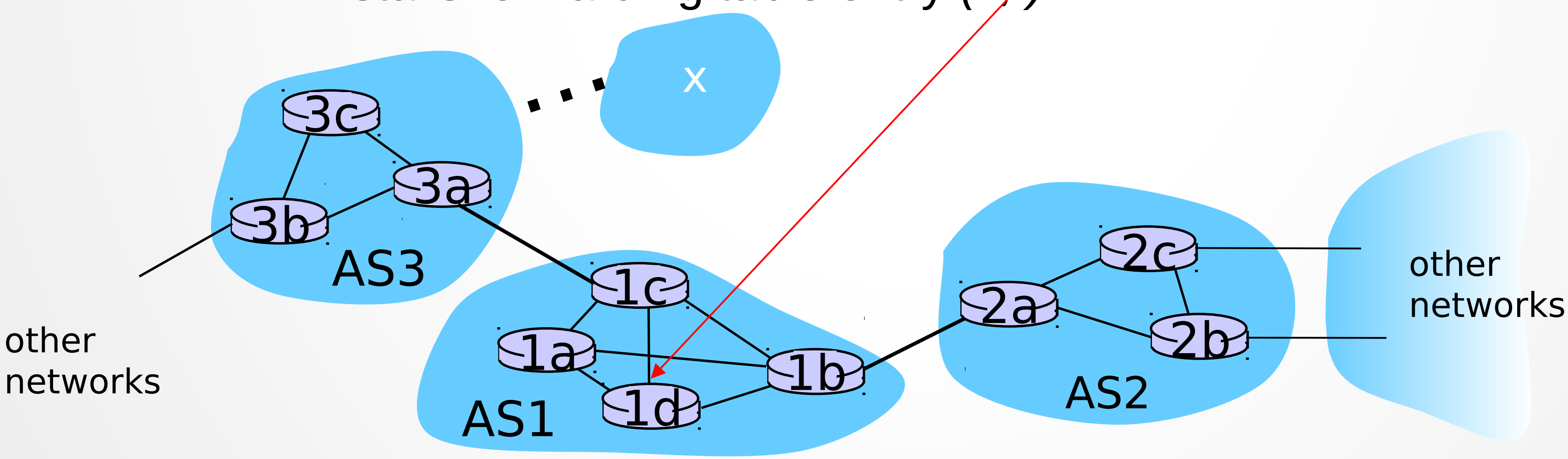
- Since inter-AS routing protocol involves the communication between 2 AS, hence the 2 communicating AS must run the same inter-AS routing protocol.

- In fact, in Internet all AS run the same inter-AS routing protocol called BGP4

Setting forwarding table in router

- suppose AS1 learns (via inter-AS protocol) that subnet *x* reachable via AS3 (gateway 1c) but not via AS2.
  - inter-AS protocol propagates reachability info to all internal routers

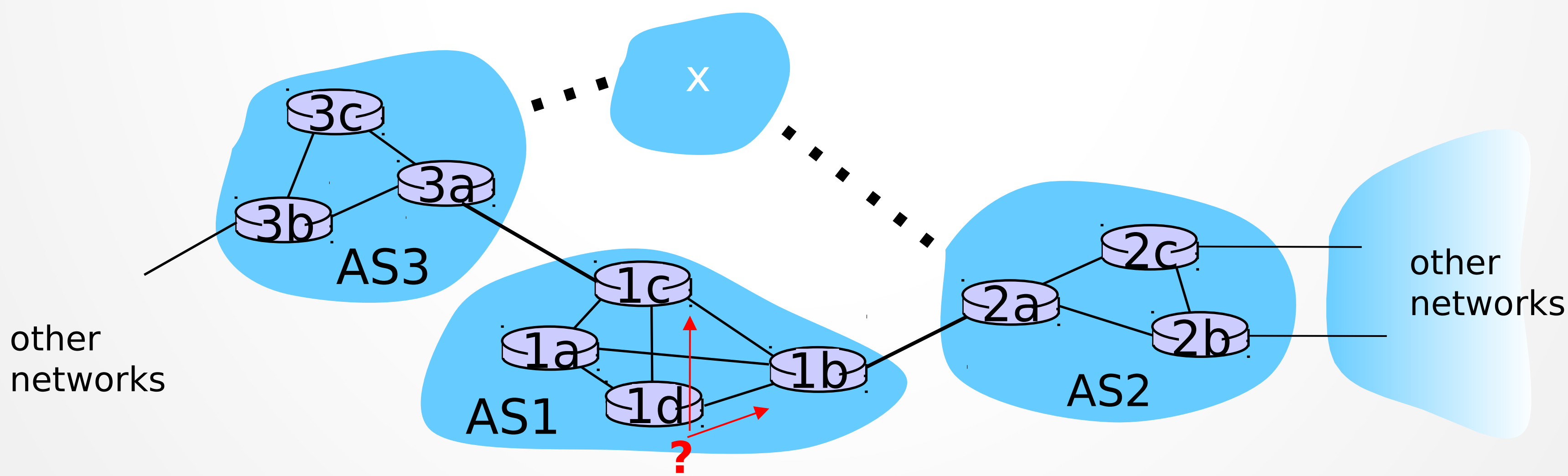
- router 1d determines from intra-AS routing info that its interface *I* is on the least cost path to 1c.
  - installs forwarding table entry *(x,I)*



Choosing among multiple ASes

- now suppose AS1 learns from inter-AS protocol that subnet *x* is reachable from AS3 *and* from AS2.

- to configure forwarding table, router 1d must determine which gateway it should forward packets towards for dest x
  - this is also job of inter-AS routing protocol!



Choosing among multiple ASes

- hot potato routing: send packet towards closest of two routers.