UCL
Université catholique de Louvain

ip networking lab
ucl, louvain-la-neuve, belgium

Computer Networking : Principles, Protocols and Practice

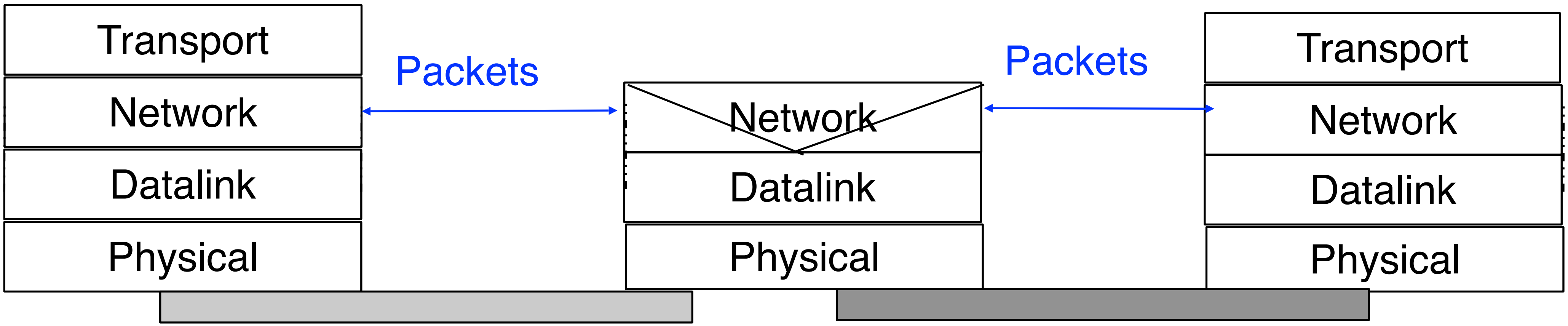
Part 4 : Network Layer

Olivier Bonaventure
http://inl.info.ucl.ac.be/

Network layer

- Basics
  - Datagram mode
  - Virtual circuits
- Routing
- IP : Internet Protocol
- Routing in IP networks

The network layer



- Goal
  - Allow packets to be forwarded from any source to any destination through heterogeneous networks and routers
- Services
  - Unreliable connectionless service
  - Reliable connection-oriented service

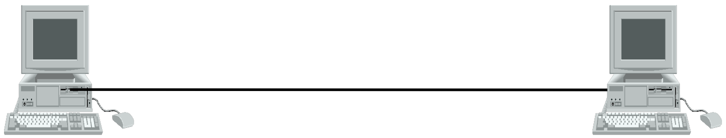
Two types of datalink layers

Two types of datalink layers

- WAN type datalink layer
  - PPP, HDLC
    - Reliable exchange of frames between two hosts attached to the same “link”
    - Mainly used by wide area networks

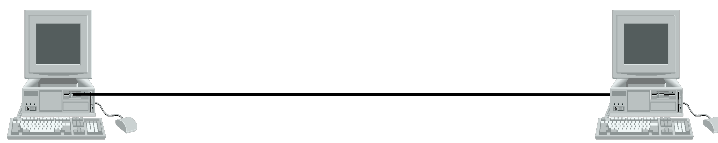
Two types of datalink layers

- WAN type datalink layer
  - PPP, HDLC
    - Reliable exchange of frames between two hosts attached to the same "link"
    - Mainly used by wide area networks

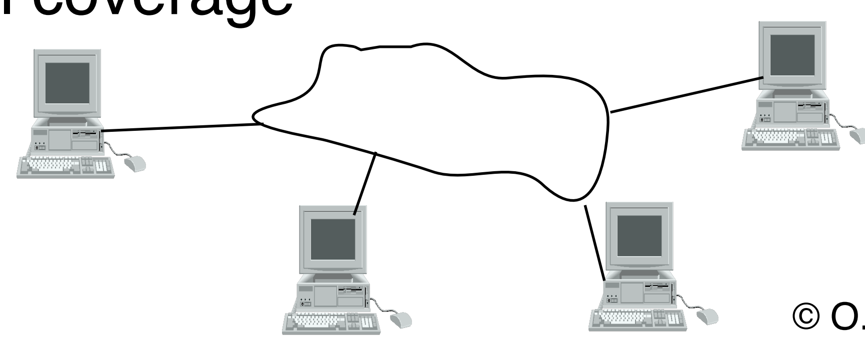


Two types of datalink layers

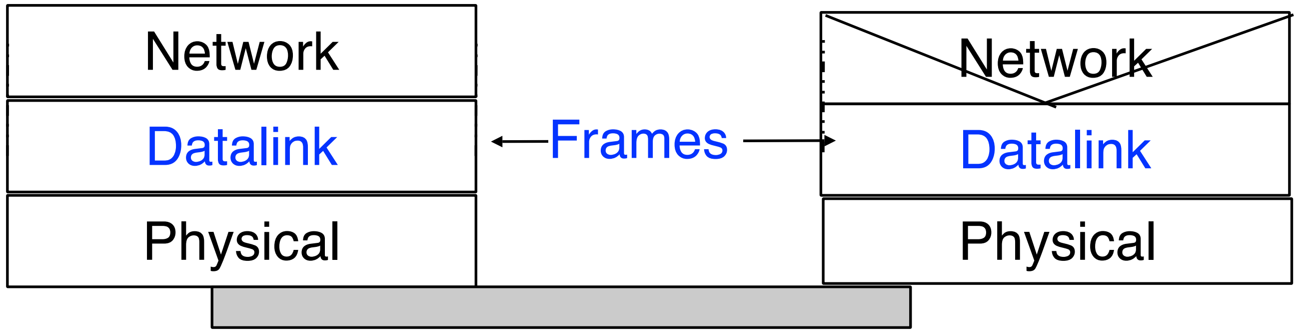
- WAN type datalink layer
  - PPP, HDLC
    - Reliable exchange of frames between two hosts attached to the same "link"
    - Mainly used by wide area networks



- LAN type datalink layer
- Ethernet, Token Ring, FDDI, WiFi, Wimax,
  - Exchange of frames between hosts attached to the same LAN
  - limited geographical coverage

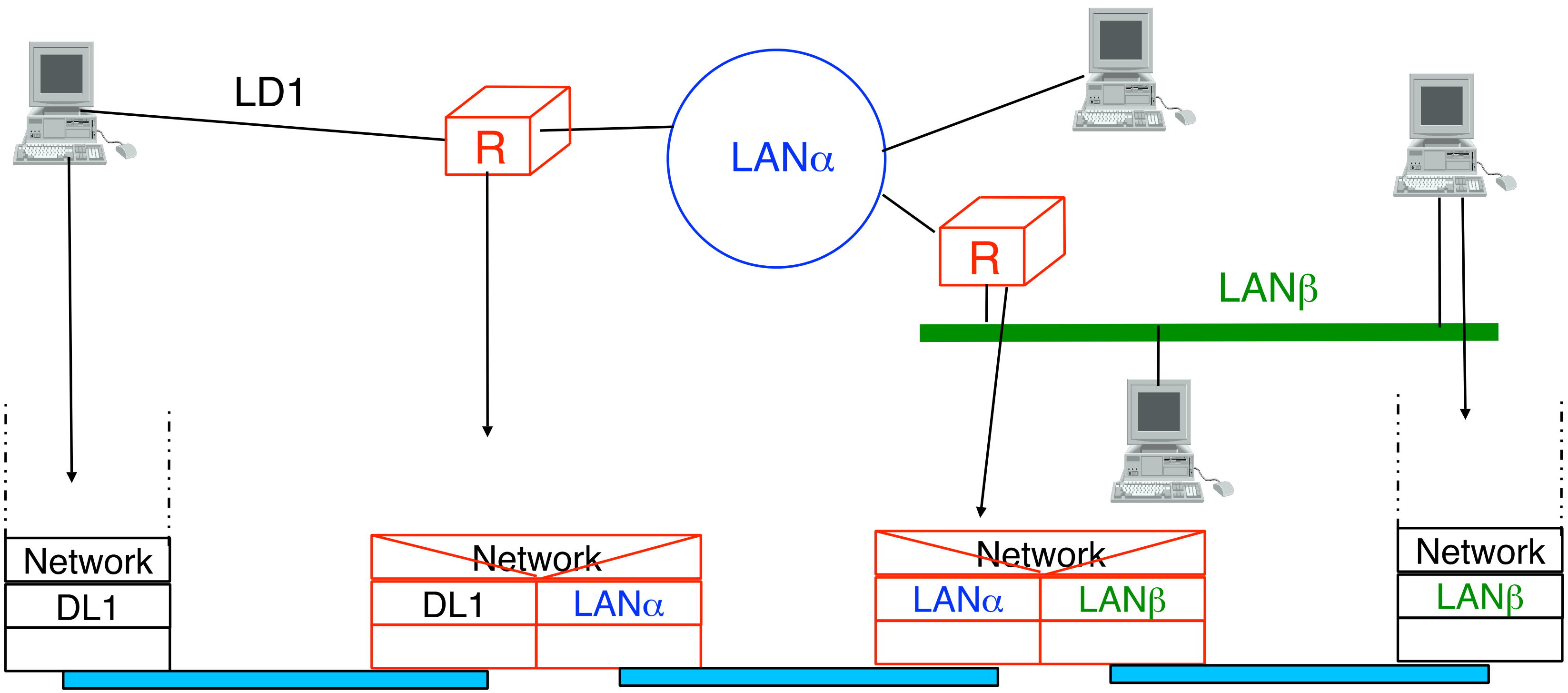


The datalink service



- Service of datalink layer
  - Unreliable connectionless service
    - Transmission of frames between hosts directly connected at the physical layer or directly attached to the same LAN
    - Unreliable transmission (frames can be lost but usually transmission errors are detected)
    - Most datalink layers have maximum frame length
  - Connection-oriented service, reliable or not
    - Transmission of frames between hosts directly connected at the physical layer or directly attached to the same LAN
    - Reliable or unreliable transmission

Routers



- Router
  - Relay within the network layer
- packet is unit of transmission

Network layer
Basic principles

Network layer
Basic principles

- Each host/router must be identified by a network layer address which is independent from its datalink layer address

Network layer
Basic principles

- Each host/router must be identified by a network layer address which is independent from its datalink layer address
- Network layer forwards packets from source to destination through multiple routers

Network layer
Basic principles

- Each host/router must be identified by a network layer address which is independent from its datalink layer address
- Network layer forwards packets from source to destination through multiple routers
- Network layer service must be completely independent from the service provided by the datalink layer

Network layer
Basic principles

- Each host/router must be identified by a network layer address which is independent from its datalink layer address
- Network layer forwards packets from source to destination through multiple routers
- Network layer service must be completely independent from the service provided by the datalink layer
- Network layer user should not need to know anything about the internal structure of the network layer to be able to send packets

Internal organisation of the network layer

Internal organisation of the network layer

- Two possible organisations
  - datagrams
  - virtual circuits

Internal organisation of the network layer

- Two possible organisations
  - datagrams
  - virtual circuits

- The internal organisation of the network is orthogonal to the service provided, but often
  - datagram mode is used to provide a connectionless service
  - virtual circuits are used to provide a connection-oriented service

Datagram transmission mode

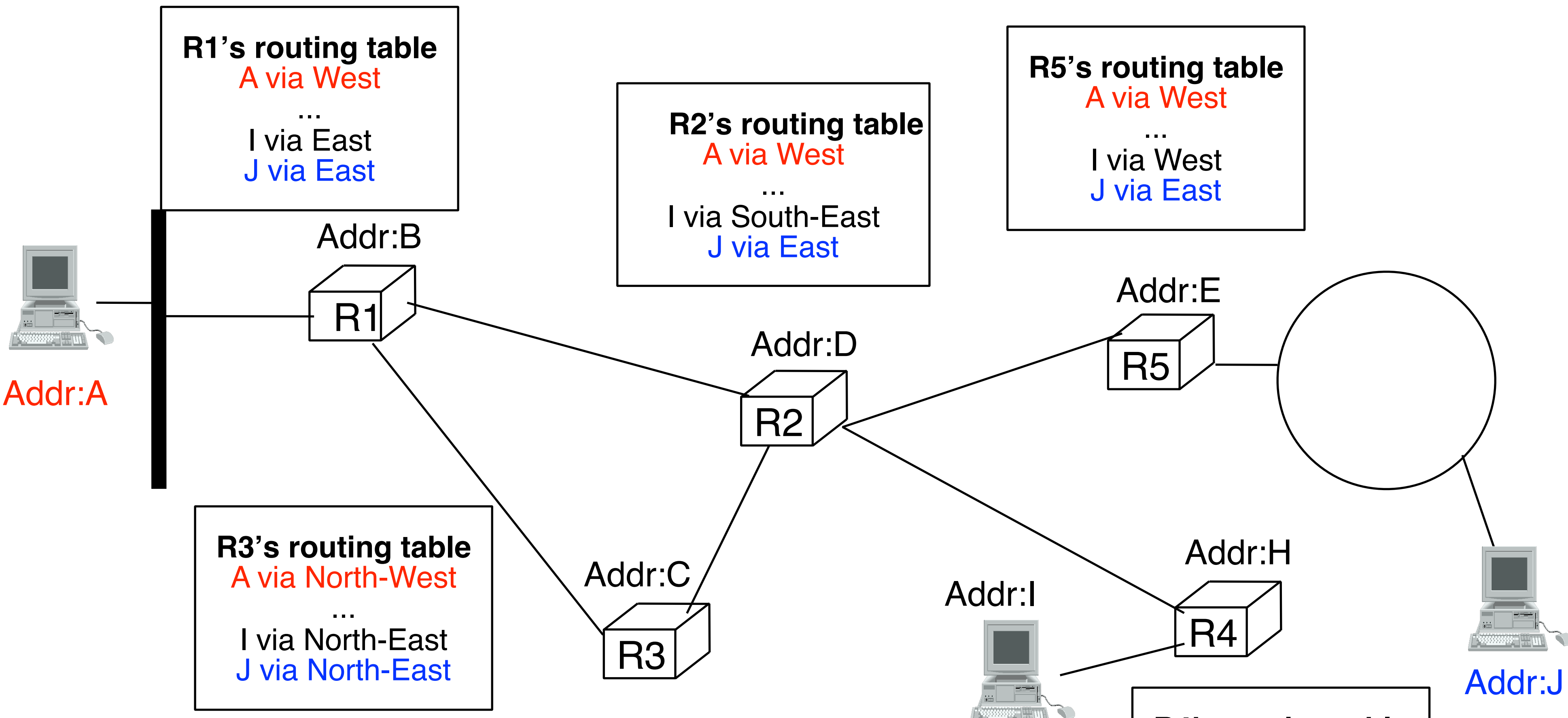
Datagram transmission mode

- Basics
  - Each route/host is identified by an address
  - Information is divided in packets
  - Each packet contains
    - Source address
    - Destination address
    - Payload
  - Router behavior
    - Upon packet arrival look at destination address and routing table to decide where the packet should be forwarded
      - hop-by-hop forwarding, each routers takes a forwarding decision

Datagram transmission mode

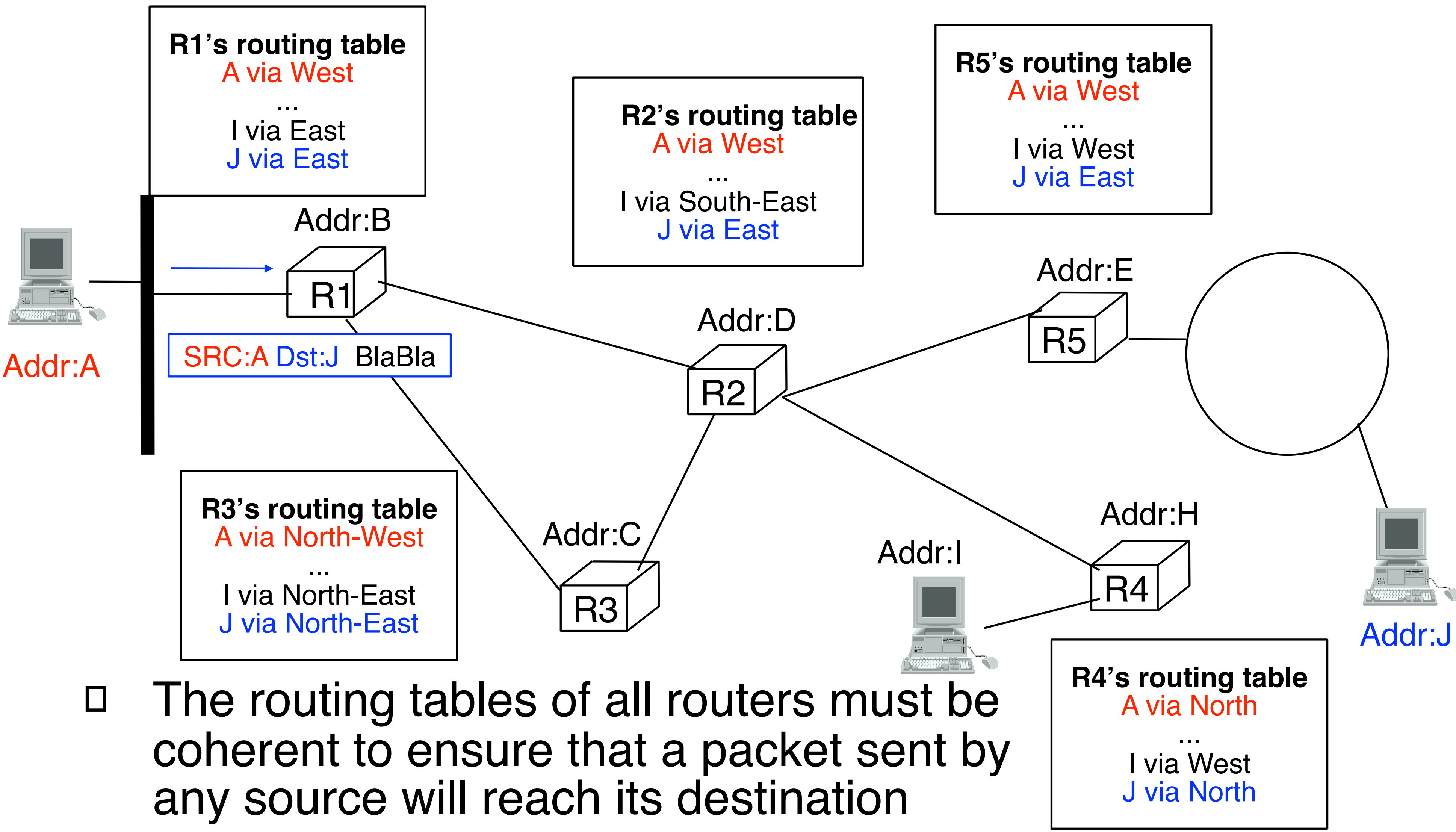
- Basics
  - Each route/host is identified by an address
  - Information is divided in packets
  - Each packet contains
    - Source address
    - Destination address
    - Payload
  - Router behavior
    - Upon packet arrival look at destination address and routing table to decide where the packet should be forwarded
      - hop-by-hop forwarding, each routers takes a forwarding decision
- Examples
  - IP (IPv4 and IPv6)
  - CLNP
  - IPX

Datagram transmission mode (2)



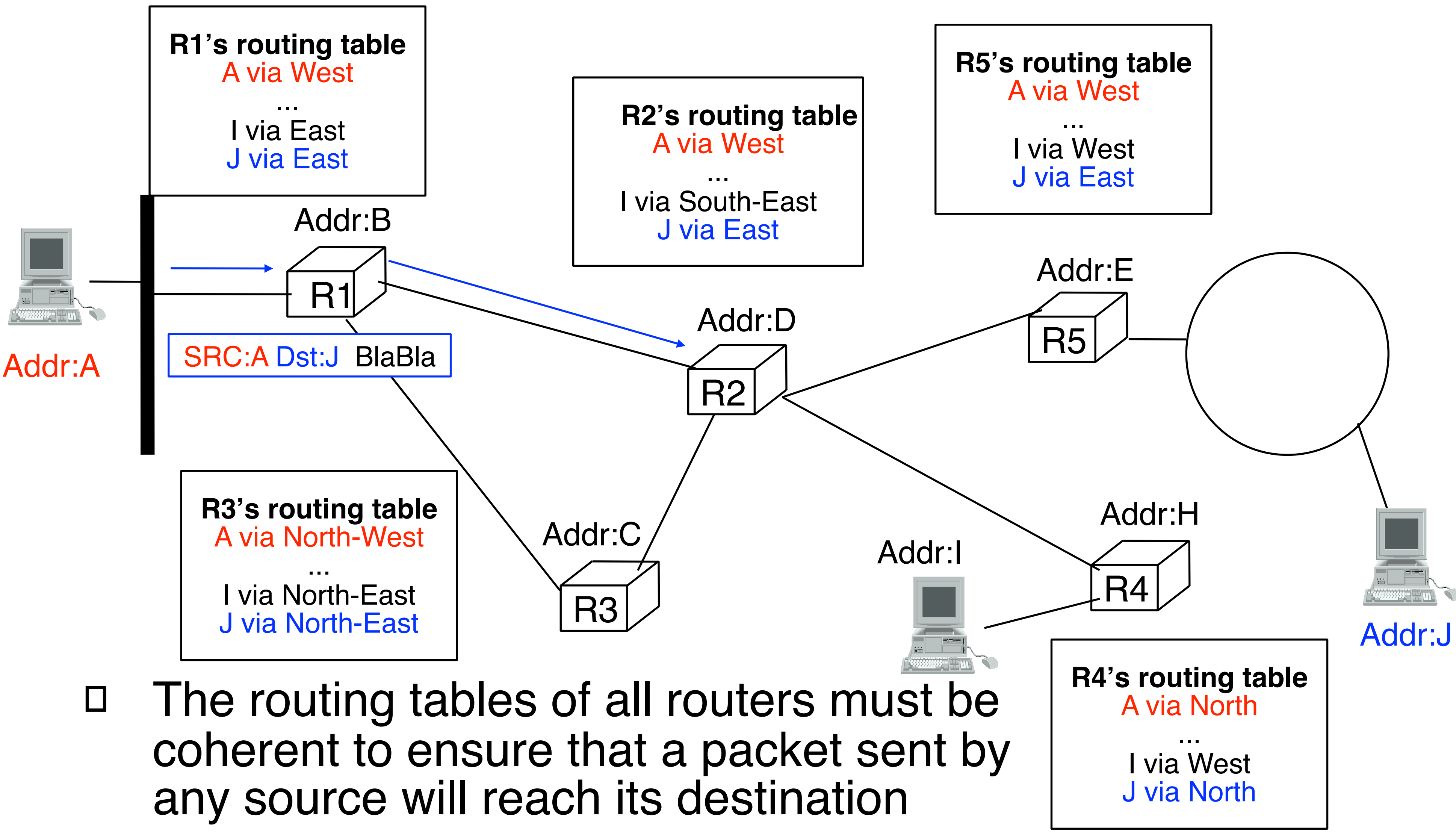
- The routing tables of all routers must be coherent to ensure that a packet sent by any source will reach its destination

Datagram transmission mode (2)



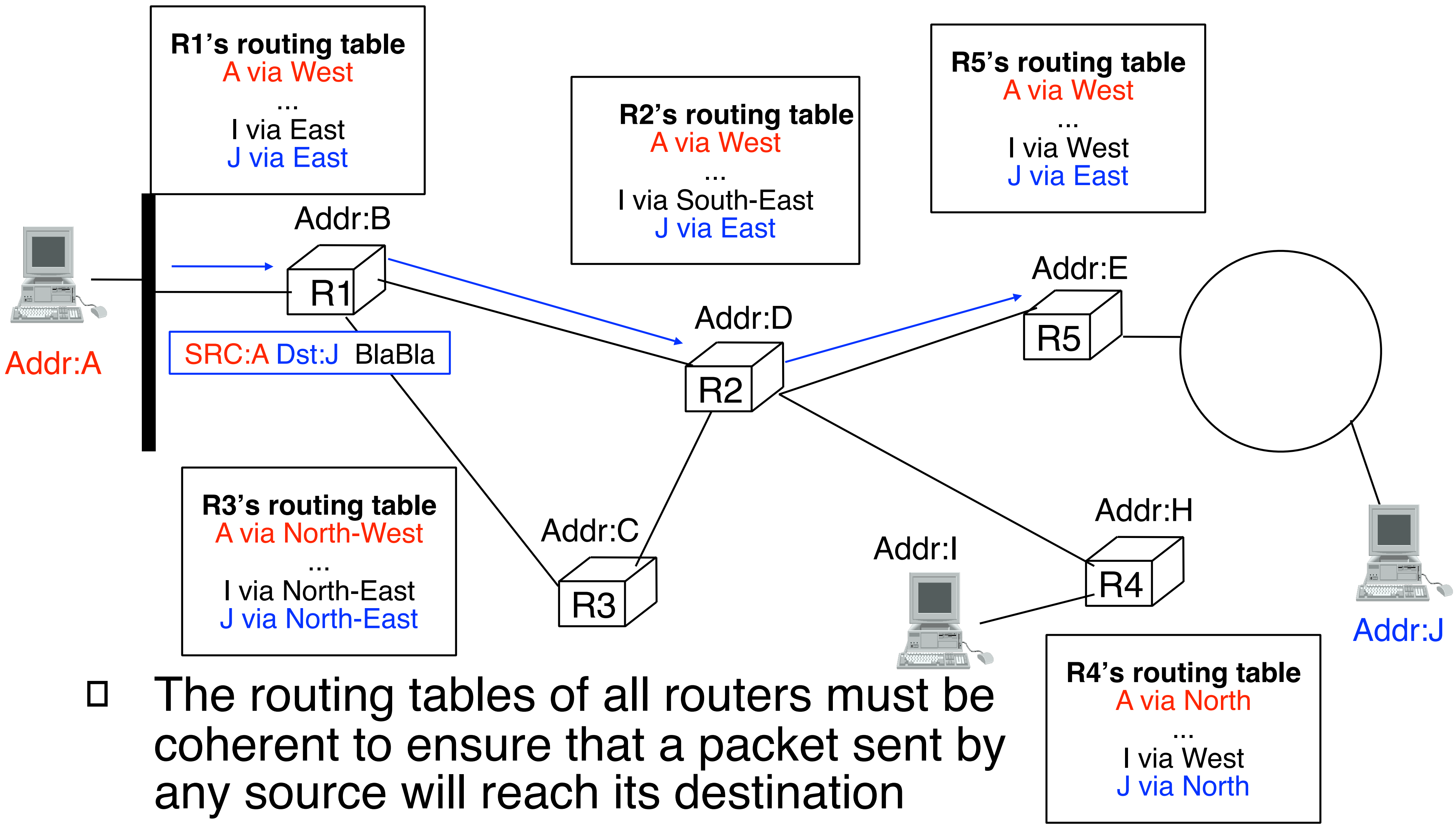
- The routing tables of all routers must be coherent to ensure that a packet sent by any source will reach its destination

Datagram transmission mode (2)



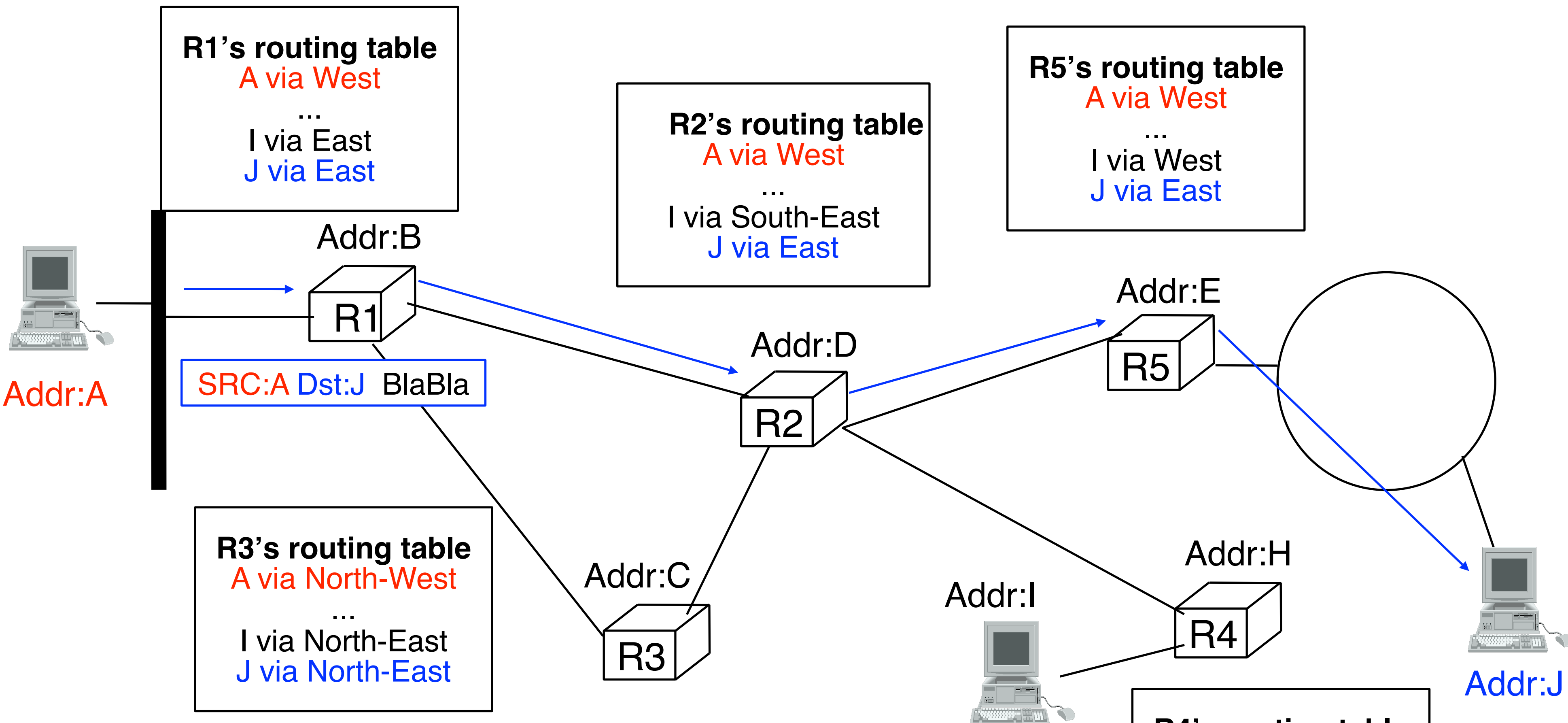
- The routing tables of all routers must be coherent to ensure that a packet sent by any source will reach its destination

Datagram transmission mode (2)

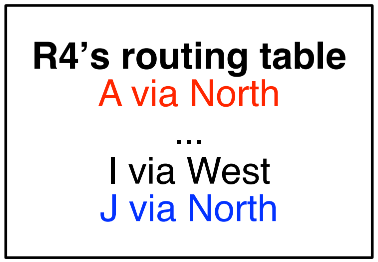


- The routing tables of all routers must be coherent to ensure that a packet sent by any source will reach its destination

Datagram transmission mode (2)



- The routing tables of all routers must be coherent to ensure that a packet sent by any source will reach its destination



Virtual circuit organisation

Virtual circuit organisation

- Goals
  - Keep forwarding on the routers as simple as possible
    - consulting a routing table for each packet is costly from a performance viewpoint

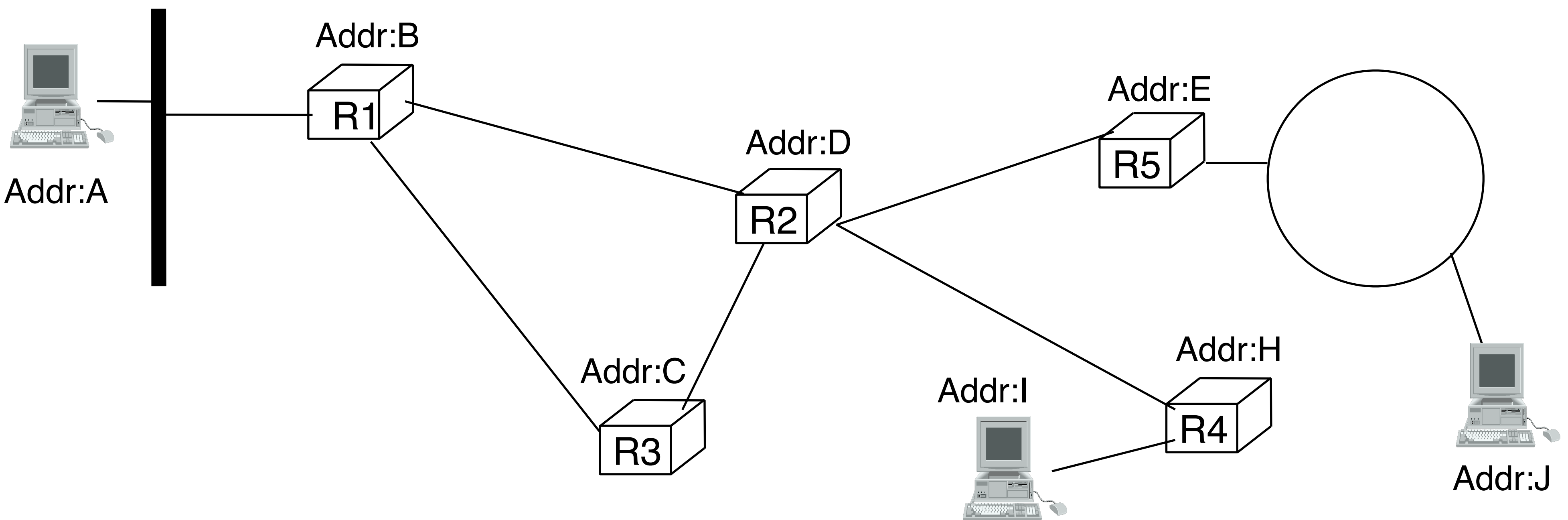
Virtual circuit organisation

- Goals
  - Keep forwarding on the routers as simple as possible
    - consulting a routing table for each packet is costly from a performance viewpoint
- Solution
  - Before transmitting packets containing data, create a virtual circuit that links source and destination through the network
    - During the virtual circuit establishment, efficient datastructures are updated on each transit router to simplify forwarding
  - Use the virtual circuits to forward the packets
    - All packets will follow the same path

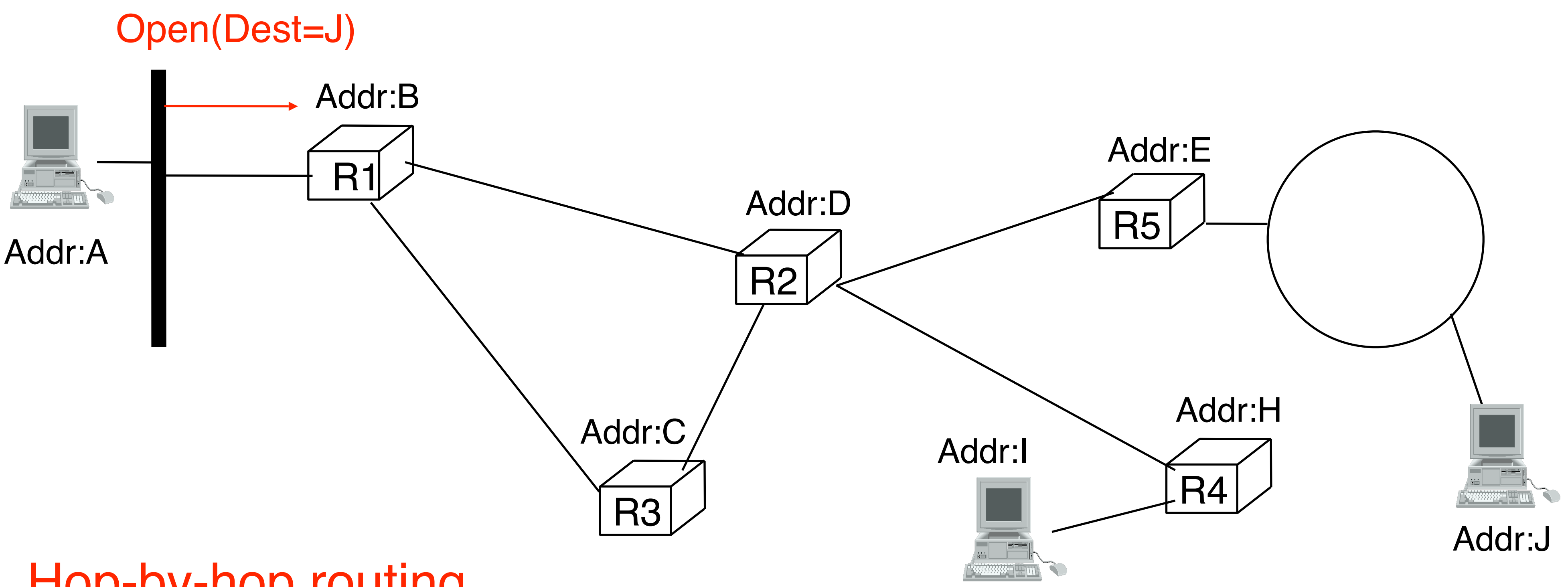
Virtual circuit organisation

- Goals
  - Keep forwarding on the routers as simple as possible
    - consulting a routing table for each packet is costly from a performance viewpoint
- Solution
  - Before transmitting packets containing data, create a virtual circuit that links source and destination through the network
    - During the virtual circuit establishment, efficient datastructures are updated on each transit router to simplify forwarding
  - Use the virtual circuits to forward the packets
    - All packets will follow the same path
- Example
  - ATM, X.25, Frame Relay, MPLS, gMPLS

Establishment of a virtual circuit



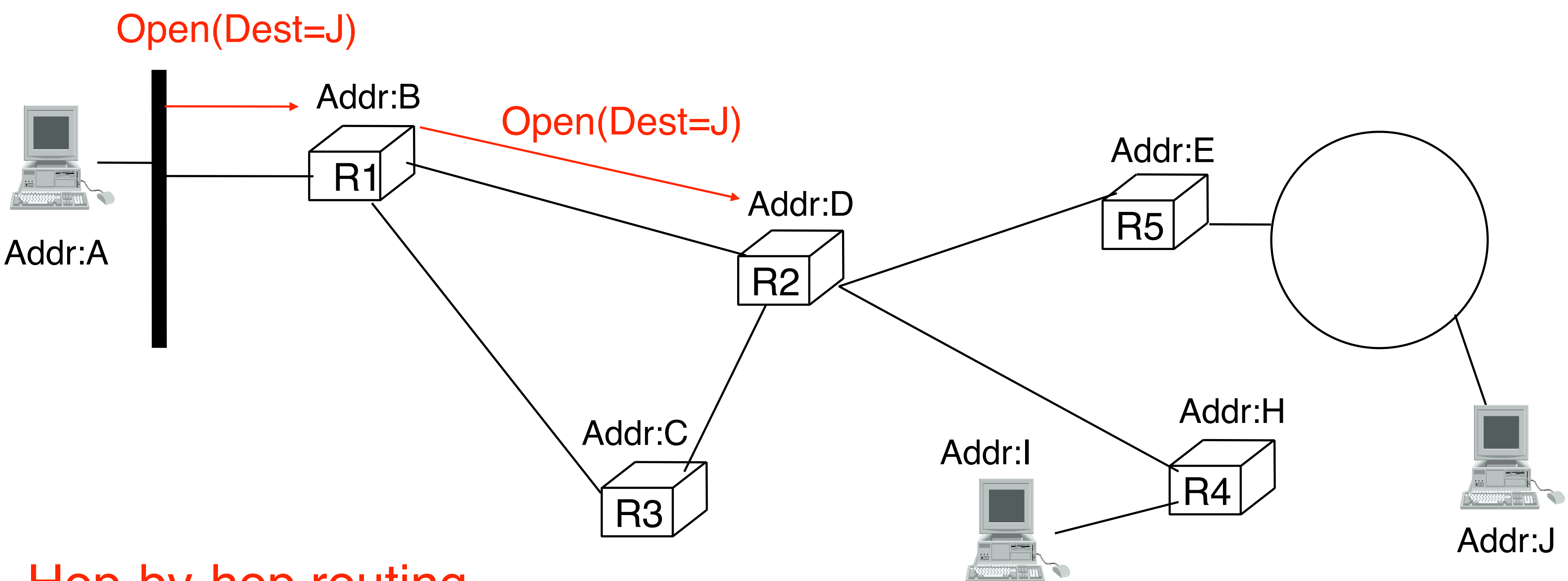
Establishment of a virtual circuit



Hop-by-hop routing

Each router consults its routing table to forward vc establishment

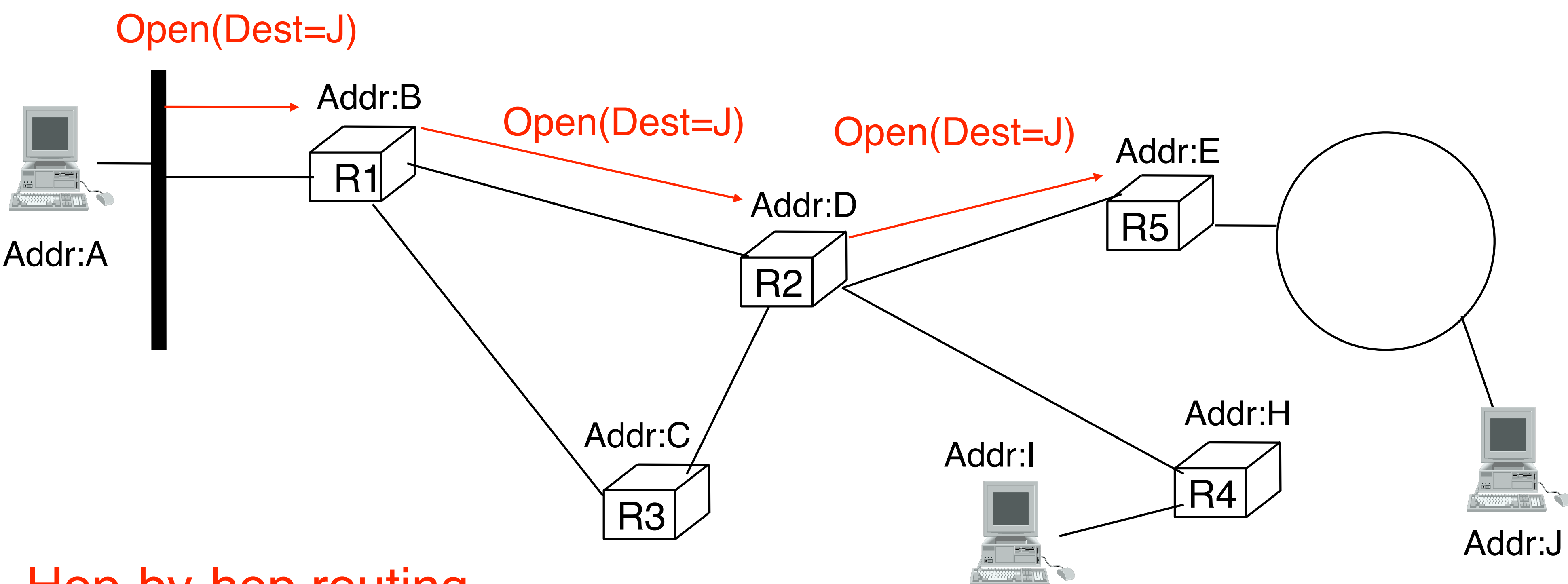
Establishment of a virtual circuit



Hop-by-hop routing

Each router consults its routing table to forward vc establishment

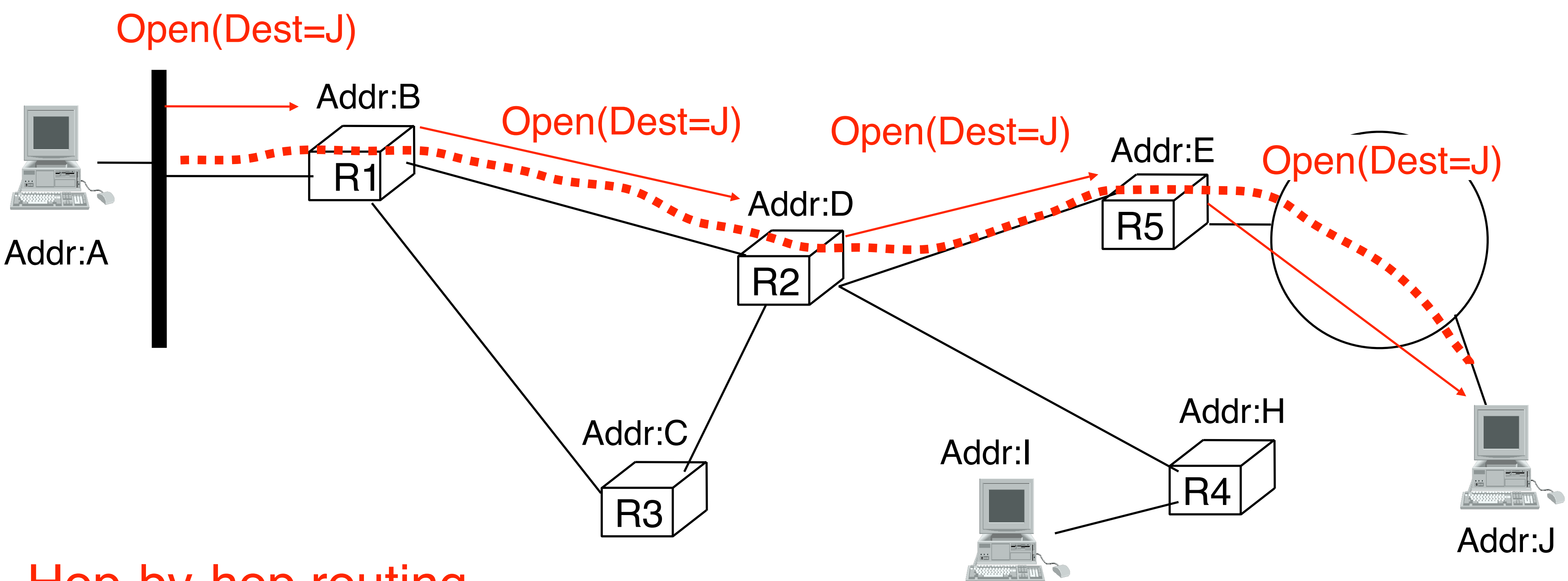
Establishment of a virtual circuit



Hop-by-hop routing

Each router consults its routing table to forward vc establishment

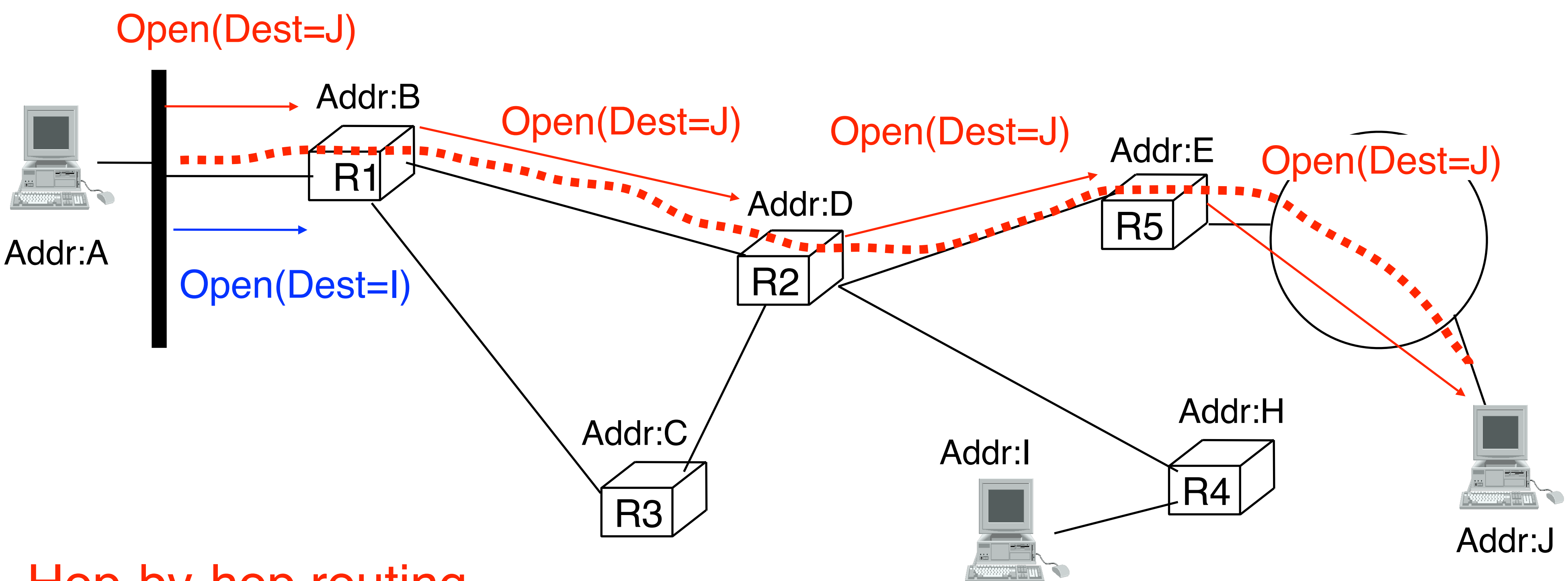
Establishment of a virtual circuit



Hop-by-hop routing

Each router consults its routing table to forward vc establishment

Establishment of a virtual circuit



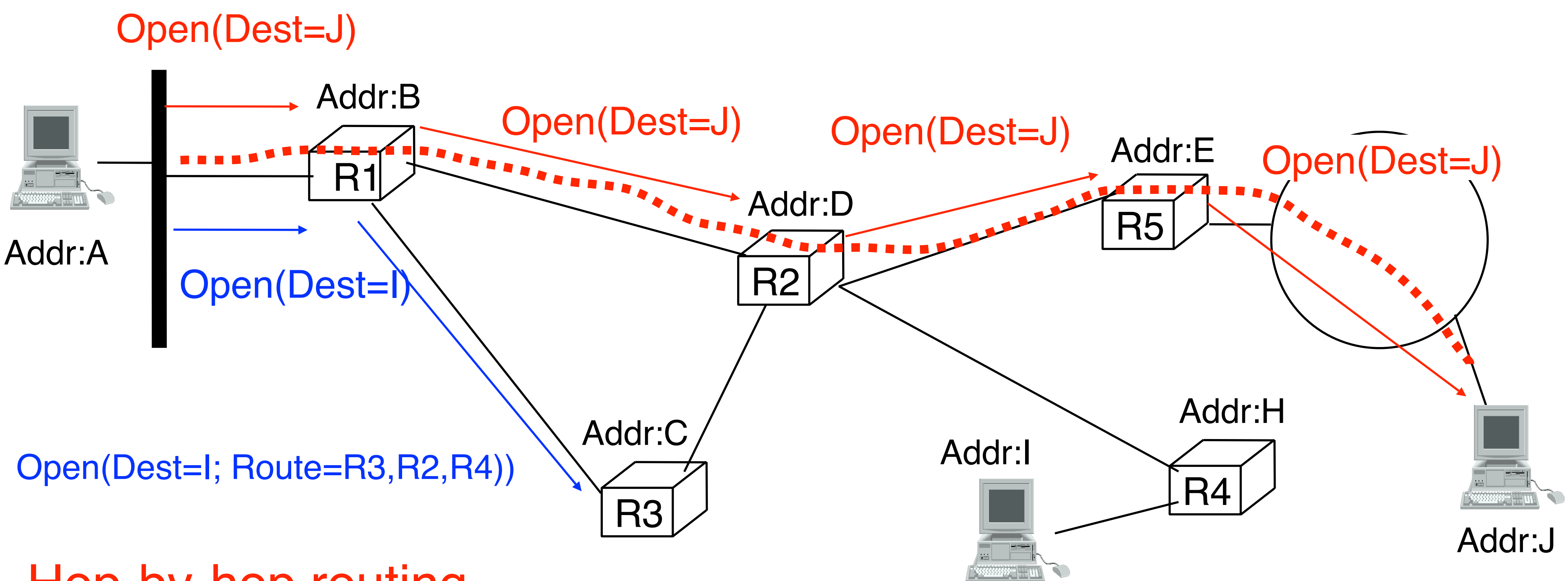
Hop-by-hop routing

Each router consults its routing table to forward vc establishment

Source routing/ explicit routing

Source (or first hop router) indicates in vc establishment packet the path to be followed

Establishment of a virtual circuit



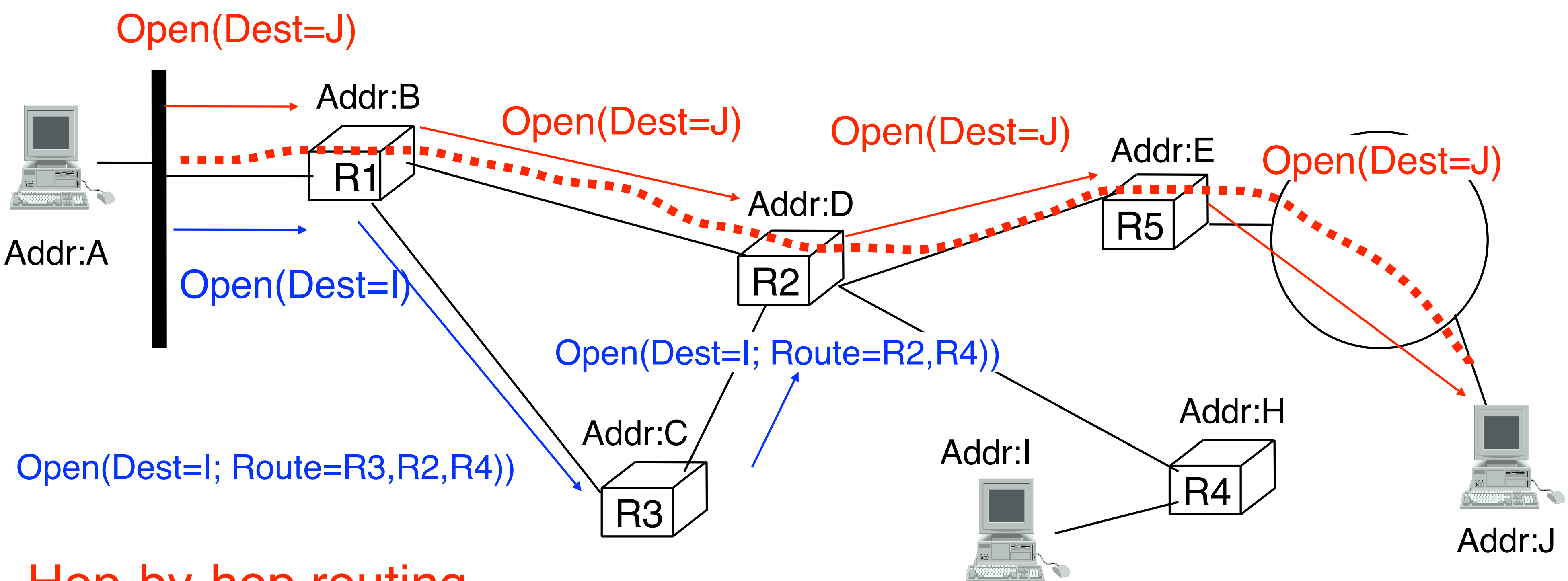
Hop-by-hop routing

Each router consults its routing table to forward vc establishment

Source routing/ explicit routing

Source (or first hop router) indicates in vc establishment packet the path to be followed

Establishment of a virtual circuit



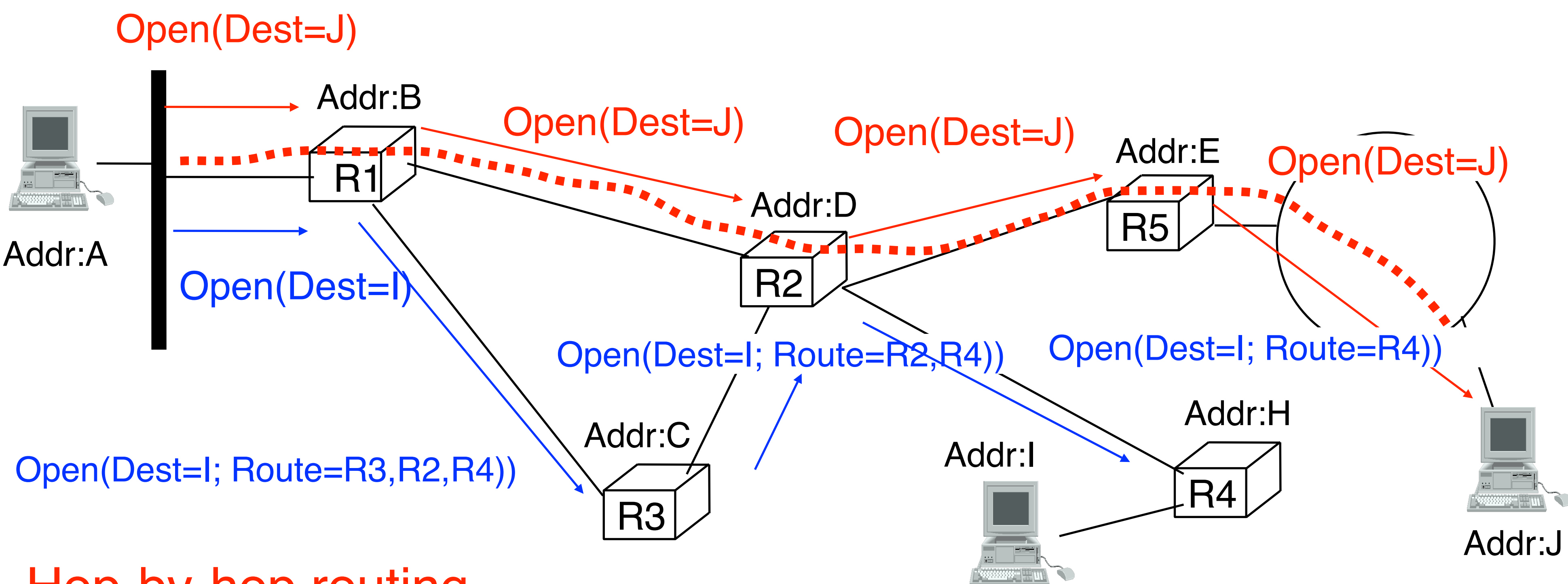
Hop-by-hop routing

Each router consults its routing table to forward vc establishment

Source routing/ explicit routing

Source (or first hop router) indicates in vc establishment packet the path to be followed

Establishment of a virtual circuit



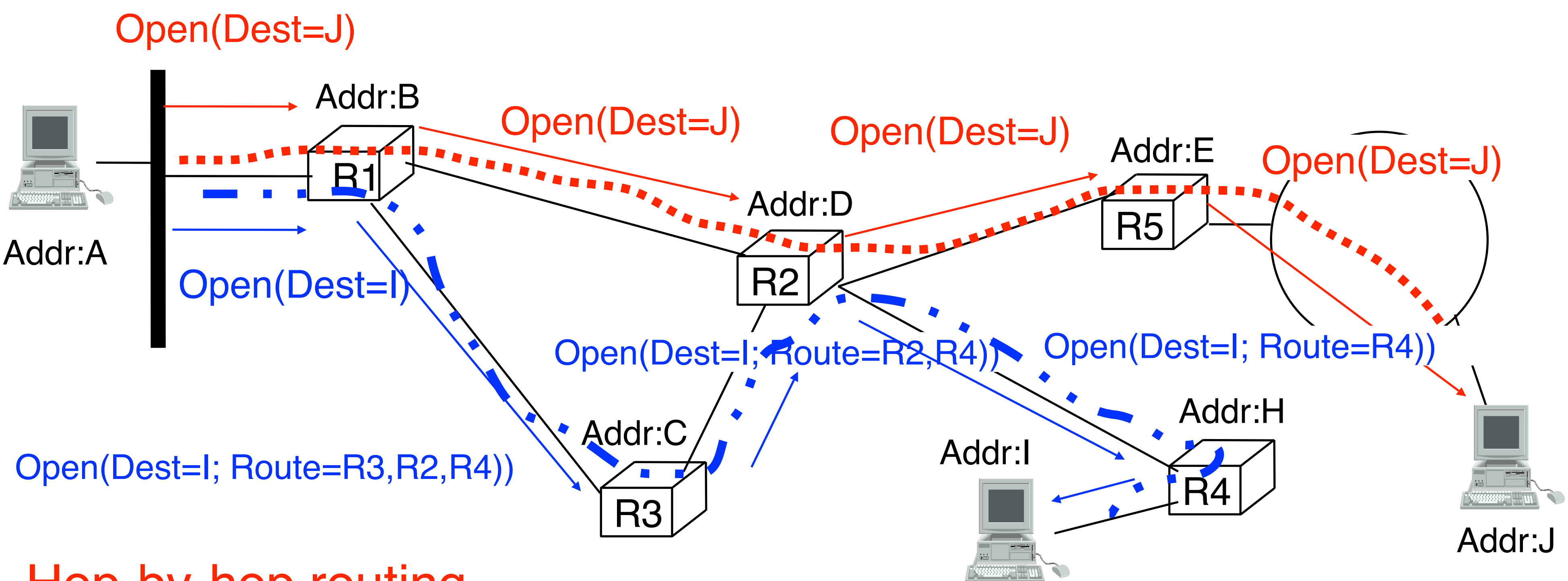
Hop-by-hop routing

Each router consults its routing table to forward vc establishment

Source routing/ explicit routing

Source (or first hop router) indicates in vc establishment packet the path to be followed

Establishment of a virtual circuit



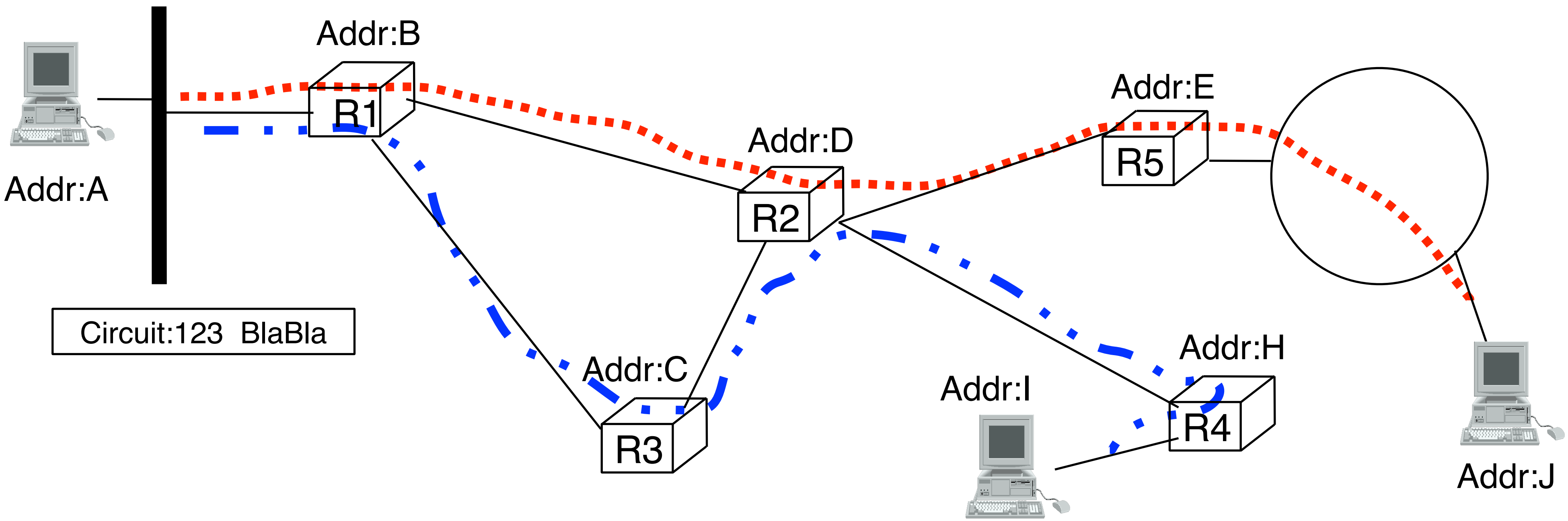
Hop-by-hop routing

Each router consults its routing table to forward vc establishment

Source routing/ explicit routing

Source (or first hop router) indicates in vc establishment packet the path to be followed

Packet transmission



- Packet contents
  - virtual circuit identifier
  - packet payload
- What kind of virtual circuit identifier
  - Naive solution
    - unique identifier for all virtual circuits inside network
      - How to coordinate allocation of vc identifiers ?

Packet transmission (2)

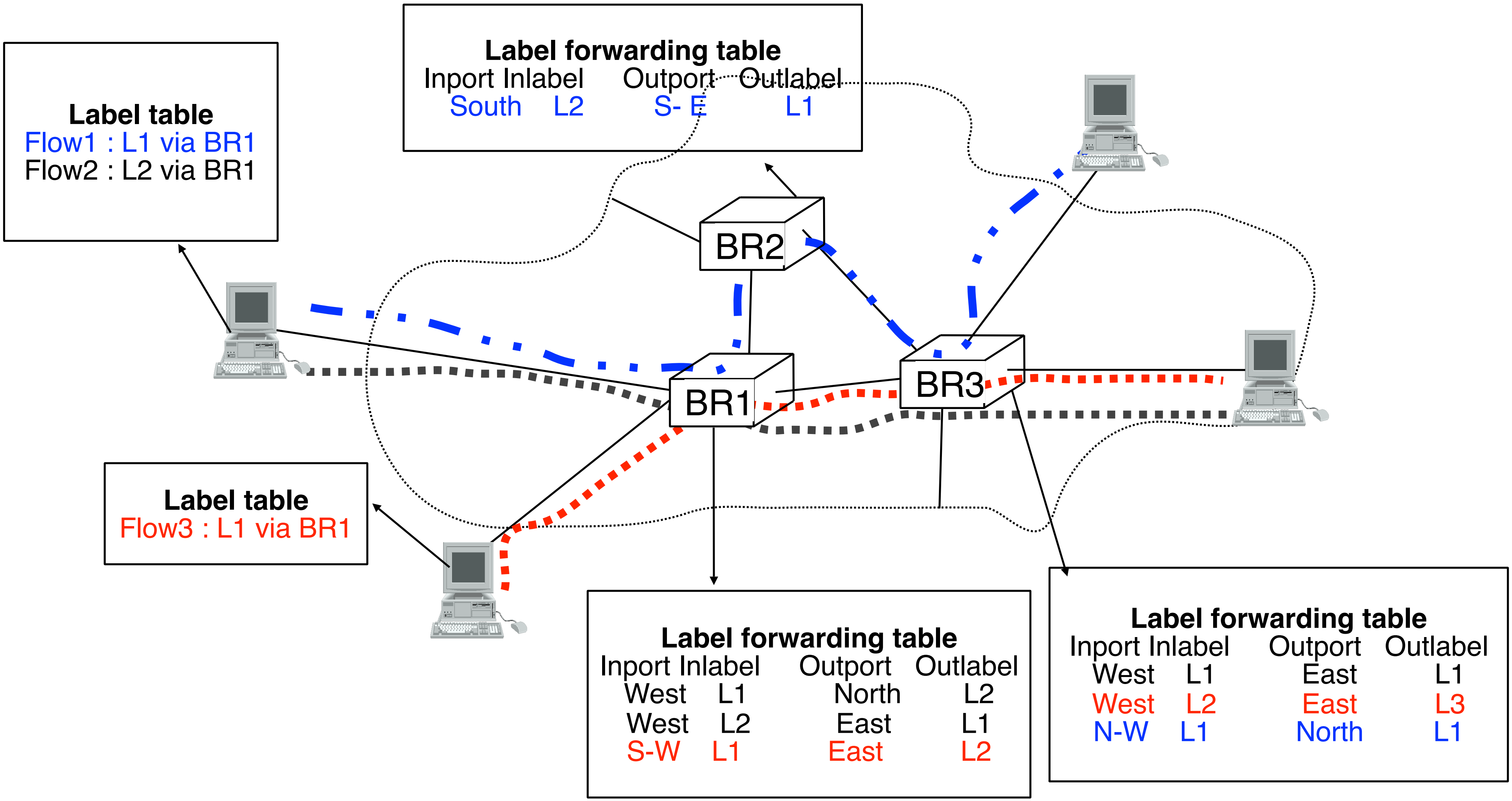
- How unique should virtual circuits identifiers be ?
  - globally unique
    - unrealistic
  - unique inside a given network
    - then coordination among routers is necessary
  - unique on a given link
    - easier to manage, no coordination required, but
    - virtual circuit identifier may need to be changed from link to link

- How to update the virtual circuit identifier of packets
  - All routers must contain a label forwarding table
    - this table is updated every time a virtual circuit is established

Label forwarding table

Inport	Inlabel	Outport	Outlabel
West	L1	East	L1
West	L2	East	L3
N-W	L1	North	L1

Virtual circuits : example



Network layer

- Basics
- Routing
  - Static routing
  - Distance vector routing
  - Link state routing
- IP : Internet Protocol
- Routing in IP networks

Routing and Forwarding

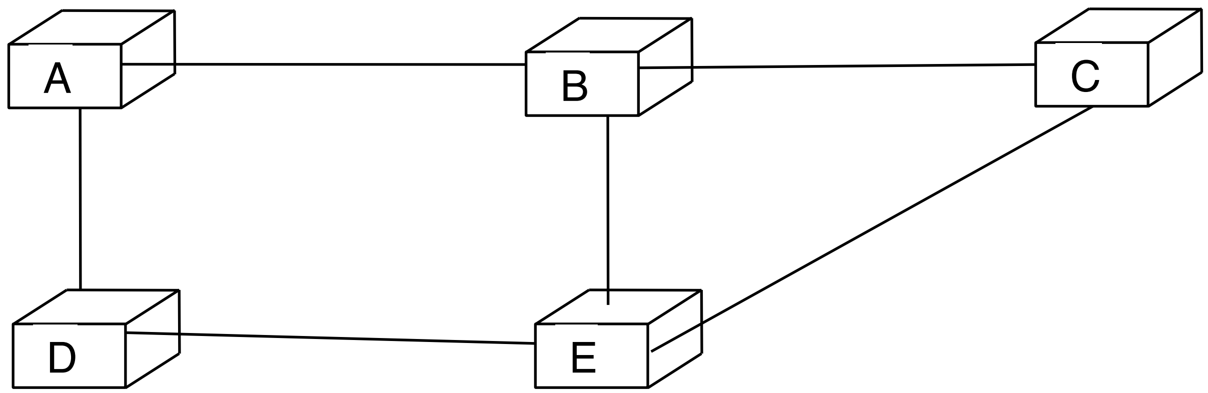
- Main objective of network layer
  - transport packets form source to destination

- Two mechanisms are used in network layer
  - forwarding
    - algorithm use by each router to determine on which interface each packet should be sent to reach its destination or follow its virtual circuit
      - relies on the routing table maintained by each router

  - routing
    - algorithm (usually distributed) that distributes to all routers the information that allows them to build their routing tables

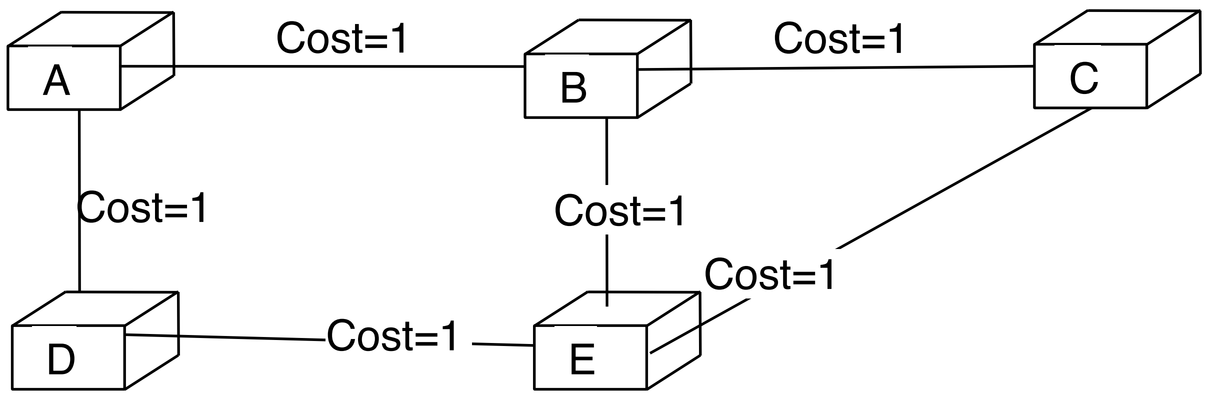
Routing (2)

- How to build the routing tables of each router ?



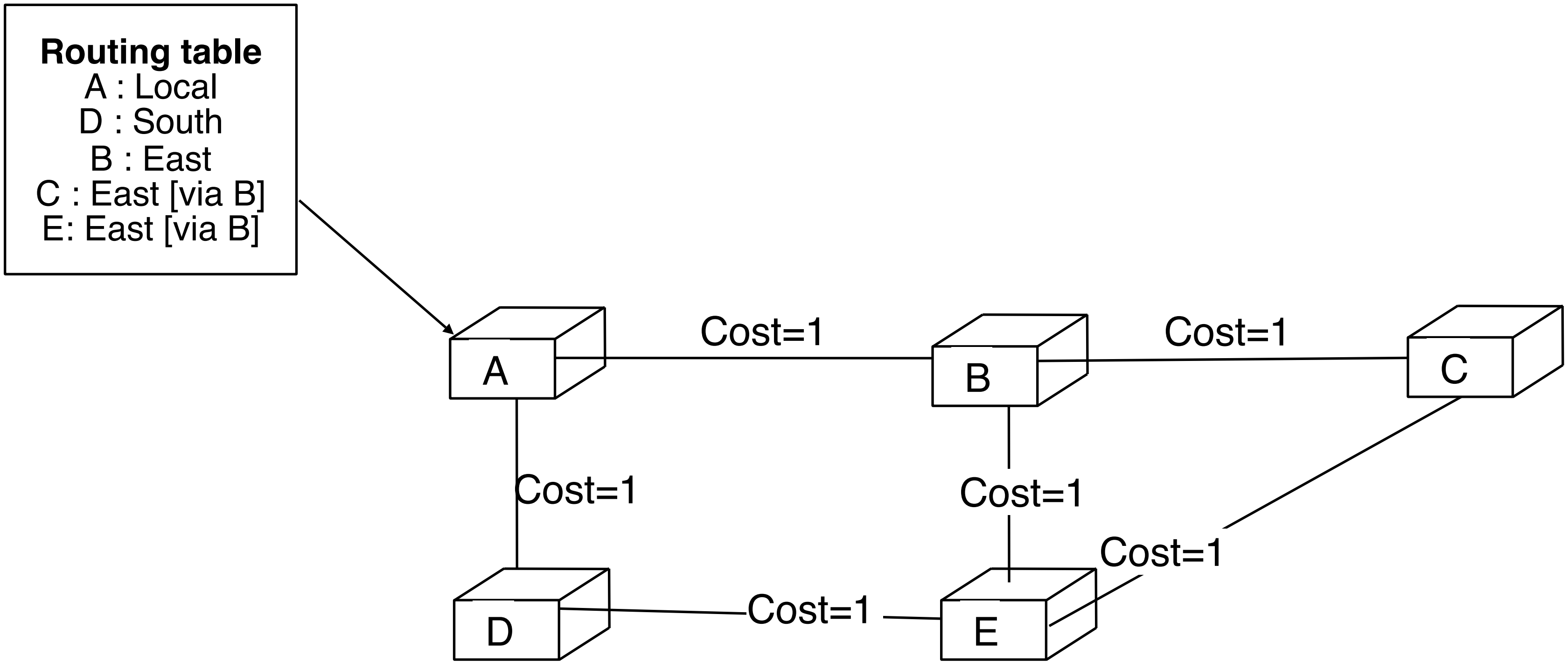
- Principle
  - Include in the routing table of each router the path to allow it to reach each destination
    - Which path to be included in the routing table
      - From A to C ?
      - From D to B

Selection of the shortest paths



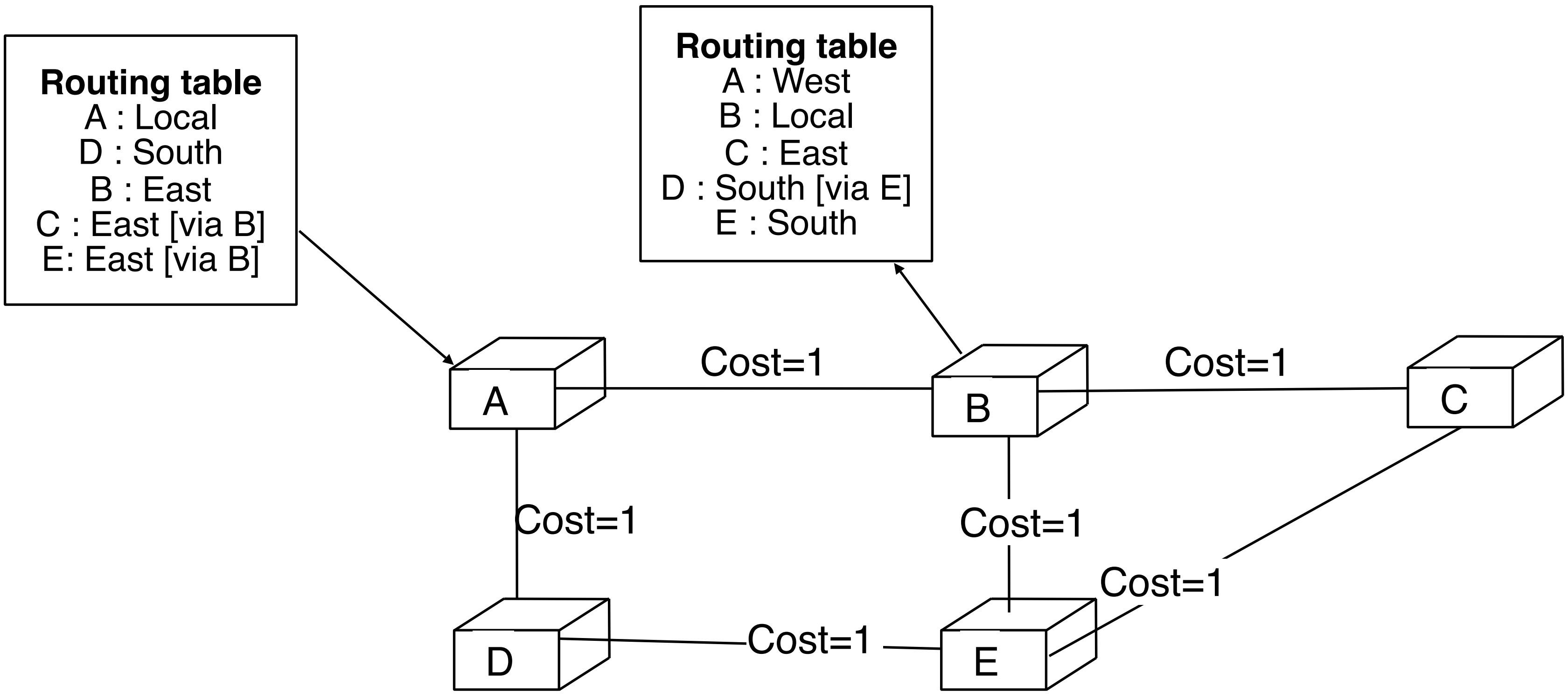
- Principle
  - Associate a weight/cost to each link
  - Each router chooses the lowest cost path
    - How to ensure that the routing tables of all routers are coherent ?

Selection of the shortest paths



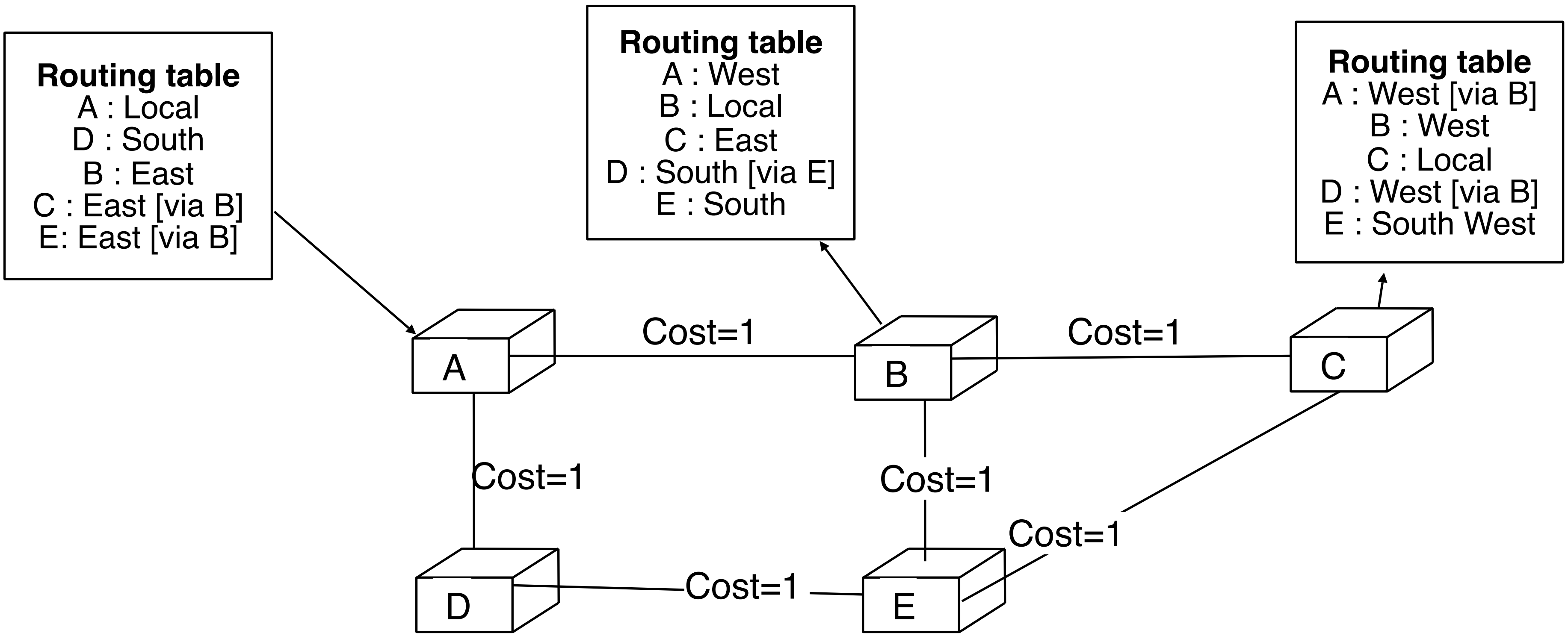
- Principle
  - Associate a weight/cost to each link
  - Each router chooses the lowest cost path
    - How to ensure that the routing tables of all routers are coherent ?

Selection of the shortest paths



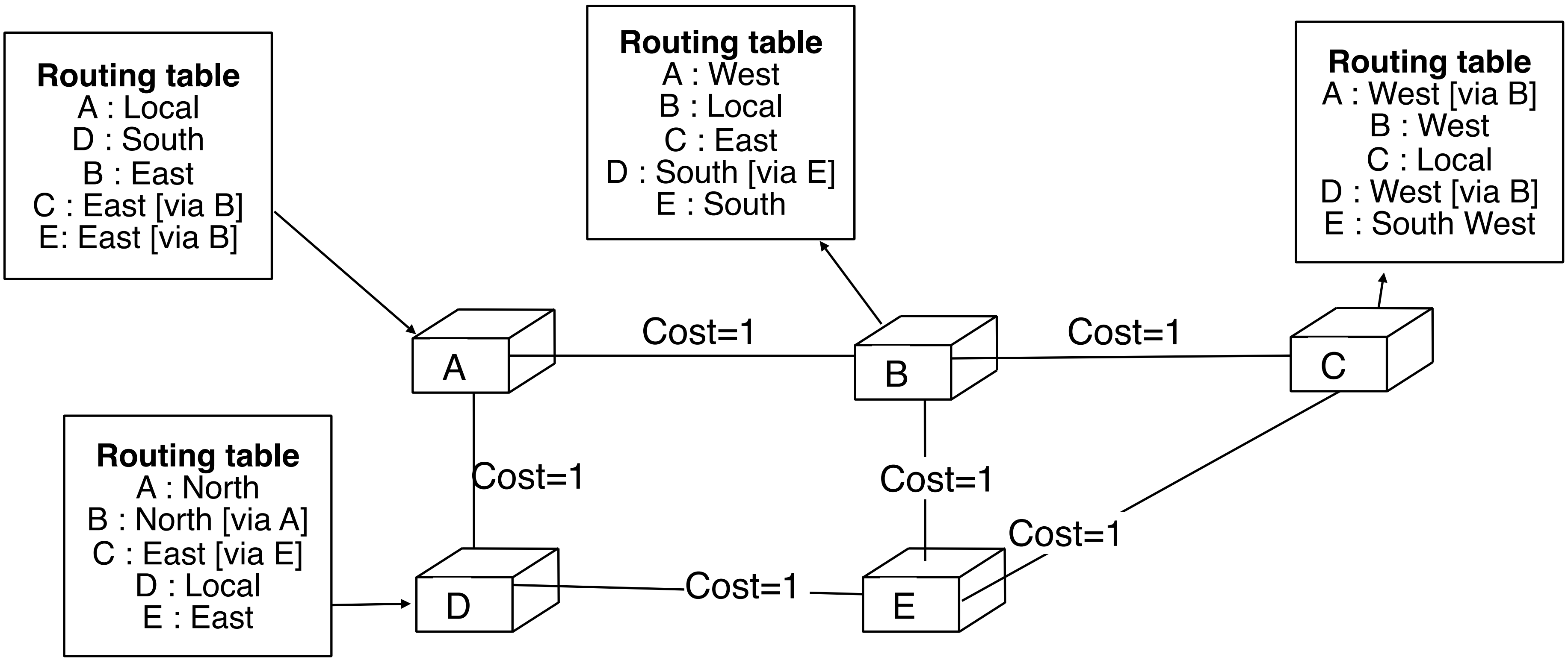
- Principle
  - Associate a weight/cost to each link
  - Each router chooses the lowest cost path
    - How to ensure that the routing tables of all routers are coherent ?

Selection of the shortest paths



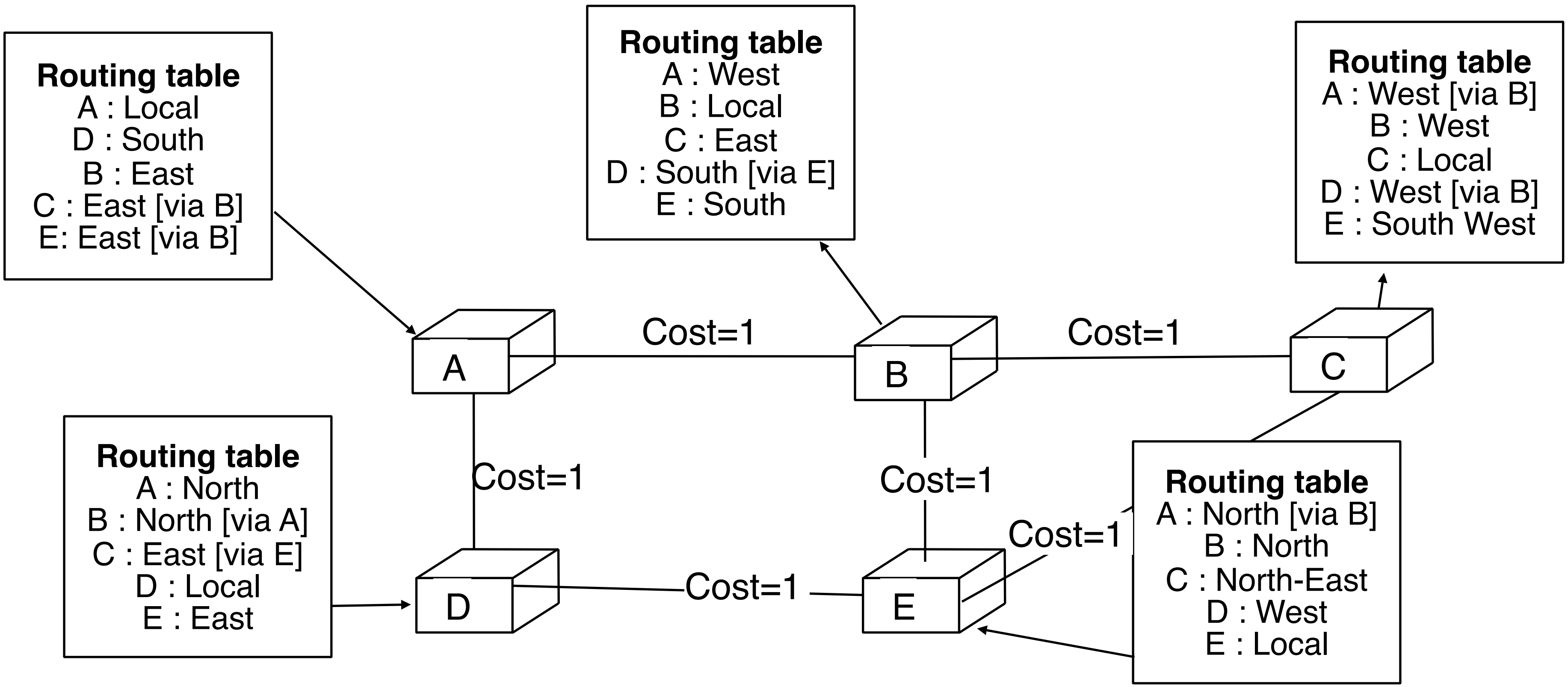
- Principle
  - Associate a weight/cost to each link
  - Each router chooses the lowest cost path
    - How to ensure that the routing tables of all routers are coherent ?

Selection of the shortest paths



- Principle
  - Associate a weight/cost to each link
  - Each router chooses the lowest cost path
    - How to ensure that the routing tables of all routers are coherent ?

Selection of the shortest paths



- Principle
  - Associate a weight/cost to each link
  - Each router chooses the lowest cost path
    - How to ensure that the routing tables of all routers are coherent ?

Static Routing

- Principle
  - Network manager or network management station computes all routing tables and downloads them on all routers
  - How to compute routing tables ?
    - shortest path algorithms
    - more complex algorithms to provide load balancing or traffic engineering

  - Advantages of static routing
    - Easy to use in a small network
    - routing tables can be optimised

  - Drawbacks of static routing
    - does not adapt dynamically to network load
    - how to deal with link and router failures ?

Dynamic or distributed routing

- Principle
  - routers exchange messages and use a distributed algorithm to build their routing tables
  - used in almost all networks

- Advantages
  - can easily adapt routing tables to events
- Drawbacks
  - more complex to implement than static routing

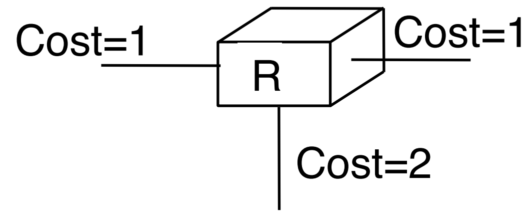
- Most common distributed routing methods
  - Distance vector routing
  - Link state routing

Network layer

- Basics
- Routing
  - Static routing
  - Distance vector routing
  - Link state routing
- IP : Internet Protocol
- Routing in IP networks

Distance vector routing

- Basic principles
  - Configuration of each router
    - Cost of each link



  - When it boots, a router only knows itself
  - Each router sends periodically to all its neighbours a vector that contains <u>for each destination that it knows</u>
    1. Destination address
    2. Distance between transmitting router and destination
       - distance vector is a summary of the router's routing table
  - Each router will update its routing table based on the information received from its neighbours

Distance vector routing (2)

- Routing table *maintained by router*
  - For each destination d inside routing table
    - R[d].cost = total cost of shortest path to reach d
    - R[d].link = outgoing link used to reach d via shortest path
- Distance vector *sent to neighbours*
  - For each destination d
    - V[d].cost = total cost of shortest path to reach d

```
Every N seconds:
 Vector=null;
 for each destination=d in R[]
 {
  Vector=Vector+Pair(d,R[d].cost);
 }
 for each interface
 {
  Send(Vector);
 }
```

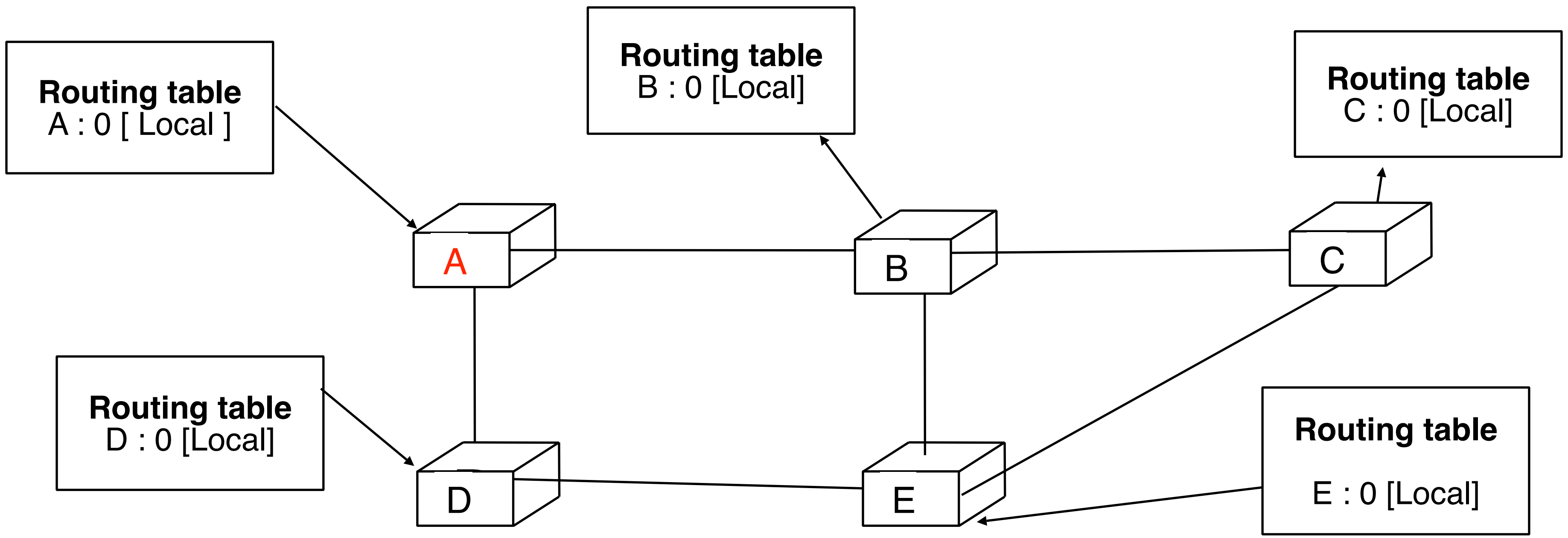
Distance vector routing (3)

- Processing of received distance vectors

```
Received(Vector V[],link l)
{ /* received vector from link l */
for each destination=d in V[]
{
  if (d isin R[])
  { if ((V[d].cost+l.cost) < R[d].cost)
     { /* shorter path */
       R[d].cost=V[d].cost+l.cost;
       R[d].link=l;
     }
   }
   else
   { /* new route */
     R[d].cost=V[d].cost+l.cost;
     R[d].link=l;
   }
}
```

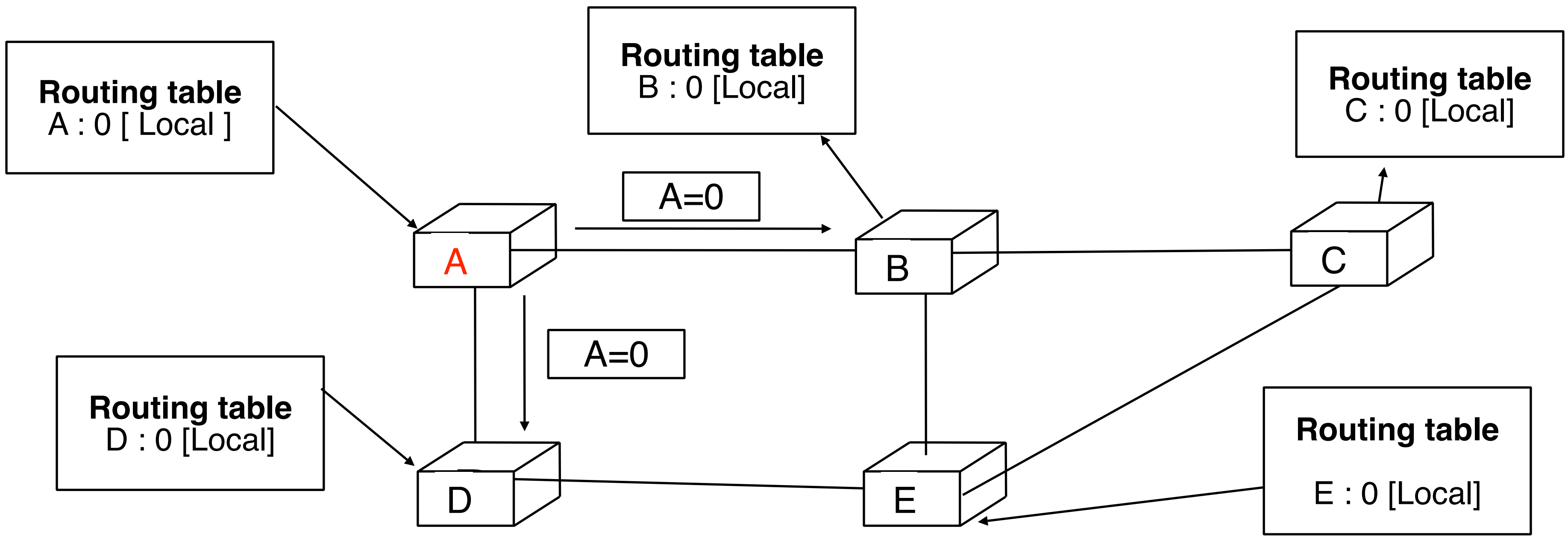
Distance vectors example

- All links have a unit cost



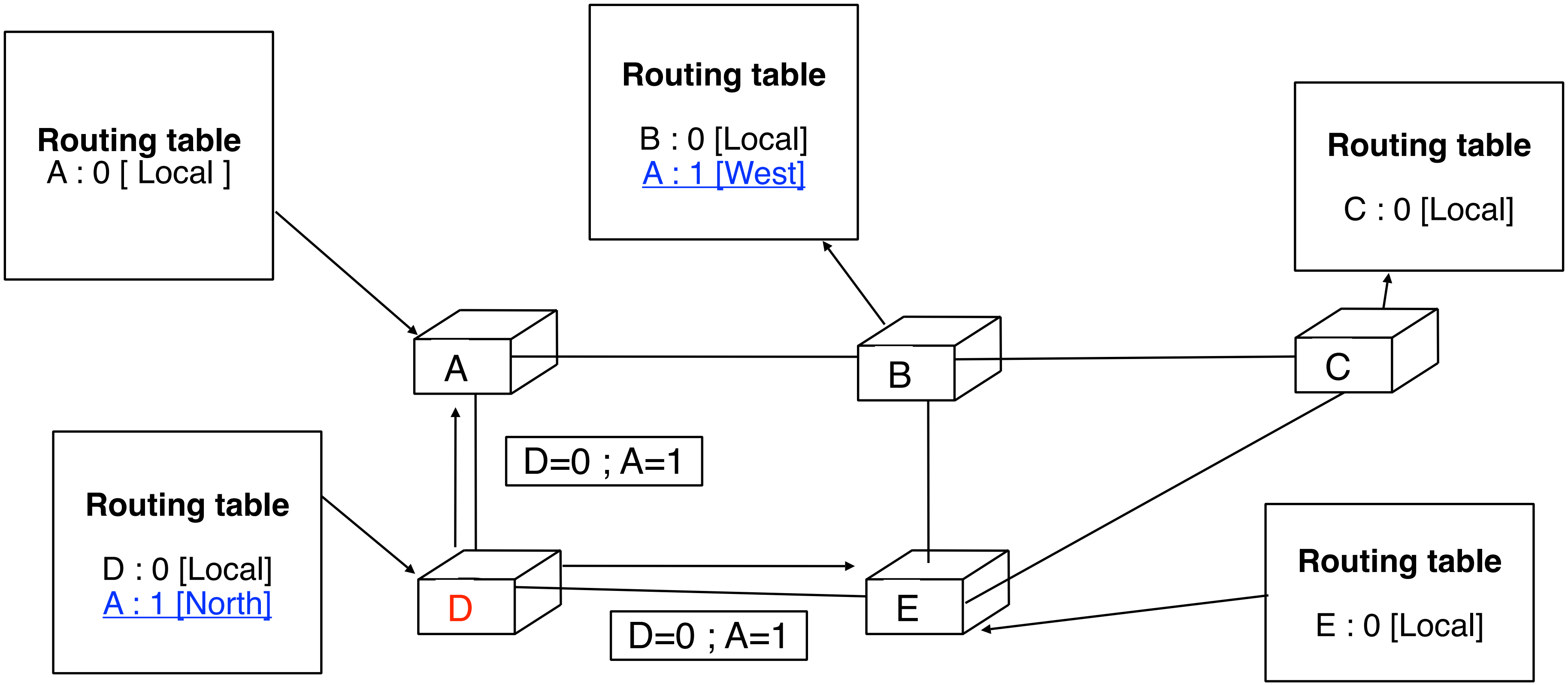
Distance vectors example

- All links have a unit cost

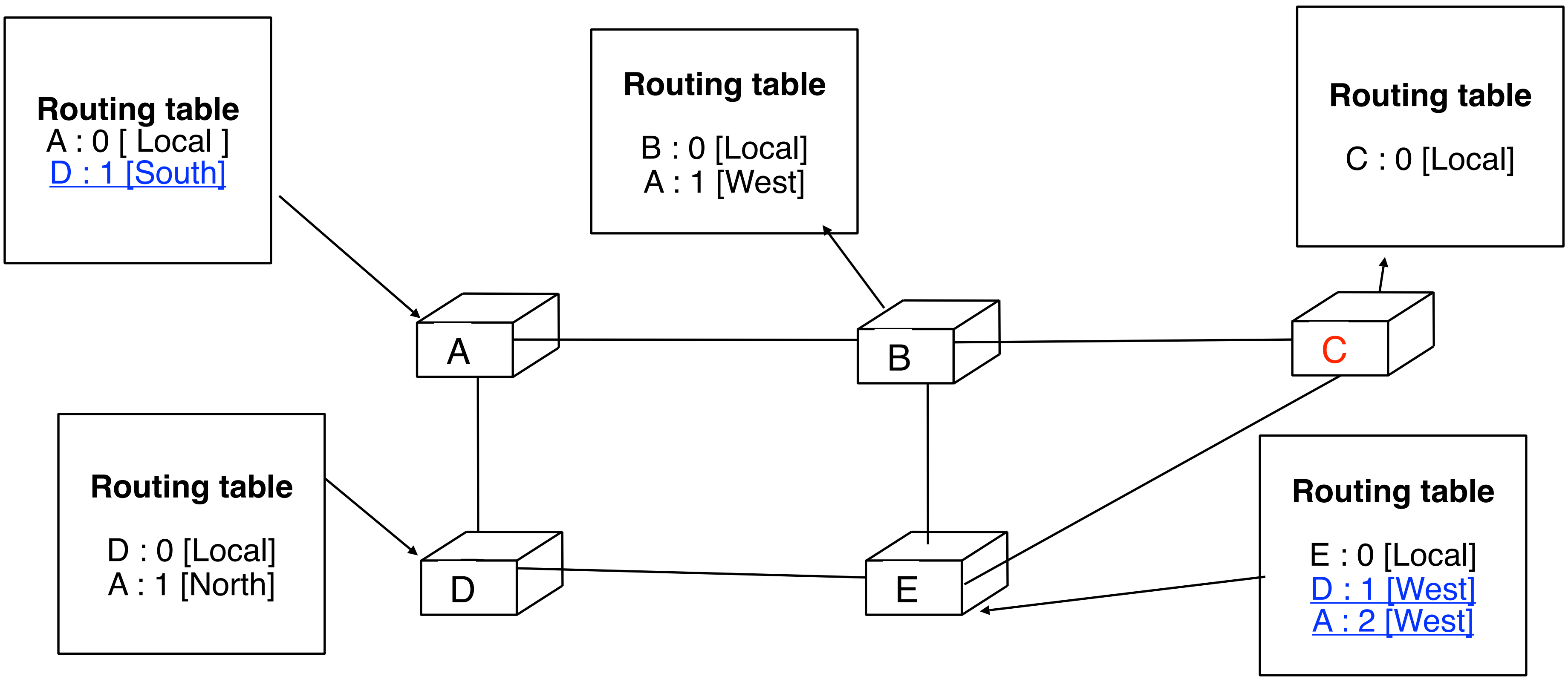


When a router boots, it only knows itself. Its distance vector thus contains only its address

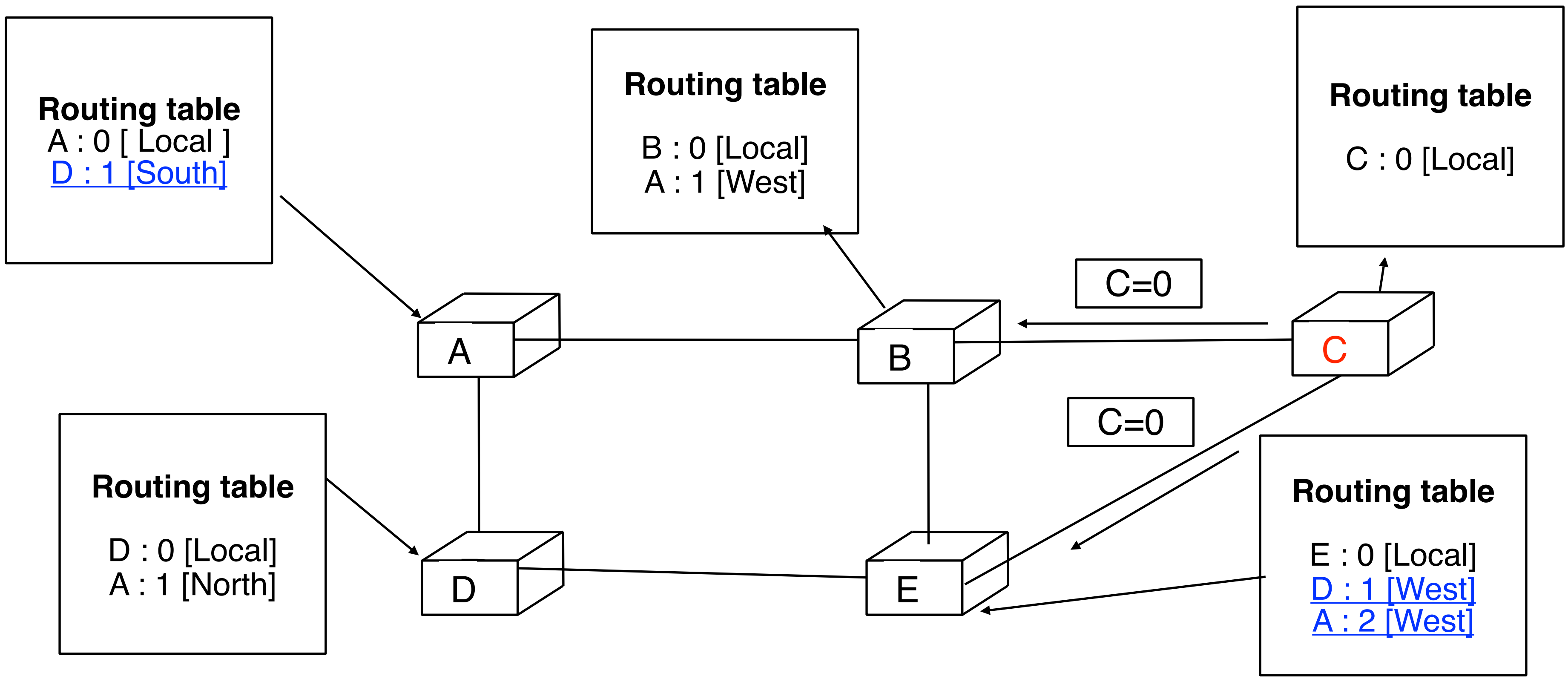
Distance vectors example (2)



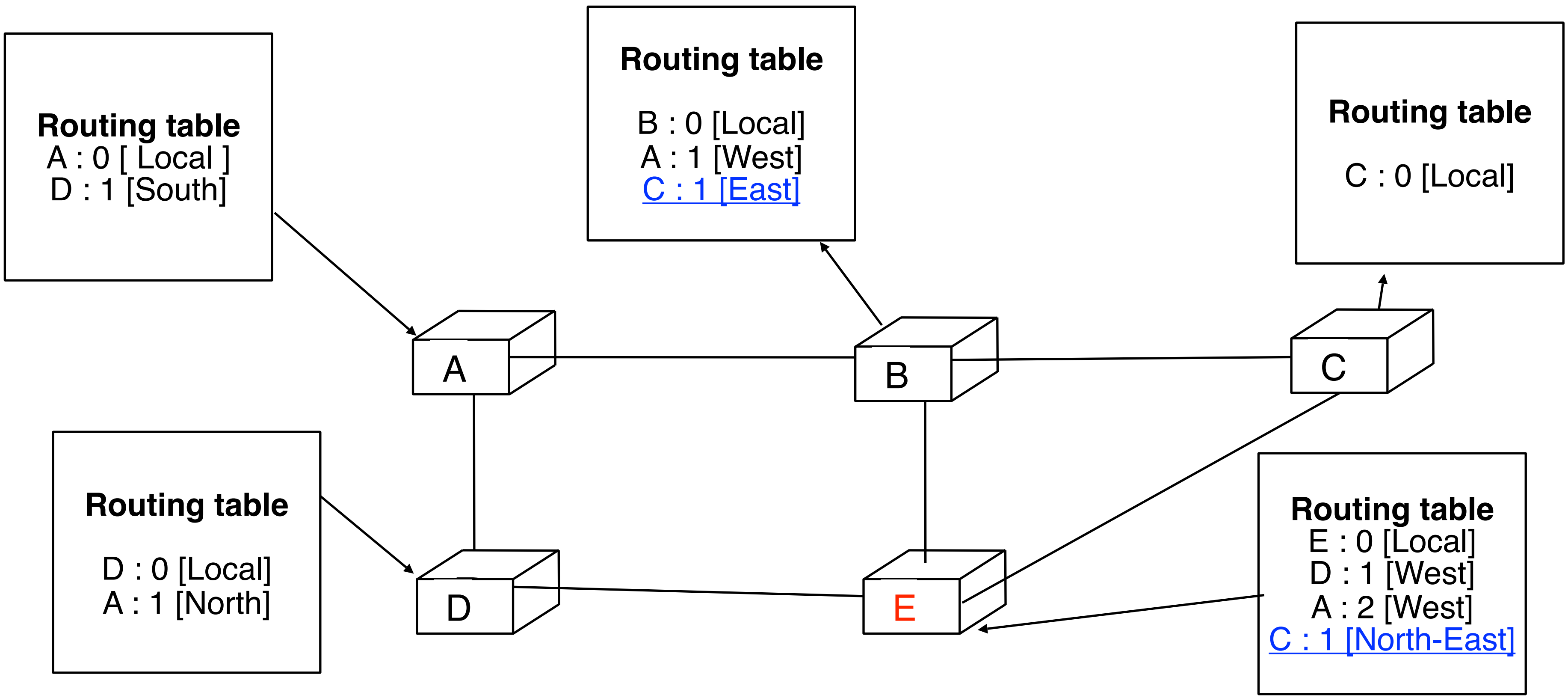
Distance vectors example (3)



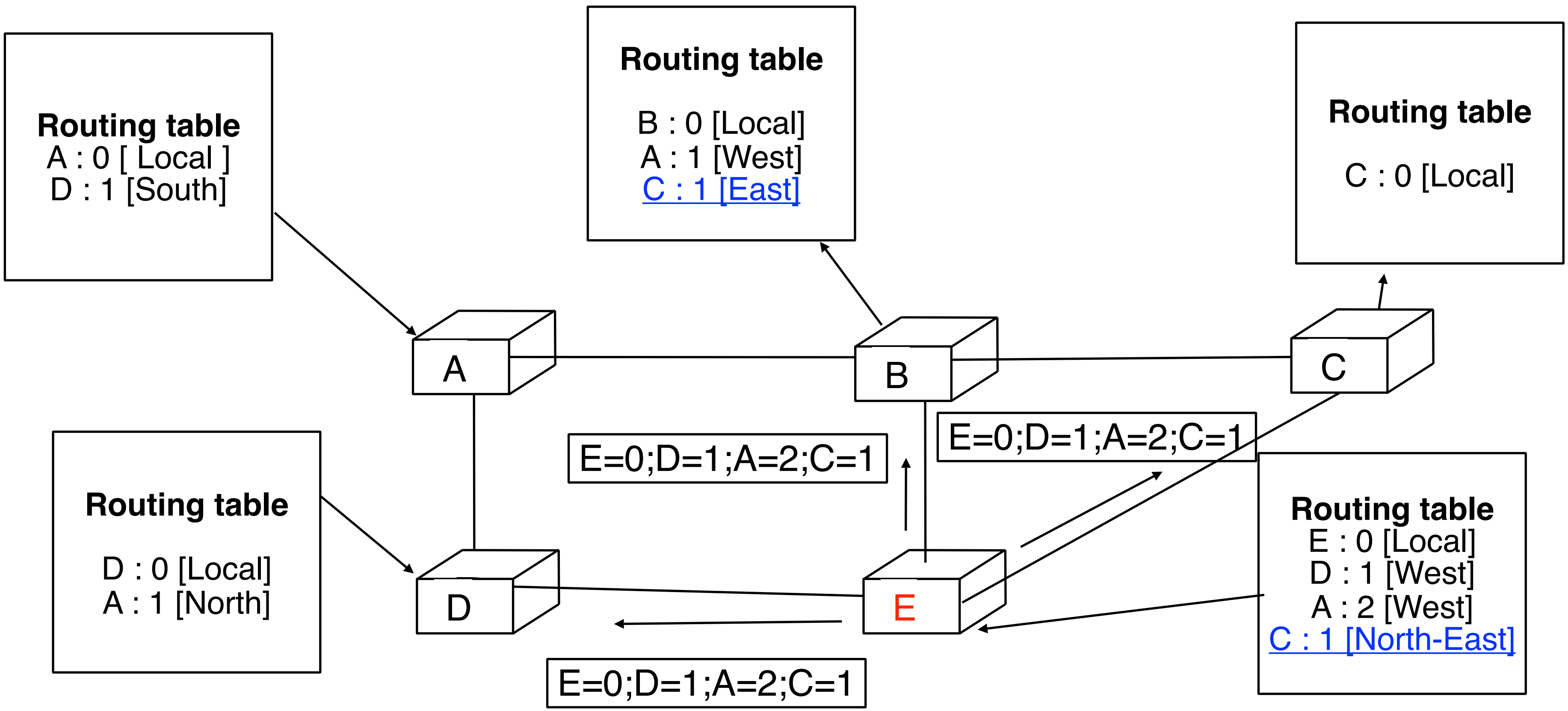
Distance vectors example (3)



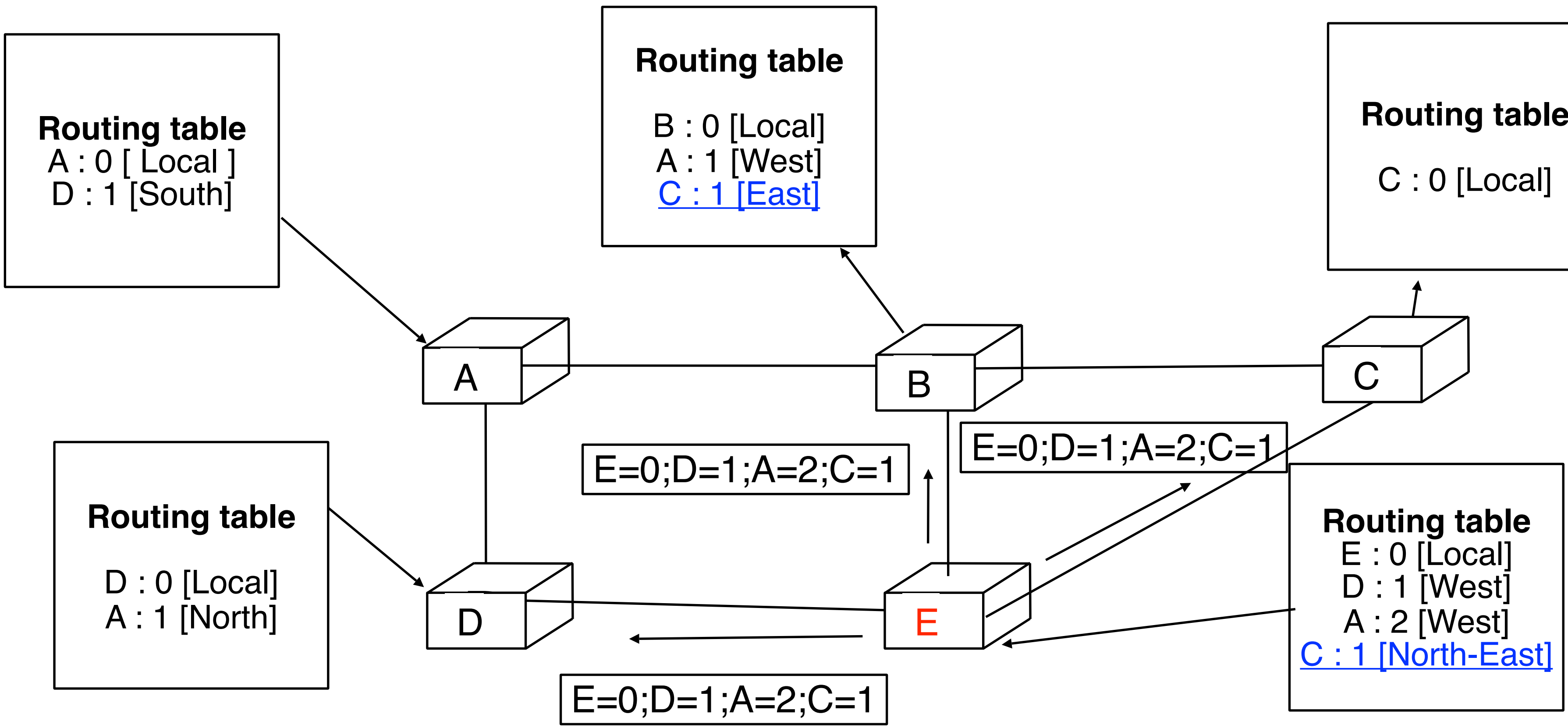
Distance vectors example (4)



Distance vectors example (4)



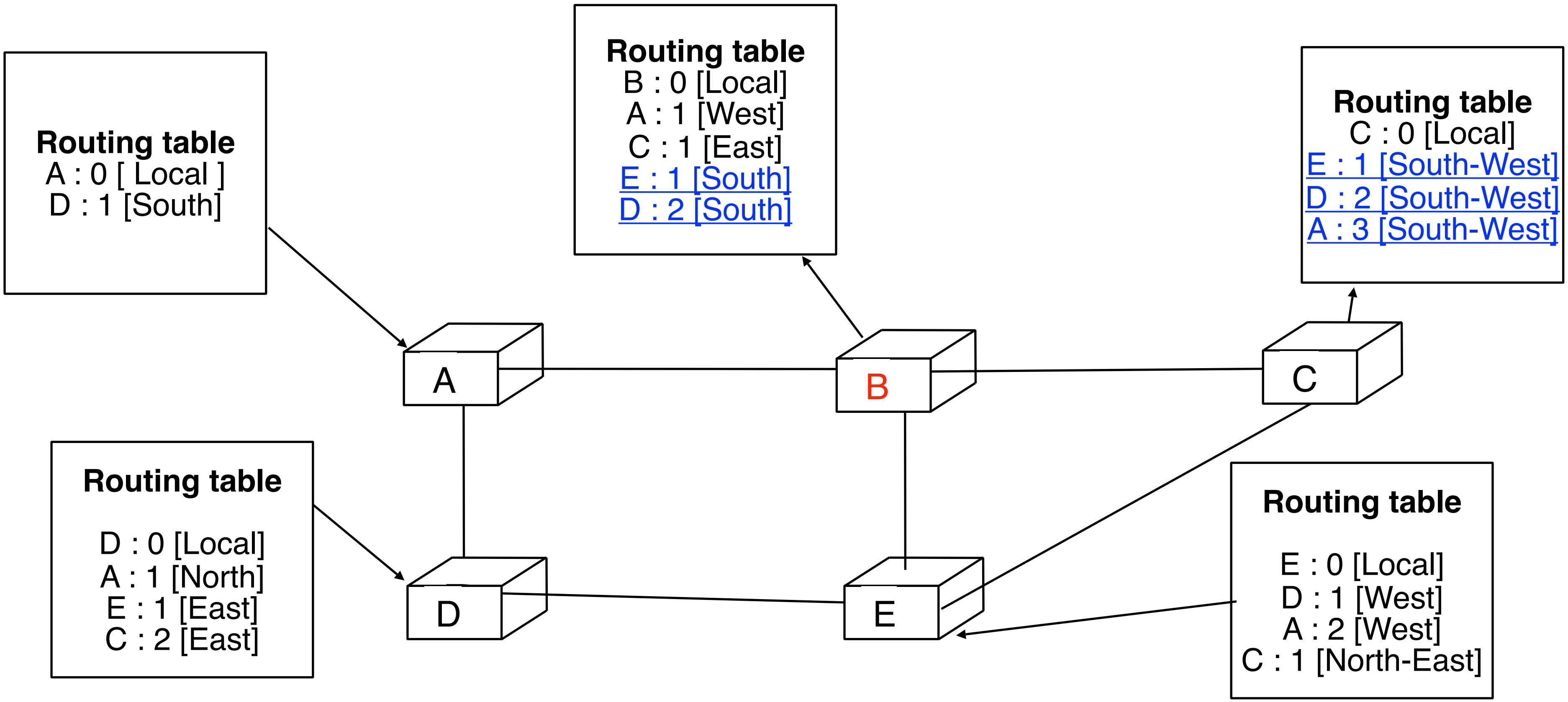
Distance vectors example (4)



- Reception of distance vector on B
  - New route to reach E and D, longer route for A
- Reception of distance vector on C
  - New routes to reach D, A and E
- Reception of distance vector on D
  - New routes to reach E and C, longer route for A

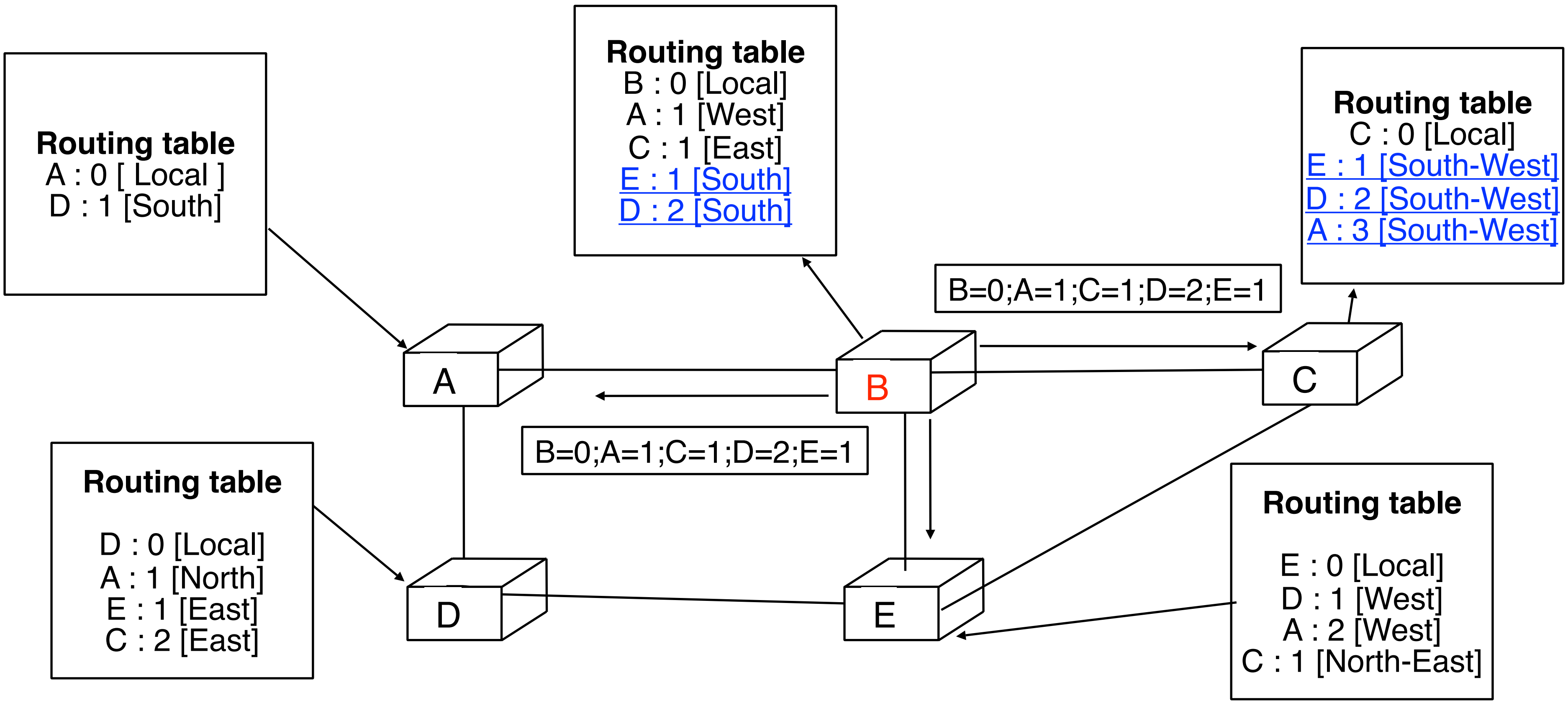
Distance vectors example (5)

- B is the first to send its vector

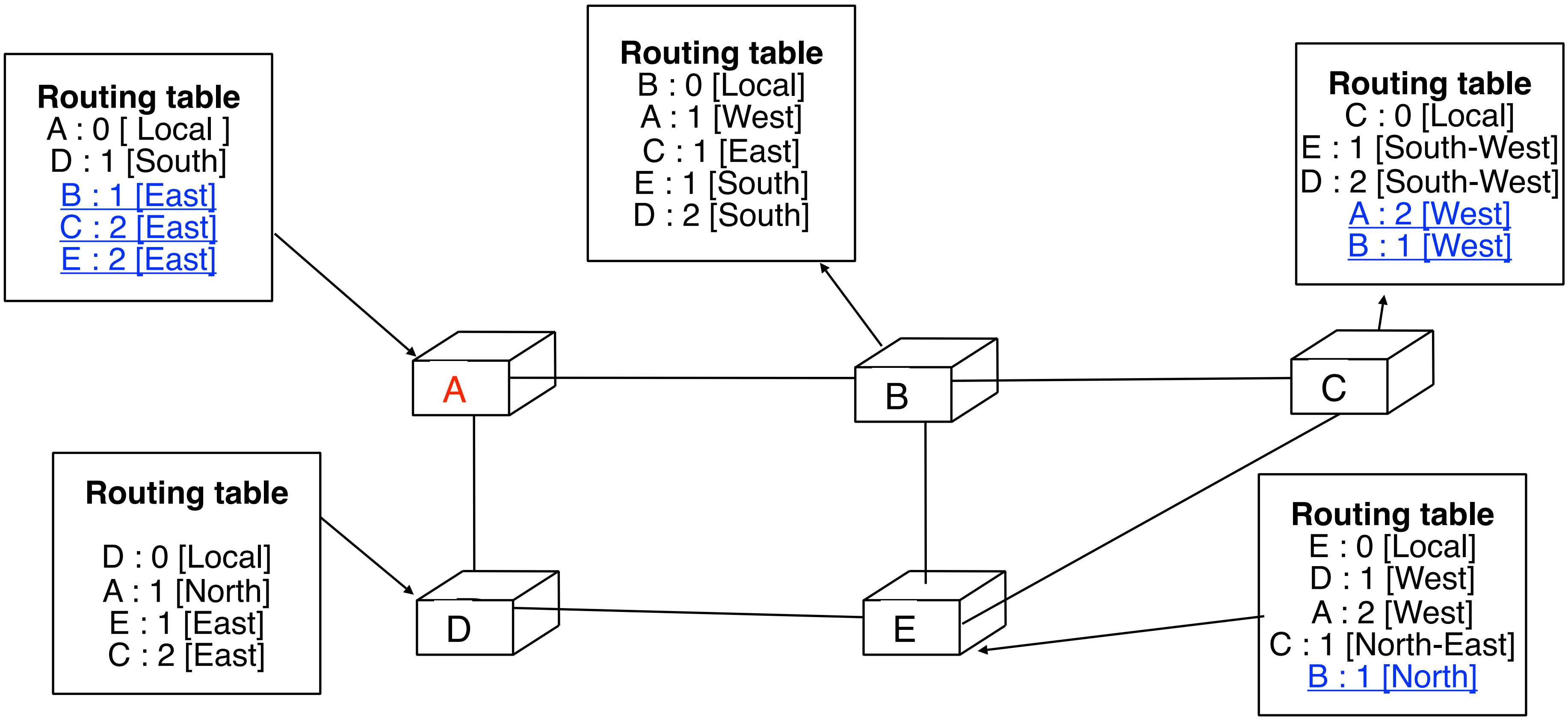


Distance vectors example (5)

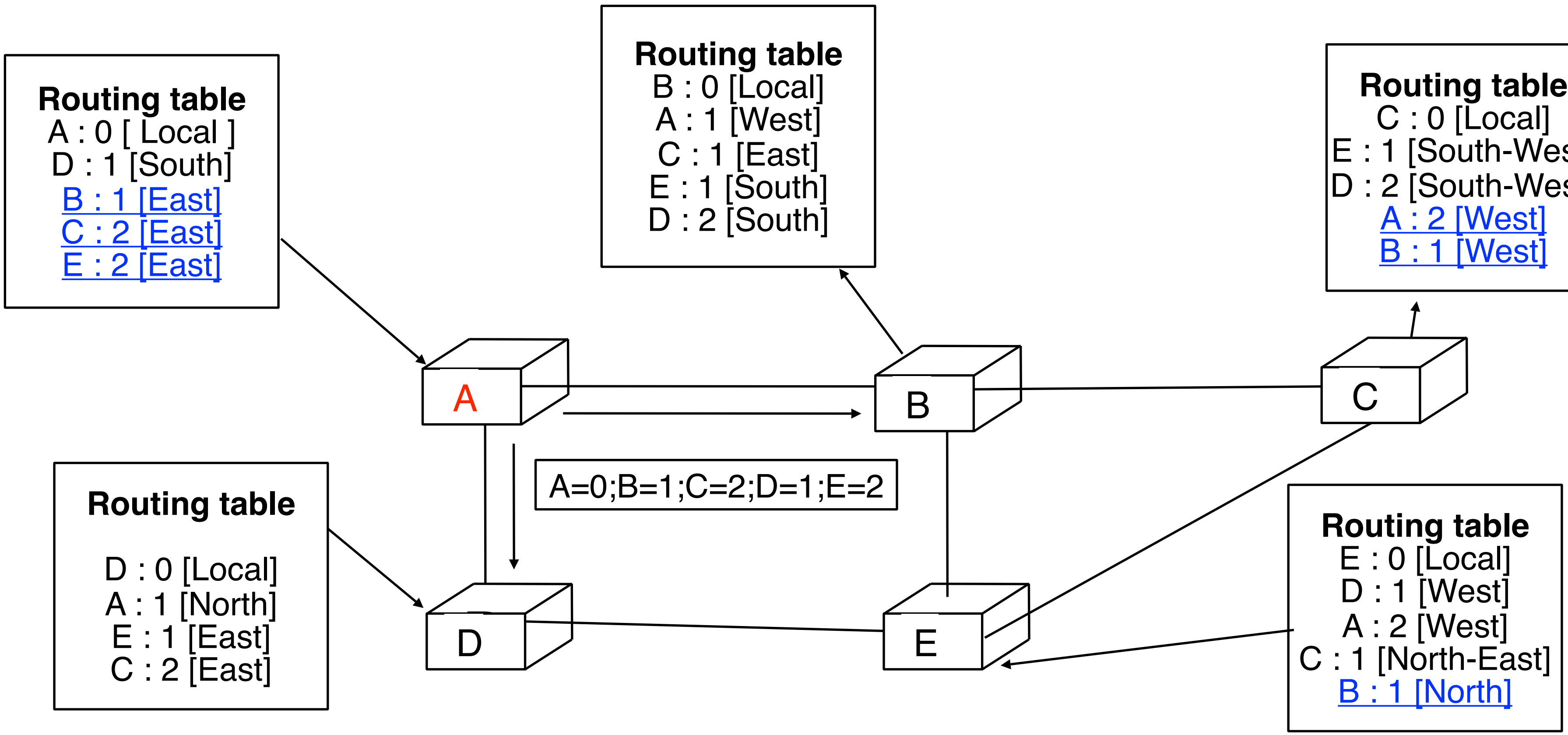
- B is the first to send its vector



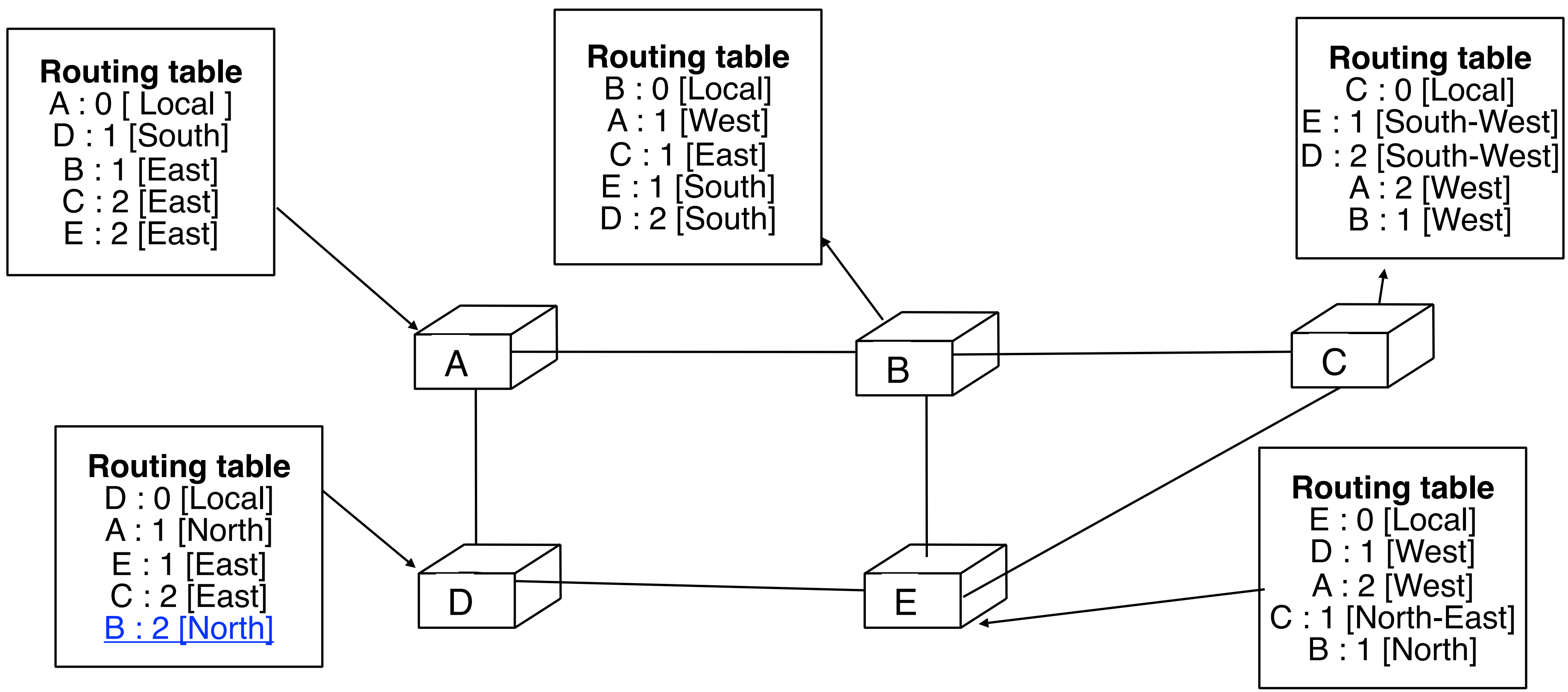
Distance vectors example (6)



Distance vectors example (6)



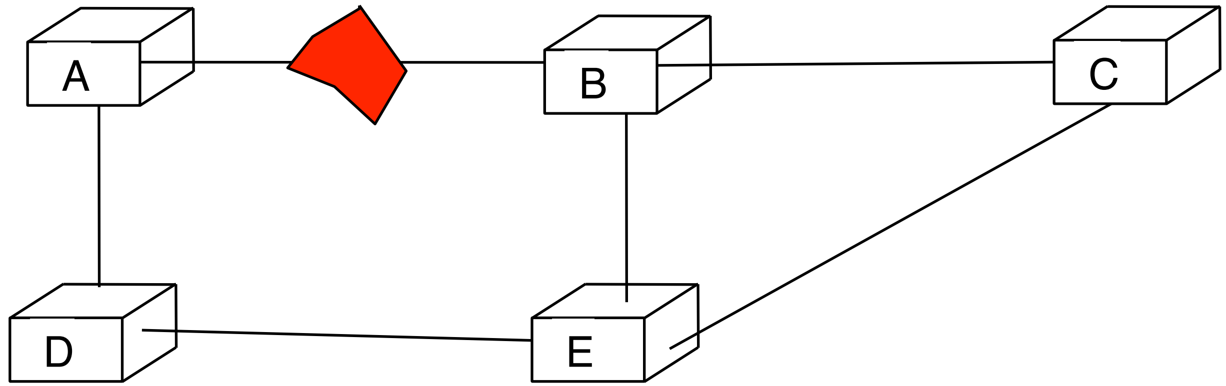
Distance vectors example (7)



- All routers know how to reach all other routers
- Routing tables are stable
  - If a distance vector is sent by one router, it will not cause any change to the routing table of other routers in the network

Distance vectors
Link failures

- How to deal with link failures ?



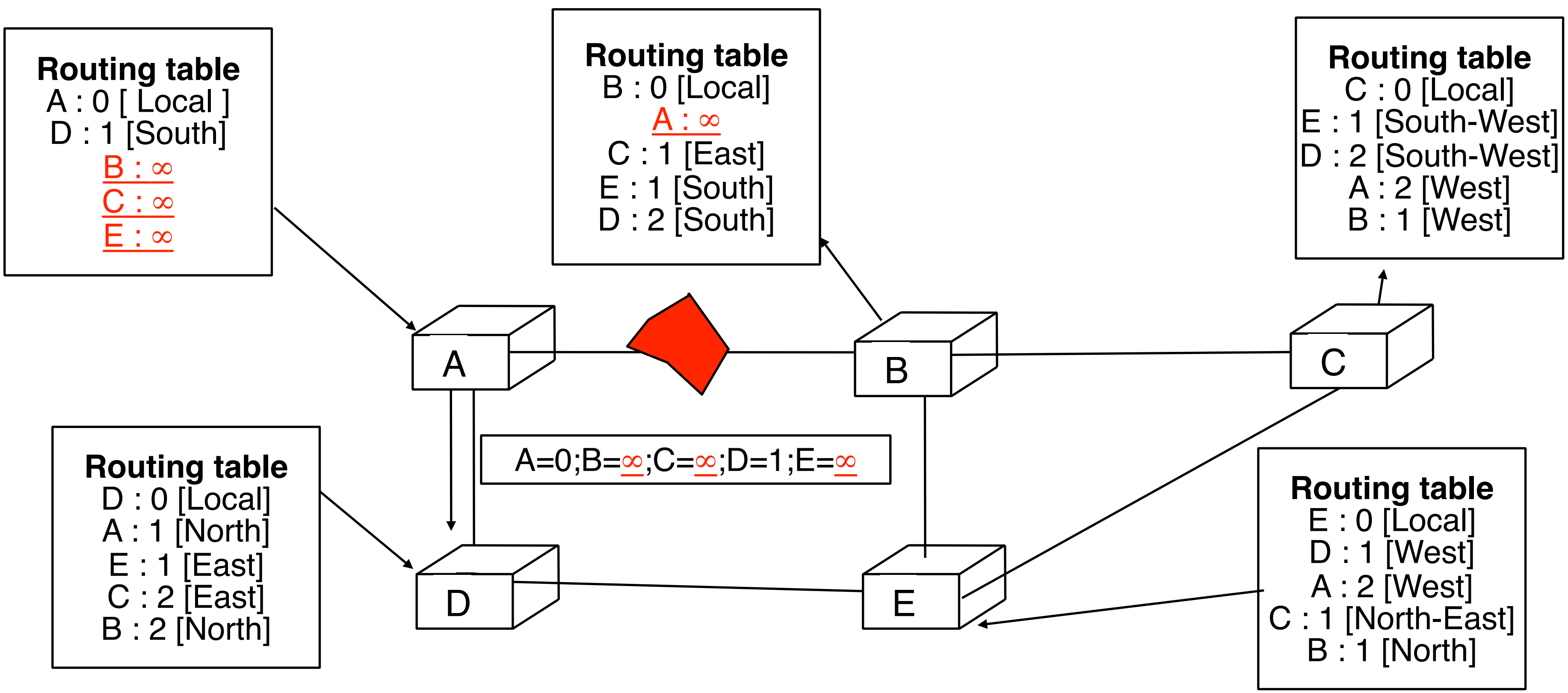
- Two problems must be solved for failures
  - How to detect that the link has failed ?
  - How to indicate to all routers that they should update their routing table since the paths that use link A-B do not work anymore ?

Detection of link failures

- Two types of solutions
  - rely on failure information from datalink or physical layer
    - fast and reliable
    - unfortunately not supported by all datalink/physical layers
  - ask each router to regularly send its distance vector (e.g. every 30 seconds)
    - If a router does not receive a refresh for a route in a distance vector from one of its neighbours during some time (e.g. 90 seconds), it assumes that the route is not available anymore

How to update the routing table ?

- All routes that use a failed link are marked with an infinite cost

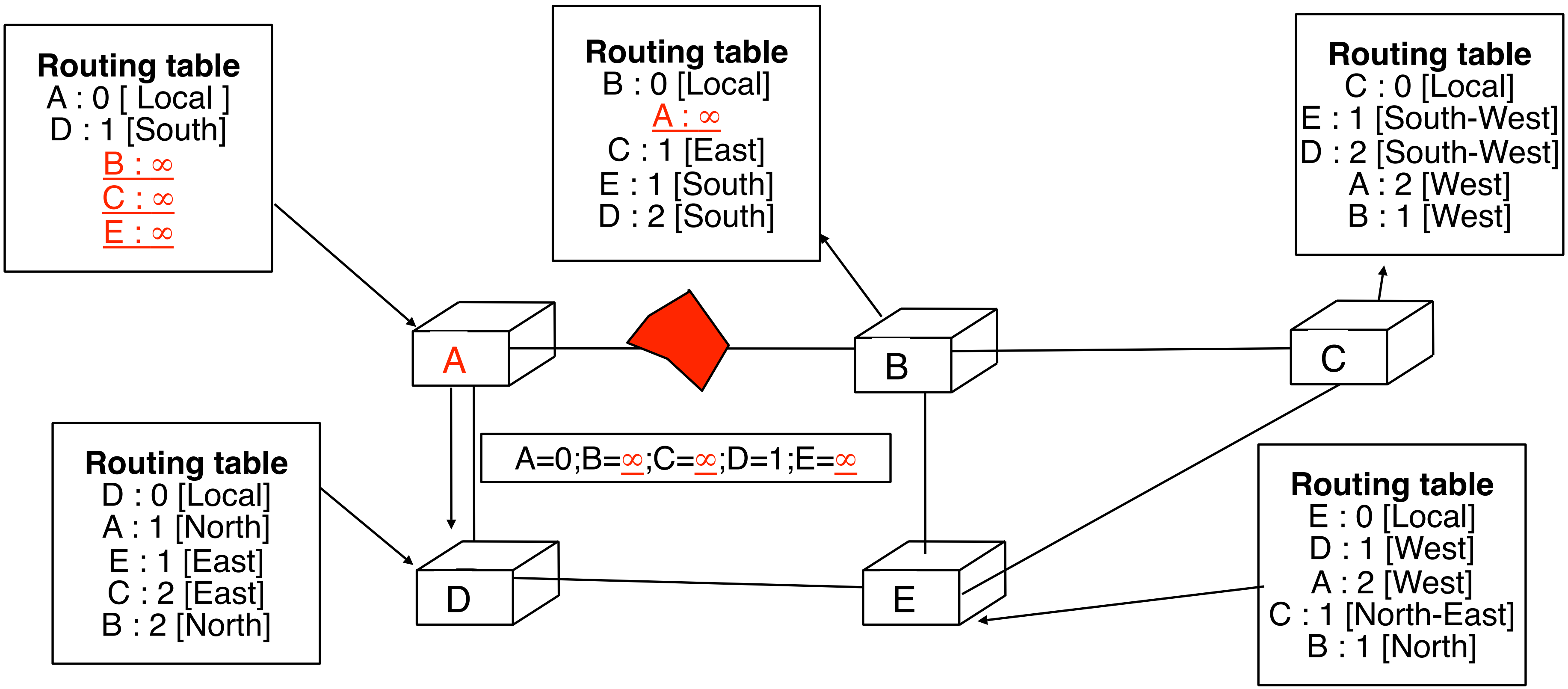


How to update the routing table ? (2)

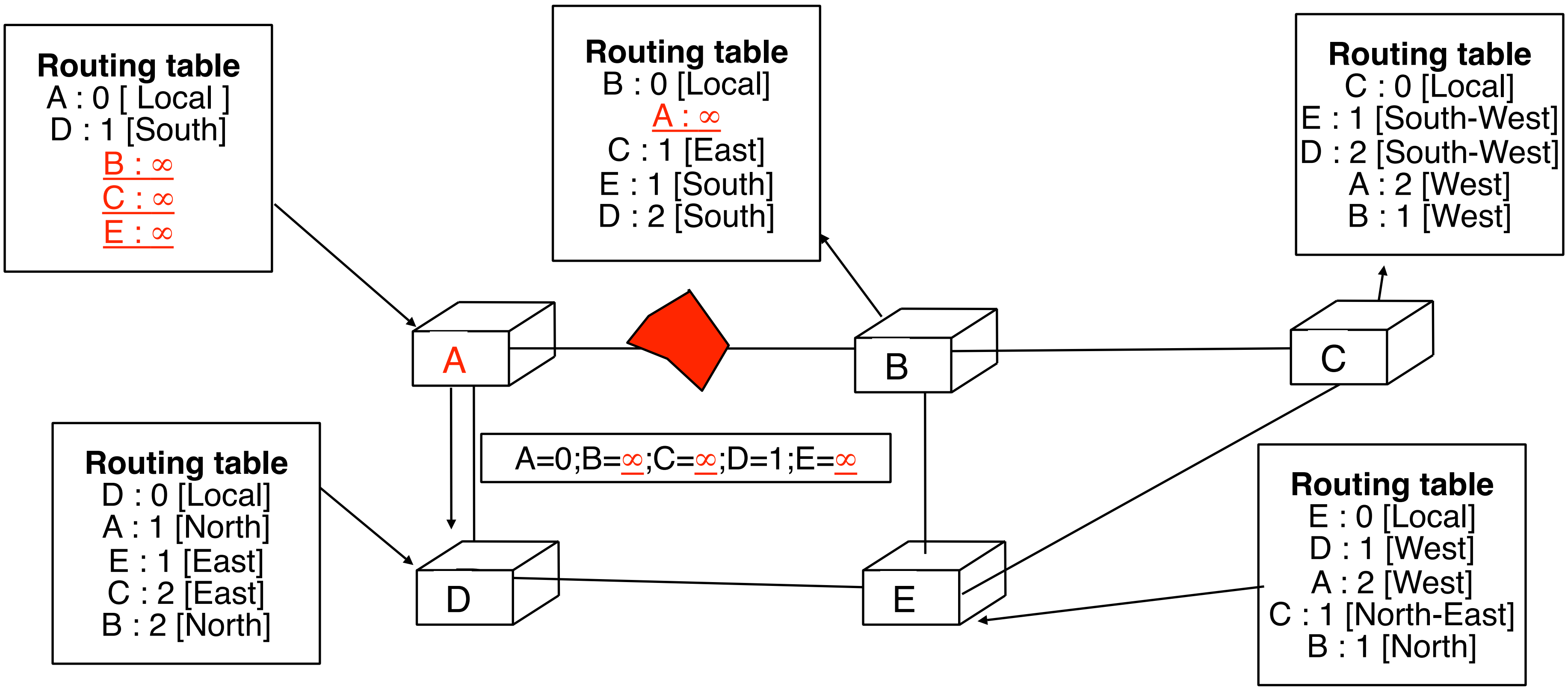
- Reception of a distance vector

```
Received(Vector V[],link l)
{ /* received vector from link l */
for each destination=d in V[]
{
  if (d isin R[])
  { if ( ((V[d].cost+l.cost) < R[d].cost) OR
         ( R[d].link == l) )
       { /* better route or change to current route */
         R[d].cost=V[d].cost+l.cost;
         R[d].link=l;
       }
   }
   else
   { /* new route */
     R[d].cost=V[d].cost+l.cost;
     R[d].link=l;
   }
}
```

How to update the routing table ? (3)

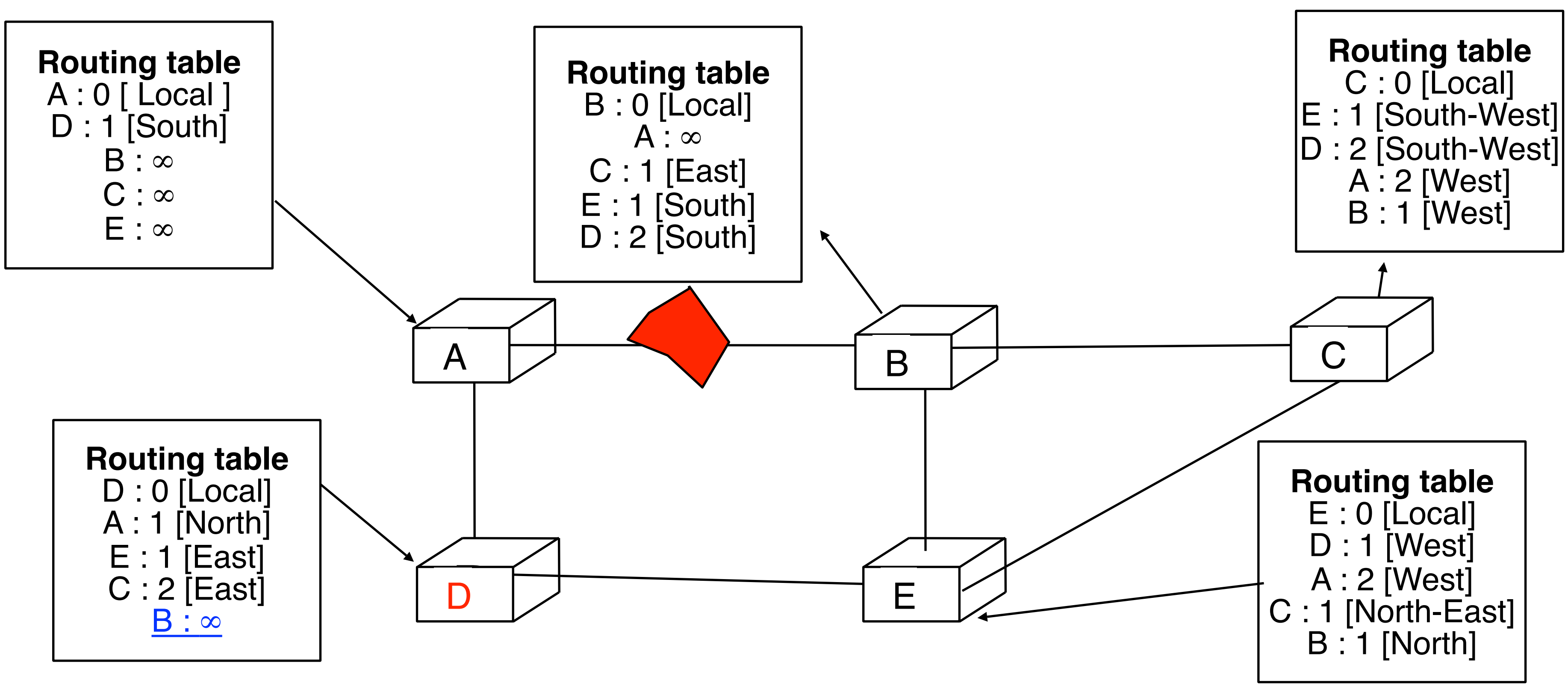


How to update the routing table ? (3)

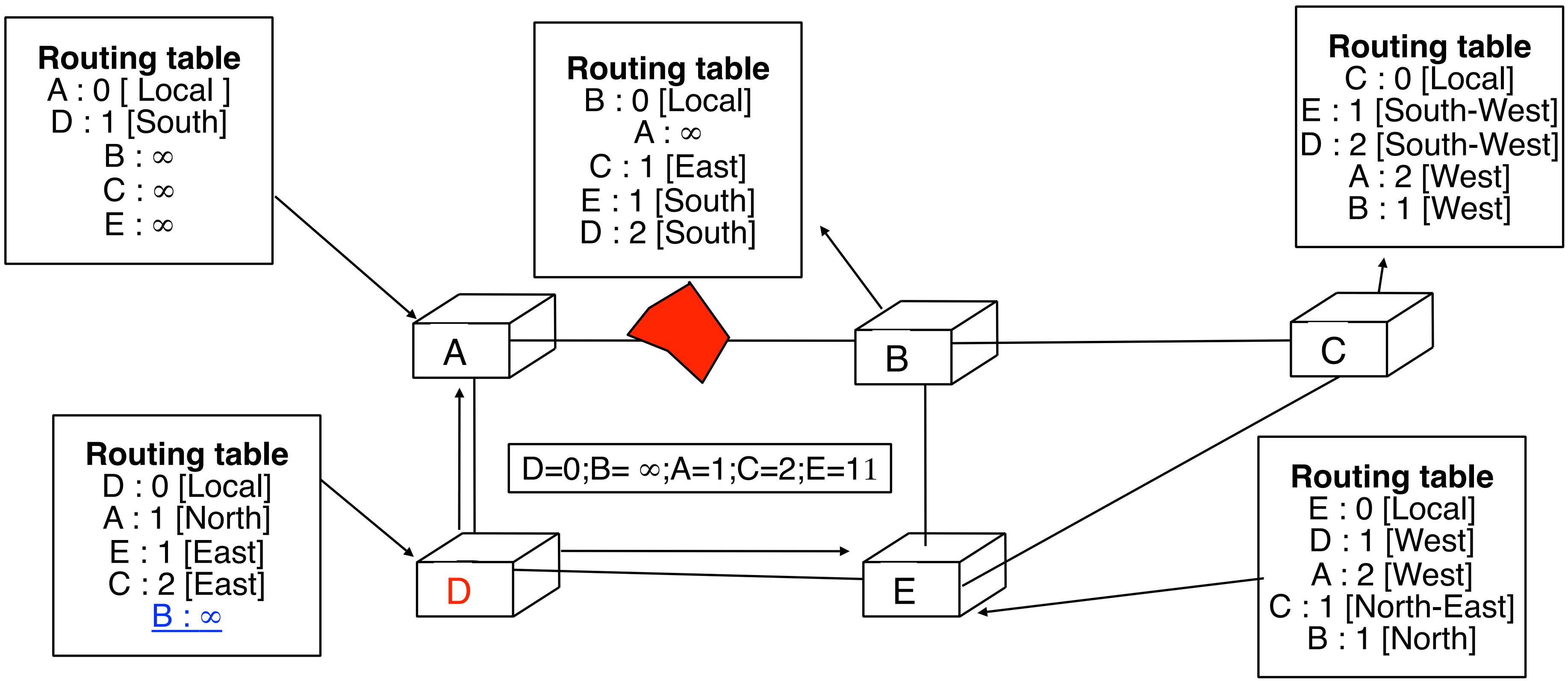


- D must remove from its routing tables all the routes that it learned from its North link and are announced now with an ∞ cost

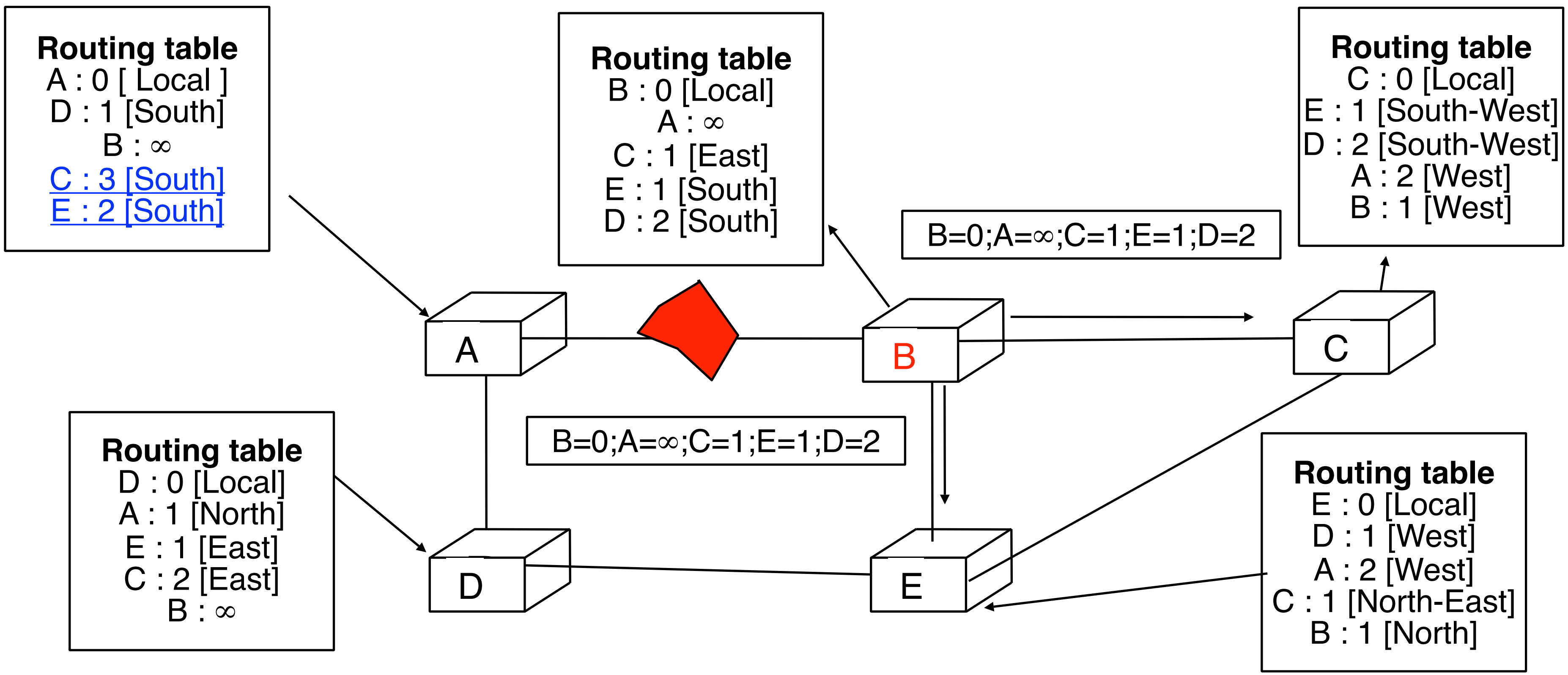
How to update the routing table ? (4)



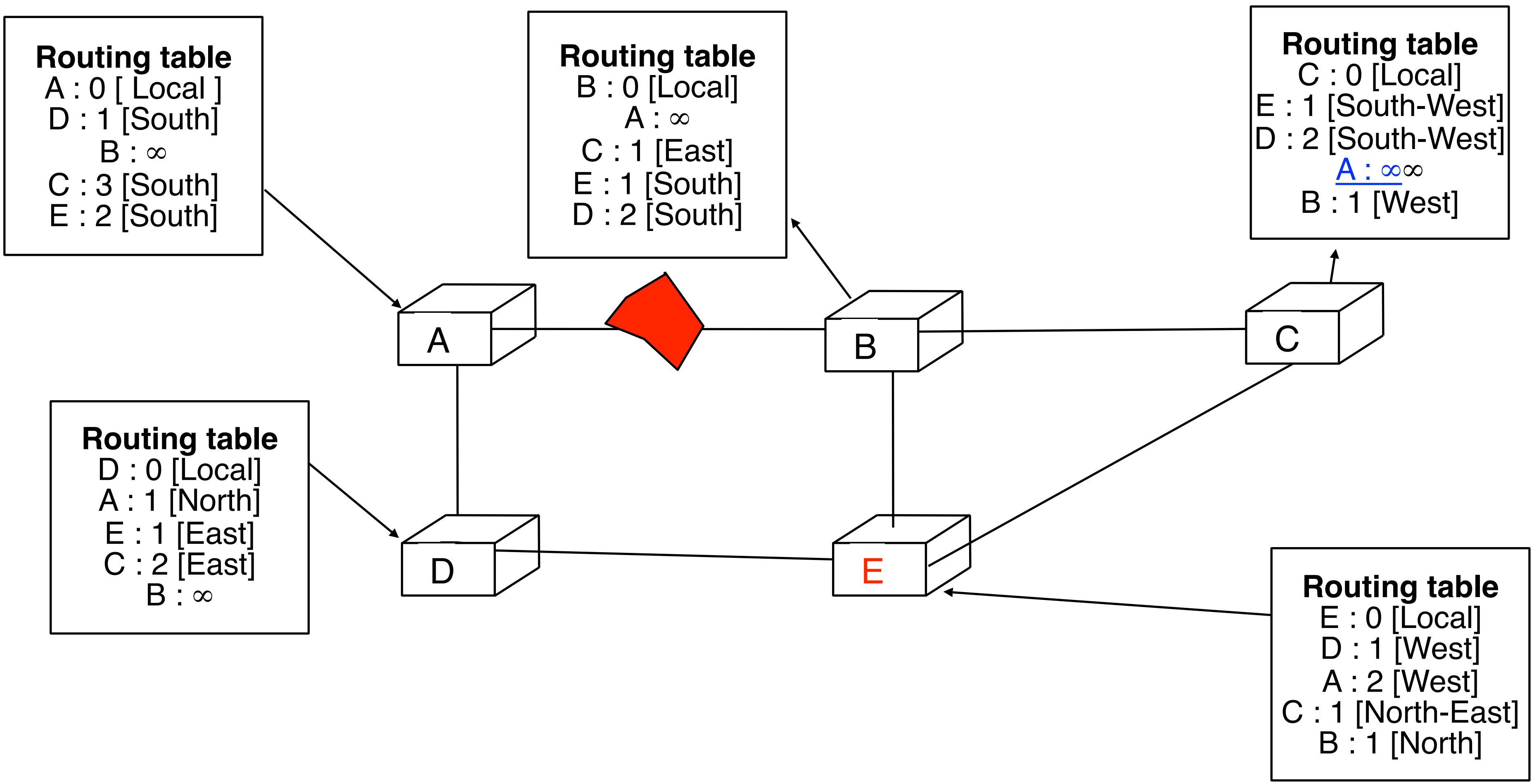
How to update the routing table ? (4)



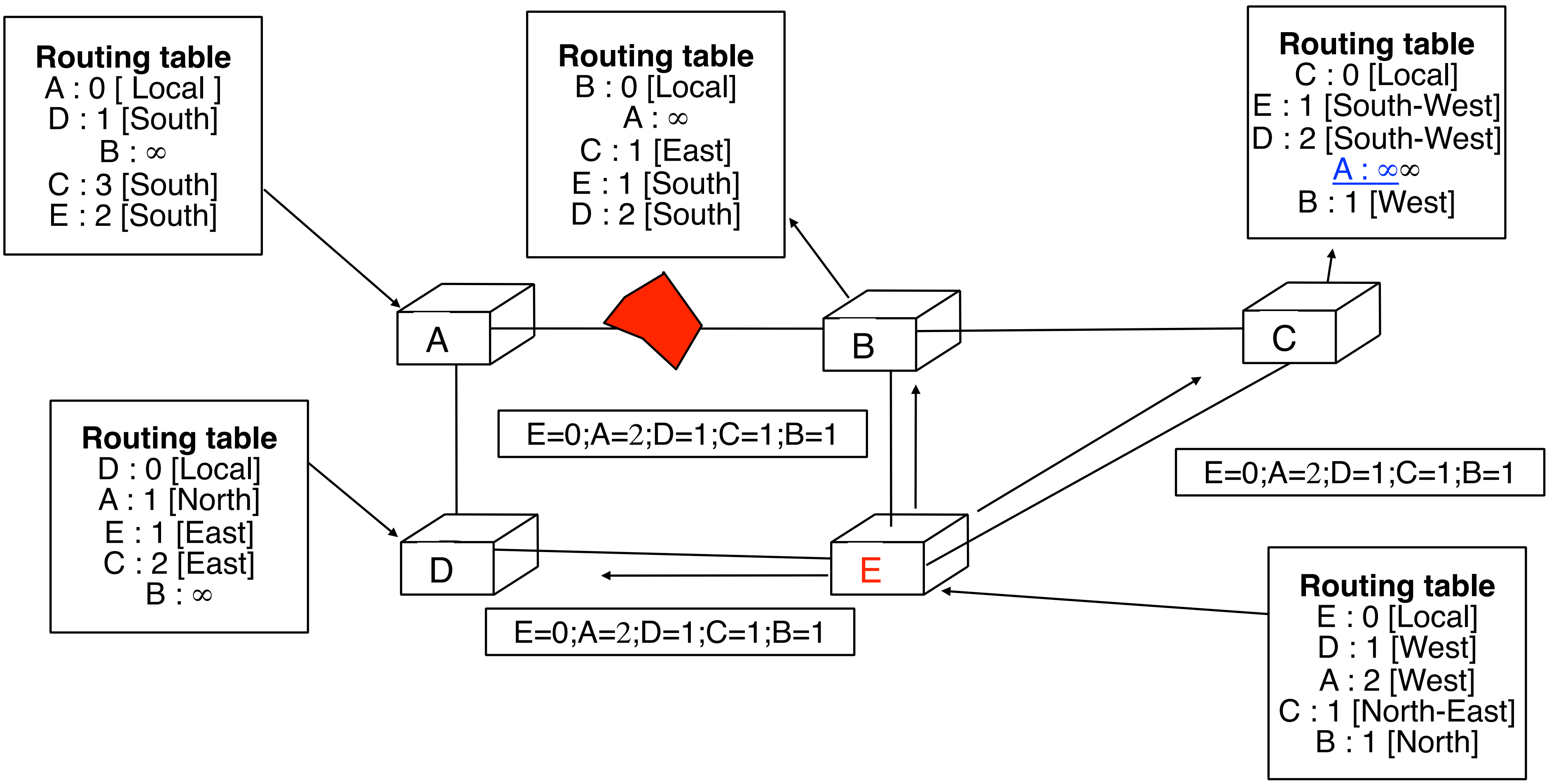
How to update the routing table ? (5)



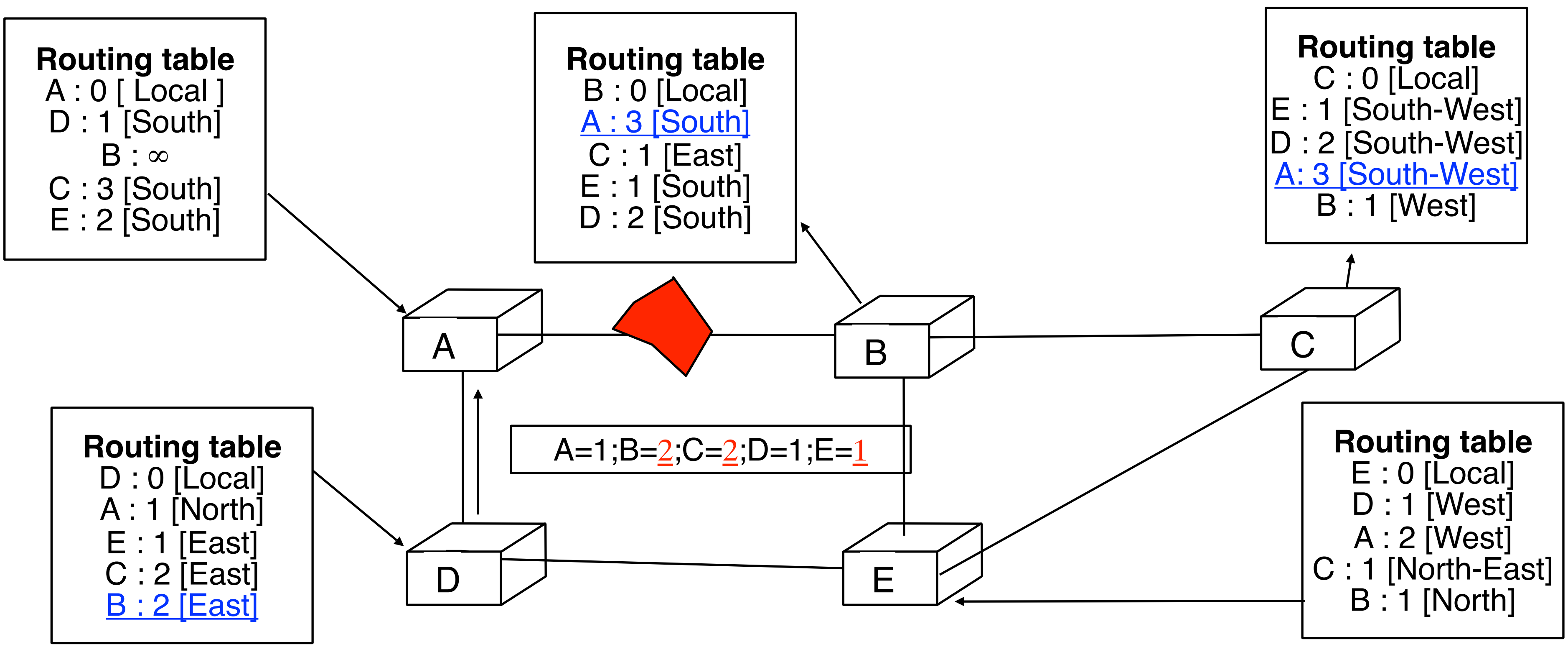
How to update the routing table ? (6)



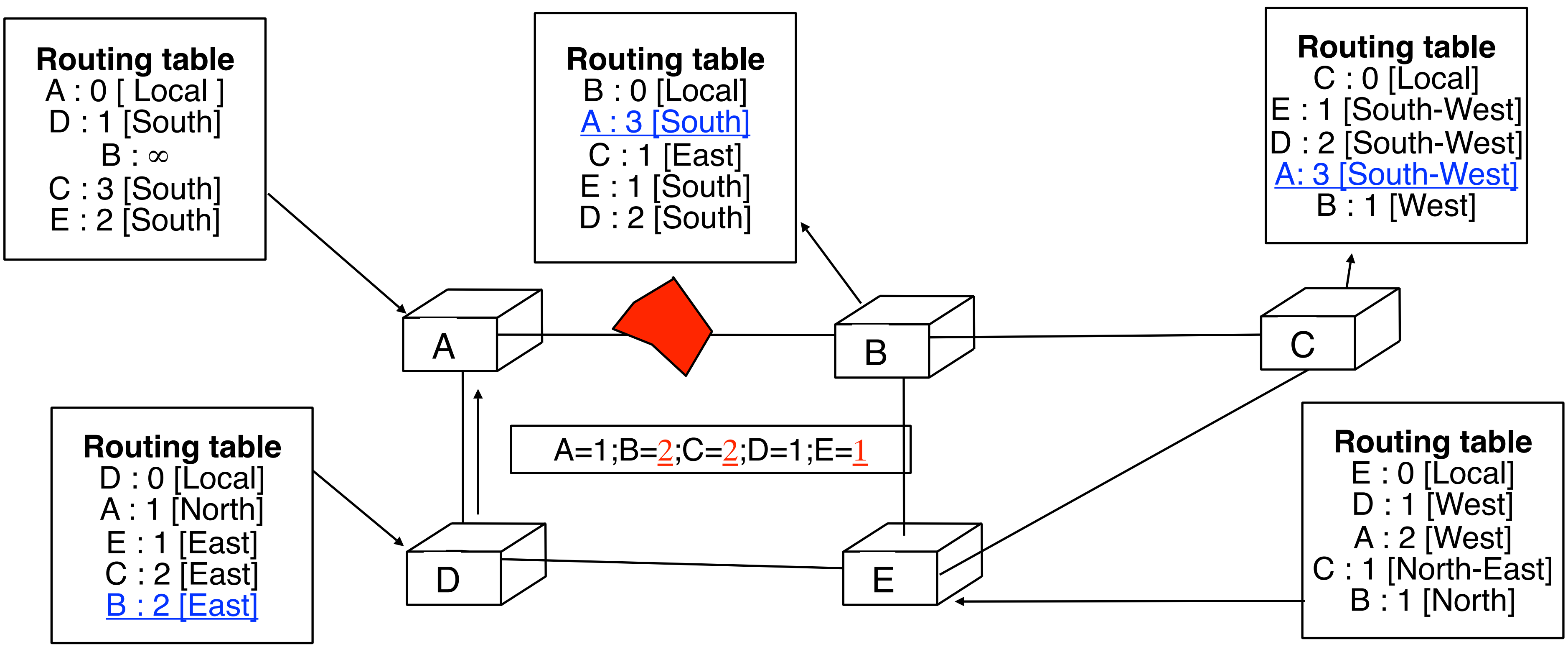
How to update the routing table ? (6)



How to update the routing table ? (7)

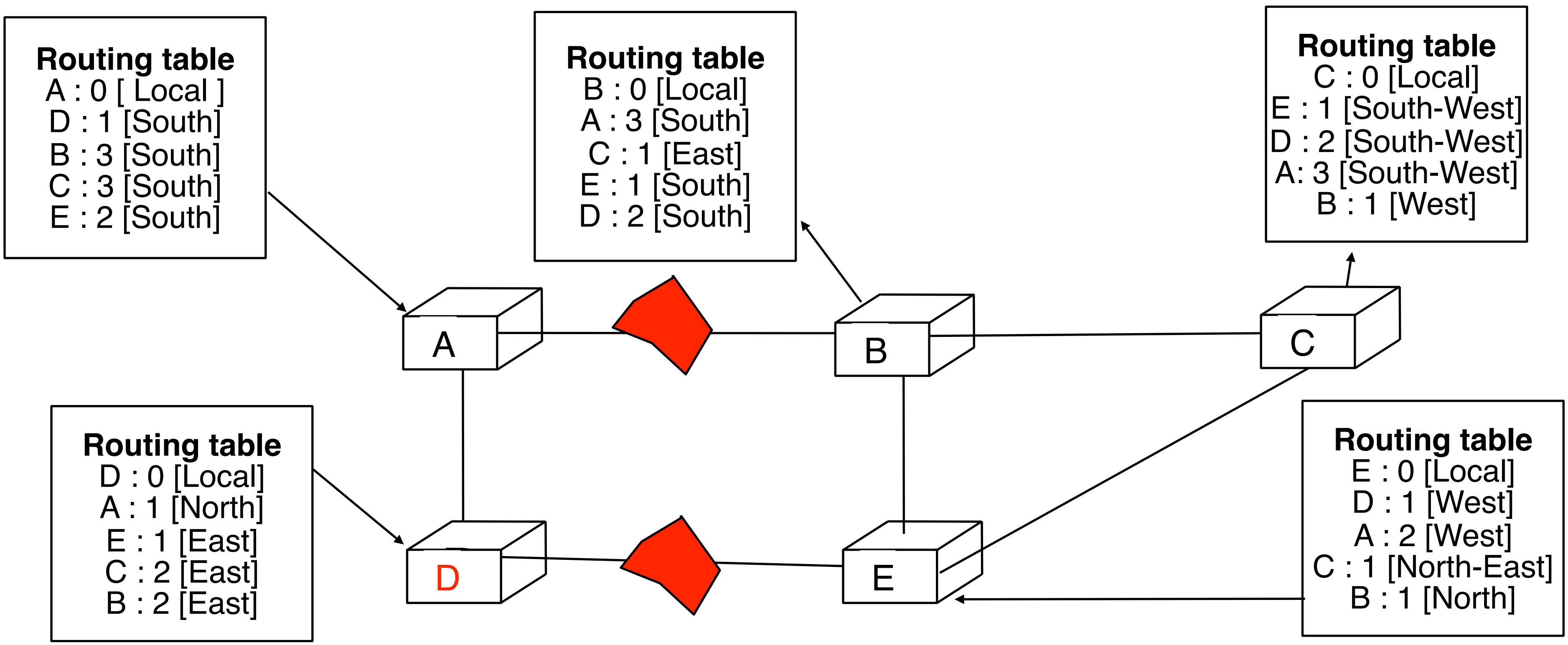


How to update the routing table ? (7)



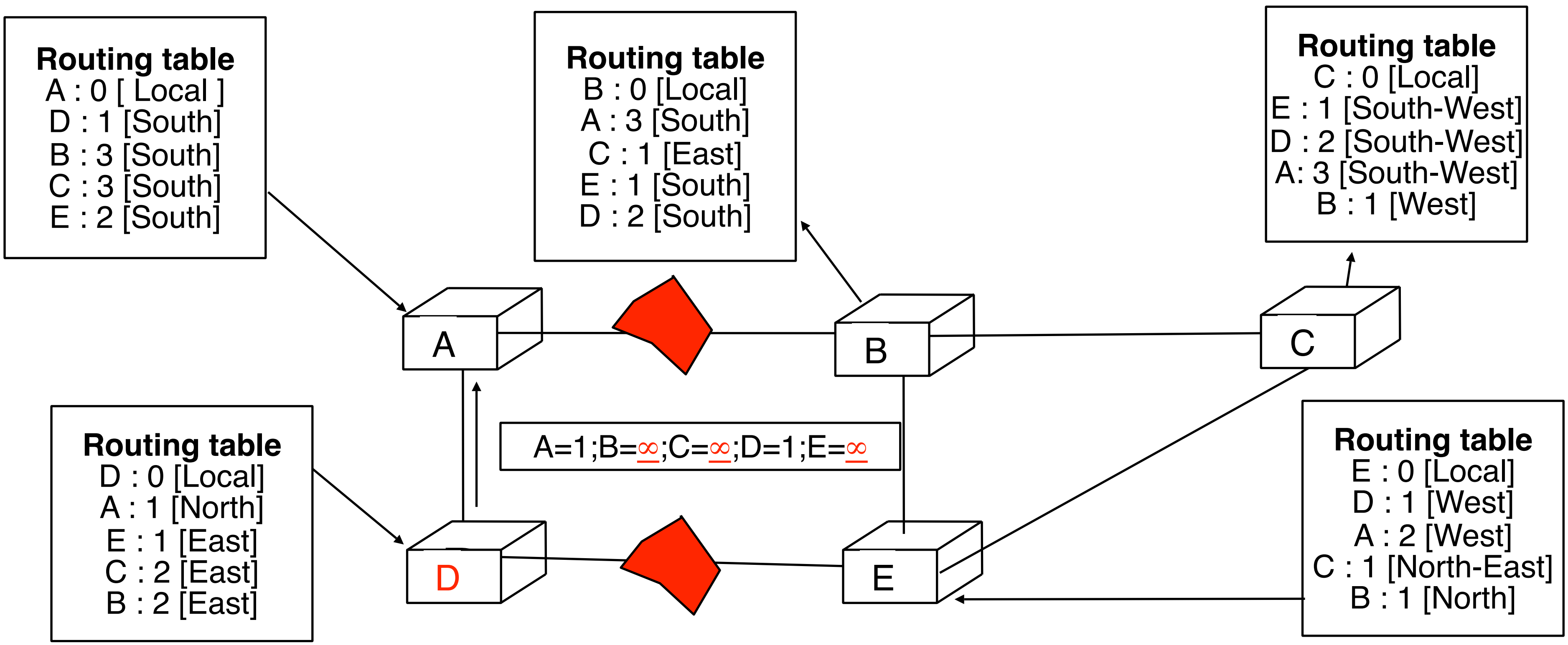
- Failure has been recovered, all routers are now reachable again from any router

Second link failure



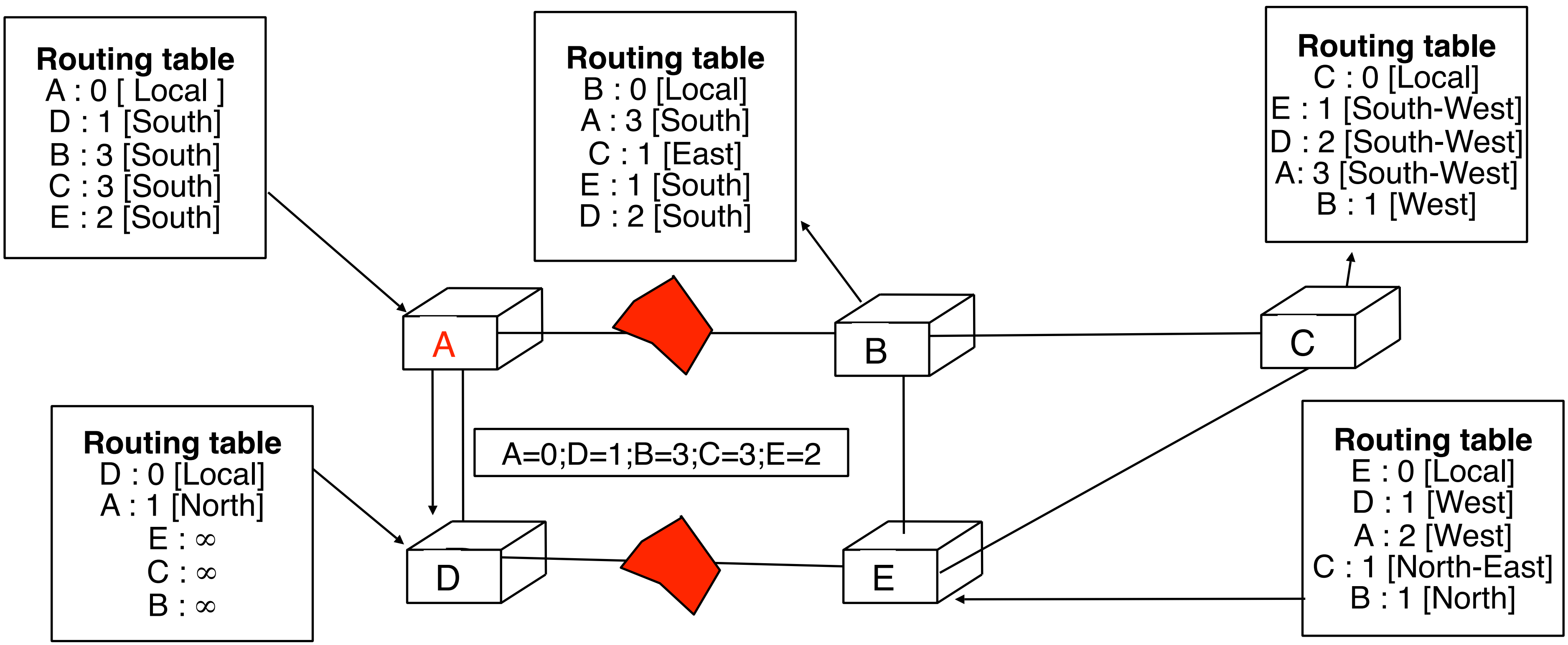
- D detects the failure
  - If it is the first to send its distance vector, failure is detected and router A updates its routing table

Second link failure



- D detects the failure
  - If it is the first to send its distance vector, failure is detected and router A updates its routing table

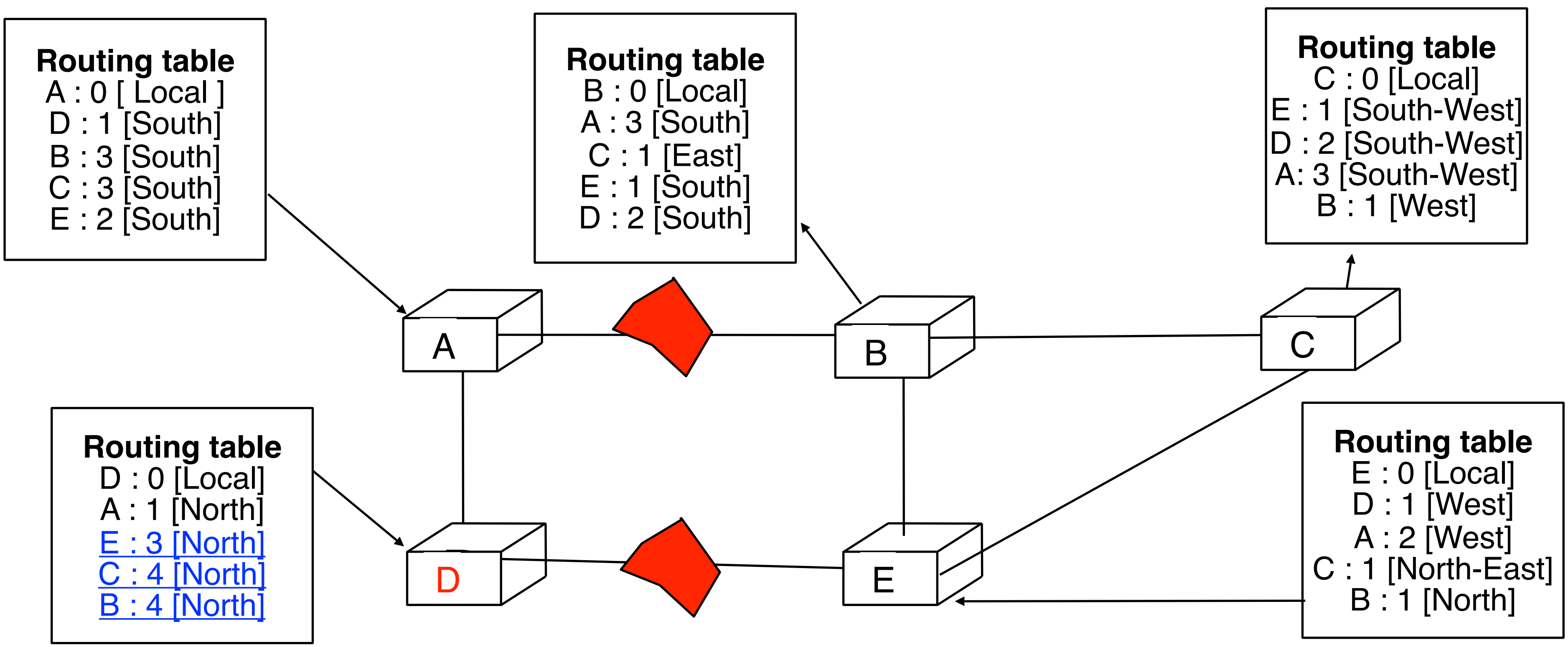
Second link failure (2)



- But if A sends its distance vector before having received or processed D's updated distance vector ...

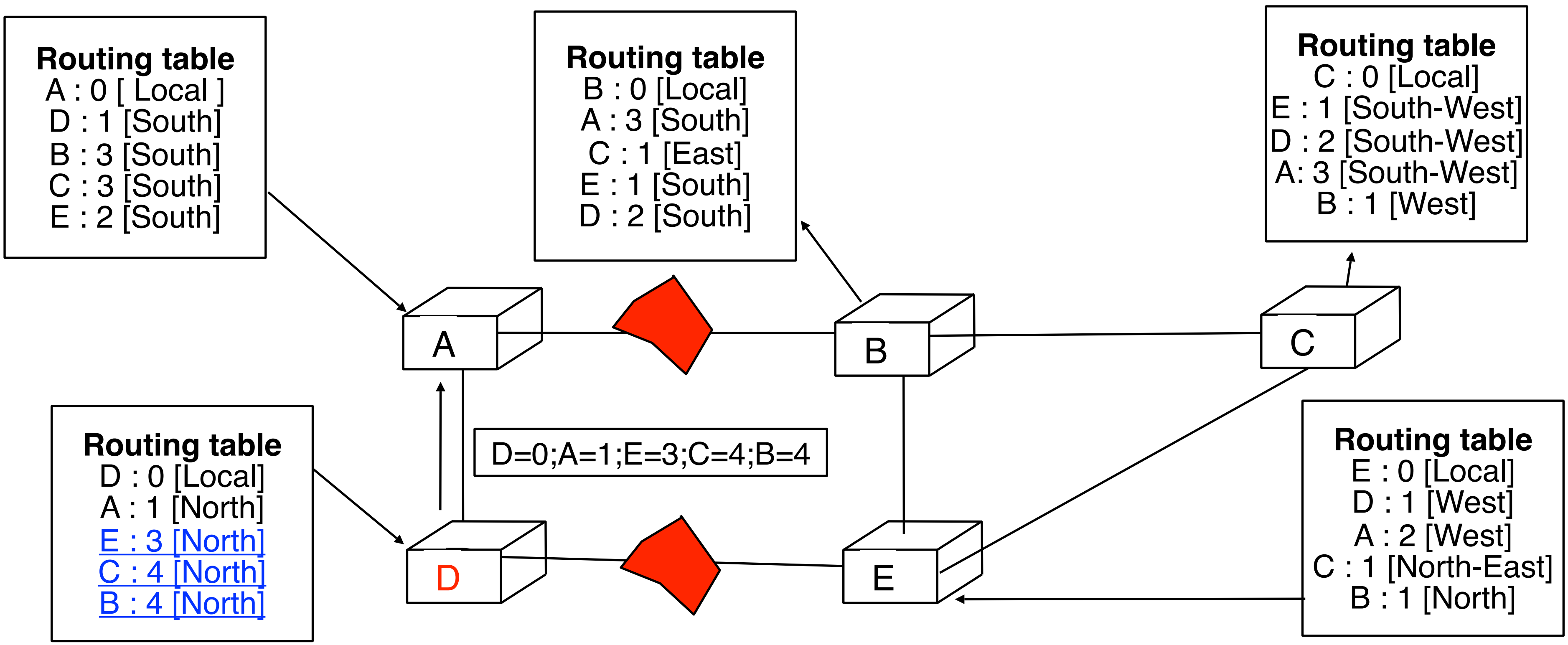
Second link failure (3)

- Upon reception of A's vector, D updates its routing table

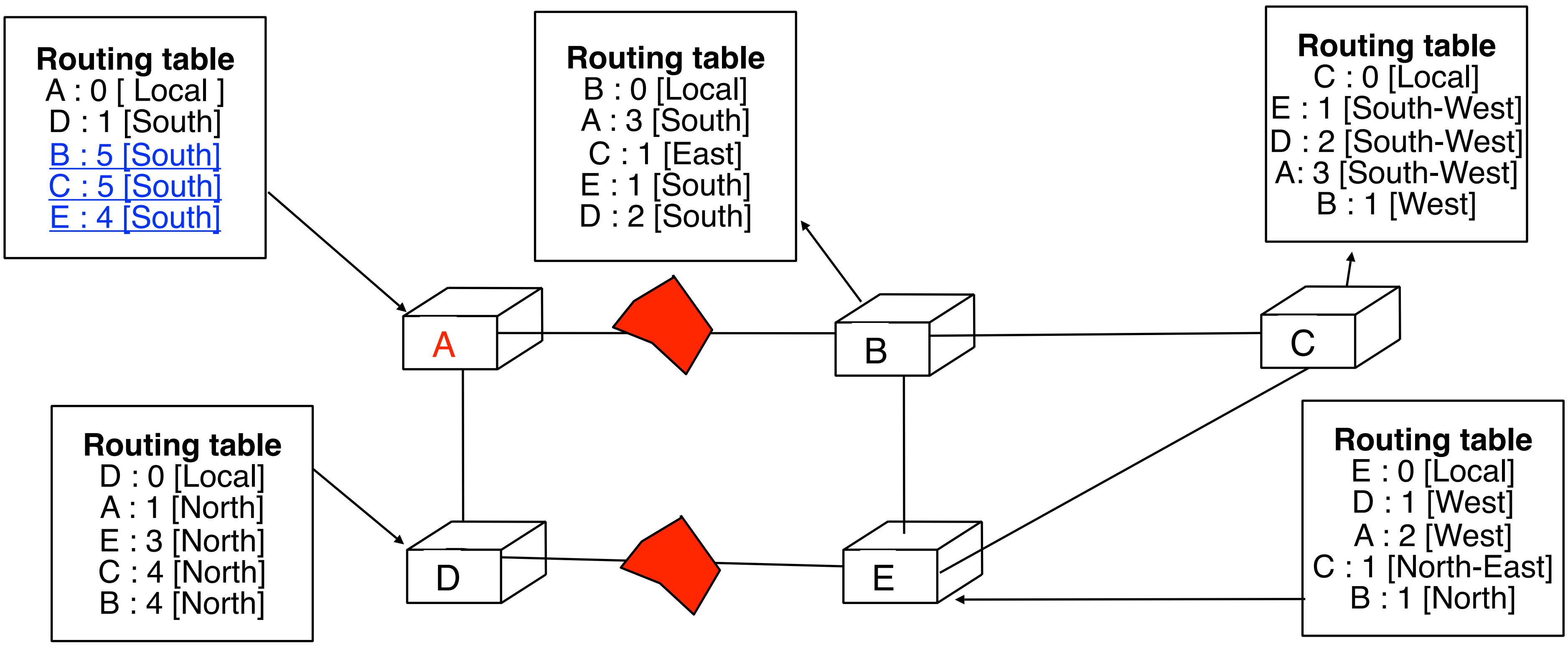


Second link failure (3)

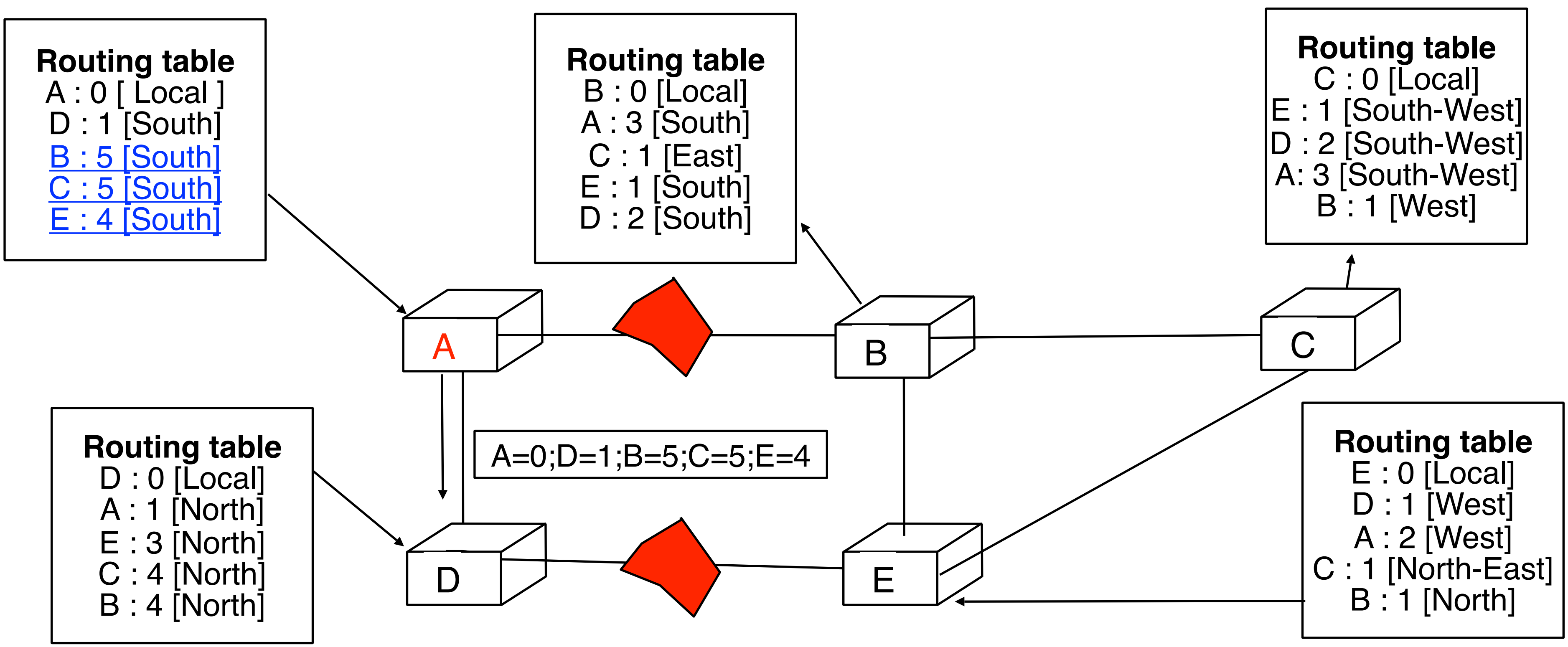
- Upon reception of A's vector, D updates its routing table



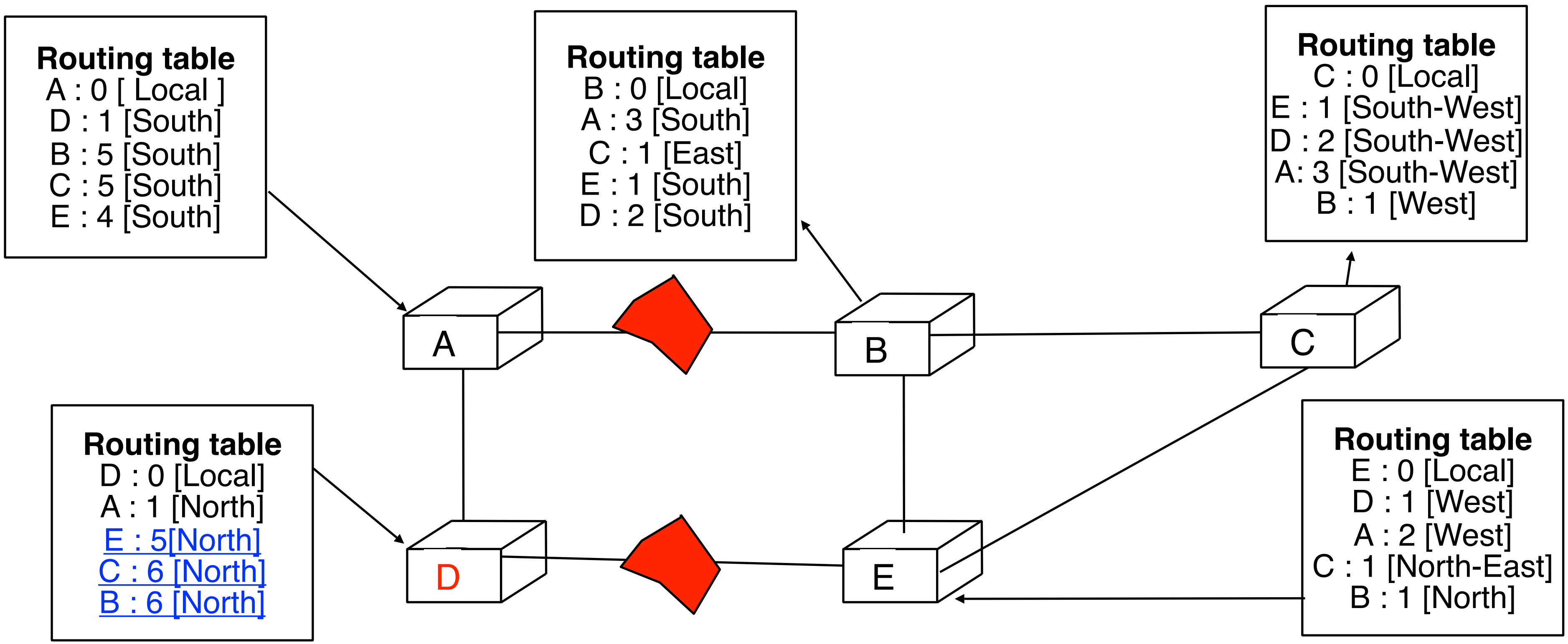
Second link failure (4)



Second link failure (4)

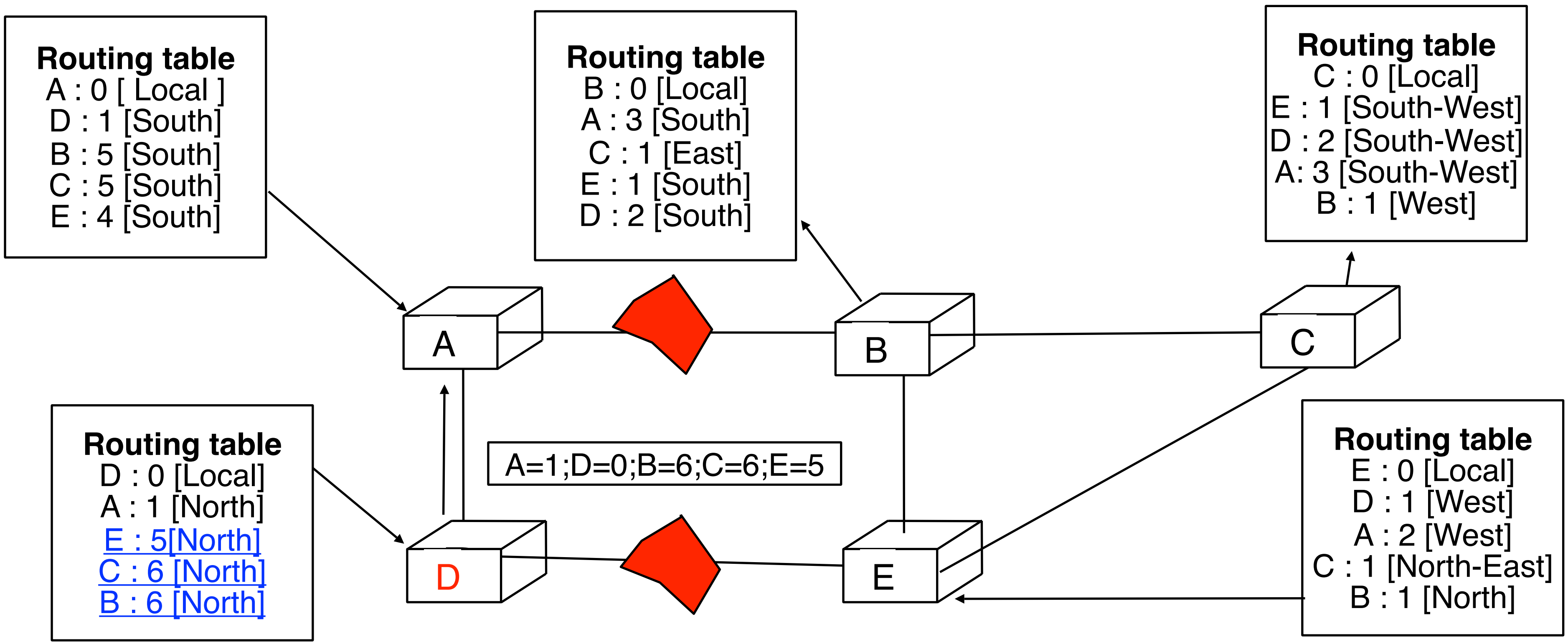


Second link failure (5)



- This problem is called counting to infinity
  - How can we avoid it ?

Second link failure (5)



- This problem is called counting to infinity
  - How can we avoid it ?

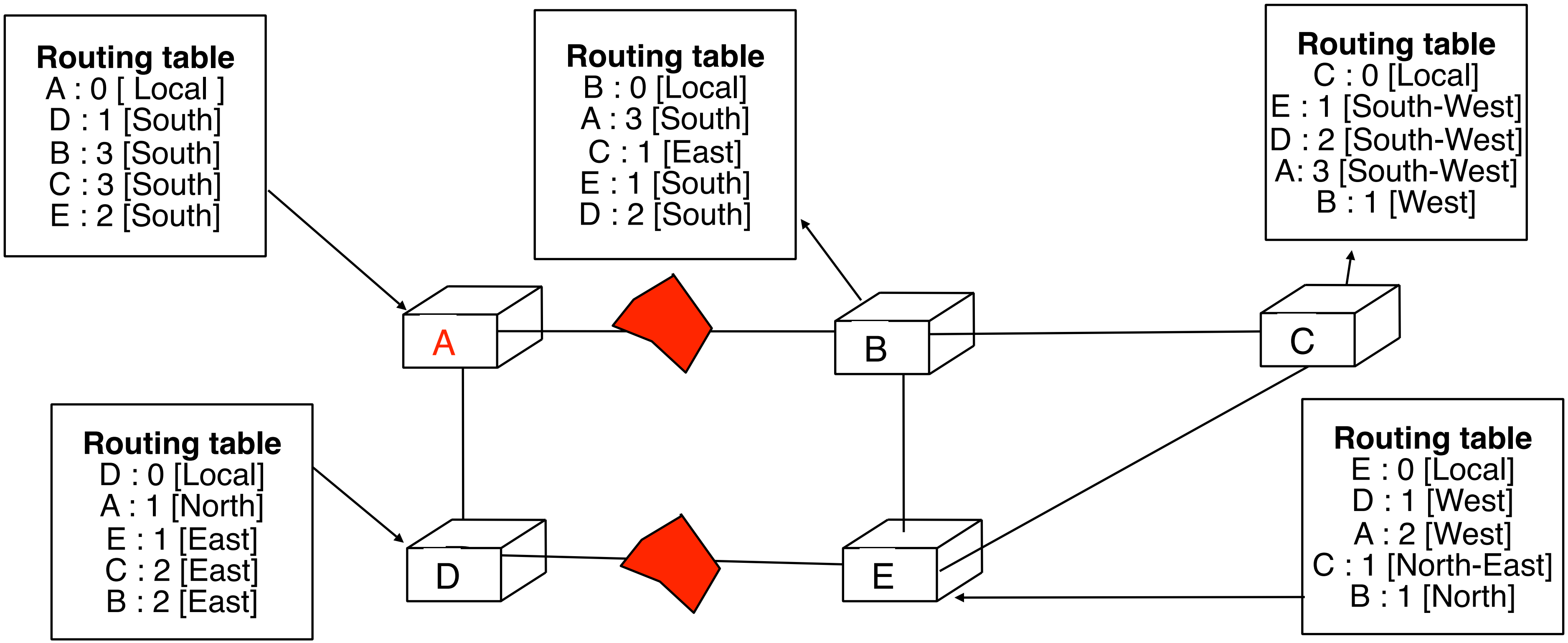
Second link failure (6)

- Where does counting to infinity comes form ?
  - A router announces on a link routes that it has already learned via this link
- How to avoid counting to infinity ?
  - split horizon
    - each router creates a distance vector <u>for each link</u>
      - on link i, router does not announce the routers learned over link i

```
Pseudocode
Every N seconds:
 for each link=l
 { /* one different vector for each link */
  Vector=null;
  for each destination=d in R[]
  {
   if (R[d].link<>l)
    { Vector=Vector+Pair(d,R[d].cost); }
  }
  Send(Vector);
 }
```

Split horizon

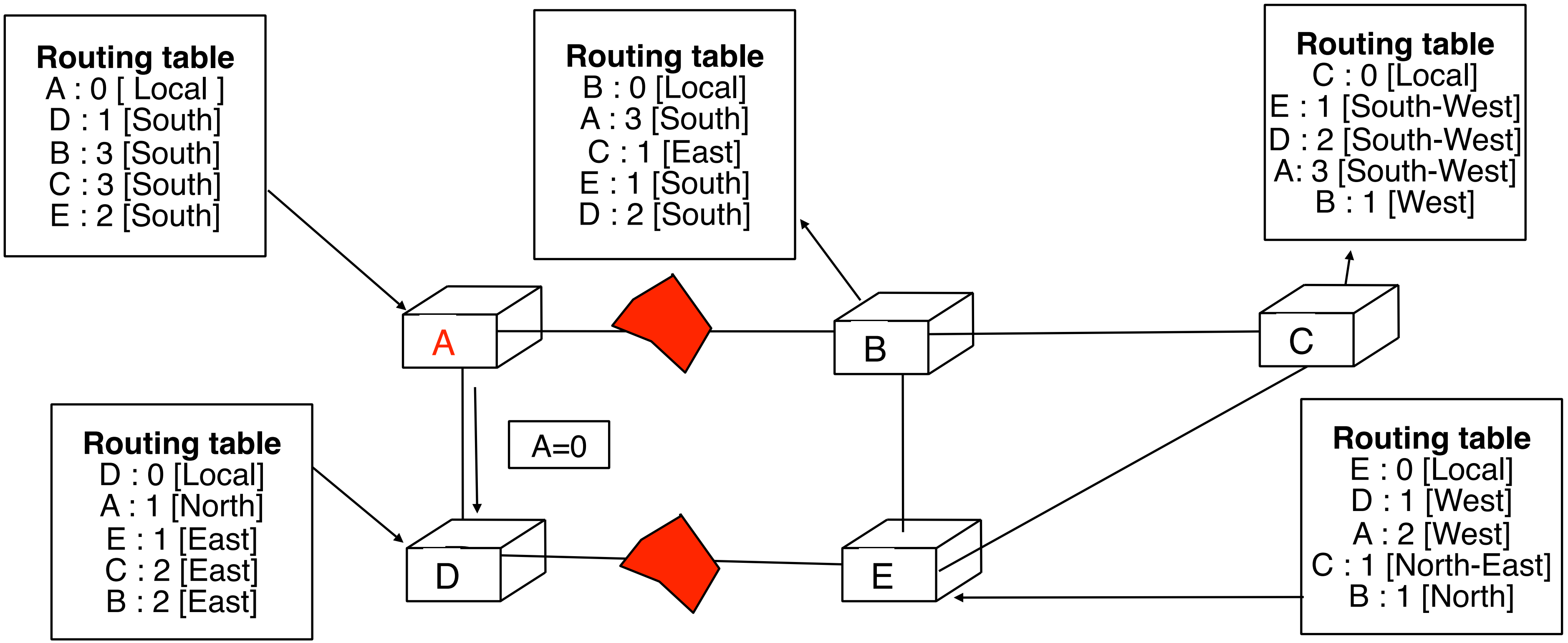
- Back to previous example



- A will not pollute D’s routing table with split horizon

Split horizon

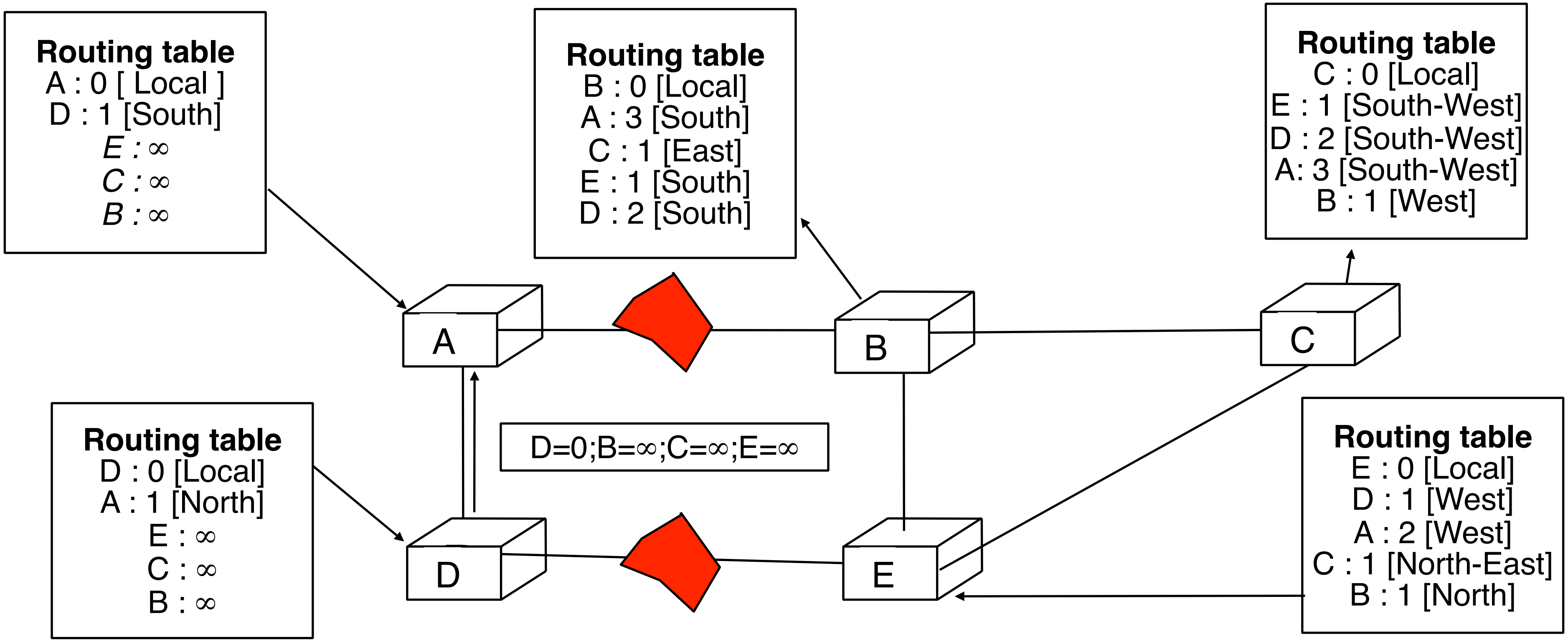
- Back to previous example



- A will not pollute D's routing table with split horizon

Split horizon (2)

- D can also send its distance vector



- Does split horizon allows to avoid all counting to infinity problems ?

Split horizon with poisoning

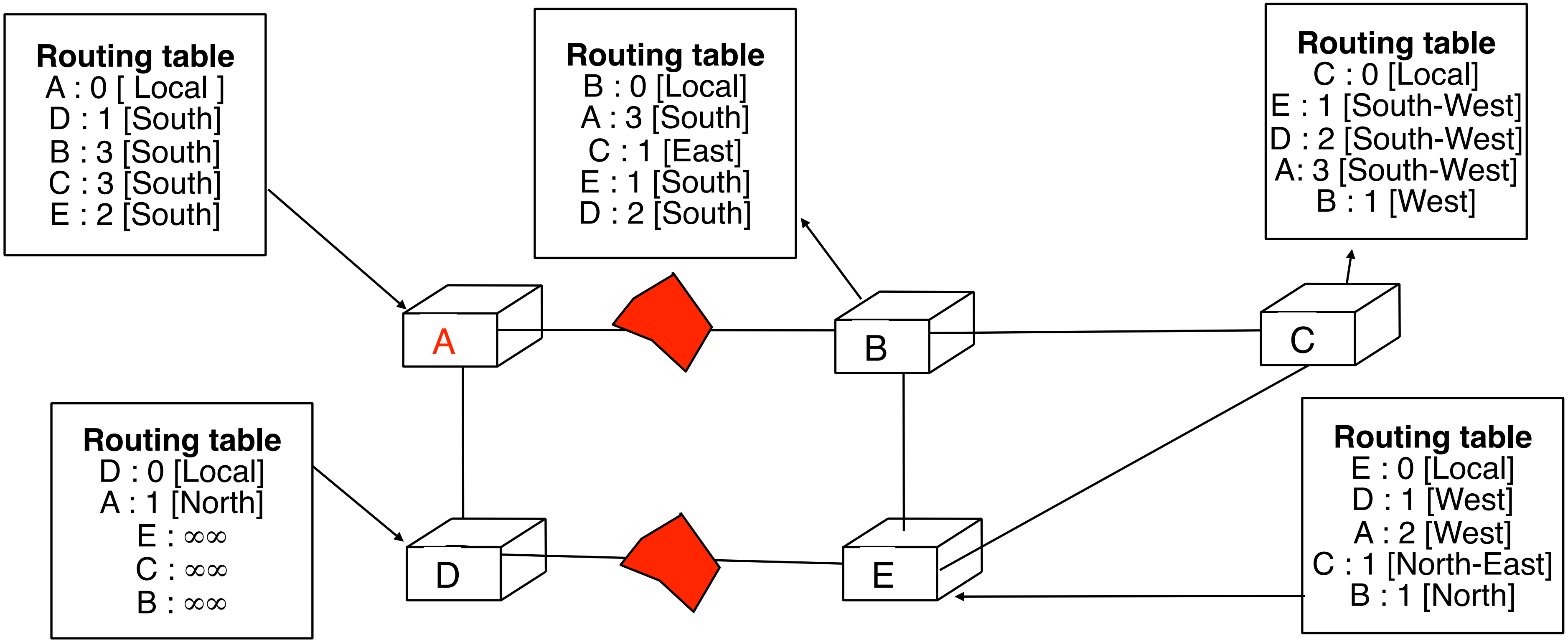
- Improvement
  - Instead of not advertising a route over the link from which it was learned, advertise it with an infinite cost

```
Pseudocode

Every N seconds:
 for each link=l
 { /* one different vector for each link */
  Vector=null;
  for each destination=d in R[]
  {
   if (R[d].link<>l)
   {
    Vector=Vector+Pair(d,R[d].cost);
   }
   else
   {
    Vector=Vector+Pair(d,∞);
   }
  }
  Send(Vector);
 }
```

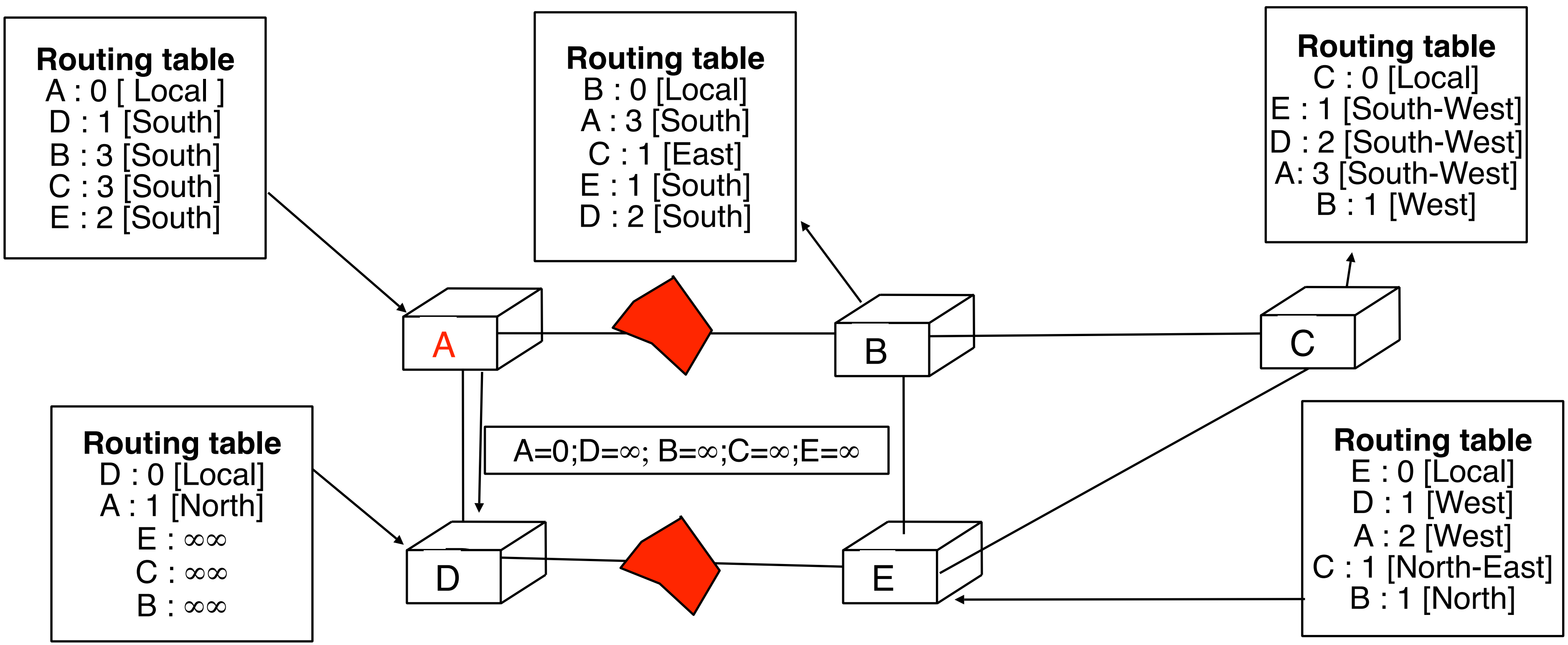
Split horizon with poisoning (2)

- Back to previous example

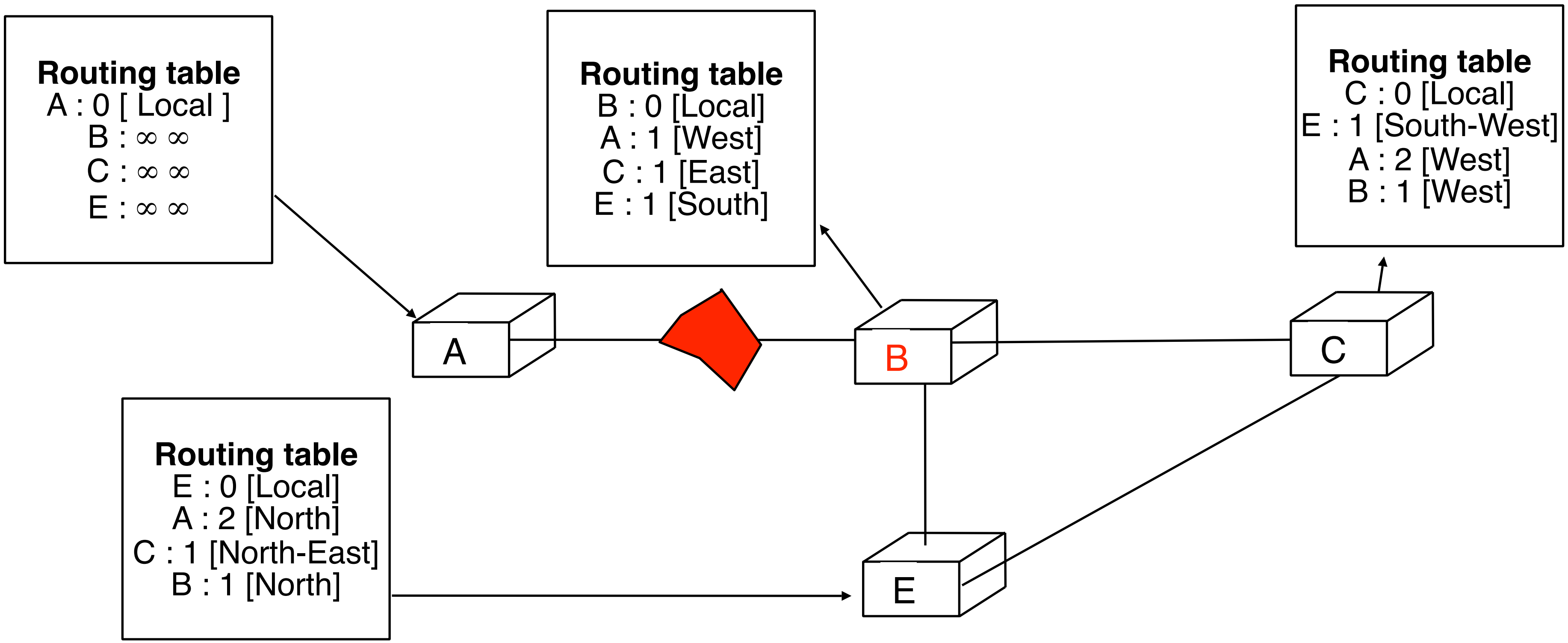


Split horizon with poisoning (2)

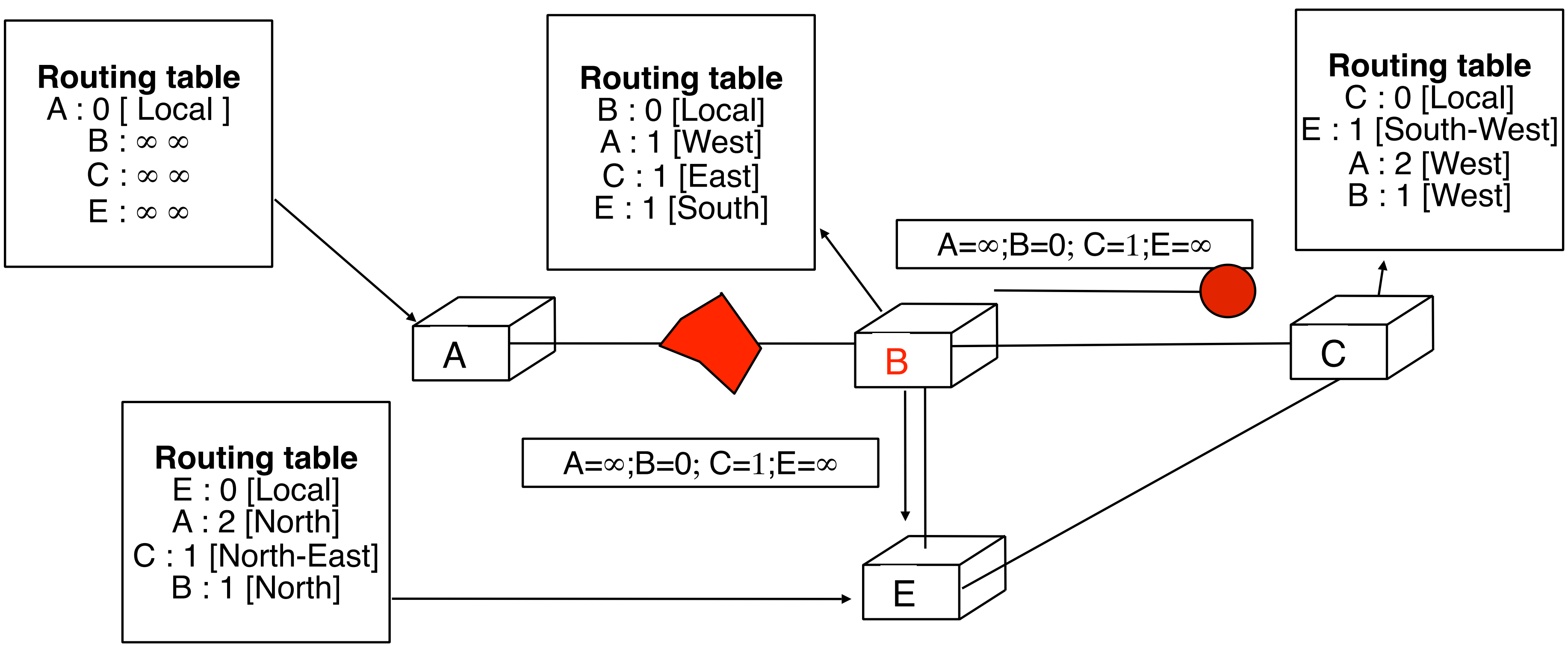
- Back to previous example



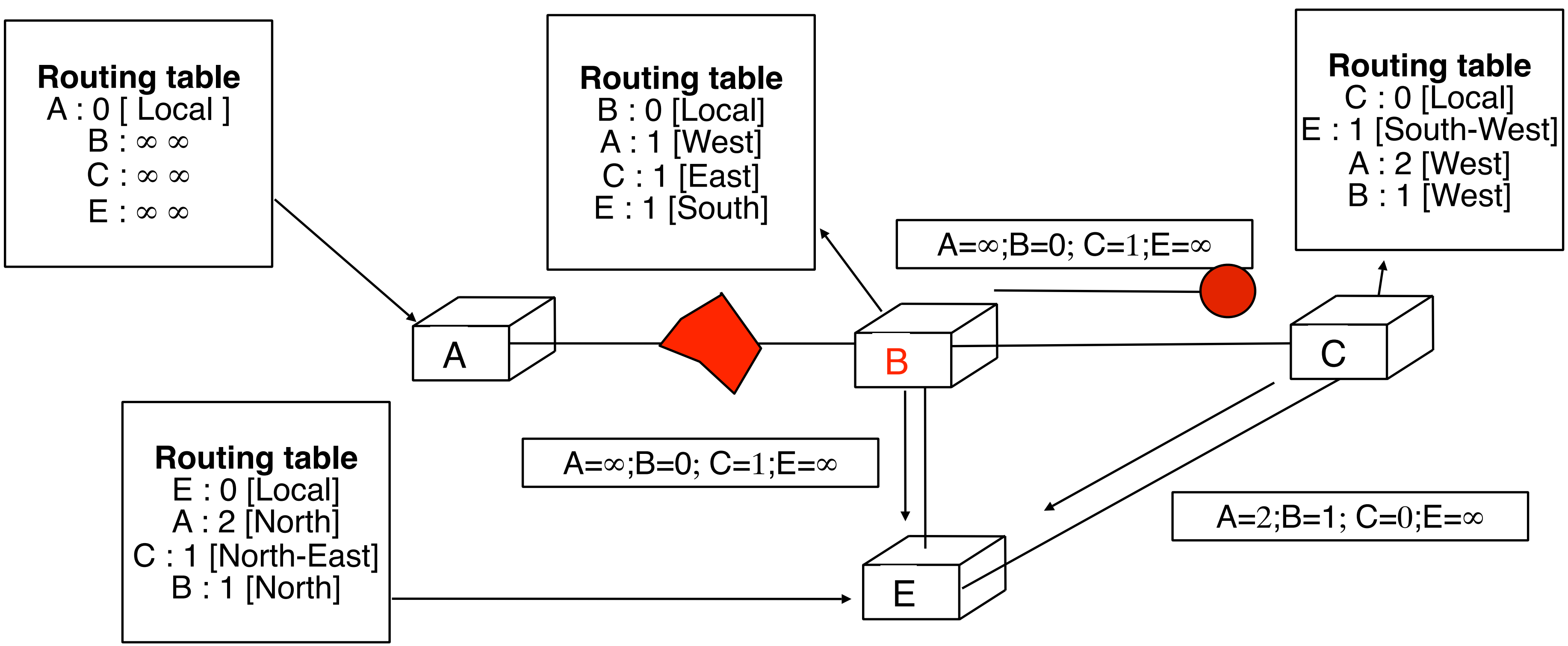
Limitations to split horizon



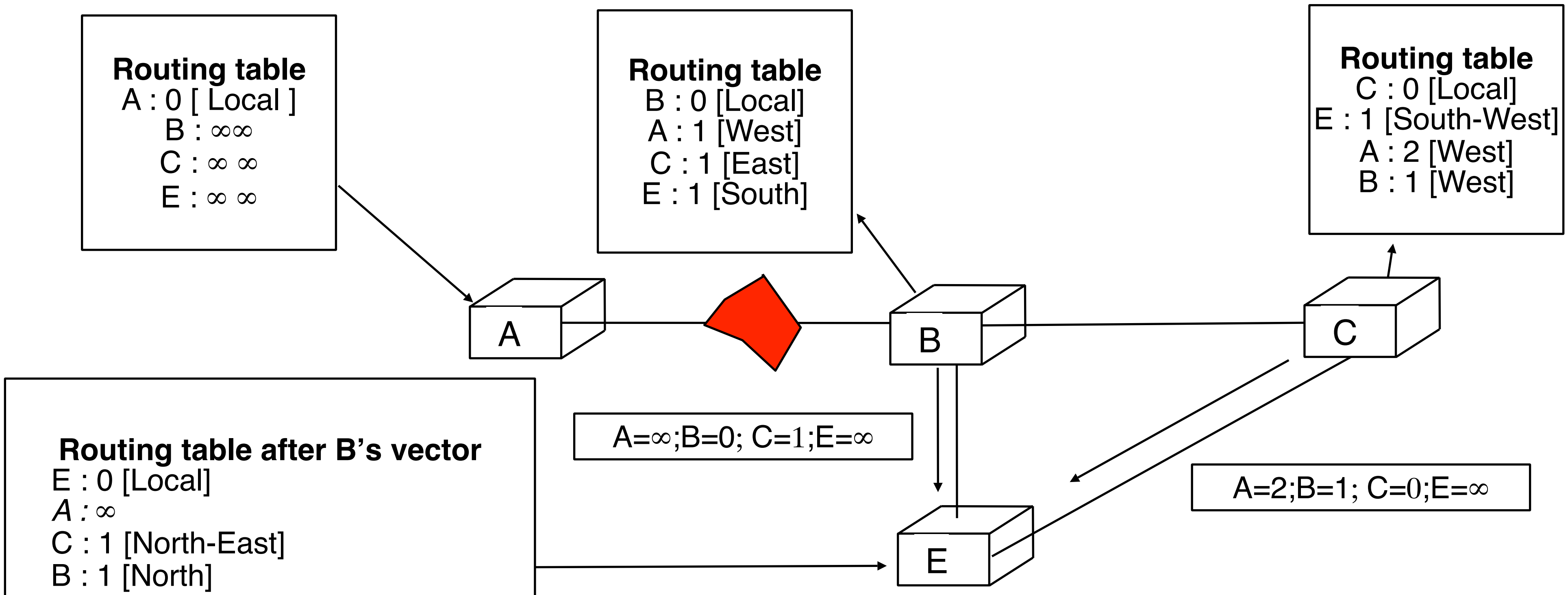
Limitations to split horizon



Limitations to split horizon



Limitations to split horizon (2)



- E will send its own distance vector
  - B will discover a new route towards A via E and will advertise it to C
- New count to infinity problem

Network layer

- Basics
- Routing
  - Static routing
  - Distance vector routing
  - Link state routing
- IP : Internet Protocol
- Routing in IP networks

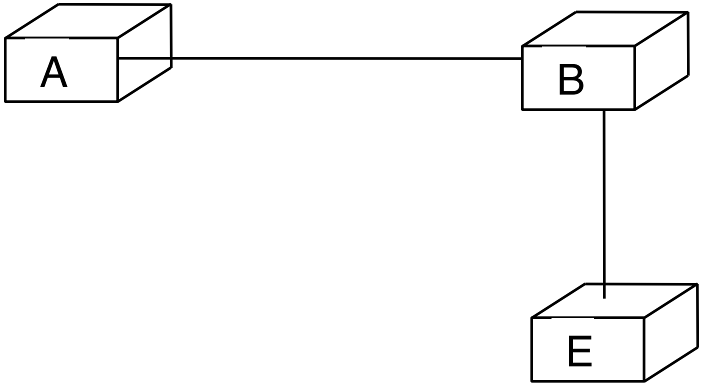
Link state routing

- Idea
  - Instead of distributing summaries of routing tables, wouldn't it be better to distribute network map ?

- How to build such as network map ?
  - Each router must discover its neighbours
  - It should be possible to associate a cost to each link since all links are not equal
  - Each router sends its local topology to all routes and assembles the information received from other routers
  - Routers build the network graph and used Dijkstra's algorithm to compute shortest paths

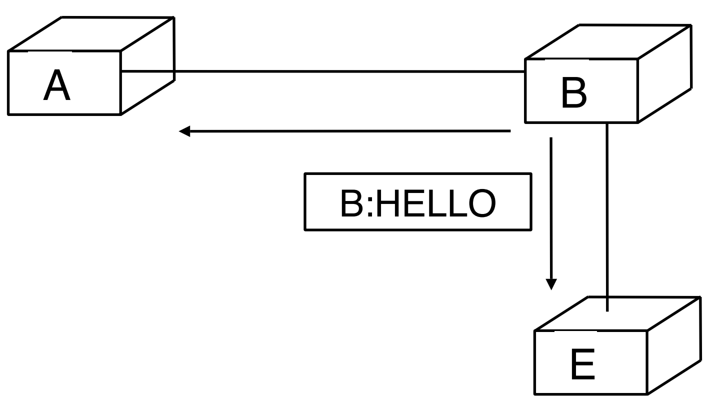
Neighbour discovery

- How does a router discover its neighbours ?
  - By manual configuration
    - Unreliable and difficult to manage
  - By using HELLO packets
    - Every N seconds, each router sends a HELLO packet on each link with its address
    - Neighbours replay by sending their own address
    - Periodic transmission allows to verify that the link remains up and detect failures



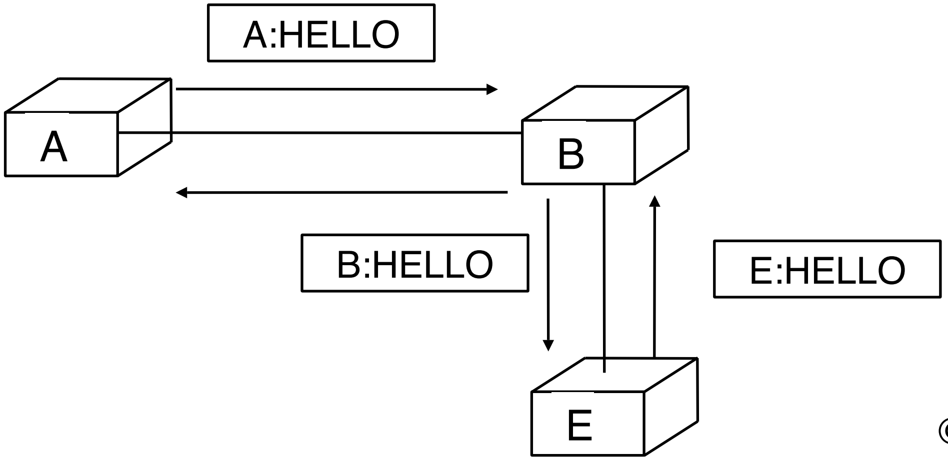
Neighbour discovery

- How does a router discover its neighbours ?
  - By manual configuration
    - Unreliable and difficult to manage
  - By using HELLO packets
    - Every N seconds, each router sends a HELLO packet on each link with its address
    - Neighbours replay by sending their own address
    - Periodic transmission allows to verify that the link remains up and detect failures



Neighbour discovery

- How does a router discover its neighbours ?
  - By manual configuration
    - Unreliable and difficult to manage
  - By using HELLO packets
    - Every N seconds, each router sends a HELLO packet on each link with its address
    - Neighbours replay by sending their own address
    - Periodic transmission allows to verify that the link remains up and detect failures



How to determine link costs ?

- Principle
  - one cost is associated with each link direction

- Commonly configured link costs
  - Unit cost
    - simplest solution but only suitable for homogeneous networks
  - Cost depends on link bandwidth
    - high cost for low bandwidth links
    - low cost for high bandwidth links
  - Cost depends on link delays
    - often used to avoid satellite links
- Cost based on measurements
  - Use HELLO to measure link rtt
    - allows to track link load, but be careful if the measurement is not stable enough as each delay change will cause a topology change ...

Assembling the network topology

- How to assemble the network topology
  - By receiving HELLOs, each routers builds its local part of the network map
  - Each router summarises its local topology inside one link state packet that contains
    - router identification
    - pairs (neighbour identification,cost to reach neighbour)
- When should a router send its link state packet ?
  - in case of modification to its local topology
    - allows to inform all other routers of the change
  - Every N seconds
    - allows to refresh information in all routers and makes sure that if an invalid information was stored on a router due to memory errors it will not remain in the router forever

How to distribute the link state packets ?

How to distribute the link state packets ?

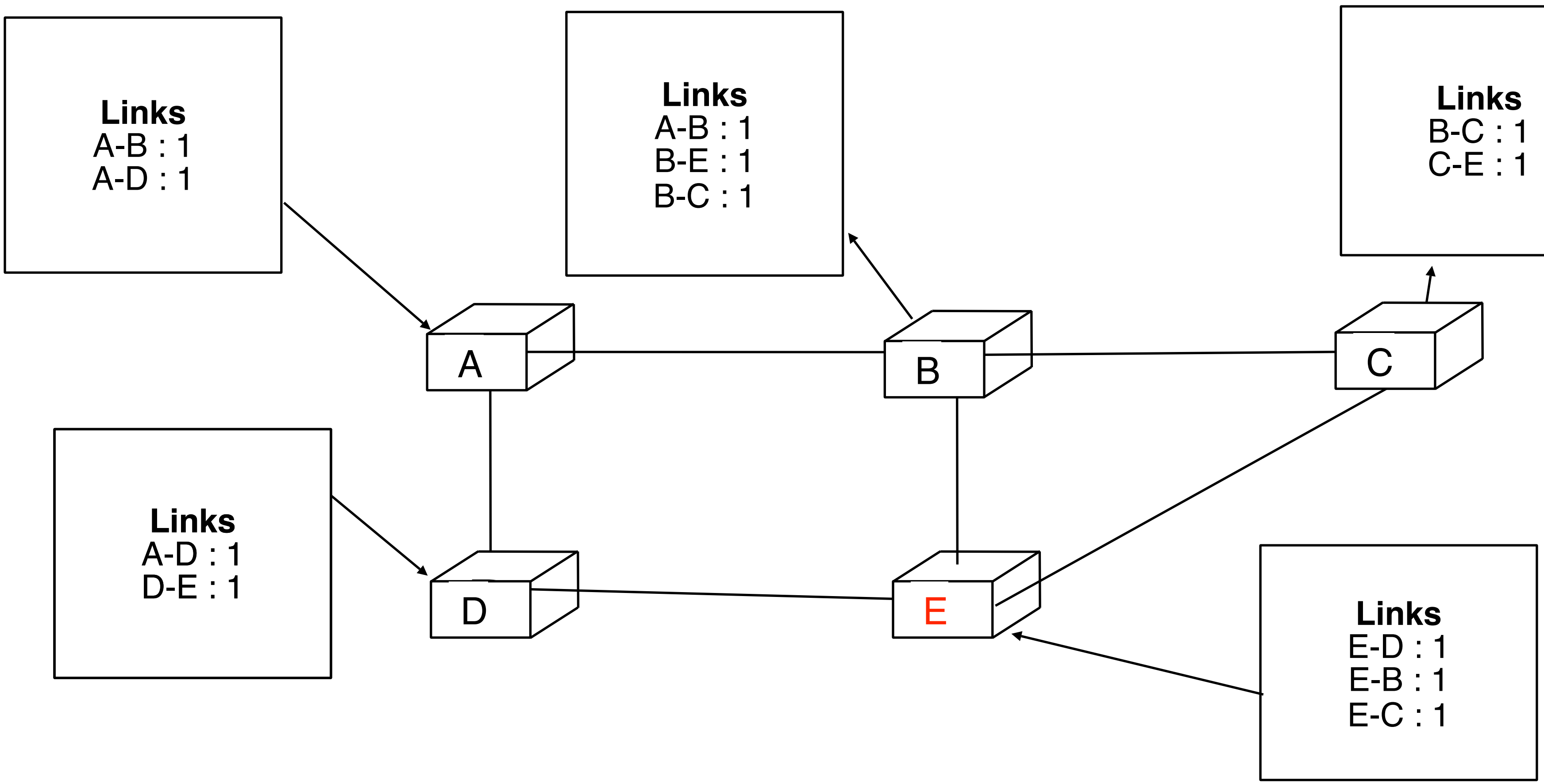
- Naive solution
  - Each router sends one packet to each other router in the network
    - This solution can only work if
      - All routers know the address of all other routers in network
      - All routers already have routing tables that allow them to forward packets to any destination

How to distribute the link state packets ?

- Naive solution
  - Each router sends one packet to each other router in the network
    - This solution can only work if
      - All routers know the address of all other routers in network
      - All routers already have routing tables that allow them to forward packets to any destination

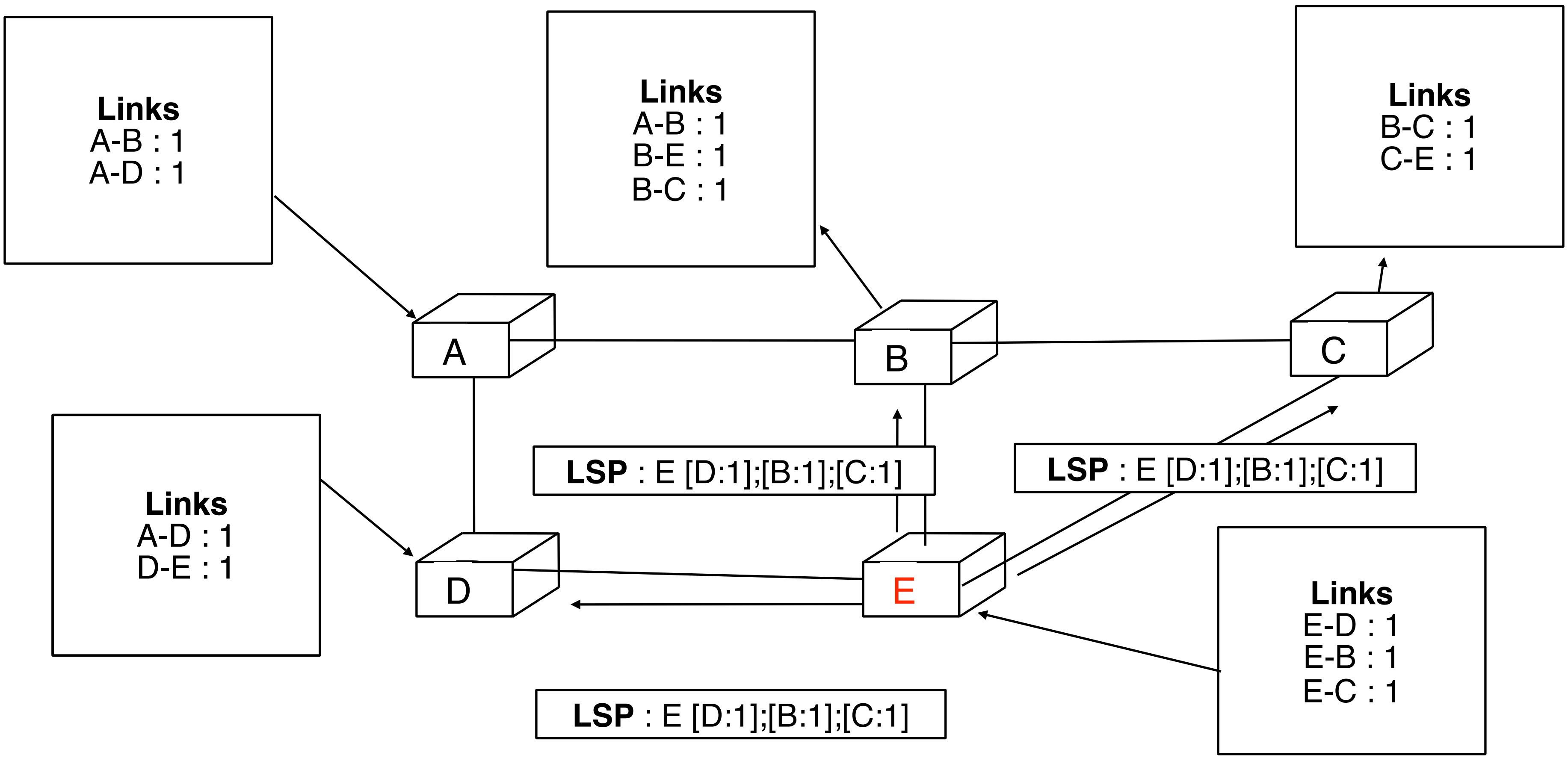
- Realistic solution
  - Does not rely on pre-existing routing tables
  - Each router must receive entire topology
  - First solution
    - Each router sends local topology in link state packet and sends it to all its outgoing links
    - When a router receives an LSP, it forwards it to all its outgoing links except the link from which it received it

LSP flooding



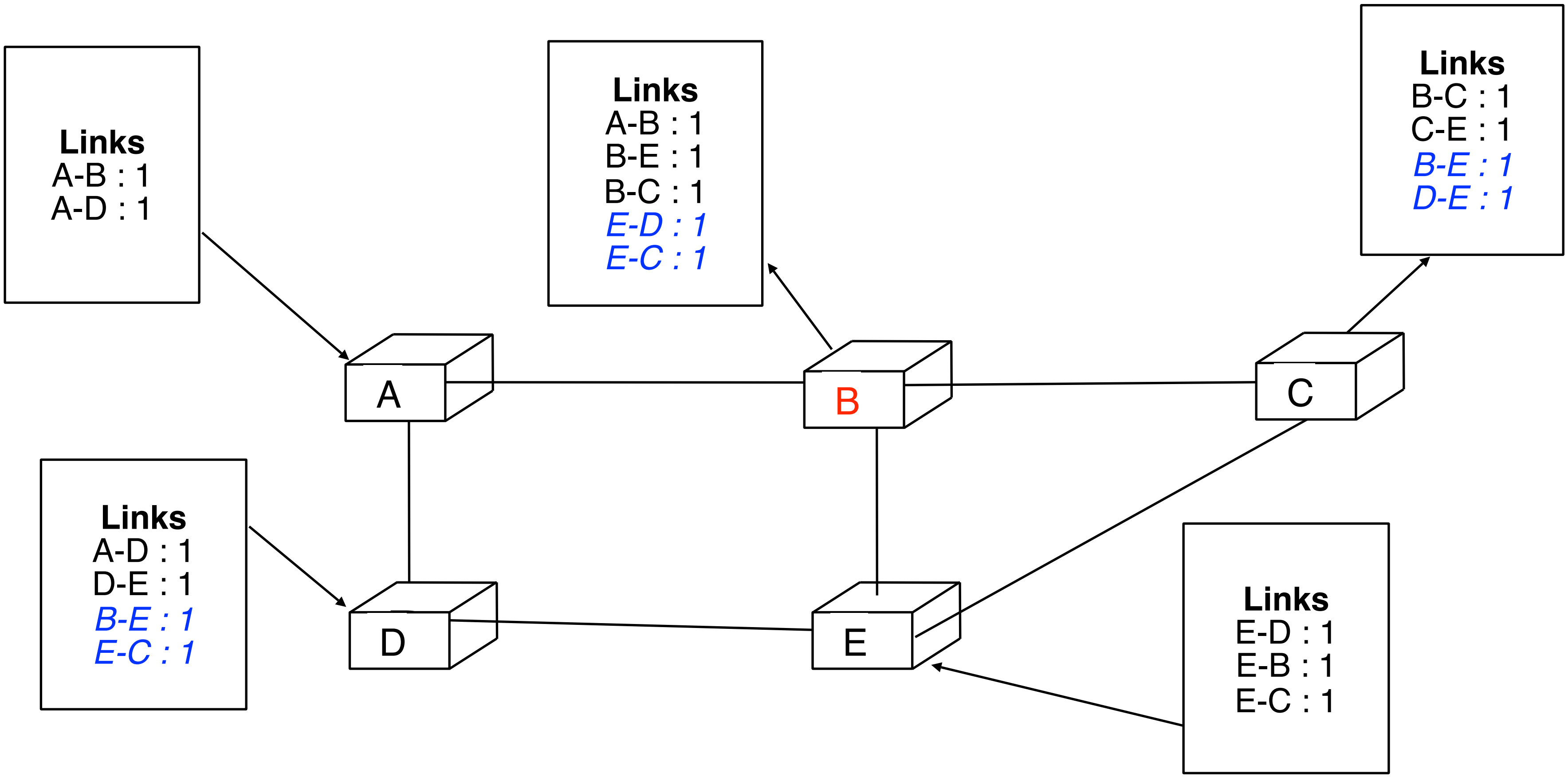
- Assumes that all links have a unit cost

LSP flooding

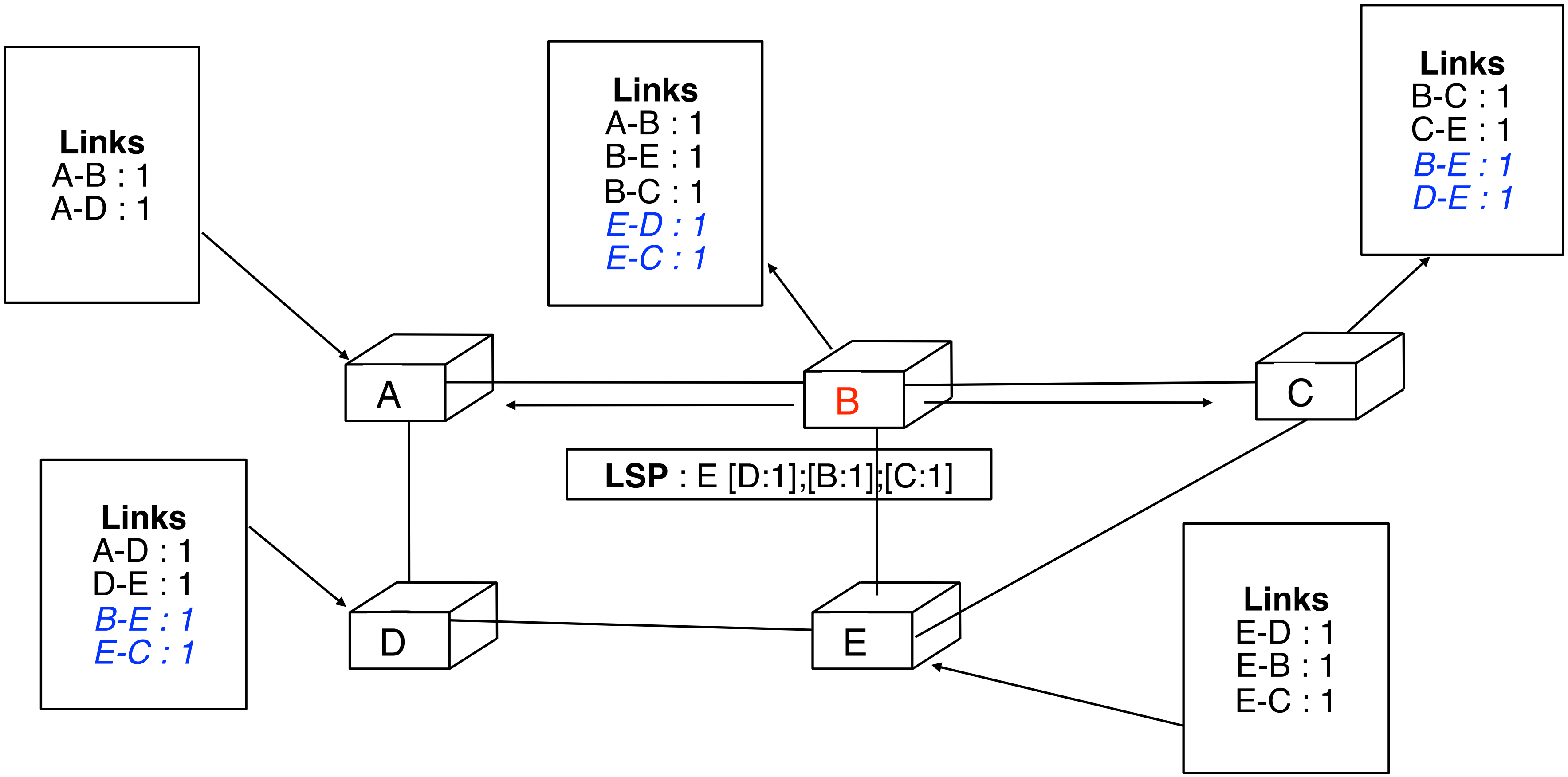


- Assumes that all links have a unit cost

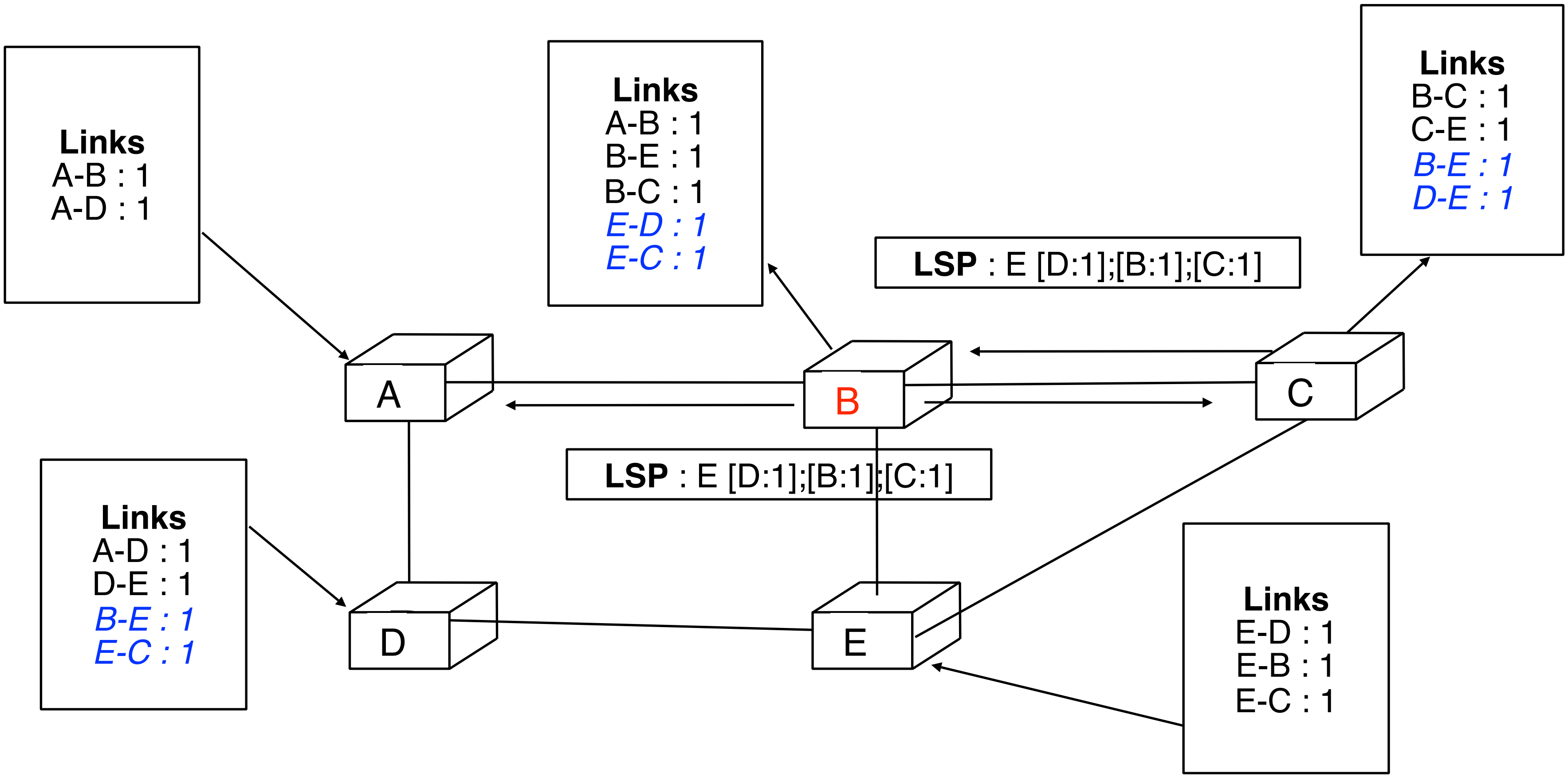
LSP flooding (2)



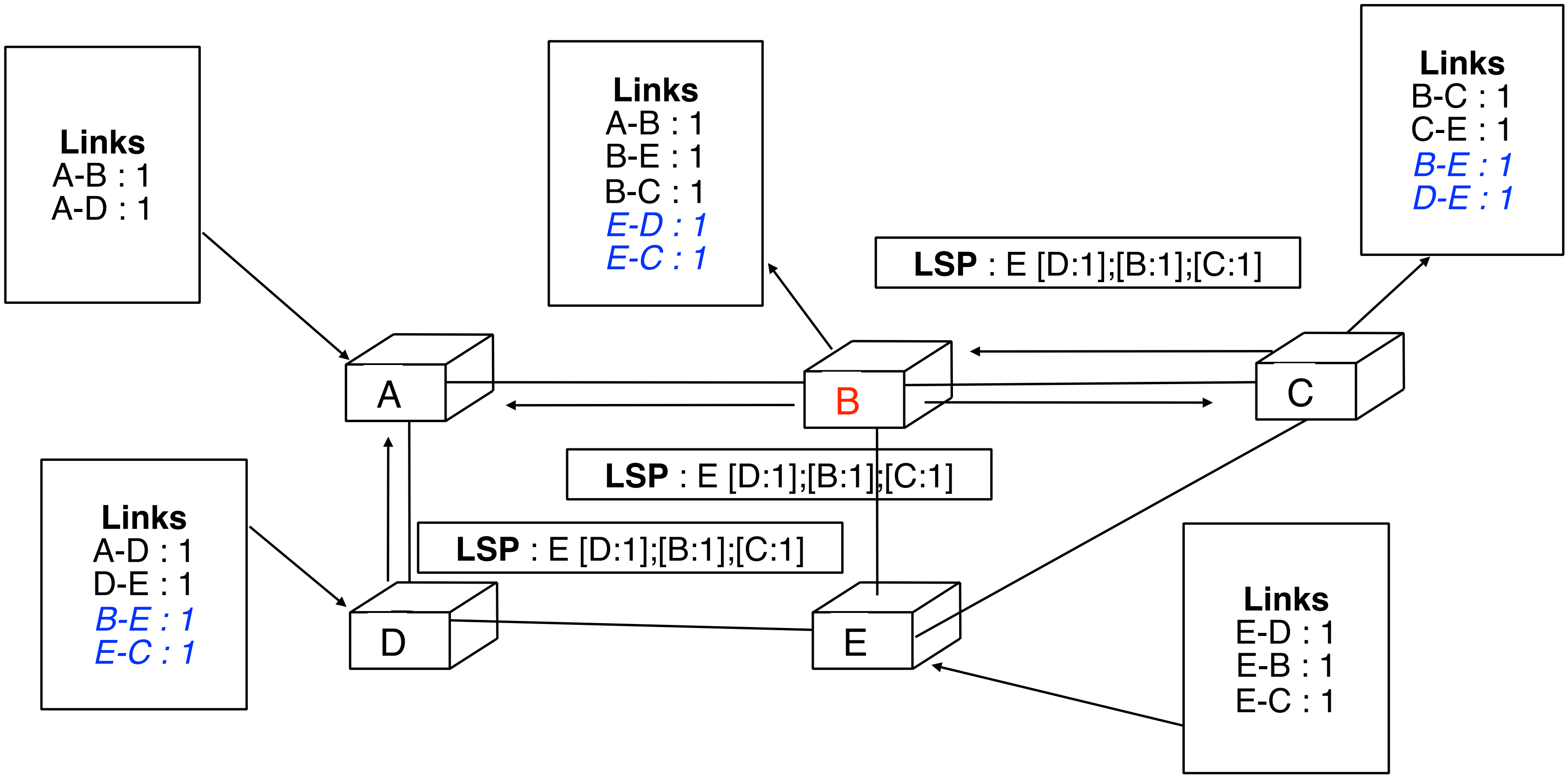
LSP flooding (2)



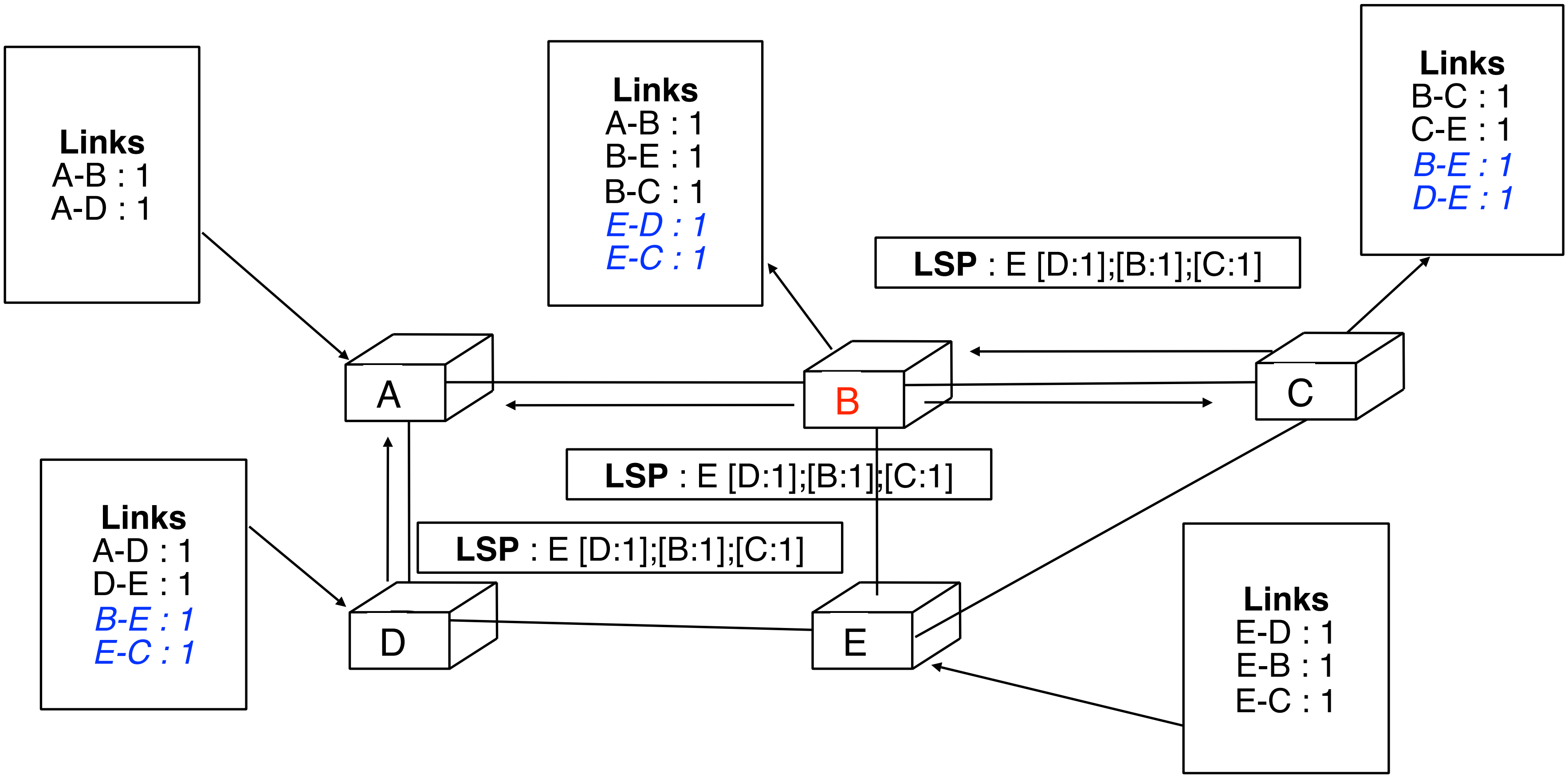
LSP flooding (2)



LSP flooding (2)



LSP flooding (2)

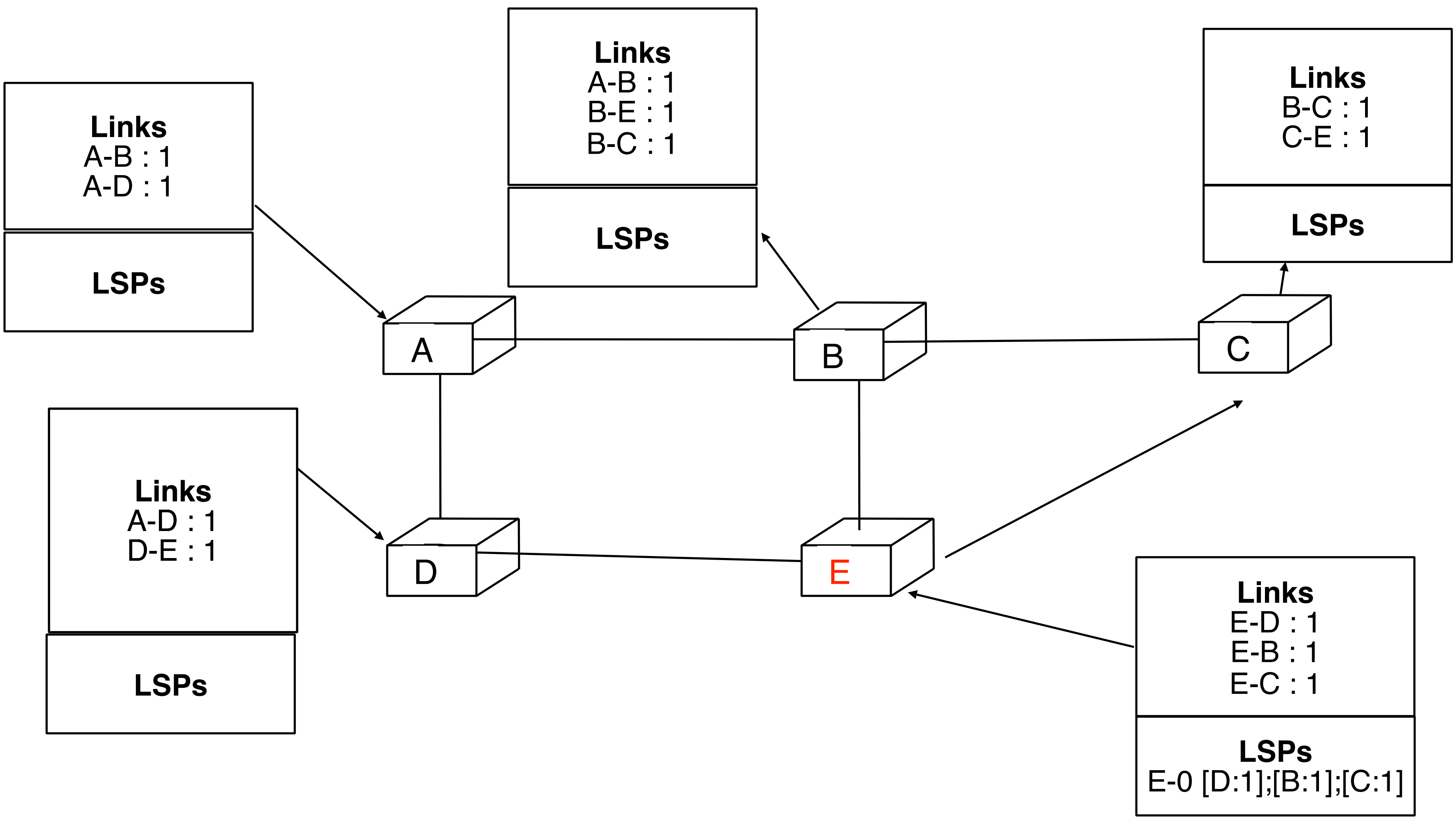


- How to ensure that an LSP will not loop ?

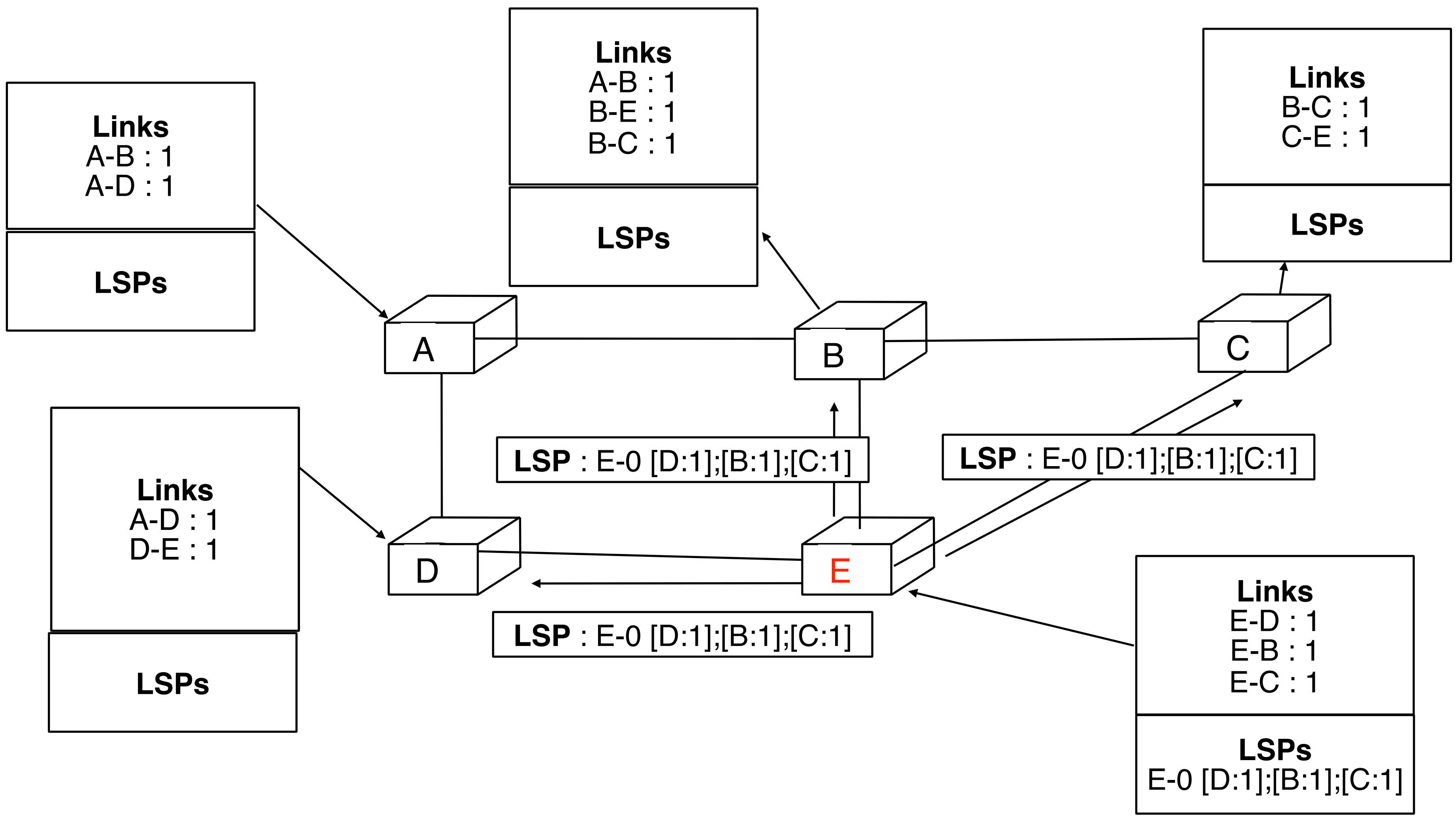
LSP flooding (3)

- How to avoid LSP flooding loops ?
  - A router should not reflood an LSP that it has already and flooded
- Solution
  - LSP contents
    - sequence number
      - incremented every time an LSP is generated by a router
    - address of LSP originator
    - pairs address:distance for all neighbours of the originator
  - Each router must store the last LSP received from each router of the network
  - A received LSP is processed and flooded only if is it more recent than the LSP stored in the LSDB

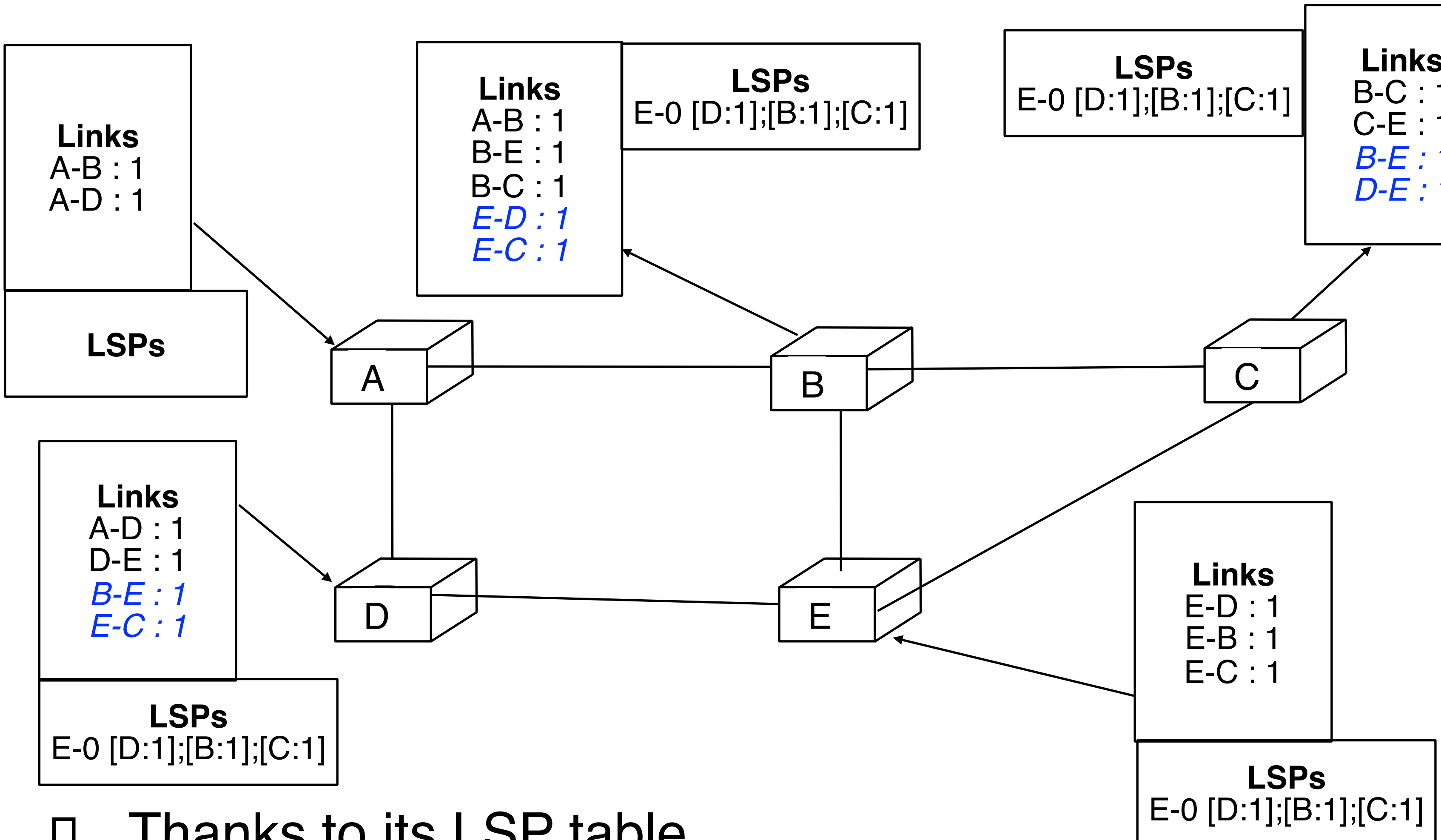
LSP flooding (4)



LSP flooding (4)

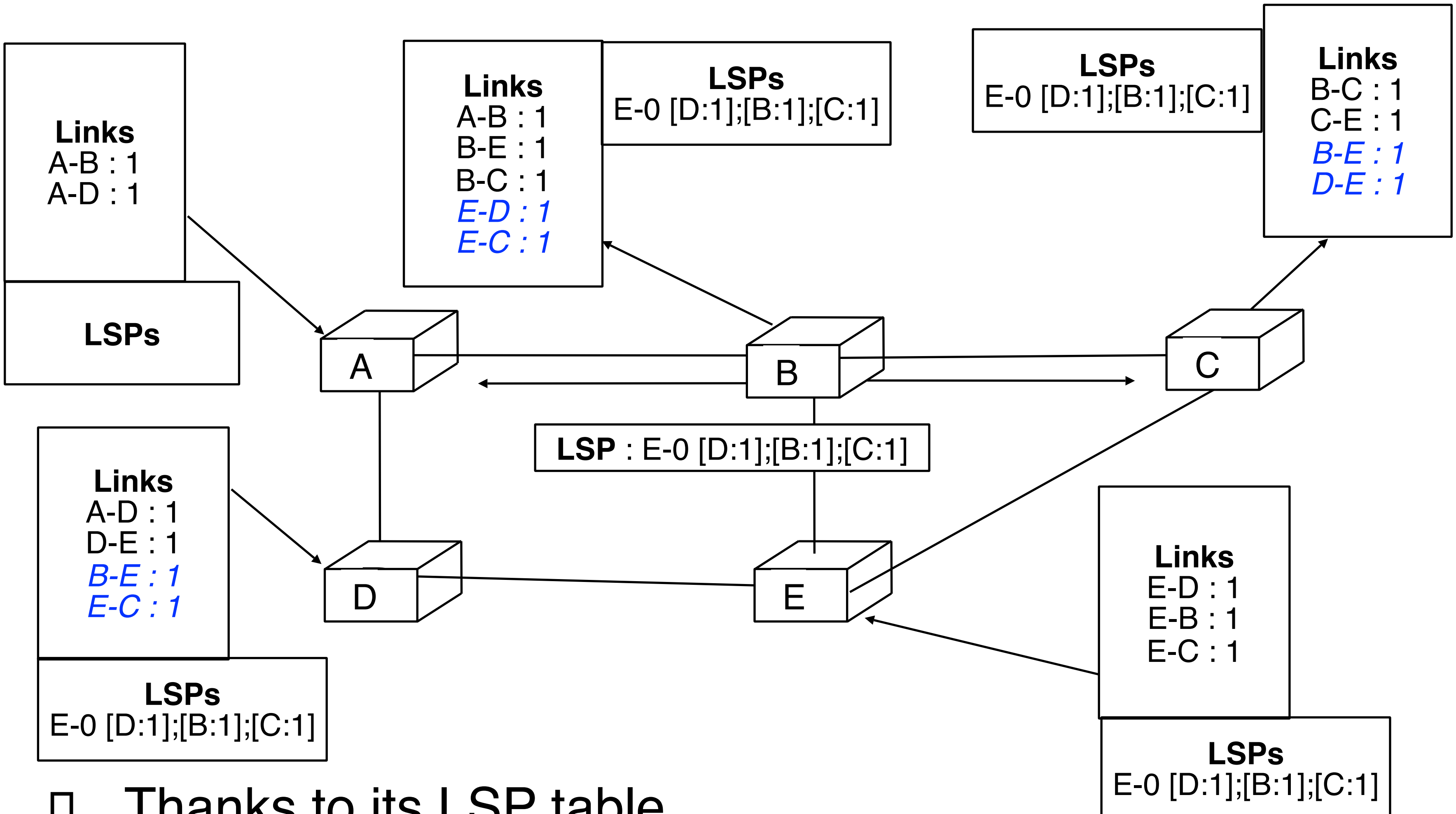


LSP flooding (5)



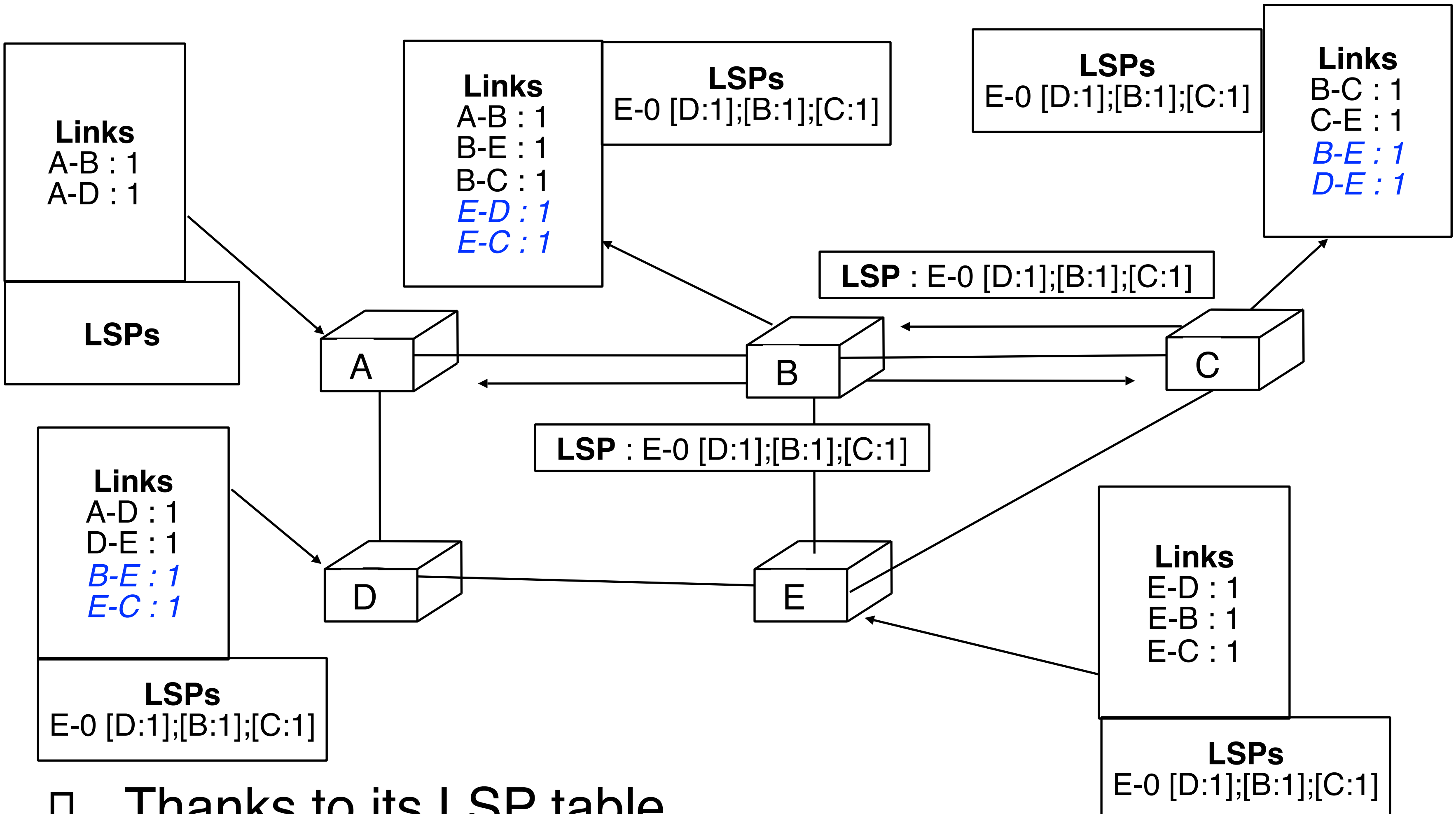
- Thanks to its LSP table
  - A can detect that it received same LSP via B and D
  - C can detect that it received same LSP via B and E

LSP flooding (5)



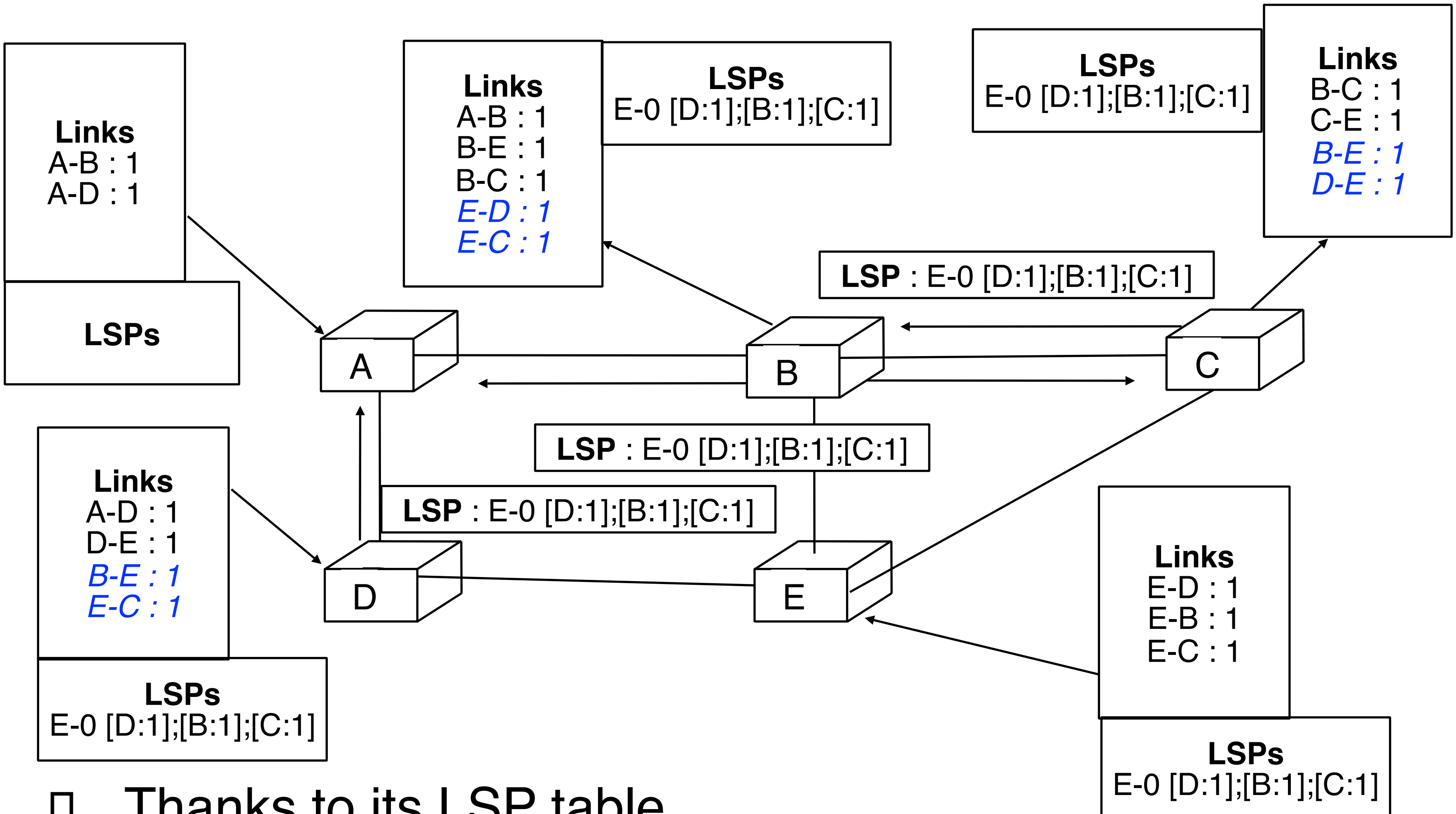
- Thanks to its LSP table
  - A can detect that it received same LSP via B and D
  - C can detect that it received same LSP via B and E

LSP flooding (5)



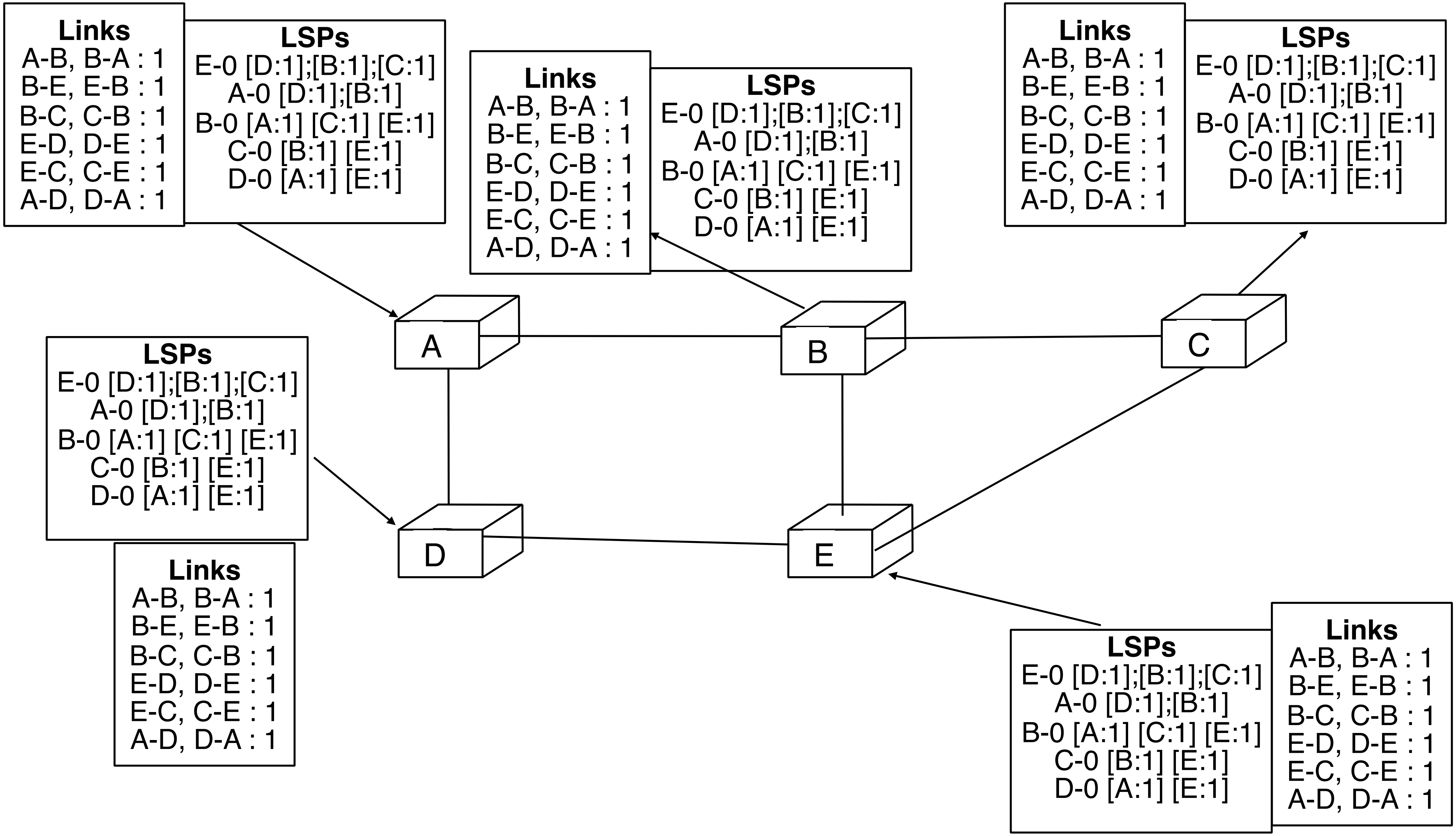
- Thanks to its LSP table
  - A can detect that it received same LSP via B and D
  - C can detect that it received same LSP via B and E

LSP flooding (5)

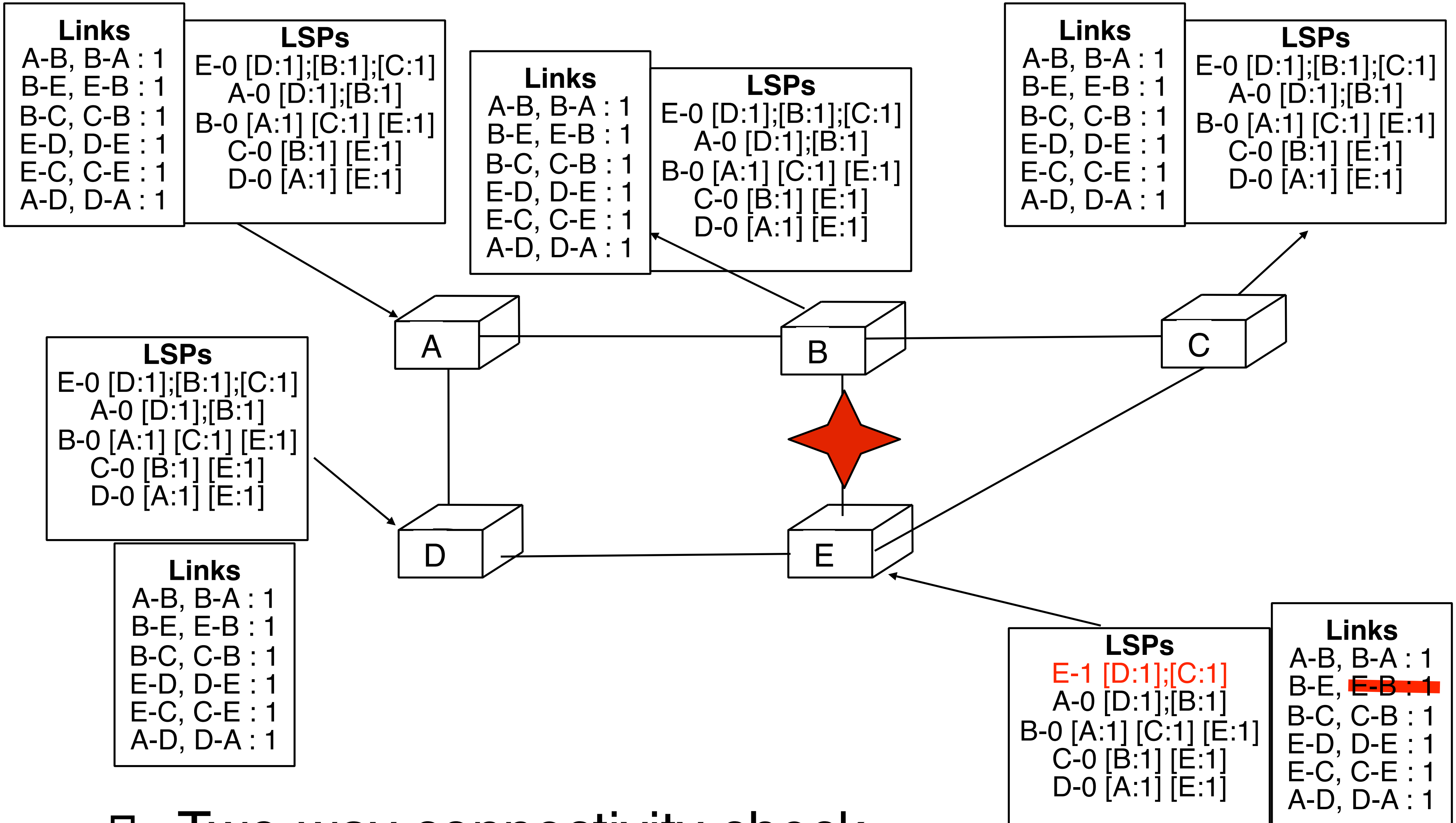


- Thanks to its LSP table
  - A can detect that it received same LSP via B and D
  - C can detect that it received same LSP via B and E

Full topology

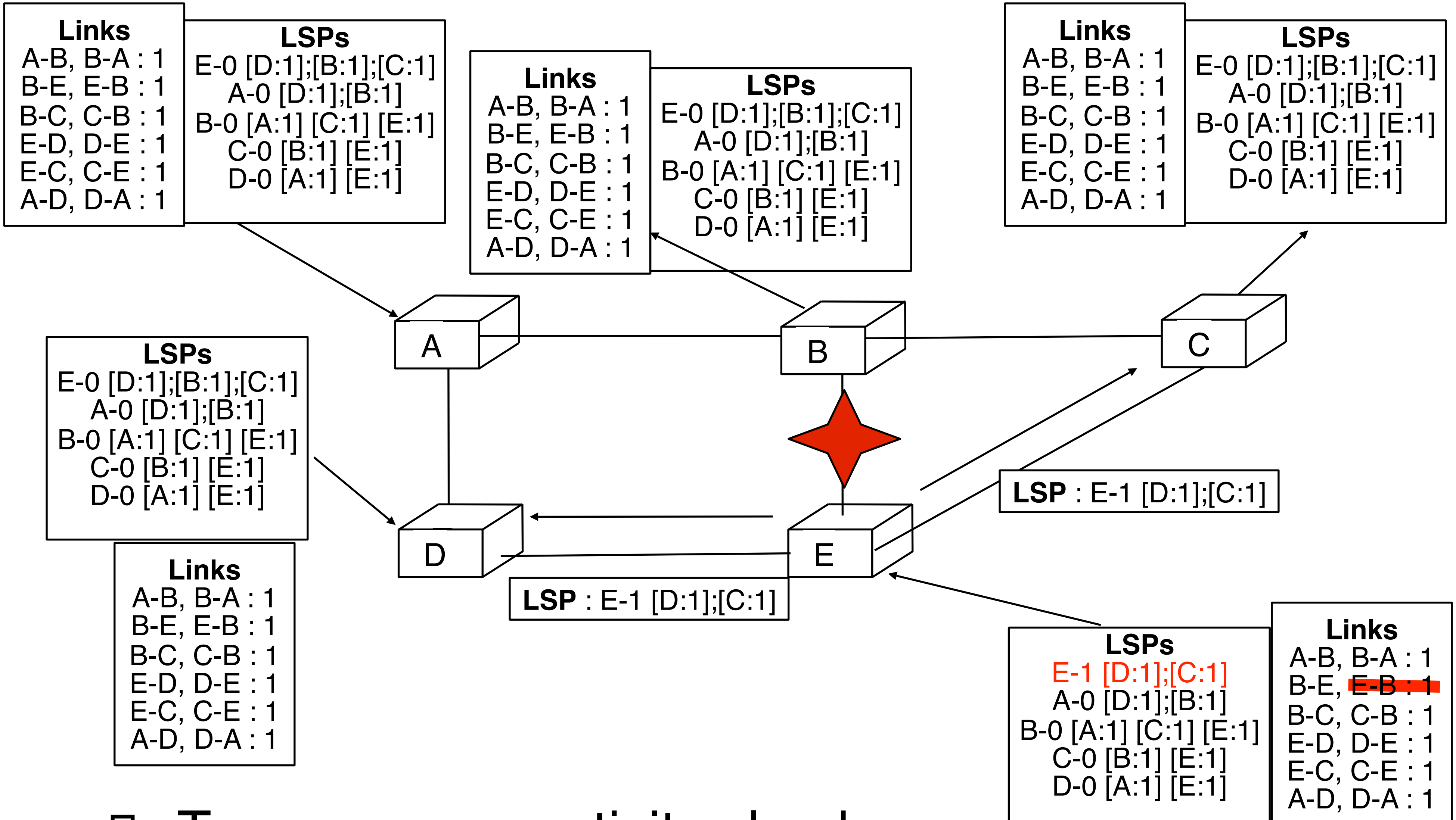


How to deal with link failures ?



- Two-way connectivity check
  - A link is only considered useable if both directions have been advertised

How to deal with link failures ?



- Two-way connectivity check
  - A link is only considered useable if both directions have been advertised

Router failures

Router failures

- What happens if a router fails ?
  - All its interfaces become unusable and do not reply anymore to HELLO packets

Router failures

- What happens if a router fails ?
  - All its interfaces become unusable and do not reply anymore to HELLO packets
- What happens when the router reboots ?
  - It will send its LSP with its sequence number set to zero
    - If older LSPs from same router were still in network, then the new LSP will not be flooded

Router failures

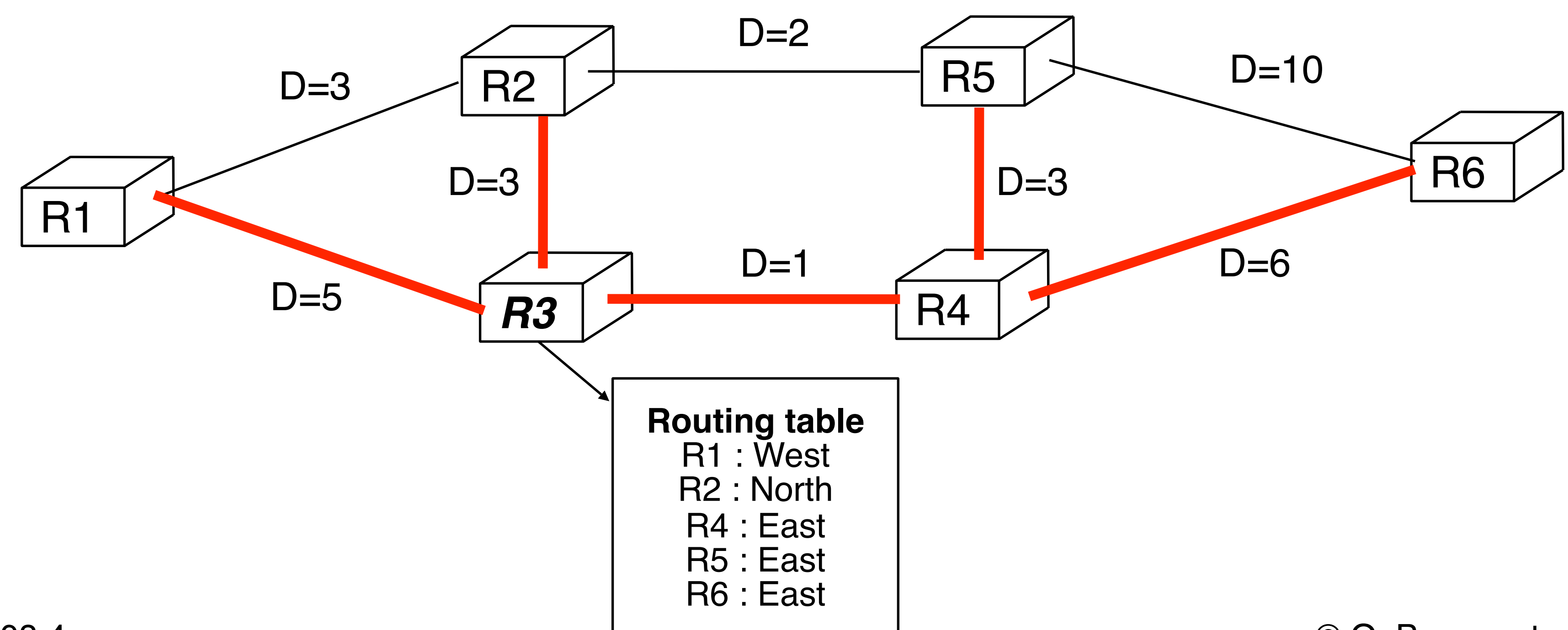
- What happens if a router fails ?
  - All its interfaces become unusable and do not reply anymore to HELLO packets
- What happens when the router reboots ?
  - It will send its LSP with its sequence number set to zero
    - If older LSPs from same router were still in network, then the new LSP will not be flooded
- Solution
  - Add "age" field inside each LSP
  - Each router must decrement age regularly
    - even for the LSPs stored in its LSDB
  - LSP having age=0 is too old and must be deleted
  - Each router must flood regularly its own LSP with age>0 to ensure that it remains inside network

Improvements to LSP flooding

- Avoid sending twice same LSP on a link
  - When an LSP needs to be flooded on a link, wait some time to let other router flood the LSP
    - reduces number of LSPs exchanged on a link but
    - increases flooding time
- Reliable flooding
  - CRC inside each LSP to detect transmission errors
  - Acknowledgements on each link for the LSPs exchanged on this link
    - each transmission is protected by a timer
- Link state database exchange/synchronisation
  - Routers can compare the content of their LSDB and exchange only missing LSPs form neighbour
    - useful when the router boots and wants to receive quickly all LSPs from the network

Computation of routing table

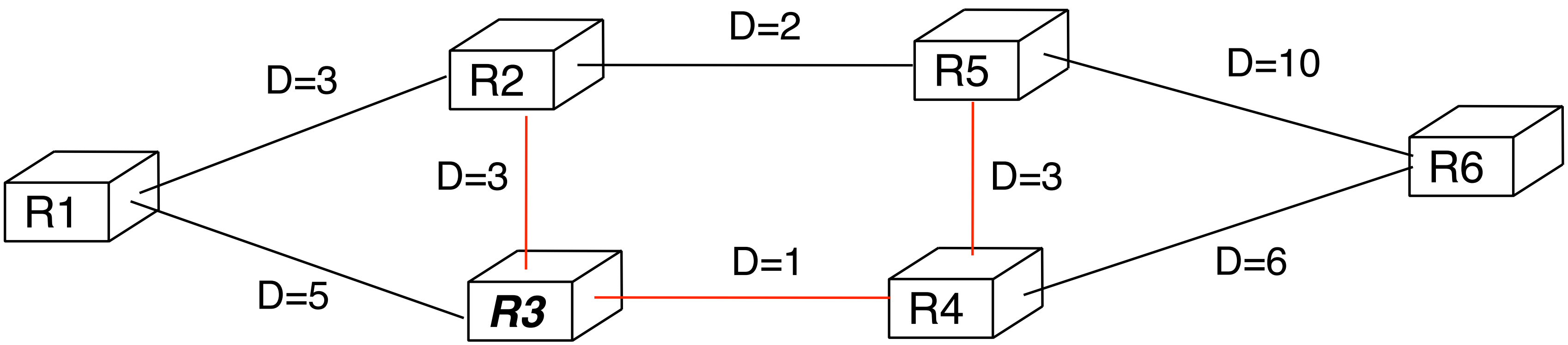
- Principle
  - Each router uses the received LSPs to build a graph and then computes the shortest spanning tree rooted on itself
    - From this spanning tree, it is easy to compute the the routing table



Dijkstra's shortest path

- Computing the shortest path tree
  - At the beginning, the tree only contains the root node
  - Adjacent routers are placed with the cost of their link in the candidates list
  - Candidate router with lowest cost is chosen and added to the tree
  - Consider the neighbours of the chosen candidate router and update the candidate router list if
    - one of the new neighbours was not already in the candidates list
    - one of the new neighbours was already in the candidates list but with a longer path than the one in the current list
  - Algorithm continues with the new candidates list and ends when all routers belong to shortest path tree

Dijkstra's shortest path (2)

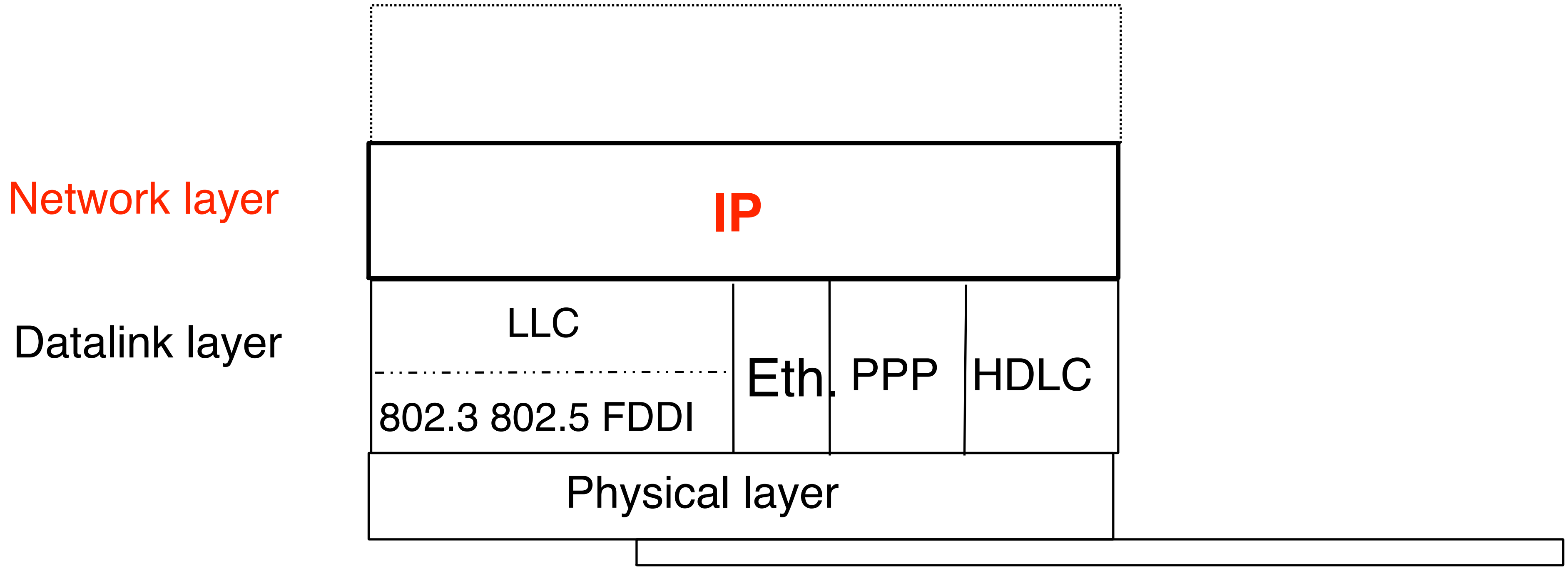


1) Routers : [R1, R2, R4, R5,R6] ; Candidates : [-] ; Tree : R3
2) Routers : [R5, R6] ; Candidates : [R1(5) ; R2(3) ; R4 (1)]
   selected candidate : R4
   New tree : R3 - R4
   New Candidates ? [R1(5) ; R2 (3) ; R5(R4-4) ; R6(R4-7)]
3) Routers [-]
   Selected candidate : R2 ; New tree : R2 - R3 - R4
   New Candidates ? [R1(5) ; R5(R4-4) ; R6 (R4-7)]
4) Selected candidate : R5 ; New tree : R2 - R3 - R4 - R5
   New candidates ? [R1(5) ; R6(R4-7)]
5) ...

Network layer

- Basics
- Routing
  - Static routing
  - Distance vector routing
  - Link state routing
- IP : Internet Protocol
  - IP version 4
  - IP version 6
- Routing in IP networks

IP : Internet Protocol

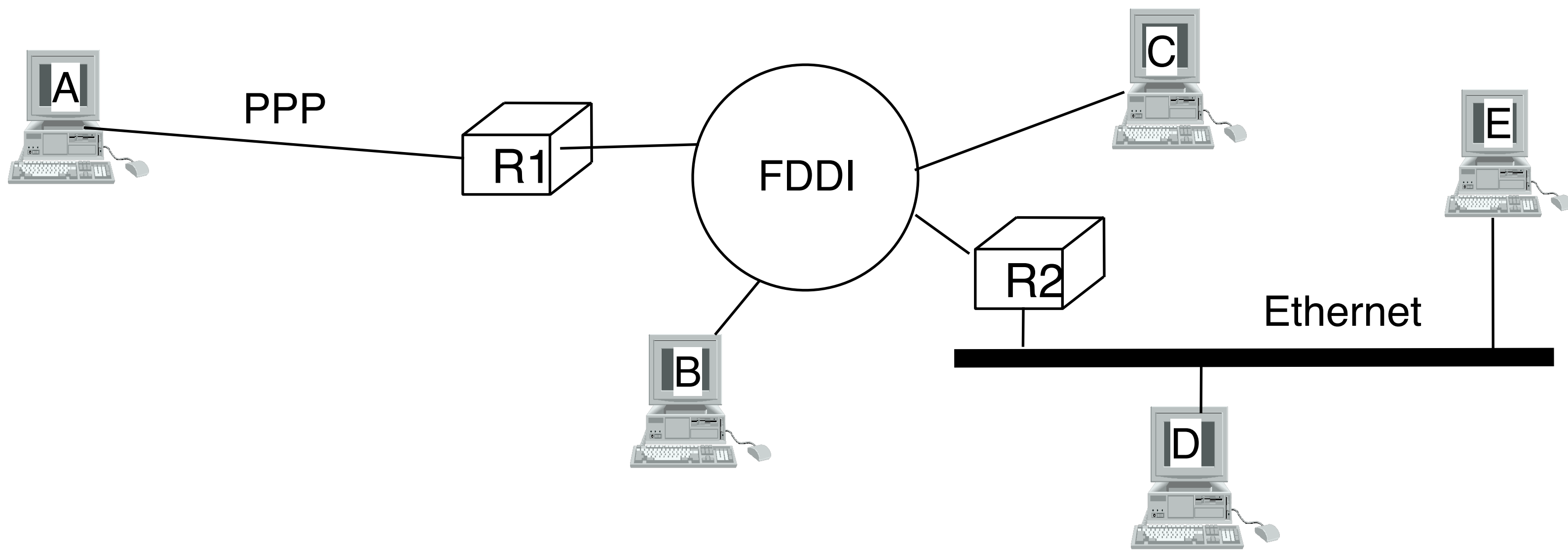


- Internet network layer
  - provides unreliable connectionless service
    - some packets can be lost
    - packets can suffer from transmission errors
    - packets can be misordered

IPv4 is defined in RFC791

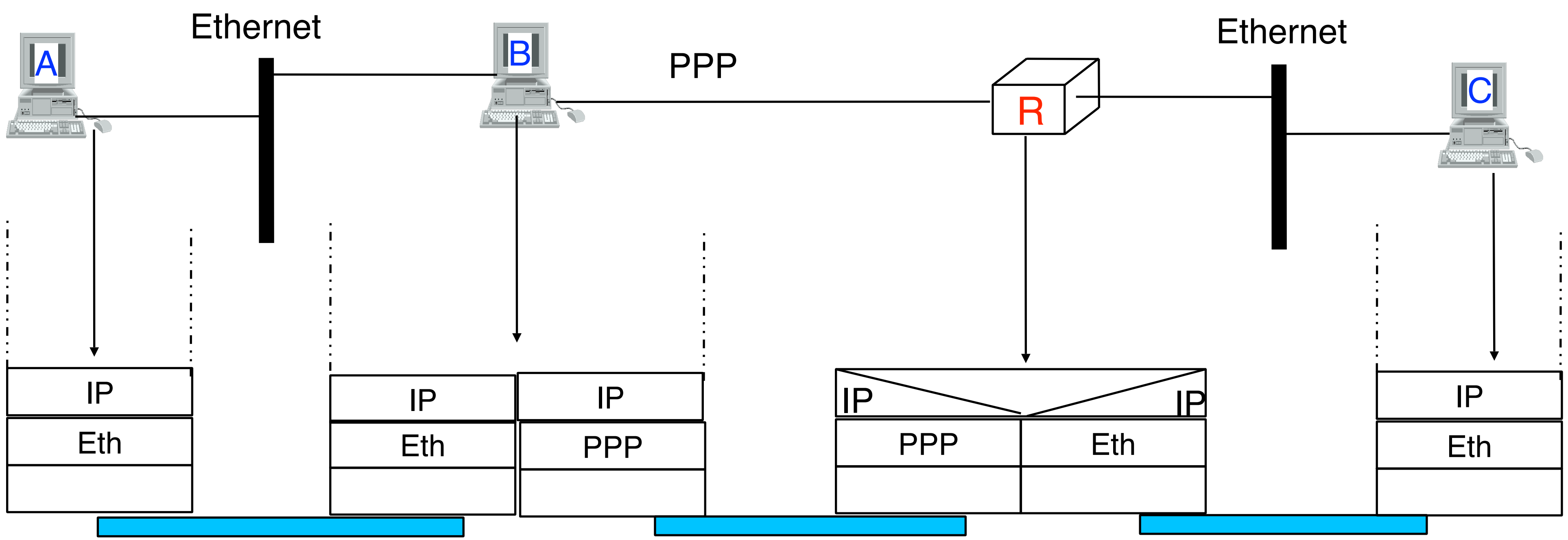
Basic principles

- Datagram mode



- Each host is identified by one IP address (encoded as 32 bits number)
- Each host knows how to reach at least one router
- Routers know how to reach other routers

Basic principles (2)



- Endhost
  - equipment able to send and receive packets originated by or destined to it
- Router
  - equipment able to send and receive packets originated by or destined to it
  - equipment able to forward toward theirs destination packets that it did not originate

IP Addressing

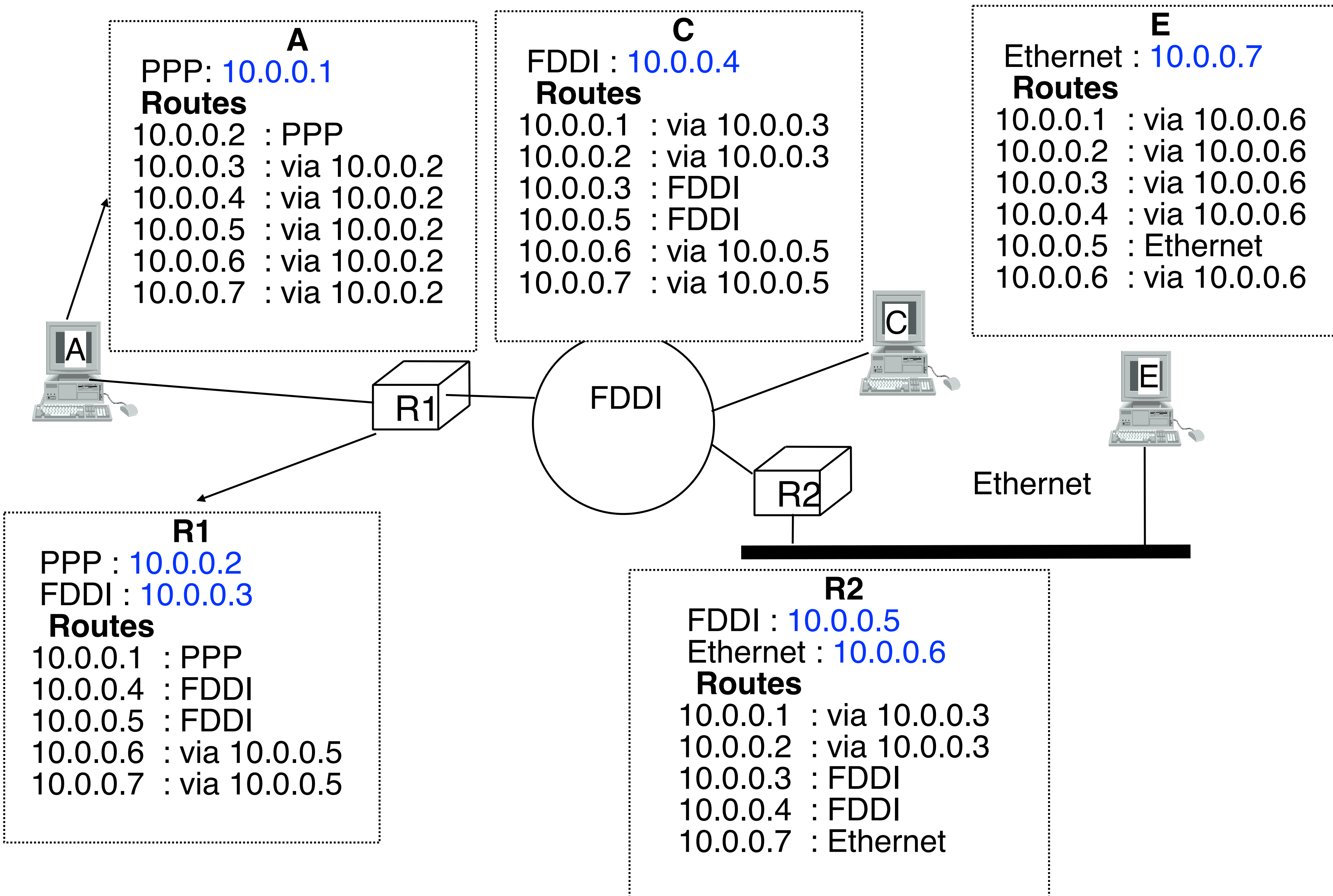
IP Addressing

- Utilisation of IP address
  - identify a host/router that implements IP
    - usually, one IP address identifies one (physical) interface on one endhost or router
      - (physical) interface is access point to datalink layer
      - usually endhosts have a single interface
      - routers have more than one interface
  - Encoding of 32 bits IP address
    - 10001010 00110000 00011010 00000001
    - **138 . 48 . 26 . 1**

IP Addressing

- Utilisation of IP address
  - identify a host/router that implements IP
    - usually, one IP address identifies one (physical) interface on one endhost or router
      - (physical) interface is access point to datalink layer
      - usually endhosts have a single interface
      - routers have more than one interface
  - Encoding of 32 bits IP address
    - 10001010 00110000 00011010 00000001
    - **138 . 48 . 26 . 1**
- How to allocate IP addresses to hosts in a campus network
    - Naive solution
      - First come first served

Naive IP addressing



Hierarchical allocation of IP addresses

- Allocation of IP addresses
  - one address per interface
  - each address composed of two parts
    1. subnetwork identifier
       - M high order bits of IP address
    2. equipment identifier inside the subnetworks
       - *32-M* bits low order bits of IP address

  Example

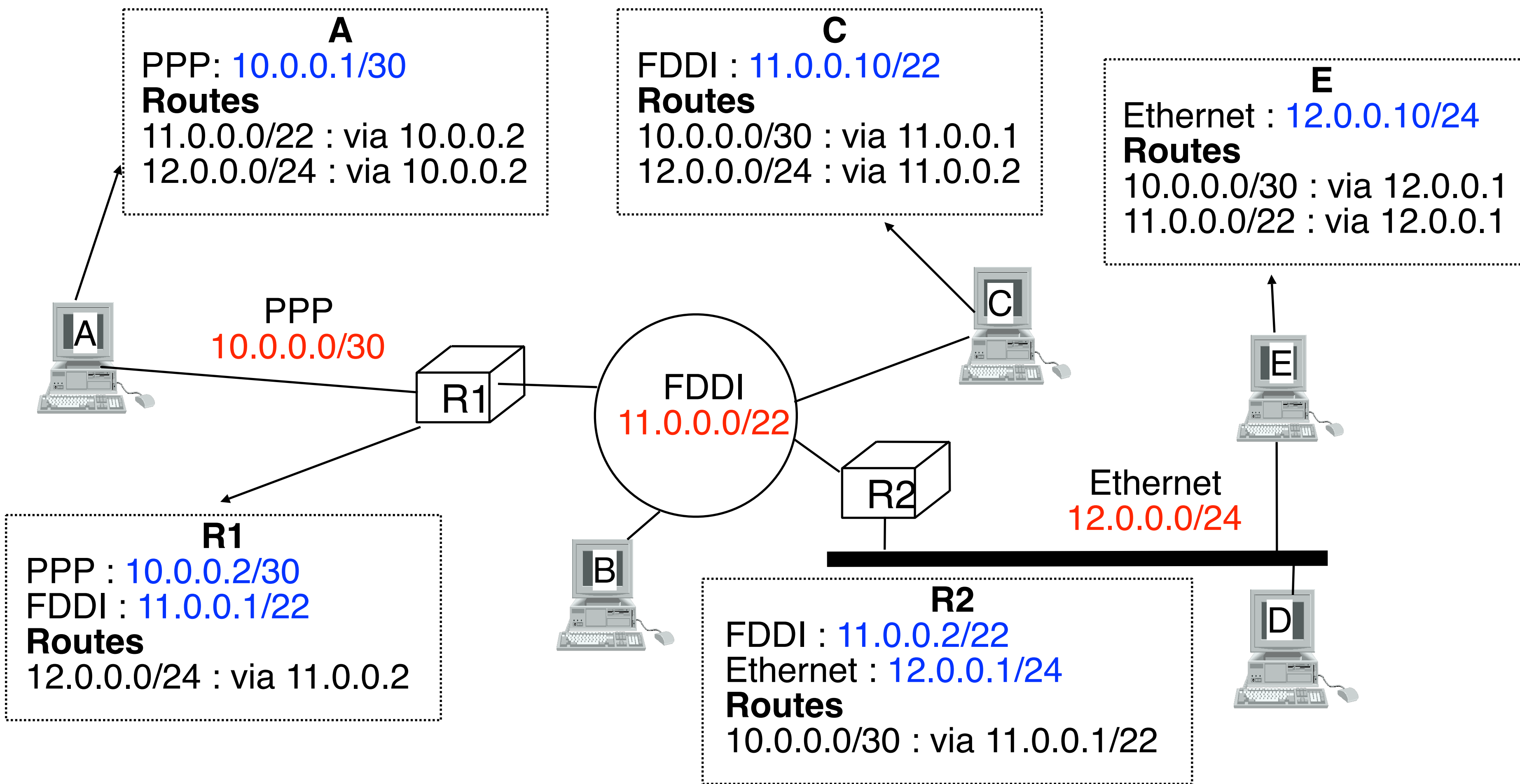
  10001010 00110000 0001101*0 00000001*

  subnetwork id — *host id*

  Notation 138.48.26.1/23 or 138.48.26.1 255.255.254.0

  - All hosts that belong to the same subnetwork can directly exchange frames through datalink layer

IP addressing : examples



- Drawbacks of subnetworks
  - most subnetworks are not fully occupied
  - a campus network will need more IP addresses than the number of hosts attached to the network

IP addresses

IP addresses

- Most addresses are allocated by IANA
  - and the regional registries RIPE, ARIN, ...

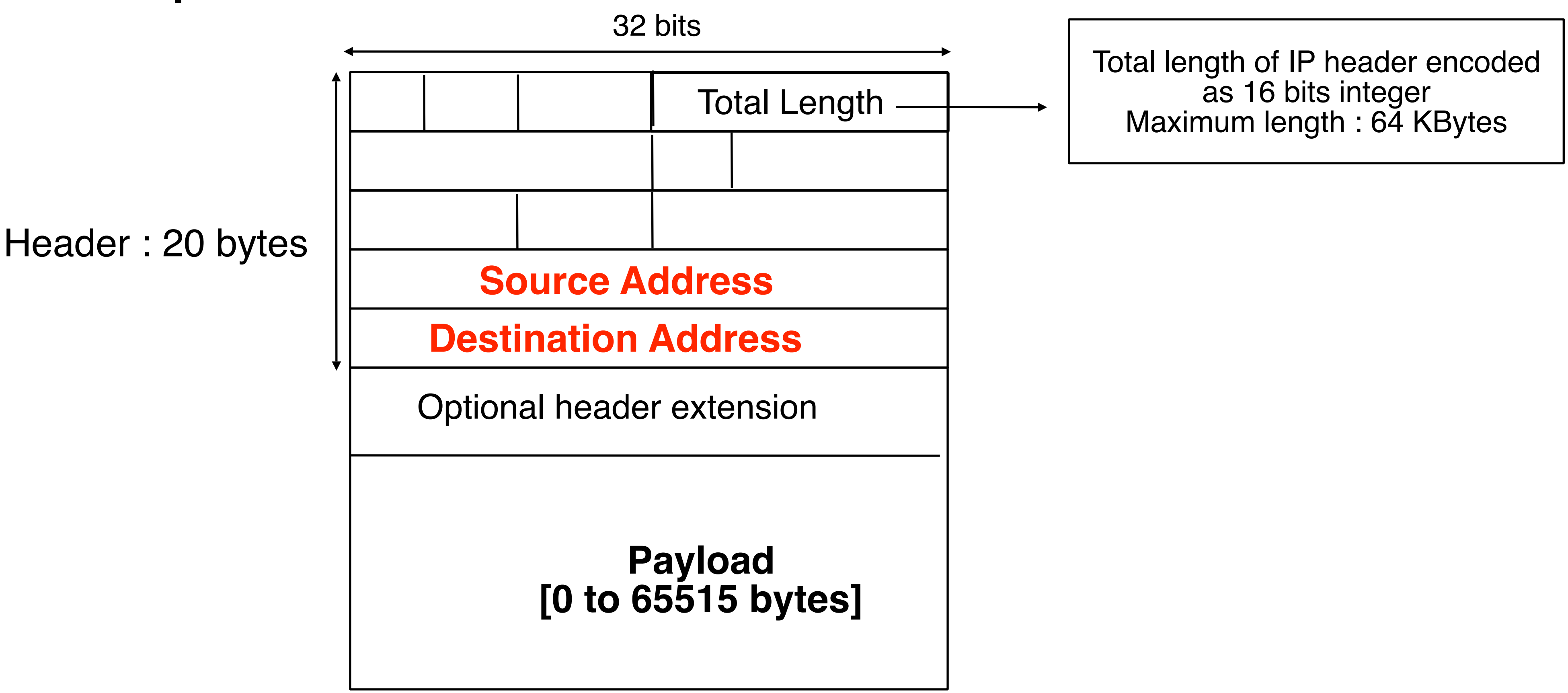
IP addresses

- Most addresses are allocated by IANA
  - and the regional registries RIPE, ARIN, ...

- But some addresses play a special role
  - 127.0.0.1
    - Loopback address on each host
      - Allows to reach servers on the local host
  - 10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16
    - used for private networks (not directly attached to Internet)
  - 218.0.0.0/8 - 223.0.0.0/8 and 240.0.0.0/8 - 255.0.0.0/8
    - reserved for further utilization
  - 224.0.0.0/8 - 239.0.0.0/8
    - used by IP multicast
  - 255.255.255.255
    - broadcast address
  - 0.0.0.0
    - used when a host is booting and does not yet know its address

IP Packets

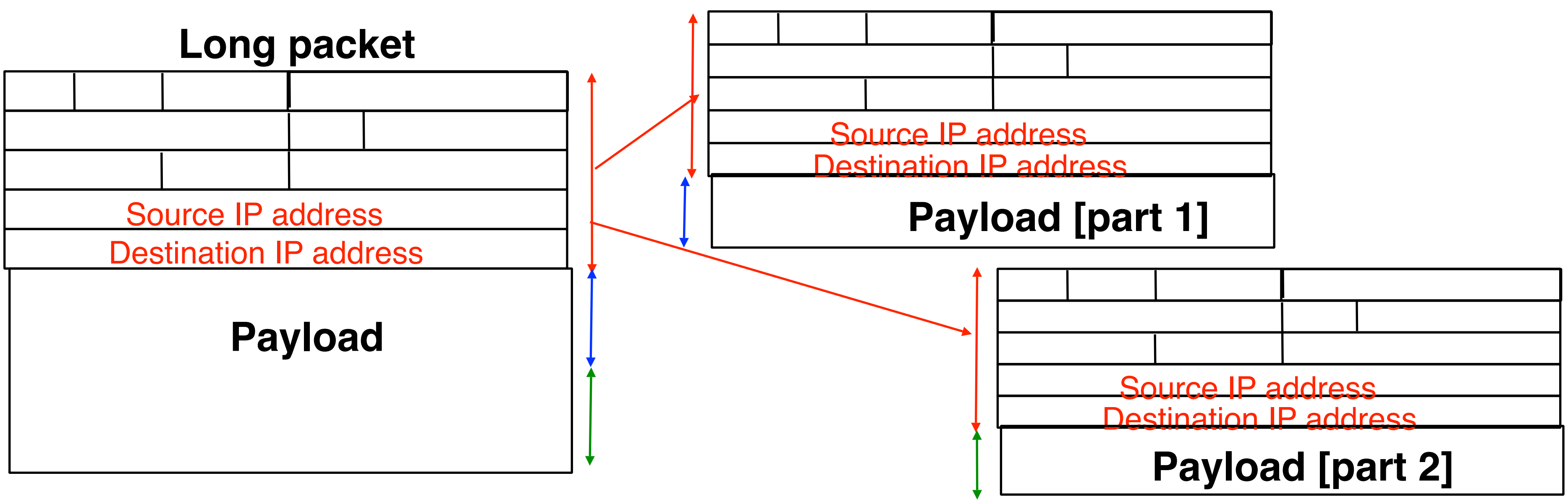
- IP packet format



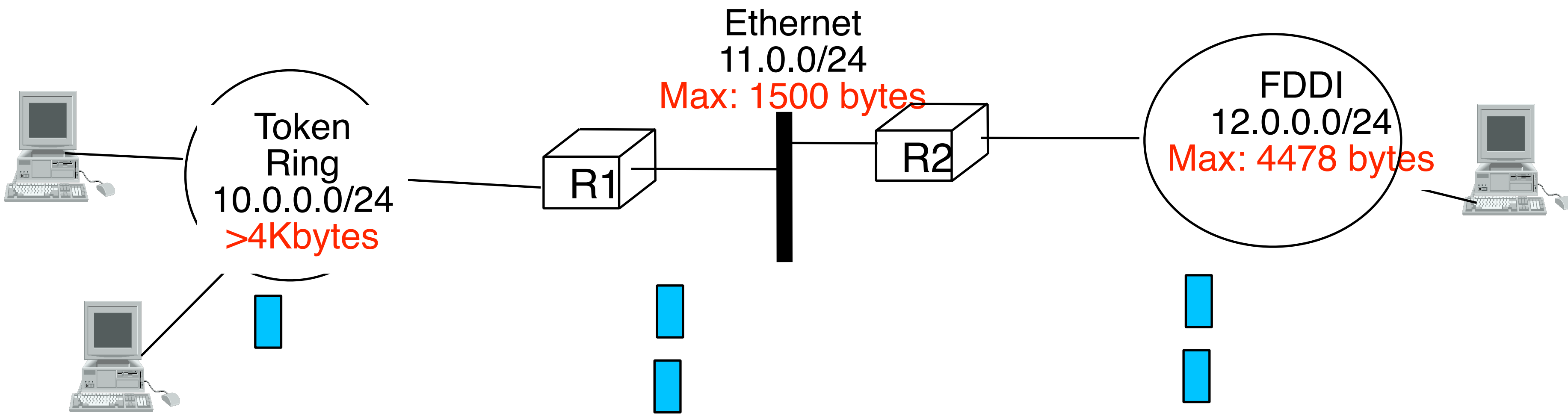
- How can we transmit a 64 KBytes packet ?

Transmission of long IP packets

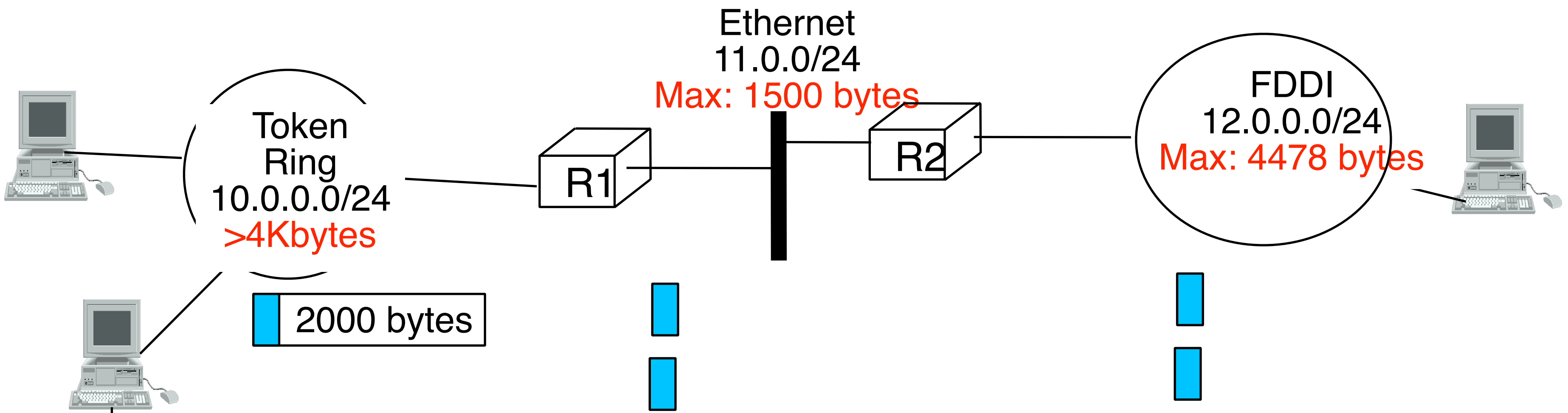
- Principe
  - Each host and each router can fragment packets
    - Each fragment is a complete IP packet that contains source and destination IP addresses
  - Only the destination host performs reassembly



Transmission of long IP packets (2)

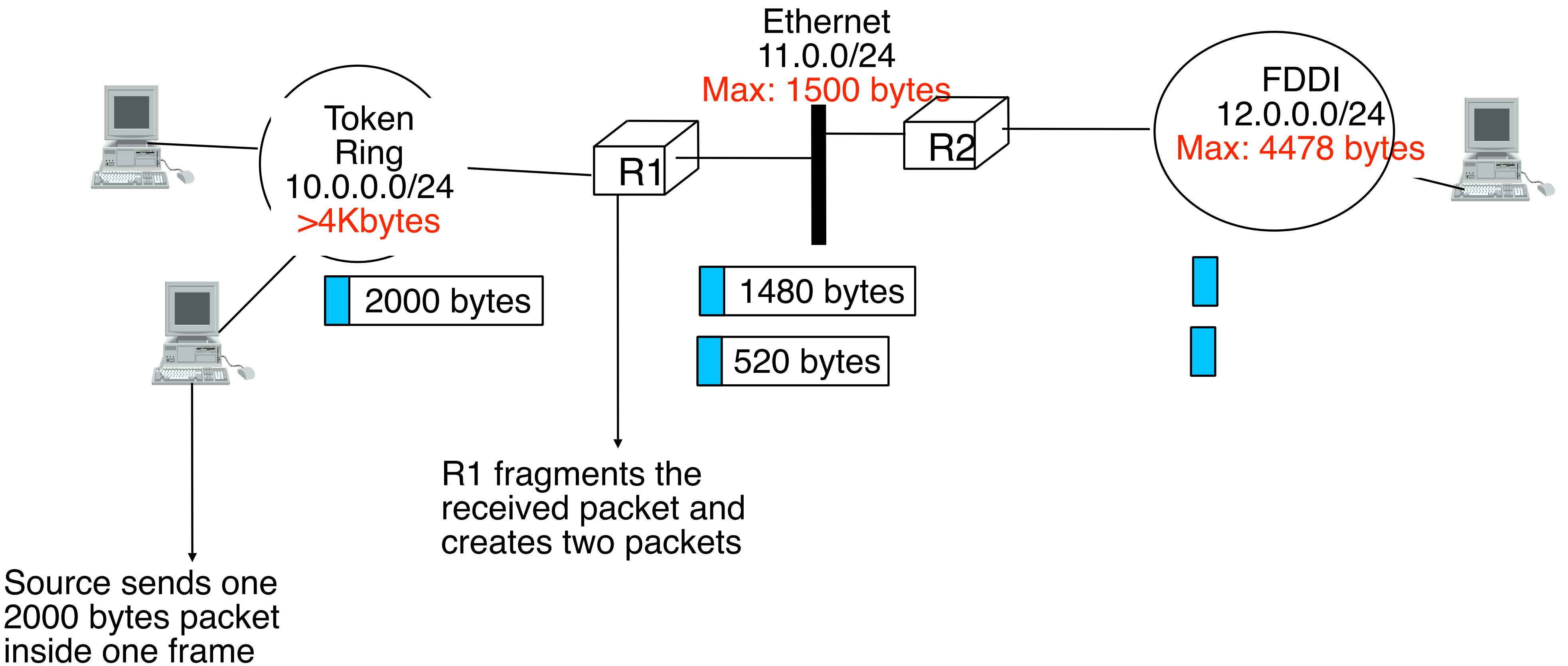


Transmission of long IP packets (2)

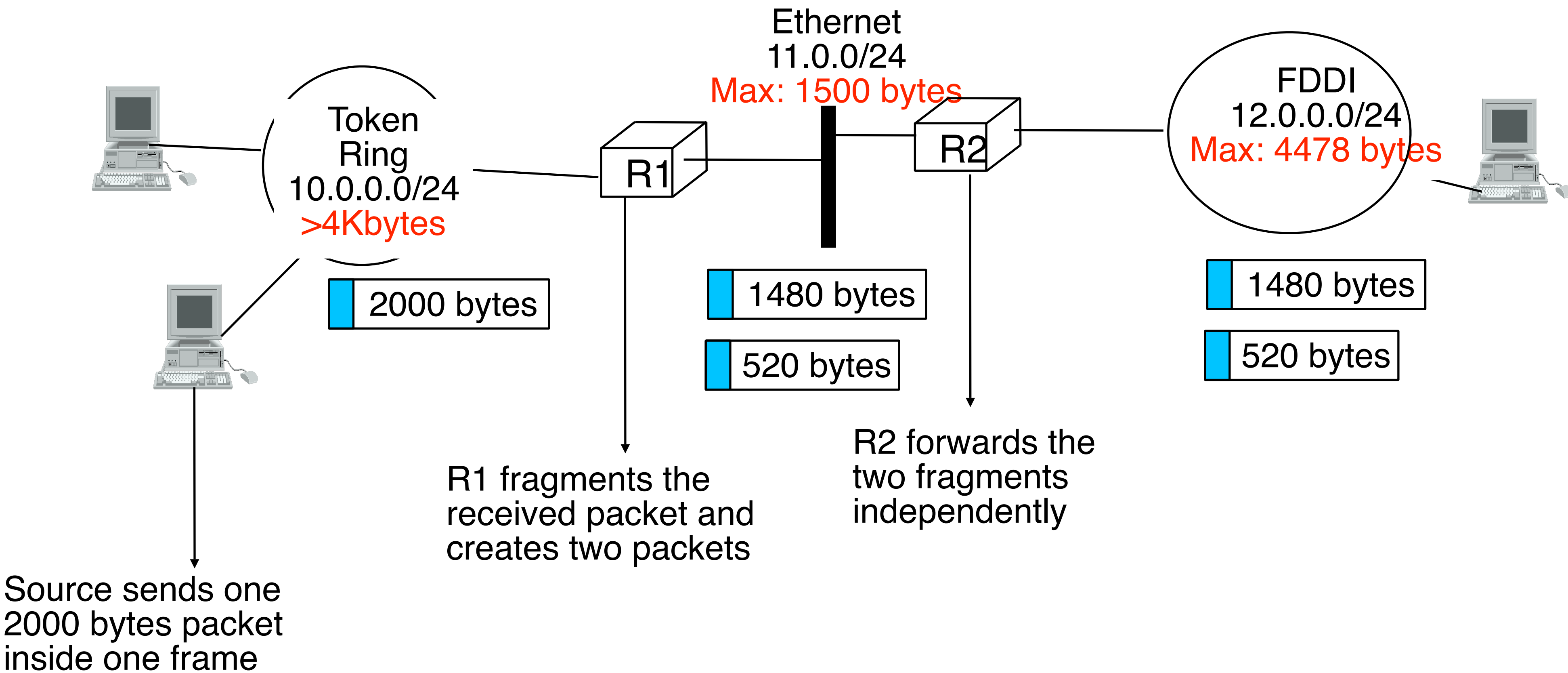


Source sends one 2000 bytes packet inside one frame

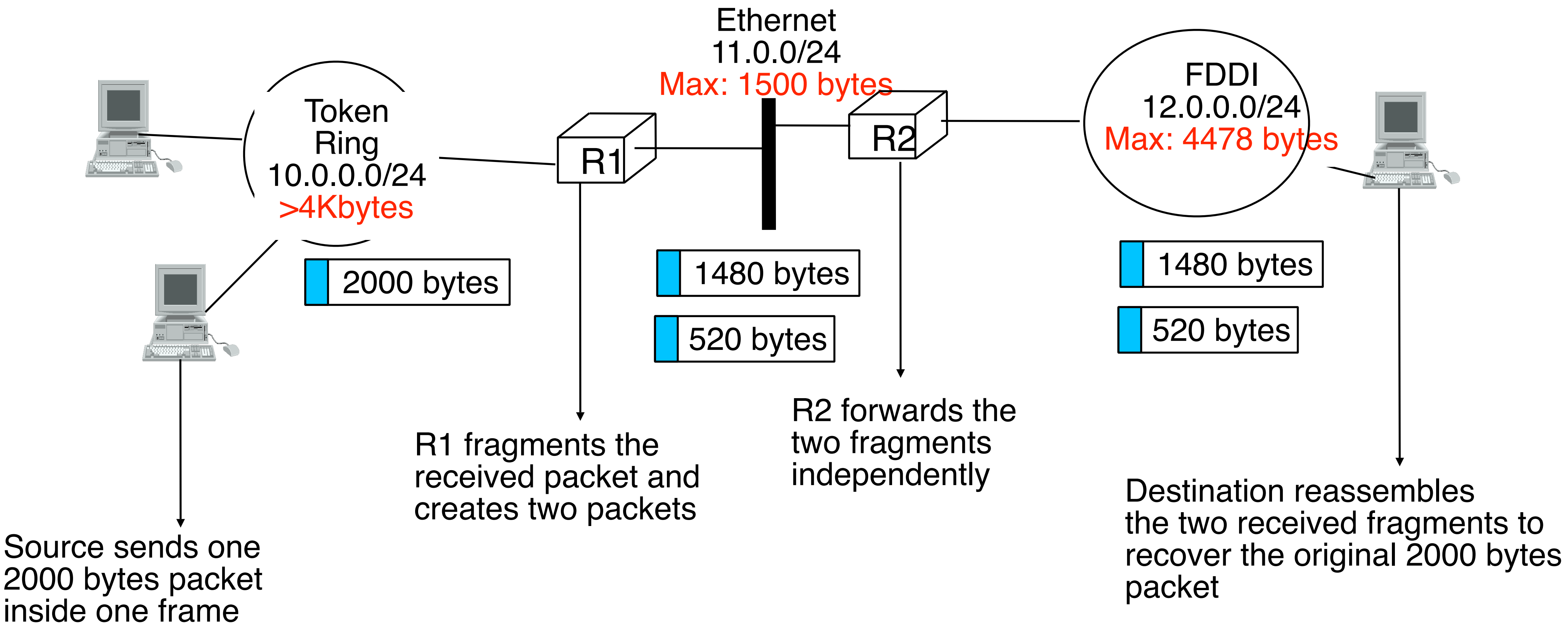
Transmission of long IP packets (2)



Transmission of long IP packets (2)

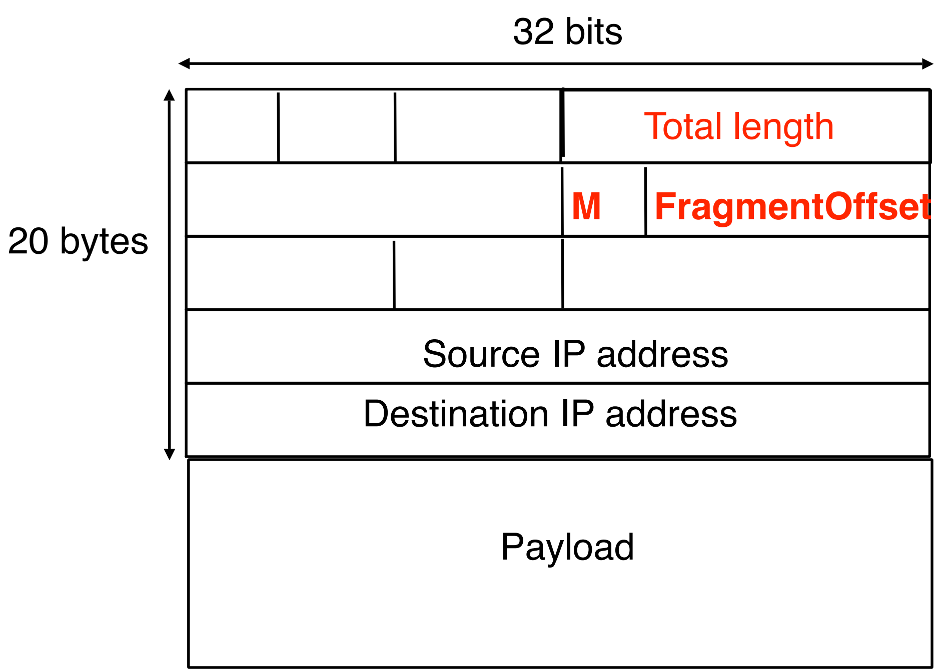


Transmission of long IP packets (2)



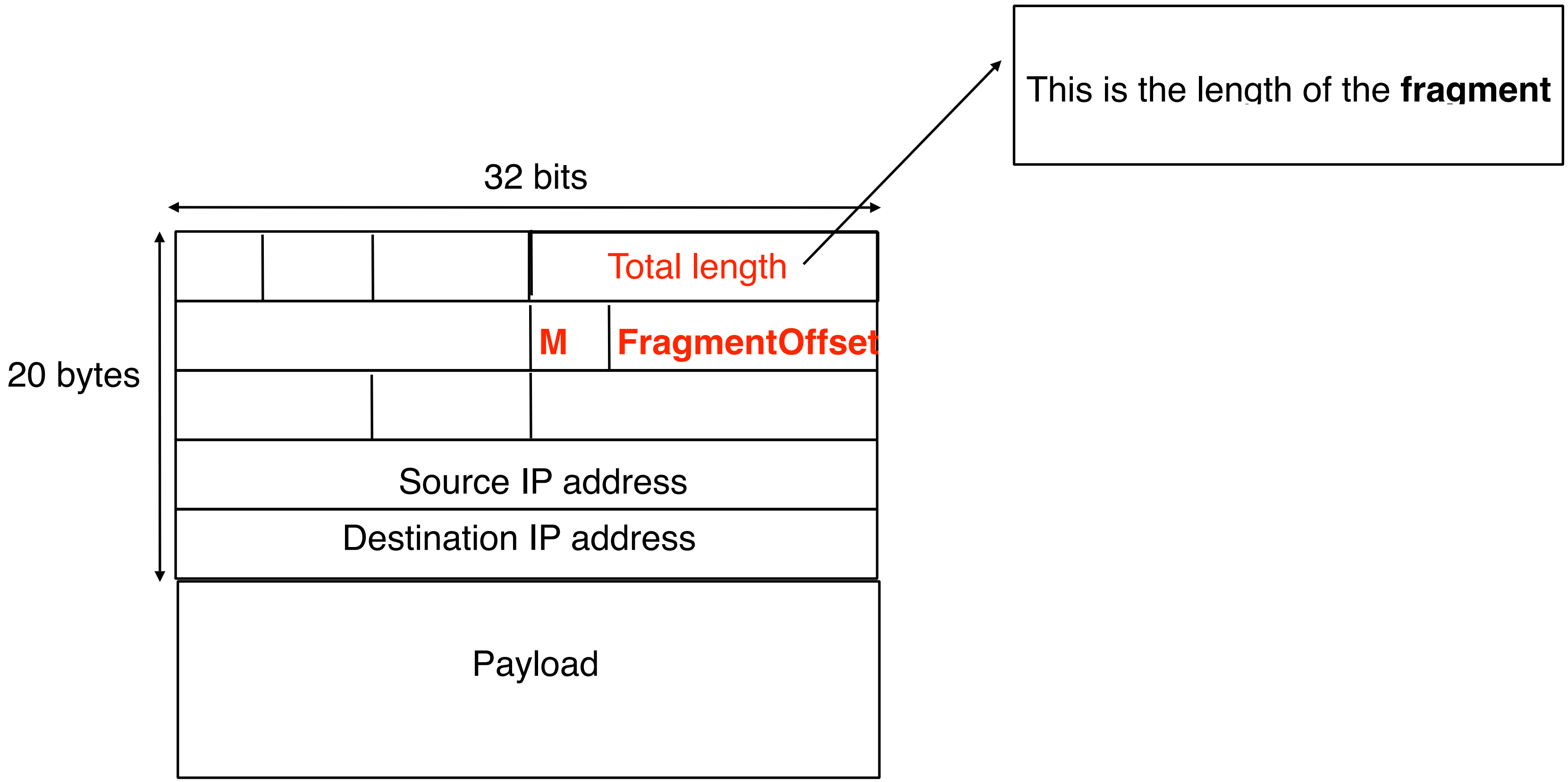
How to deal with limited MTU links ?

- IP fragmentation
  - Fragment the payload of IP packet
  - Each fragment must be number to recover from misordering



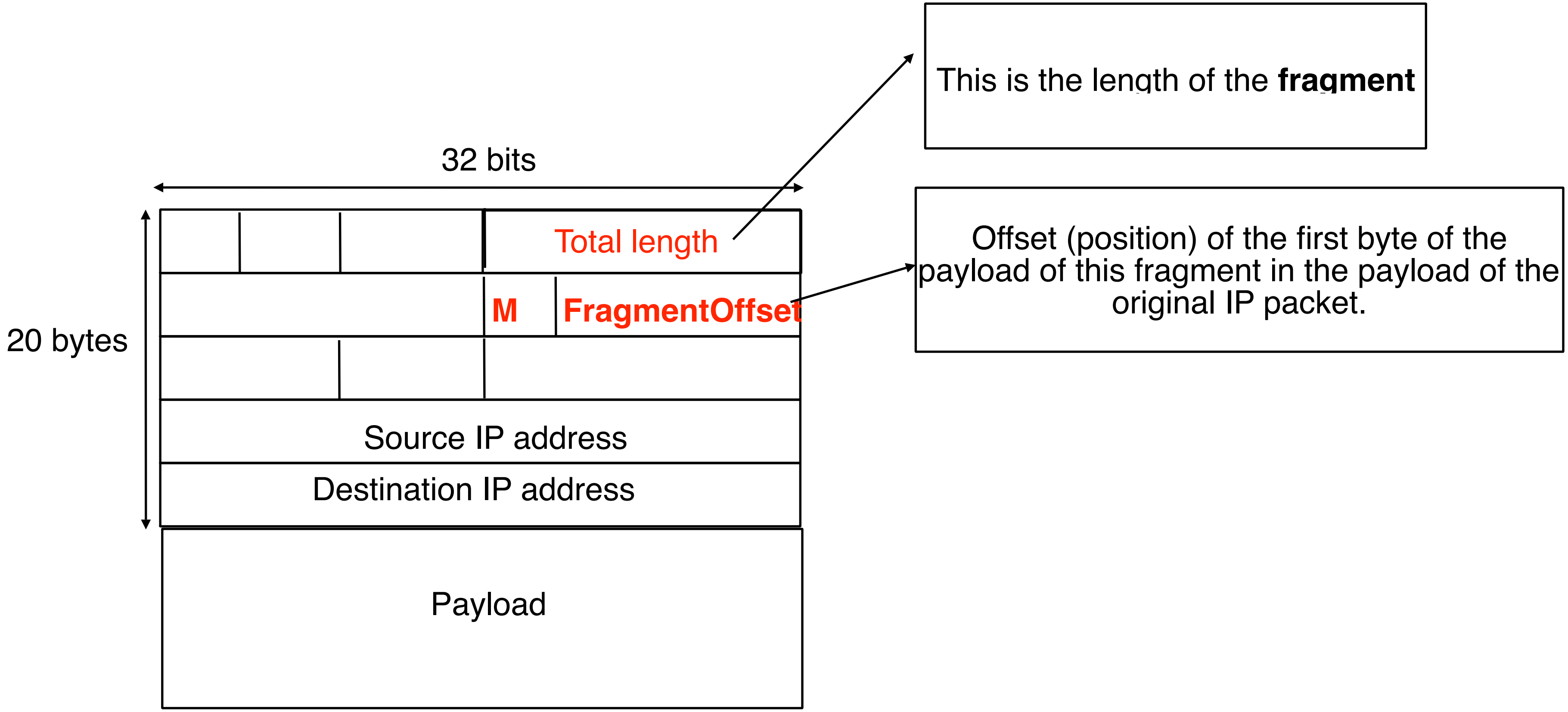
How to deal with limited MTU links ?

- IP fragmentation
  - Fragment the payload of IP packet
  - Each fragment must be number to recover from misordering



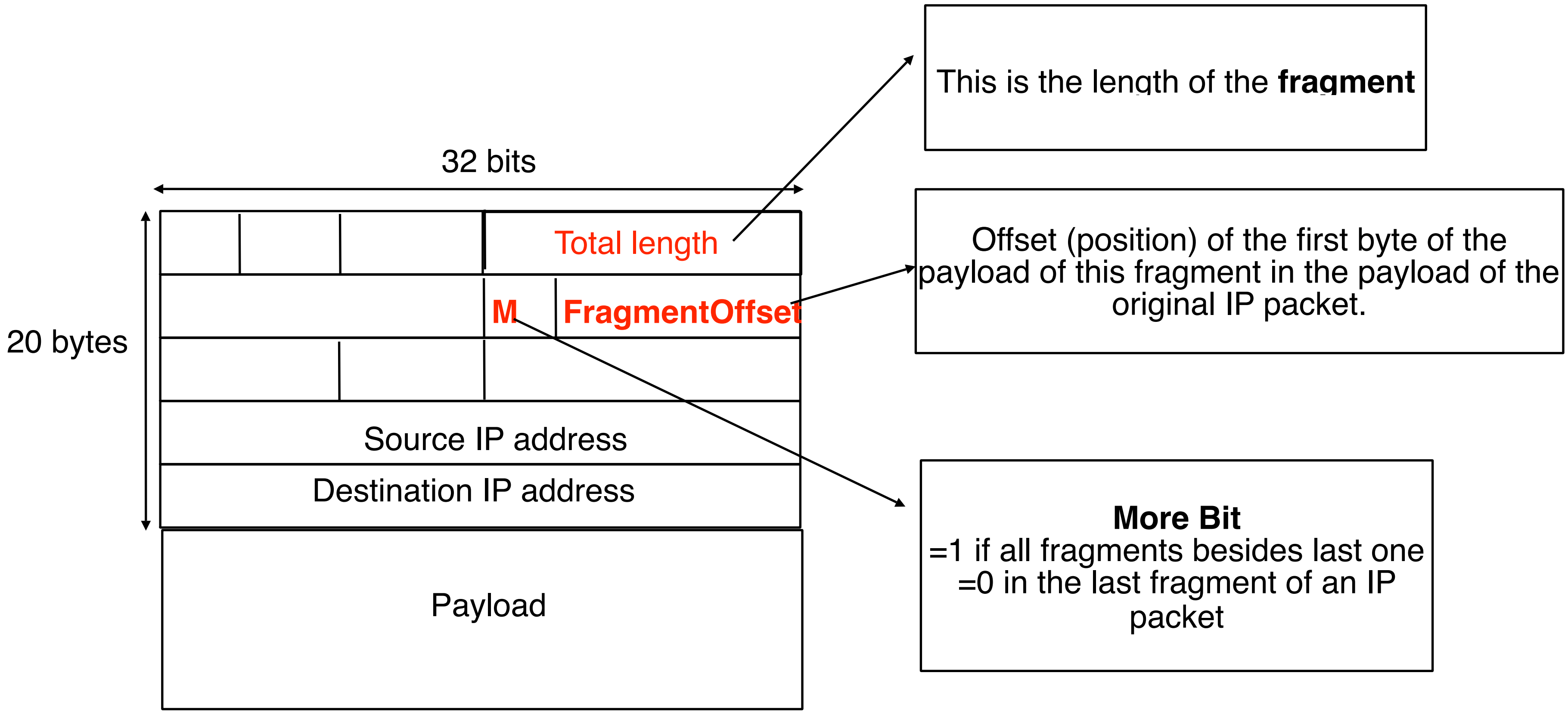
How to deal with limited MTU links ?

- IP fragmentation
  - Fragment the payload of IP packet
  - Each fragment must be number to recover from misordering

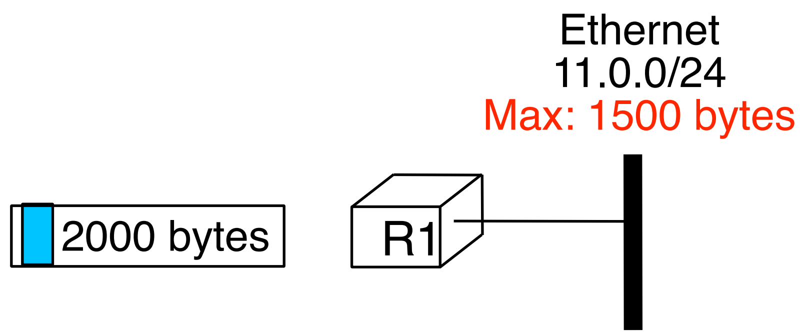


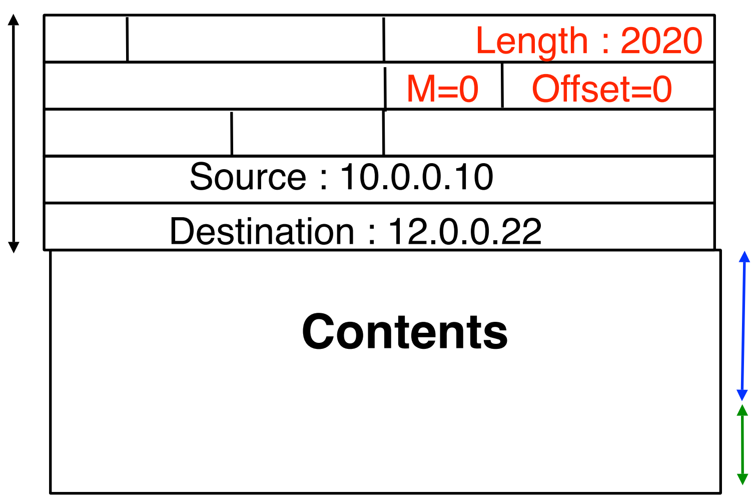
How to deal with limited MTU links ?

- IP fragmentation
  - Fragment the payload of IP packet
  - Each fragment must be number to recover from misordering

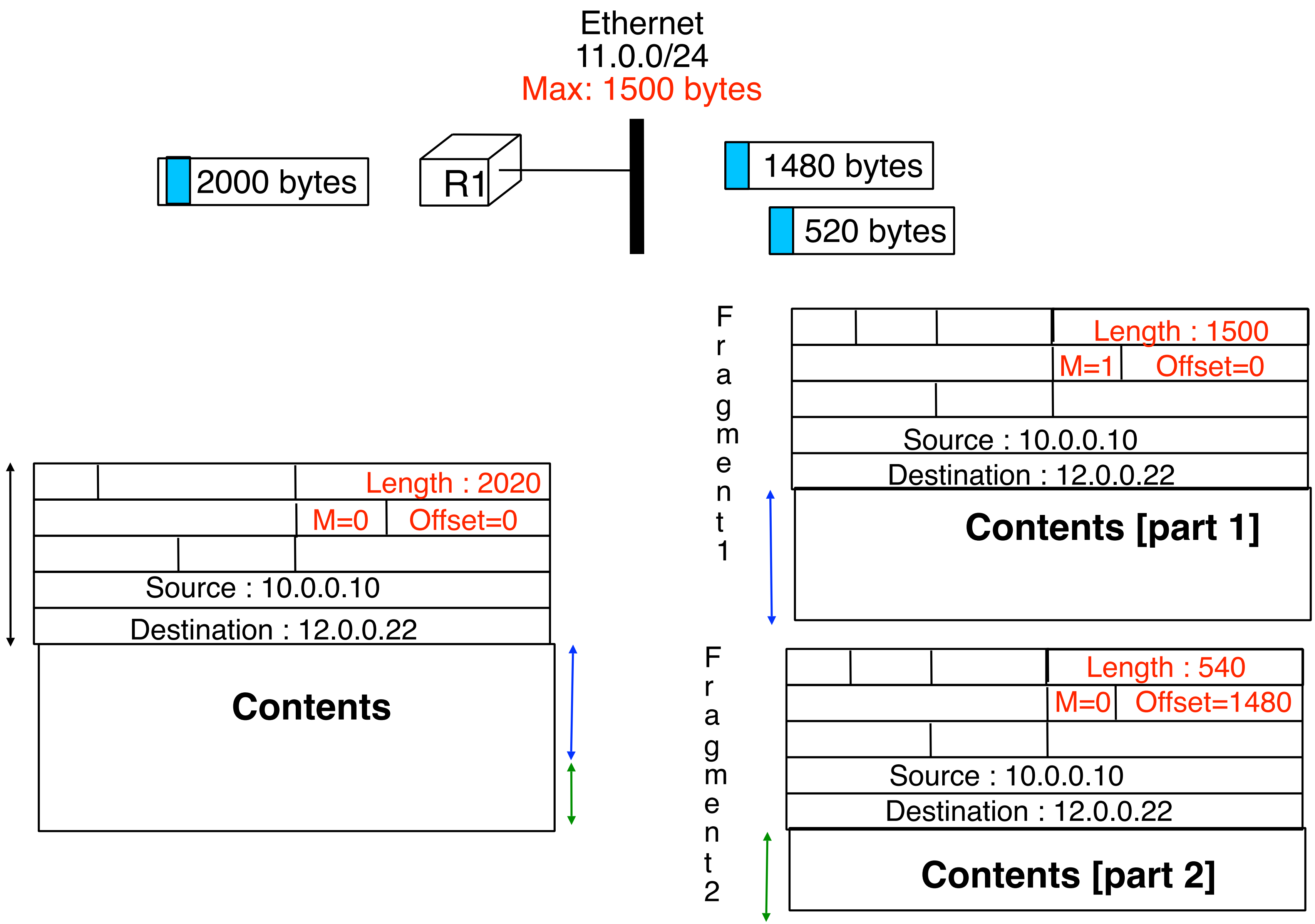


Fragmentation : example





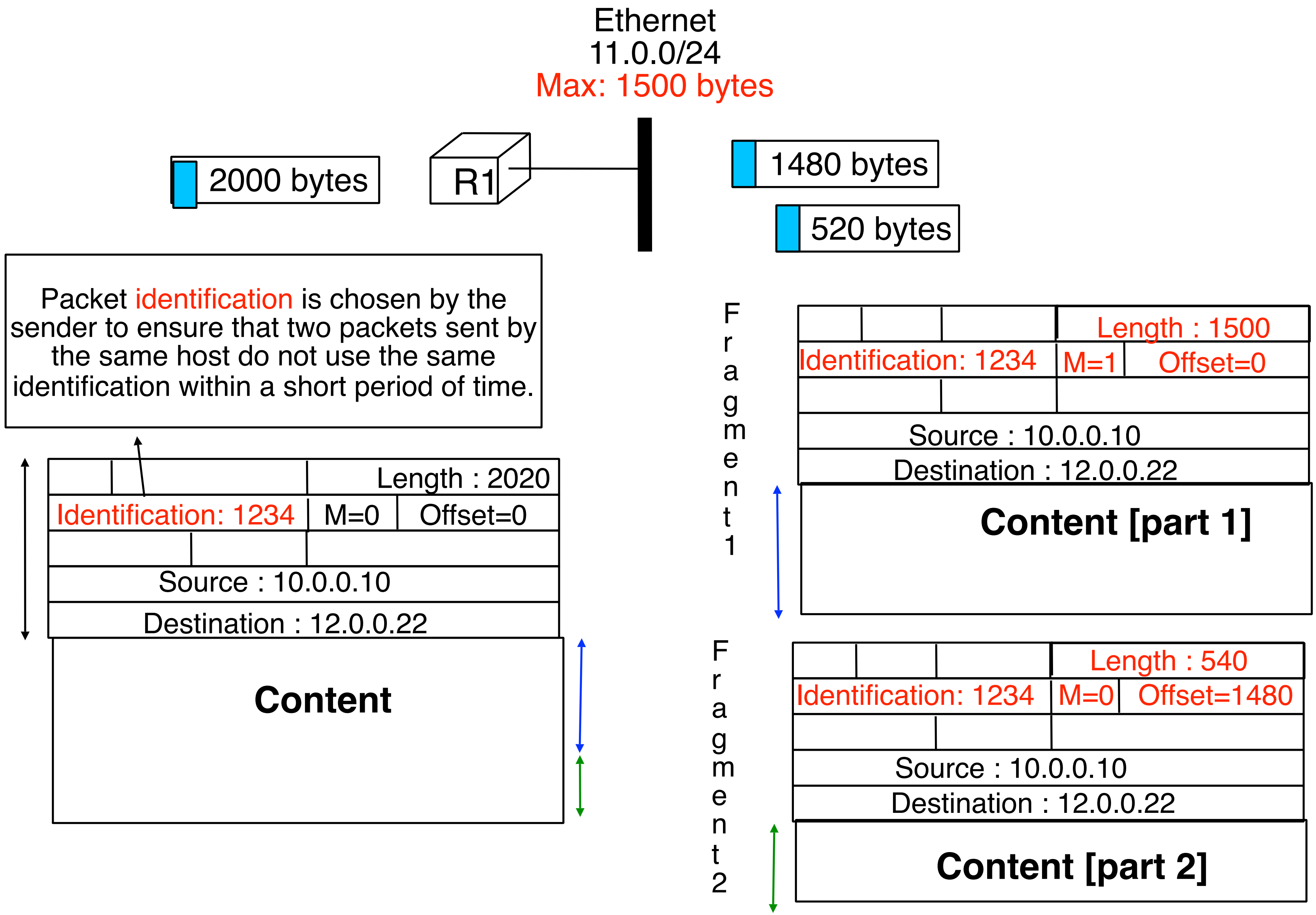
Fragmentation : example



Reassembly

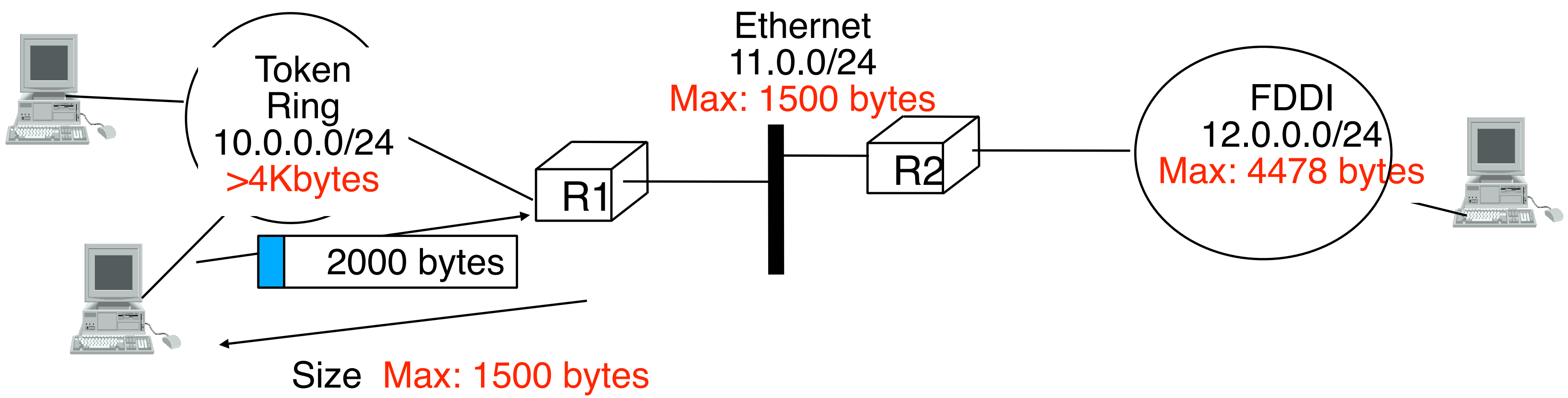
- Issues
  - When does the destination has received all fragments ?
    - Last fragment contains bit More=0
    - How to handle lost fragments ?
      - the IP packet will not be reassembled by destination and received fragments of this packet will be discarded
  - How to deal with misordering
    - Offset field allows to reorder fragments from same packet
    - But misordering can cause fragments from multiple packets to be mixed
      - Each fragment must contain an identification of the original packet from which is was created

Packets and fragments identification



How to avoid fragmentation ?

- Problem
  - How can a host determine the maximum packet size that he can use to reach a destination ?
  Solution
    - Instead of performing fragmentation, the router could indicate the maximum packet size that it supports



    - Knowing this maximum packet size, the endhost can send correctly sized packets

Transmission errors

- How should IP react to transmission errors ?
  - Transmission error inside packet content
    - some applications may continue to work despite this error
    - IP : no detection of transmission errors in packet payload
  - Transmission error inside packet header
    - could cause more problems
      - imagine that the transmission error changes the source or destination IP address
    - IP uses a checksum to detect transmission errors in header
      - 16 bits checksum (same as TCP/UDP) computed only on header
      - each router and each end host verifies the chacksum of all packets that it receives. A packet with an errored header is immediately discarded

Transient and permanent loops

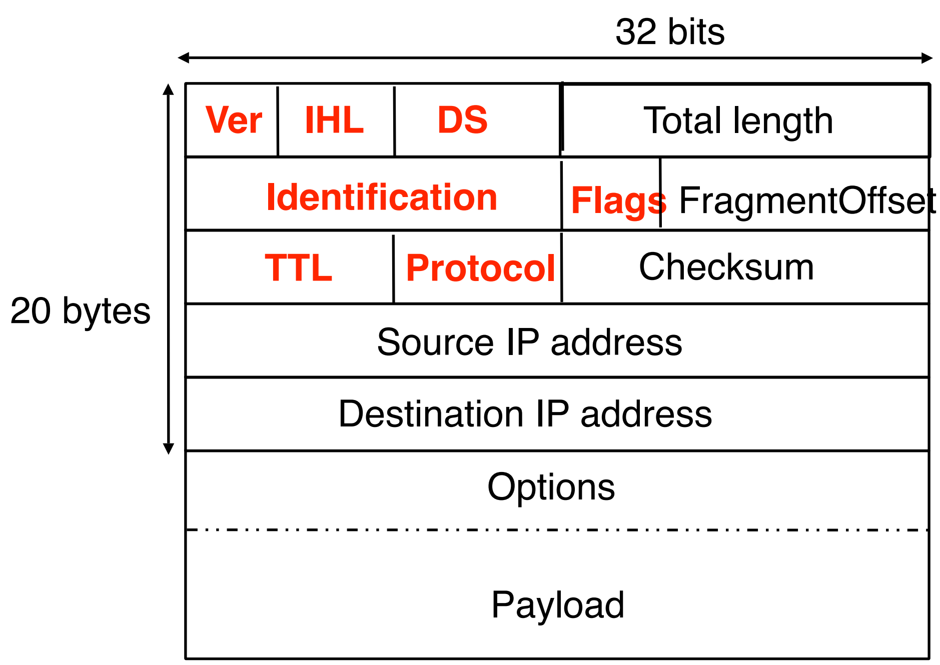
Transient and permanent loops

- Problem
  - Loops can occur in an IP network
    - permanent loops due to configuration errors
    - transient loops while routing tables are being updated

Transient and permanent loops

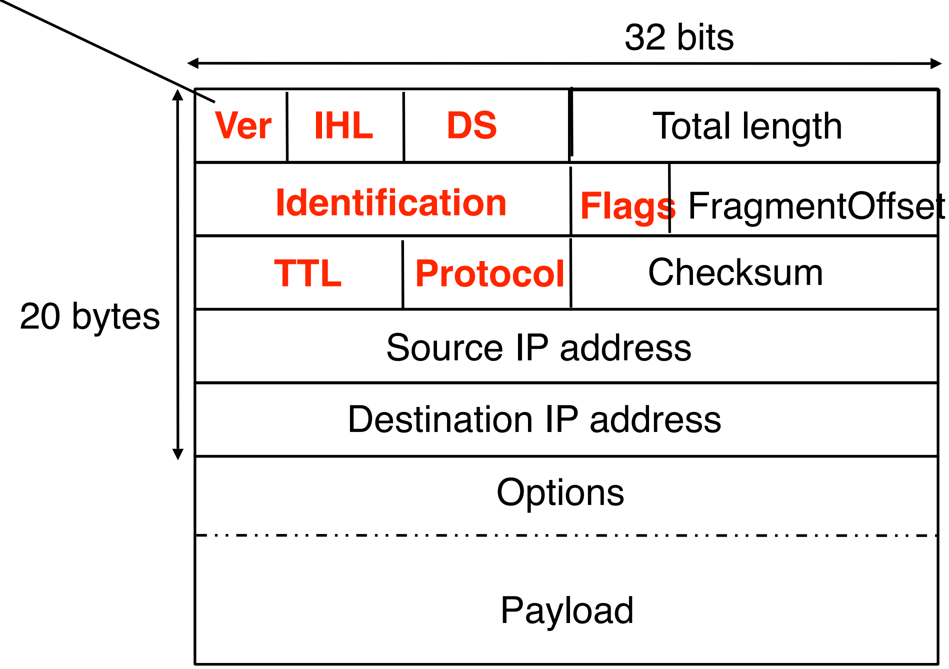
- Problem
  - Loops can occur in an IP network
    - permanent loops due to configuration errors
    - transient loops while routing tables are being updated
- Solution
  - Each packet contains a Time-to-Live (TTL) that indicates the maximum number of intermediate routers that the packet can cross
    - many hosts set the initial TTL of their packets to 32 or 64
  - each router checks the TTL of all packets
    - If TTL=1, packet is discarded and source is notified
    - If TTL>1, packet is forwarded and TTL is decremented by at least 1
      - routers thus must recompute checksum of all forwarded packets
  - Utilisation of TTL is a means to bound the lifetime of packets inside the Internet

IP header format

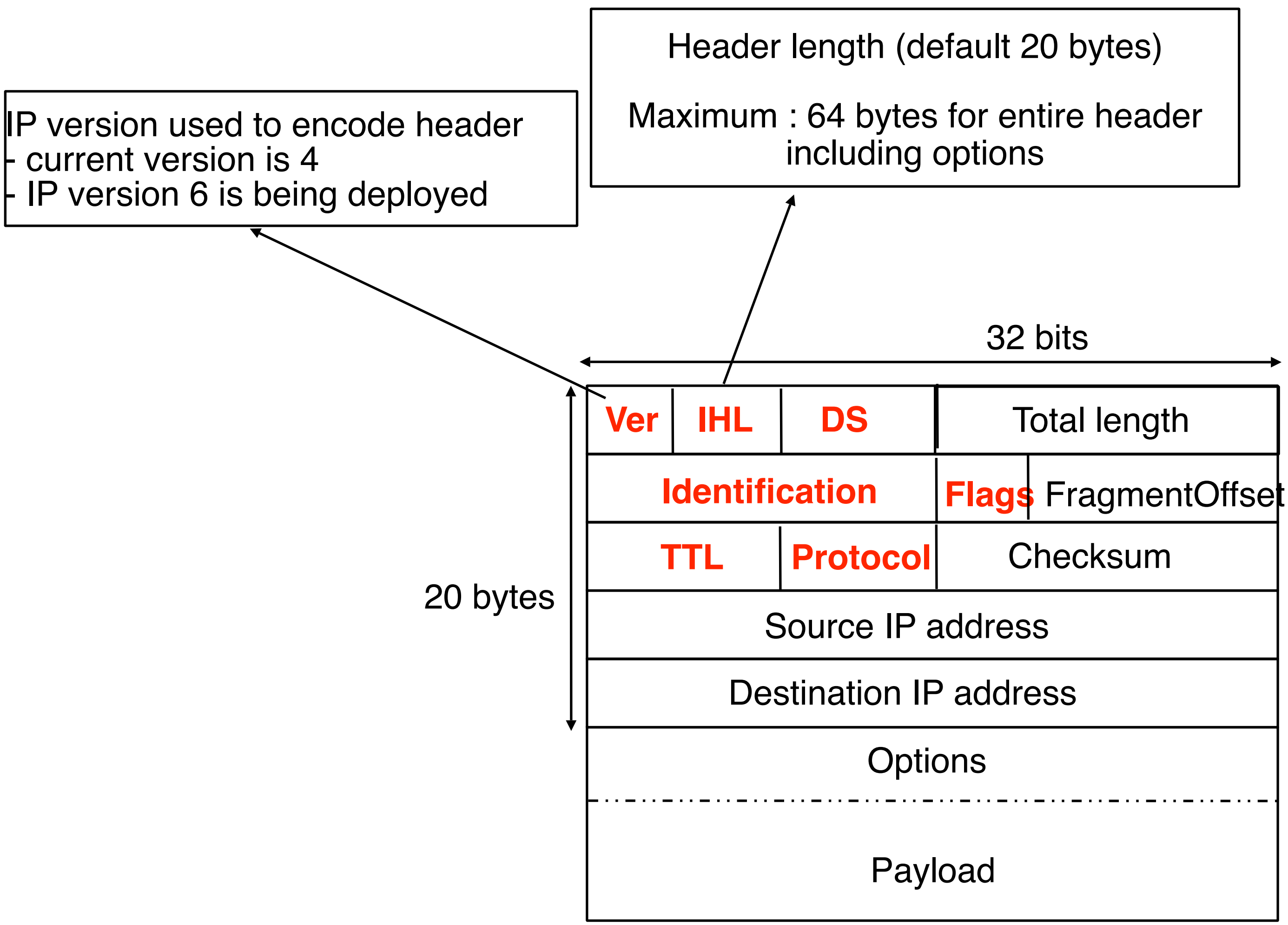


IP header format

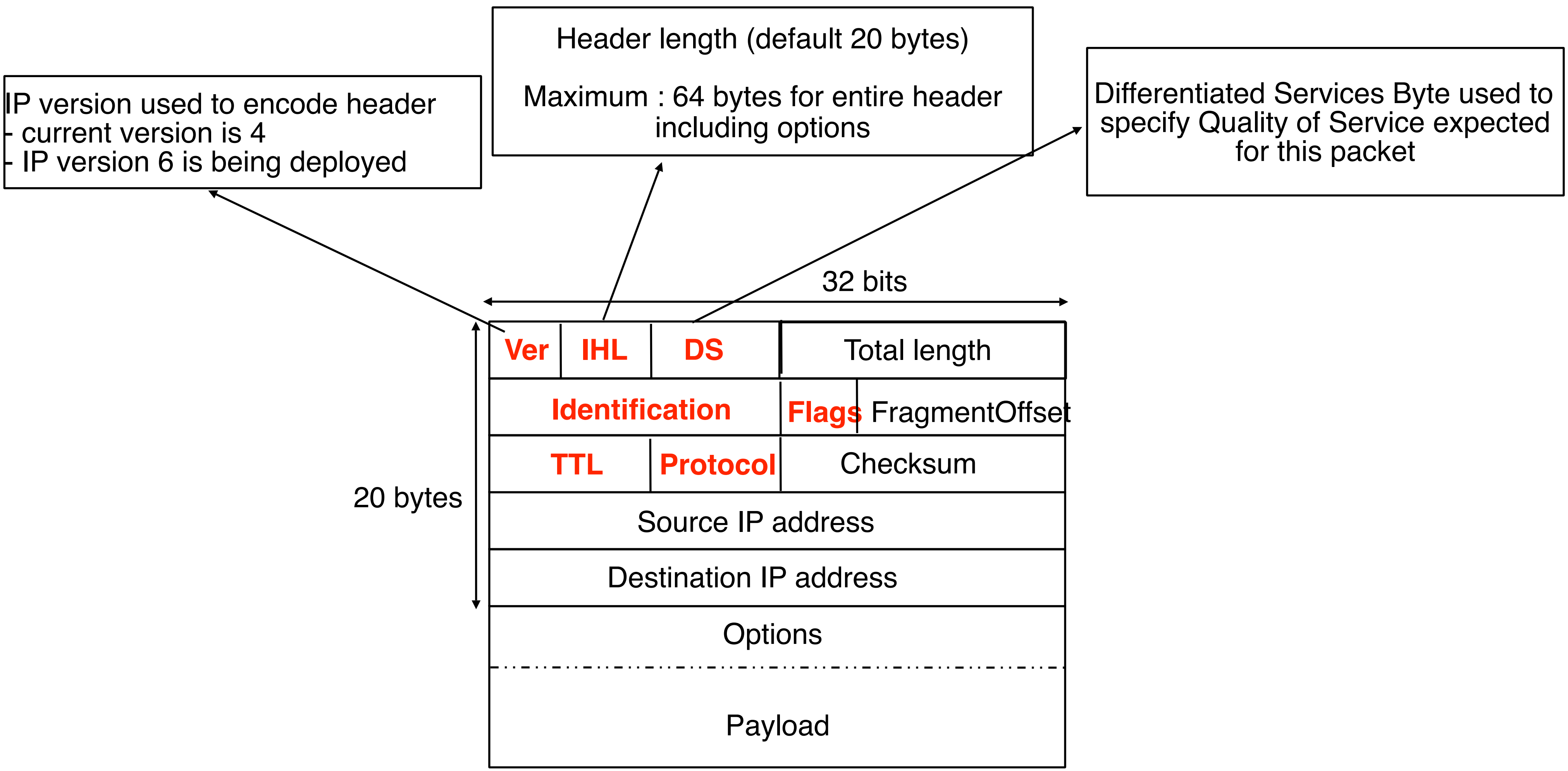
IP version used to encode header
- current version is 4
- IP version 6 is being deployed



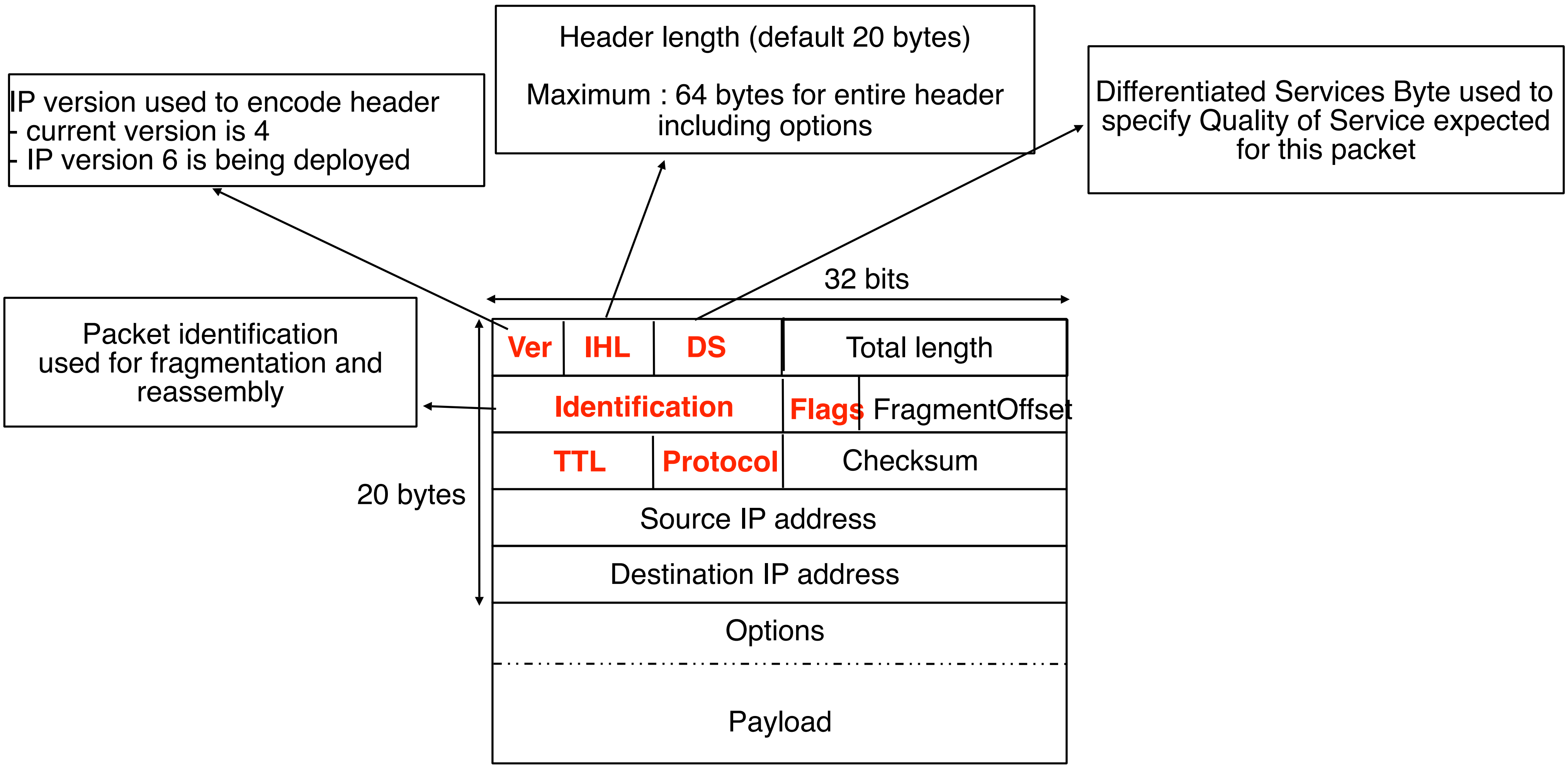
IP header format



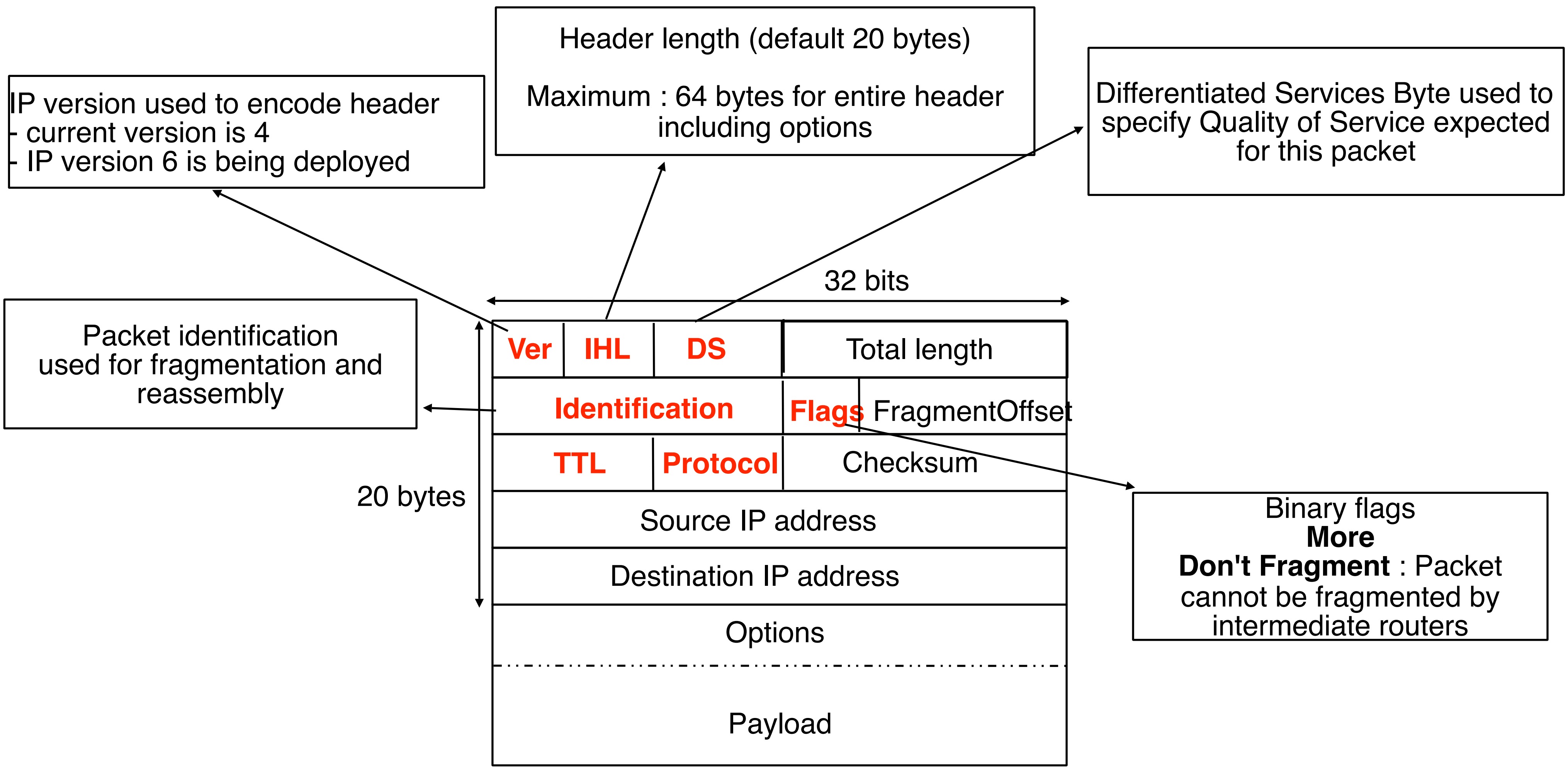
IP header format



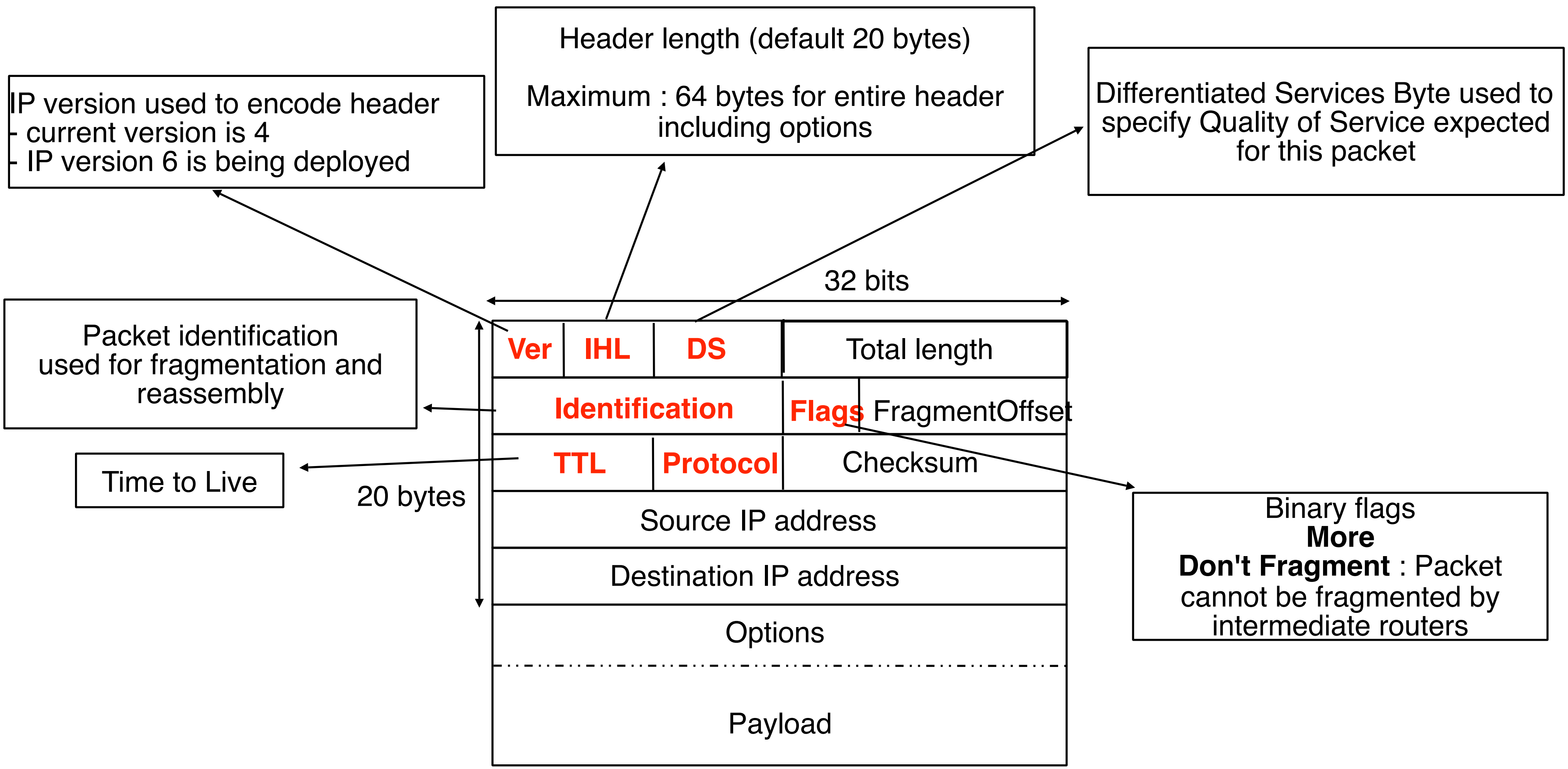
IP header format



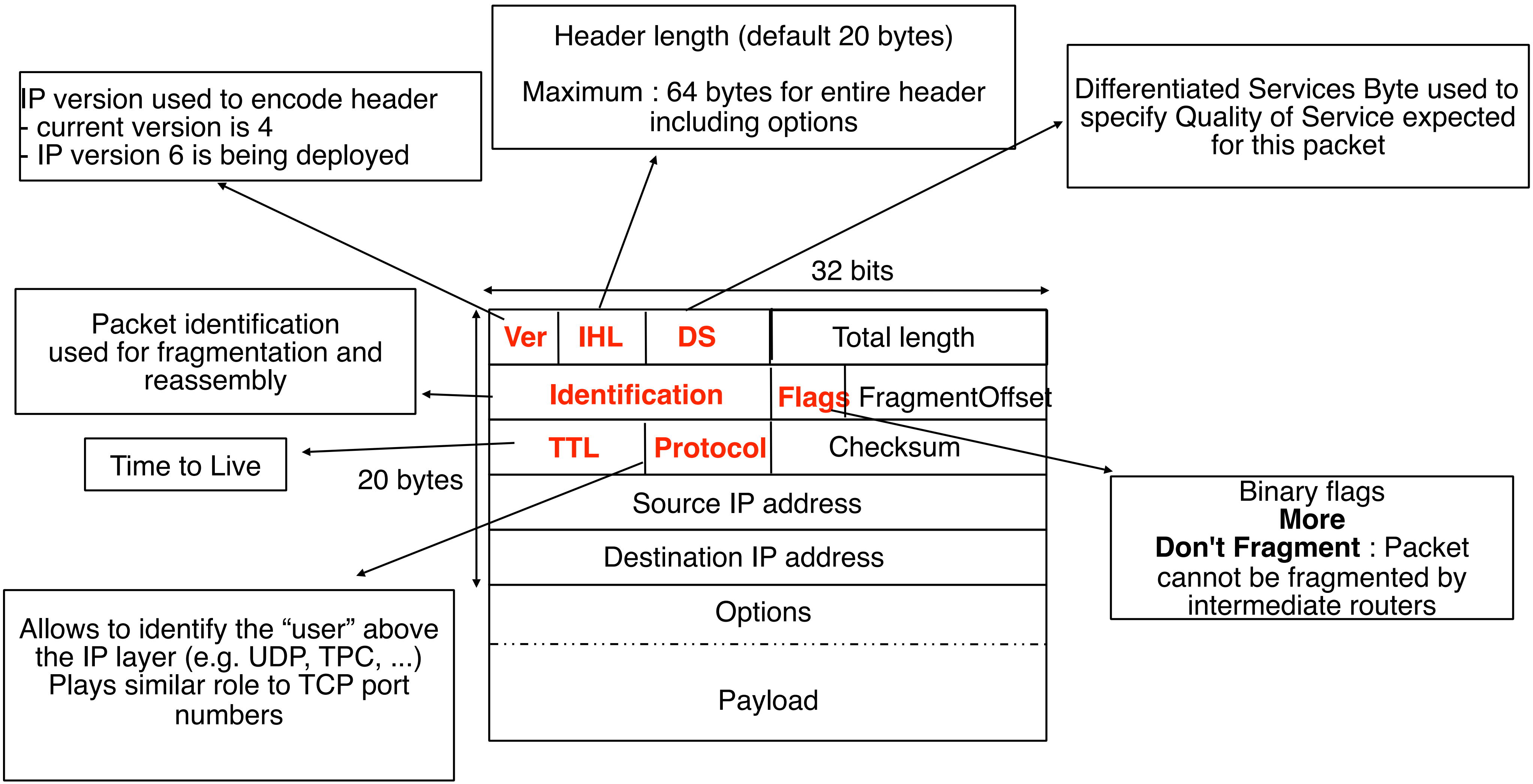
IP header format



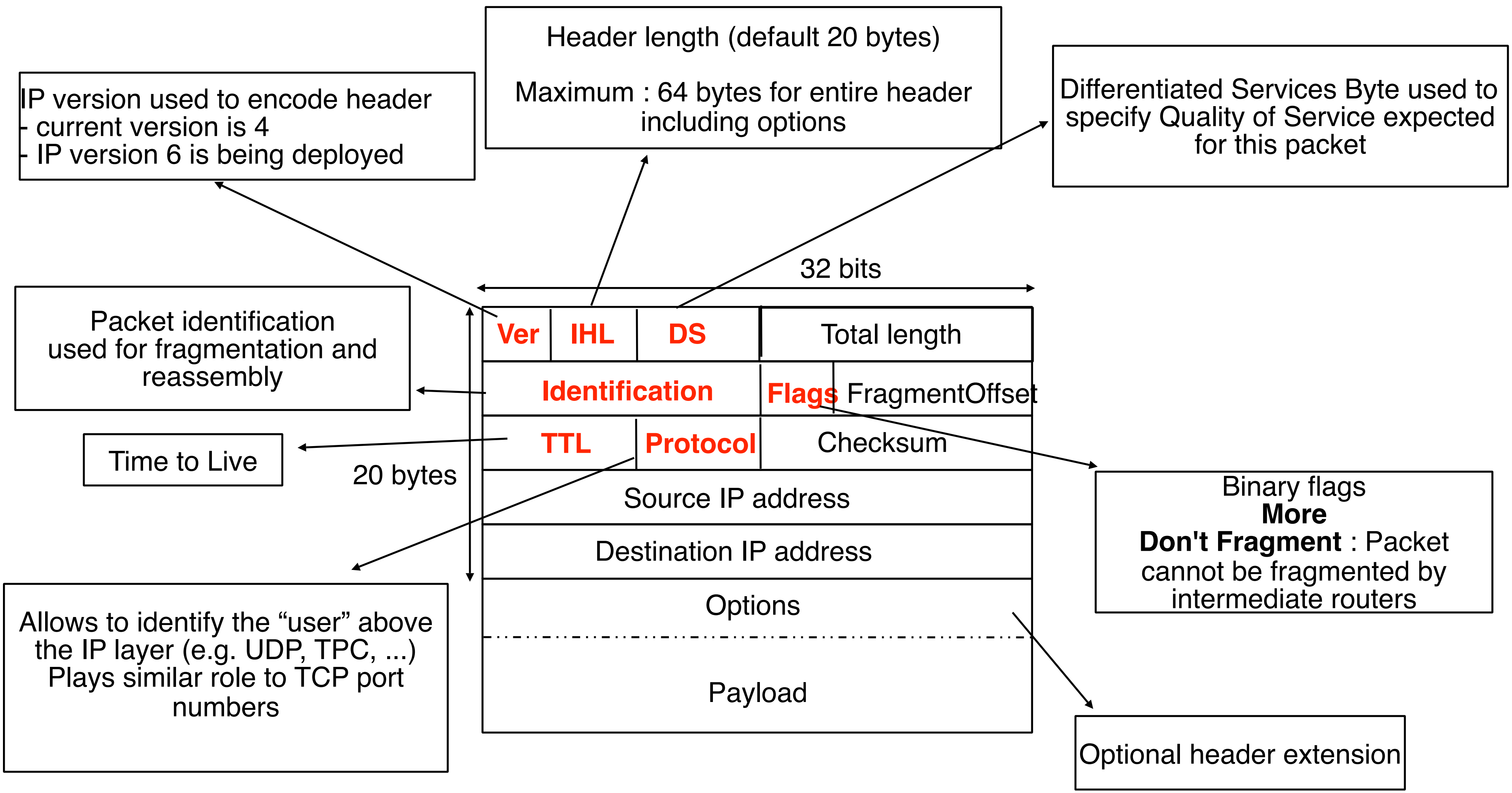
IP header format



IP header format



IP header format



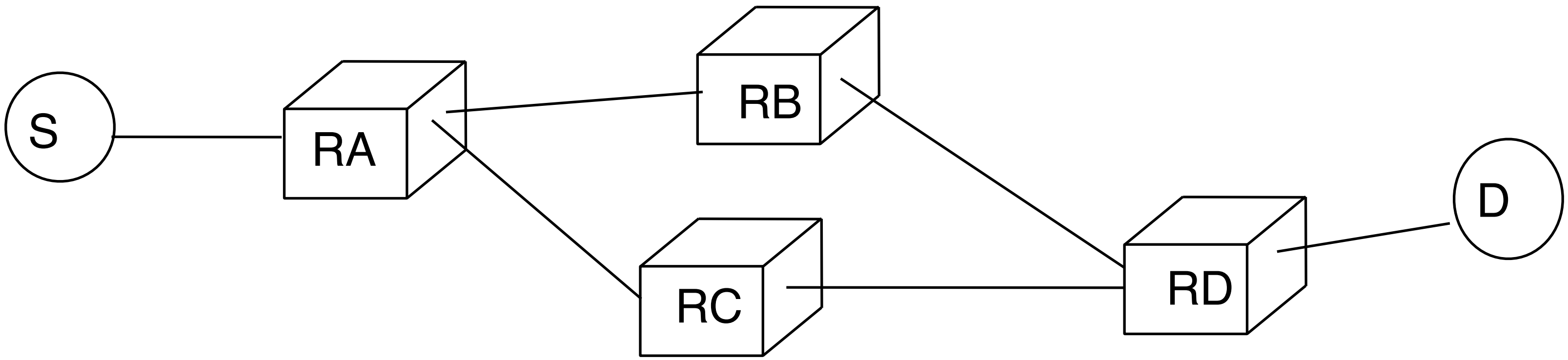
IP Options

- Sample IP header options
  - Strict source route option
    - allows the source to list IP addresses of all intermediate routers to reach destination between source and destination
  - Loose source route option
    - allows the source to list IP addresses of some intermediate routers to reach destination between source and destination
  - Record route option
    - allows each router to insert its IP address in the header
      - rarely used because limited header length
  - Router alert
    - allows the source to indicate to routers that there is something special to be done when processing this packet

Constraint : maximum header size with option 64 bytes

IP Source Routing

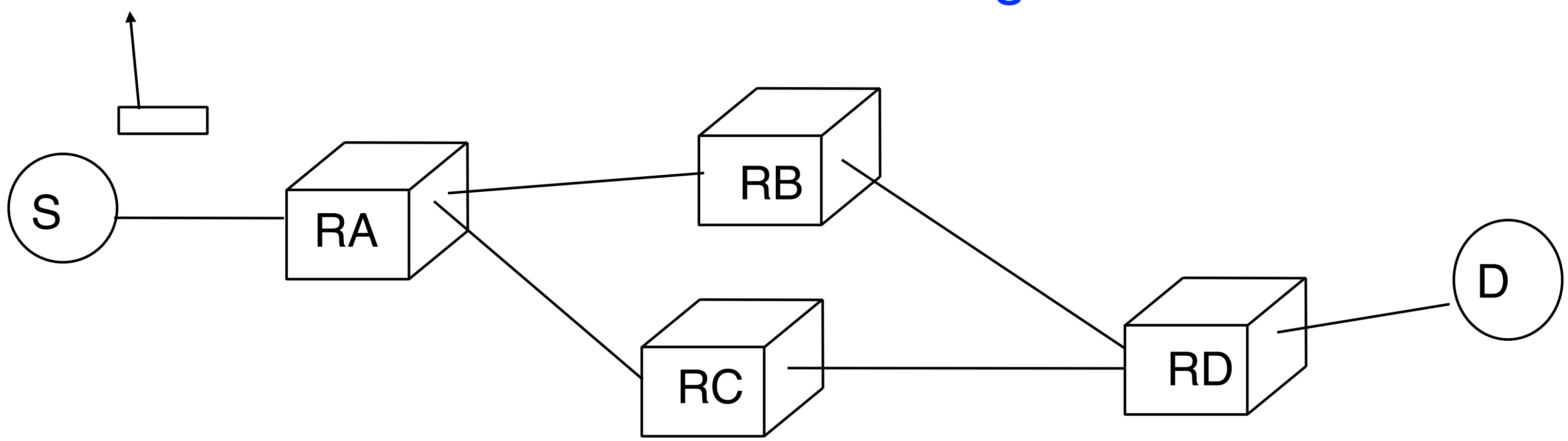
Strict
source routing



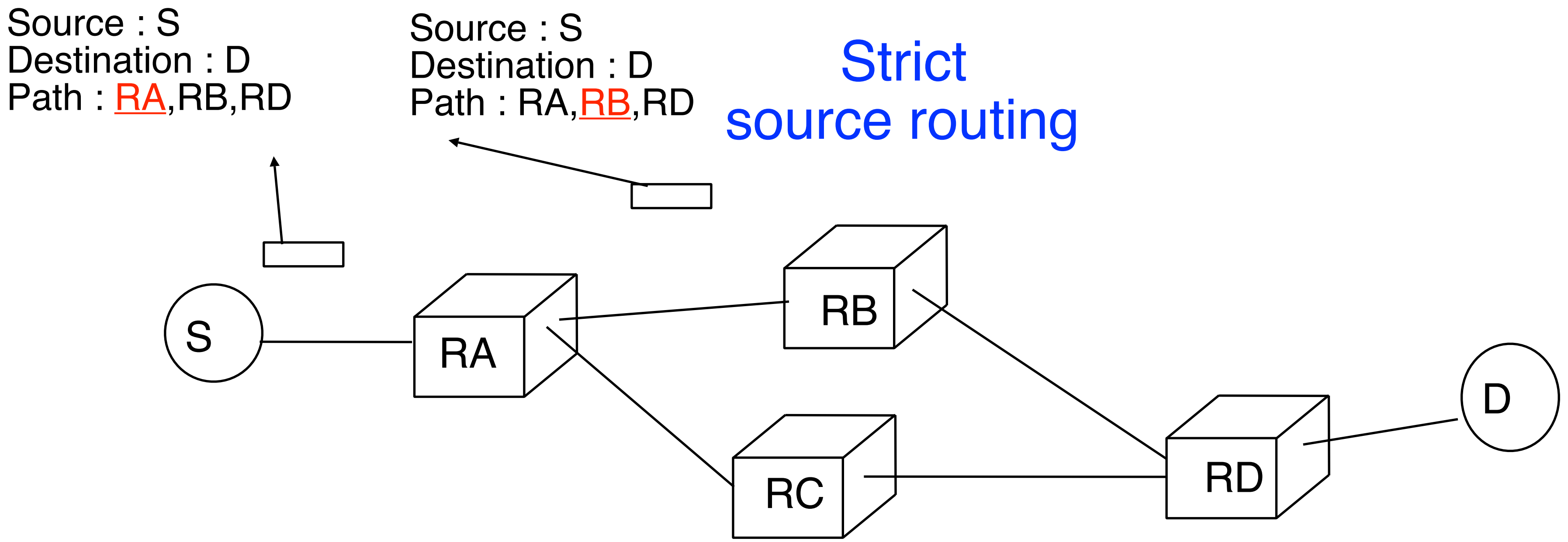
IP Source Routing

Source : S
Destination : D
Path : RA,RB,RD

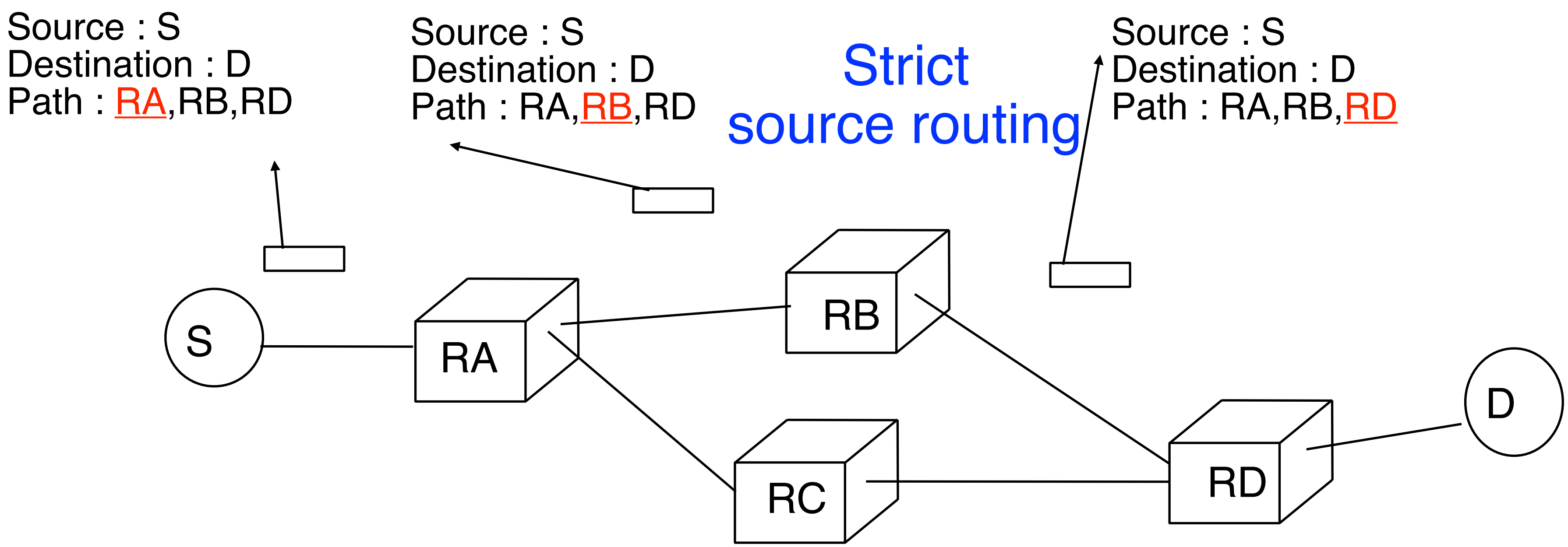
Strict
source routing



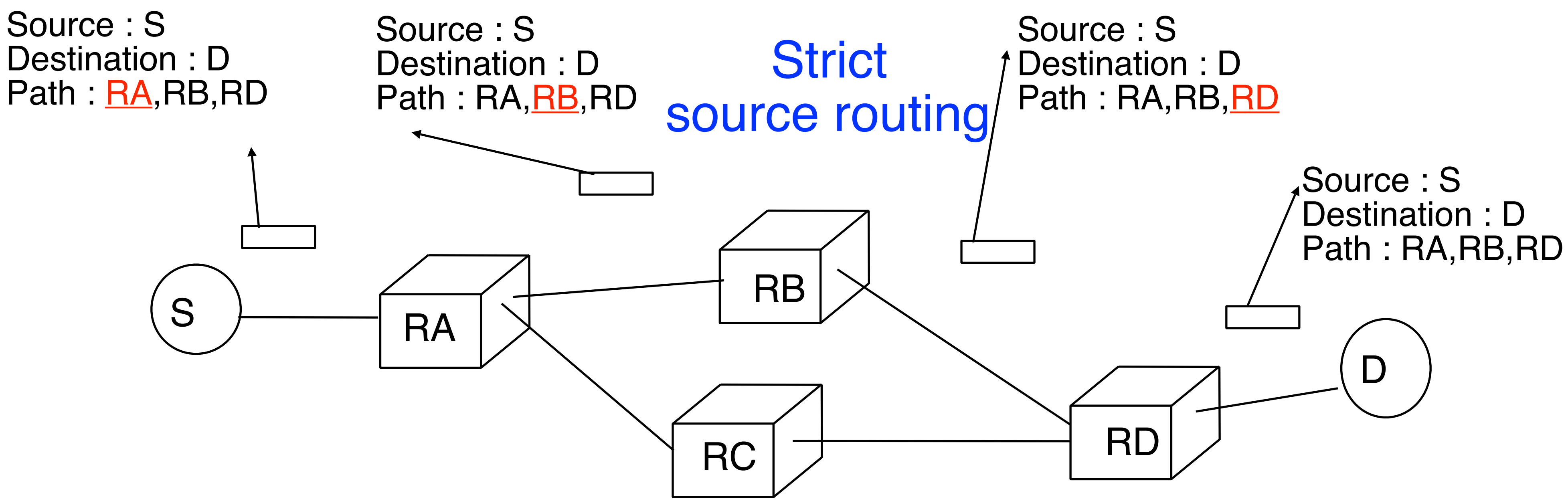
IP Source Routing



IP Source Routing

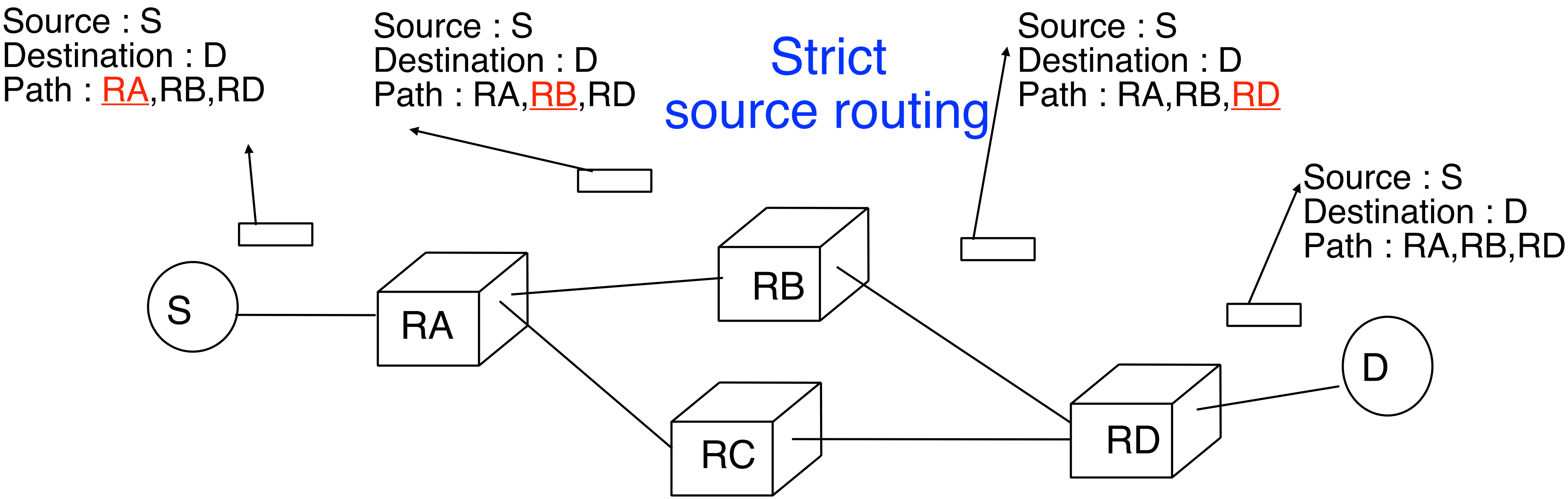


IP Source Routing

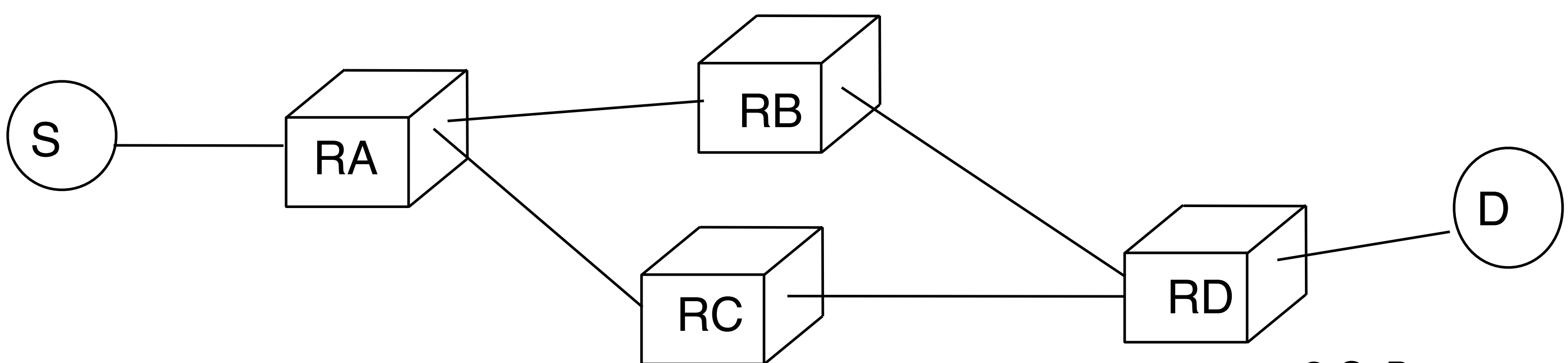


IP Source Routing

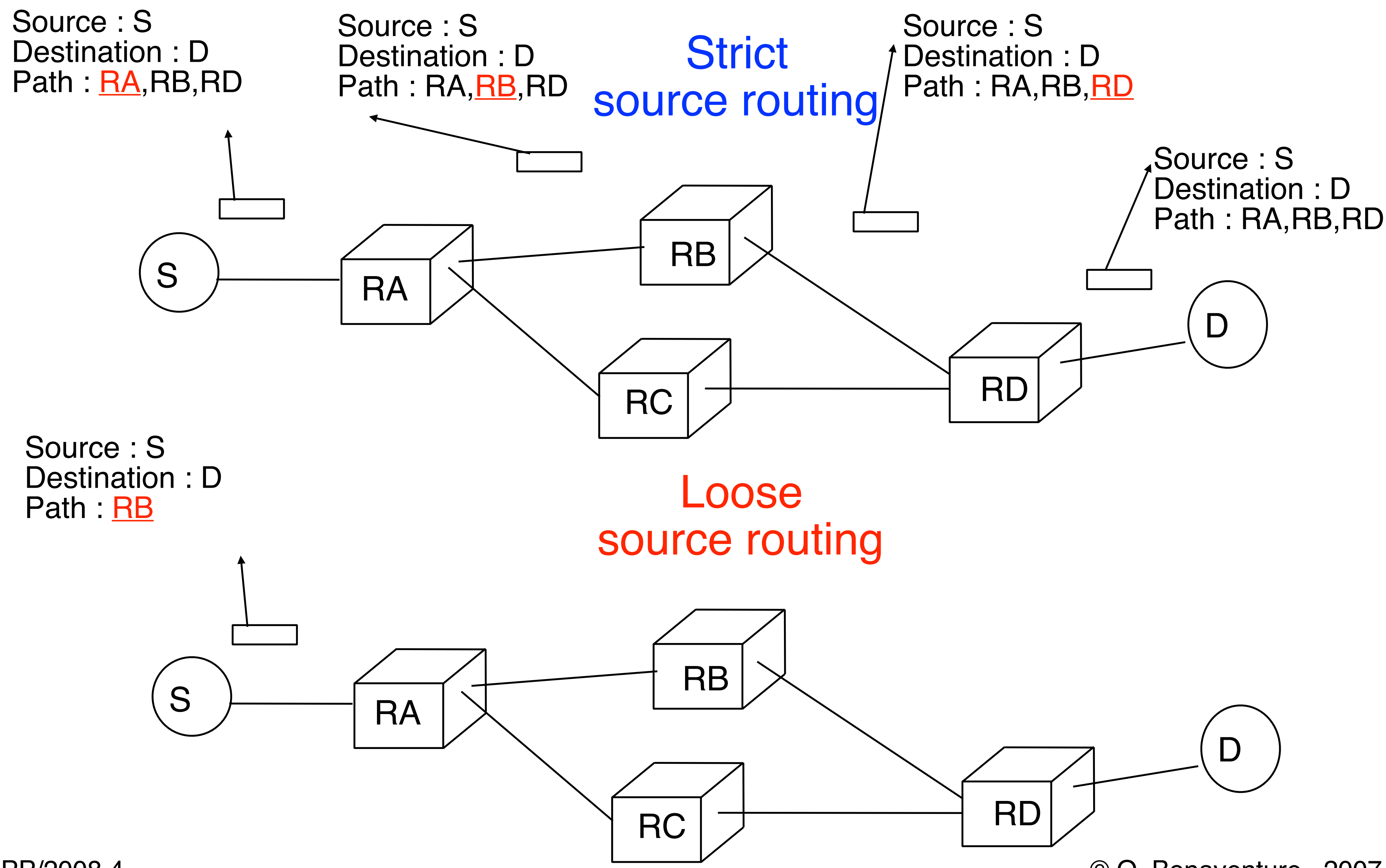
Strict source routing



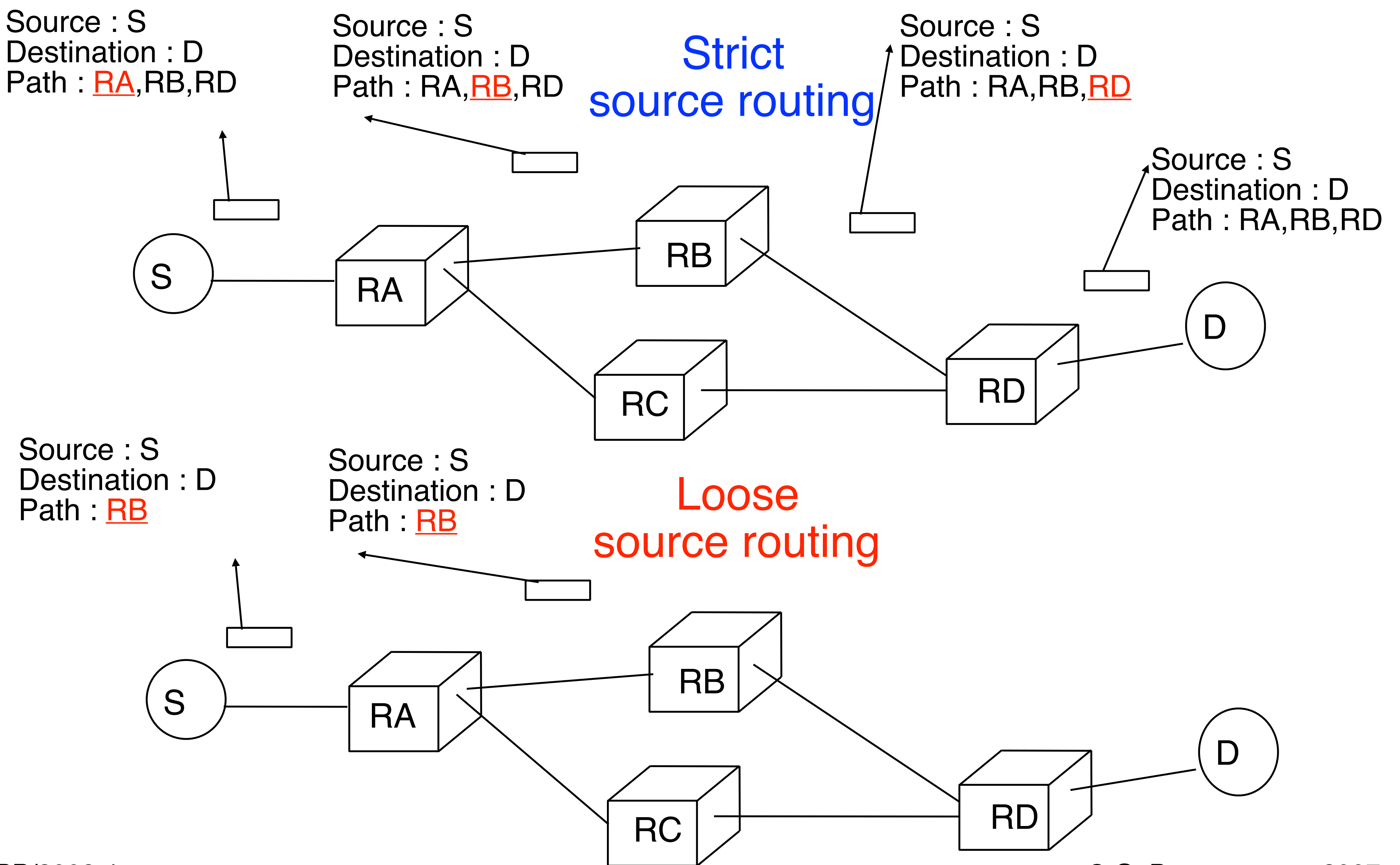
Loose source routing



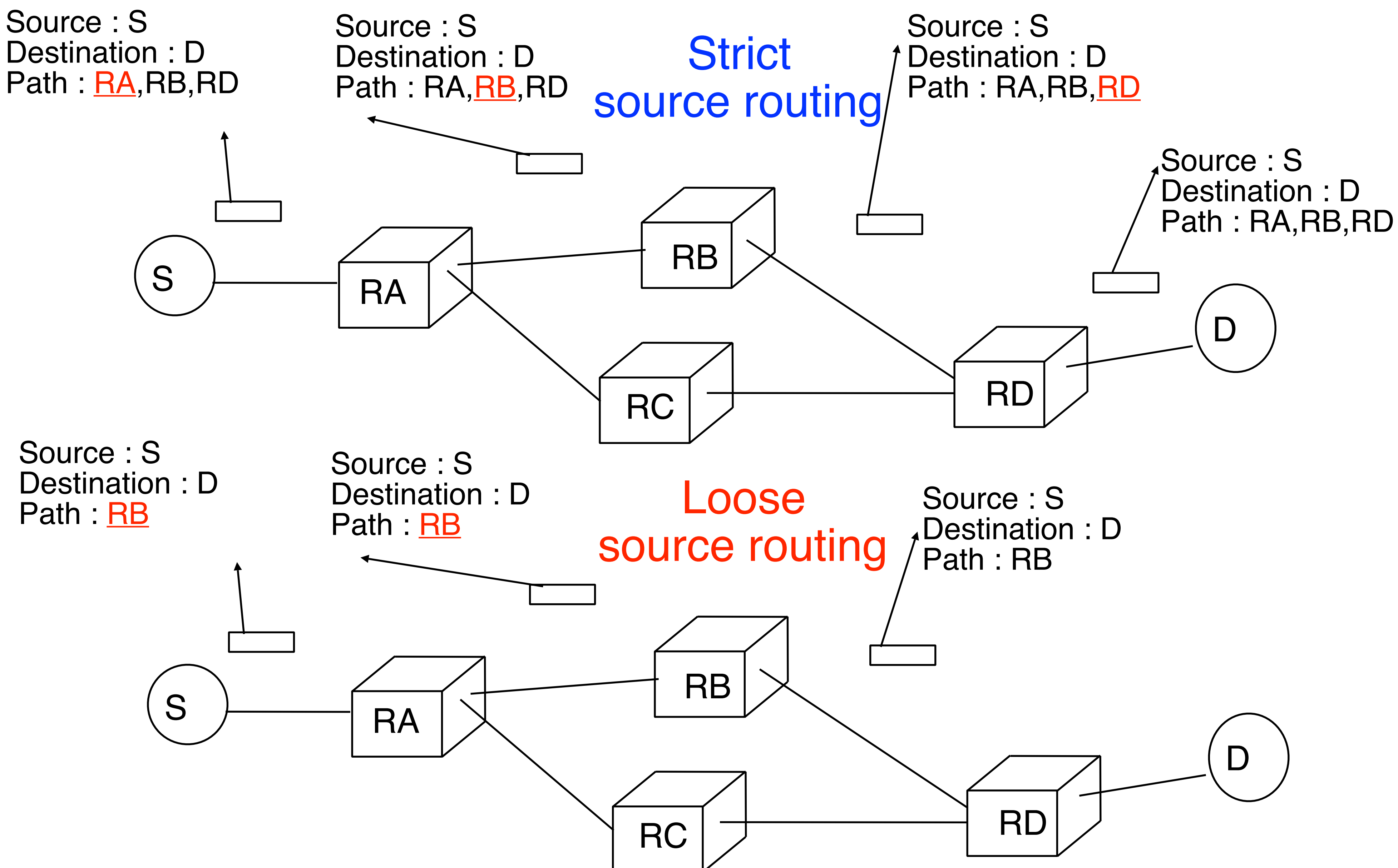
IP Source Routing



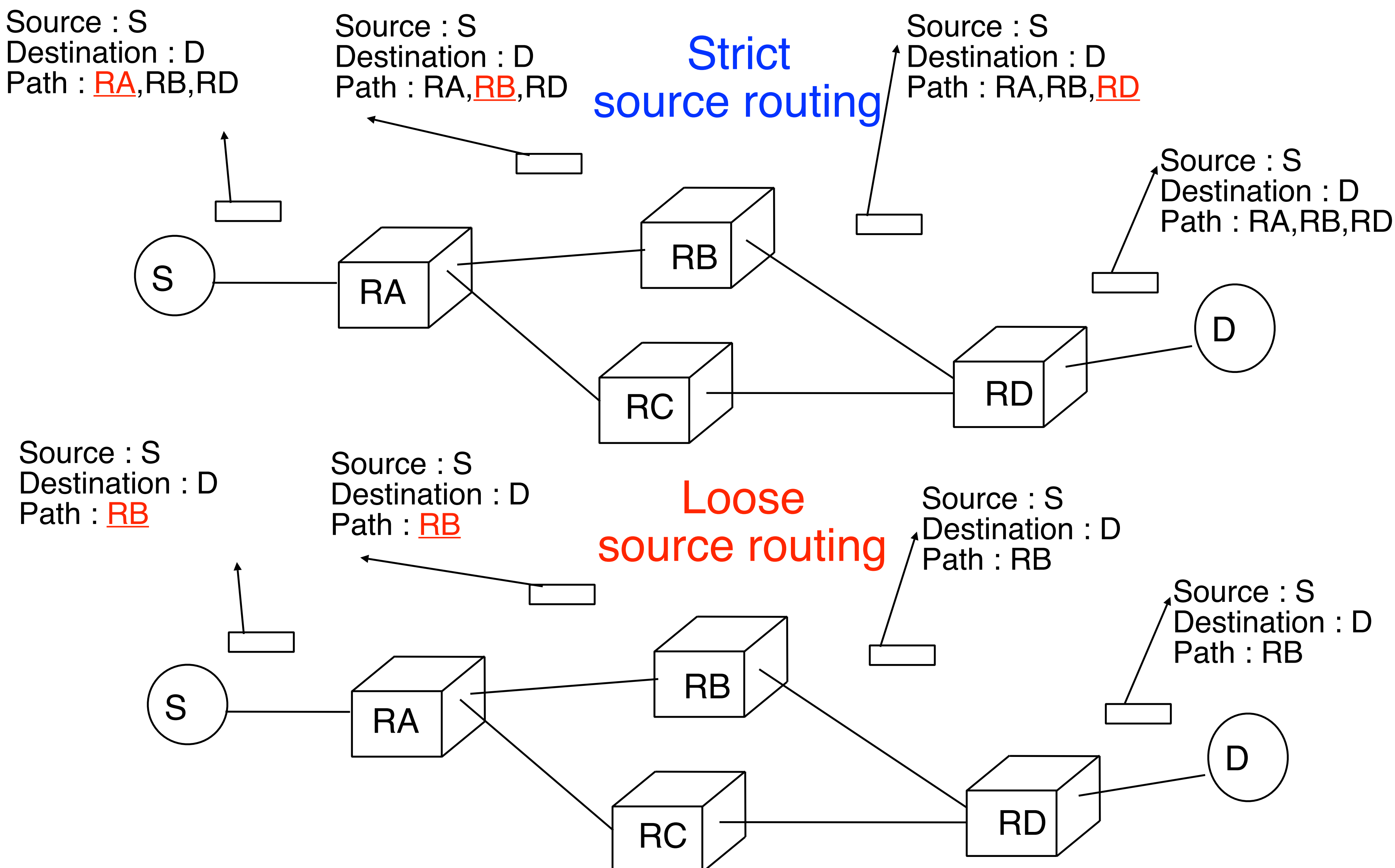
IP Source Routing



IP Source Routing



IP Source Routing



Operation of an IP endhost

- Required information on an IP endhost
  - IP addresses of its interfaces
    - For each address, the subnet mask allows the endhost to determine the addresses that are directly reachable through the interface
  - (small) routing table
    - Directly connected subnets
      - From the subnet mask of its own IP addresses
    - Default router
      - Router used to reach any unknown address
      - By convention, default route is 0.0.0.0/0
    - Other subnets known by endhost
      - Could be manually configured or learned through routing protocols are special packets (see later)

IP address configuration

- How does a host know its IP address
  - Manual configuration
    - Used in many small networks

  - Server-based autoconfiguration RARP
    - DHCP
      - Dynamic Host Configuration Protocol
      - Principle
      - When it attaches to a subnet, endhost broadcasts a request to find DHCP server
      - DHCP server replies and endhost can contact it to obtain IP address
      - DHCP server allocates an IP address for some time period and can also provide additional information (subnet, default router, DNS resolver, ...)
        - DHCP servers can be configured to always provide the same IP address to a given endhost or not
        - Endhost reconfirms its allocation regularly
  - Serverless autoconfiguration
    - Used by IPv6

Operation of an IP router

- Required information on an IP router
  - IP addresses of its interfaces
    - For each address, the subnet mask allows the endhost to determine the addresses that are directly reachable through the interface
  - Routing table
    - Directly connected subnets
      - From the subnet mask of its own IP addresses
    - Other known subnets
      - Usually learned via routing protocols, sometimes manually configured
    - Default router
      - Router used to reach any unknown address
      - By convention, default route is 0.0.0.0/0

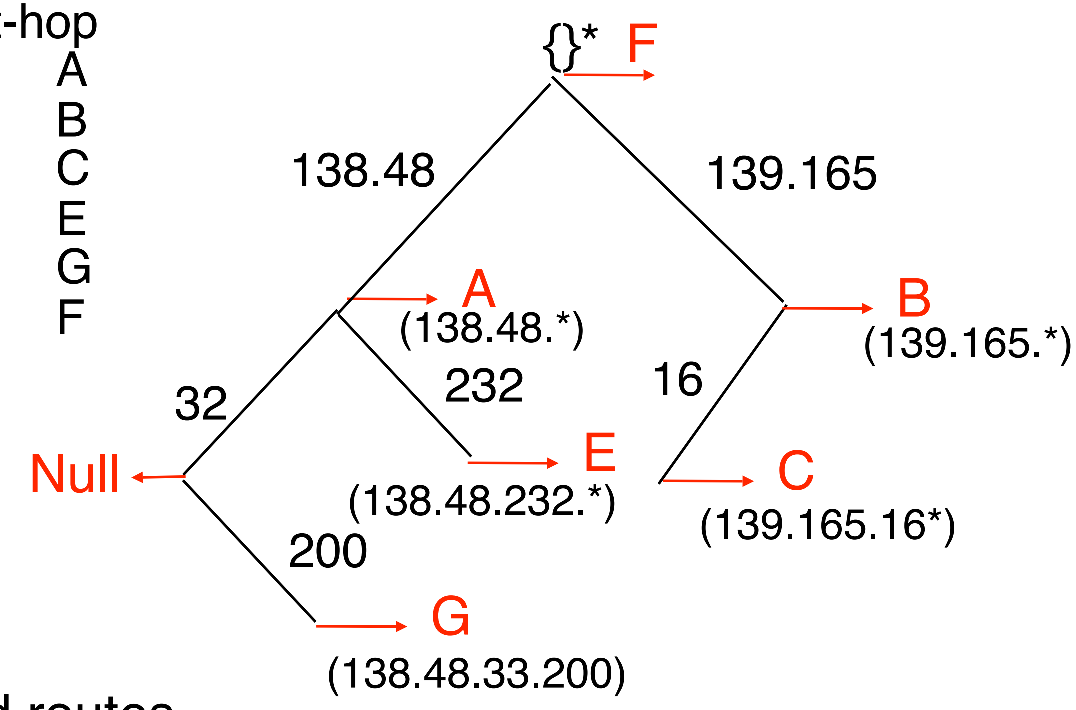
Operation of an IP router (2)

- Operations performed for each packet
  1. Check whether the packet's destination address is one of the router's addresses
     - If yes, packet reached destination
  2. Query Forwarding Information Base that contains
     - list of directly connected networks with masks
     - list of reachable networks and intermediate router
  3. Lookup the most specific route in FIB
     - For each route A.B.C.D/M via Rx
     - compare M higher order bits of destination address with M higher order bits of routes to find longest match
     - forward packet along this route

Forwarding Information Base Lookup

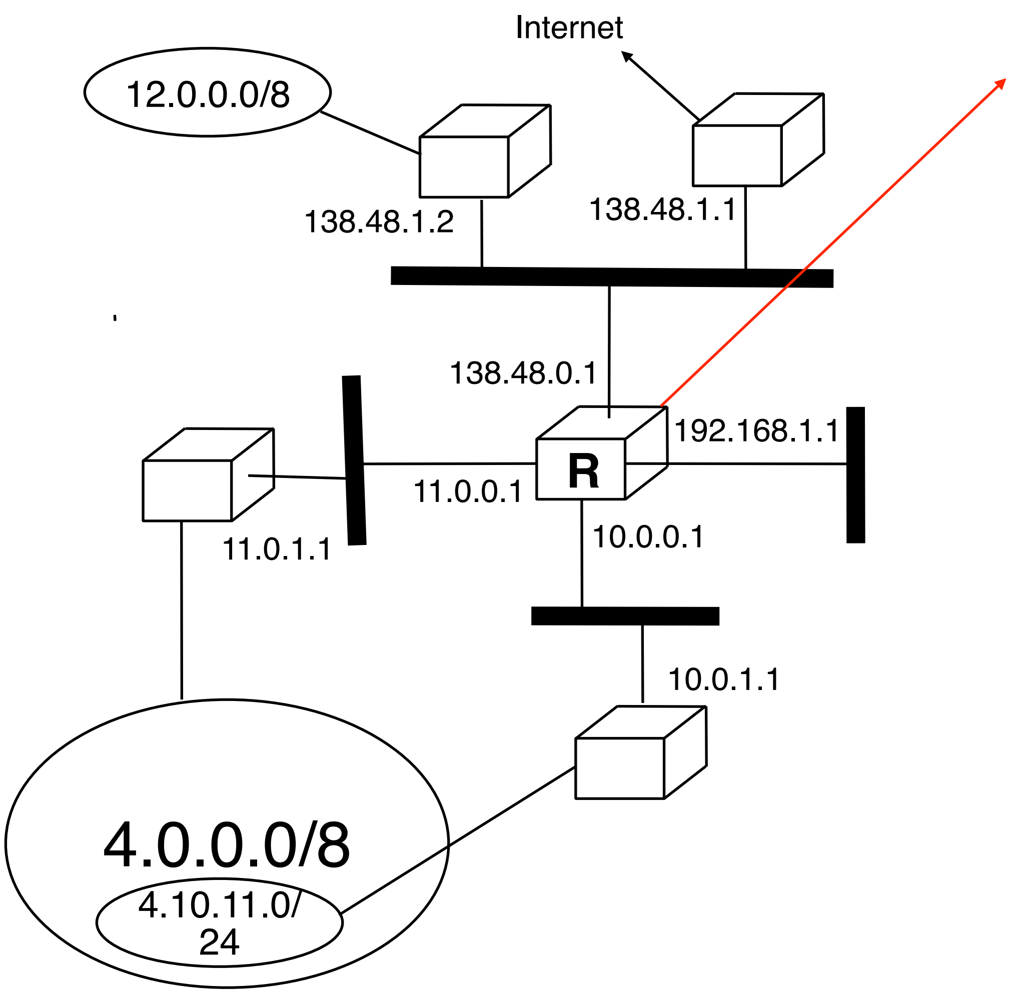
- How to find most specific route ?
  - similar to longest prefix match in a text
  - Trie

Subnet	Prefix	Next-hop
138.48.0.0	16	A
139.165.0.0	16	B
139.165.16.0	24	C
138.48.232.0	24	E
138.48.32.200	32	G
0.0.0.0	0	F



Cost of lookup
- f(average length of prefixes)
  - comparisons
  - memory accesses
- caches for most frequently used routes

Example



Local addresses
11.0.0.1 ; 138.48.0.1 ; 192.168.1.1 ; 10.0.0.1

Routing table
138.48.0.0/16 INorth]
11.0.0.0/8 [West]
192.168.1.0/24 [East]
10.0.0.0/8 [South]
4.0.0.0/8 via 11.0.1.1 [West]
4.10.11.0/24 via 10.0.1.1 [South]
12.0.0.0/8 via 138.48.1.2 [North]
0.0.0.0/0 via 138.48.1.1 [North]

Handling IP packets in error

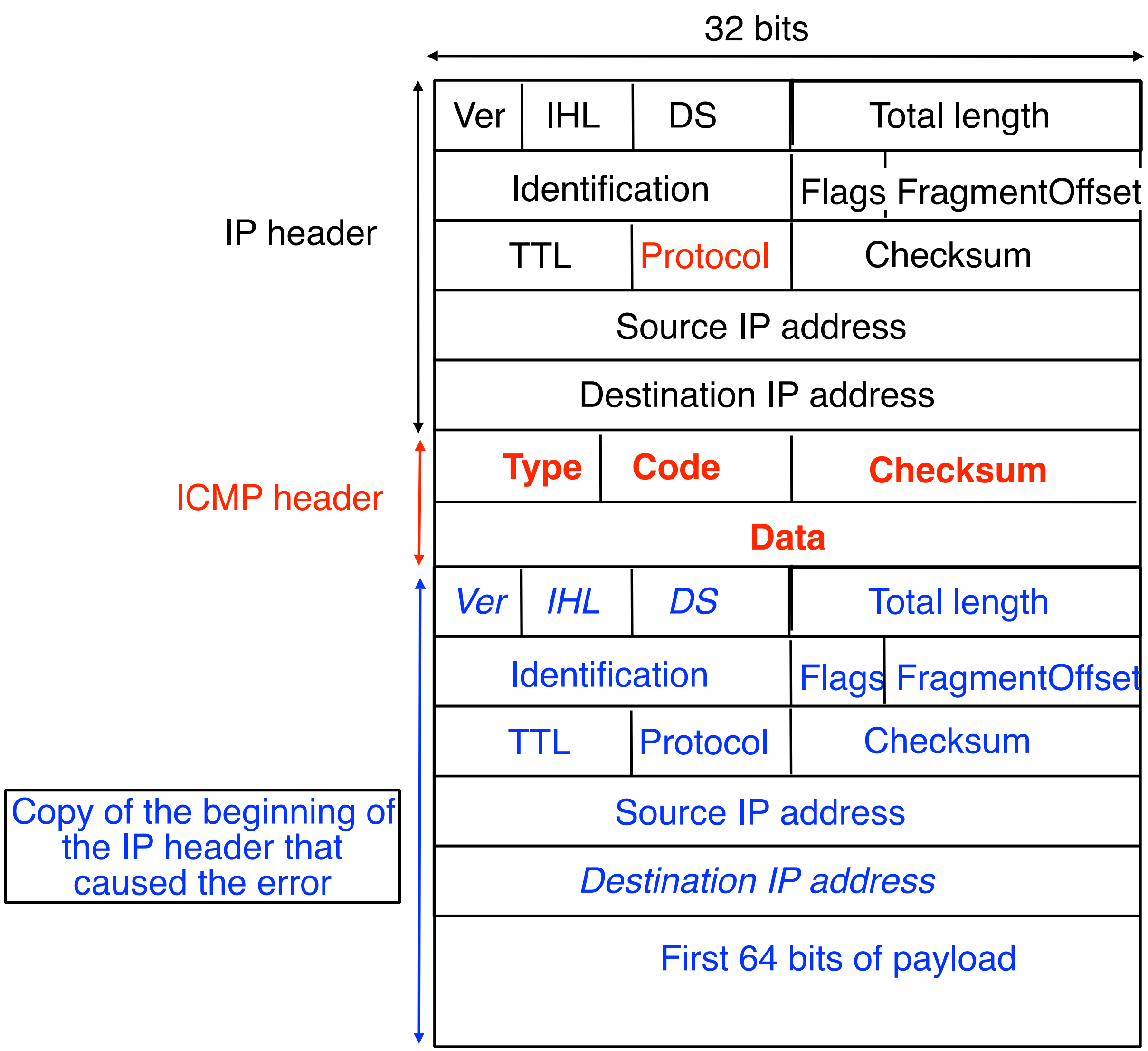
- Problem
  - What should a router/host do when it receives an errored packet
    - Example
      - Packet whose destination is not the current endhost
      - Packet containing a header with invalid syntax
      - Packet received with TTL=1
      - Packet destined to protocol not supported by host
- Solutions
  - Ignore and discard the errored packet
  - Send a message to the packet's source to warn it about the problem
    - ICMP : Internet Control Message Protocol
    - ICMP messages are sent inside IP packets by routers (mainly) and hosts
      - To avoid performance problems, most hosts/routers limit the amount of ICMP messages that they send

ICMP is defined in RFC792

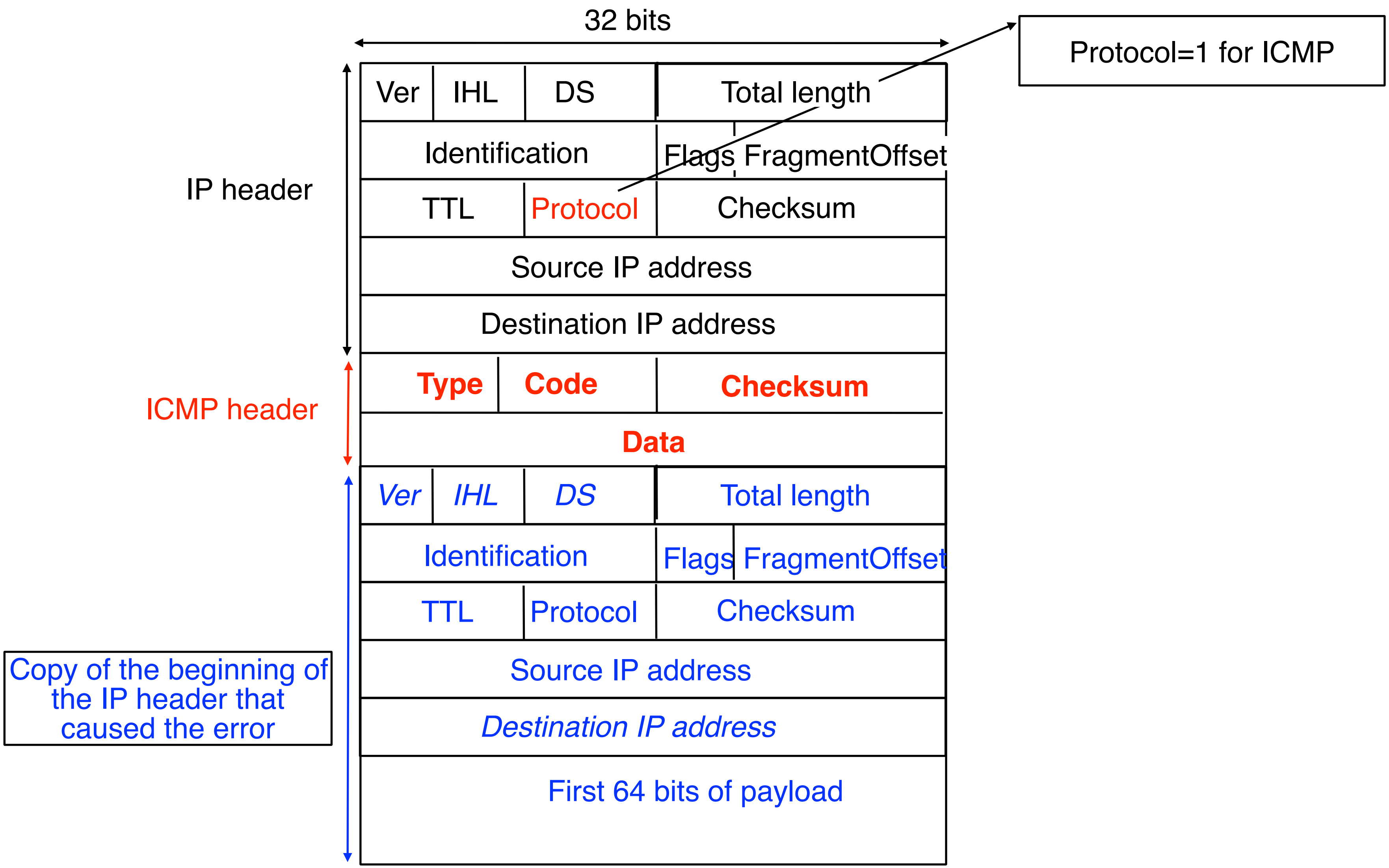
Sample ICMP messages

- Routing error
  - Destination unreachable
    - Final destination of packet cannot be reached
      - Network unreachable for entire subnet
      - Host unreachable for an individual host
      - Protocol/Port unreachable for protocol/port on a reachable host
    - Redirect
      - The packet was sent to an incorrect first-hop router and should have been instead sent to another first-hop router
- Error in the IP header
  - Parameter Problem
    - Incorrect format of IP packet
  - TTL Exceeded
    - Router received packet with TTL=1
  - Fragmentation
    - the packet should have been fragmented, but its DF flag was true

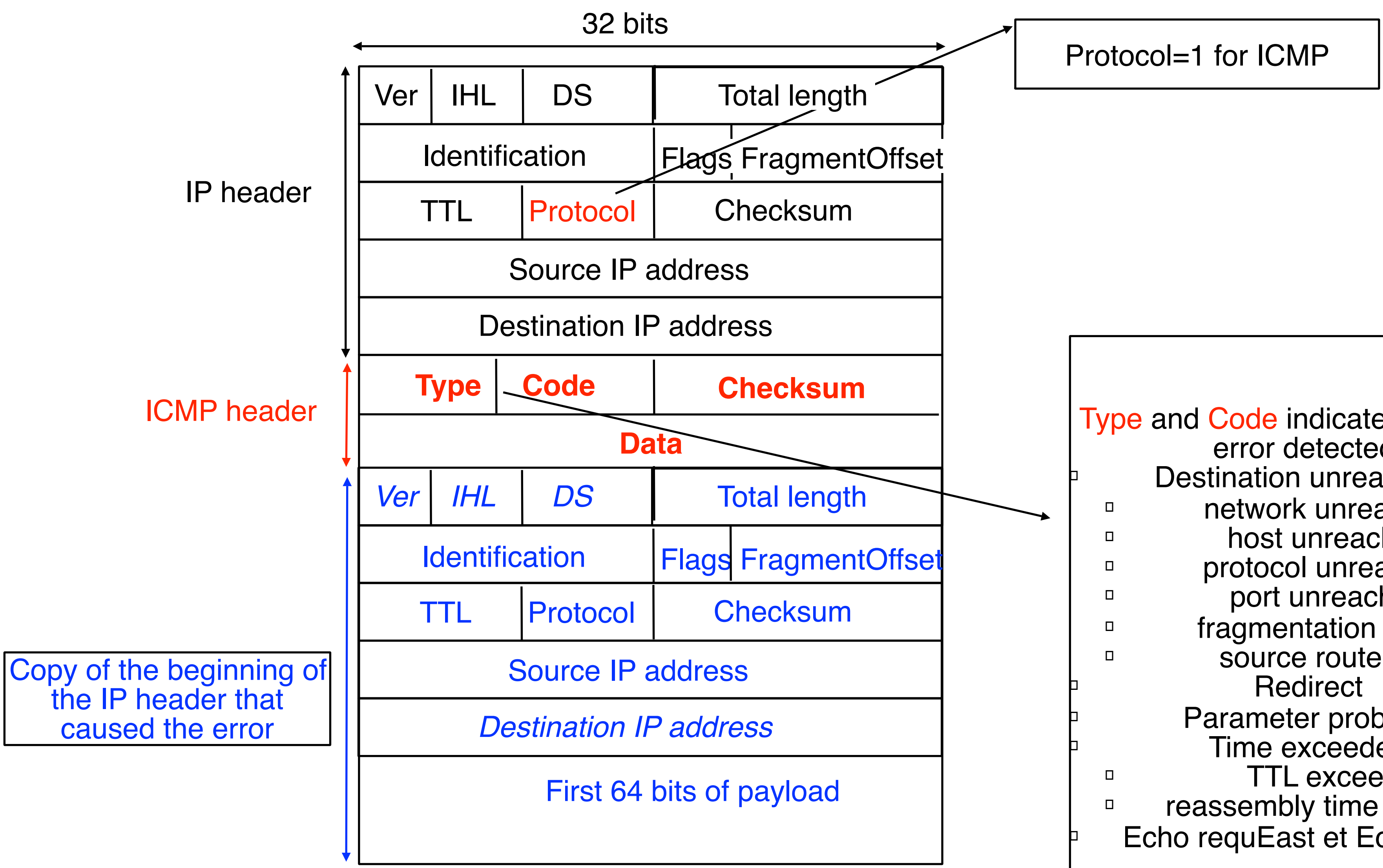
ICMP messages



ICMP messages



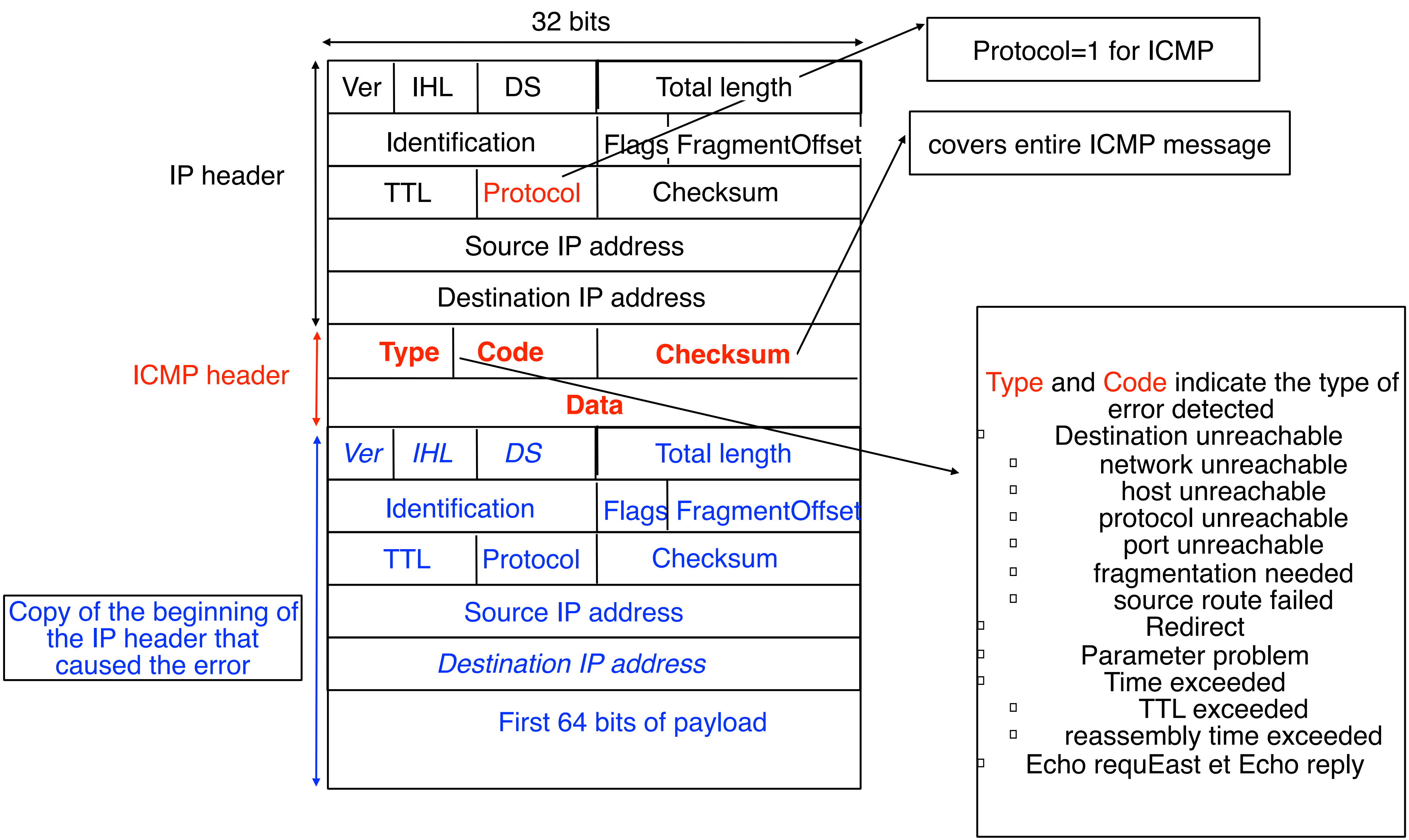
ICMP messages



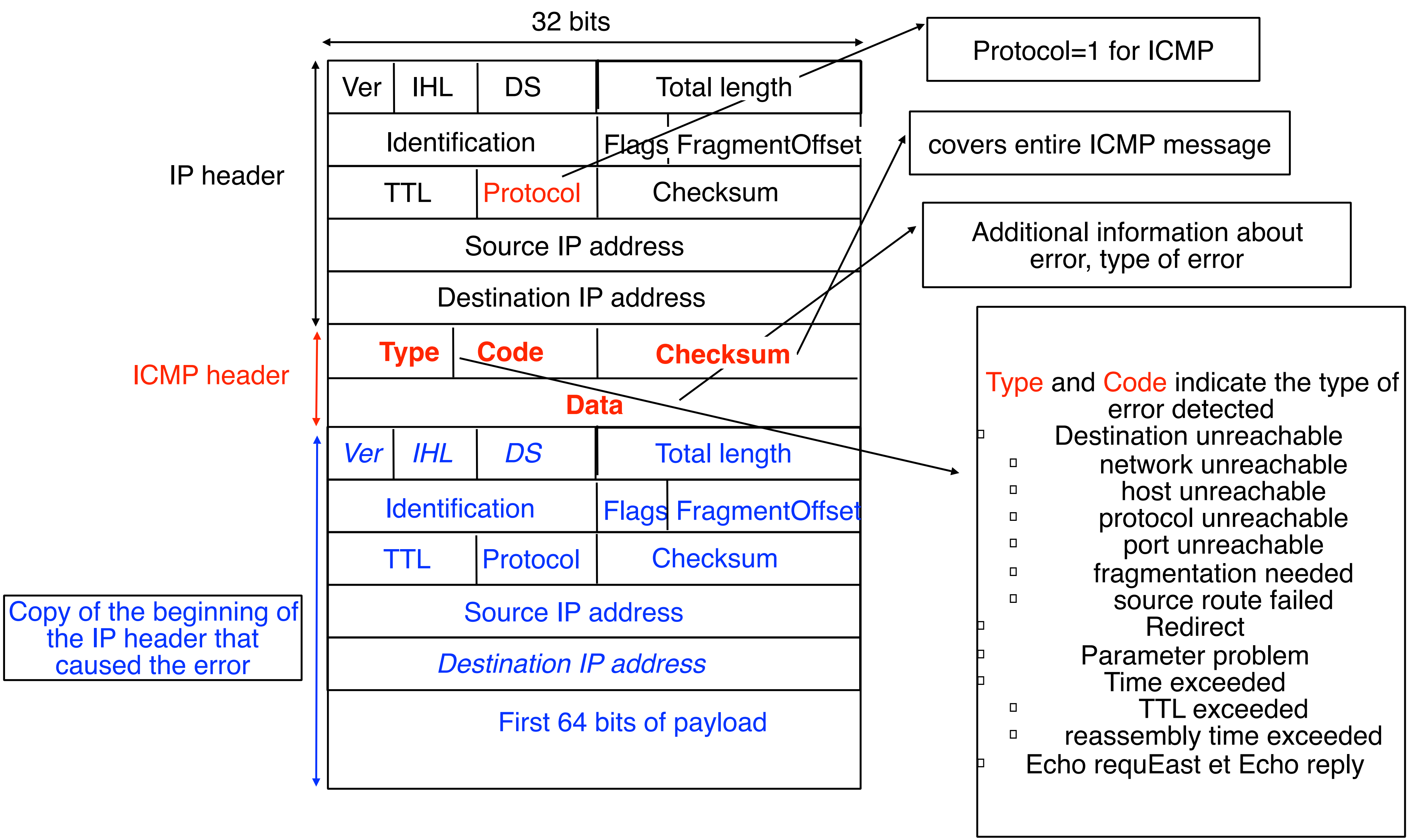
Type and Code indicate the type of error detected

- Destination unreachable
  - network unreachable
  - host unreachable
  - protocol unreachable
  - port unreachable
  - fragmentation needed
  - source route failed
- Redirect
- Parameter problem
- Time exceeded
  - TTL exceeded
  - reassembly time exceeded
- Echo requEast et Echo reply

ICMP messages



ICMP messages



Usage of ICMP messages

- Examples
  - destination unreachable
    - the router sending this message did not have a route to reach the destination
  - time exceeded
    - the router sending the message received an IP packet with TTL=0
    - used by `traceroute`
  - redirect
    - to reach destination, another router must be used and ICMP message provides address of this router
  - echo request / echo reply
    - used by `ping`
  - fragmentation impossible
    - the packet should have been fragmented by the router sending the ICMP message by this packet had "Don't Fragment" set to true

Network layer

- Basics
- Routing
  - Static routing
  - Distance vector routing
  - Link state routing
- IP : Internet Protocol
  - IP version 4
  - IP version 6
- Routing in IP networks

Issues with IPv4

- Late 1980s
  - Exponential growth of Internet
- 1990
  - Other network protocols exist
  - Governments push for CLNP
- 1992
  - Most class B networks will soon be assigned
  - Networking experts warn that IPv4 address space could become exhausted

Issues with IPv4 (2)

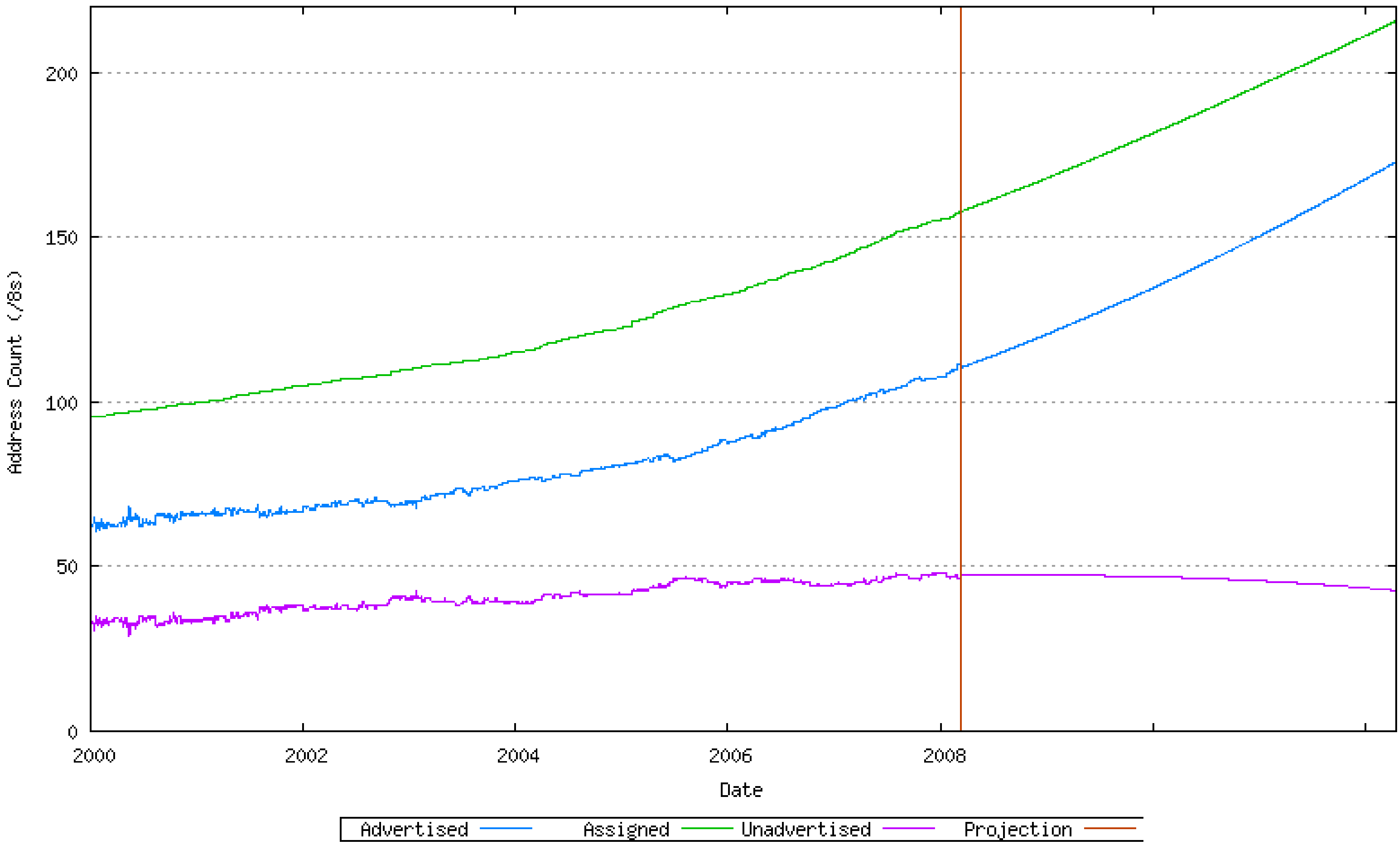
- How to solve the exhaustion of class B addresses ?
- Short term solution
  - Define Classless Interdomain Routing (CIDR) and introduce the necessary changes in routers
  - Deployment started in 1994
- Long term solution
  - Develop Internet Protocol - next generation (IPng)
    - call for proposals RFC1550, Dec 1993
    - Criteria for choix, RFC1719 and RFC1726, Dec. 1994
    - Proposed solutions
      - TUBA - RFC1347, June 1992
      - PIP – RFC1621, RFC1622, May 1994
      - CATNIP – RFC1707, October 1994
      - SIP – RFC1710, October 1994
      - NIMROD – RFC1753, December 1994
      - ENCAPS – RFC1955, June 1996

Issues with IPv4 (3)

- Implementation issues - 1990s
  - IPv4 packet format is complex
  - IP forwarding is difficult in hardware

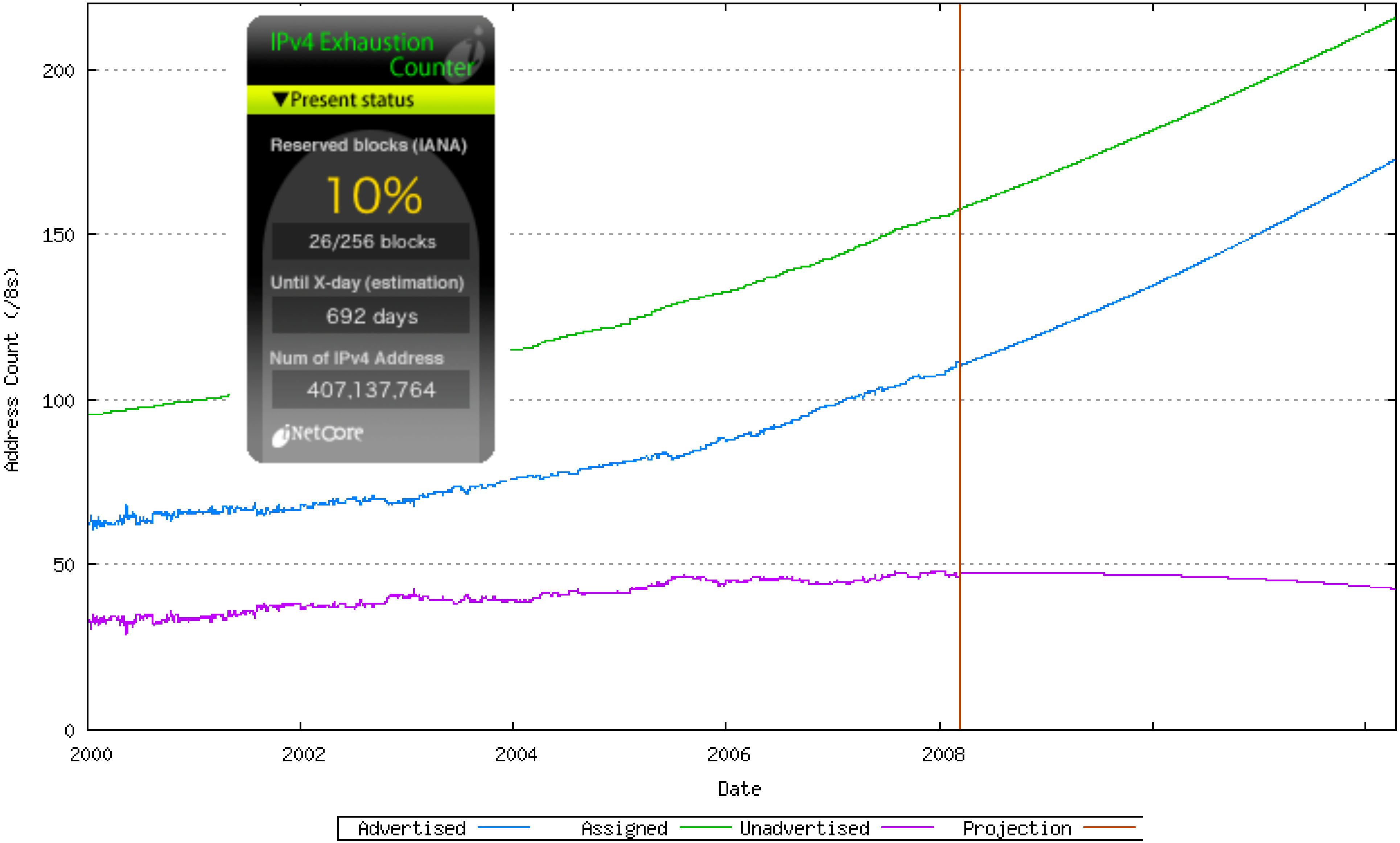
- Missing functions - 1990s
  - IPv4 requires lots of manual configuration
    - Competing protocols (CLNP, Appletalk, IPX, ...) already supported autoconfiguration in 1990s
  - How to support Quality of Service in IP ?
    - Integrated services and Differentiated services did not exist then
  - How to better support security in IP ?
    - Security problems started to appear but were less important than today
  - How to better support mobility in IP ?
    - GSM started to appear and some were dreaming of mobile devices attached to the Internet

Main motivation today
IPv4 address exhaustion



Source http://www.potaroo.net/tools/ipv4/index.html

Main motivation today IPv4 address exhaustion



Source http://www.potaroo.net/tools/ipv4/index.html

IPv6 addresses

IPv4

IP version 6

IPv6 addresses

IPv4

IP version 6

- Each IPv6 address is encoded in 128 bits
  - 3.4 x 10^38 possible addressable devices
    - 340,282,366,920,938,463,463,374,607,431,768,211,456
  - ~ 5 x 10^28 addresses per person on the earth
  - 6.65 x 10^23 addresses per square meter
  - Looks unlimited.... today

IPv6 addresses

IPv4

IP version 6

- Each IPv6 address is encoded in 128 bits
  - 3.4 x 10^38 possible addressable devices
    - 340,282,366,920,938,463,463,374,607,431,768,211,456
  - ~ 5 x 10^28 addresses per person on the earth
  - 6.65 x 10^23 addresses per square meter
  - Looks unlimited.... today
- Why 128 bits ?
  - Some wanted variable size addresses
    - to support IPv4 and 160 bits OSI NSAP
  - Some wanted 64 bits
    - Efficient for software, large enough for most needs
  - Hardware implementers preferred fixed size

The IPv6 addressing architecture

- Three types of IPv6 addresses
  - Unicast addresses
    - An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address
  - Anycast addresses
    - An identifier for a set of interfaces. A packet sent to an
    - anycast address is delivered to the “nearest” one of the interfaces identified by that address
  - Multicast addresses
    - An identifier for a set of interfaces. A packet sent to a multicast address is delivered to all interfaces identified by that address.

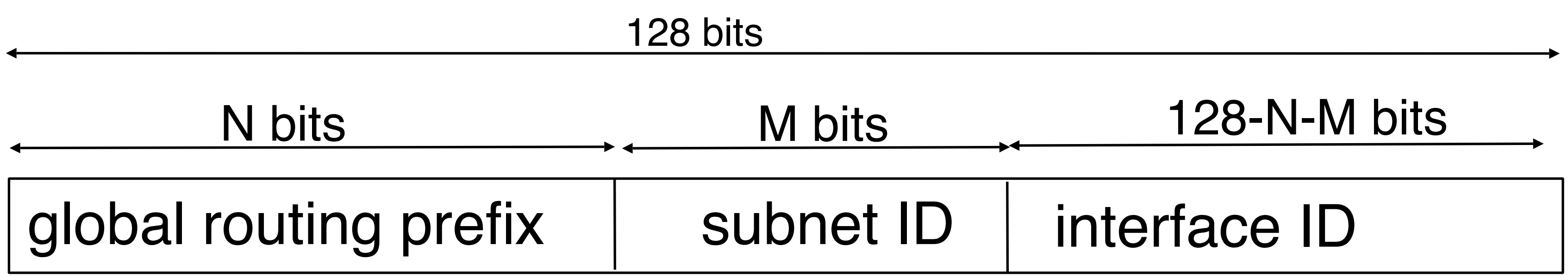
Representation of IPv6 addresses

- How can we write a 128 bits IPv6 address ?
  - Hexadecimal format
    - FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
    - 1080:0:0:0:8:800:200C:417A
  - Compact hexadecimal format
    - Some IPv6 addresses contain lots of zero
      - use "::" to indicate one or more groups of 16 bits of zeros. The "::" can only appear once in an address
      - Examples
        - 1080:0:0:0:8:800:200C:417A = 1080::8:800:200C:417A
        - FF01:0:0:0:0:0:0:101 = FF01::101
        - 0:0:0:0:0:0:0:1 = ::1

The IPv6 unicast addresses

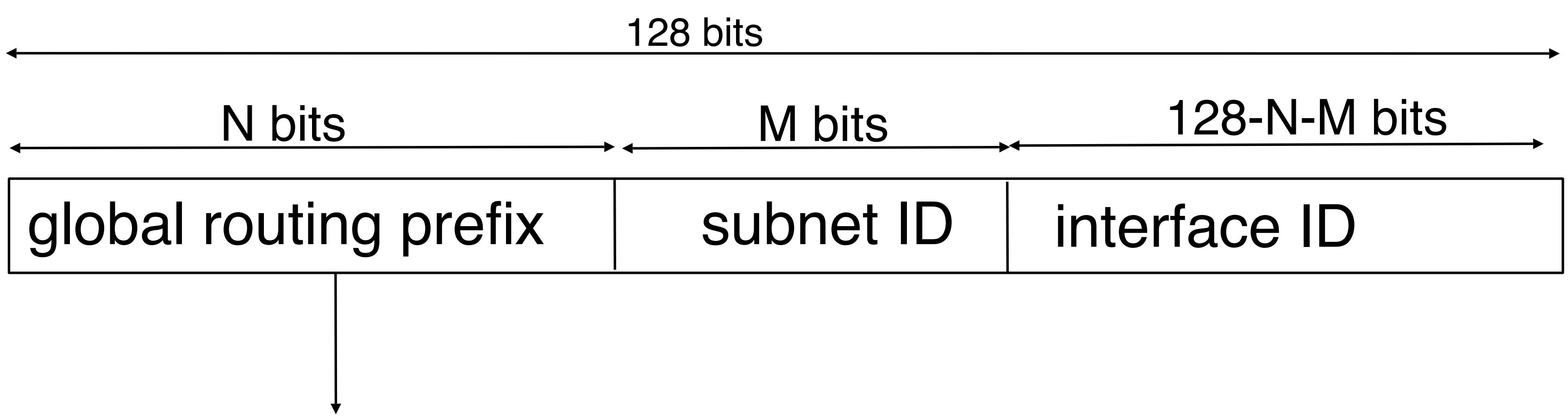
- Special addresses
  - Unspecified address : 0:0:0:0:0:0:0:0 (aka *::*)
  - Loopback address : 0:0:0:0:0:0:0:1 (aka *::1*)

- Global unicast addresses
  - Addresses will be allocated hierarchically



The IPv6 unicast addresses

- Special addresses
  - Unspecified address : 0:0:0:0:0:0:0:0 (aka *::*)
  - Loopback address : 0:0:0:0:0:0:0:1 (aka *::1*)
- Global unicast addresses
  - Addresses will be allocated hierarchically

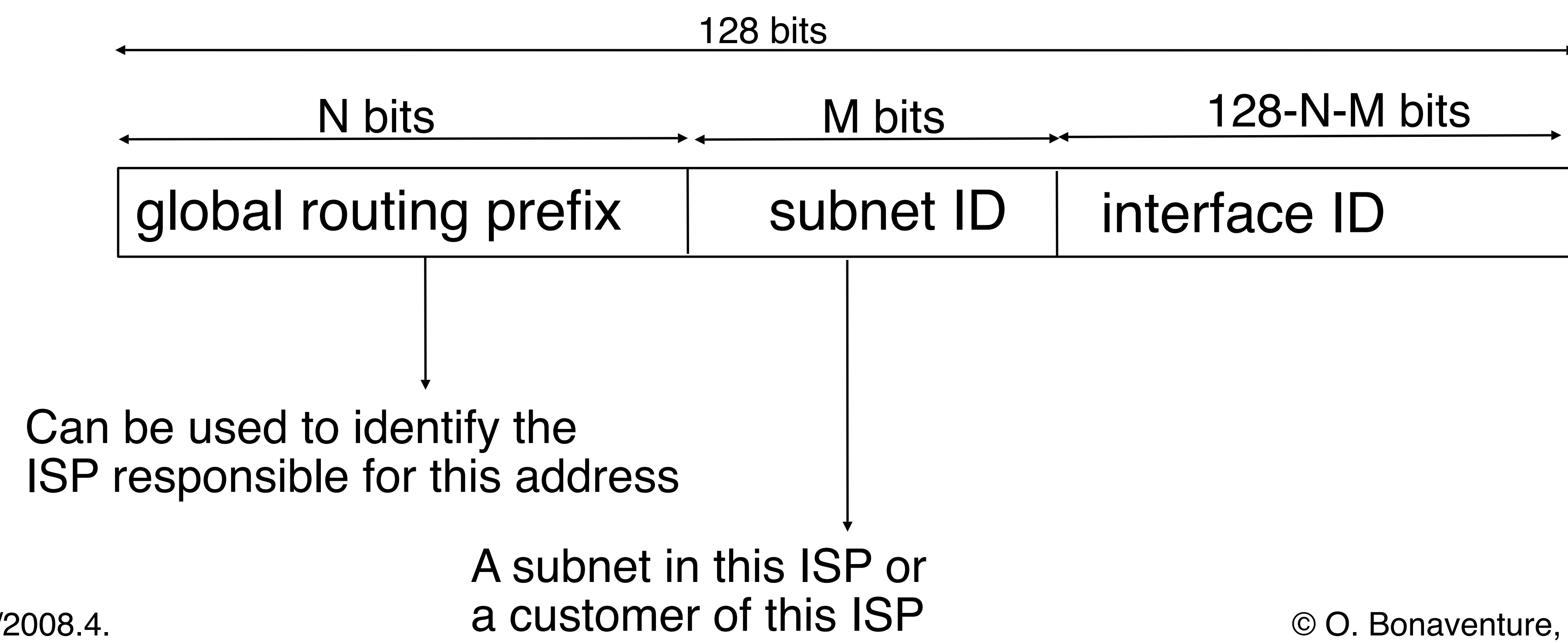


Can be used to identify the
ISP responsible for this address

The IPv6 unicast addresses

- Special addresses
  - Unspecified address : 0:0:0:0:0:0:0:0 (aka *::*)
  - Loopback address : 0:0:0:0:0:0:0:1 (aka *::1*)

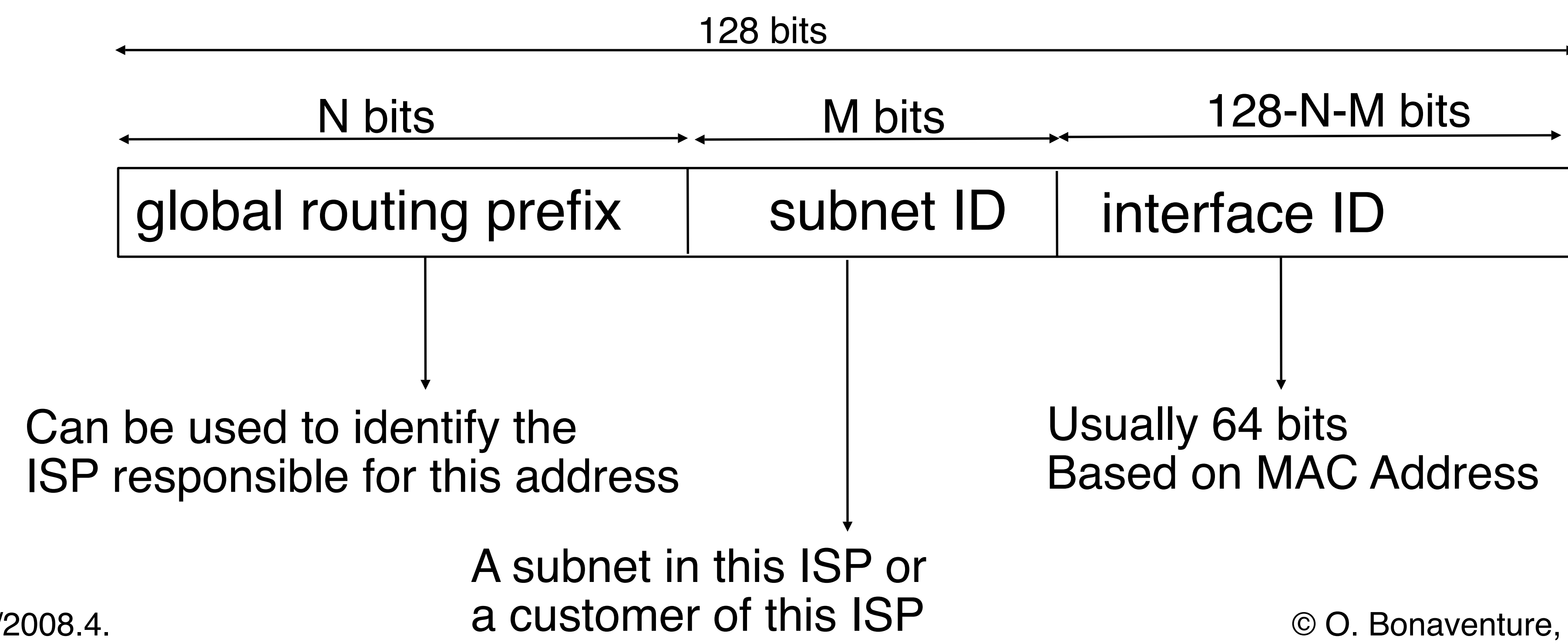
- Global unicast addresses
  - Addresses will be allocated hierarchically



The IPv6 unicast addresses

- Special addresses
  - Unspecified address : 0:0:0:0:0:0:0:0 (aka *::*)
  - Loopback address : 0:0:0:0:0:0:0:1 (aka *::1*)

- Global unicast addresses
  - Addresses will be allocated hierarchically



Allocation of IPv6 addresses

- IANA controls all IP addresses and delegates assignments of blocks to Regional IP Address Registries (RIR)
  - RIPE, ARIN, APNIC, AFRINIC, ...
- An organisation can be allocated two different types of IPv6 addresses
  - Provider Independent (PI) addresses
    - Usually allocated to ISPs or very large enterprises directly by RIRs
    - Default size is /32
  - Provider Aggregatable (PA) addresses
    - Smaller prefixes, assigned by ISPs from their PI block
    - Size
      - /48 in the general case, except for very large subscribers
      - /64 when t one and only one subnet is needed by design
      - /128 when it is absolutely known that one and only one device is connecting.

The IPv6 link-local addresses

- Used by hosts and routers attached to the same LAN to exchange IPv6 packets when they don't have/need globally routable addresses

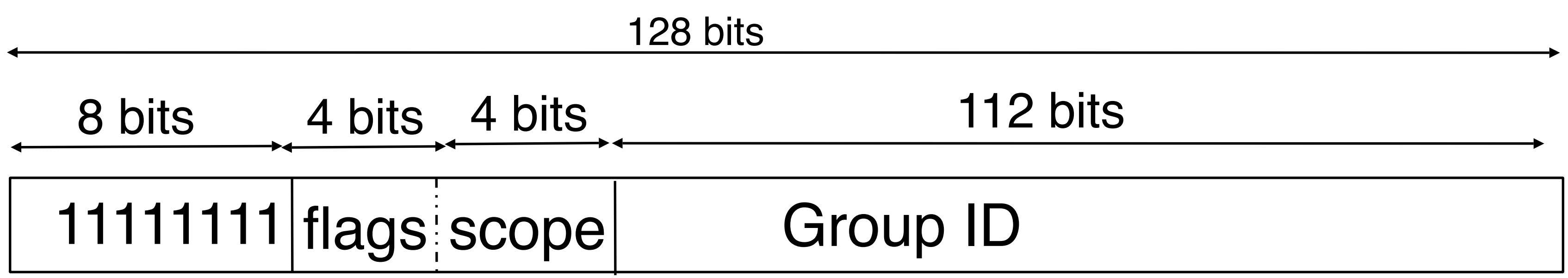
128 bits

10 bits	54 bits	64 bits
FE80	0000000000.....00000000000	interface ID

- Each host must generate one link local address for each of its interfaces
  - Each IPv6 host will use several IPv6 addresses
- Each routers must generate one link local address for each of its interfaces

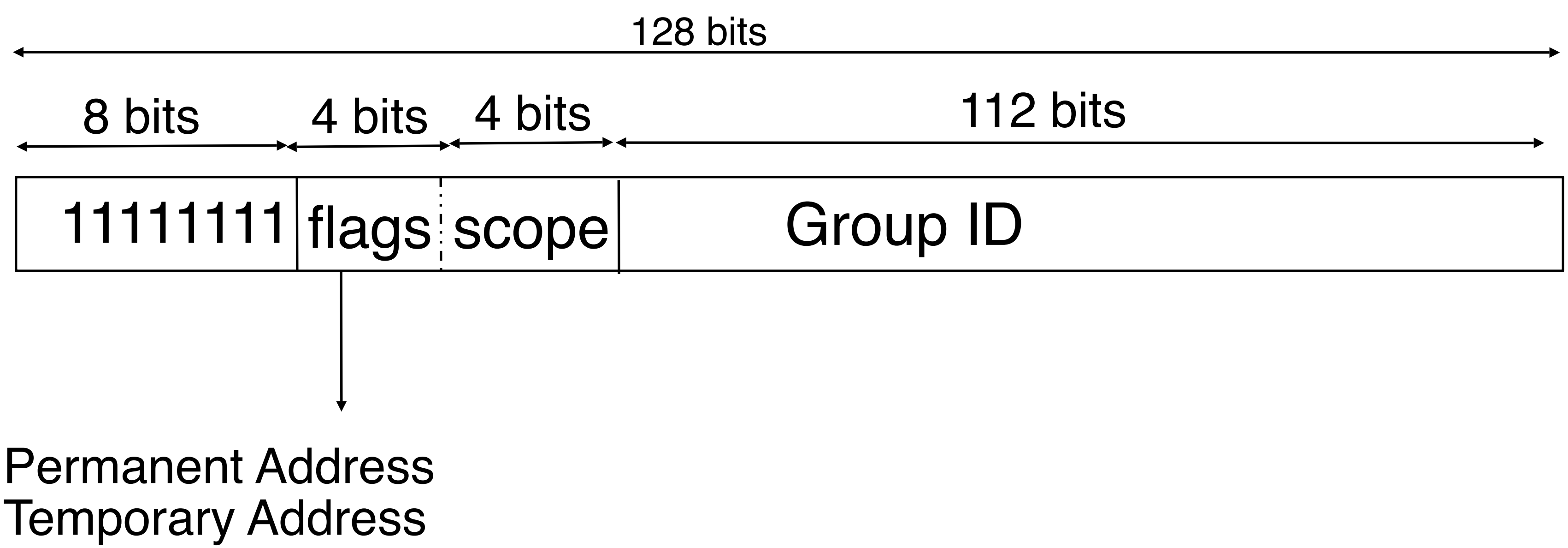
The IPv6 multicast addresses

- An IPv6 multicast address identifies a group a receivers



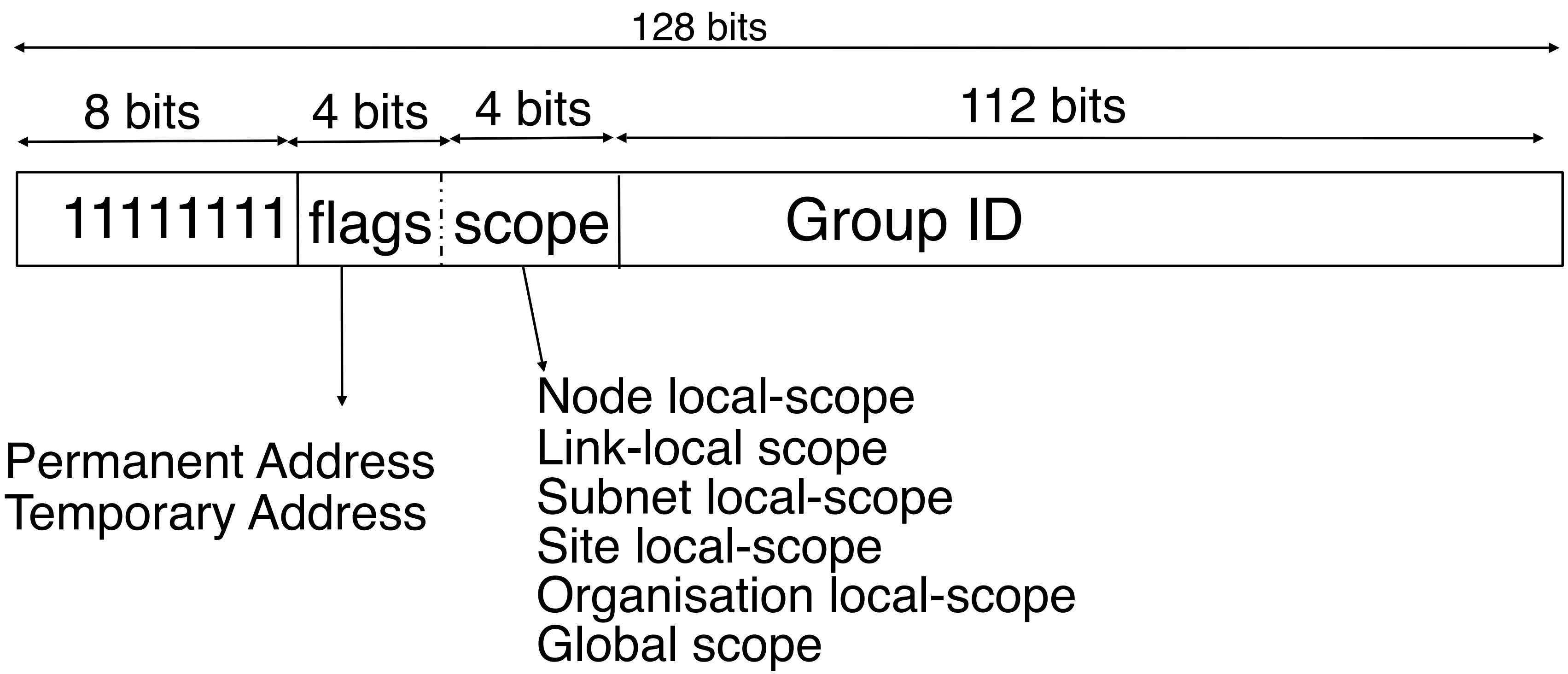
The IPv6 multicast addresses

- An IPv6 multicast address identifies a group a receivers



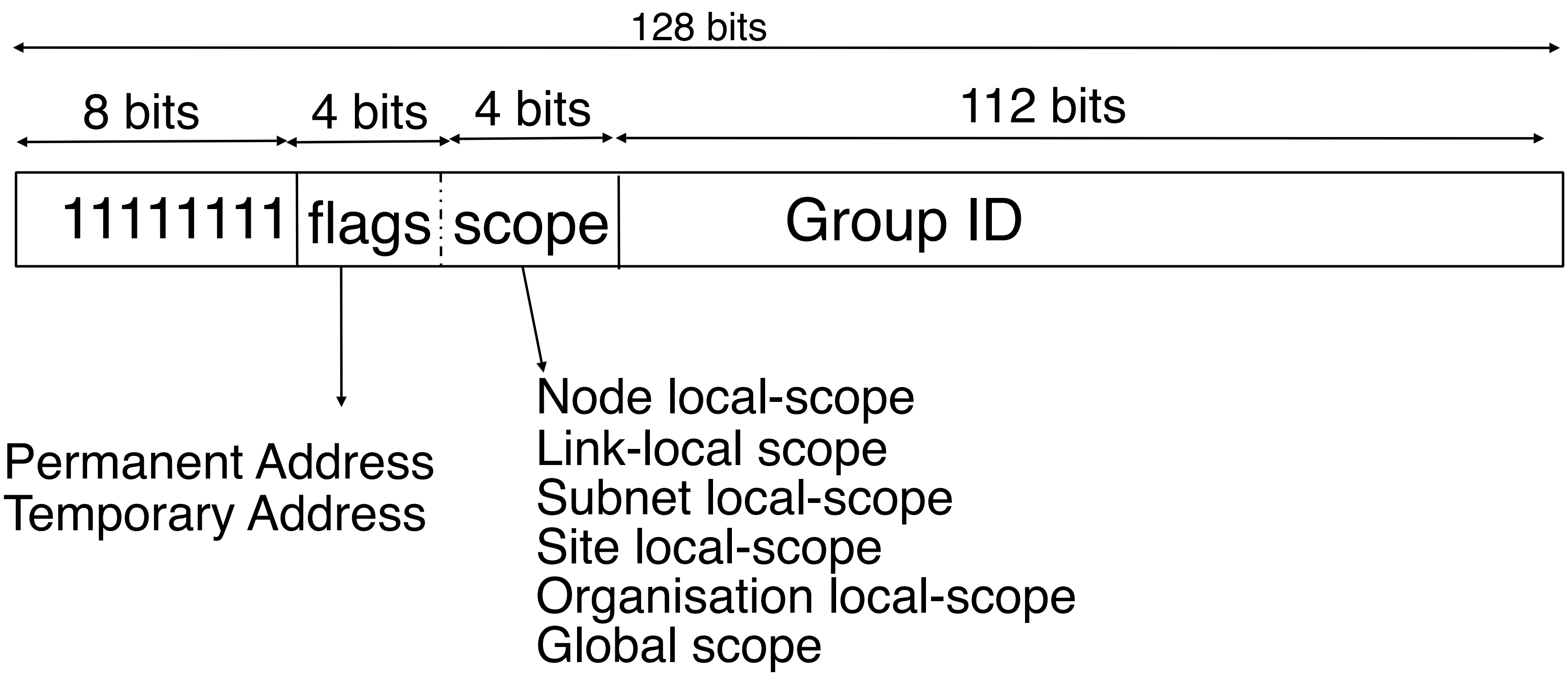
The IPv6 multicast addresses

- An IPv6 multicast address identifies a group a receivers



The IPv6 multicast addresses

- An IPv6 multicast address identifies a group a receivers



- Well known groups
  - All endsystem automatically belong to the FF02::1 group
  - All routers automatically belong to the FF02::2 group

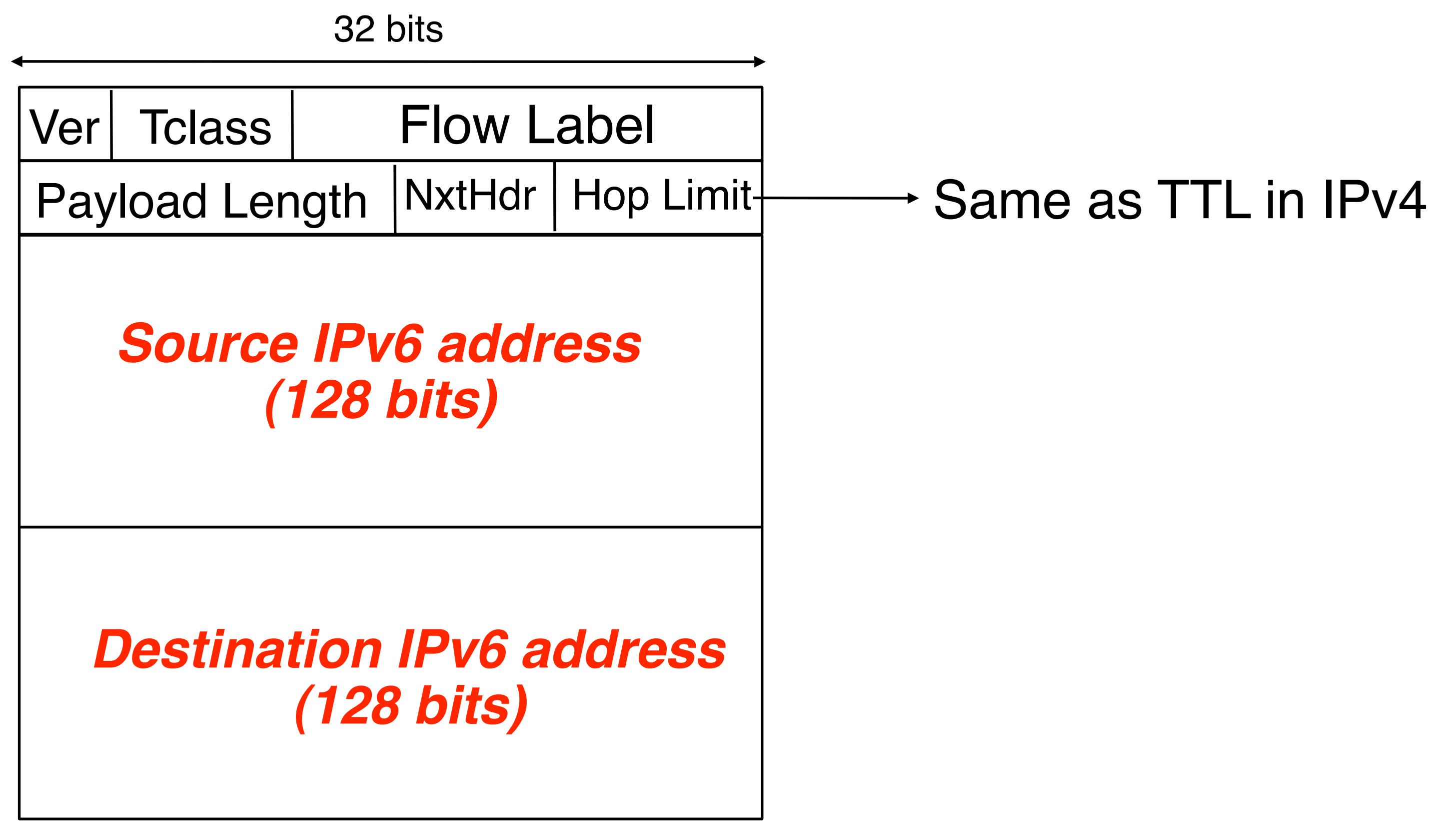
The IPv6 anycast addresses

- Definition
  - An IPv6 anycast address is an address that is assigned to more than one interface (typically belonging to different nodes), with the property that a packet sent to an anycast address is routed to the "nearest" interface having that address, according to the routing protocols' measure of distance.
- Usage
  - Multiple redundant servers using same address
  - Example DNS resolvers and DNS servers
- Representation
  - IPv6 anycast addresses are unicast addresses
  - Required subnet anycast address

n bits	128-n bits
IPv6 subnet prefix	00000000000000000000000

The IPv6 packet format

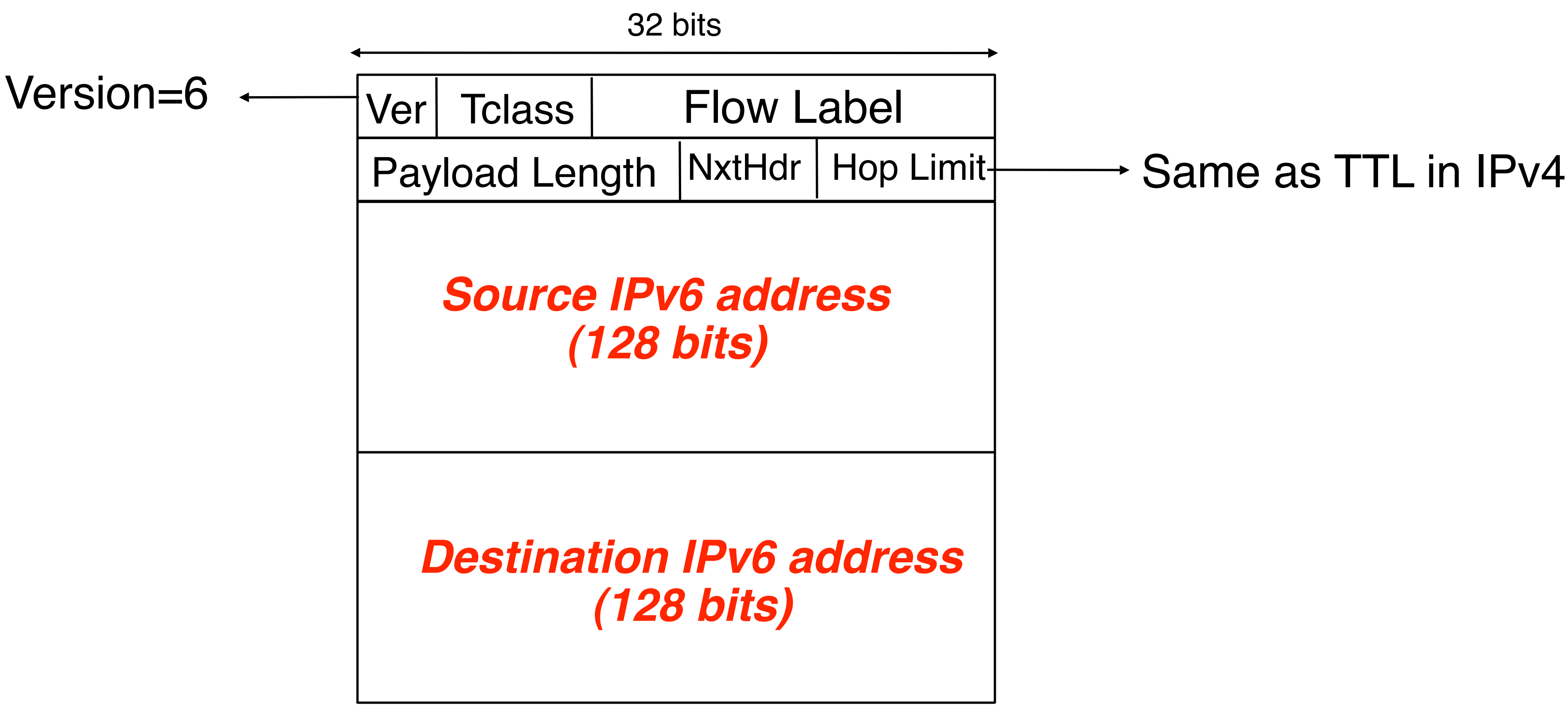
- Simplified packet format
  - Fields aligned on 32 bits boundaries to ease implementation



- No checksum in IPv6 header
  - rely on datalink and transport checksums

The IPv6 packet format

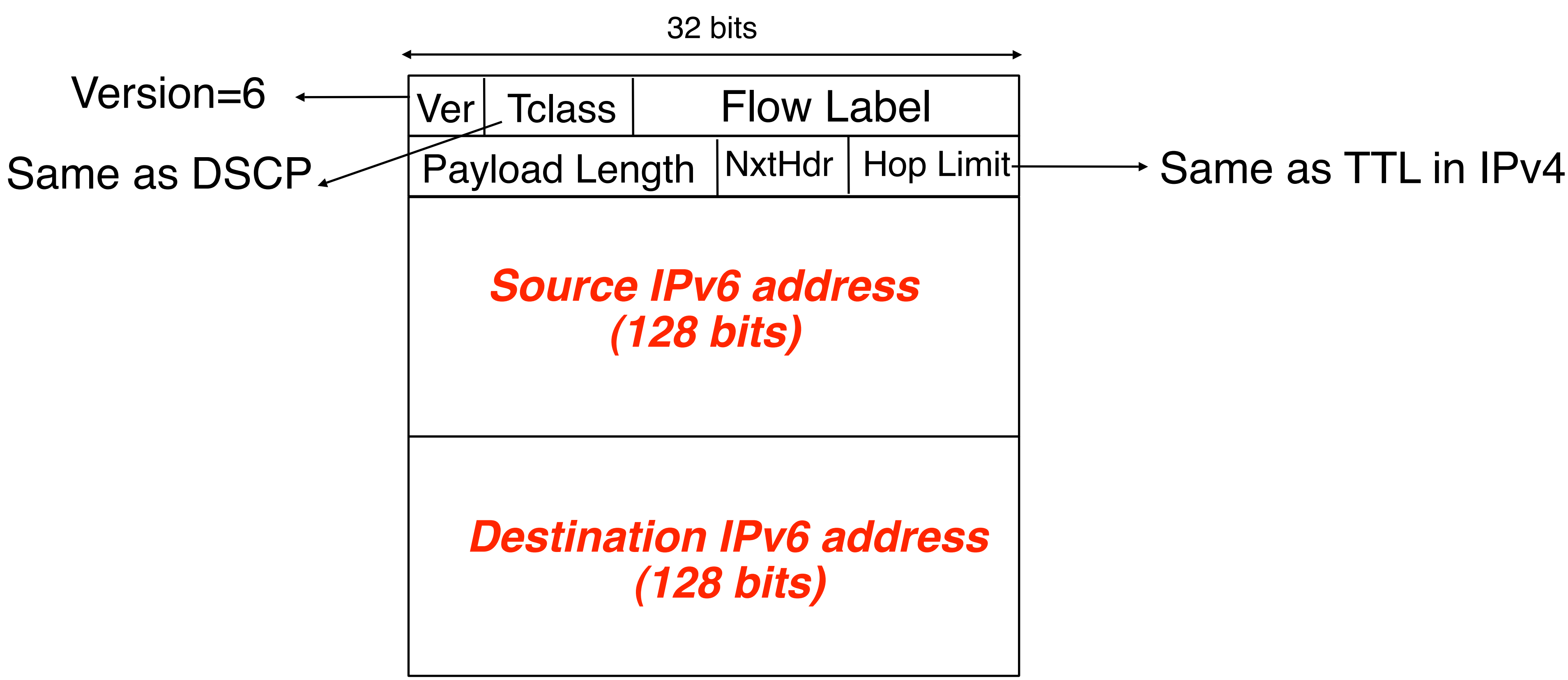
- Simplified packet format
  - Fields aligned on 32 bits boundaries to ease implementation



- No checksum in IPv6 header
  - rely on datalink and transport checksums

The IPv6 packet format

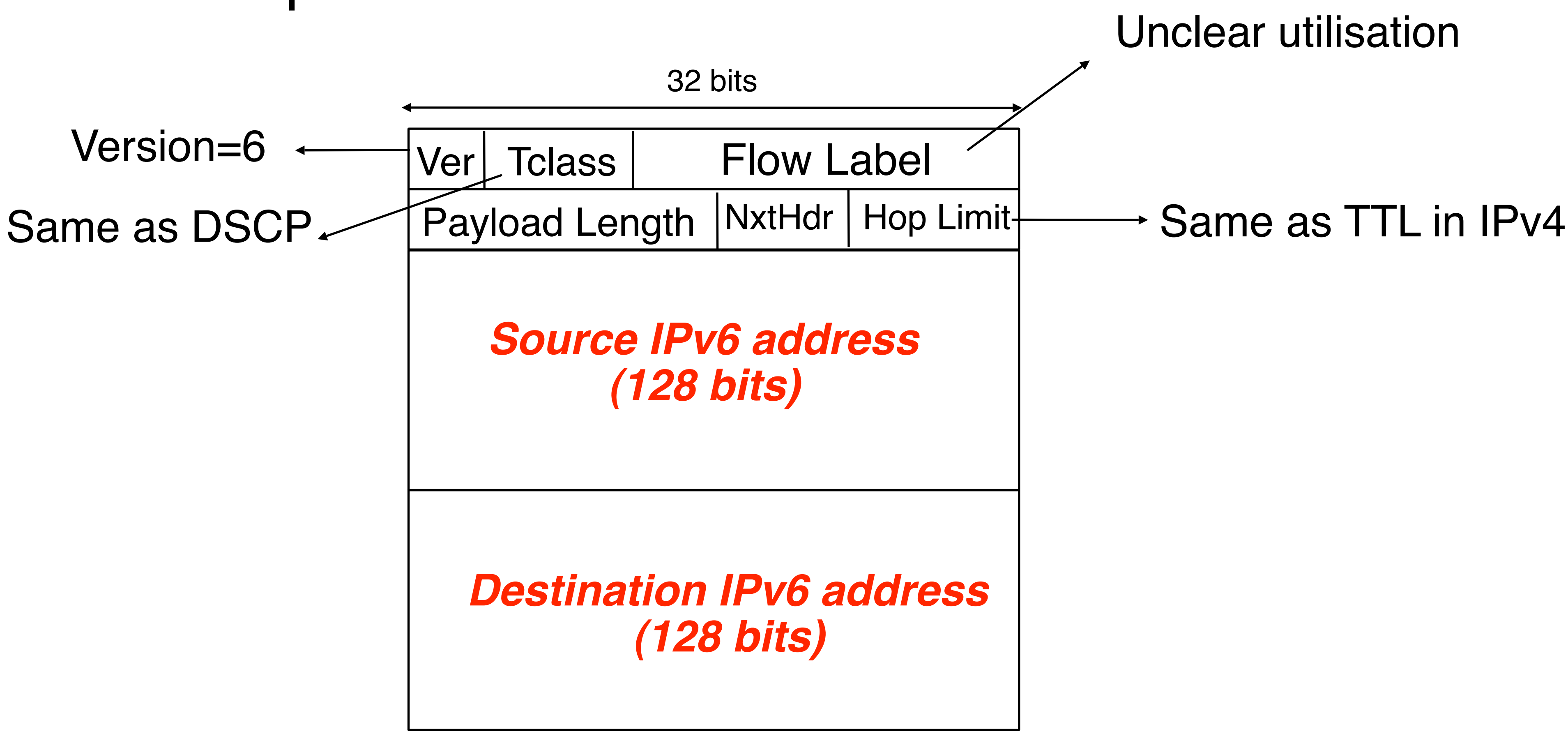
- Simplified packet format
  - Fields aligned on 32 bits boundaries to ease implementation



- No checksum in IPv6 header
  - rely on datalink and transport checksums

The IPv6 packet format

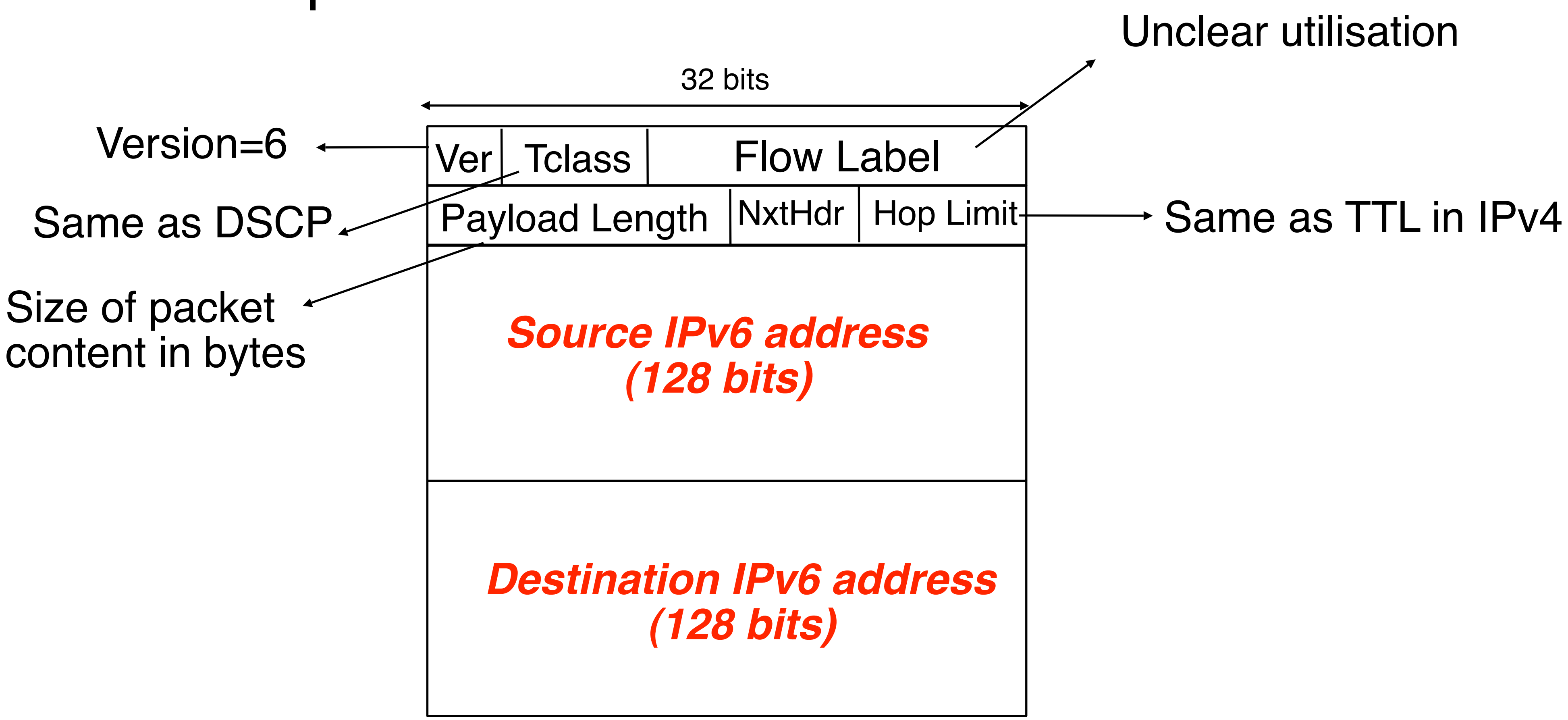
- Simplified packet format
  - Fields aligned on 32 bits boundaries to ease implementation



- No checksum in IPv6 header
  - rely on datalink and transport checksums

The IPv6 packet format

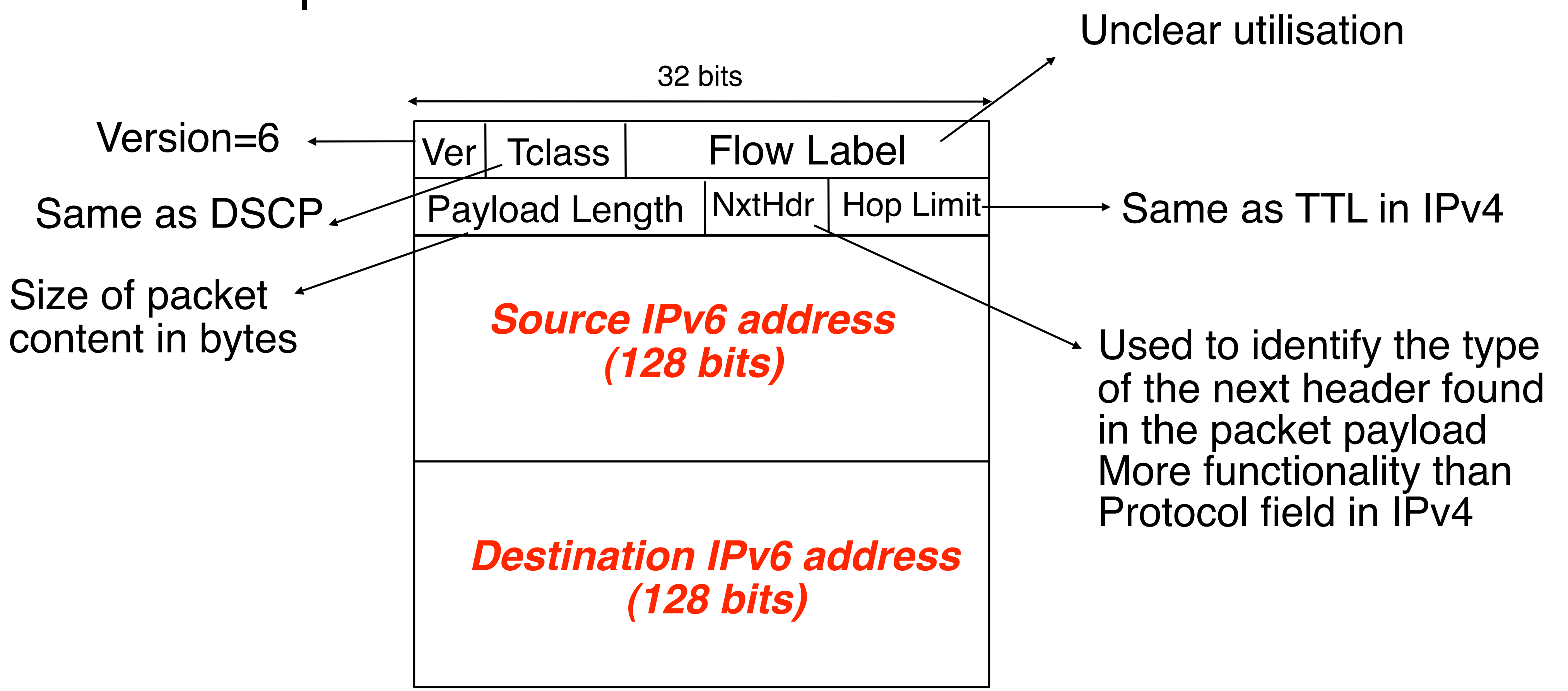
- Simplified packet format
  - Fields aligned on 32 bits boundaries to ease implementation



- No checksum in IPv6 header
  - rely on datalink and transport checksums

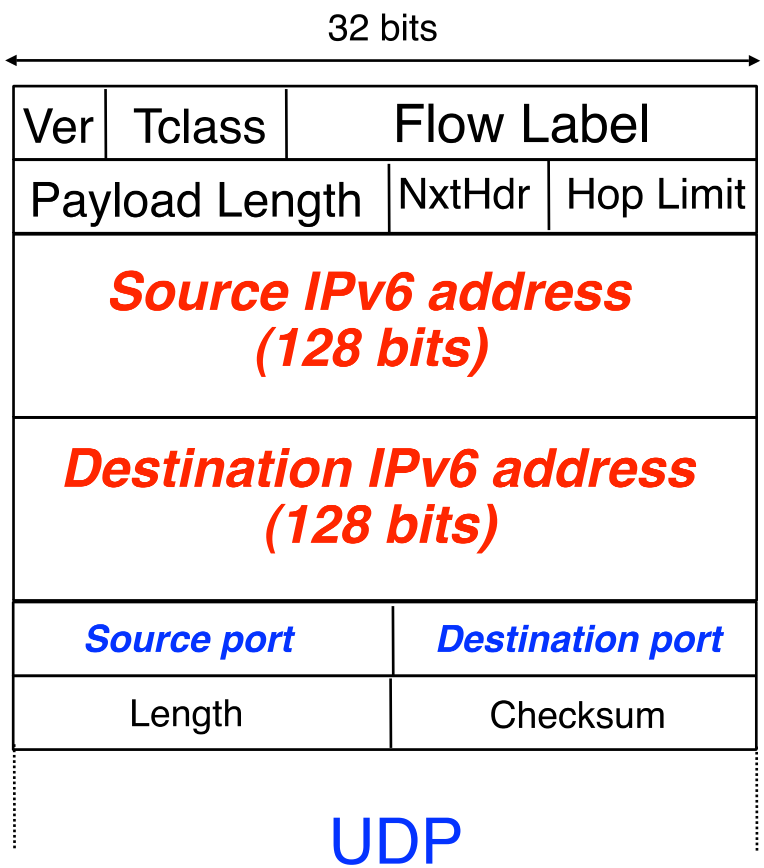
The IPv6 packet format

- Simplified packet format
  - Fields aligned on 32 bits boundaries to ease implementation

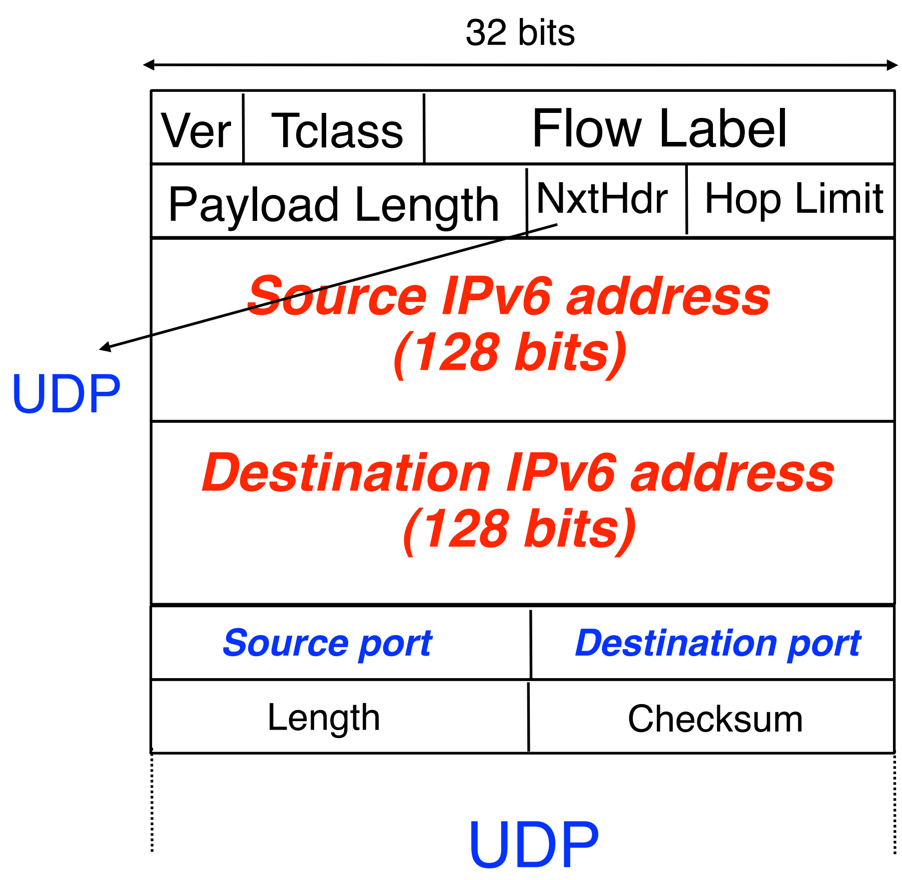


- No checksum in IPv6 header
  - rely on datalink and transport checksums

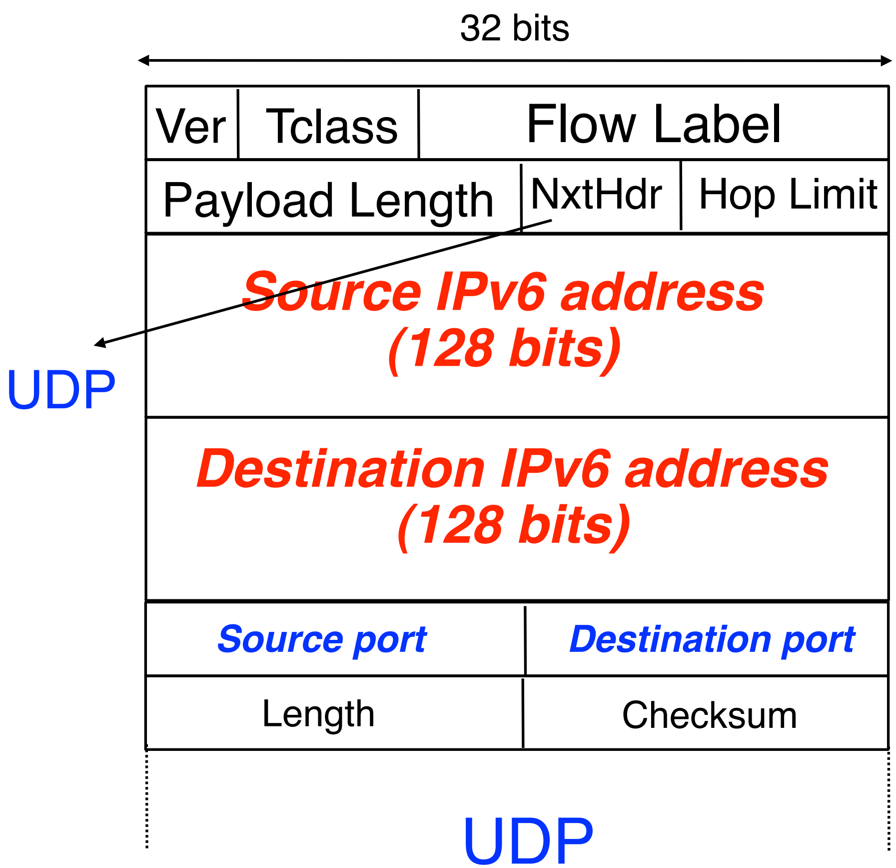
Sample IPv6 packets

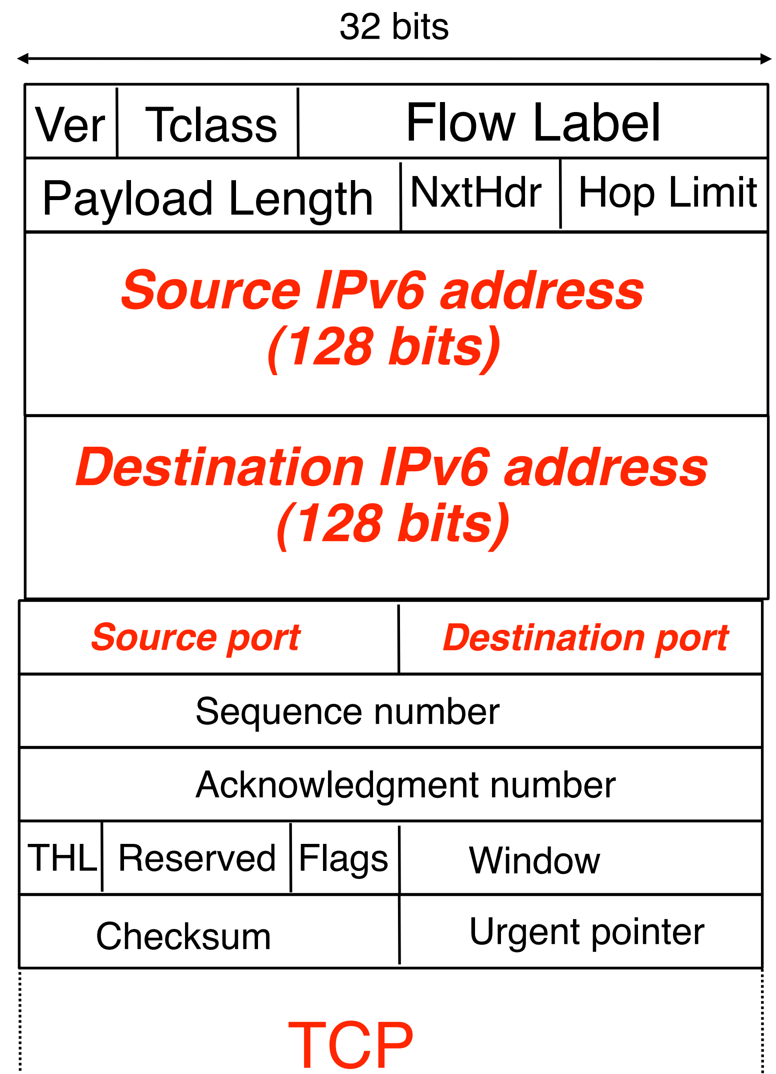


Sample IPv6 packets

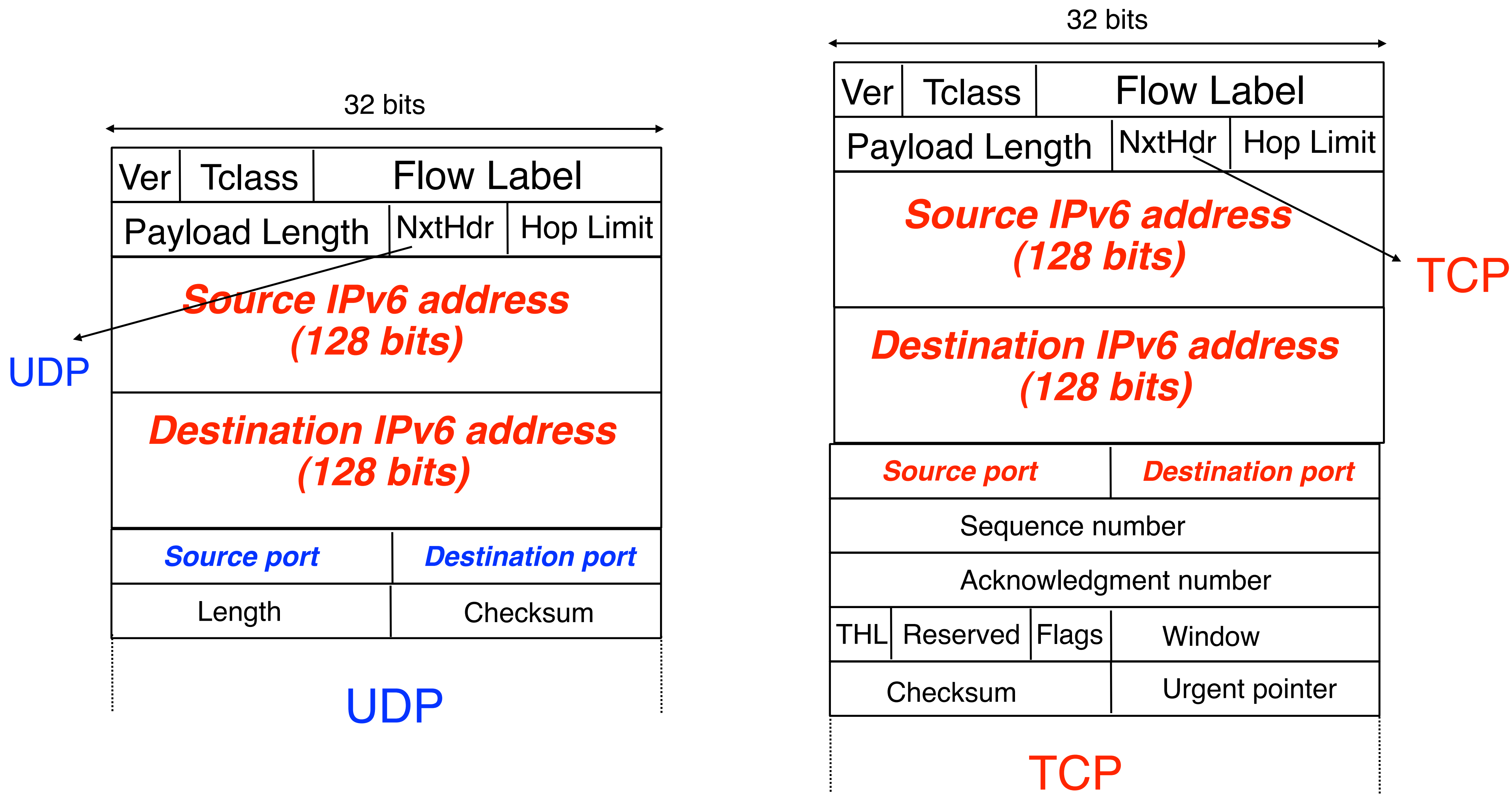


Sample IPv6 packets

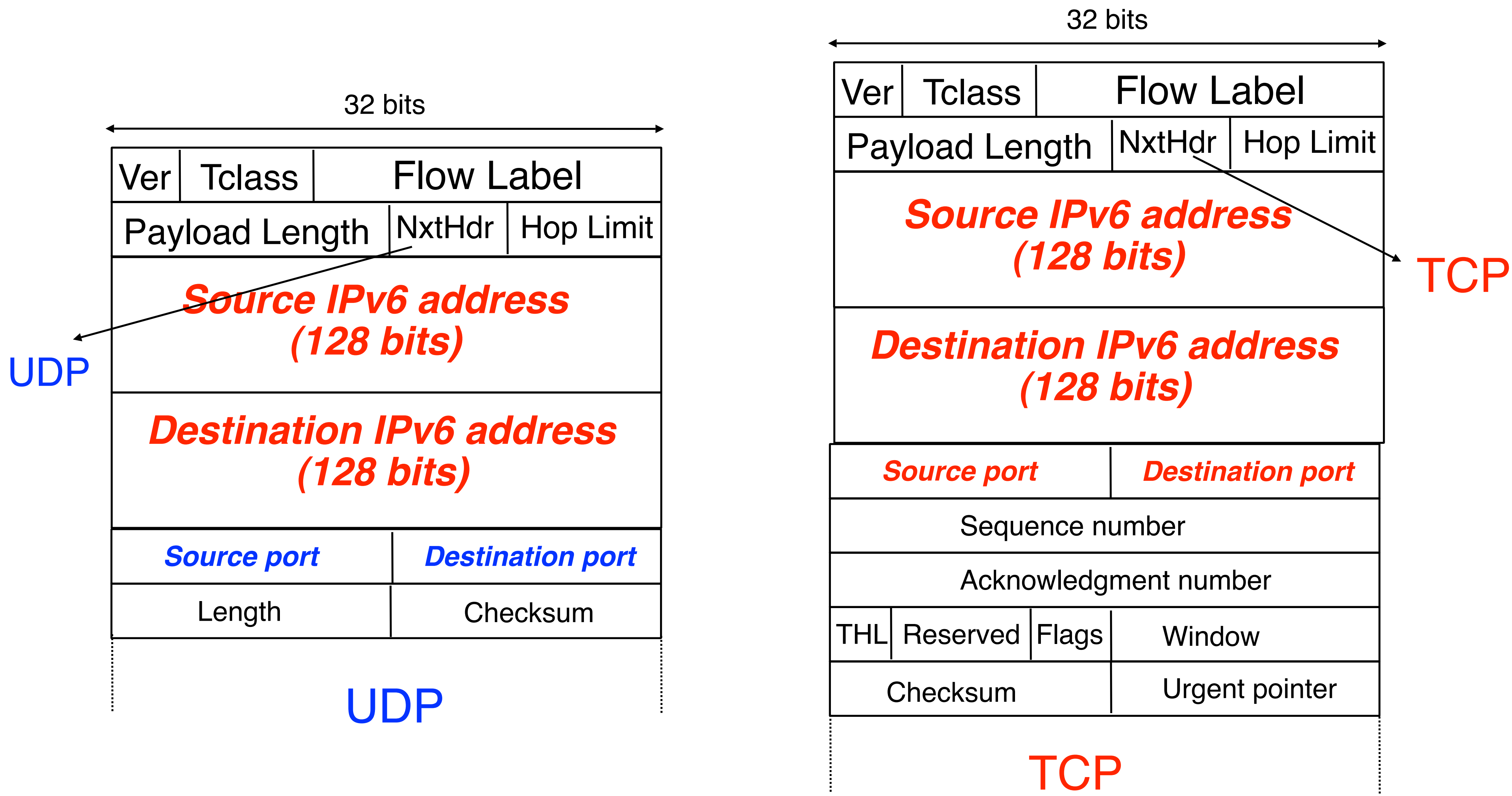




Sample IPv6 packets



Sample IPv6 packets



- Identification of a TCP connection
  - IPv6 source, IPv6 destination, Source and Destination ports

The IPv6 extension headers

The IPv6 extension headers

- Several types of extension headers
  - Hop-by-Hop Options
    - contains information to be processed by each hop
  - Routing (Type 0 and Type 2)
    - contains information affecting intermediate routers
  - Fragment
    - used for fragmentation and reassembly
  - Destination Options
    - contains options that are relevant for destination
  - Authentication
    - for IPSec
  - Encapsulating Security Payload
    - for IPSec
- Each header must be encoded as n*64 bits

Hop-by-hop and destination option headers

- TLV format of these options

NxtHdr	HLen	Type	Len
Data (var. length)			

Hop-by-hop and destination option headers

- TLV format of these options

NxtHdr	HLen	Type	Len
Data (var. length)			

- Two leftmost bits
  - How to deal with unknown option ?
    - 00 ignore and continue processing
    - 01 silently discard packet
    - 10 discard packet and send ICMP parameter problem back to source
    - 11 discard packet and send ICMP parameter problem to source if destination isn't multicast
  - Third bit
    - Can option content be changed en-route
  - Five rightmost bits
    - Type assigned by IANA

IPv6 jumbograms

- IPv6 packet format only supports 64 KBytes packets
  - packet size is encoded in 16 bits field
- on most hosts throughput increases with packet size

- Hop-by-hop jumbogram option
  - Increases packet size to 32 bits
    - when used, packet size in IPv6 header should be set to zero

NxtHdr	HLen	C2	Len:4
Packet size			

IPv6 jumbograms

- IPv6 packet format only supports 64 KBytes packets
  - packet size is encoded in 16 bits field
- on most hosts throughput increases with packet size

- Hop-by-hop jumbogram option
  - Increases packet size to 32 bits
    - when used, packet size in IPv6 header should be set to zero

NxtHdr	HLen	C2	Len:4
Packet size			

C2 : 11 0 00020
11 -> ICMP must be sent if option is unrecognised
0 -> content of option does not change en-route

Packet fragmentation

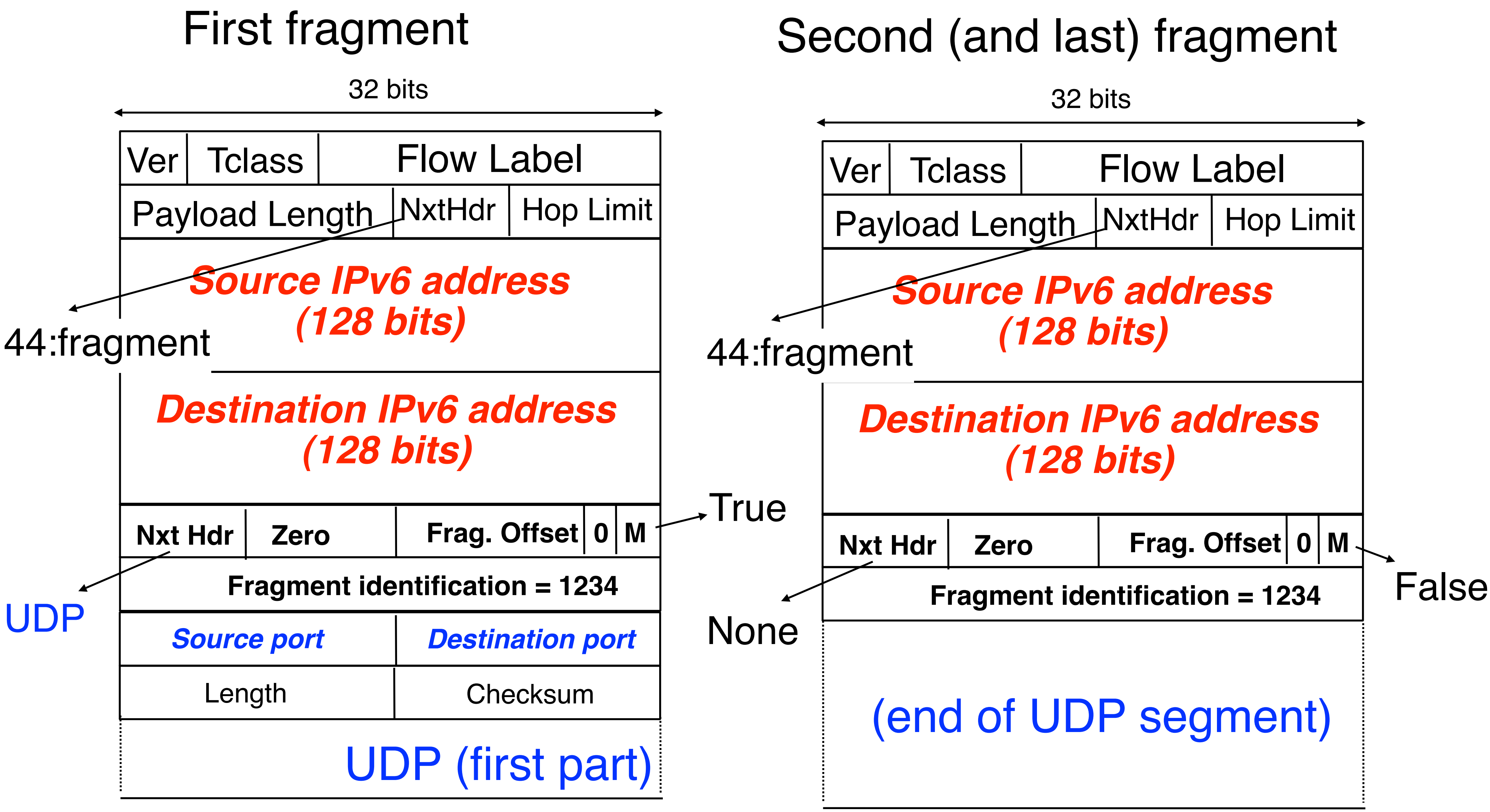
Packet fragmentation

- IPv4 used packet fragmentation on routers
  - All hosts must handle 576+ bytes packets
  - experience showed fragmentation is costly for routers and difficult to implement in hardware
  - PathMTU discovery is now widely implemented

Packet fragmentation

- IPv4 used packet fragmentation on routers
  - All hosts must handle 576+ bytes packets
  - experience showed fragmentation is costly for routers and difficult to implement in hardware
  - PathMTU discovery is now widely implemented
- IPv6
  - IPv6 requires that every link in the internet have an MTU of 1280 octets or more
    - otherwise link-specific fragmentation and reassembly must be provided at a layer below IPv6
  - Routers do not perform fragmentation
    - Only end hosts perform fragmentation and reassembly by using the fragmentation header
    - But PathMTU discovery should avoid fragmentation most of the time

A fragmented IPv6 packet



ICMPv6

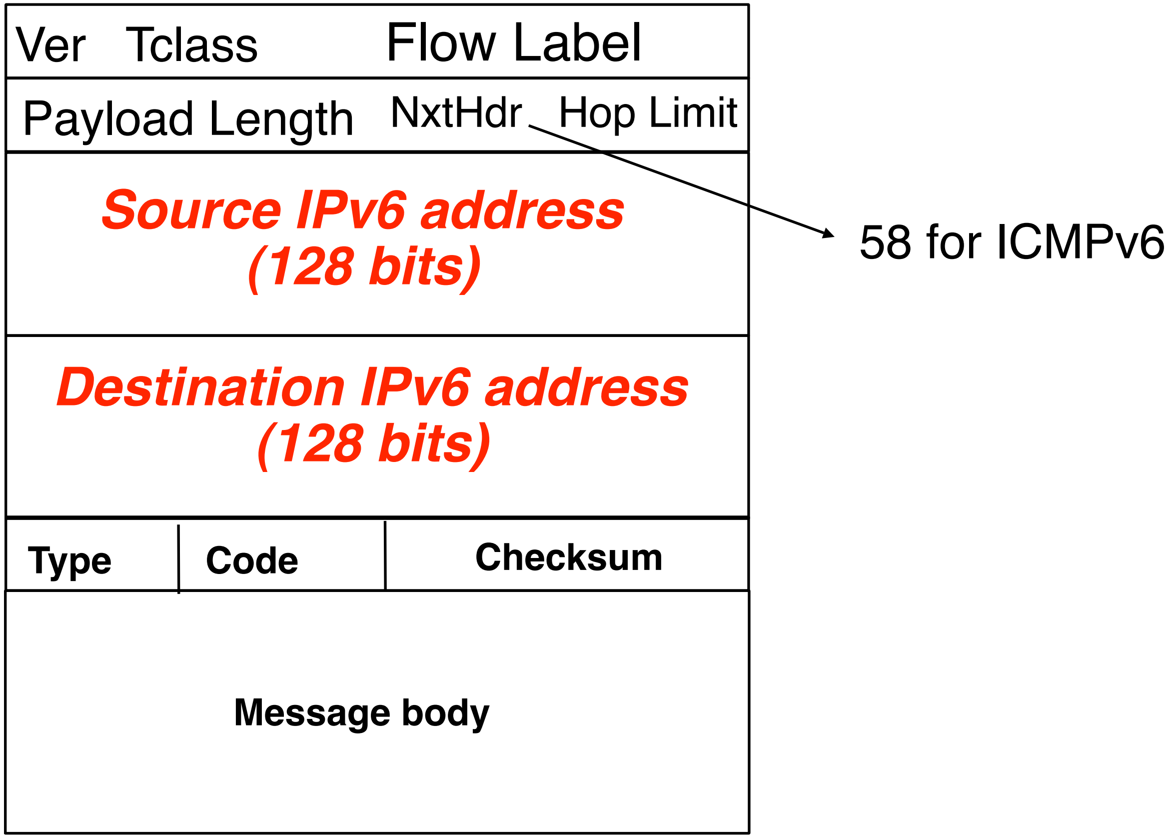
- Provides the same functions as ICMPv4, and more

- Types of ICMPv6 messages
  - Destination unreachable
  - Packet too big
    - Used for PathMTU discovery
  - Time expired (Hop limit exhausted)
    - Traceroute v6
  - Echo request and echo reply
    - Pingv6
  - Multicast group membership
  - Router advertisments
  - Neighbor discovery
  - Autoconfiguration

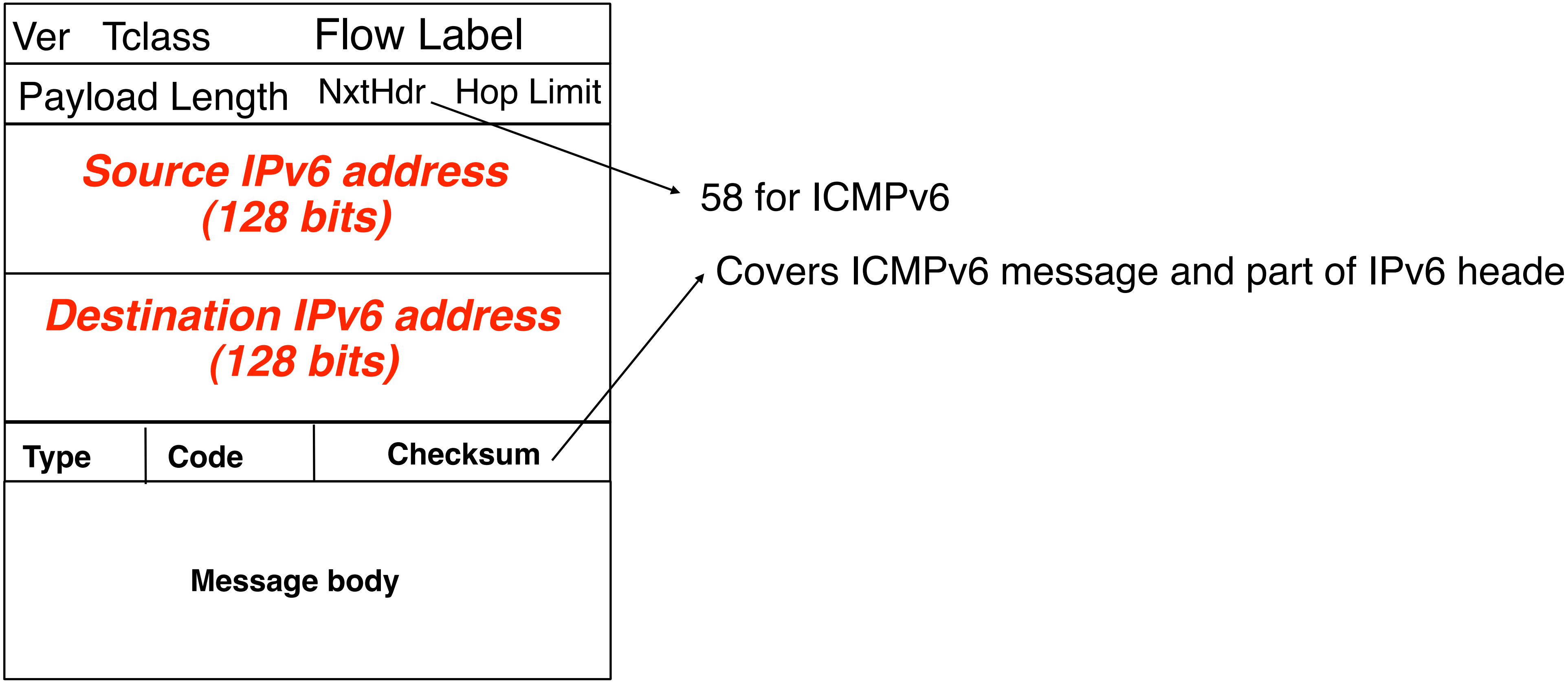
ICMPv6 packet format

Ver Tclass		Flow Label
Payload Length	NxtHdr	Hop Limit
Source IPv6 address (128 bits)		
Destination IPv6 address (128 bits)		
Type	**Code**	**Checksum**
Message body		

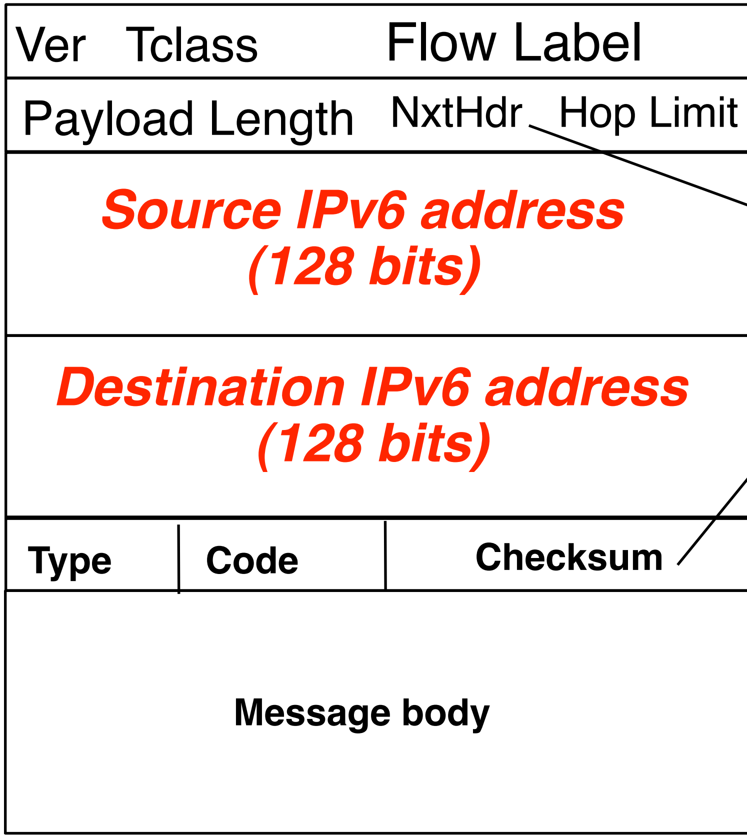
ICMPv6 packet format



ICMPv6 packet format



ICMPv6 packet format



58 for ICMPv6

Covers ICMPv6 message and part of IPv6 header

- Type
- ICMPv6 error messages (0<type<127)
  - 1 Destination Unreachable
  - 3 Time Exceeded
  - 2 Packet Too Big
  - 4 Parameter Problem
  - 100 Private experimentation
  - 101 Private experimentation
  - 127 Reserved for expansion
- ICMPv6 informational messages:
  - 128 Echo Request
  - 129 Echo Reply
  - 200 Private experimentation
  - 201 Private experimentation
  - 255 Reserved for expansion of ICMPv6 informational

Middleboxes

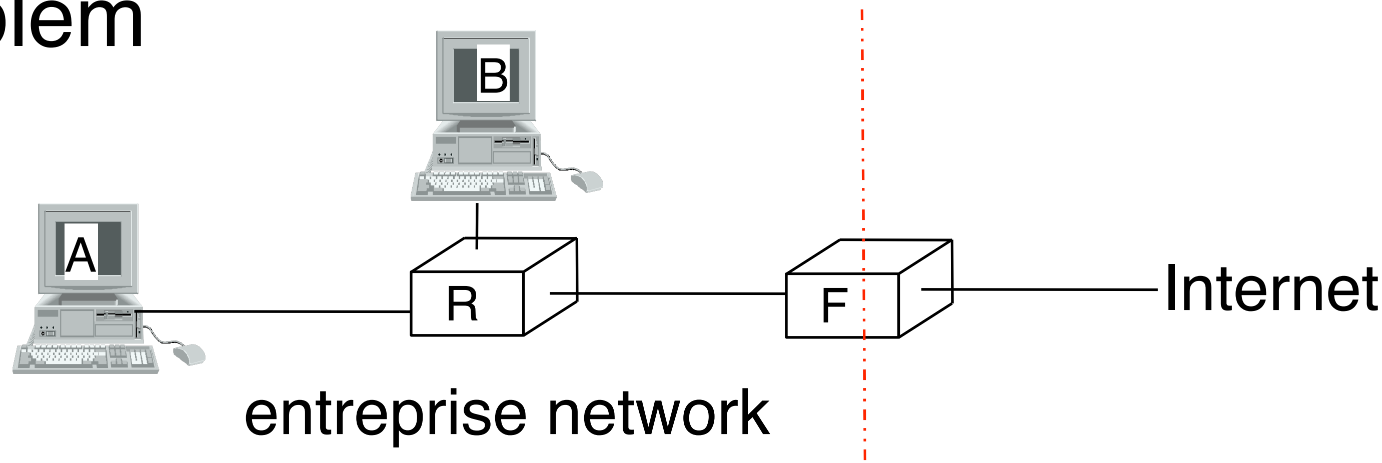
- The original TCP/IP architecture only defined hosts and routers

- Today's networks contain devices that
  - process
  - analyse
  - and possibly modify IP packets

- Examples
  - Firewall
  - Network Address Translator
  - Traffic shaper
  - Deep Packet Inspection
  - Intrusion Detection System
  - Load balancer

Firewall

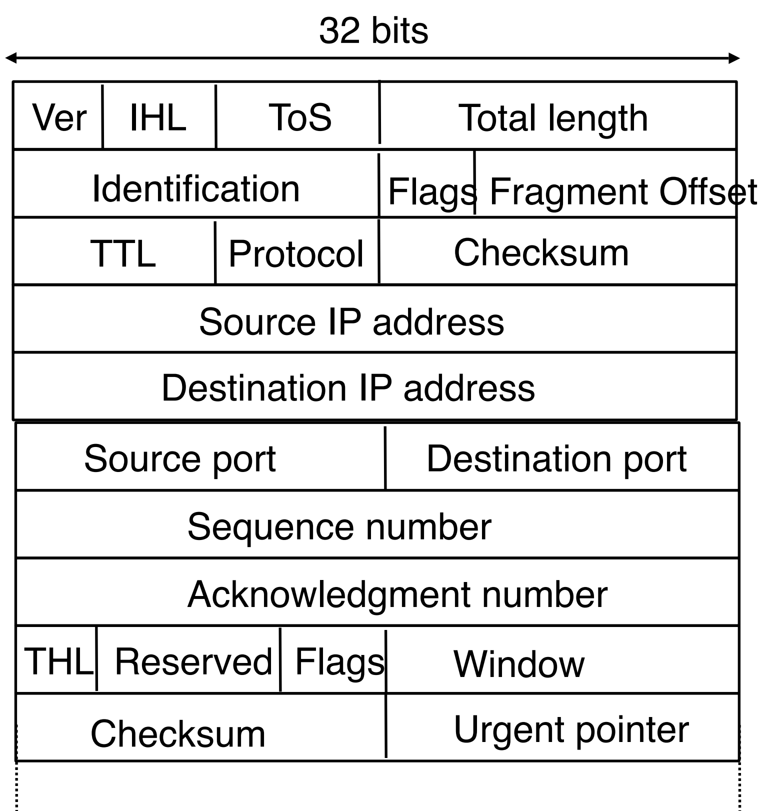
- Problem



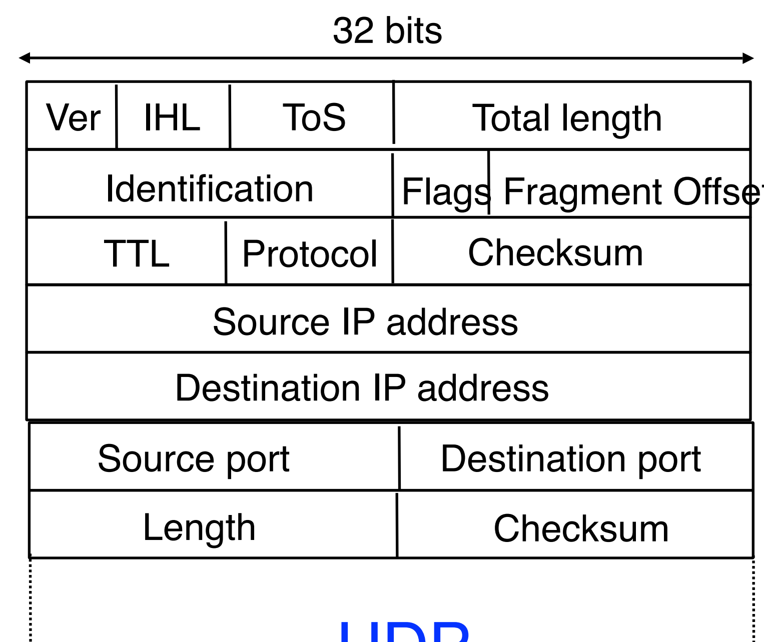
- How to control the packets entering the entreprise network ?
  - only allow external access to some servers
  - allow internal clients to use web
  - allow smtp (inbound and outbound) only on some dedicated servers
  - ...

Firewalls (2)

- Principle
  - Firewall analyses all packet headers
  - rules specify which packets should be accepted and rejected



TCP



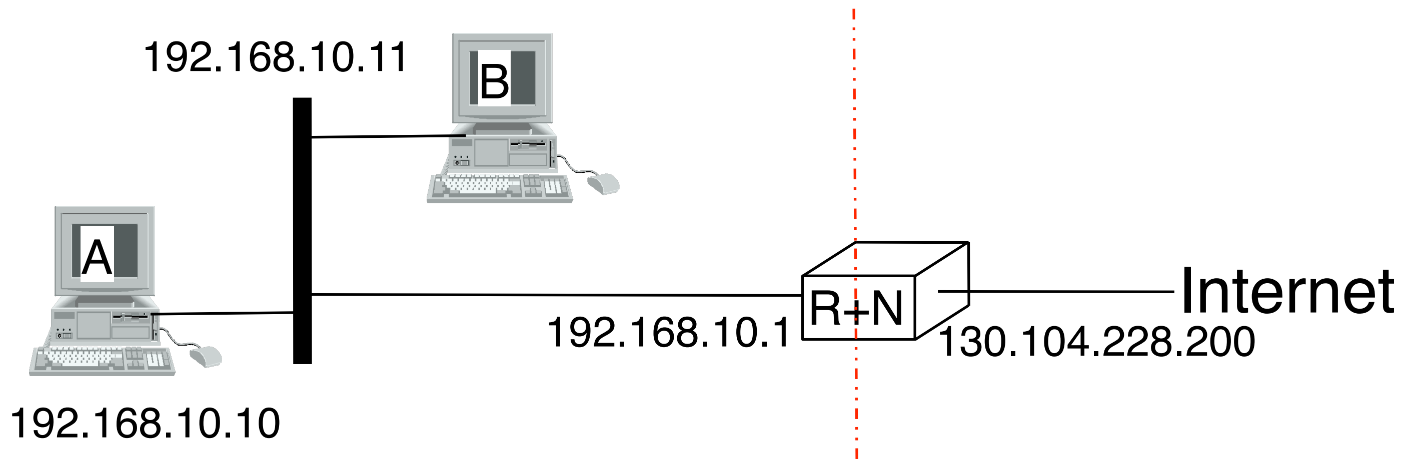
UDP

Firewalls : example

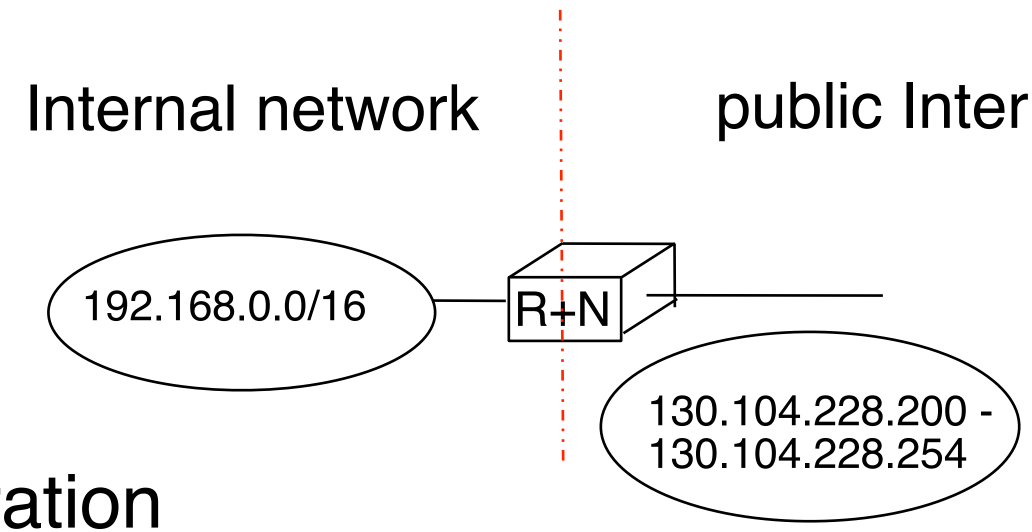
- Wifi UCLouvain
  - Outbound
    - Standard IPSec VPN...
    - SSH: TCP/22
    - HTTP: TCP/80, HTTPS: TCP/443
    - IMAP2+4: TCP/143, IMAP3: TCP/220, IMAPS: TCP/993, POP: TCP/110, POP3S: TCP/995, SMTPS: TCP/465, SMTP submit with STARTTLS: TCP/587
    - Passive (S)FTP: TCP/21
    - RDP: TCP/3389
    - IPv6 Tunnel Broker service: IP protocol 41
  - Inbound
    - OpenVPN 2.0: UDP/1194, IPsec NAT-Traversal UDP/4500, PPTP VPN: IP protocol 47 (GRE), Standard IPSec VPN: IP protocols 50 (ESP) and 51 (AH)
    - IPv6 Tunnel Broker service: IP protocol 41

Network Address Translator

- Problem
  - Limited number of public IPv4 addresses
- Solution
  - Use private addresses inside enterprise and home networks
  - Use one or a few public addresses
  - translate packets sent to public Internet



Simple enterprise NAT



- Operation
  - Mapping table

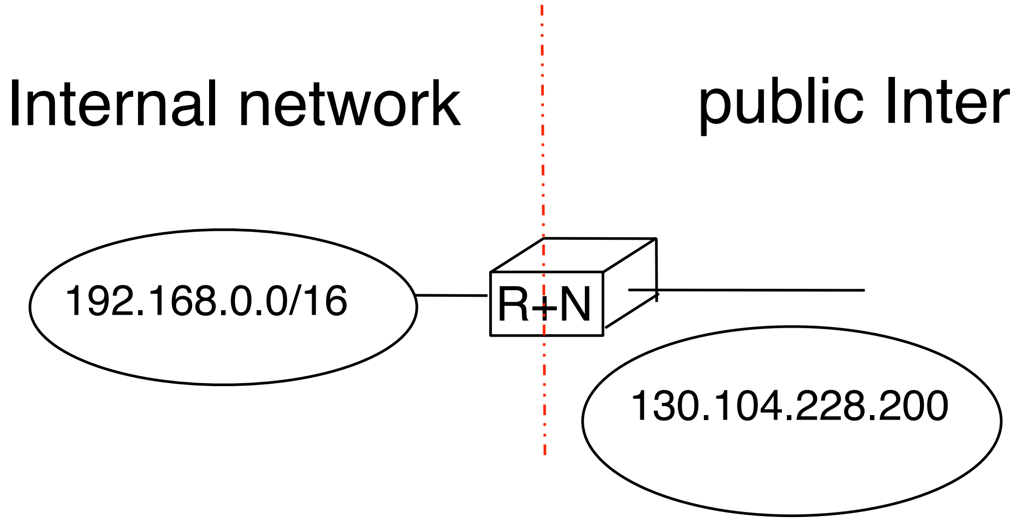
    Internal address <-> public address

  - Packet arrival from internal network

  - Packet arrival from public network

  - How long should mapping remain ?

Single address NAT



- Single address NAT
  - NAT translates IP addresses and TCP/UDP port numbers

Private address	Protocol	Port inside	public address	Port outside
192.168.10.10	UDP	2340	130.104.228.200	4567
192.168.10.10	TCP	512	130.104.228.200	520
192.168.10.11	TCP	1024	130.104.228.200	2048

Network layer

- Basics
- Routing
- IP : Internet Protocol
- Routing in IP networks
  - Internet routing organisation
  - Intradomain routing : RIP
  - Intradomain routing : OSPF
  - Interdomain routing : BGP

Internet organisation

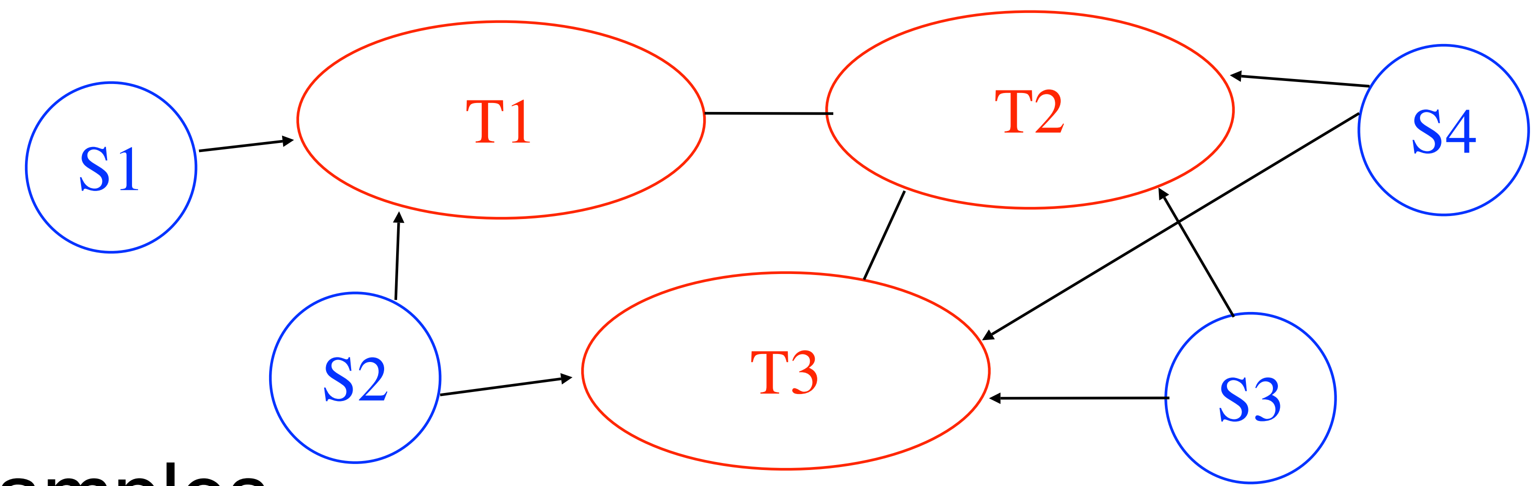
- Internet is an internetwork with a large number of Autonomous Systems (AS)
  - an AS is a set of routers that are managed by the same administrative entity
    - Examples : BELNET, UUNET, SKYNET, ...
    - about 20000 ASes in 2007
  - Autonomous Systems are interconnected to allow the transmission of IP packets from any source to any destination
    - On the Internet, most packets need to travel through several transit Autonomous Systems

Organisation of the Internet

- Internet is composed of about 30.000 autonomous routing domains
  - A domain is a set of routers, links, hosts and local area networks under the same administrative control
    - A domain can be very large...
      - AS568: SUMNET-AS DISO-UNRRA contains 73154560 IP addresses
    - A domain can be very small...
      - AS2111: IST-ATRIUM TE Experiment a single PC running Linux...
  - Domains are interconnected in various ways
    - The interconnection of all domains should in theory allow packets to be sent anywhere
    - Usually a packet will need to cross a few ASes to reach its destination

Types of domains

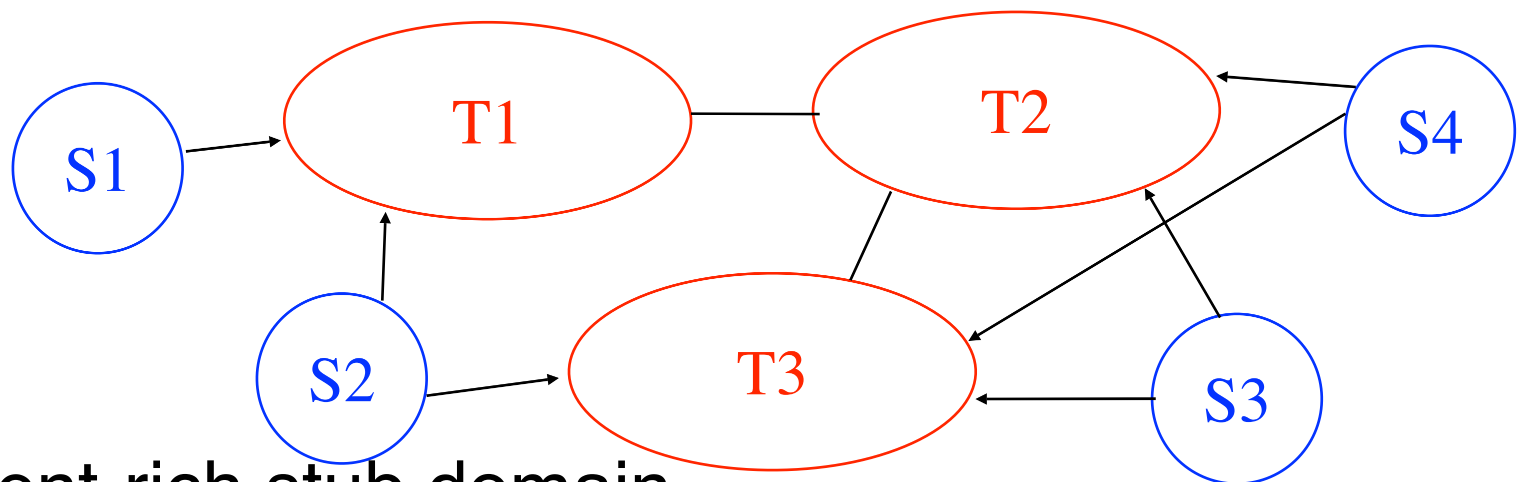
- Transit domain
  - A transit domain allows external domains to use its own infrastructure to send packets to other domains



- Examples
  - UUNet, OpenTransit, GEANT, Internet2, RENATER, EQUANT, BT, Telia, Level3,...

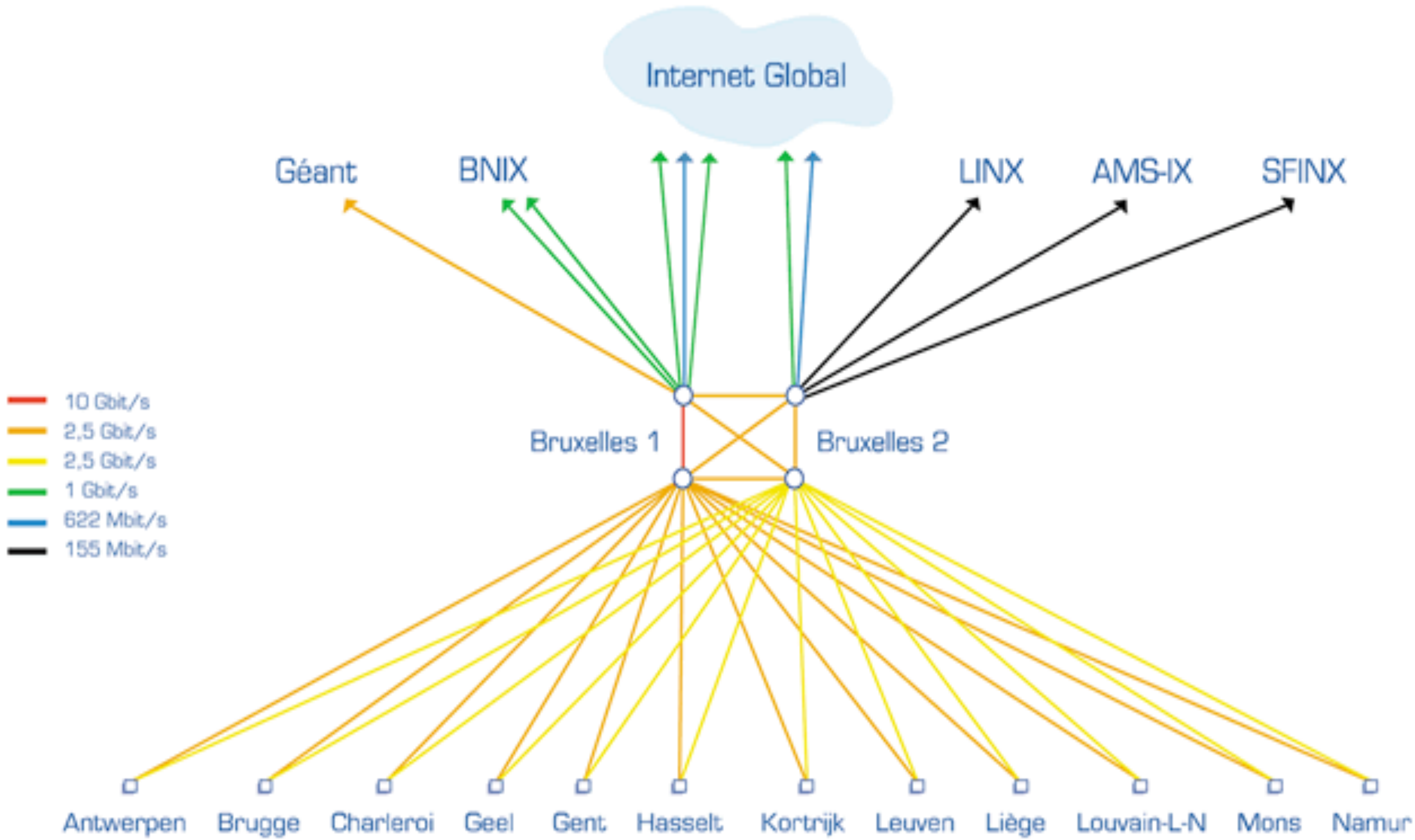
Types of domains (2)

- Stub domain
  - A stub domain does not allow external domains to use its infrastructure to send packets to other domains
    - A stub is connected to at least one transit domain
      - Single-homed stub : connected to one transit domain
      - Dual-homed stub : connected to two transit domains

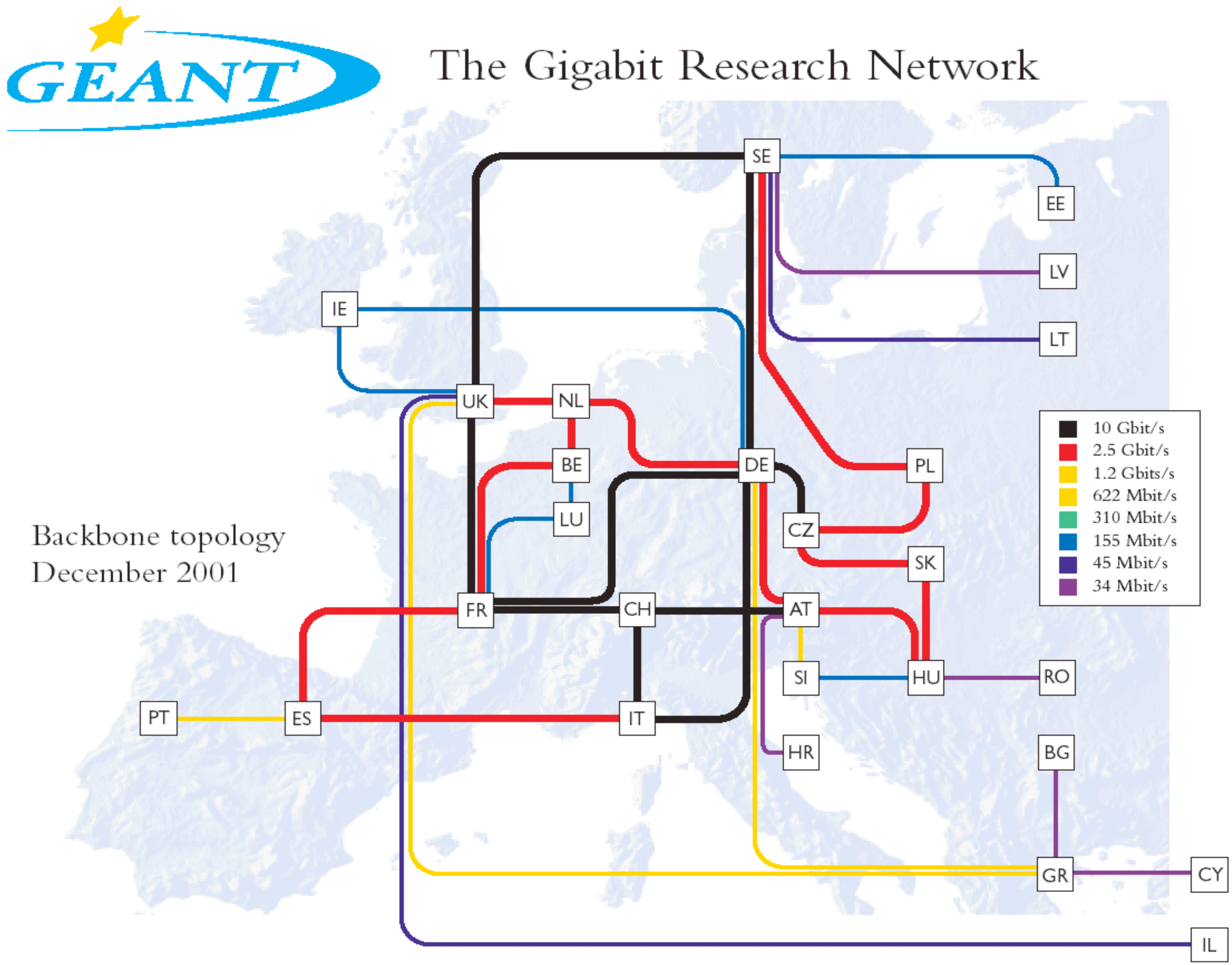


  - Content-rich stub domain
    - Large web servers : Yahoo, Google, MSN, TF1, BBC,...
  - Access-rich stub domain
    - ISPs providing Internet access via CATV, ADSL, ...

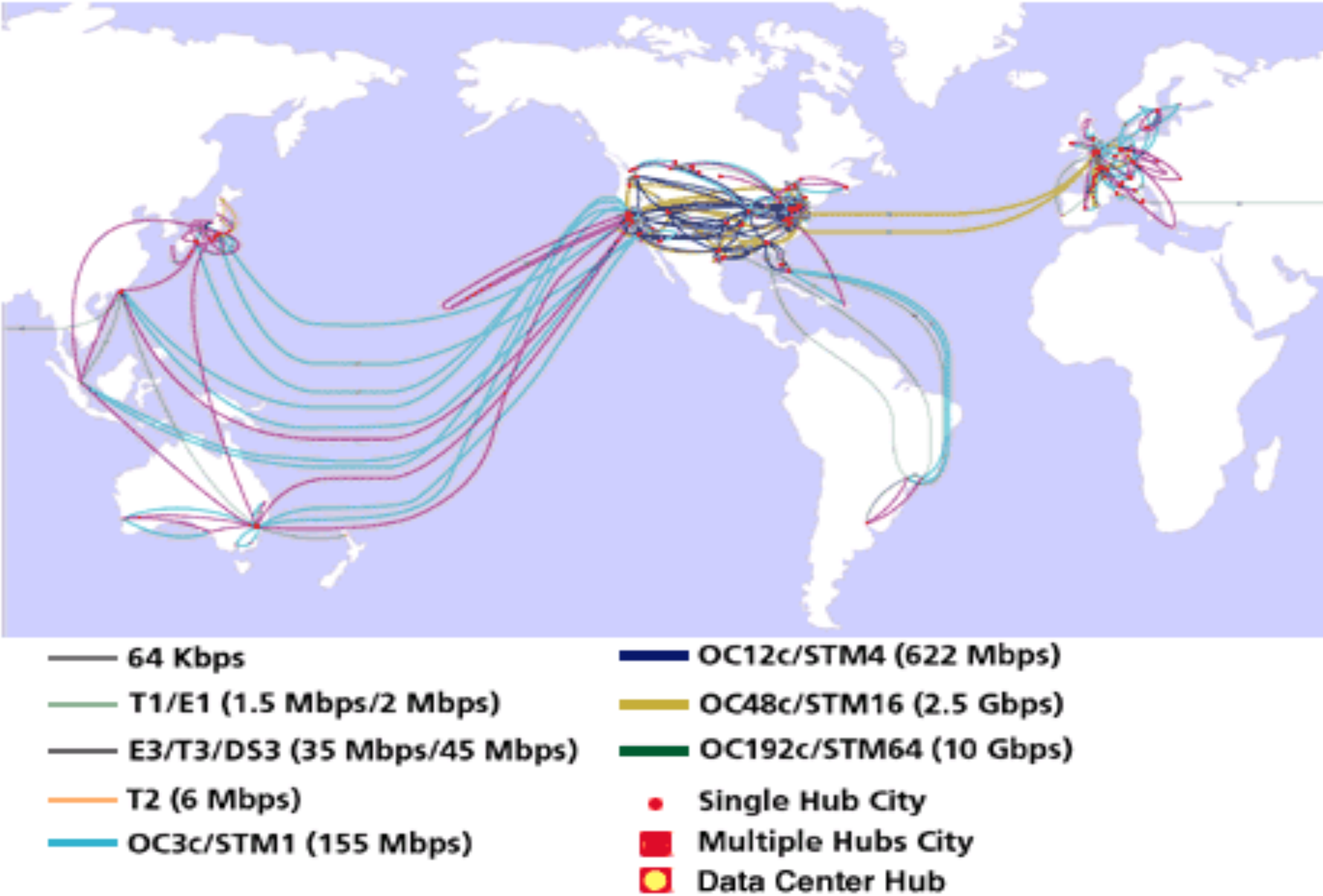
Sample network : Belnet



Sample network : GEANT

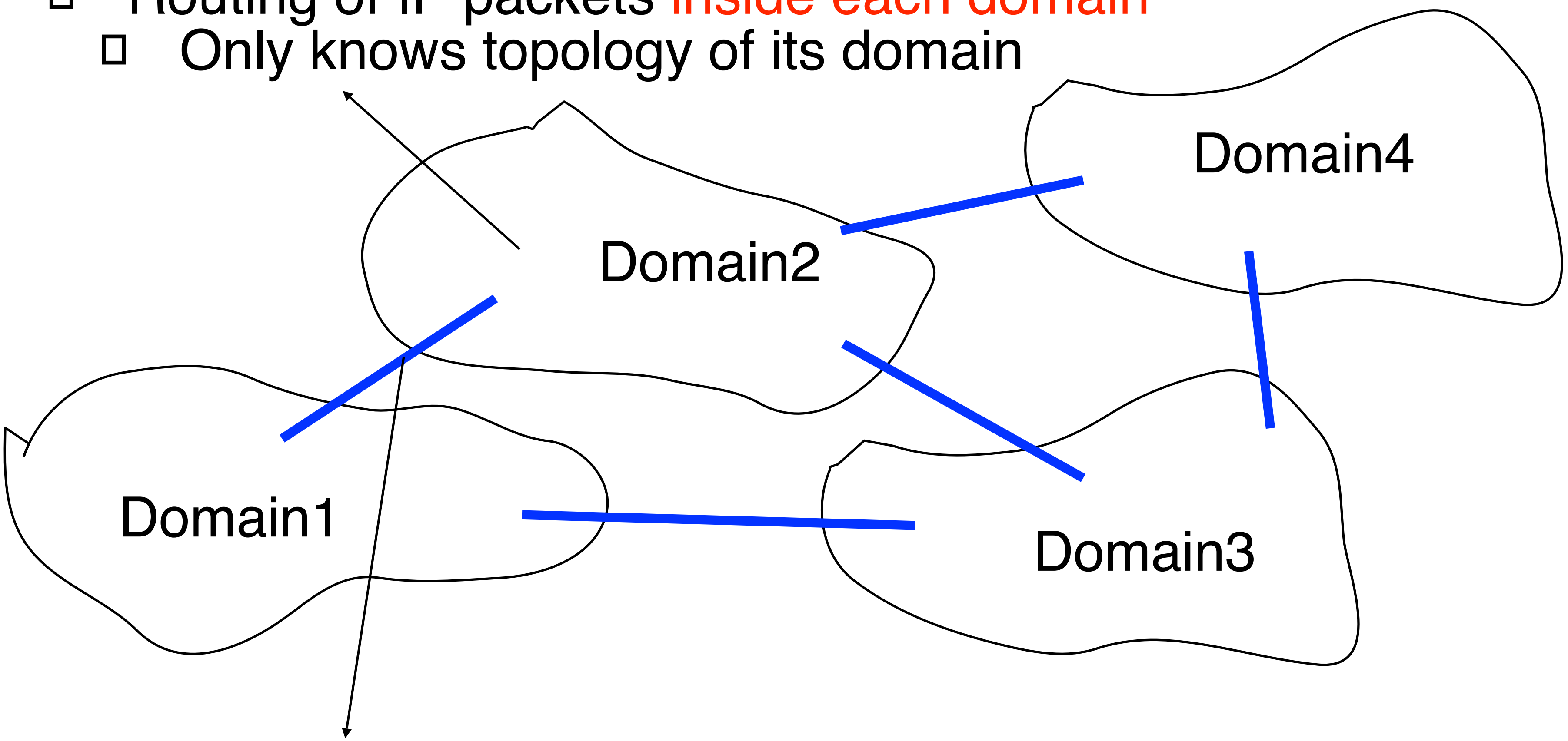


A large worldwide network : UUNet



Internet routing

- Interior Gateway Protocol (IGP)
  - Routing of IP packets inside each domain
    - Only knows topology of its domain



- Exterior Gateway Protocol (EGP)
  - Routing of IP packets between domains
    - Each domain is considered as a blackbox

Intradomain routing

- Goal
  - Allow routers to transmit IP packets along the best path towards their destination
    - best usually means the shortest path
      - Shortest measured in seconds or as number of hops
    - sometimes best means the less loaded path
  - Allow to find alternate routes in case of failures

- Behaviour
  - All routers exchange routing information
    - Each domain router can obtain routing information for the whole domain
    - The network operator or the routing protocol selects the cost of each link

Three types of Interior Gateway Protocols

- Static routing
  - Only useful in very small domains

- Distance vector routing
  - Routing Information Protocol (RIP)
    - Still widely used in small domains despite its limitations

- Link-state routing
  - Open Shortest Path First (OSPF)
    - Widely used in enterprise networks

  - Intermediate System- Intermediate-System (IS-IS)
    - Widely used by ISPs

Network layer

- Basics
- Routing
- IP : Internet Protocol
- Routing in IP networks
  - Internet routing organisation
  - Intradomain routing : RIP
  - Intradomain routing : OSPF
  - Interdomain routing : BGP

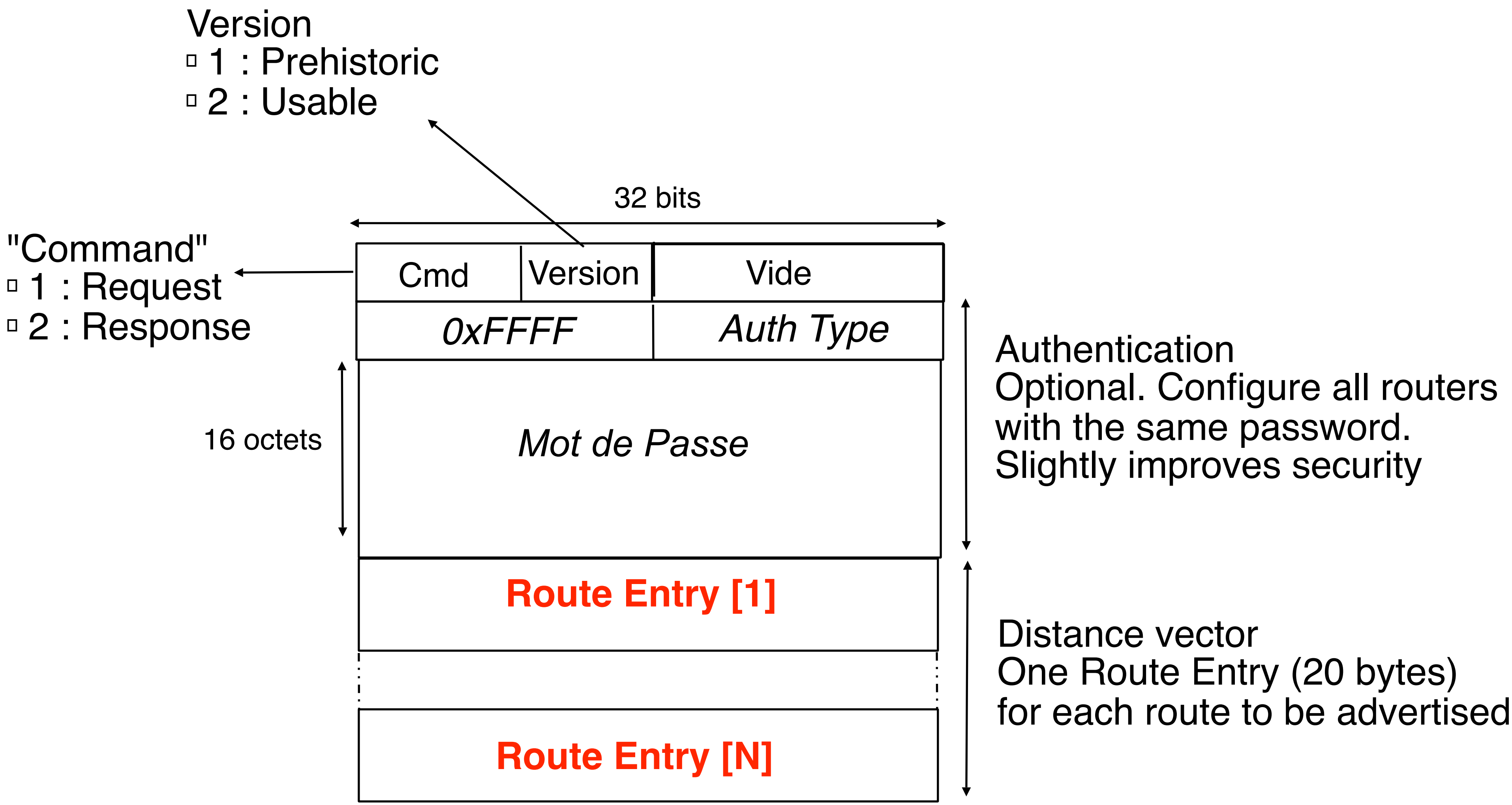
RIP
Routing Information Protocol

- Simple routing protocol that relies on distance vectors
  - Defined in RFC2453

- Principle
  - Each router periodically sends its distance vectors
    - default period : 30 seconds
    - distance vector is sent in UDP message with TTL=1 to all routers in local subnets (via IP multicast)

  - Optional extension : send a distance vector when the routing table changes
    - simple solution : send distance vector after each change
    - but some links flaps...
      - solution : send a distance vector if routing table changed and we did not send another vector within the last 5 seconds

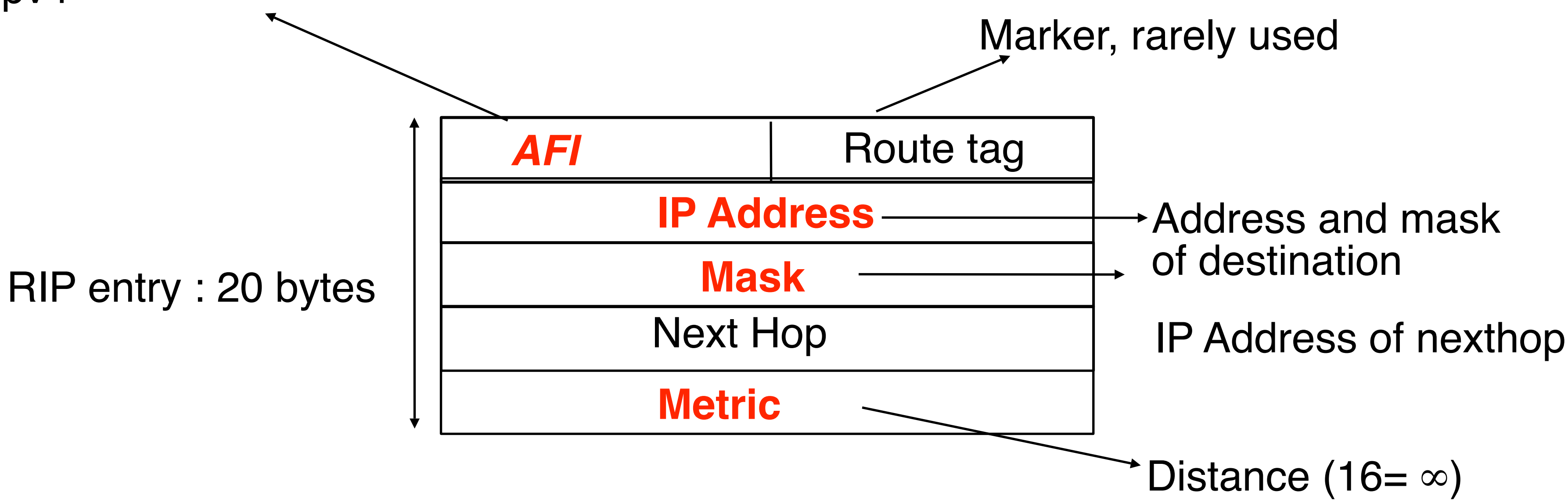
RIP : message format



- RIP messages are sent by UDP
  - port 520

RIP : Route Entries

AFI : Address Family Identifier
- type of addresses used
  - 2= Ipv4



- Default route
  - IP Address = 0.0.0.0, Mask = 0
- Each RIP message can contain up to 25 route entries (24 with authentication)
- If the routing table is larger than 25 entries, router will need to send several RIP messages

RIP timers

- Operation
  - At each expiration of its 30-sec timer, each router sends its distance vector and restarts its timer
- Problem
  - After a power failure, all routers might restart at same time and have synchronised RIP timers
    - Each router will need to process bursts of RIP messages
- Solution
  - Add some randomness to the timers
    - Restart timer after random[27.5, 32.5] instead of 30 seconds
      - commonly used technique to avoid synchronisation problems in distributed protocols

Network layer

- Basics
- Routing
- IP : Internet Protocol
- Routing in IP networks
  - Internet routing organisation
  - Intradomain routing : RIP
  - Intradomain routing : OSPF
  - Interdomain routing : BGP

OSPF

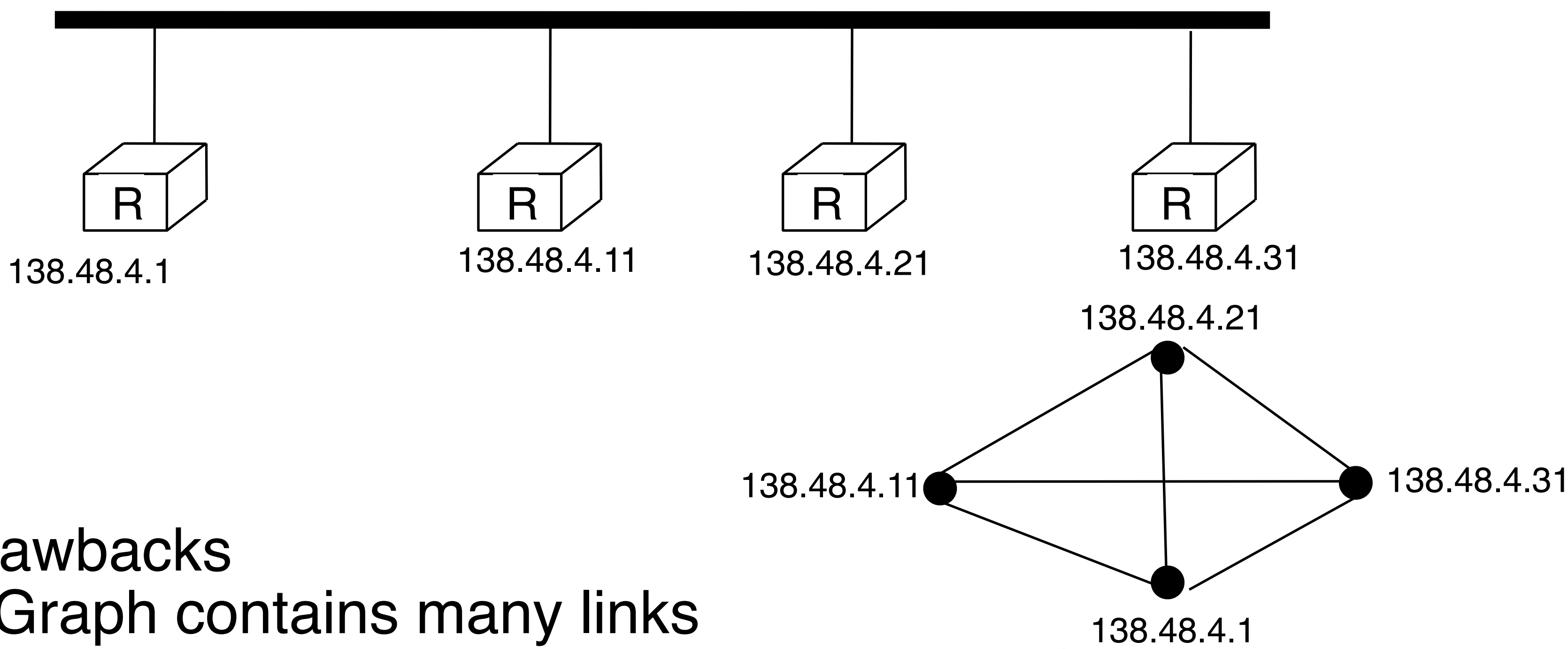
- Standardised link state routing protocol

- Operation
  - Router startup
    - HELLO packets to discover neighbours
  - Update of routing tables
    - Link state packets
      - acknowledgements, sequence numbers, age
    - periodic transmission
    - transmission upon link changes
  - Database description
    - provides the list of sequence numbers of all LSPs stored by router
  - Link state Request
    - used when a router boots to request link state packets from neighbours

OSPF is defined in RFC2328

OSPF details

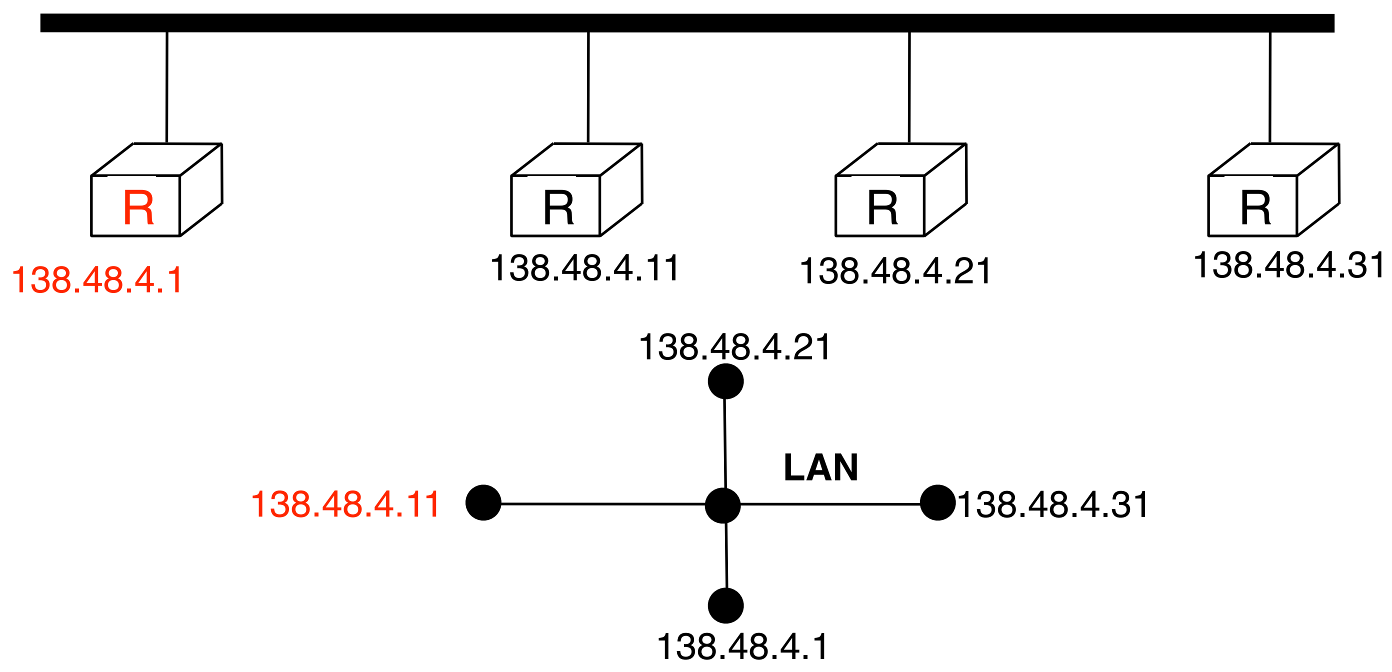
- Routers are often attached to LANs
  - How to describe a LAN full of routers as a graph



- Drawbacks
  - Graph contains many links
  - Routers need to exchange lots of HELLOs
  - Does not really describe the LAN
    - a failure of the LAN would cause a disconnection of all routers while the graph indicates a redundant topology

OSPF details (2)

- Solution
  - represent the LAN as a star with one router acting as the LAN
    - Designated router
      - One router is elected in the LAN to originate link state packets for the LAN
    - Adjacent router
      - Maintain adjacencies with the designated router



OSPF details (3)

OSPF details (3)

- OSPF in large networks
  - avoid too large routing tables in OSPF routers

OSPF details (3)

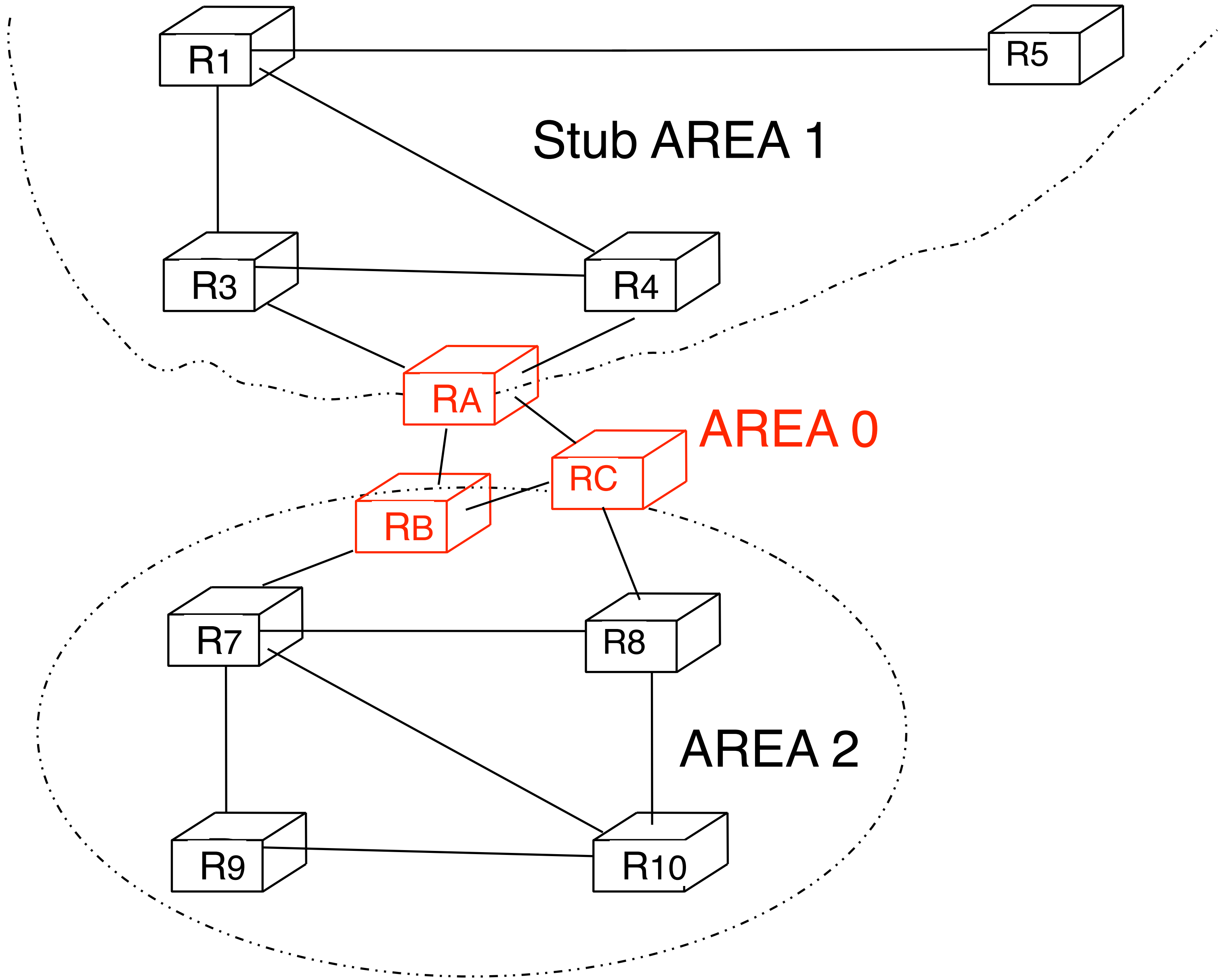
- OSPF in large networks
  - avoid too large routing tables in OSPF routers

- Solution
  - Divide network in areas
  - Backbone area : network backbone
    - all routers connected to two or more areas belong to the backbone area
  - All non-backbone areas must be attached to the backbone area
    - at least one router inside each area must be attached to the backbone

OSPF details (3)

- OSPF in large networks
  - avoid too large routing tables in OSPF routers
- Solution
  - Divide network in areas
  - Backbone area : network backbone
    - all routers connected to two or more areas belong to the backbone area
  - All non-backbone areas must be attached to the backbone area
    - at least one router inside each area must be attached to the backbone
- OSPF routing must allow any router to send packets to any other router

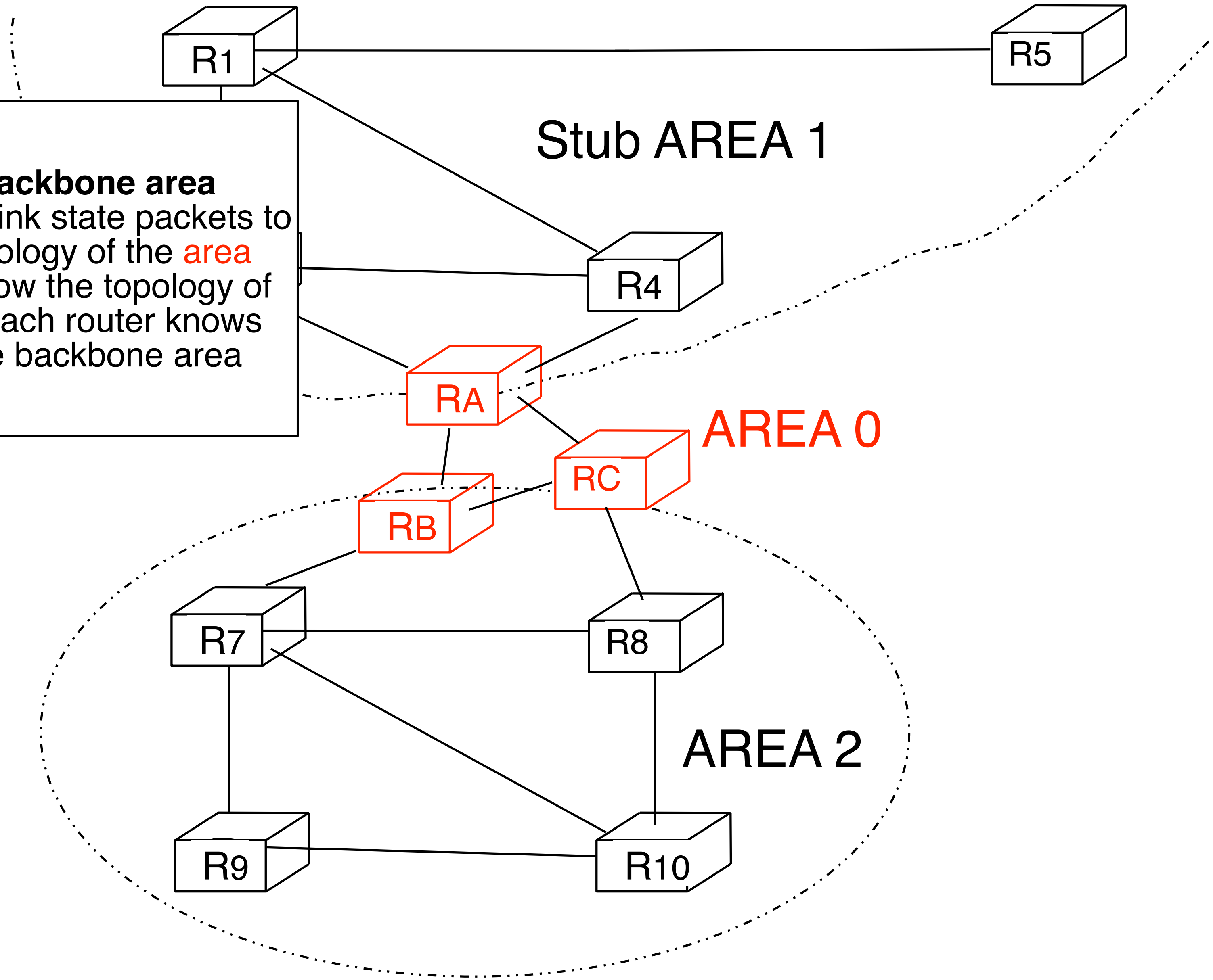
OSPF details (4)



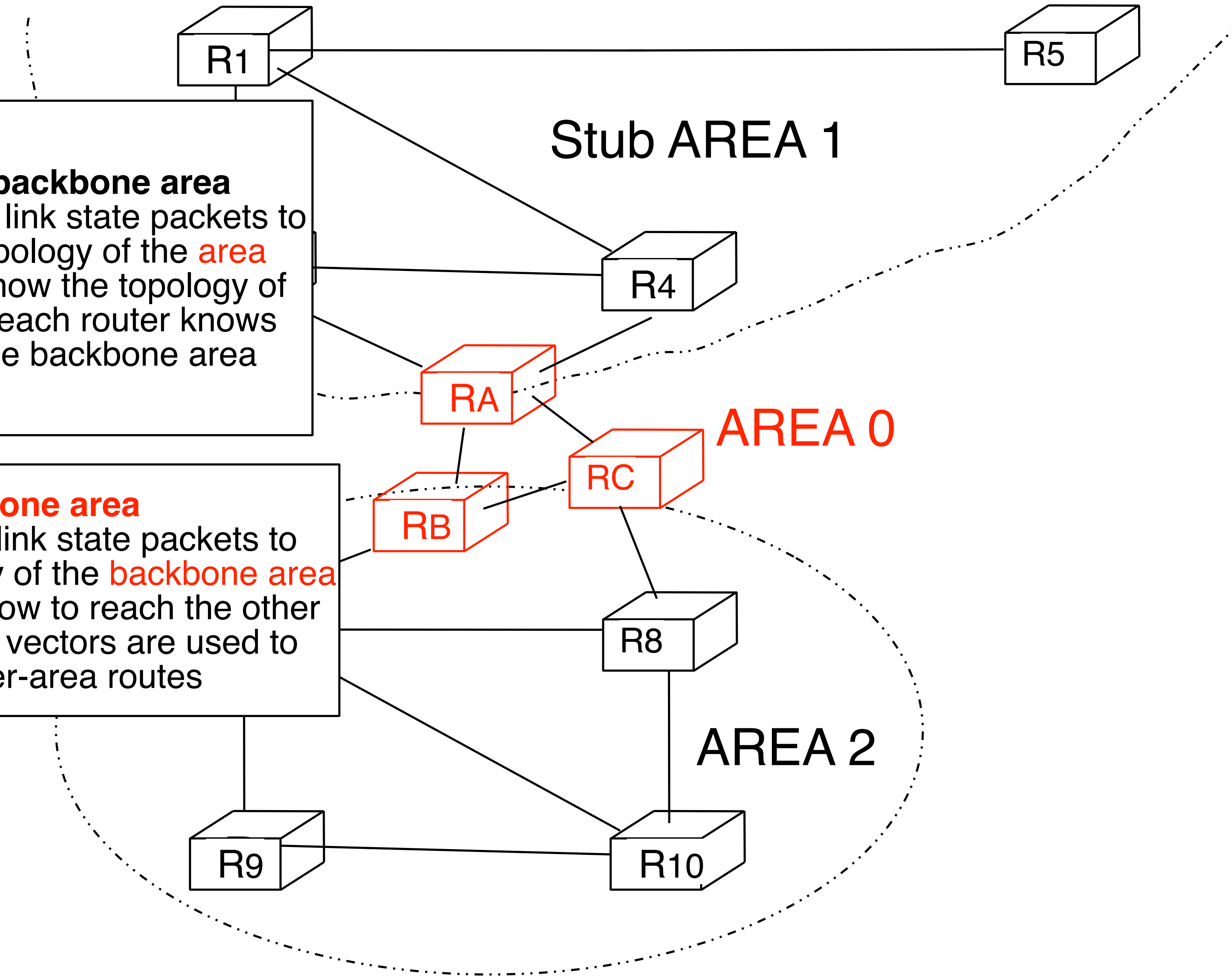
OSPF details (4)

Inside each non-backbone area

- Routers exchange link state packets to distribute the topology of the area
- Routers do not know the topology of other areas, but each router knows how to reach the backbone area



OSPF details (4)



Inside each non-backbone area

- Routers exchange link state packets to distribute the topology of the area
- Routers do not know the topology of other areas, but each router knows how to reach the backbone area

Inside backbone area

- Routers exchange link state packets to distribute the topology of the backbone area
- Each router knows how to reach the other areas and distance vectors are used to distribute inter-area routes

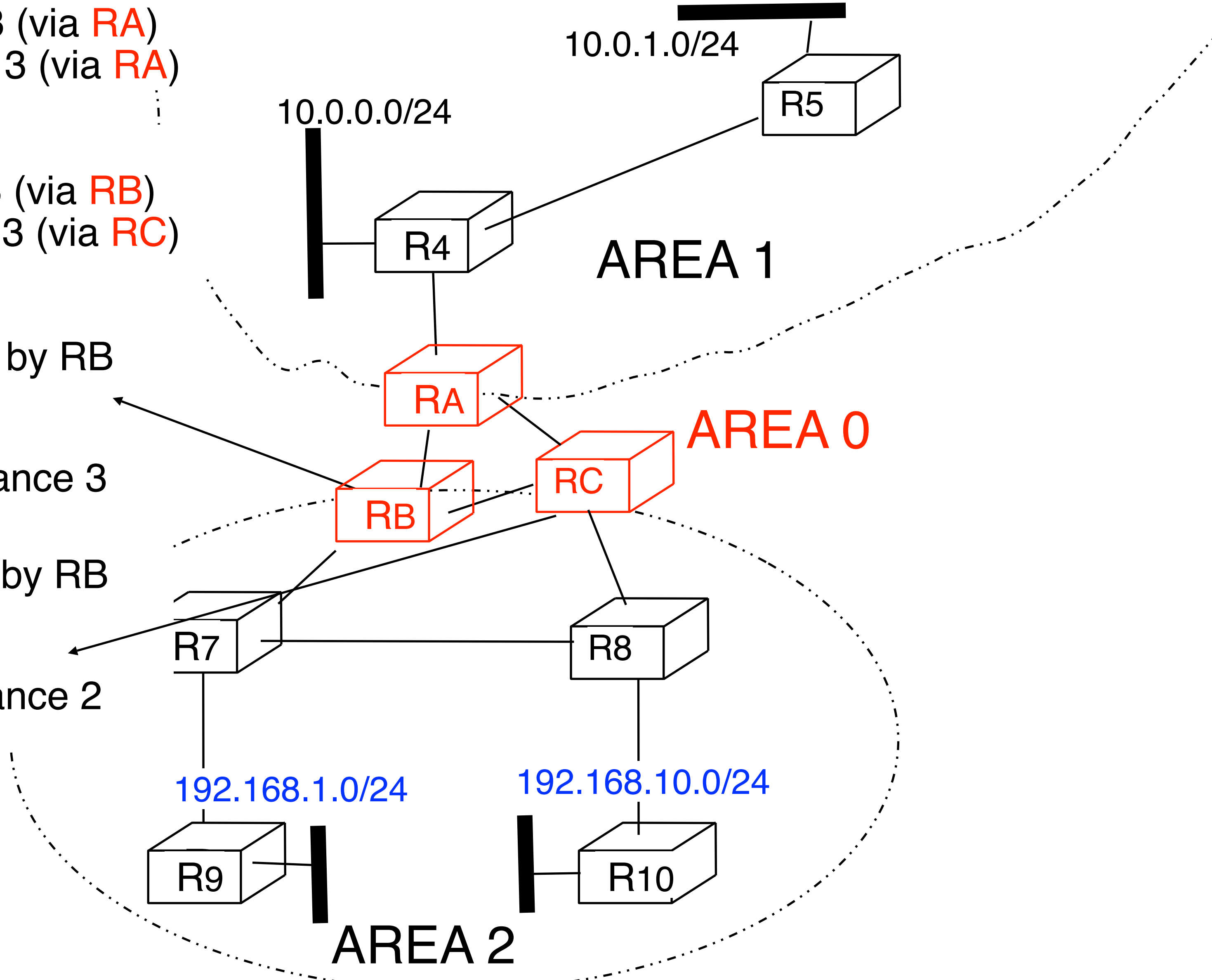
OSPF areas : Example

Routes learned by R4
- 192.168.1.0/24, distance 3 (via RA)
- 192.168.10.0/24, distance 3 (via RA)

Routes chosen par RA
- 192.168.1.0/24, distance 3 (via RB)
- 192.168.10.0/24, distance 3 (via RC)

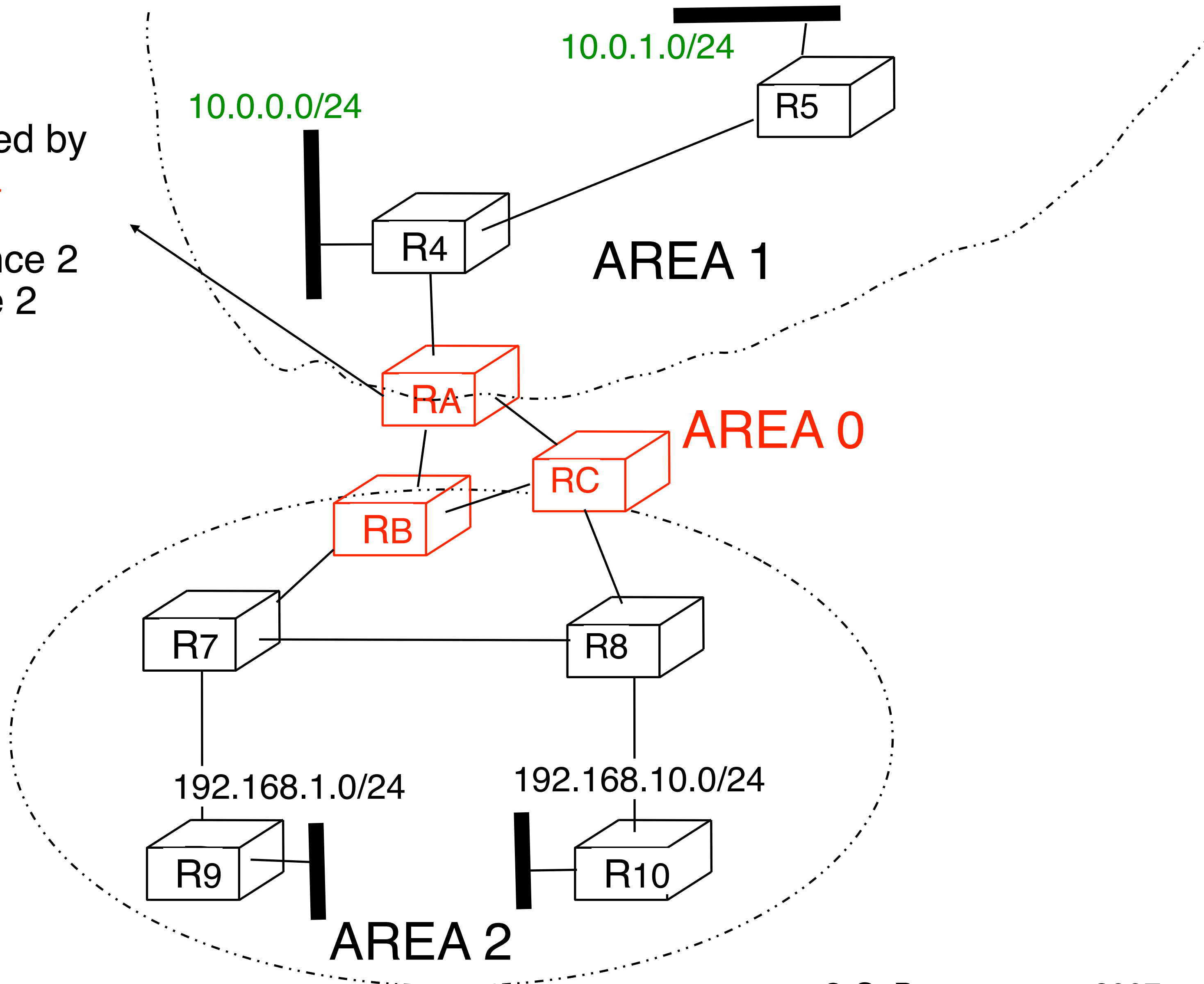
Distance vectors advertised by RB in backbone area
- 192.168.1.0/24, distance 2 and 192.168.10.0/24, distance 3

Distance vectors advertised by RB in backbone area
- 192.168.1.0/24, distance 3 and 192.168.10.0/24, distance 2



Areas OSPF : Example (2)

Distance vector advertised by RA in the backbone area
10.0.0.0/24, distance 1
and 10.0.1.0/24, distance 2
▫ or 10.0.0.0/23, distance 2



Network layer

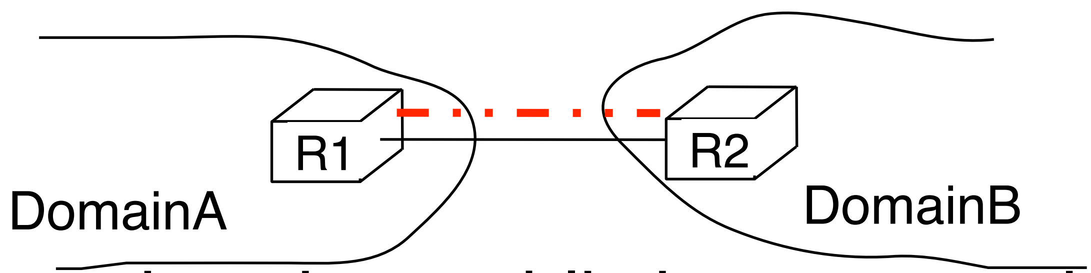
- Basics
- Routing
- IP : Internet Protocol
- Routing in IP networks
  - Internet routing organisation
  - Intradomain routing : RIP
  - Intradomain routing : OSPF
  - Interdomain routing : BGP

Interdomain routing

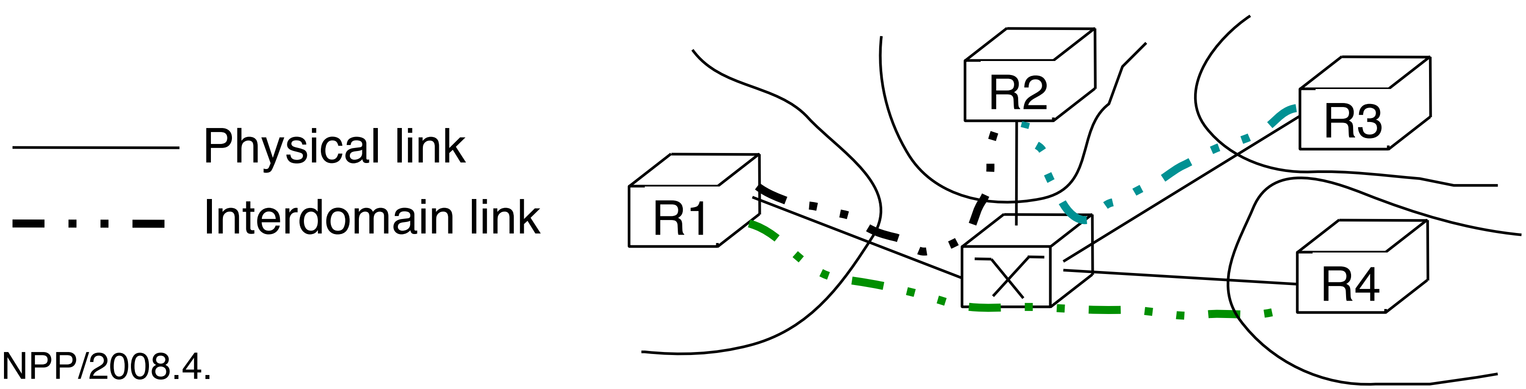
- Goals
  - Allow to transmit IP packets along the best path towards their destination through several transit domains while taking into account the routing policies of each domain without knowing the detailed topology of those domains
    - From an interdomain viewpoint, best path often means *cheapest* path
    - Each domain is free to specify inside its routing policy the domains for which it agrees to provide a transit service and the method it uses to select the best path to reach each destination

Types of interdomain links

- Two types of interdomain links
  - Private link
    - Usually a leased line between two routers belonging to the two connected domains



  - Connection via a public interconnection point
    - Usually Gigabit or higher Ethernet switch that interconnects routers belonging to different domains



Routing policies

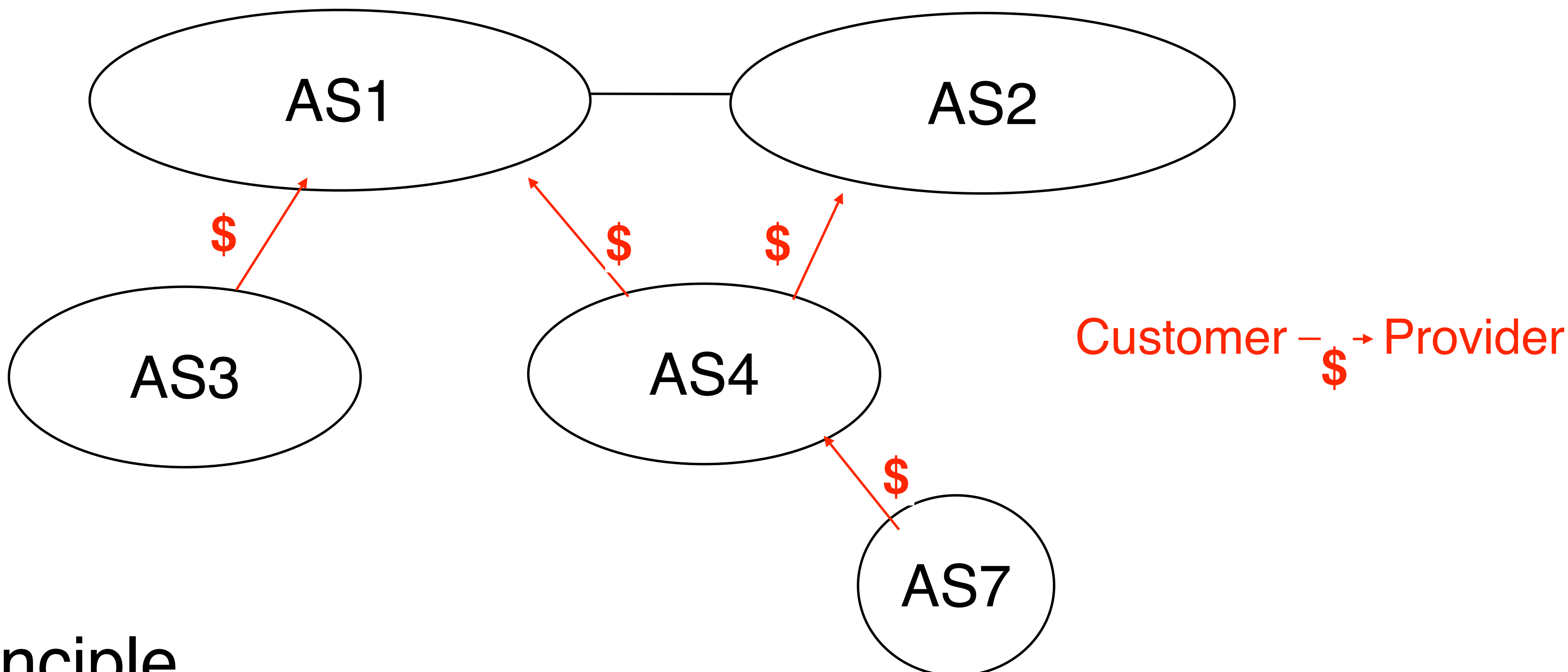
- In theory BGP allows each domain to define its own routing policy...

- In practice there are two common policies

  - customer-provider peering
    - **Customer c** buys Internet connectivity from **provider P**

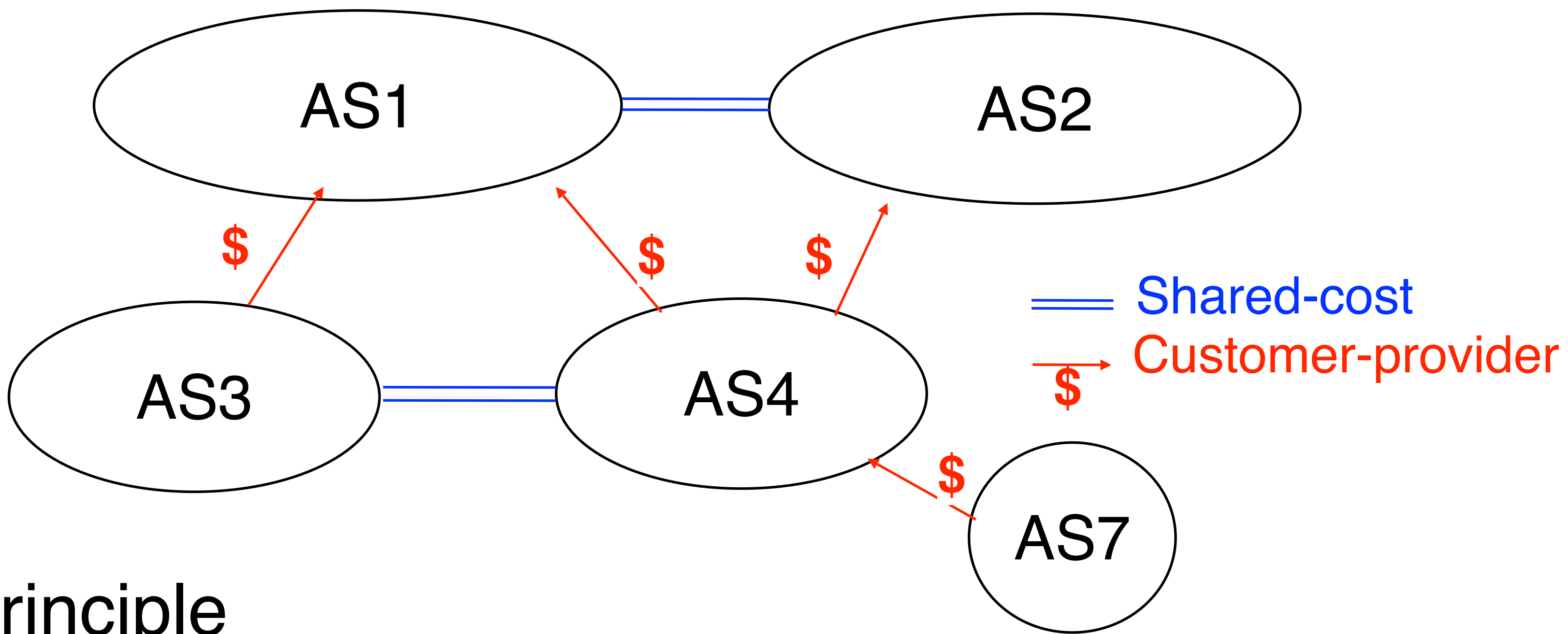
  - shared-cost peering
    - **Domains x** and **y** agree to exchange packets by using a direct link or through an interconnection point

Customer-provider peering



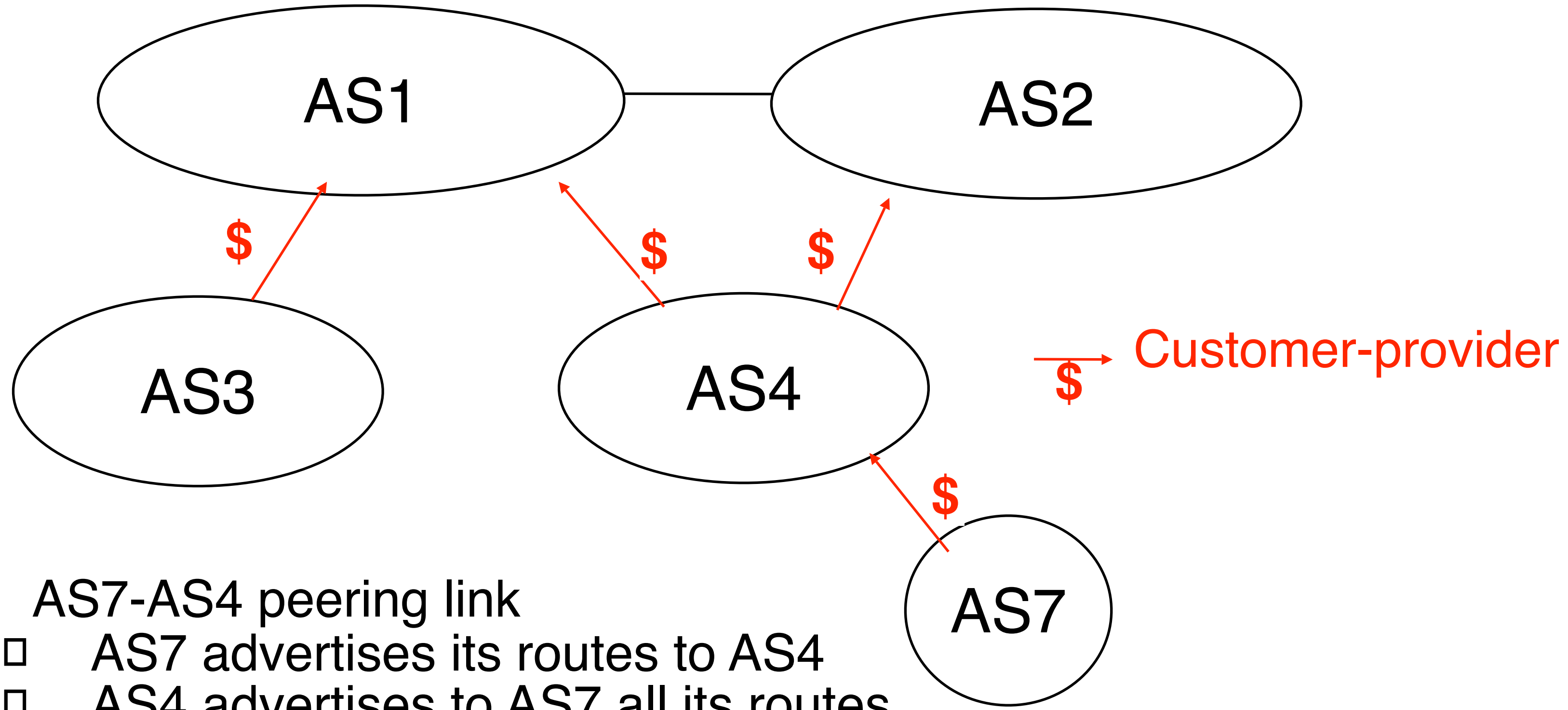
- Principle
  - Customer sends to its provider its internal routes and the routes learned from its own customers
    - Provider will advertise those routes to the entire Internet to allow anyone to reach the Customer
  - Provider sends to its customers all known routes
    - Customer will be able to reach anyone on the Internet

Shared-cost peering



- Principle
  - PeerX sends to PeerY its internal routes and the routes learned from its own customers
    - PeerY will use shared link to reach PeerX and PeerX's customers
    - PeerX's providers are not reachable via the shared link
  - PeerY sends to PeerX its internal routes and the routes learned from its own customers
    - PeerX will use shared link to reach PeerY and PeerY's customers
    - PeerY's providers are not reachable via the shared link

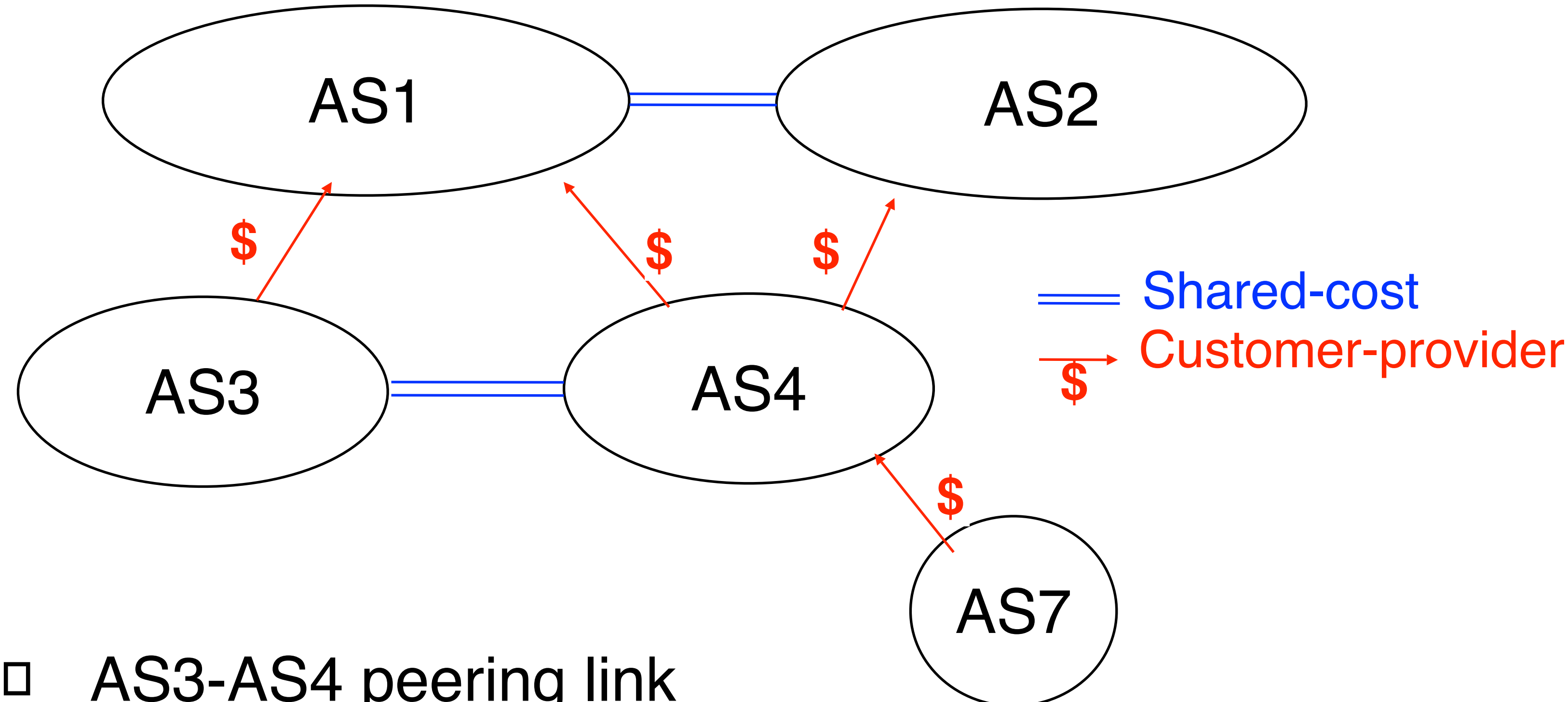
Customer-provider peering : example



- AS7-AS4 peering link
  - AS7 advertises its routes to AS4
  - AS4 advertises to AS7 all its routes

- AS4-AS2 peering link
  - AS4 advertises its own routes et those of its customers (AS7)
  - AS2 advertises to AS2 all known routes

Shared-cost peering : example



- AS3-AS4 peering link
  - AS3 advertises its own routes
  - AS4 advertises its own routes and those received from its clients (AS7)
- AS1-AS2 peering link
  - AS1 advertises its own routes and those received from its clients (AS3 and AS4)
  - AS1 advertises its own routes and those received from its clients (AS4)

Routing policies

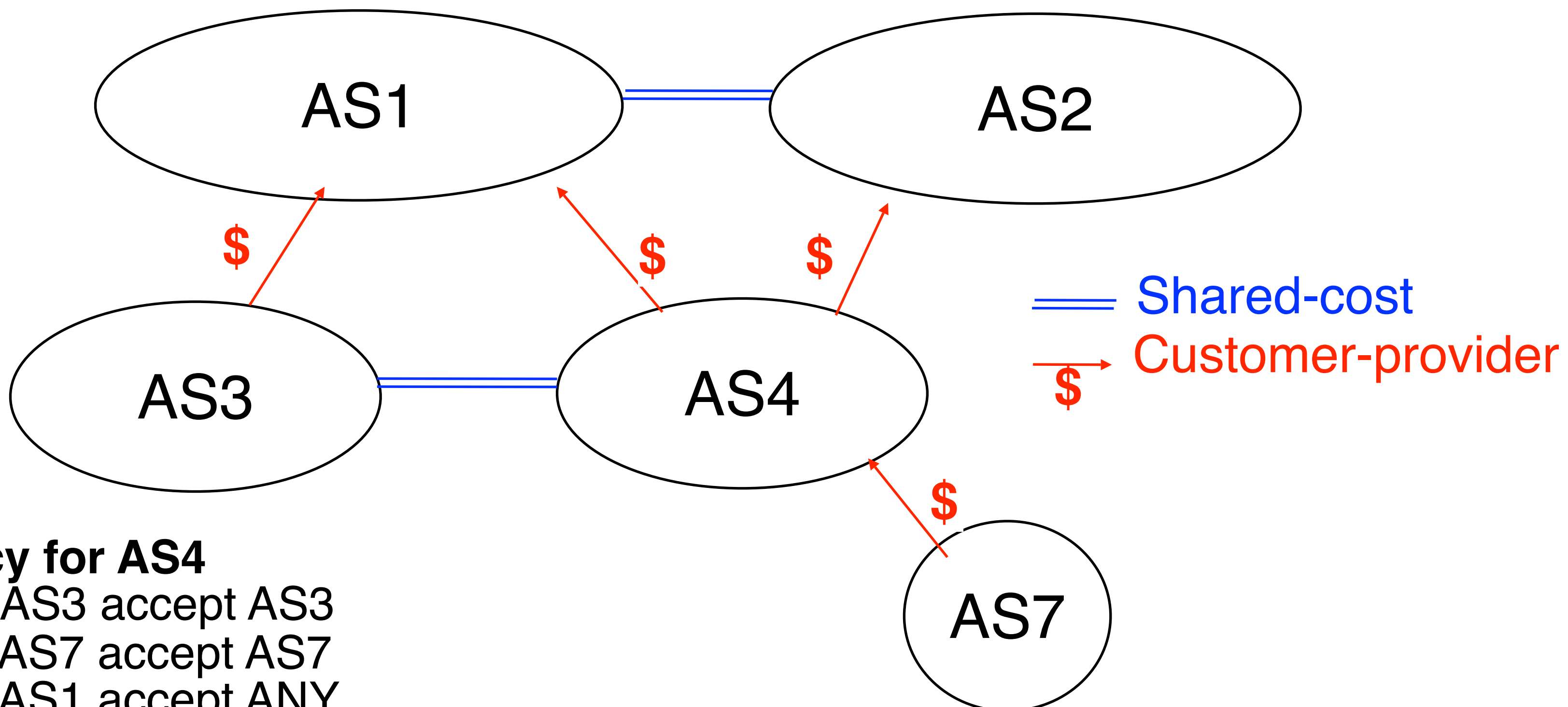
- A domain specifies its routing policy by defining on each BGP router two sets of filters for each peer
  - Import filter
    - Specifies which routes can be accepted by the router among all the received routes from a given peer
  - Export filter
    - Specifies which routes can be advertised by the router to a given peer
- Filters can be defined in RPSL
  - Routing Policy Specification Language
    - defined in RFC2622 and examples in RFC2650
      - See also http://www.ripe.net/ripencc/pub-services/whois.html

RPSL

- Simple import policies
  - Syntax
    - `import: from AS# accept list_of_AS`
  - Examples
    - `Import: from Belgacom accept Belgacom WIN`
    - `Import: from Provider accept ANY`

- Simple export policies
  - Syntax
    - `Export: to AS# announce list_of_AS`
  - Example
    - `Export: to Customer announce ANY`
    - `Export: to Peer announce Customer1 Customer2`

Routing policies
Simple example with RPSL



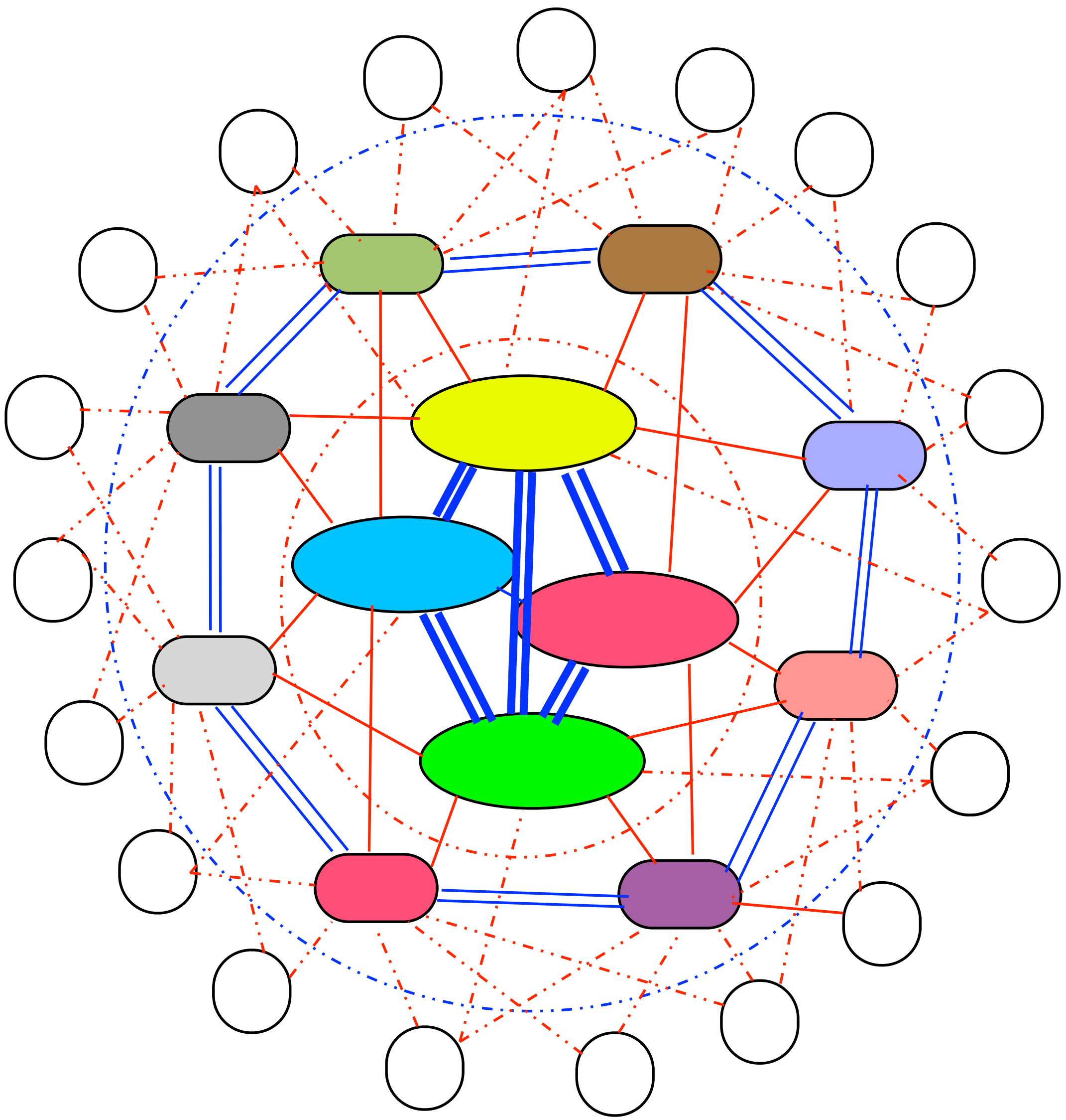
Import policy for AS4
Import: from AS3 accept AS3
import: from AS7 accept AS7
import: from AS1 accept ANY
import: from AS2 accept ANY

Export policy for AS4
export: to AS3 announce AS4 AS7
export: to AS7 announce ANY
export: to AS1 announce AS4 AS7
export: to AS2 announce AS4 AS7

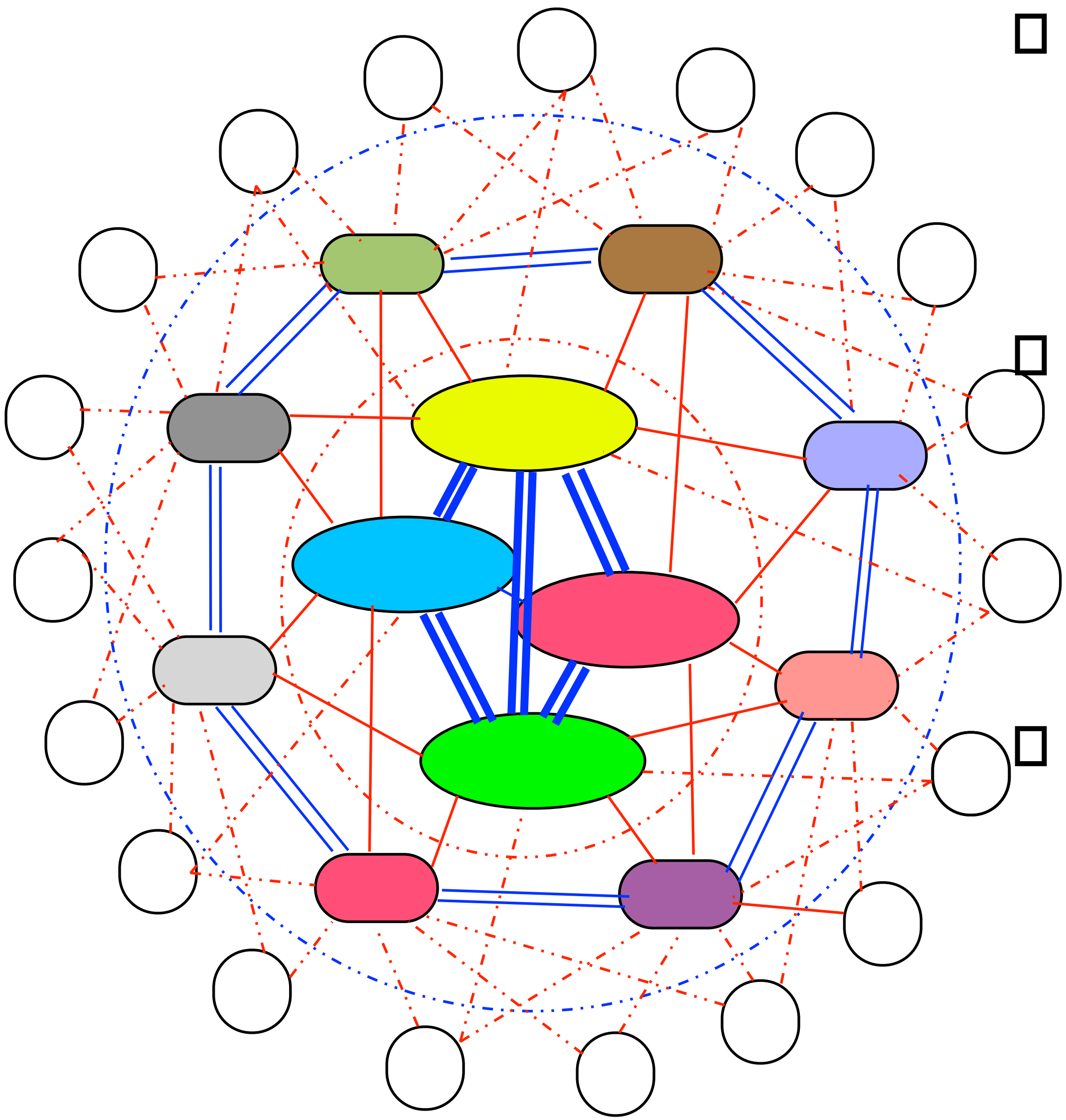
Import policy for AS7
Import: from AS4 accept ANY

Export policy for AS4
export: to AS4 announce AS7

The organisation of the Internet



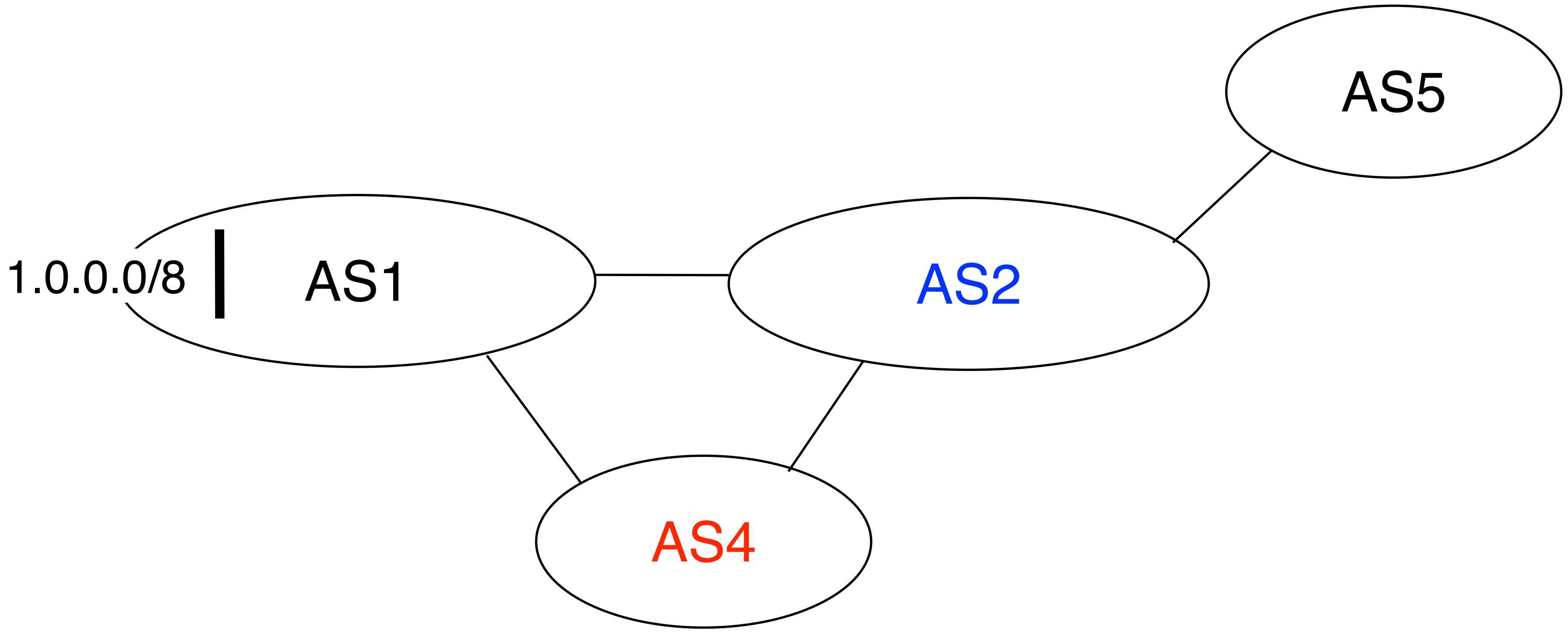
The organisation of the Internet



- Tier-1 ISPs
  - Dozen of large ISPs interconnected by shared-cost
  - Provide transit service
    - Uunet, Level3, OpenTransit, ...
- Tier-2 ISPs
  - Regional or National ISPs
  - Customer of T1 ISP(s)
  - Provider of T3 ISP(s)
  - shared-cost with other T2 ISPs
    - France Telecom, BT, Belgacom
- Tier-3 ISPs
  - Smaller ISPs, Corporate Networks, Content providers
  - Customers of T2 or T1 ISPs
  - shared-cost with other T3 ISPs

The Border Gateway Protocol

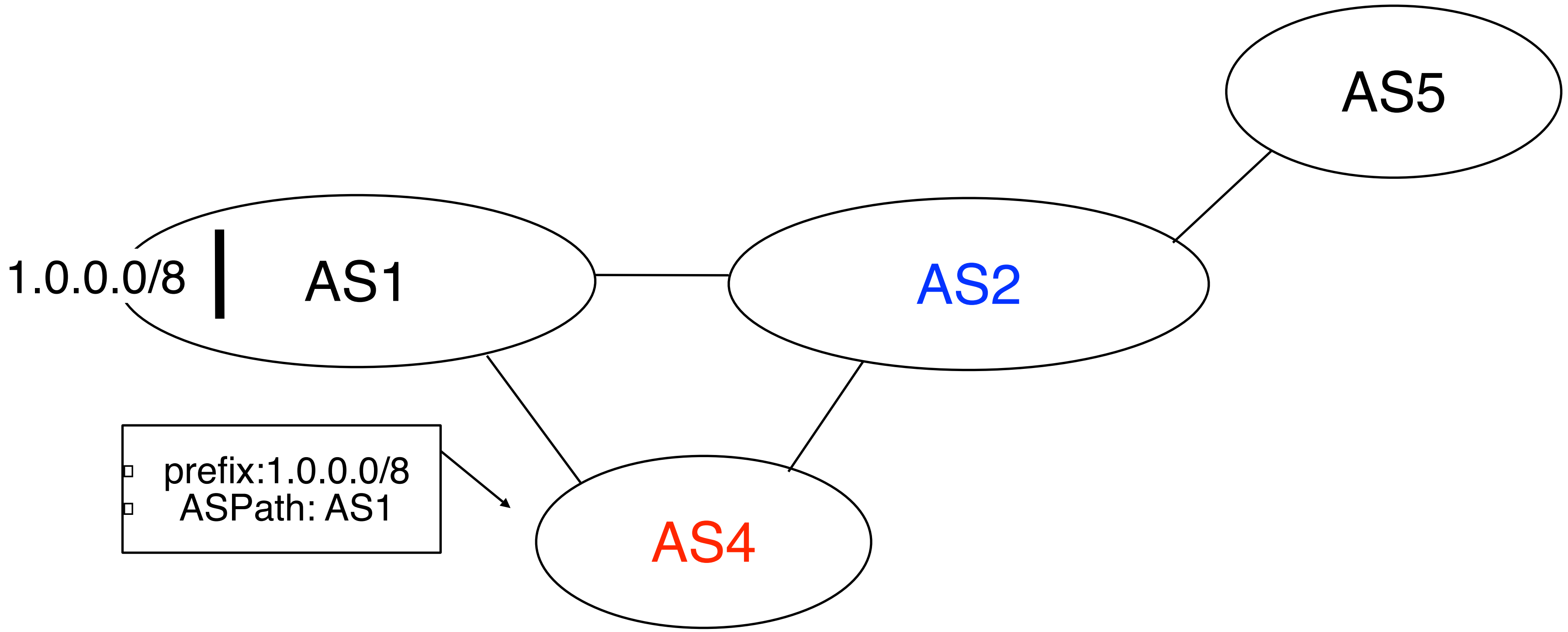
- Principle
  - Path vector protocol
    - BGP router advertises its best route to each destination



BGP is defined in RFC4271

The Border Gateway Protocol

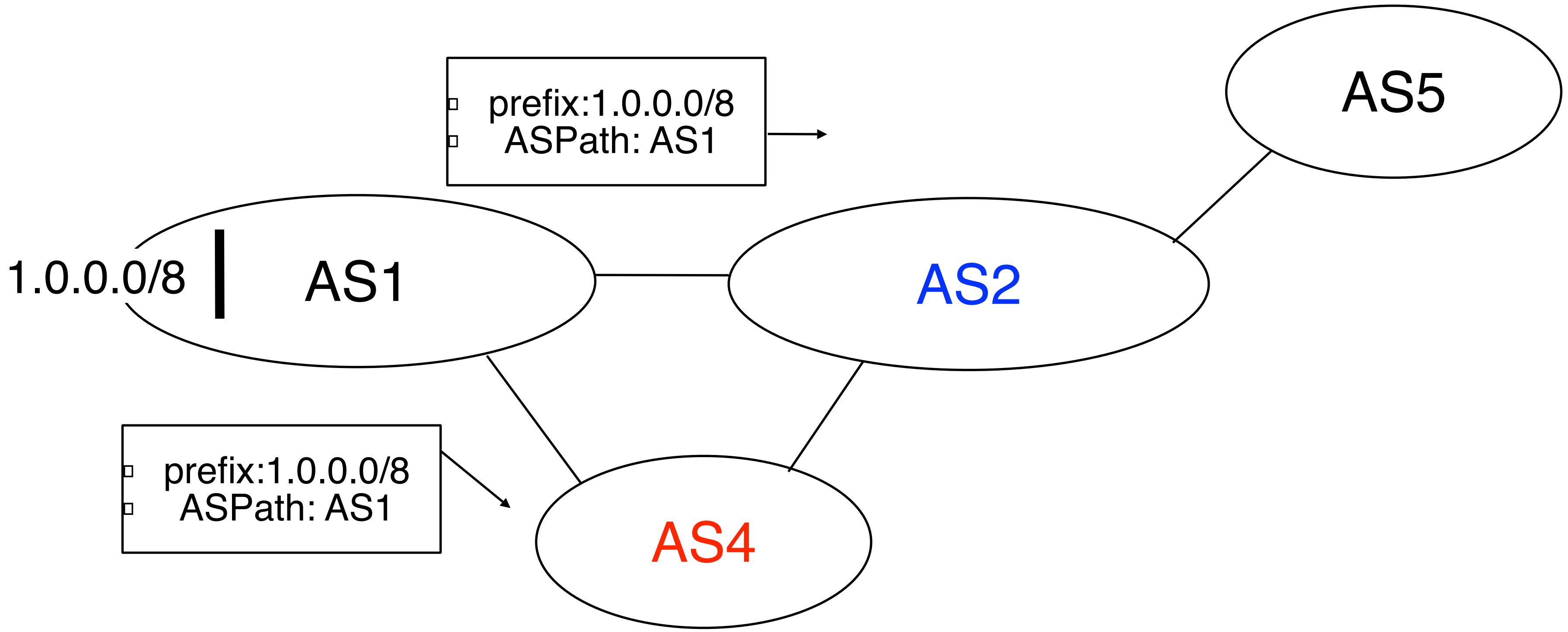
- Principle
  - Path vector protocol
    - BGP router advertises its best route to each destination



BGP is defined in RFC4271

The Border Gateway Protocol

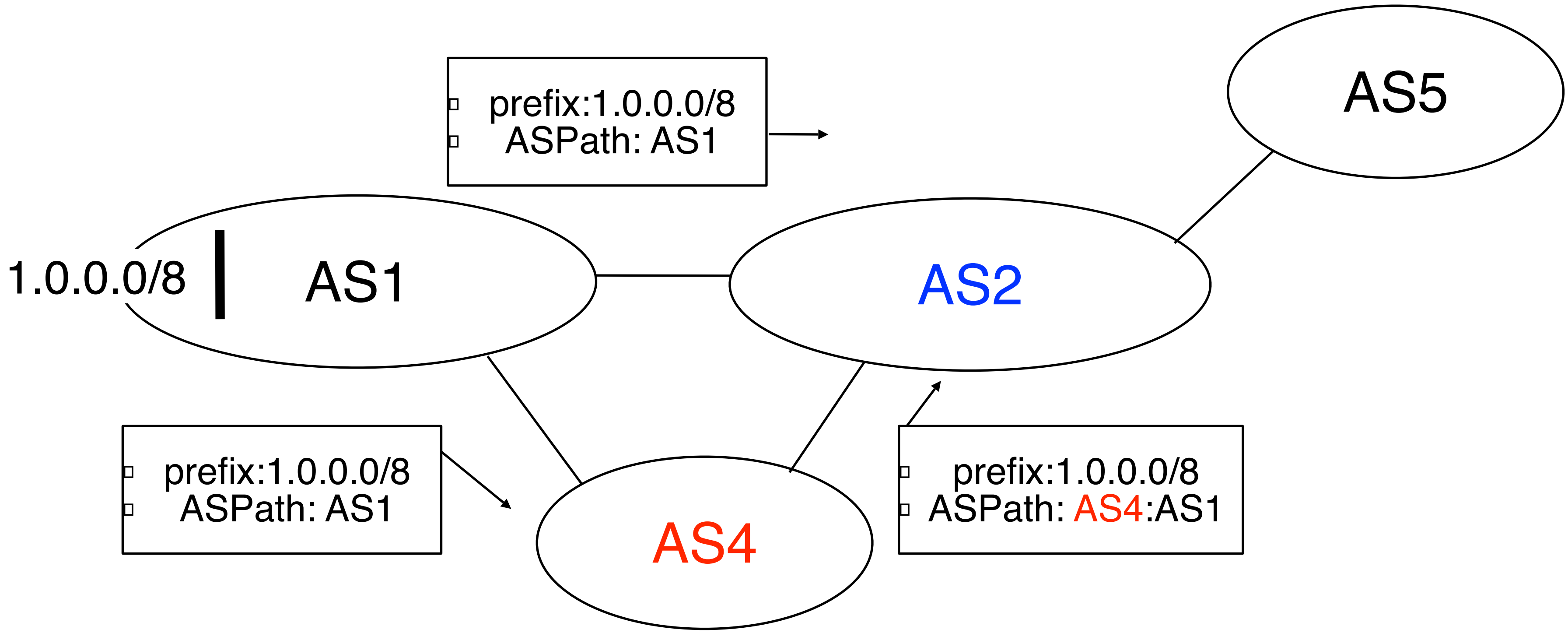
- Principle
  - Path vector protocol
    - BGP router advertises its best route to each destination



BGP is defined in RFC4271

The Border Gateway Protocol

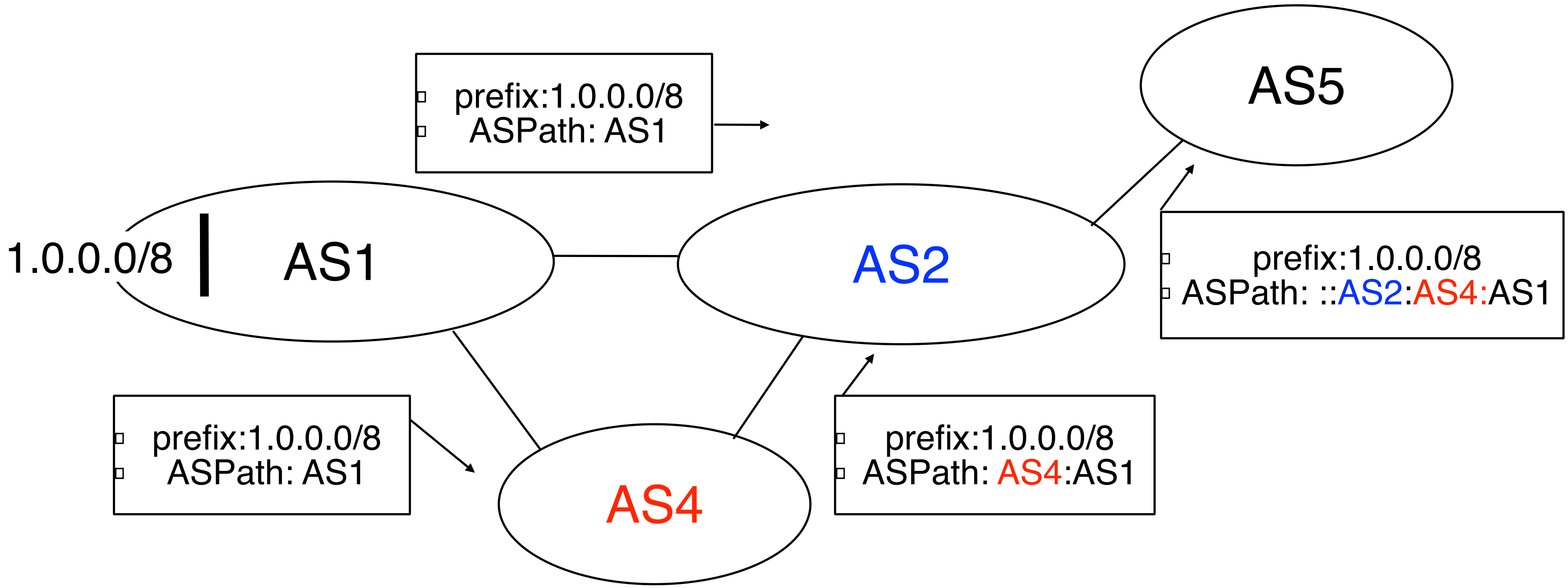
- Principle
  - Path vector protocol
    - BGP router advertises its best route to each destination



BGP is defined in RFC4271

The Border Gateway Protocol

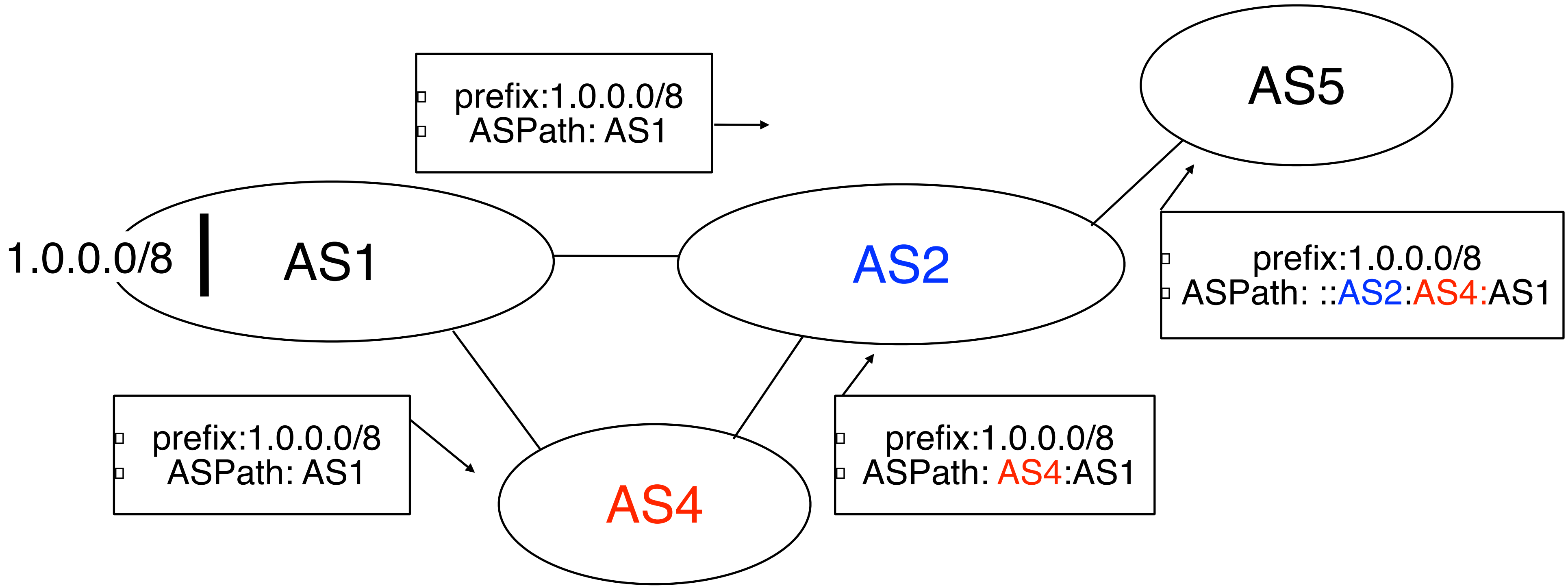
- Principle
  - Path vector protocol
    - BGP router advertises its best route to each destination



BGP is defined in RFC4271

The Border Gateway Protocol

- Principle
  - Path vector protocol
    - BGP router advertises its best route to each destination

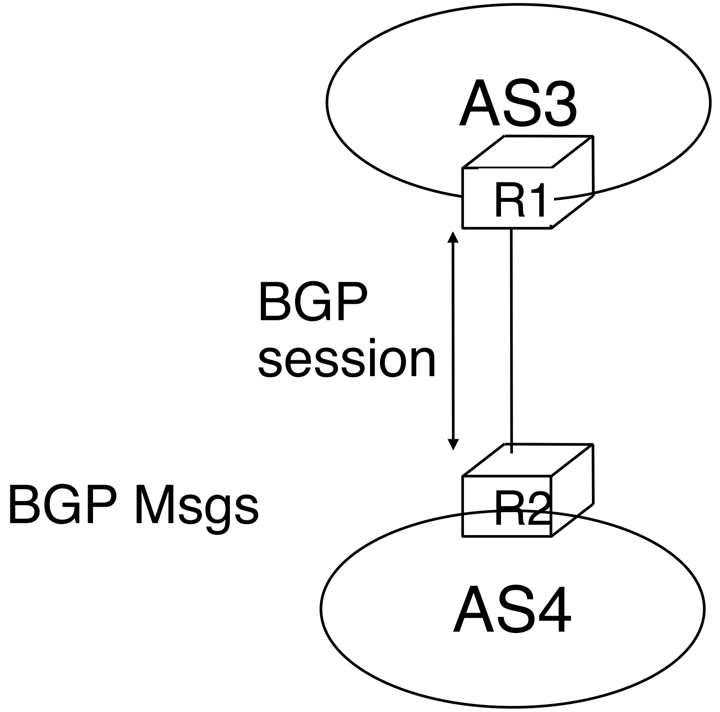


- ... with incremental updates
  - Advertisements are only sent when their content changes

BGP is defined in RFC4271

BGP : Principles of operation

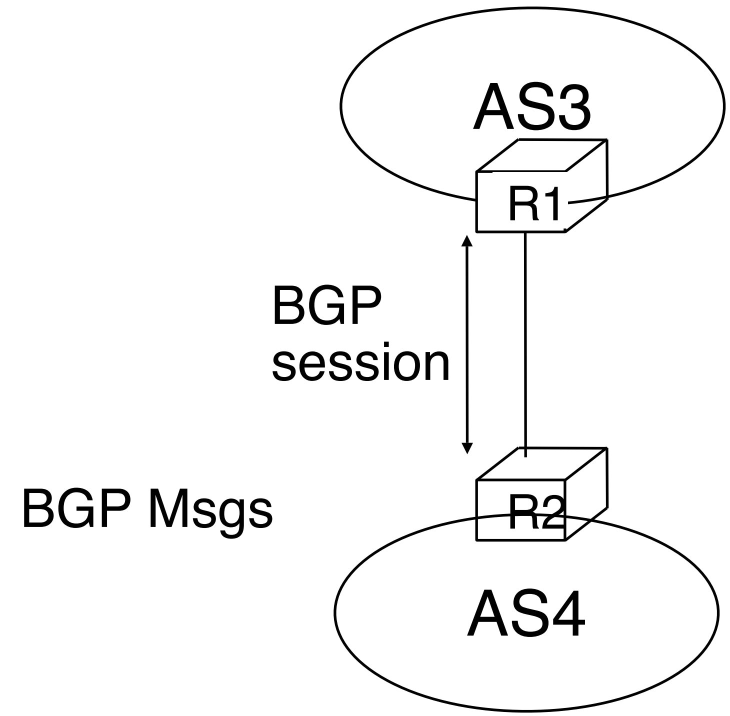
- Principles
  - BGP relies on the incremental exchange of path vectors



BGP : Principles of operation

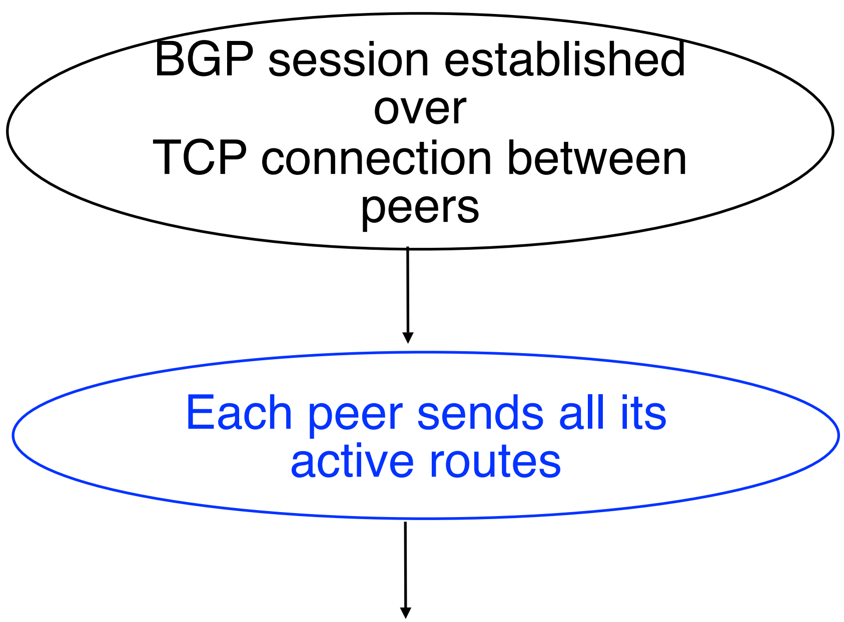
- Principles
  - BGP relies on the incremental exchange of path vectors

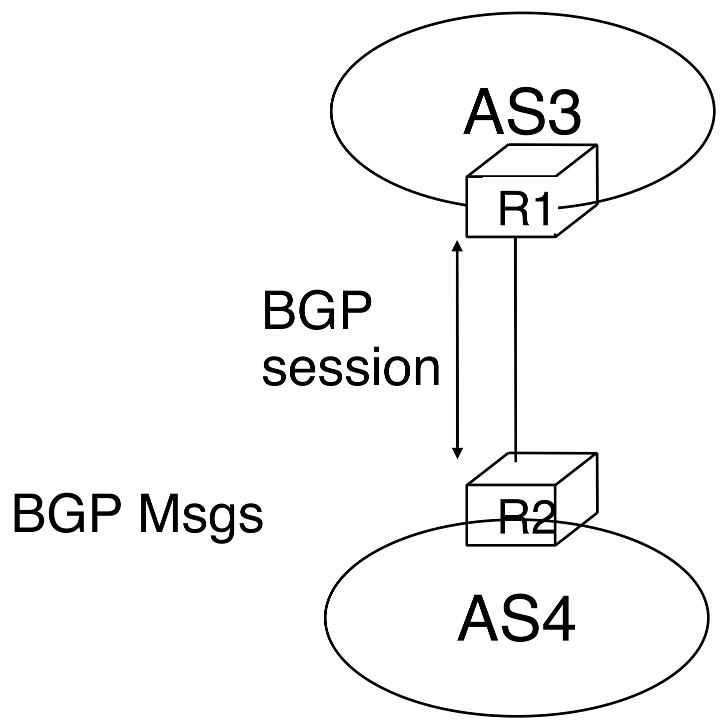
BGP session established over TCP connection between peers



BGP : Principles of operation

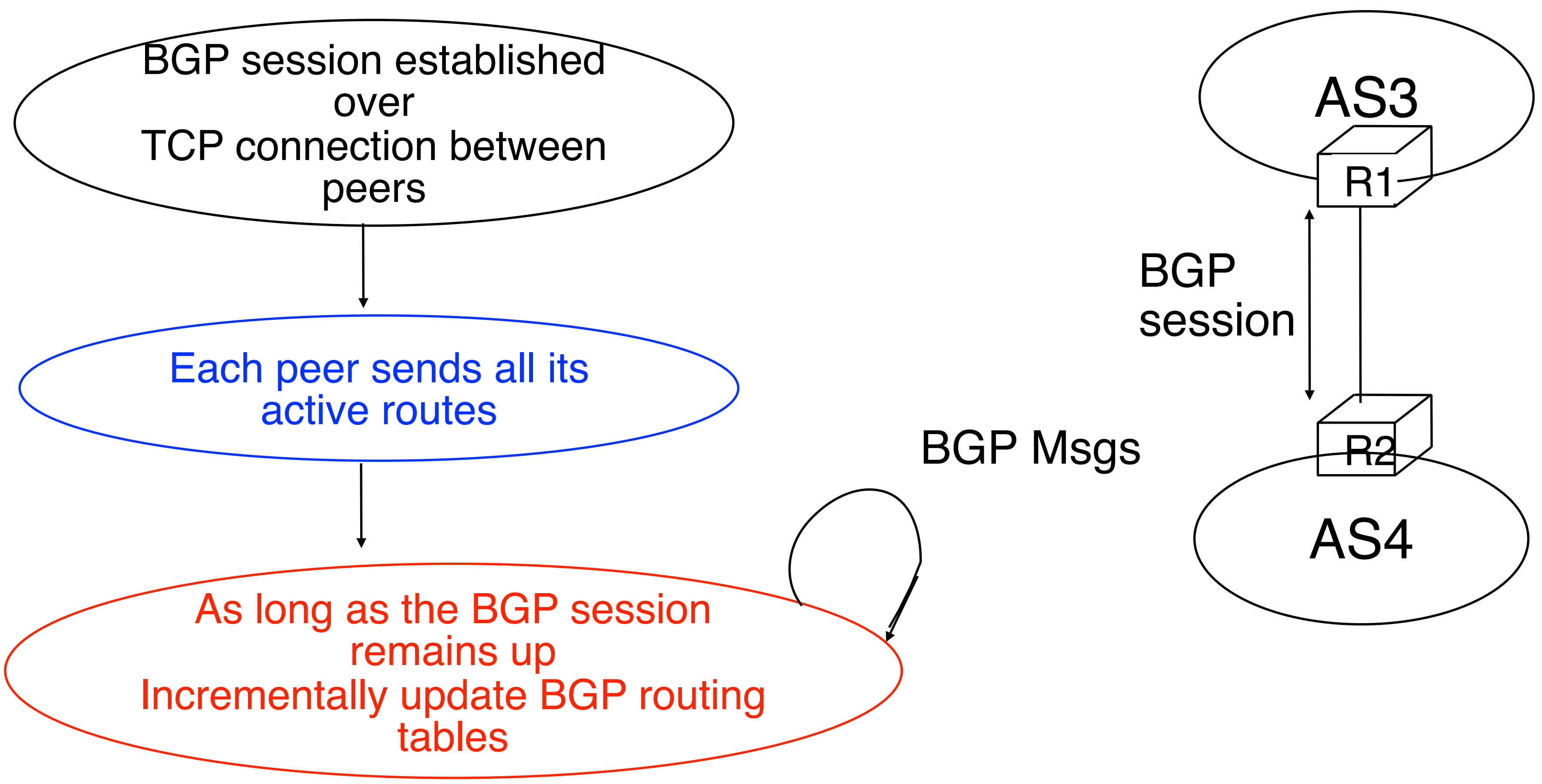
- Principles
  - BGP relies on the incremental exchange of path vectors





BGP : Principles of operation

- Principles
  - BGP relies on the
    incremental exchange of path vectors



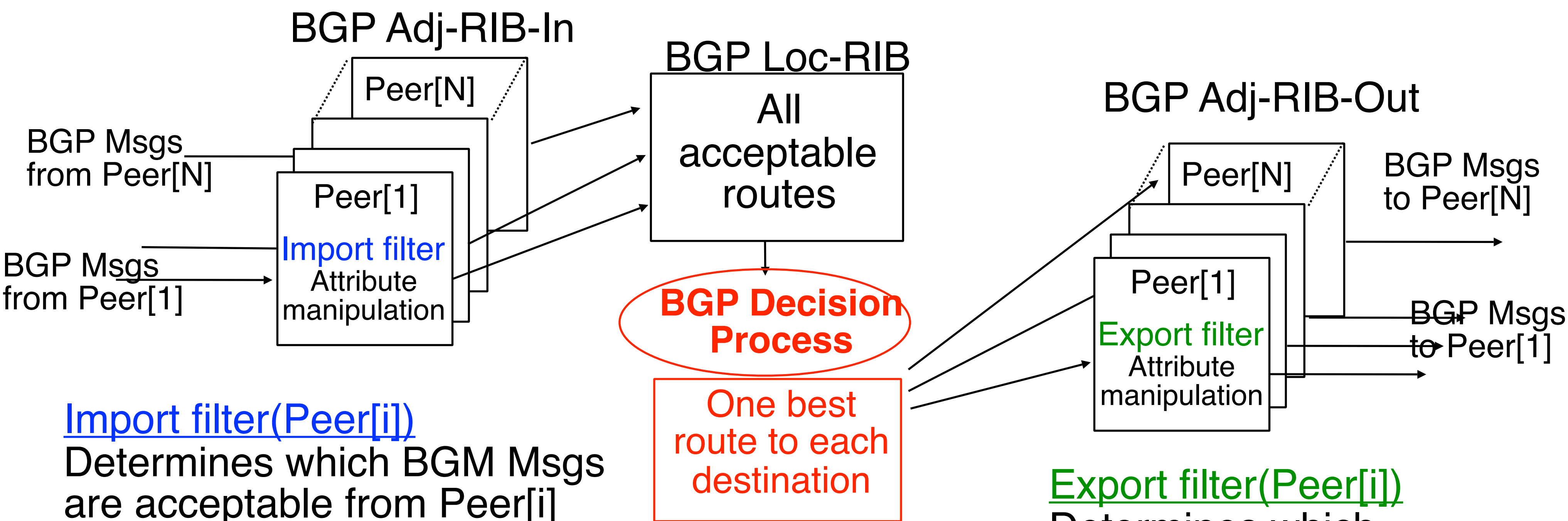
BGP : Principles of operation (2)

- Simplified model of BGP
  - 2 types of BGP path vectors

  - UPDATE
    - Used to announce a route towards one prefix
    - Content of UPDATE
      - Destination address/prefix
      - Interdomain path used to reach destination (AS-Path)
      - Nexthop (address of the router advertising the route)

  - WITHDRAW
    - Used to indicate that a previously announced route is not reachable anymore
    - Content of WITHDRAW
      - Unreachable destination address/prefix

Conceptual model of a BGP router



Import filter(Peer[i])
Determines which BGM Msgs
are acceptable from Peer[i]

Export filter(Peer[i])
Determines which
routes can be sent to Peer[i]

BGP Routing Information Base
Contains all the acceptable routes
learned from all Peers + internal routes

- BGP decision process selects *the* best route towards each destination

Where do the routes advertised by BGP routers come from ?

- Learned from another BGP router
  - Each BGP router advertises best route towards each destination
- Static route
  - Configured manually on the router
    - Ex : The BGP router at UCL advertises 130.104.0.0/16
    - Drawback
      - Requires manual configuration
    - Advantage
      - BGP advertisements are stable
- Learned from an intradomain routing protocol
  - BGP might try to aggregate the route before advertising it
  - Advantage :
    - BGP advertisements correspond to network status
  - Drawback
    - Routing instabilities inside a domain might propagate in Internet

BGP : Session Initialization

```
Initialize_BGP_Session(RemoteAS, RemoteIP)
{ /* Initialize and start BGP session */
/* Send BGP OPEN Message to RemoteIP on port 179*/
/* Follow BGP state machine */

/* advertise local routes and routes learned from peers*/
foreach (destination=d inside RIB)
 {
  B=build_BGP_UPDATE(d);
  S=apply_export_filter(RemoteAS,B);
  if (S<>NULL)
     { /* send UPDATE message */
       send_UPDATE(S,RemoteAS, RemoteIP)
     }
 }
/* entire RIB was sent */
/* new UPDATE will be sent only to reflect local or distant
   changes in routes */
...
}
```

Events during a BGP session

1. Addition of a new route to RIB
   - A new internal route was added on local router
     - static route added by configuration
     - Dynamic route learned from IGP
   - Reception of UPDATE message announcing a new or modified route
2. Removal of a route from RIB
   - Removal of an internal route
     - Static route is removed from router configuration
     - Intradomain route declared unreachable by IGP
   - Reception of WITHDRAW message
3. Loss of BGP session
   - All routes learned from this peer removed from RIB

Export and Import filters

```
BGPMsg Apply_export_filter(RemoteAS, BGPMsg)
{ /* check if Remote AS already received route */
if (RemoteAS isin BGPMsg.ASPath)
   BGPMsg==NULL;
/* Many additional export policies can be configured : */
/* Accept or refuse the BGPMsg */
/* Modify selected attributes inside BGPMsg */
}

BGPMsg apply_import_filter(RemoteAS, BGPMsg)
{ /* check that we are not already inside  ASPath */
 if (MyAS isin BGPMsg.ASPath)
   BGPMsg==NULL;
/* Many additional import policies can be configured : */
/* Accept or refuse the BGPMsg */
/* Modify selected attributes inside BGPMsg */
}
```

BGP : Processing of UPDATES

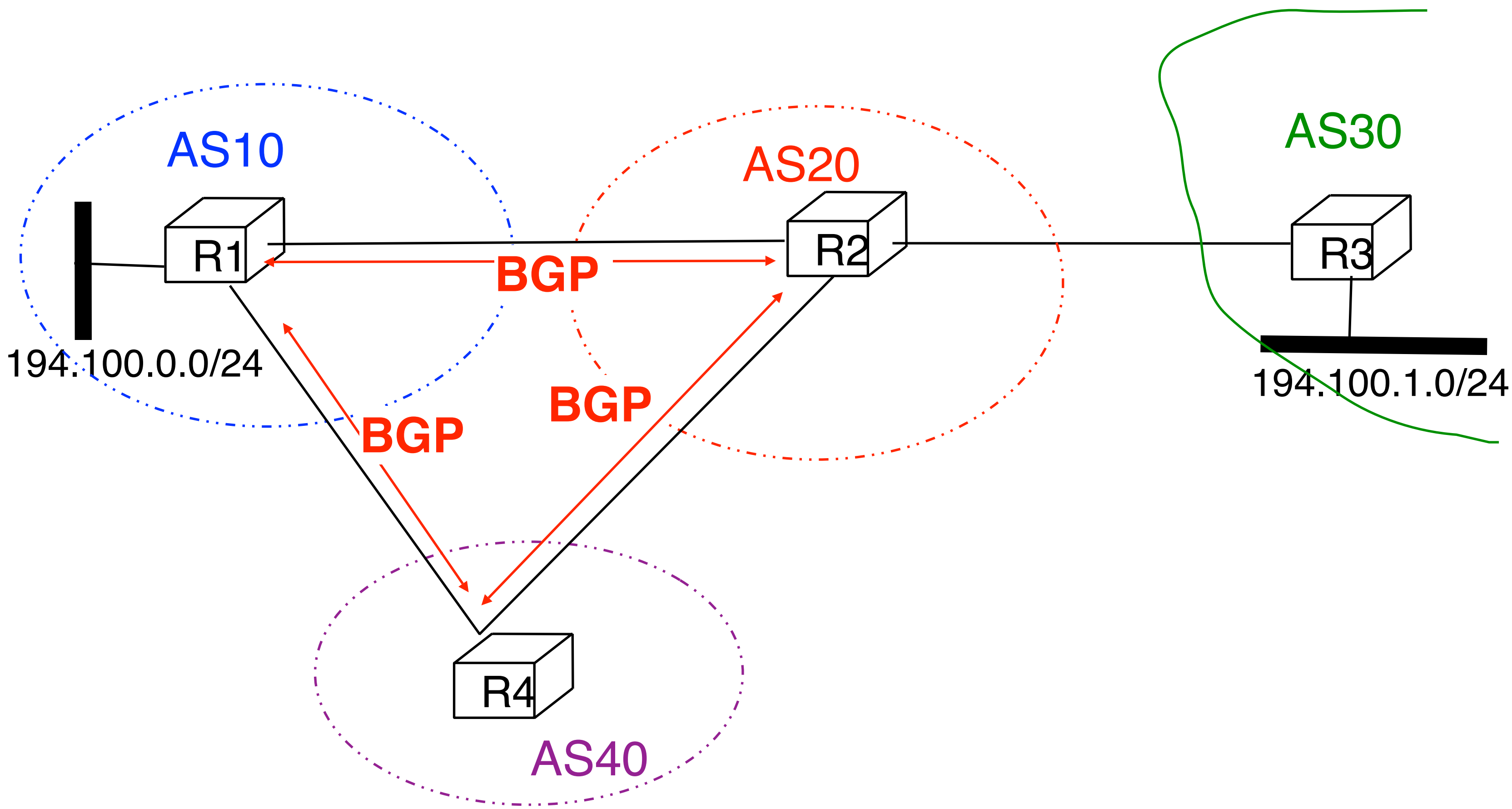
```
Recvd_BGPMsg(Msg, RemoteAS)
{
 B=apply_import_filer(Msg,RemoteAS);
 if (B==NULL) /* Msg not acceptable */
     exit();
 if IsUPDATE(Msg)
 {
  Old_Route=BestRoute(Msg.prefix);
  Insert_in_RIB(Msg);
  Run_Decision_Process(RIB);
  if (BestRoute(Msg.prefix)<>Old_Route)
  { /* best route changed */
    B=build_BGP_Message(Msg.prefix);
    S=apply_export_filter(RemoteAS,B);
    if (S<>NULL) /* announce best route */
     send_UPDATE(S,RemoteAS);
    else if (Old_Route<>NULL)
     send_WITHDRAW(Msg.prefix);
 } ...
```

BGP : Processing of WITHDRAW

```
Recvd_Msg(Msg, RemoteAS)
...
if IsWITHDRAW(Msg)
 {
  Old_Route=BestRoute(Msg.prefix);
  Remove_from_RIB(Msg);
  Run_Decision_Process(RIB);
  if (Best_Route(Msg.prefix)<>Old_Route)
  { /* best route changed */
    B=build_BGP_Message(d);
    S=apply_export_filter(RemoteAS,B);
    if (S<>NULL) /* still one best route */
       send_UPDATE(S,RemoteAS, RemoteIP);
    else if(Old_Route<>NULL)/* no best route anymore */
       send_WITHDRAW(Msg.prefix,RemoteAS,RemoteIP);
  }
 }
}
```

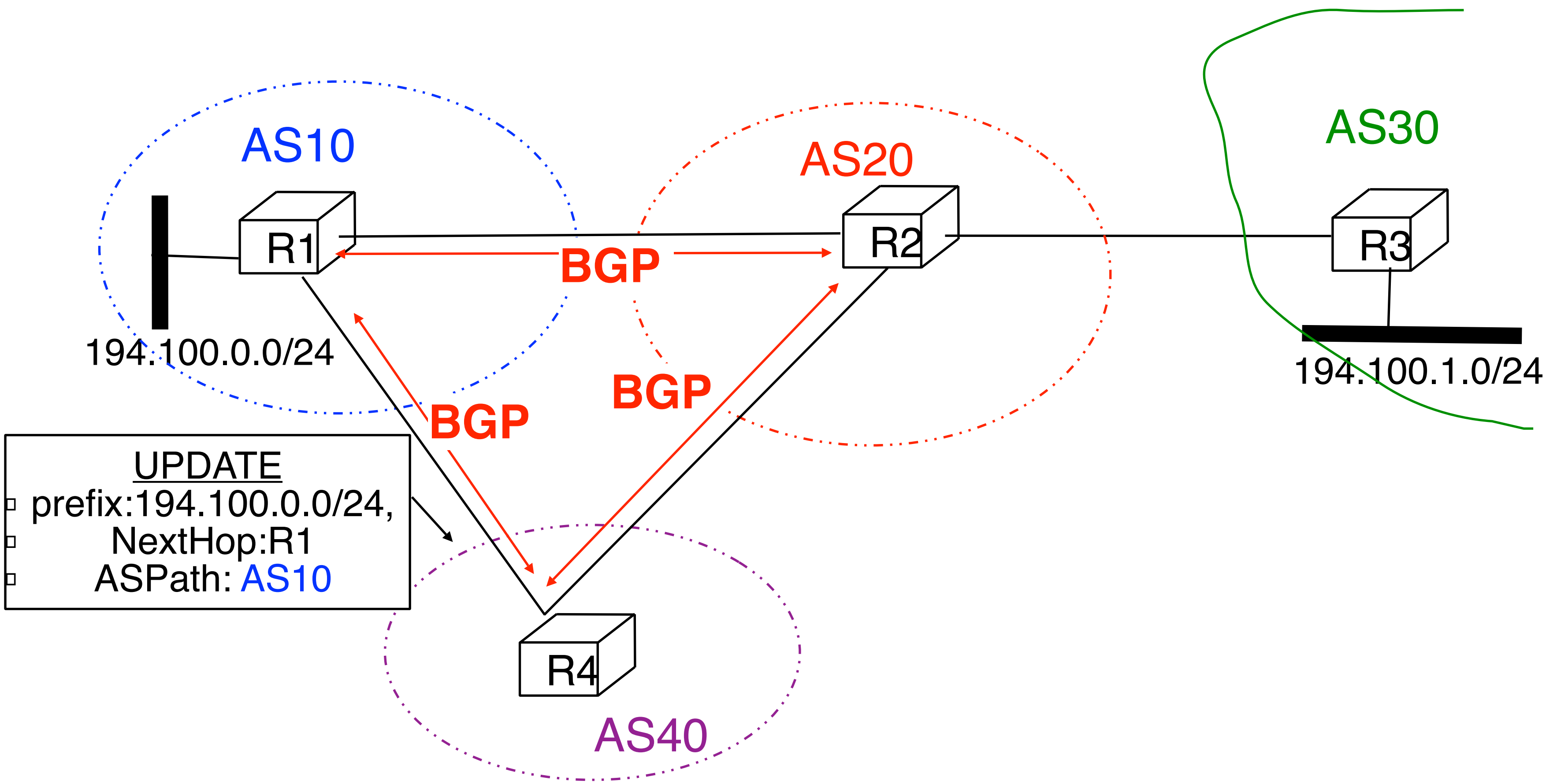
BGP and IP
A first example

- Initial updates



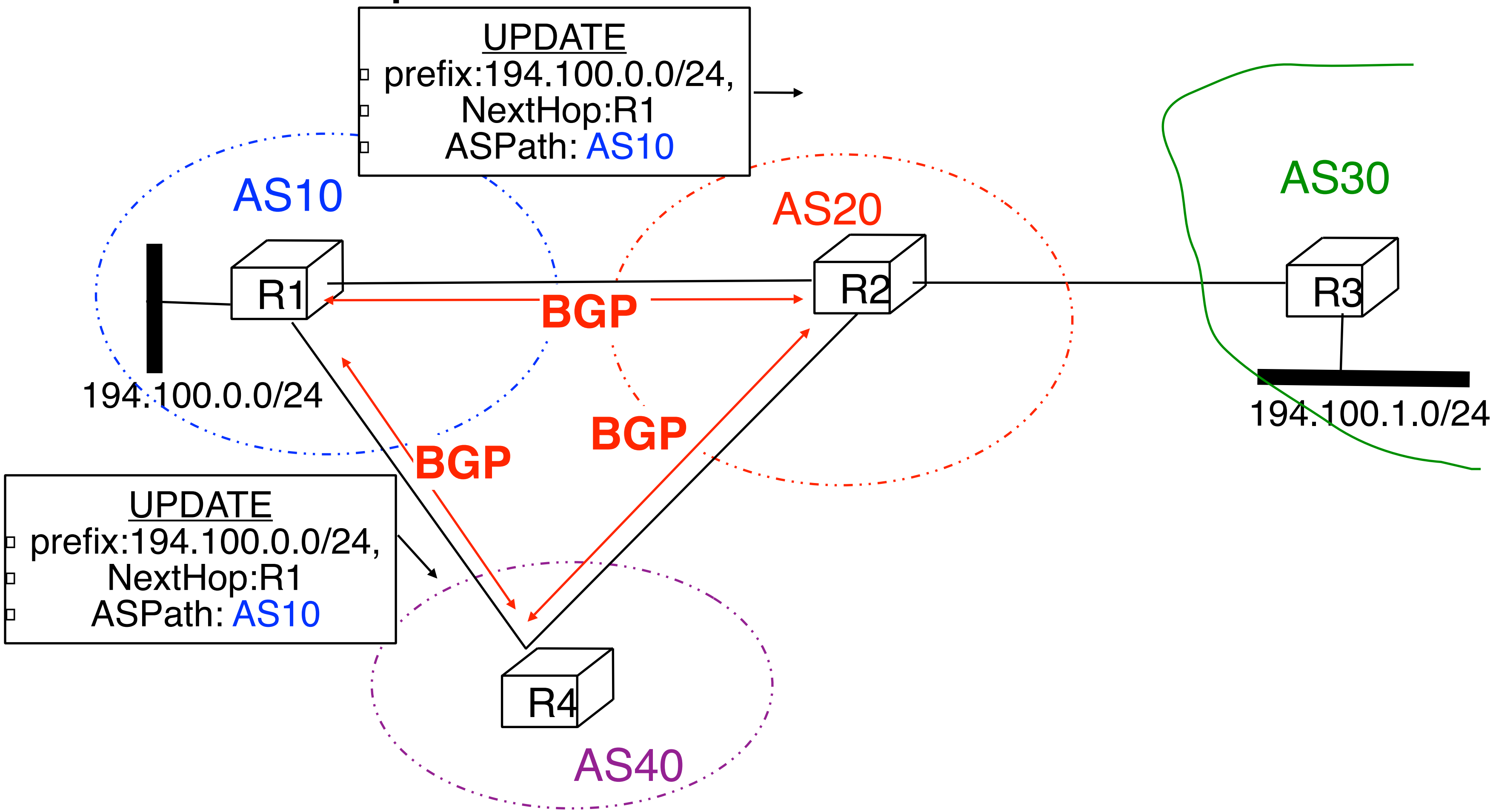
BGP and IP
A first example

- Initial updates



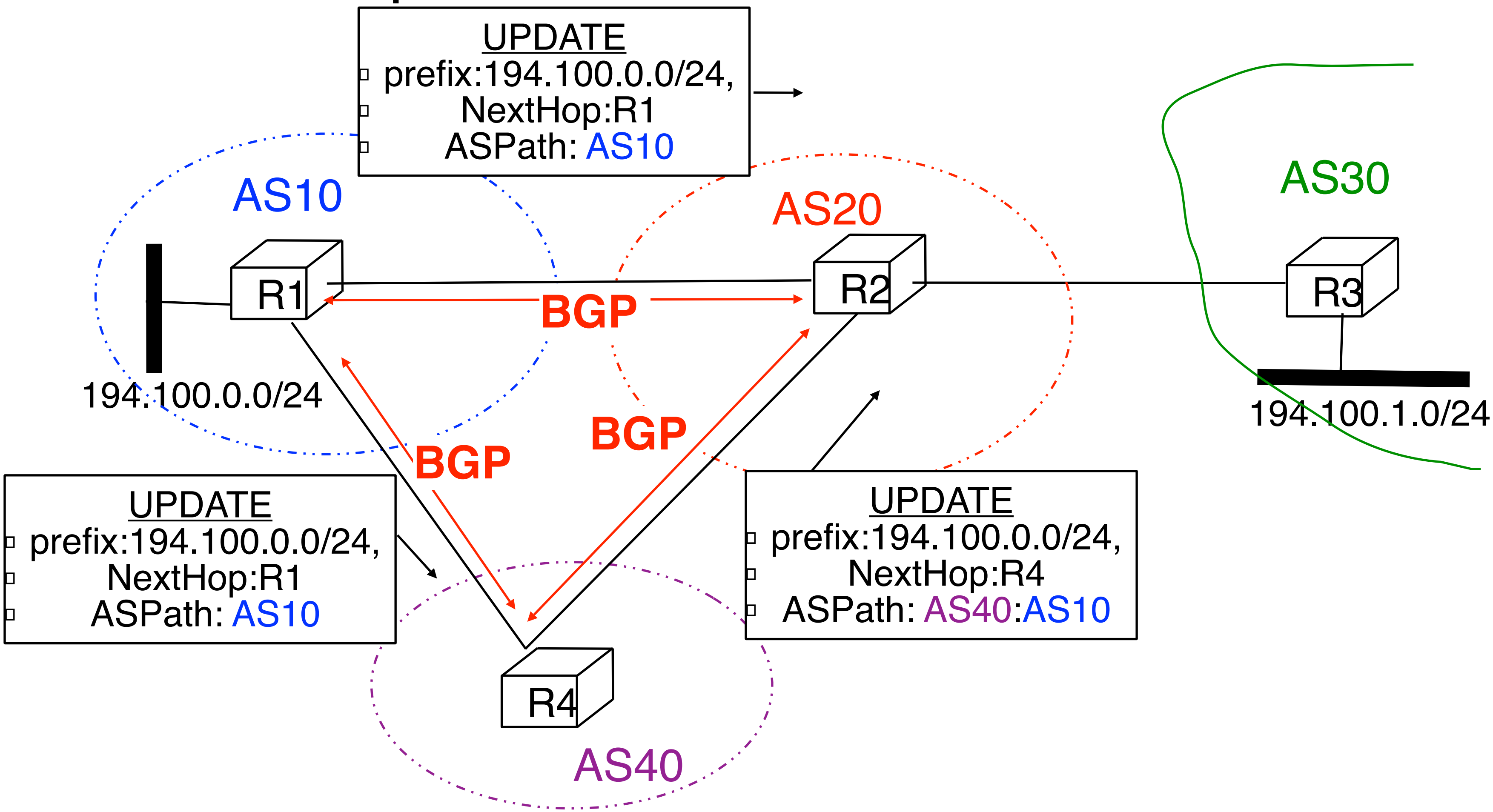
BGP and IP
A first example

- Initial updates



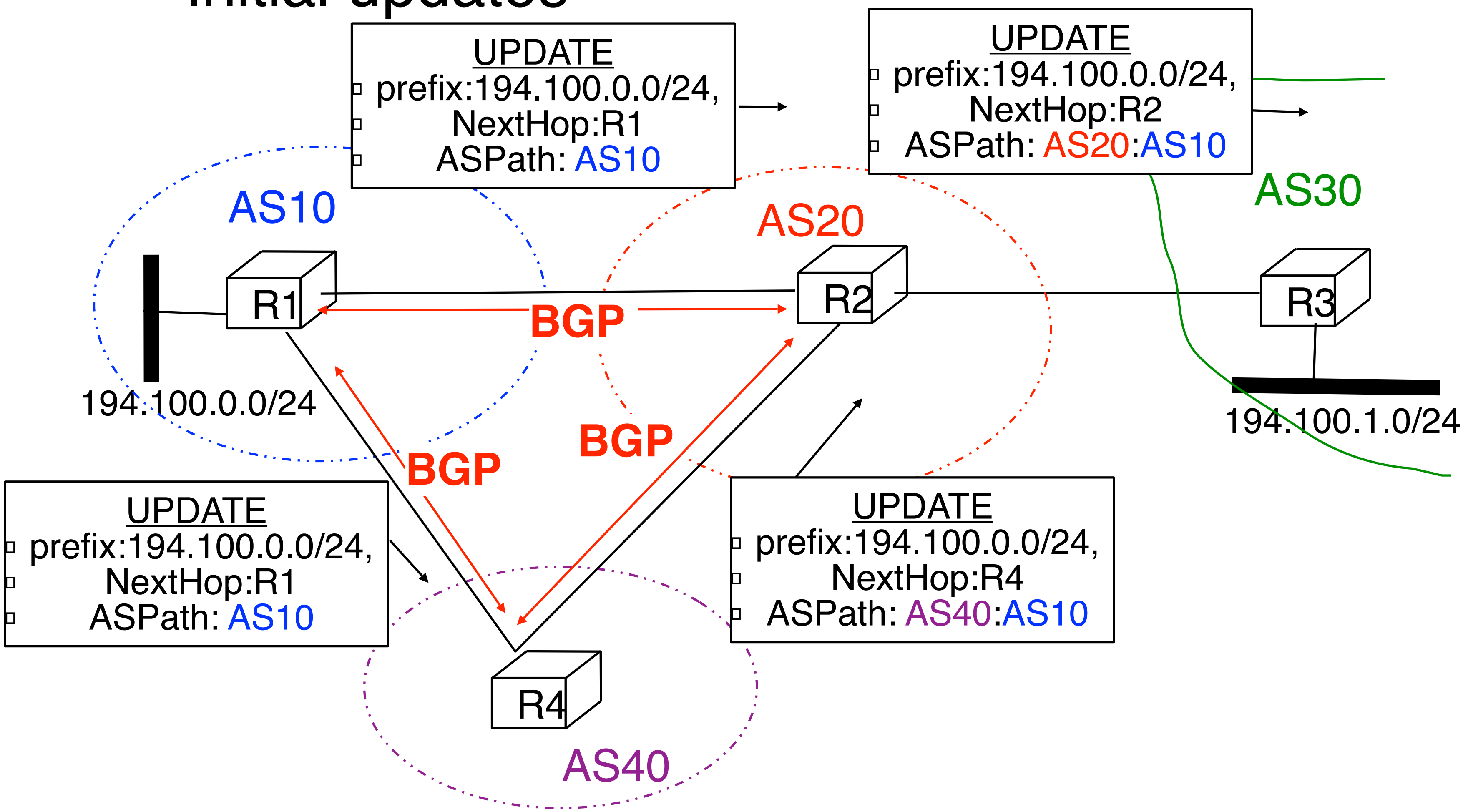
BGP and IP
A first example

- Initial updates



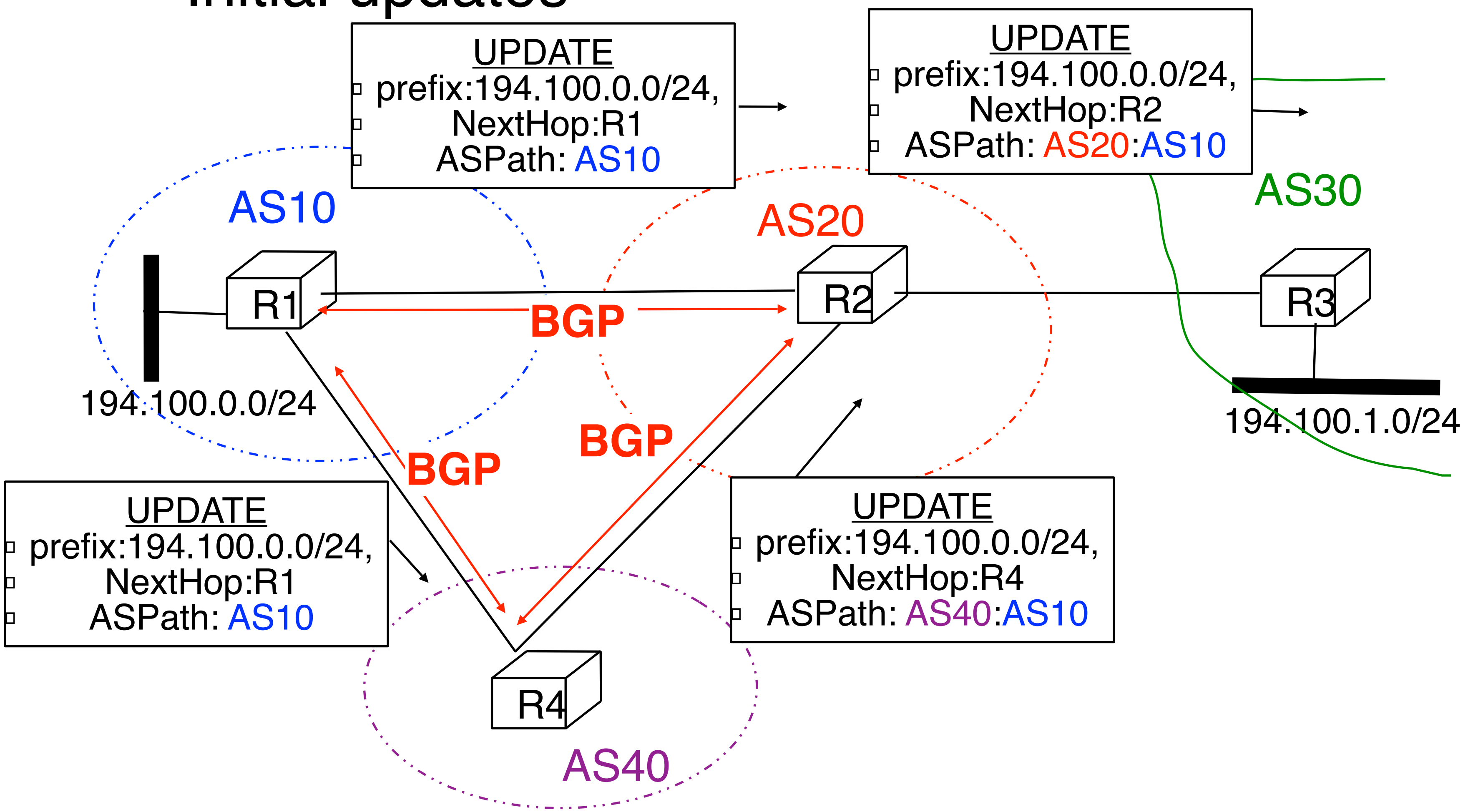
BGP and IP
A first example

- Initial updates



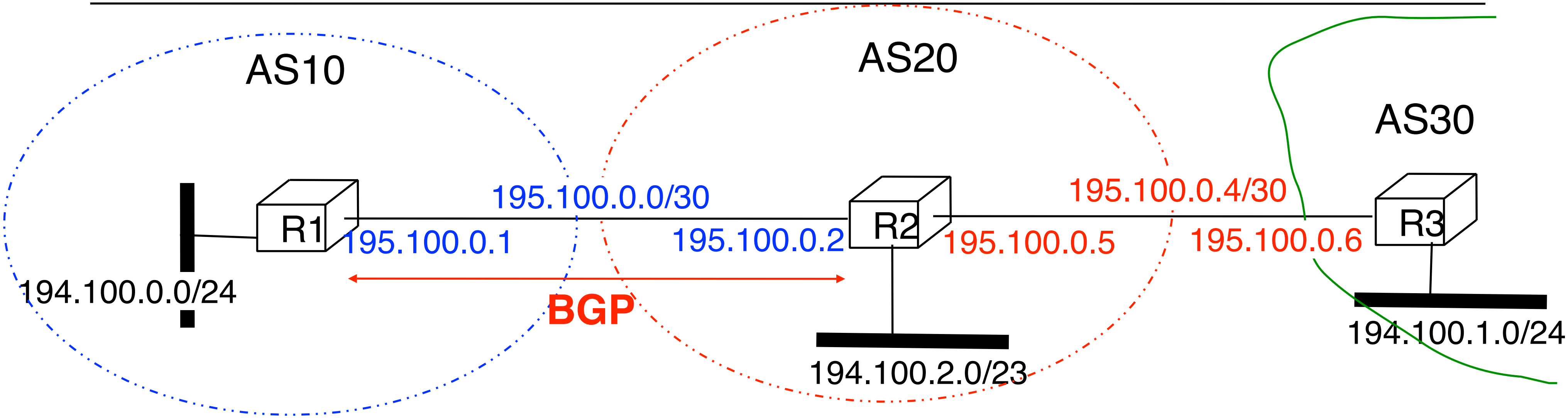
BGP and IP
A first example

- Initial updates



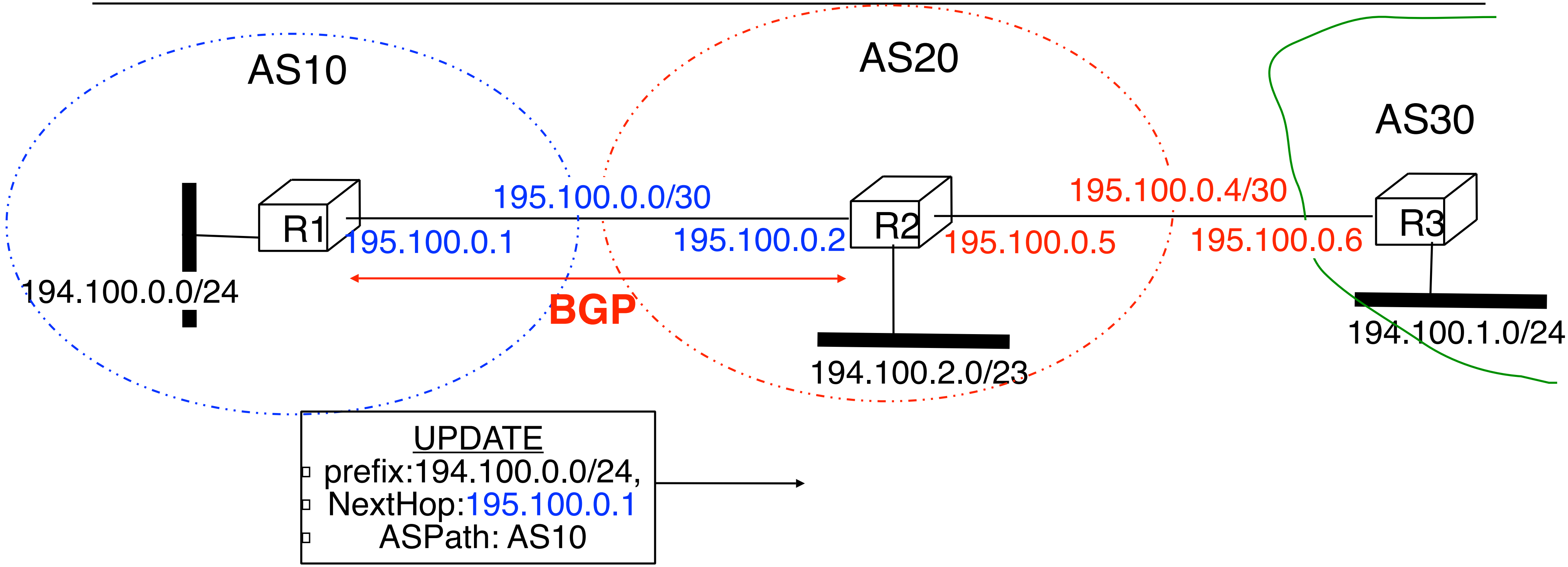
- What happens if link AS10-AS20 goes down ?

BGP and IP
A second example



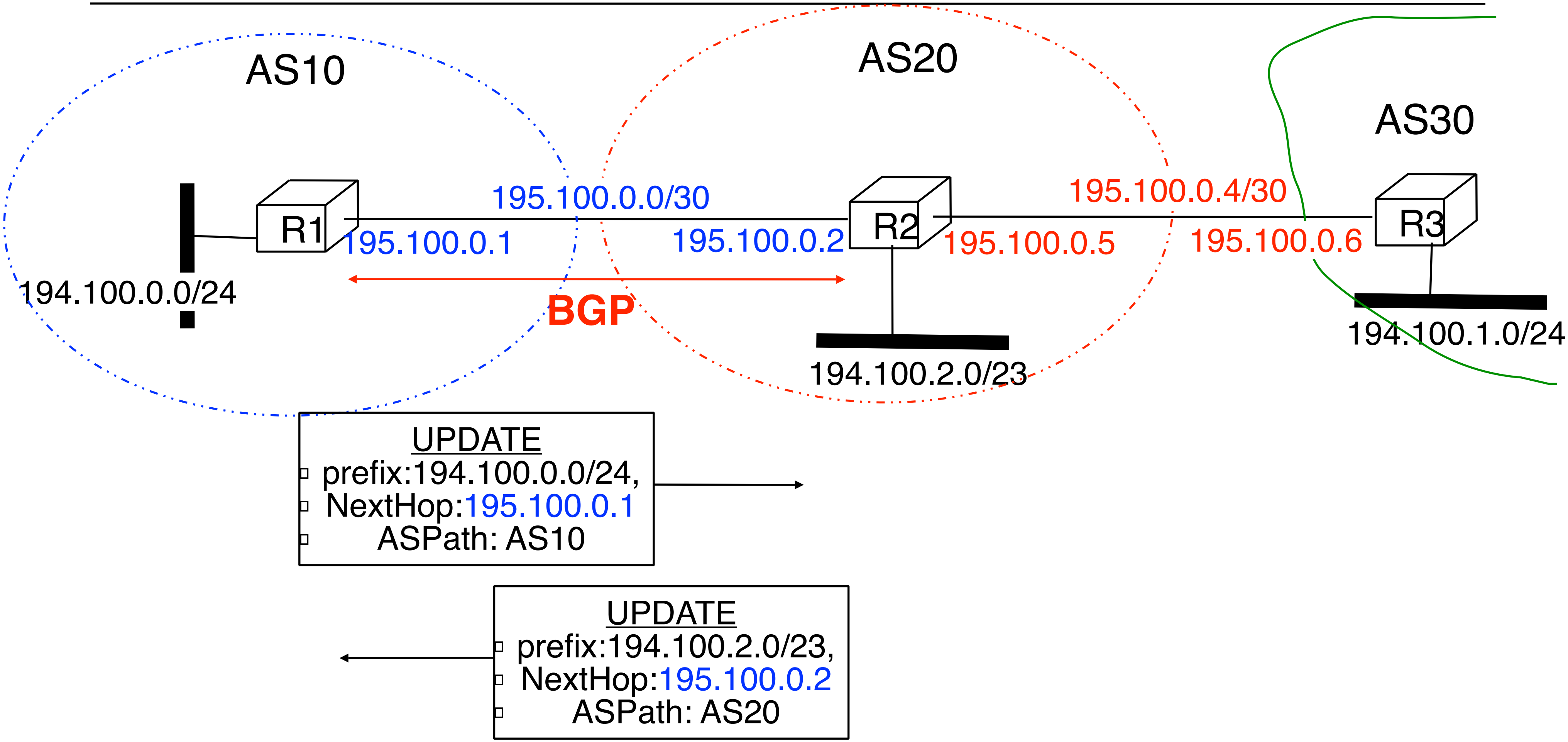
- Main Path attributes of UPDATE message
  - NextHop : IP address of router used to reach destination
  - ASPath : Path followed by the route advertisement

BGP and IP
A second example



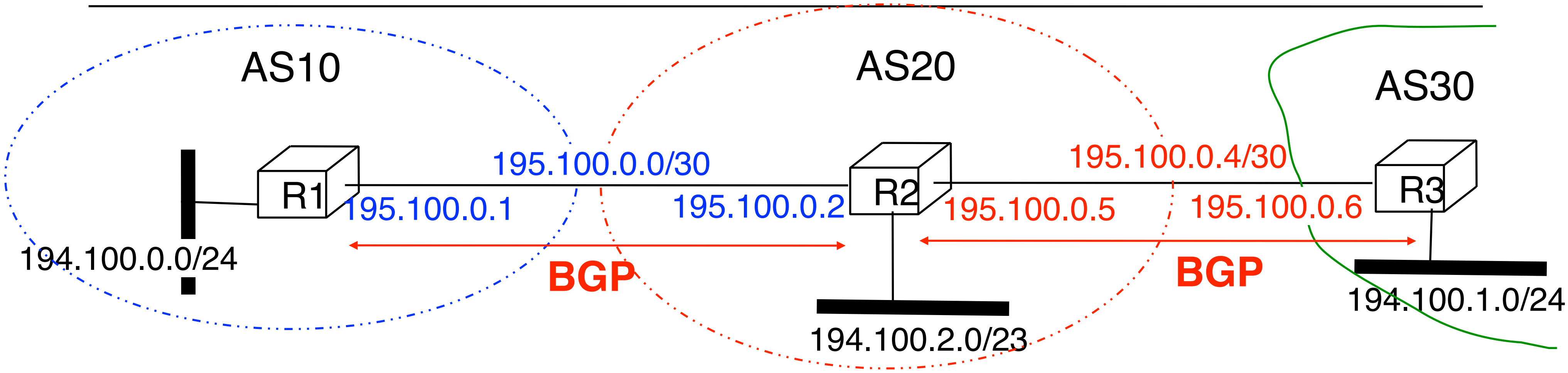
- Main Path attributes of UPDATE message
  - NextHop : IP address of router used to reach destination
  - ASPath : Path followed by the route advertisement

BGP and IP
A second example

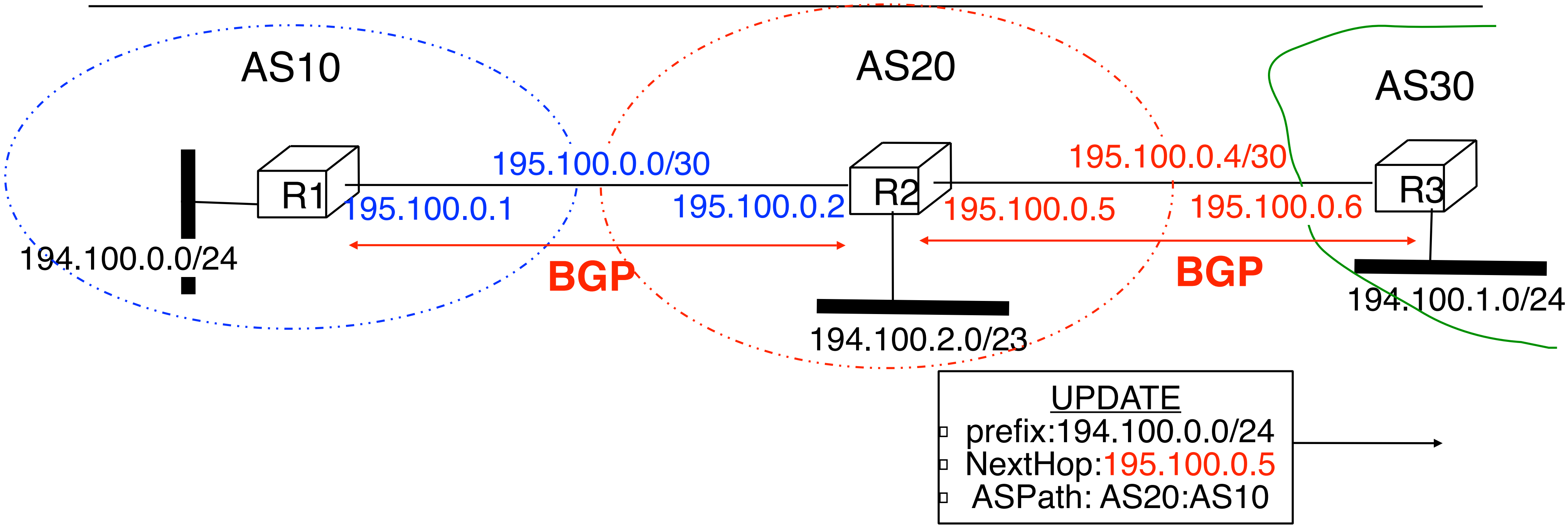


- Main Path attributes of UPDATE message
  - NextHop : IP address of router used to reach destination
  - ASPath : Path followed by the route advertisement

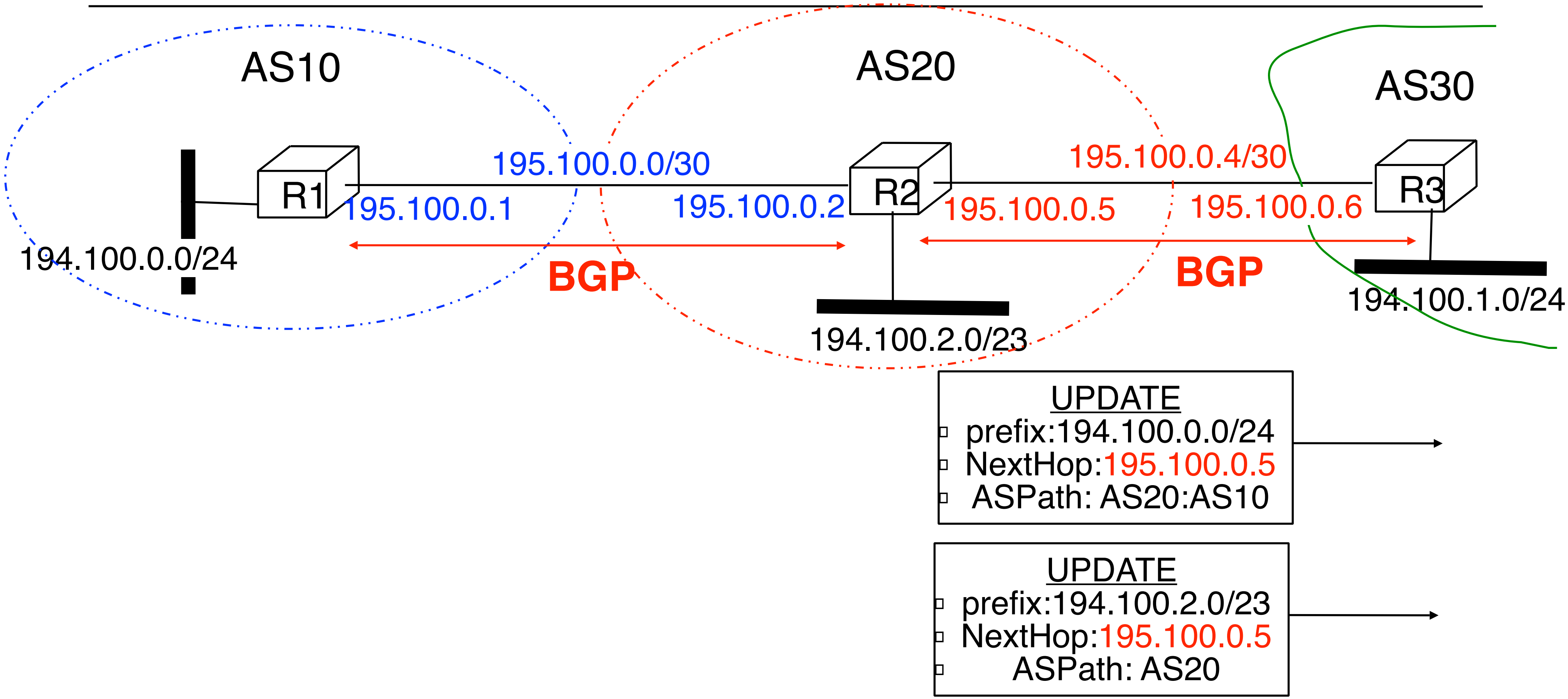
BGP and IP
A second example (2)



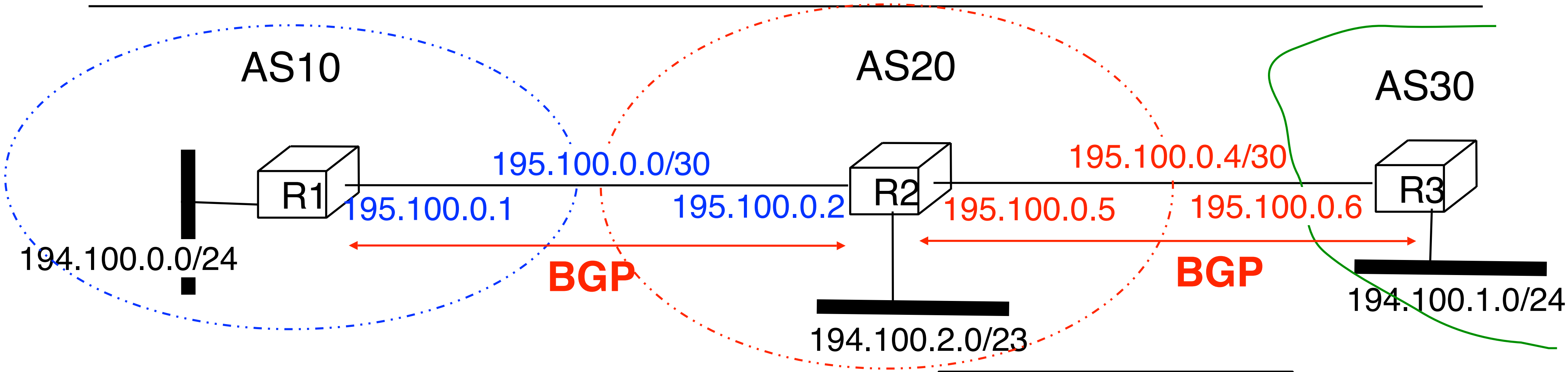
BGP and IP
A second example (2)



BGP and IP
A second example (2)



BGP and IP
A second example (2)

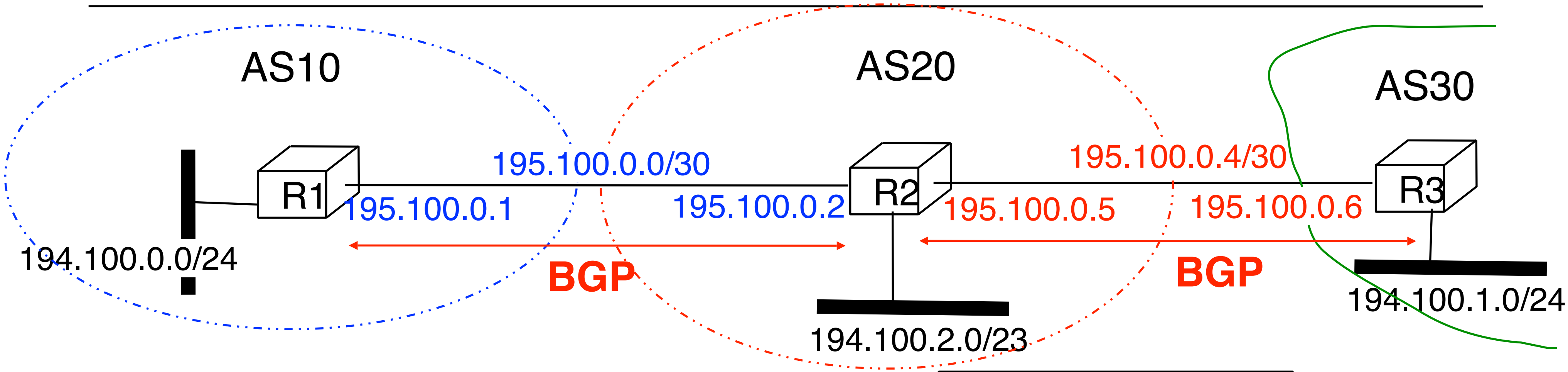


UPDATE
- prefix:194.100.0.0/24
- NextHop:195.100.0.5
- ASPath: AS20:AS10

UPDATE
- prefix:194.100.2.0/23
- NextHop:195.100.0.5
- ASPath: AS20

UPDATE
- prefix:194.100.1.0/24,
- NextHop:195.100.0.6
- ASPath: AS30

BGP and IP
A second example (2)



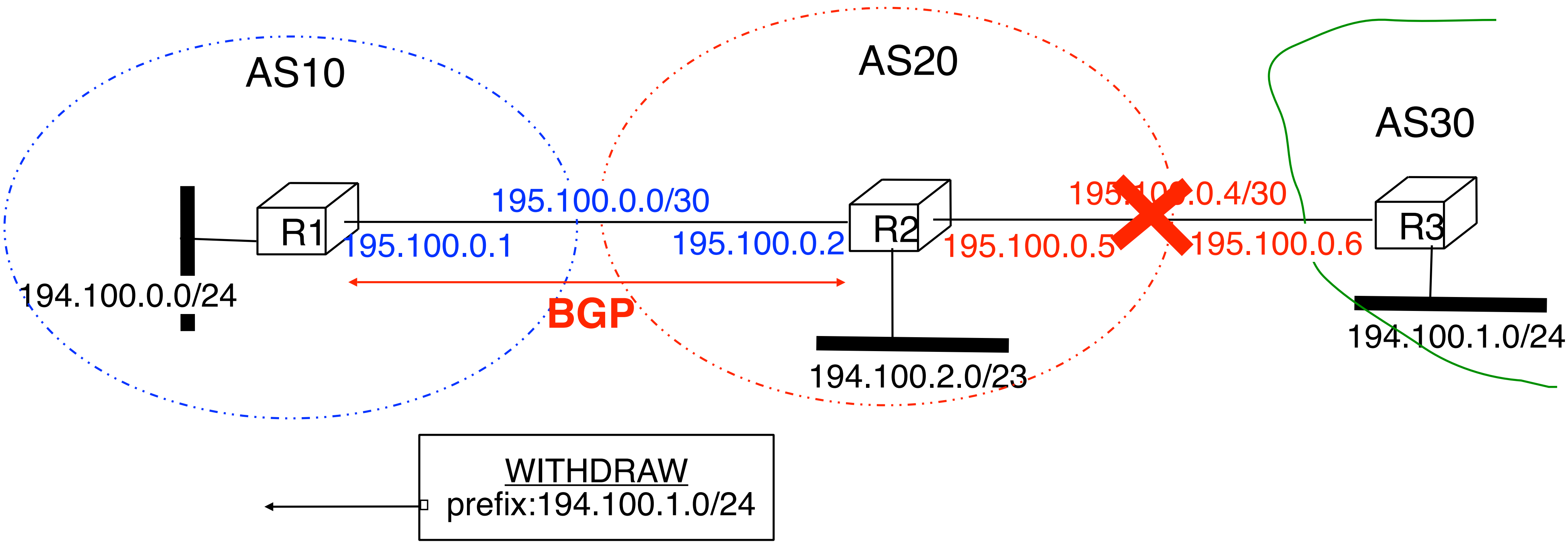
UPDATE
- prefix:194.100.0.0/24
- NextHop:195.100.0.5
- ASPath: AS20:AS10

UPDATE
- prefix:194.100.2.0/23
- NextHop:195.100.0.5
- ASPath: AS20

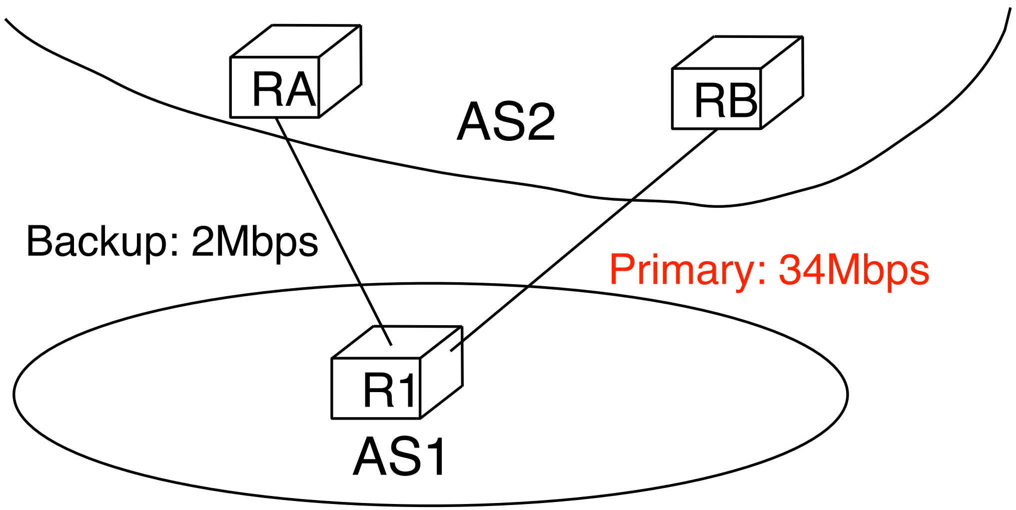
UPDATE
- prefix:194.100.1.0/24,
- NextHop:195.100.0.2
- ASPath: AS20;AS30

UPDATE
- prefix:194.100.1.0/24,
- NextHop:195.100.0.6
- ASPath: AS30

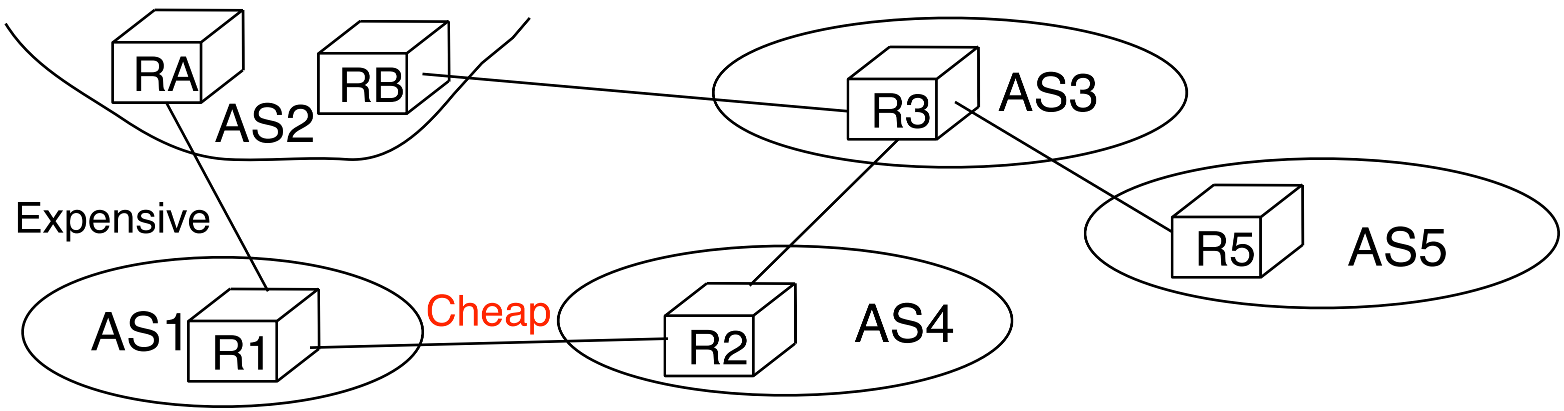
BGP and IP
A second example (3)



How to prefer some routes over others ?

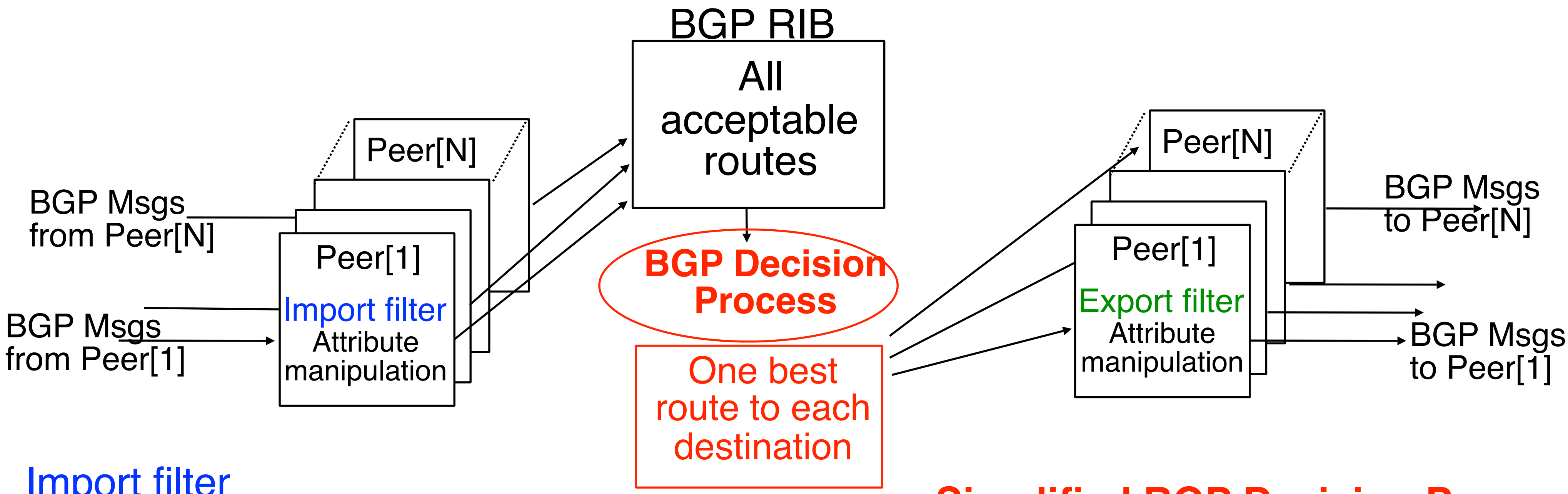


- How to ensure that packets will flow on primary link ?



- How to prefer cheap link over expensive link ?

How to prefer some routes over others (2) ?



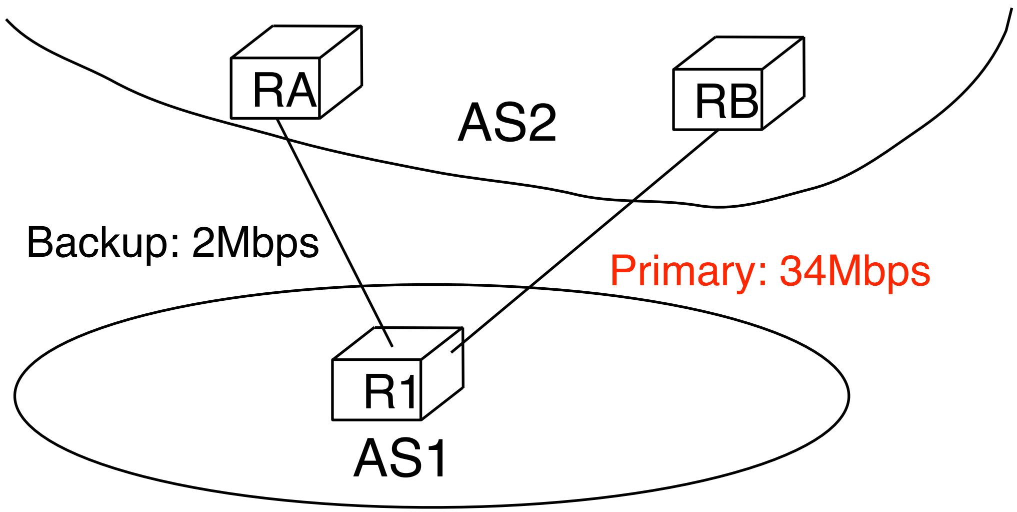
Import filter

- Selection of acceptable routes
- Addition of `local-pref` attribute inside received BGP Msg
  - Normal quality route : `local-pref`=100
  - Better than normal route :`local-pref`=200
  - Worse than normal route :`local-pref`=50

Simplified BGP Decision Process

- Select routes with highest `local-pref`
- If there are several routes, choose routes with the shortest ASPath
- If there are still several routes tie-breaking rule

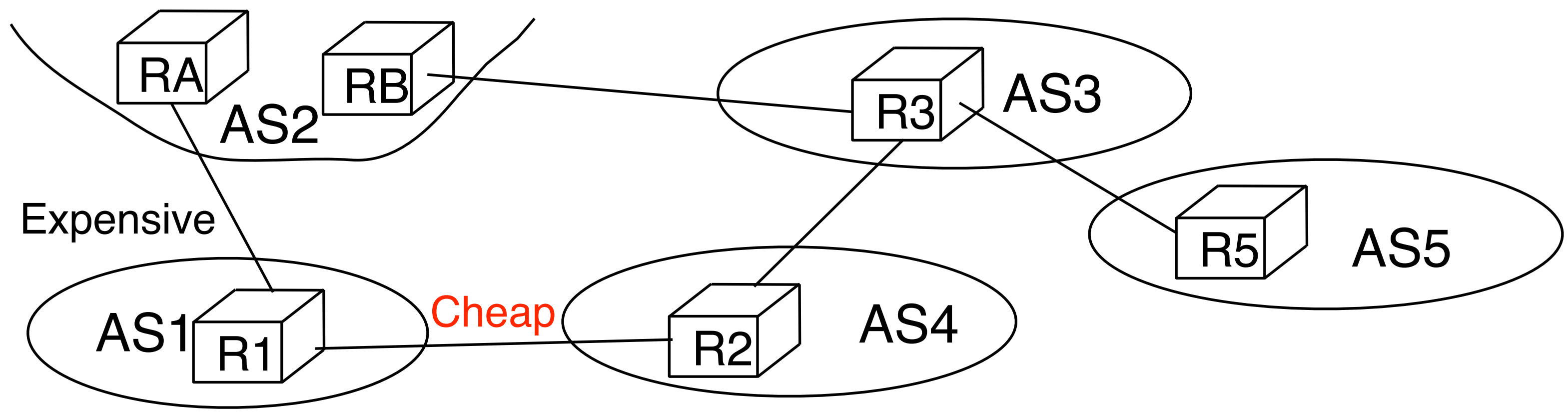
How to prefer some routes over others (3) ?



RPSL-like policy for AS1
aut-num: AS1
import: from AS2 RA at R1 set localpref=100;
from AS2 RB at R1 set localpref=200;
accept ANY
export: to AS2 RA at R1 announce AS1
to AS2 RB at R1 announce AS1

RPSL-like policy for AS2
aut-num: AS2
import: from AS1 R1 at RA set localpref=100;
from AS1 R1 at RB set localpref=200;
accept AS1
export: to AS1 R1 at RA announce ANY
to AS2 R1 at RB announce ANY

How to prefer some routes over others (4) ?

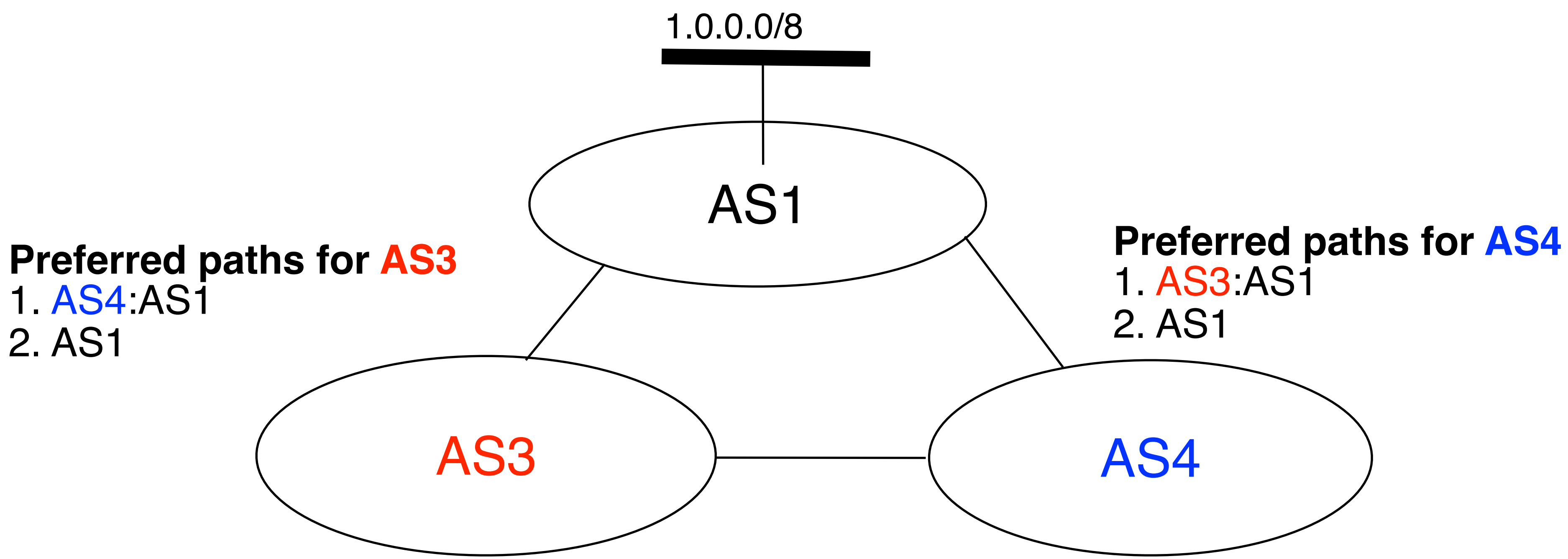


RPSL policy for AS1
aut-num: AS1
import: from AS2 RA at R1 set localpref=100;
from AS4 R2 at R1 set localpref=200;
accept ANY
export: to AS2 RA at R1 announce AS1
to AS4 R2 at R1 announce AS1

- AS1 will prefer to send packets over the cheap link
- But the flow of the packets destined to AS1 will depend on the routing policy of the other domains

Limitations of `local-pref`

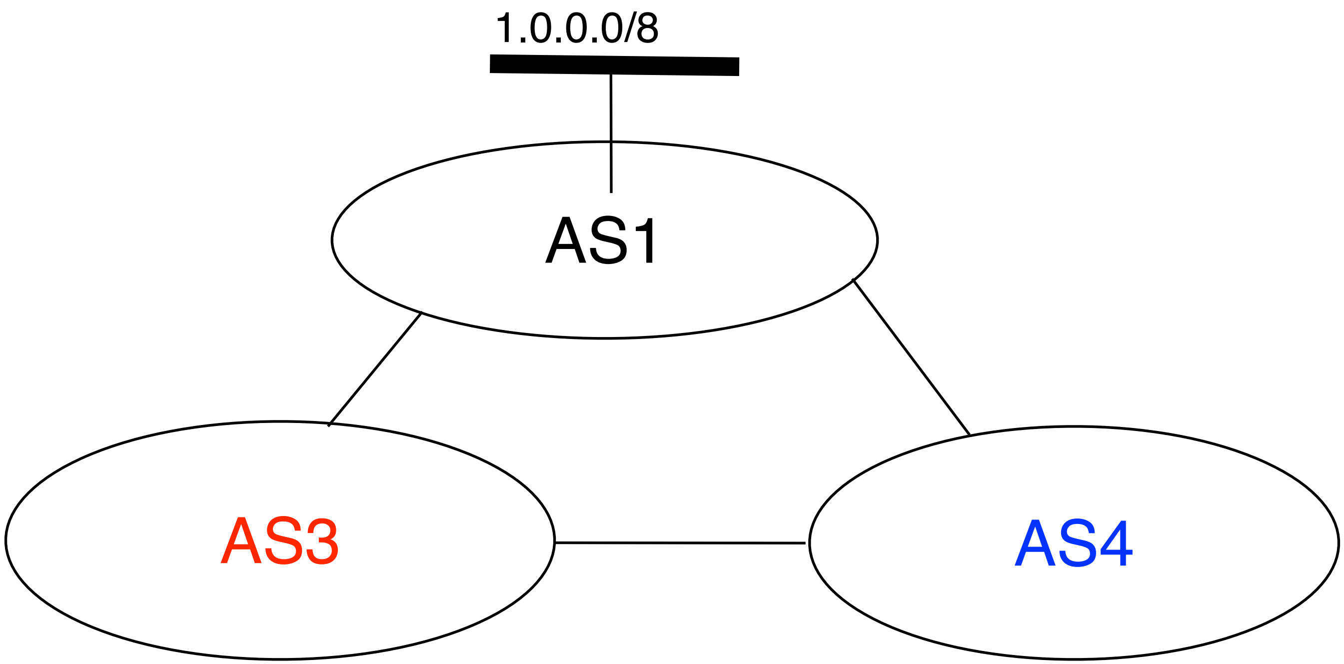
- In theory
  - Each domain is free to define its order of preference for the routes learned from external peers



- How to reach 1.0.0.0/8 from AS3 and AS4 ?

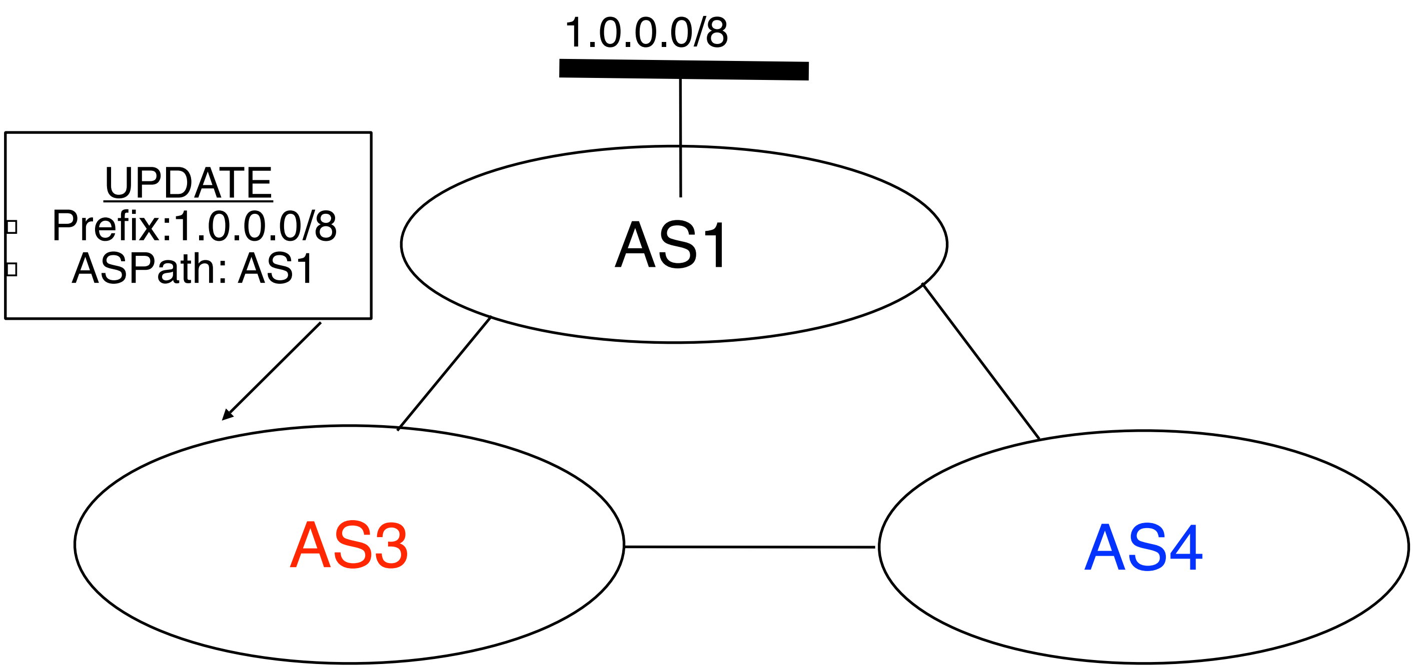
Limitations of `local-pref` (2)

- AS1 sends its UPDATE messages ...



Limitations of `local-pref` (2)

- AS1 sends its UPDATE messages ...

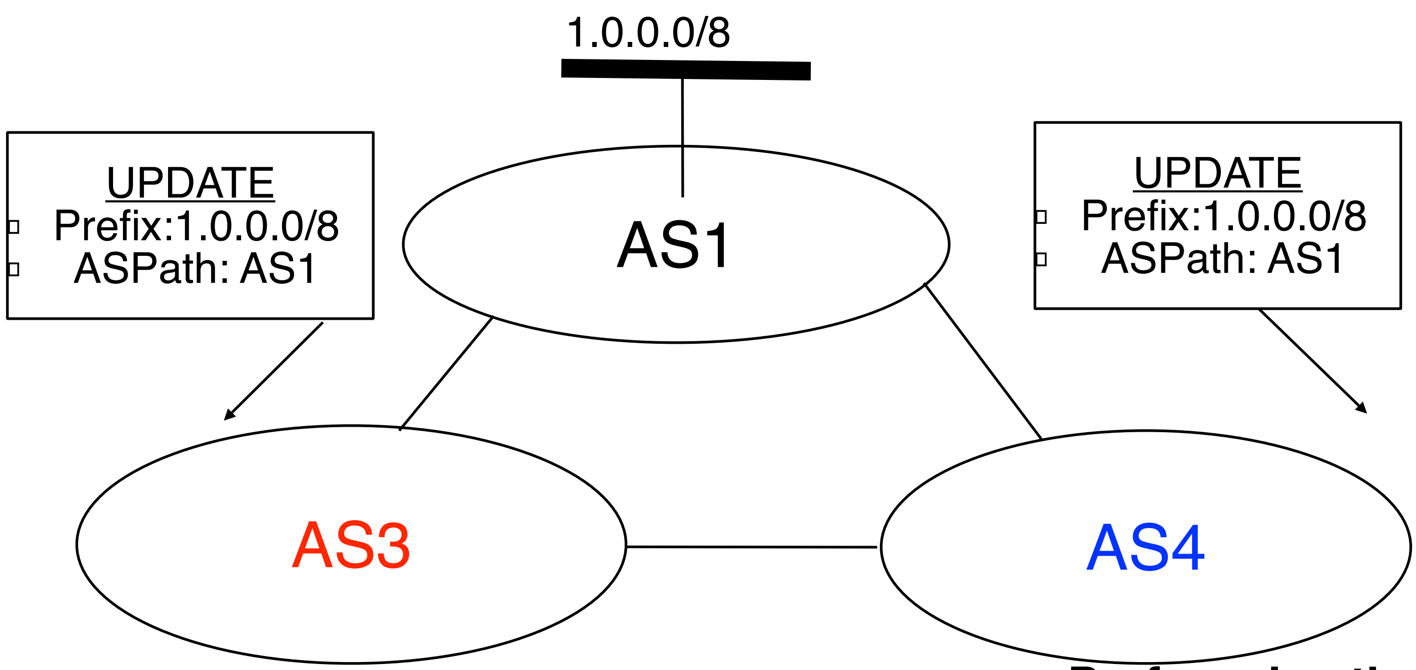


Preferred paths for AS3
1. AS4:AS1
2. AS1

Routing table for AS3
1.0.0.0/8 ASPath: AS1 (best)

Limitations of `local-pref` (2)

- AS1 sends its UPDATE messages ...



Preferred paths for AS3
1. AS4:AS1
2. AS1

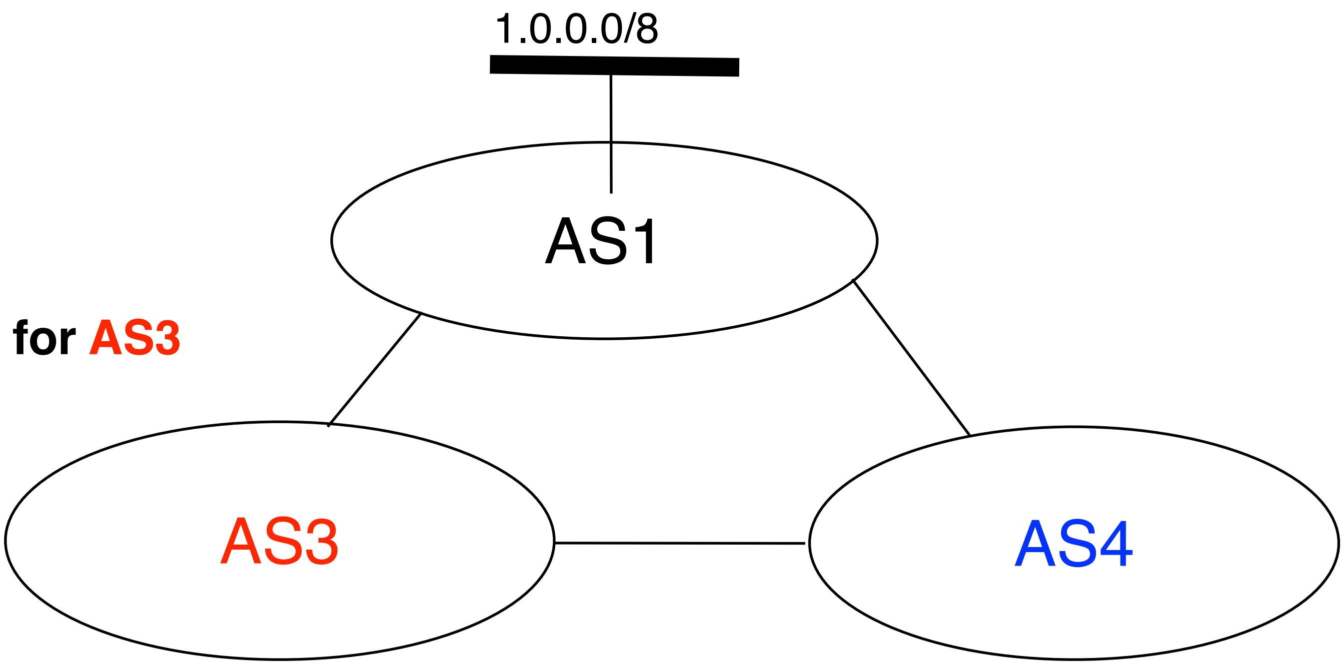
Preferred paths for AS4
1. AS3:AS1
2. AS1

Routing table for AS3
1.0.0.0/8 ASPath: AS1 (best)

Routing table for AS4
1.0.0.0/8 ASPath: AS1 (best)

Limitations of `local-pref` (3)

- First possibility
  - AS3 sends its UPDATE first...

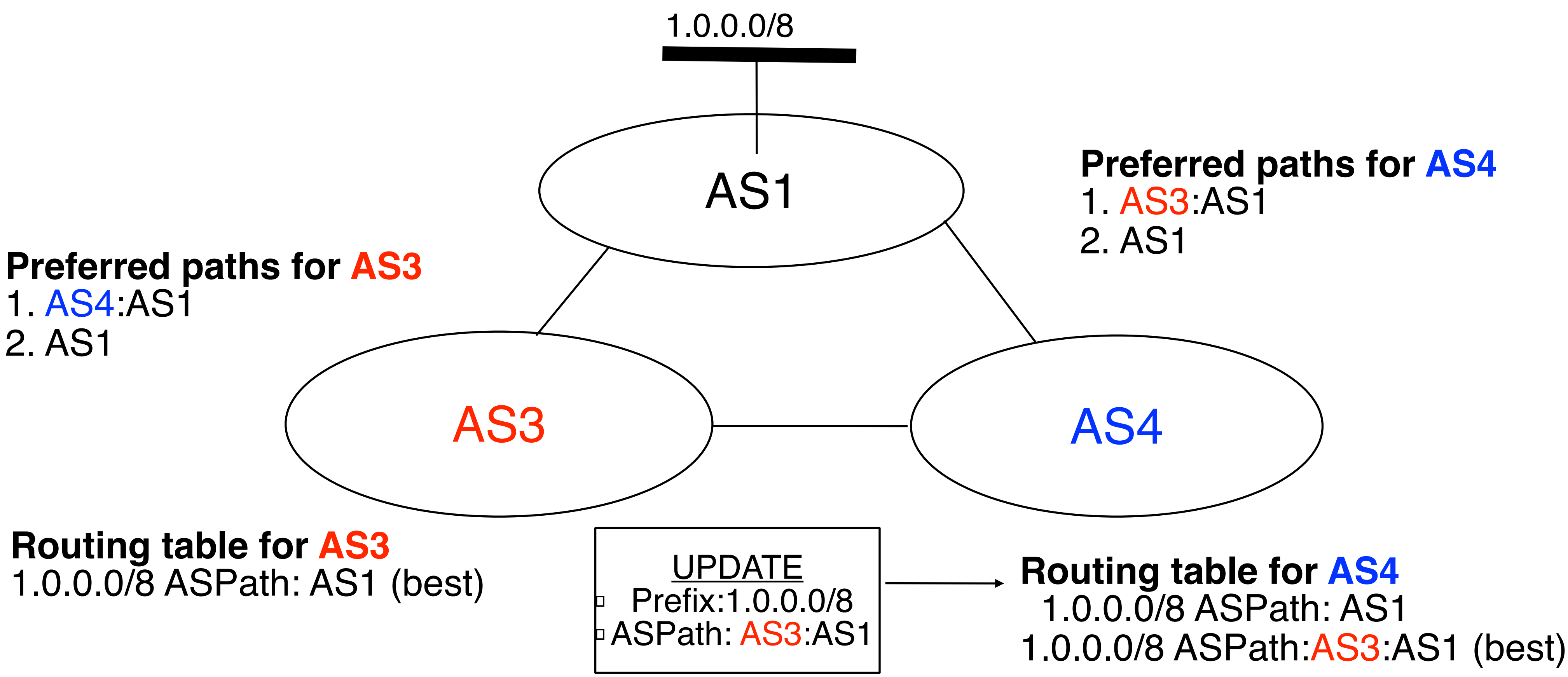


Preferred paths for AS3
1. AS4:AS1
2. AS1

Routing table for AS3
1.0.0.0/8 ASPath: AS1 (best)

Limitations of `local-pref` (3)

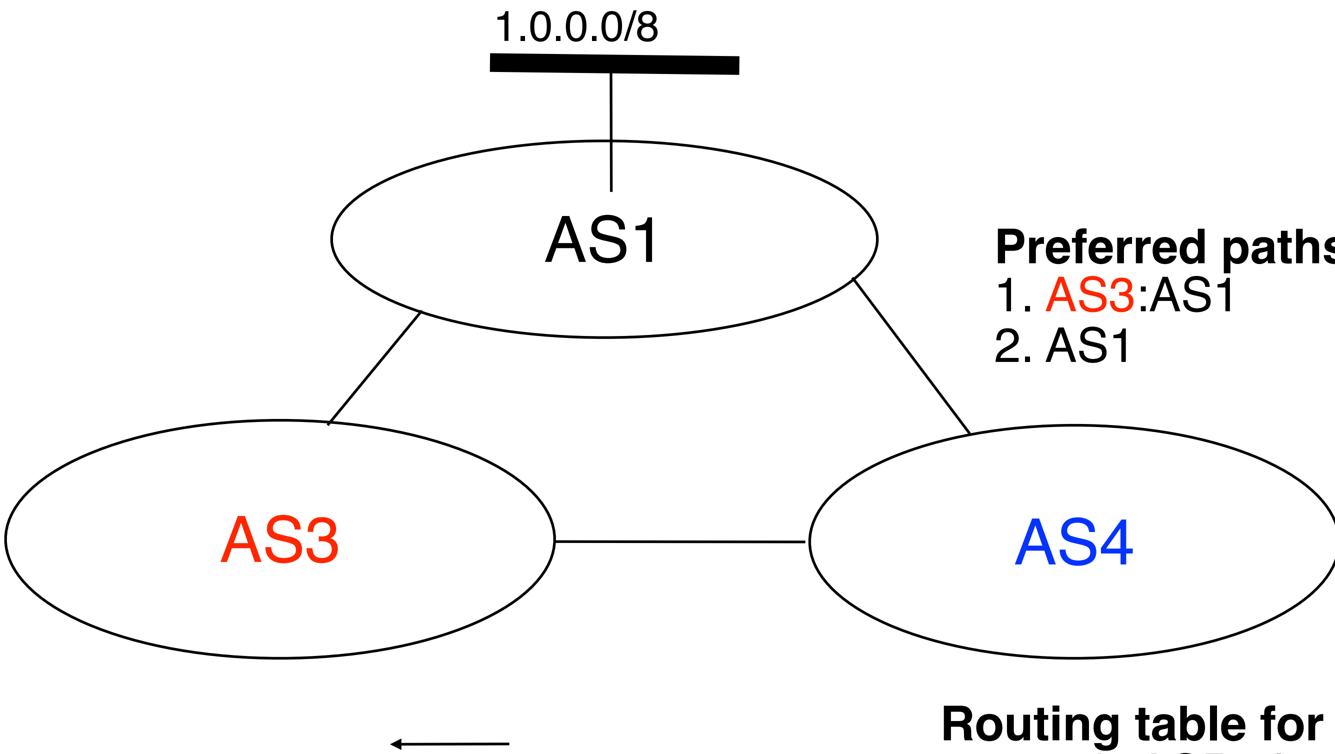
- First possibility
  - AS3 sends its UPDATE first...



- Stable route assignment

Limitations of `local-pref` (4)

- Second possibility
  - AS4 sends its UPDATE first...

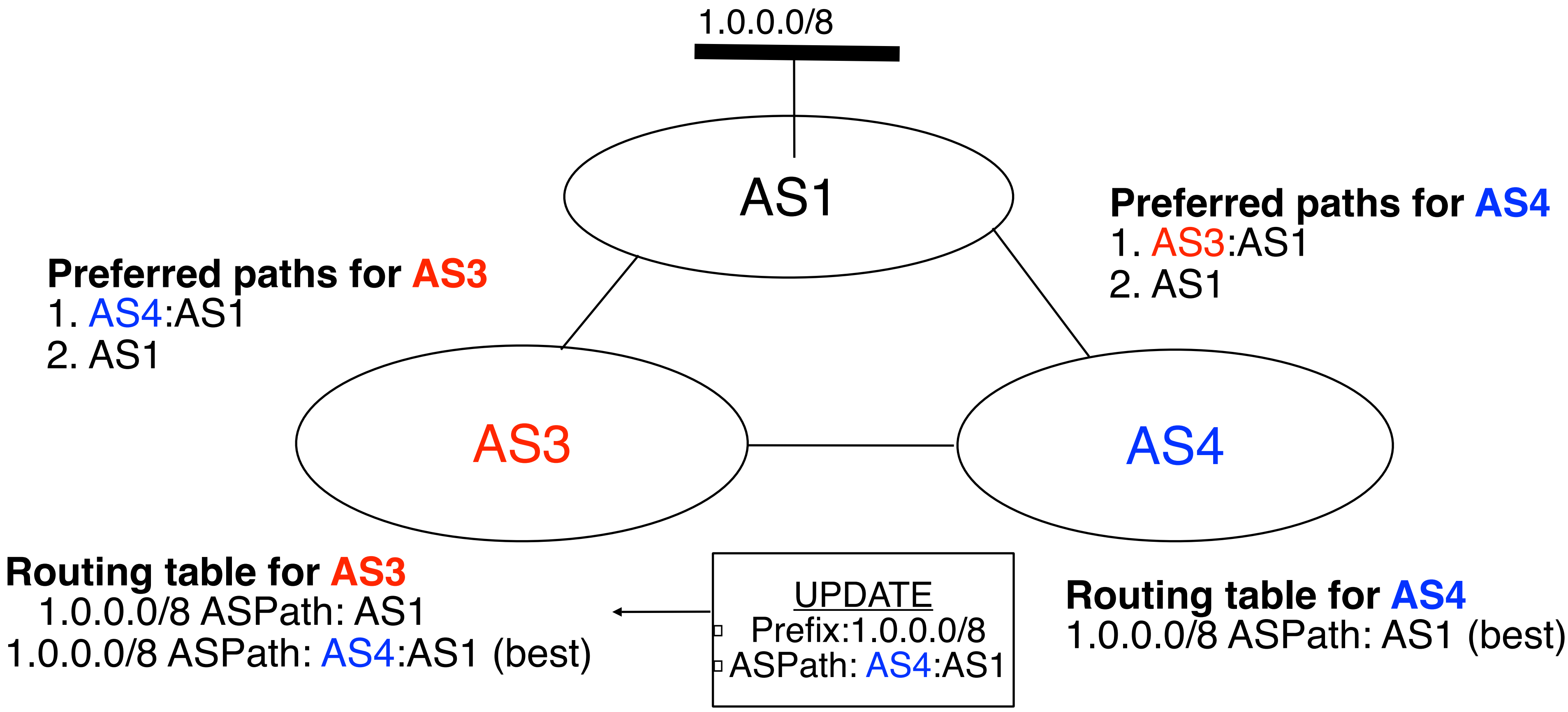


Preferred paths for AS4
1. AS3:AS1
2. AS1

Routing table for AS4
1.0.0.0/8 ASPath: AS1 (best)

Limitations of `local-pref` (4)

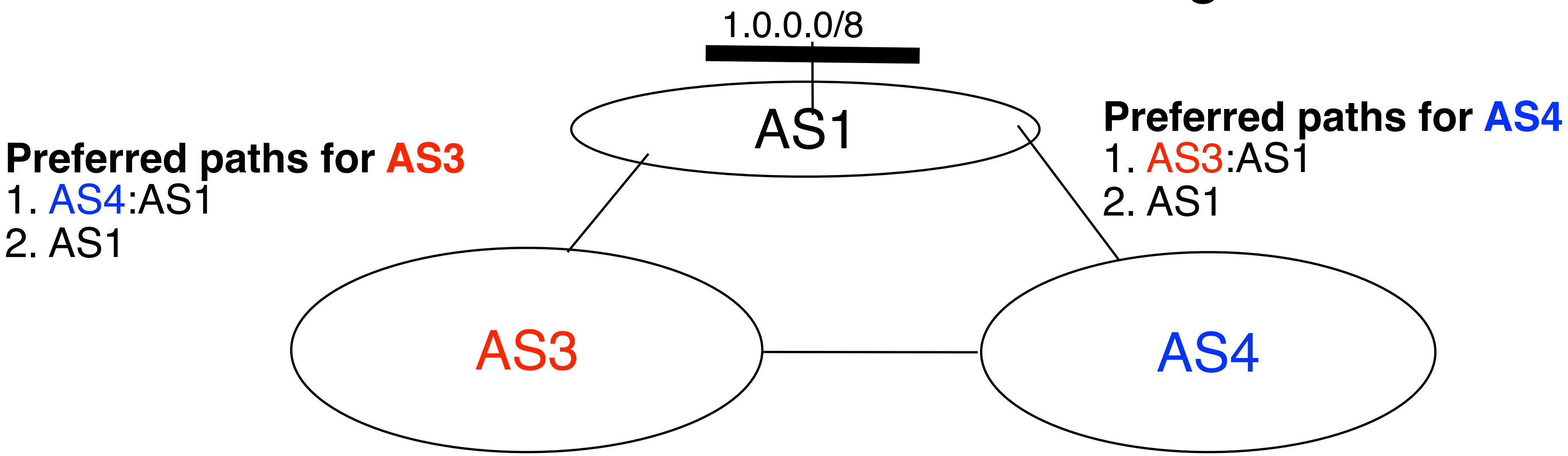
- Second possibility
  - AS4 sends its UPDATE first...



- Another (but different) stable route assignment

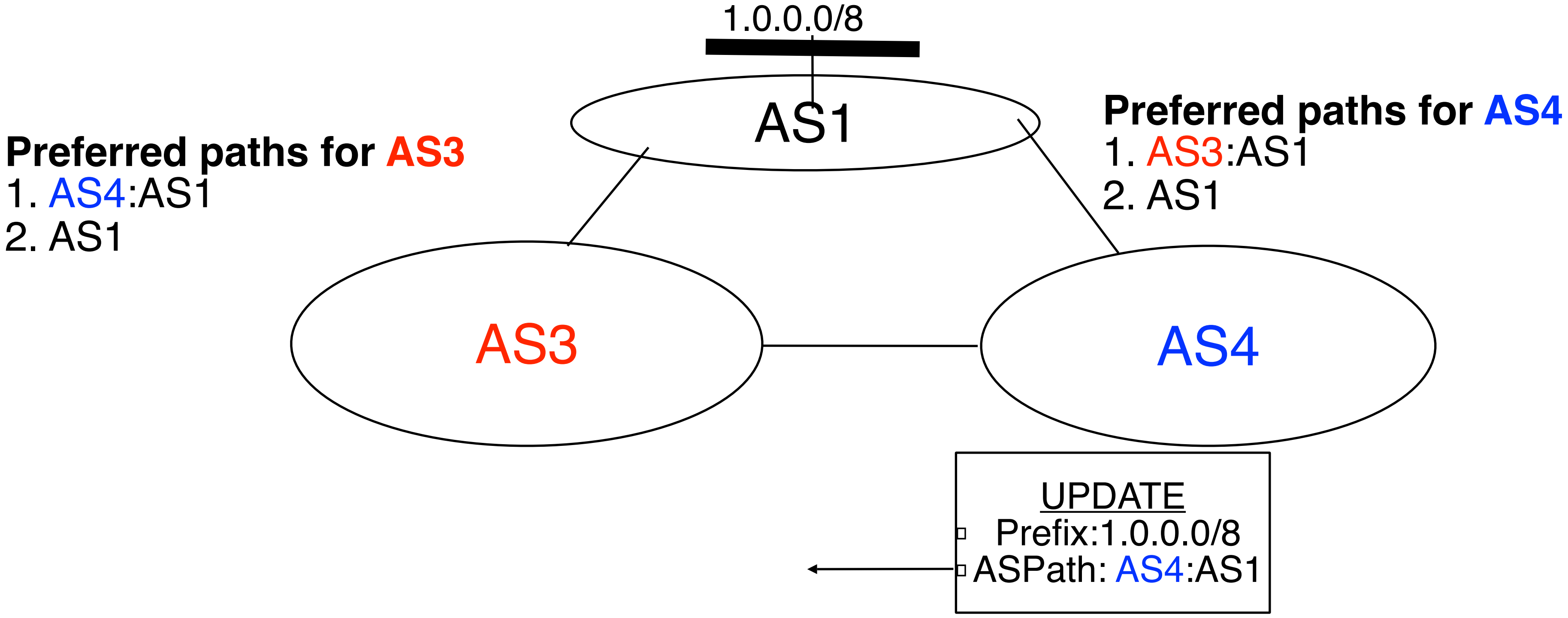
Limitations of `local-pref` (5)

- Third possibility
  - AS3 and AS4 send their UPDATE together...



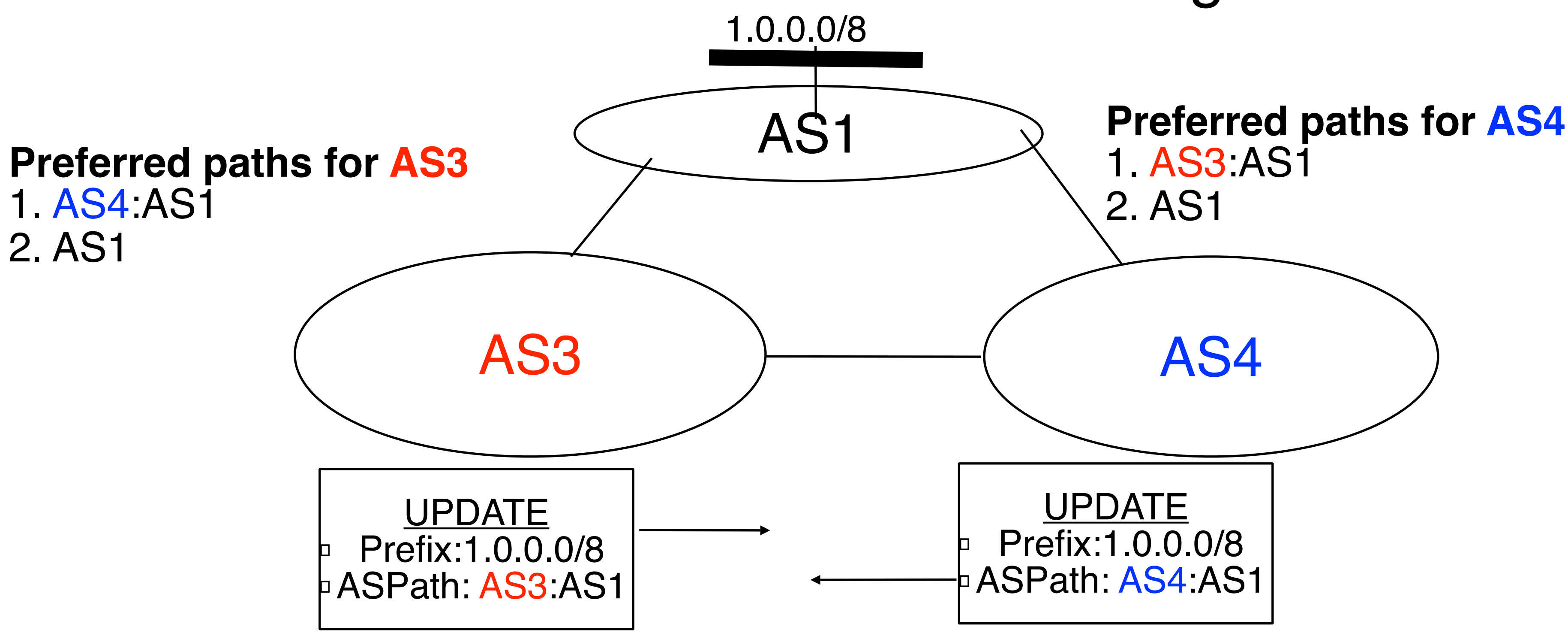
Limitations of `local-pref` (5)

- Third possibility
  - AS3 and AS4 send their UPDATE together...



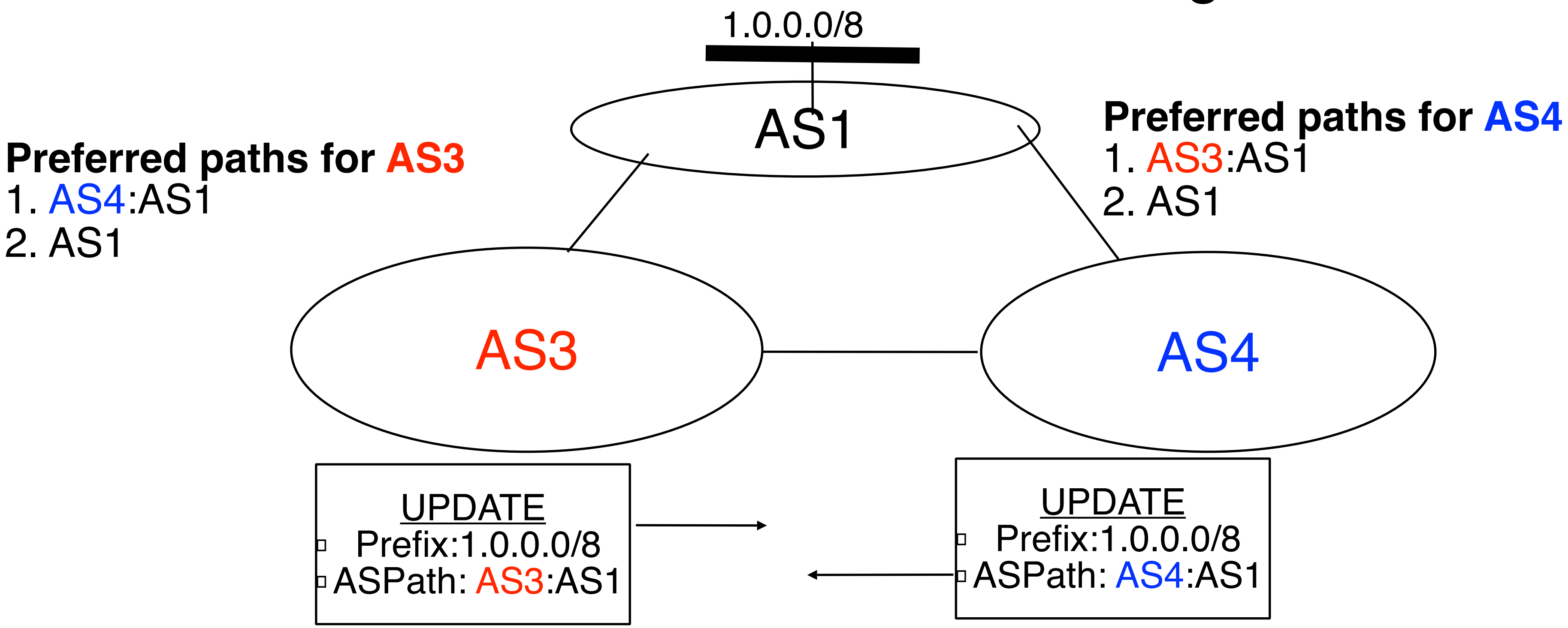
Limitations of `local-pref` (5)

- Third possibility
  - AS3 and AS4 send their UPDATE together...



Limitations of `local-pref` (5)

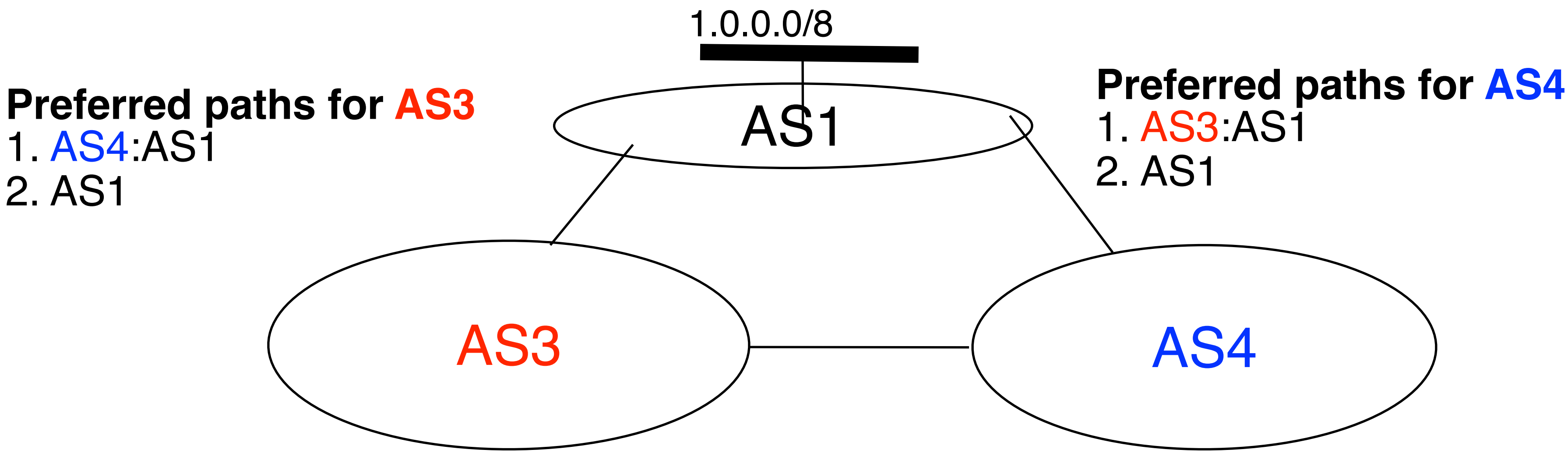
- Third possibility
  - AS3 and AS4 send their UPDATE together...



- AS3 prefers the indirect path and will thus send withdraw since the chosen best path is via AS4
- AS4 prefers the indirect path and will thus send withdraw since the chosen best path is via AS3

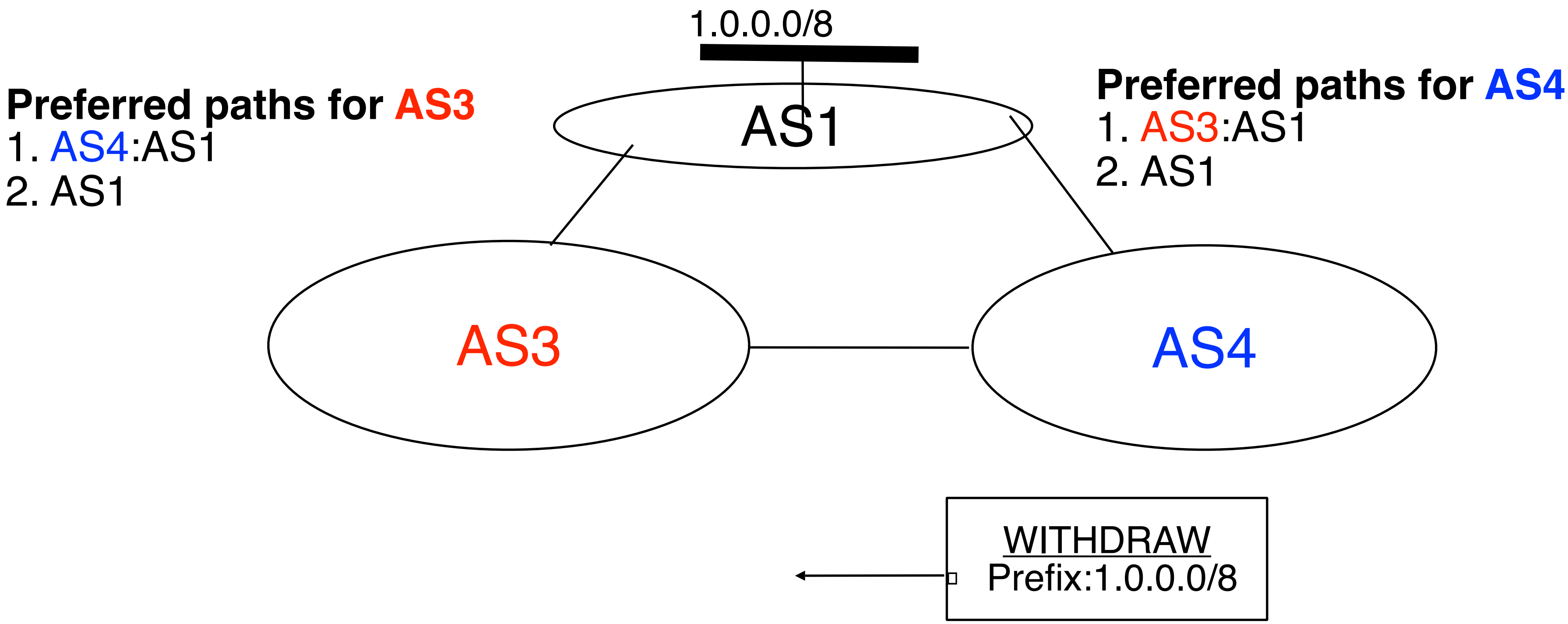
Limitations of `local-pref` (6)

- Third possibility (cont.)
  - AS3 and AS4 send their UPDATE together...



Limitations of `local-pref` (6)

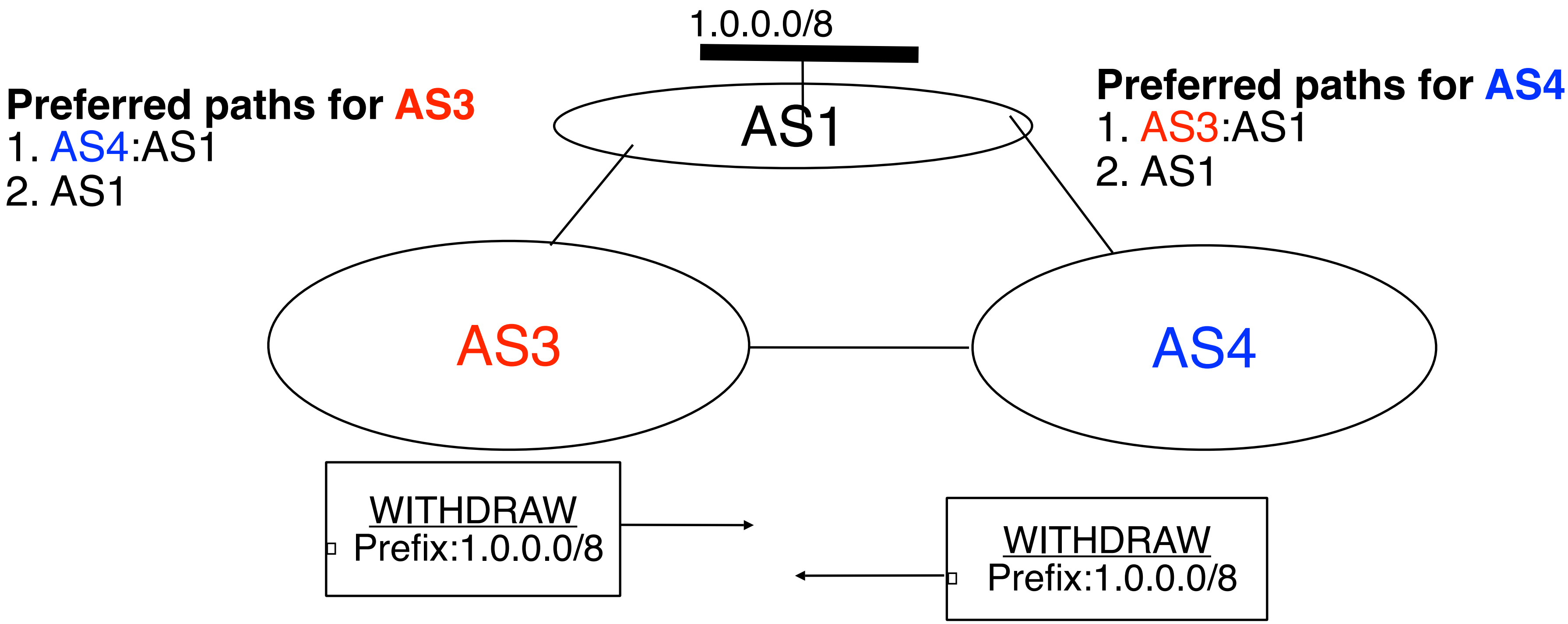
- Third possibility (cont.)
  - AS3 and AS4 send their UPDATE together...



- AS3 learns that the indirect route is not available anymore
  - AS3 will reannounce its direct route...

Limitations of `local-pref` (6)

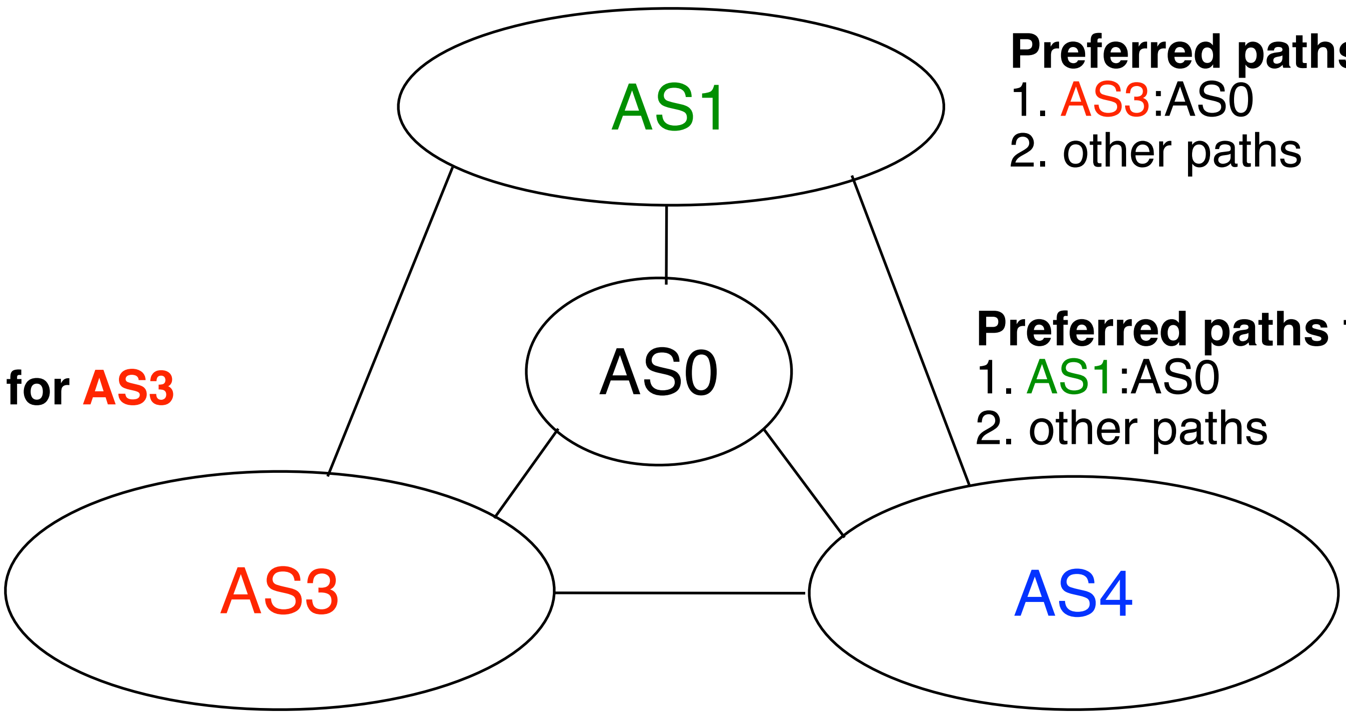
- Third possibility (cont.)
  - AS3 and AS4 send their UPDATE together...



- AS3 learns that the indirect route is not available anymore
  - AS3 will reannounce its direct route...
- AS4 learns that the indirect route is not available anymore
  - AS4 will reannounce its direct route...

More limitations of `local-pref`

- Unfortunately, interdomain routing may not converge at all in some cases...



Preferred paths for AS1
1. AS3:AS0
2. other paths

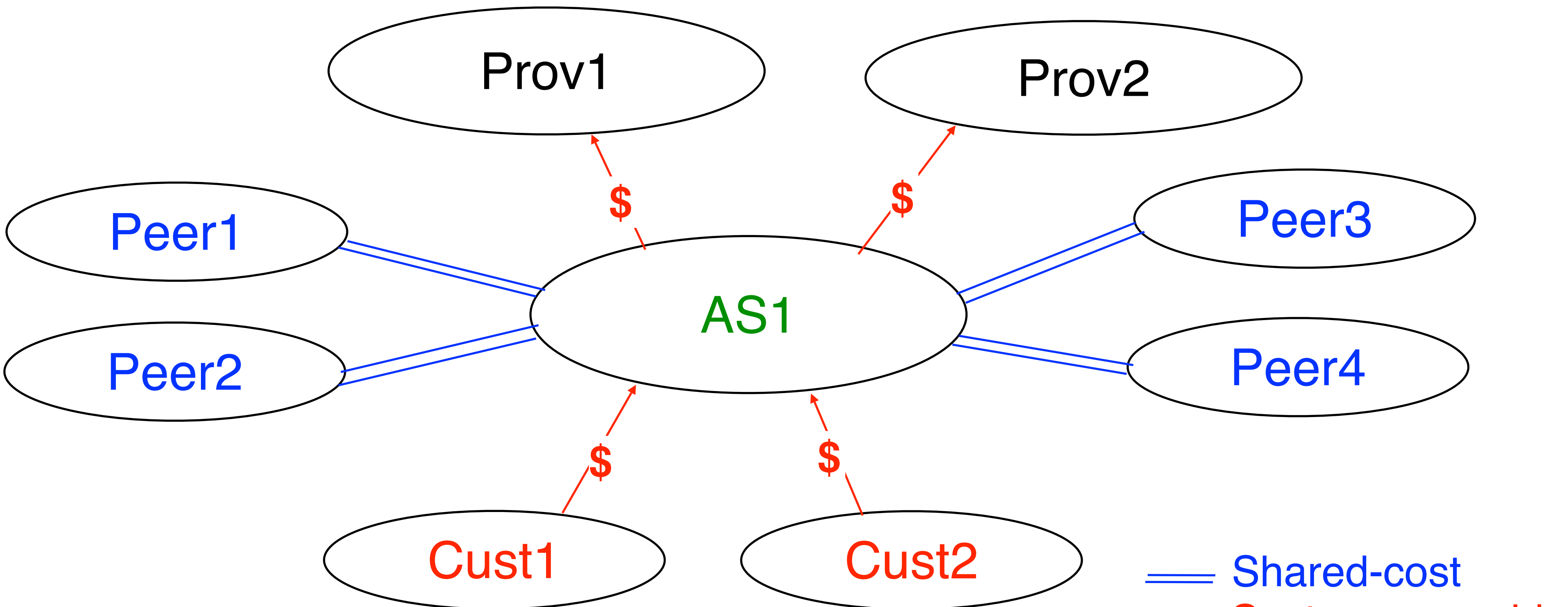
Preferred paths for AS4
1. AS1:AS0
2. other paths

Preferred paths for AS3
1. AS4:AS0
2. other paths

- How to reach a destination inside AS0 in this case ?

`local-pref` and economical relationships

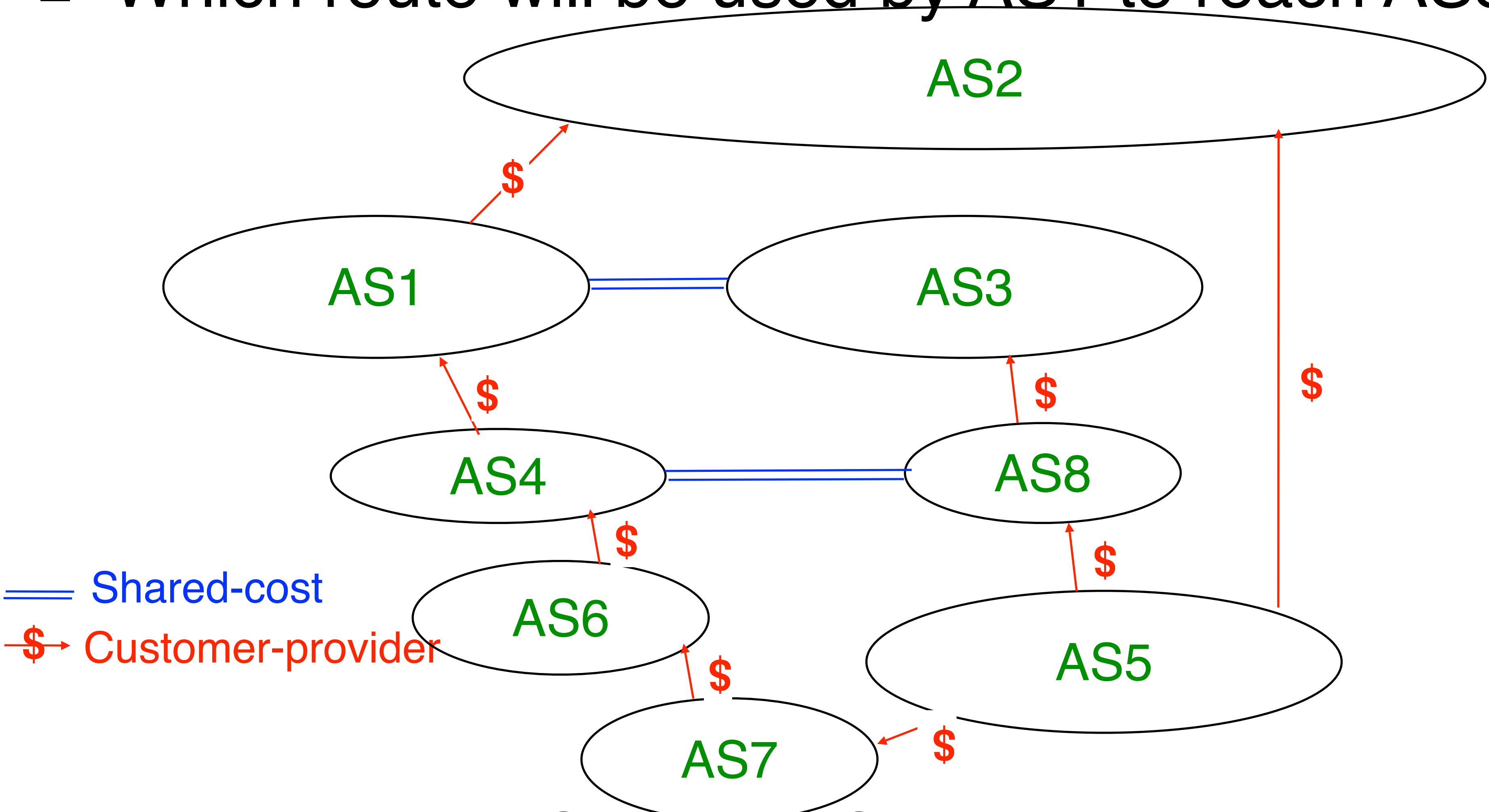
- In practice, `local-pref` is often used to enforce economical relationships



Local-pref values used by AS1
> 1000 for the routes received from a Customer
500 – 999 for the routes learned from a Peer
< 500 for the routes learned from a Provider

Consequence of this utilisation of `local-pref`

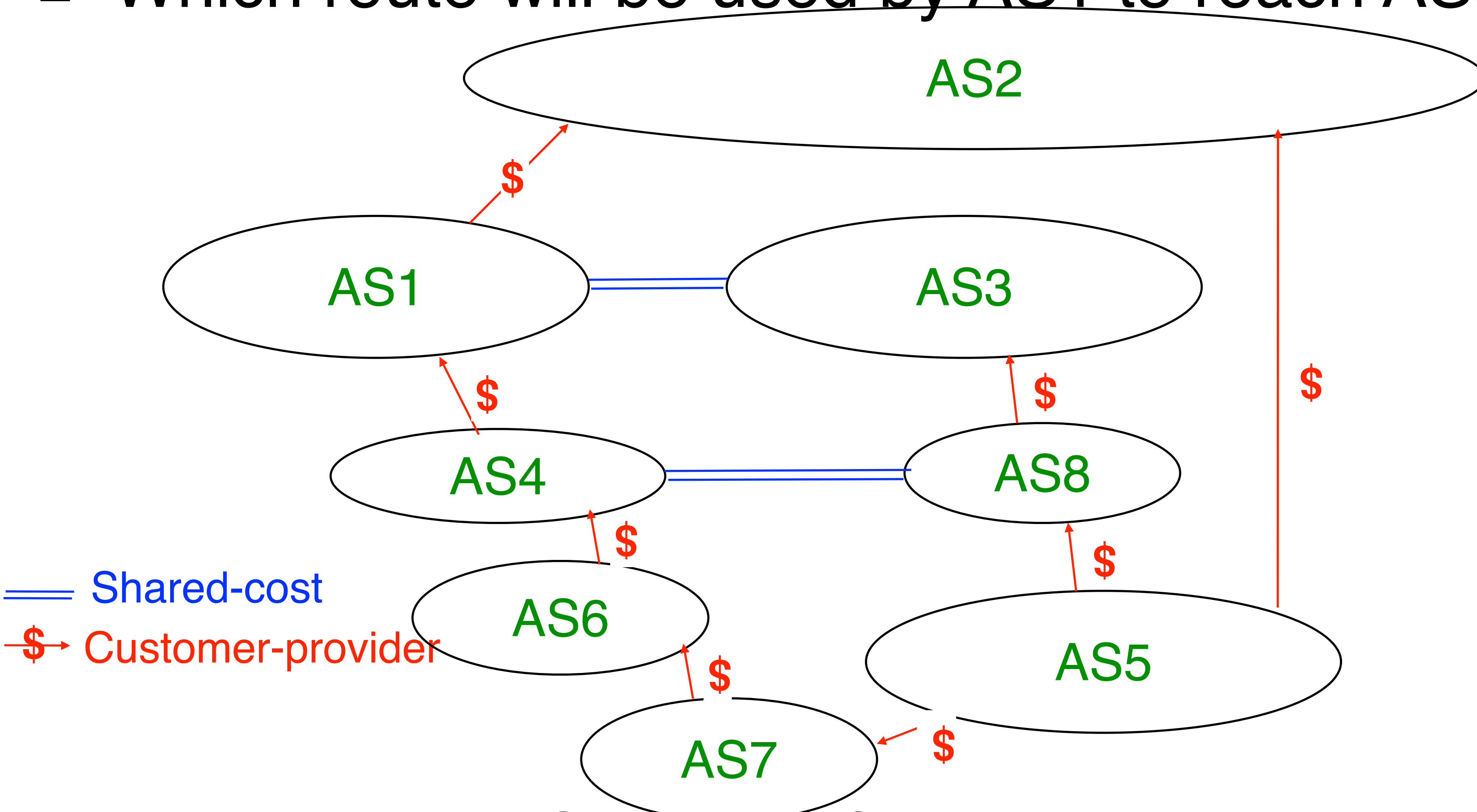
- Which route will be used by AS1 to reach AS5 ?



- and how will AS5 reach AS1 ?

Consequence of this utilisation of `local-pref`

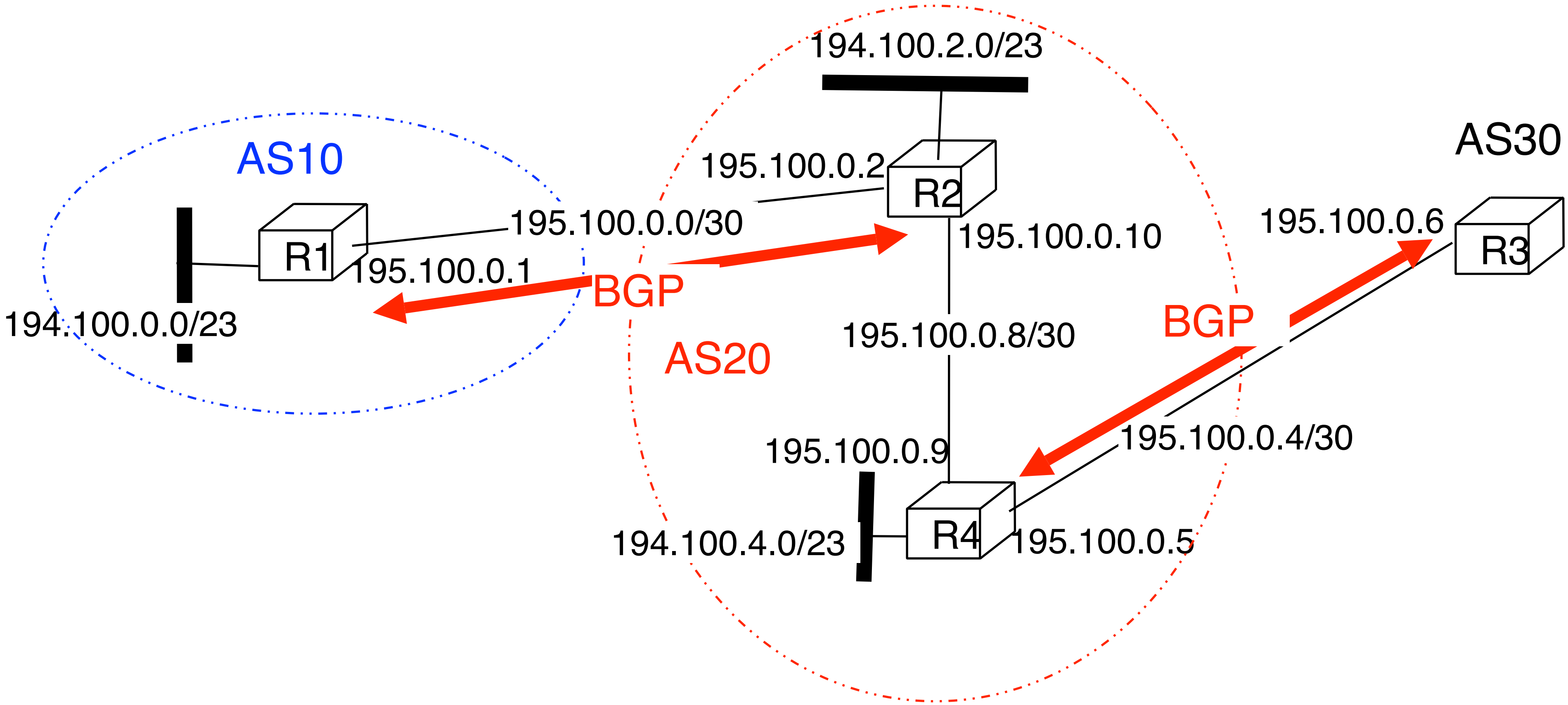
- Which route will be used by AS1 to reach AS5 ?



- and how will AS5 reach AS1 ?

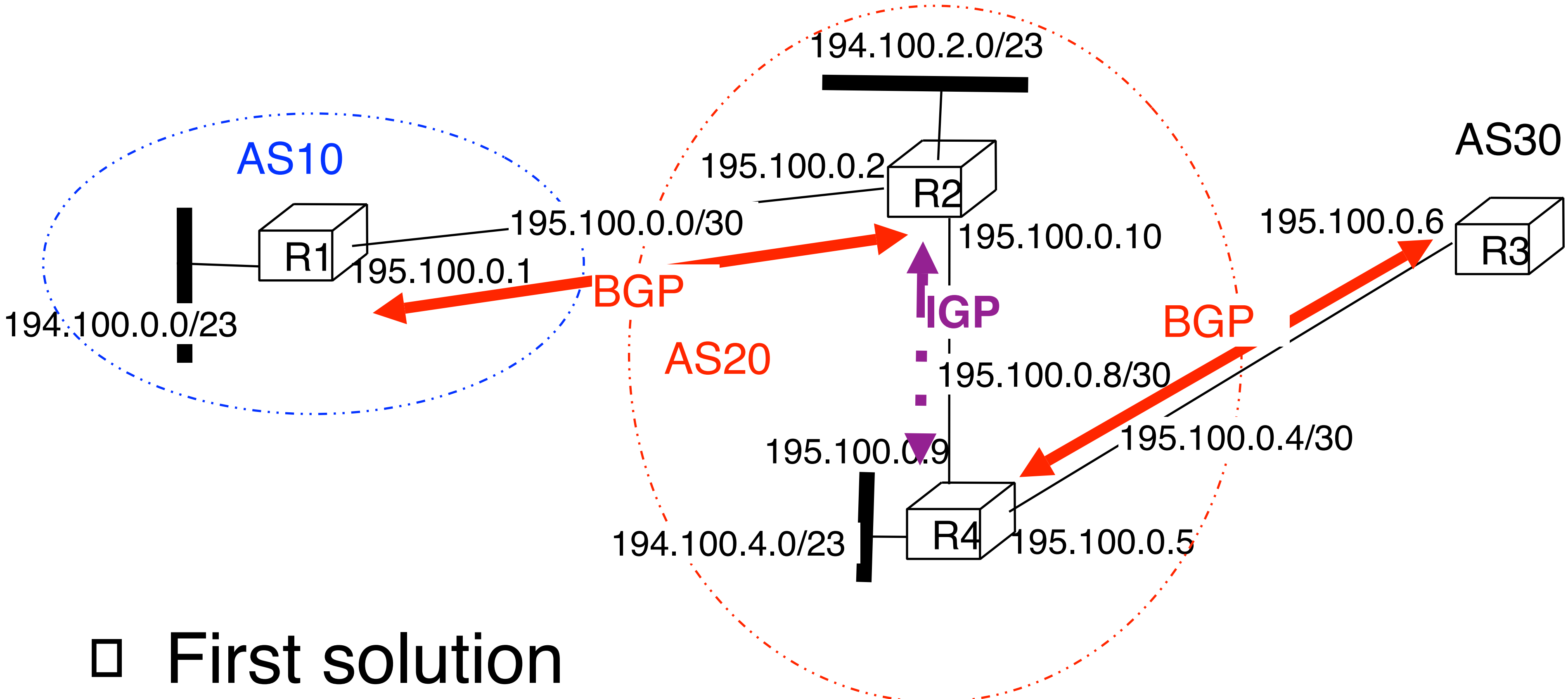
Internet paths are often asymmetrical

BGP and IP
Second example



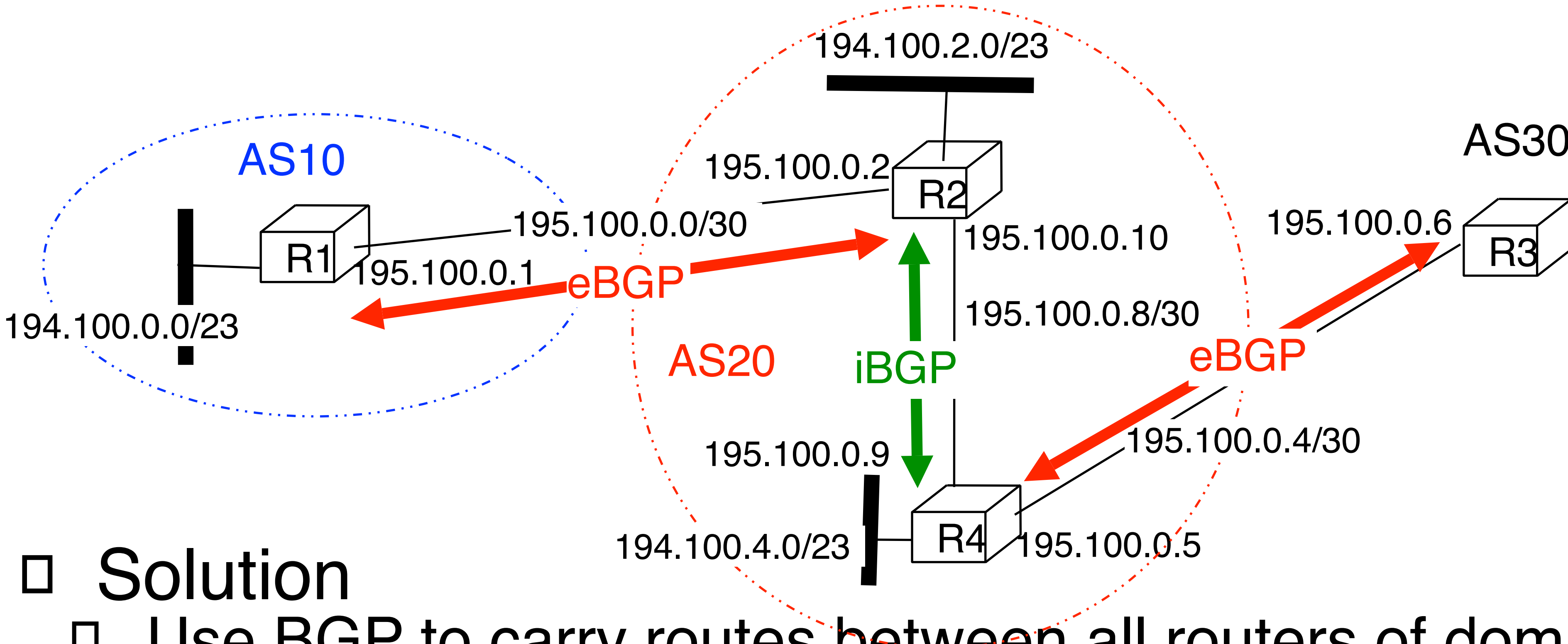
- Problem
  - How can R2 (resp. R4) advertise to R4 (resp. R2) the routes learned from AS10 (resp. AS30) ?

BGP and IP
Second example (2)



- First solution
  - Use IGP (OSPF/ISIS,RIP) to carry BGP routes
- Drawbacks
  - IGP may not be able to support so many routes
  - IGP does not carry BGP attributes like ASPath !

iBGP and eBGP

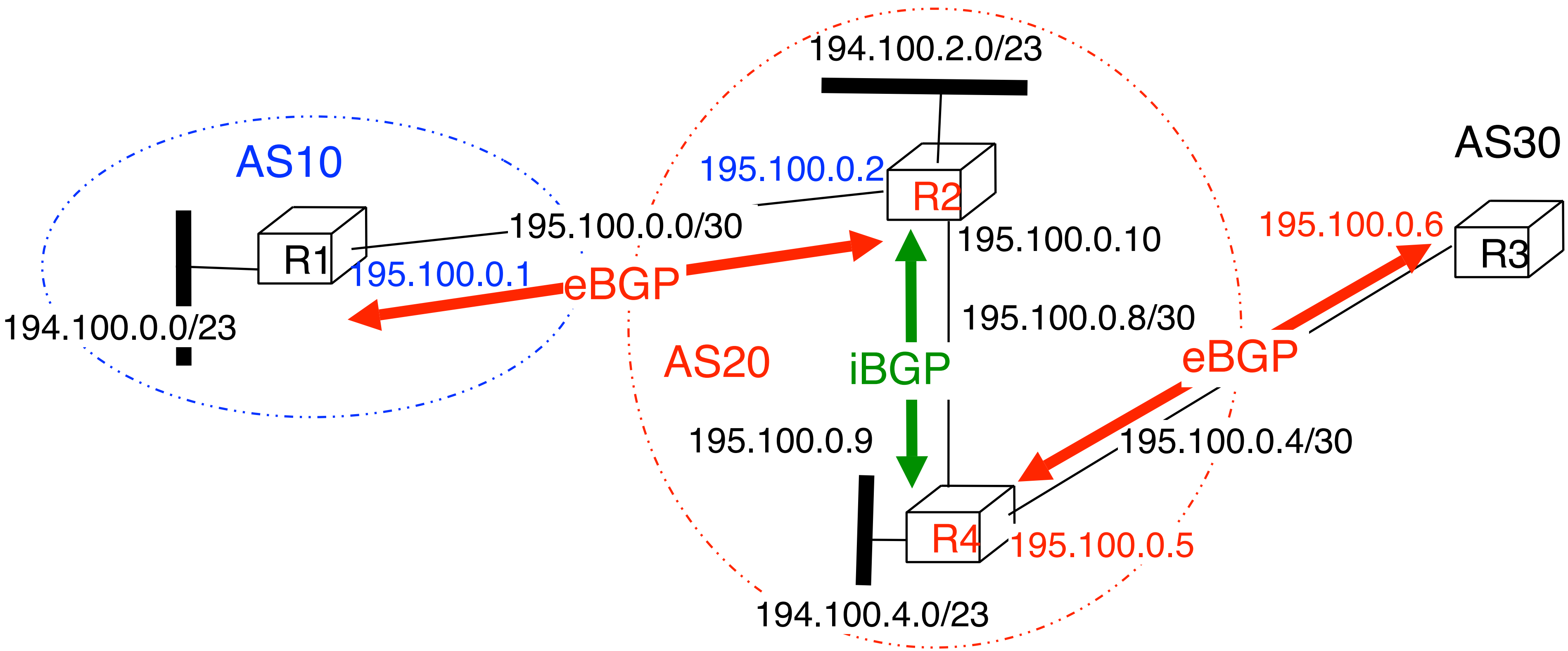


- Solution
  - Use BGP to carry routes between all routers of domain
    - Two different types of BGP sessions
    - eBGP between routers belonging to different ASes
    - iBGP between each pair of routers belonging to the same AS
      - Each BGP router inside ASx maintains an iBGP session with all other BGP routers of ASx (full iBGP mesh)
      - Note that the iBGP sessions do not necessarily follow physical topology

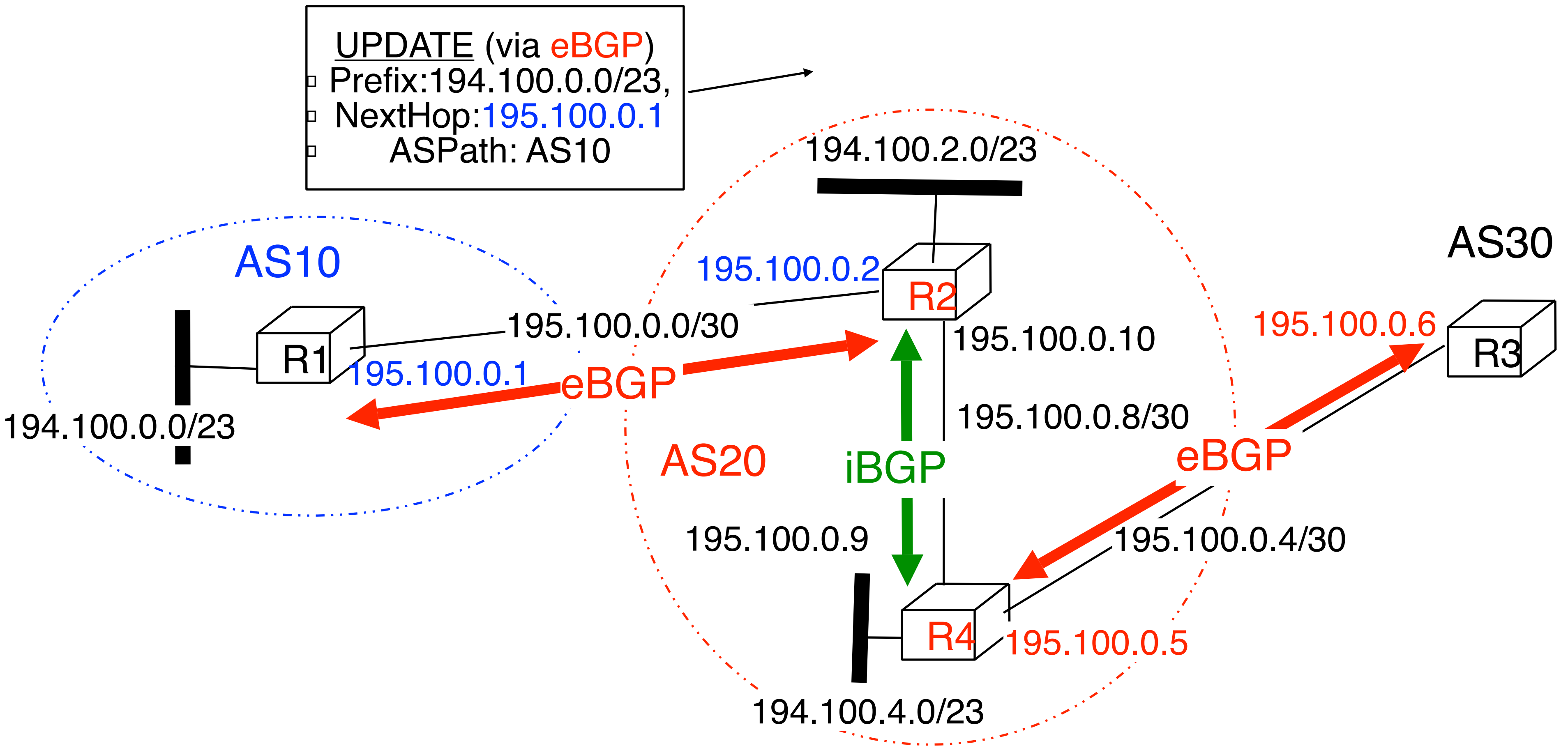
iBGP versus eBGP

- Differences between iBGP and eBGP
  - `local-pref` attribute is only carried inside messages sent over iBGP session
  - Over an eBGP session, a router only advertises its best route towards each destination
    - Usually, import and export filters are defined for each eBGP session
  - Over an iBGP session, a router advertises only its best routes learned over eBGP sessions
    - A route learned over an iBGP session is *never* advertised over another iBGP session
    - Usually, no filter is applied on iBGP sessions

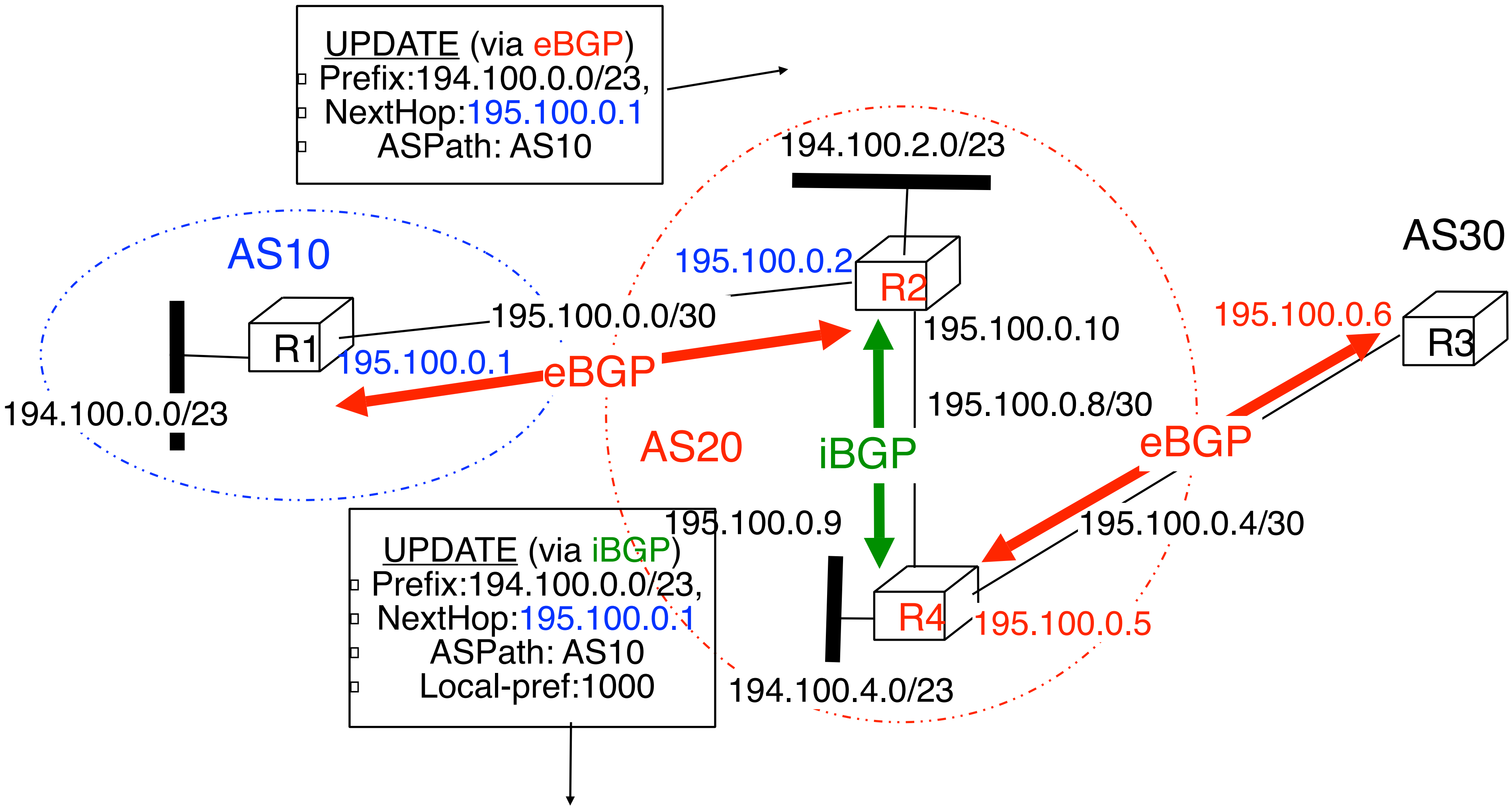
iBGP and eBGP : Example



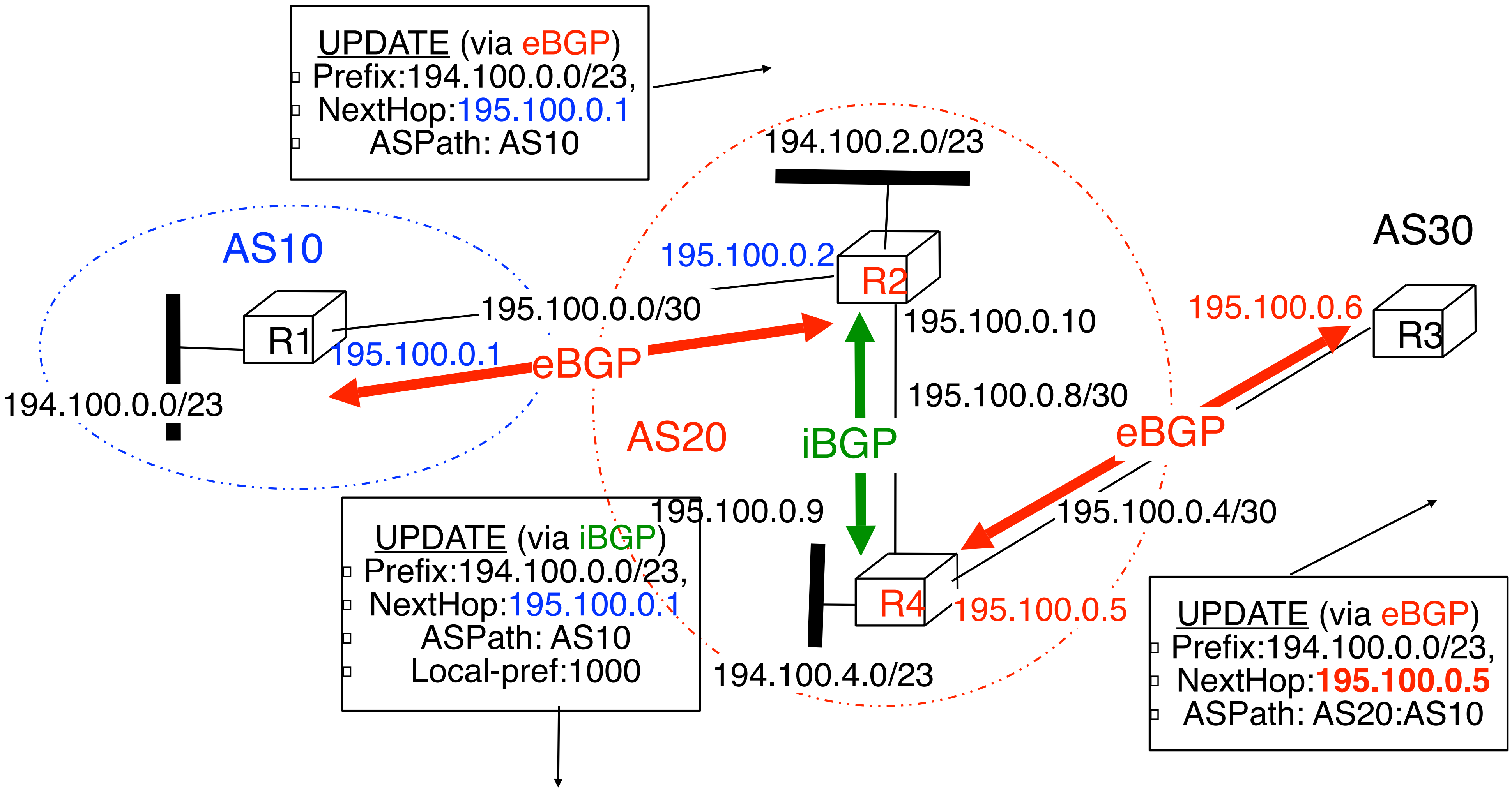
iBGP and eBGP : Example



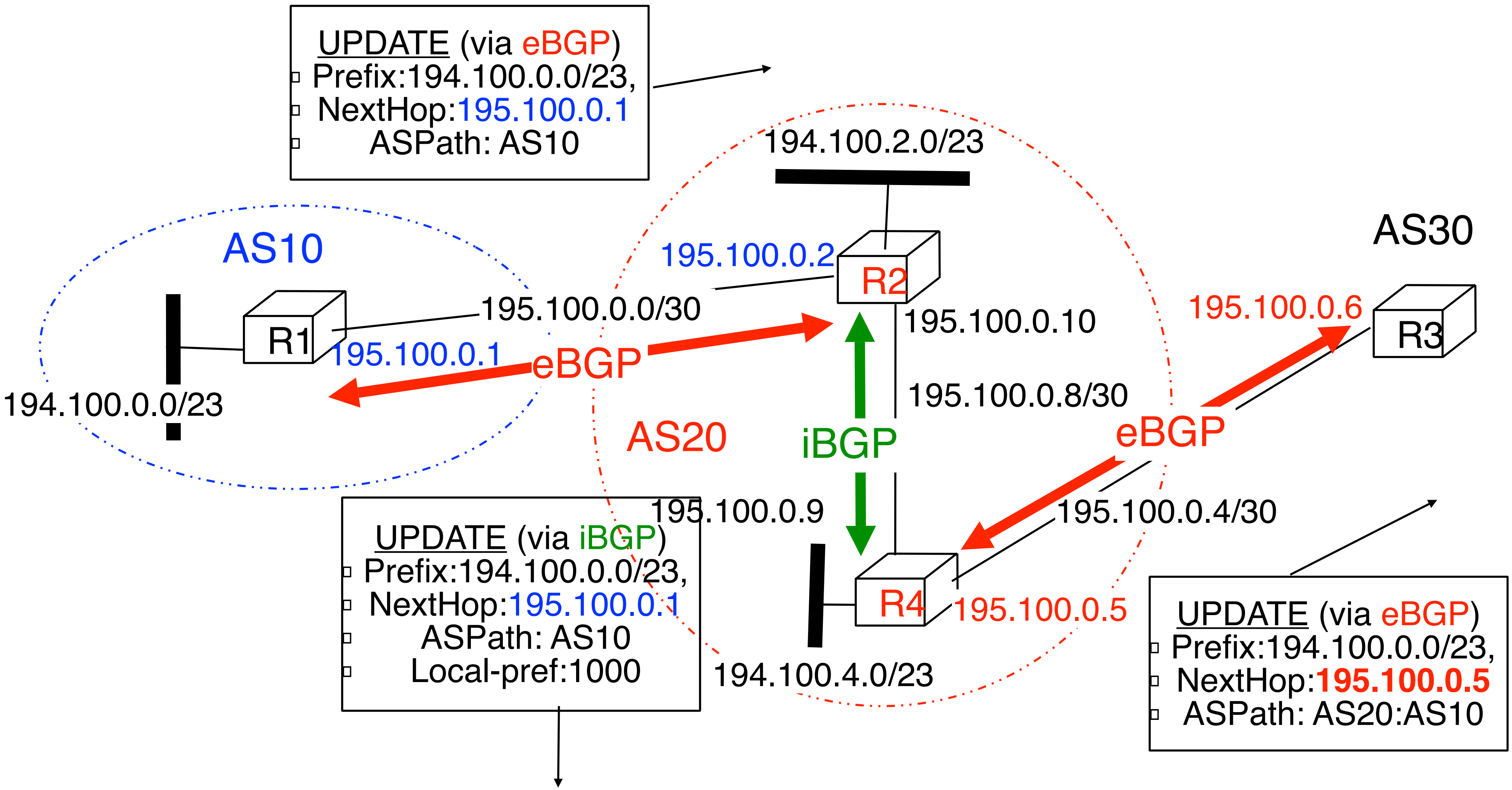
iBGP and eBGP : Example



iBGP and eBGP : Example

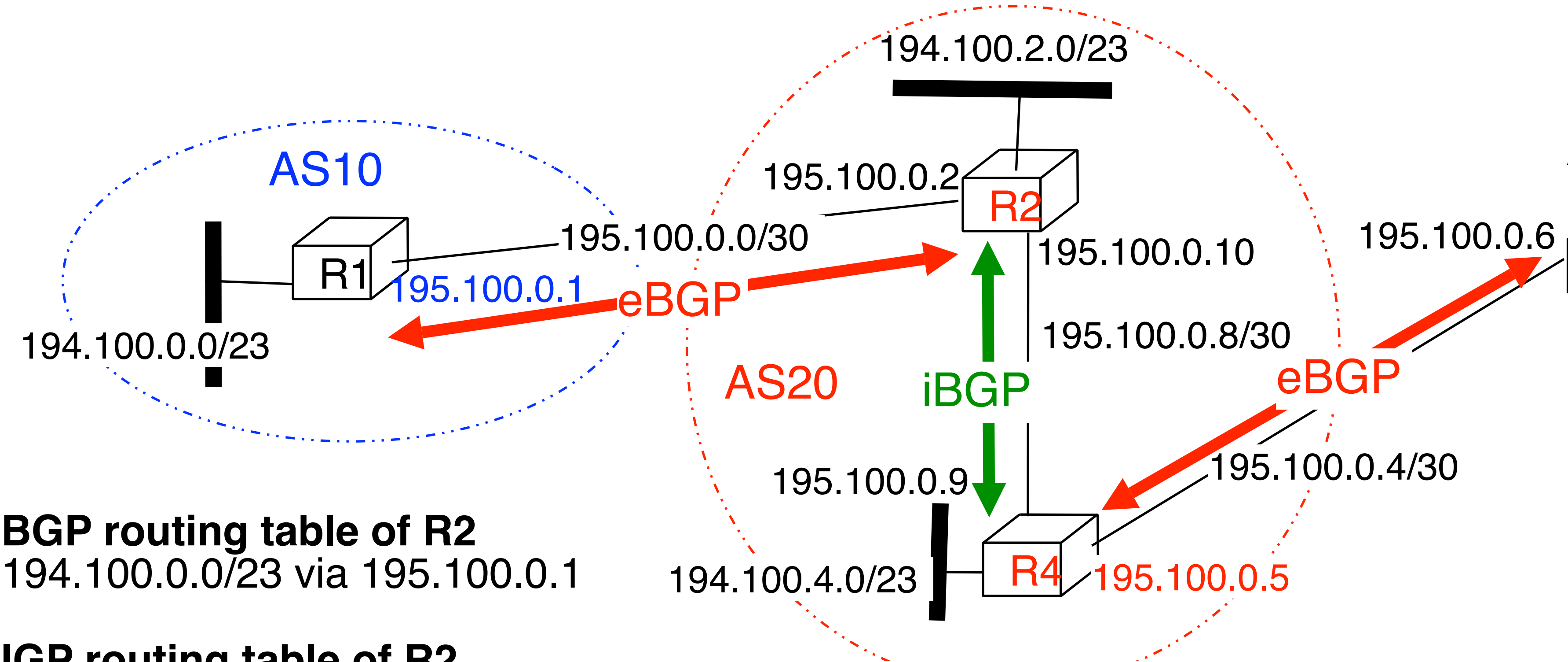


iBGP and eBGP : Example



- Note that the next-hop and the AS-Path of BGP update messages are only updated when sent over an eBGP session

iBGP and eBGP Packet Forwarding



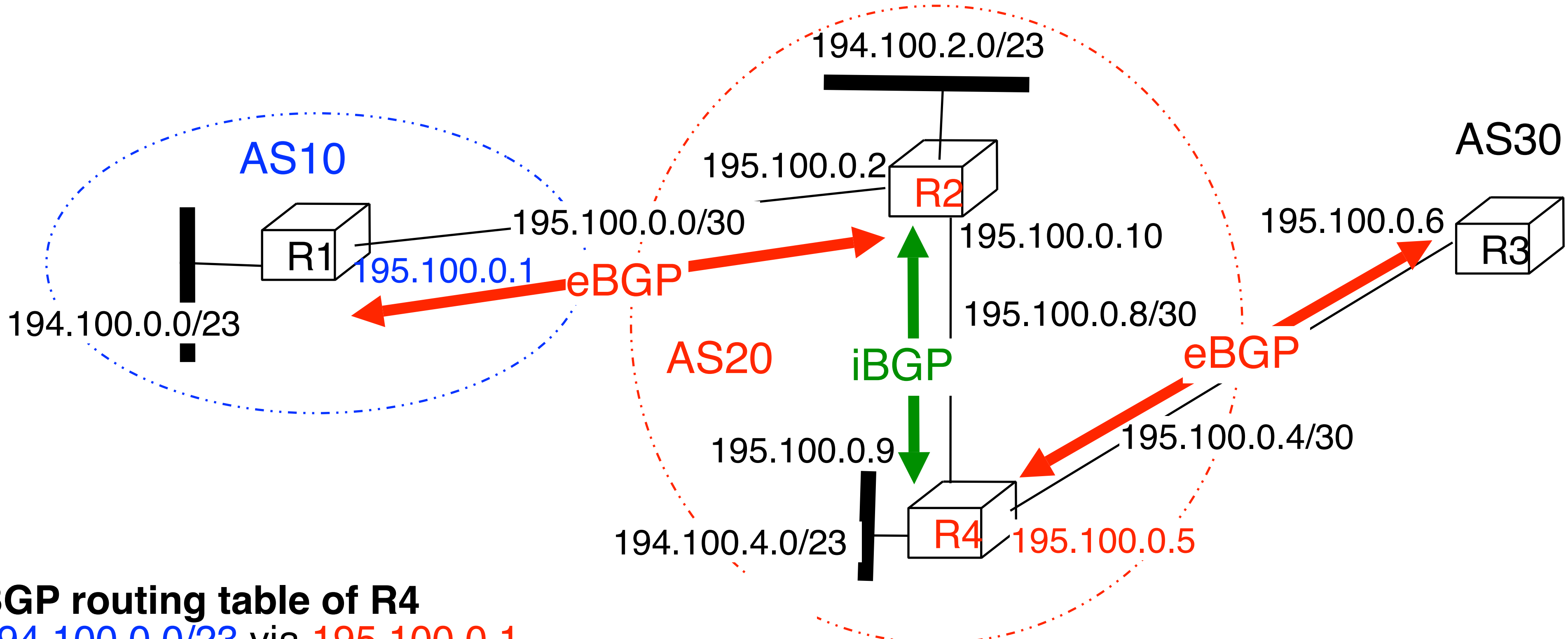
BGP routing table of R2
194.100.0.0/23 via 195.100.0.1

IGP routing table of R2
195.100.0.0/30 West
195.100.0.4/30 via 195.100.0.9
195.100.0.8/30 South
194.100.0.4/23 via 195.100.0.9
194.100.2.0/23 North

BGP routing table of R4
194.100.0.0/23 via 195.100.0.1

IGP routing table of R4
195.100.0.0/30 via 195.100.0.10
195.100.0.4/30 East
195.100.0.8/30 North
194.100.2.0/23 via 195.100.0.10
194.100.0.4/23 West

iBGP and eBGP Packet Forwarding (2)

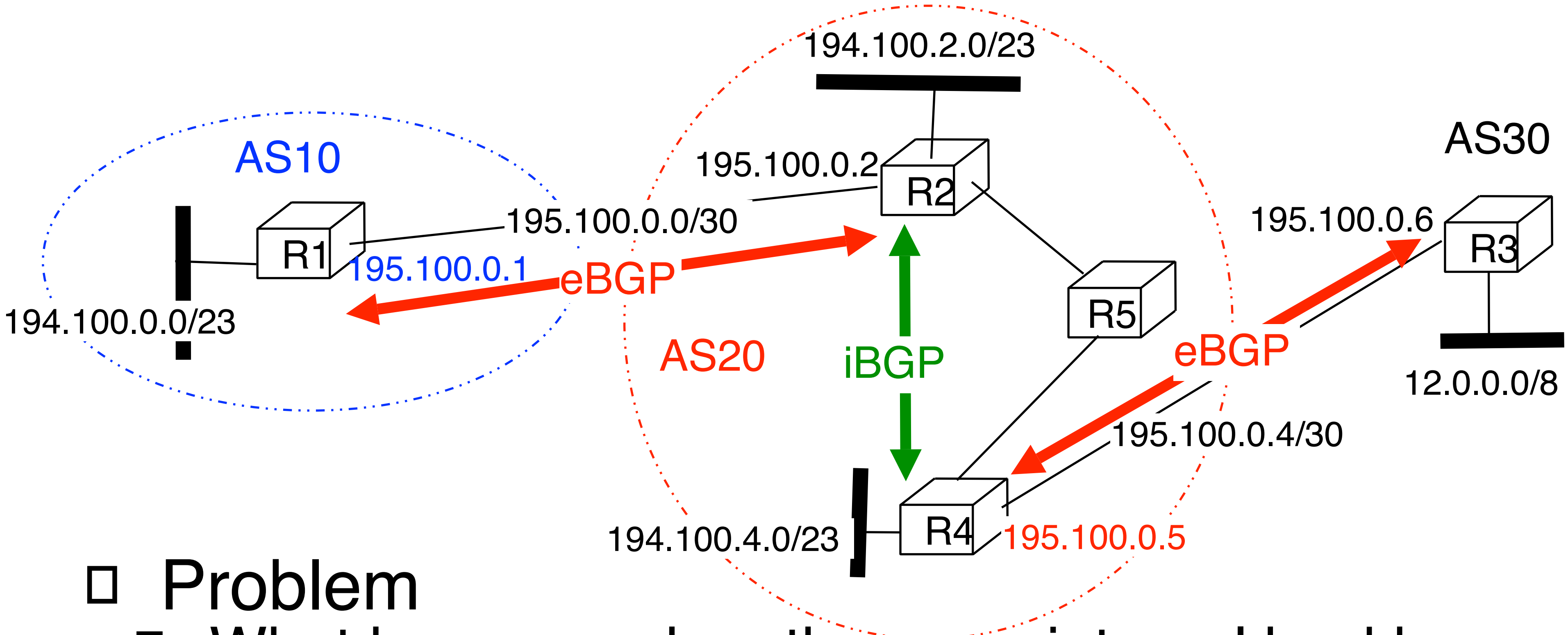


BGP routing table of R4
194.100.0.0/23 via 195.100.0.1

IGP routing table of R4
195.100.0.0/30 via **195.100.0.10**
195.100.0.4/30 East
195.100.0.8/30 North
194.100.2.0/23 via 195.100.0.10
194.100.4.0/23 West

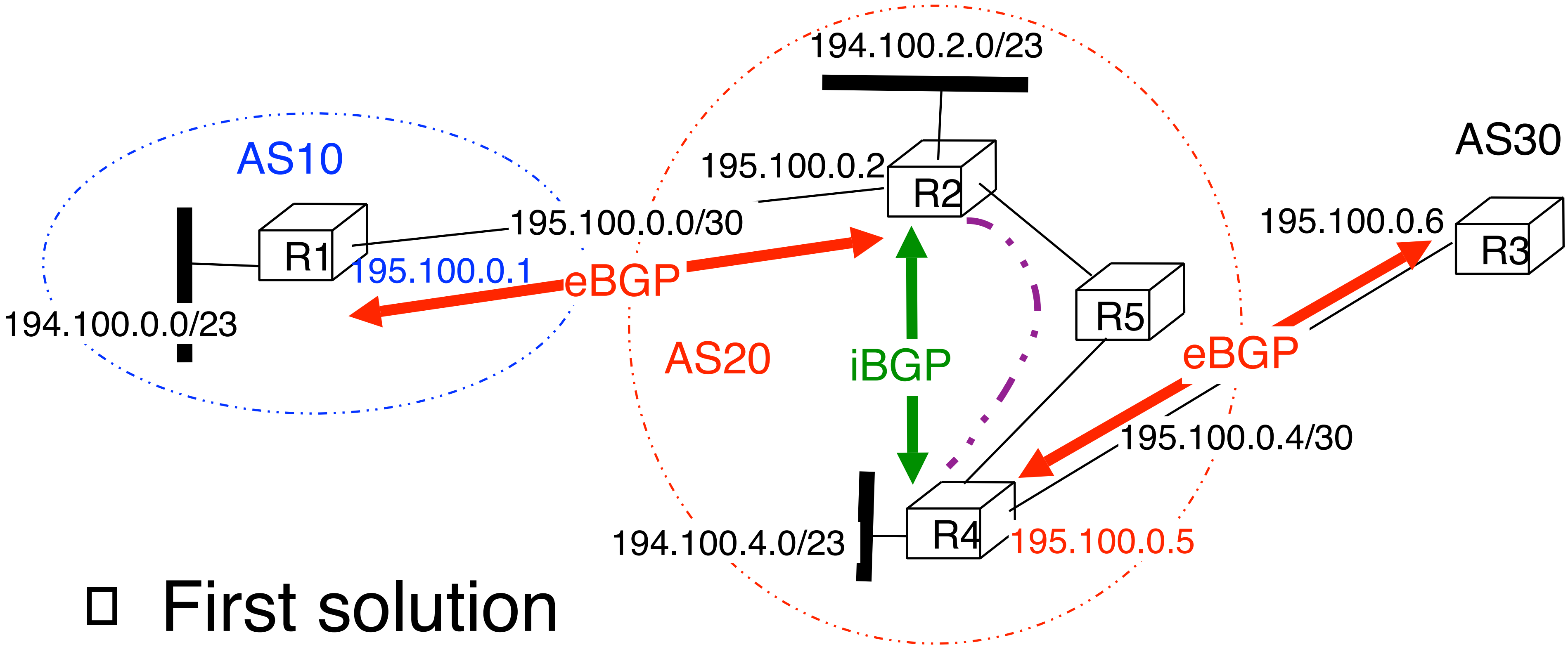
Forwarding of R4
194.100.0.0/23 via **195.100.0.10**
195.100.0.0/30 via 195.100.0.10
195.100.0.4/30 East
195.100.0.8/30 North
194.100.2.0/23 via 195.100.0.10
194.100.4.0/23 West

Using non-BGP routers



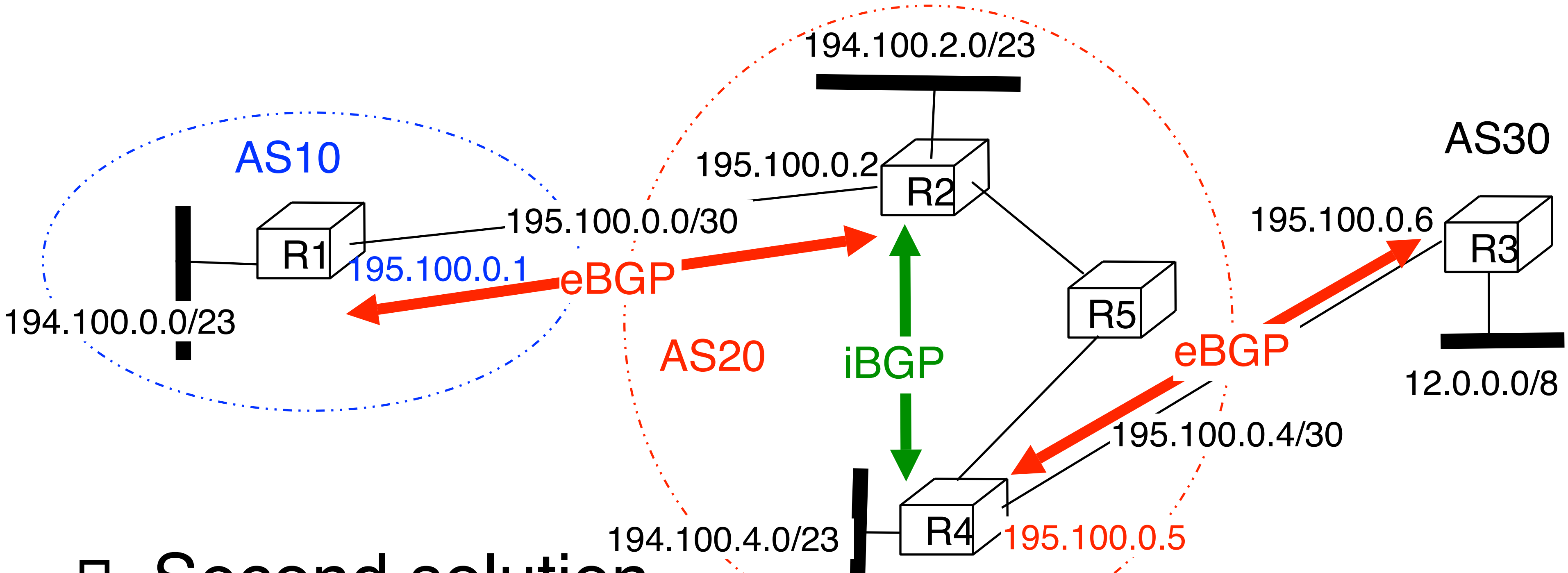
- Problem
  - What happens when there are internal backbone routers between BGP routers inside an AS ?
    - iBGP session between BGP routers is easily established when IGP is running since iBGP runs over TCP connection
    - How to populate the routing table of the backbone routers to ensure that they will be able to route any IP packet ?

Using non-BGP routers (2)



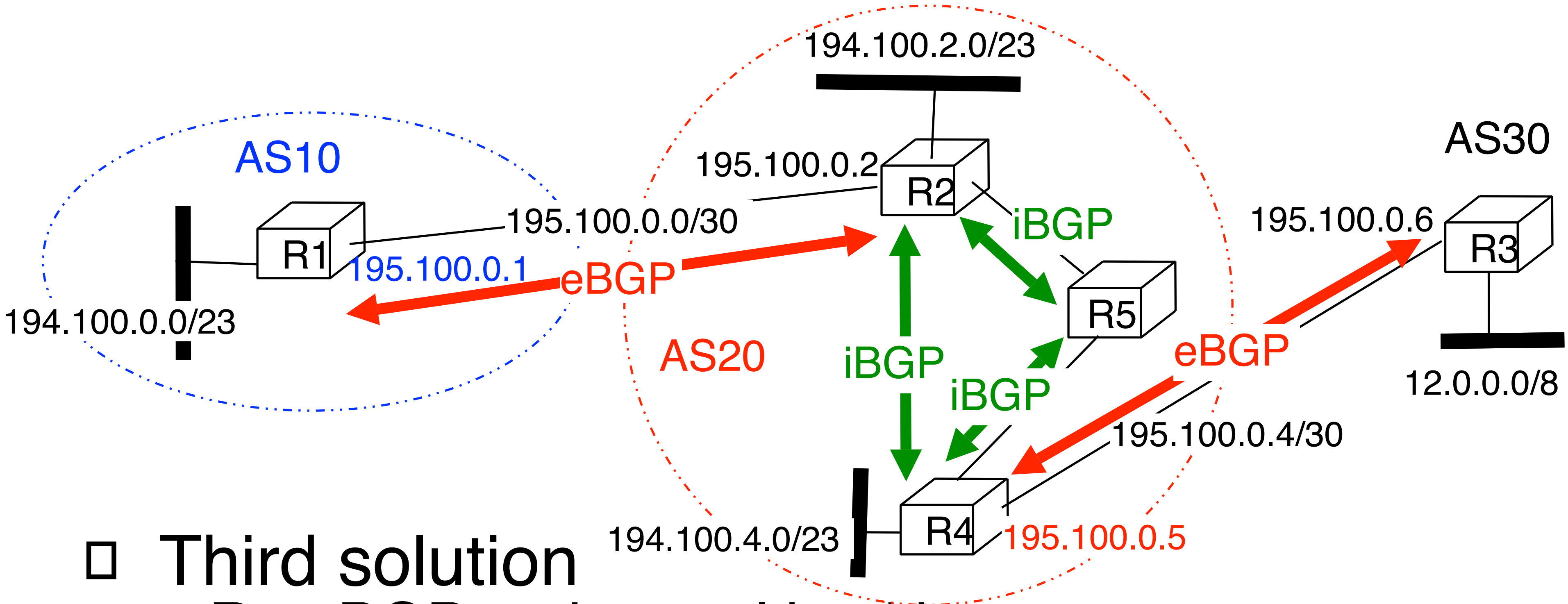
- First solution
  - Use tunnels between BGP routers to encapsulate interdomain packets
    - GRE tunnel
      - Needs static configuration and be careful with MTU issues
    - MPLS tunnel
      - Can be dynamically established in MPLS enabled backbone

Using non-BGP routers (3)



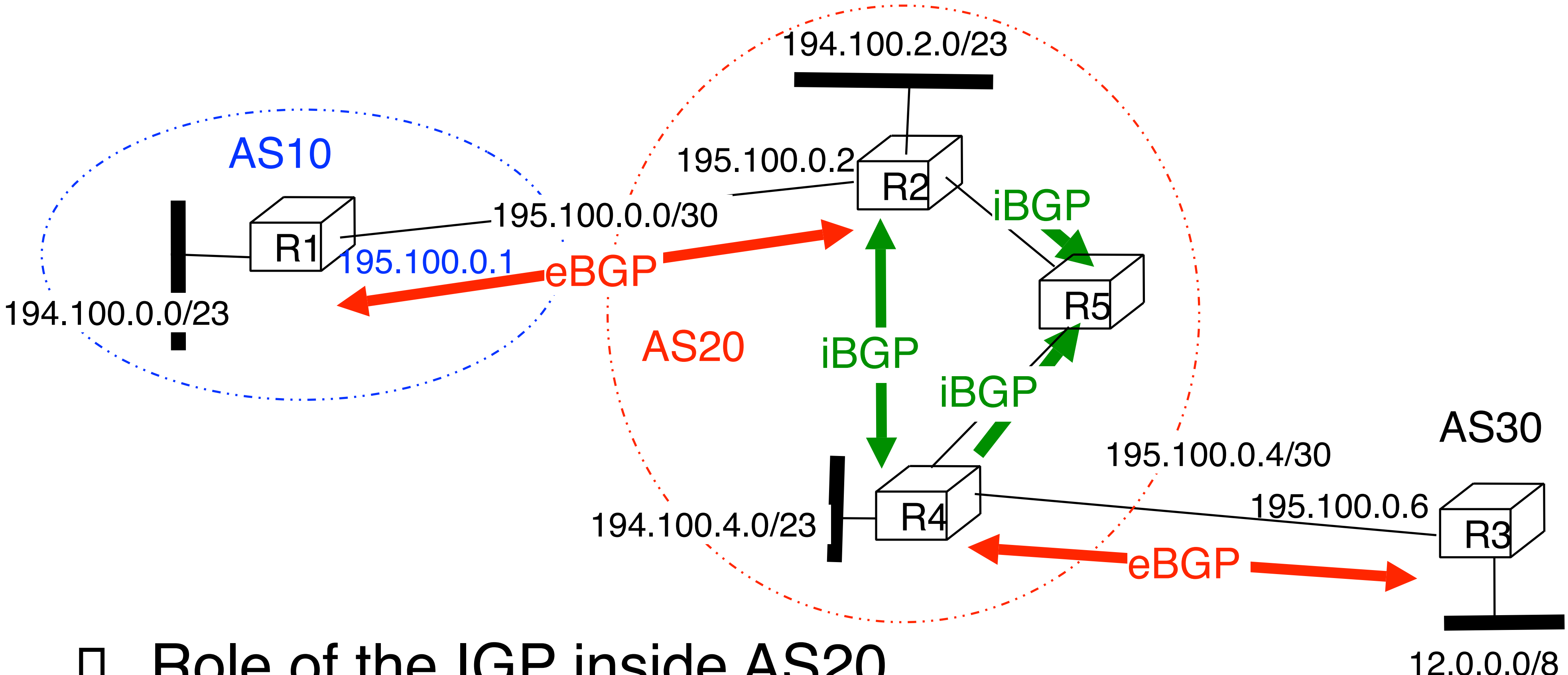
- Second solution
  - Use IGP (OSPF/IS-IS - RIP) to redistribute interdomain routes to internal backbone routers
  - Drawbacks
    - Size of BGP tables may completely overload the IGP
    - Make sure that BGP routes learned by R2 and injected inside IGP will not be re-injected inside BGP by R4 !

Using non-BGP routers (4)



- Third solution
  - Run BGP on internal backbone routers
  - Internal backbone routers need to participate in iBGP full mesh
    - Internal backbone routers receive BGP routes via iBGP but never advertise any routes
      - Remember : a route learned over an iBGP session is never advertised over another iBGP session

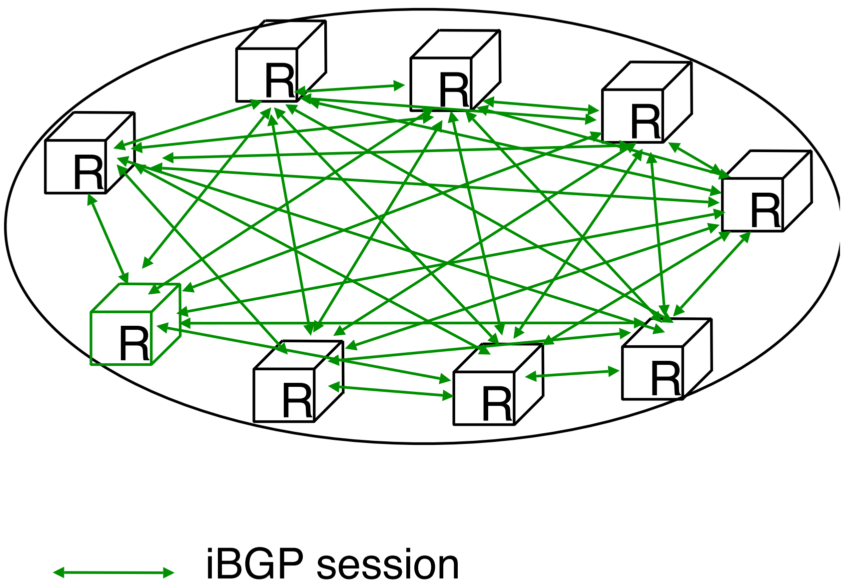
The roles of IGP and BGP

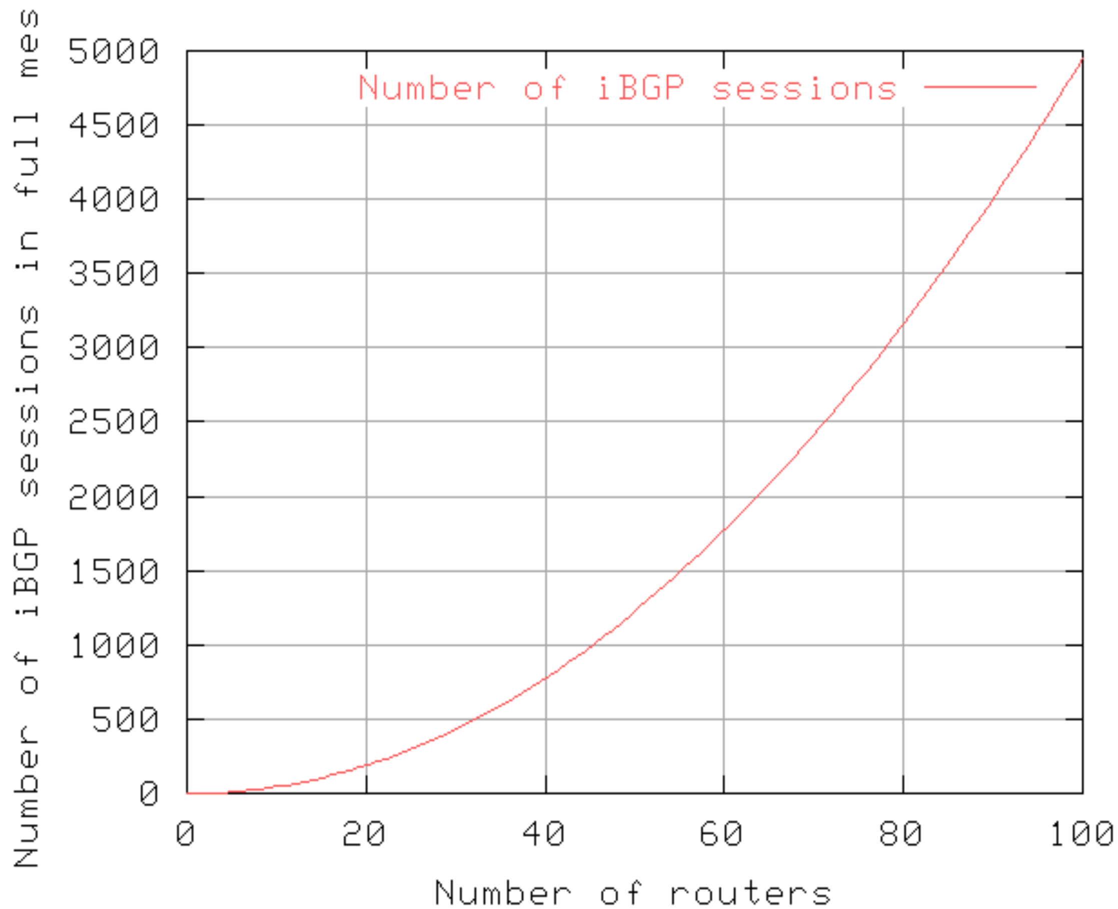


- Role of the IGP inside AS20
  - Distribute internal topology and internal addresses R2-R4-R5)
- Role of BGP inside AS20
  - Distribute the routes towards external destinations
  - IGP must run to allow BGP routers to establish iBGP sessions

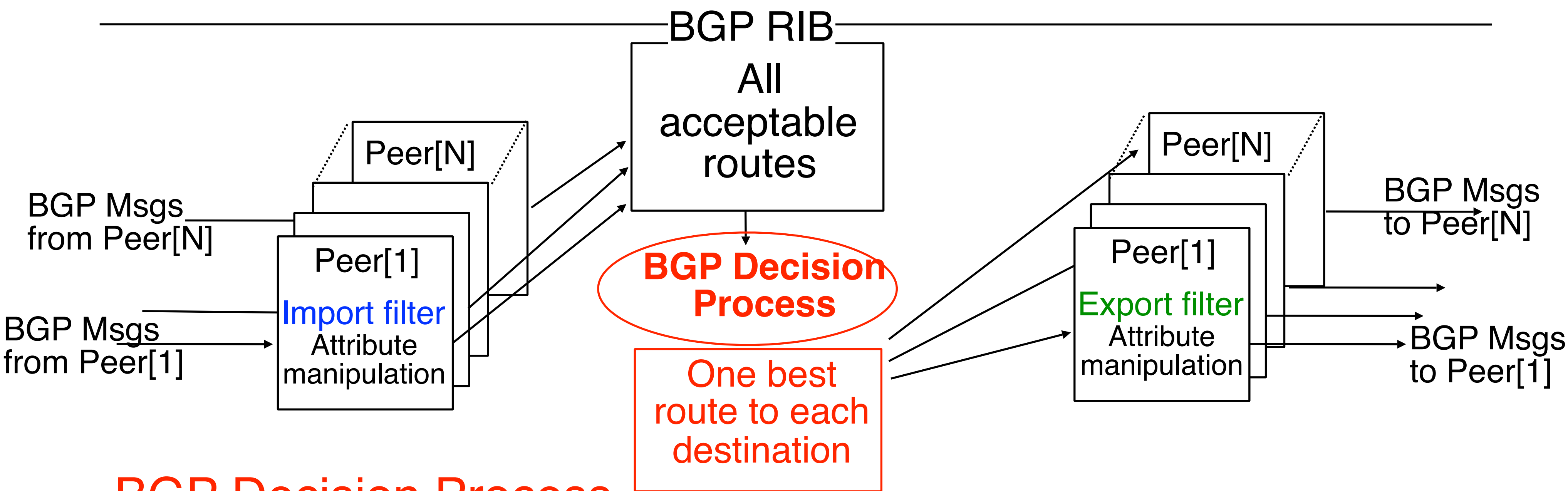
The iBGP full mesh

- Drawback
  - N*(N-1)/2 iBGP sessions for N routers





The BGP decision process

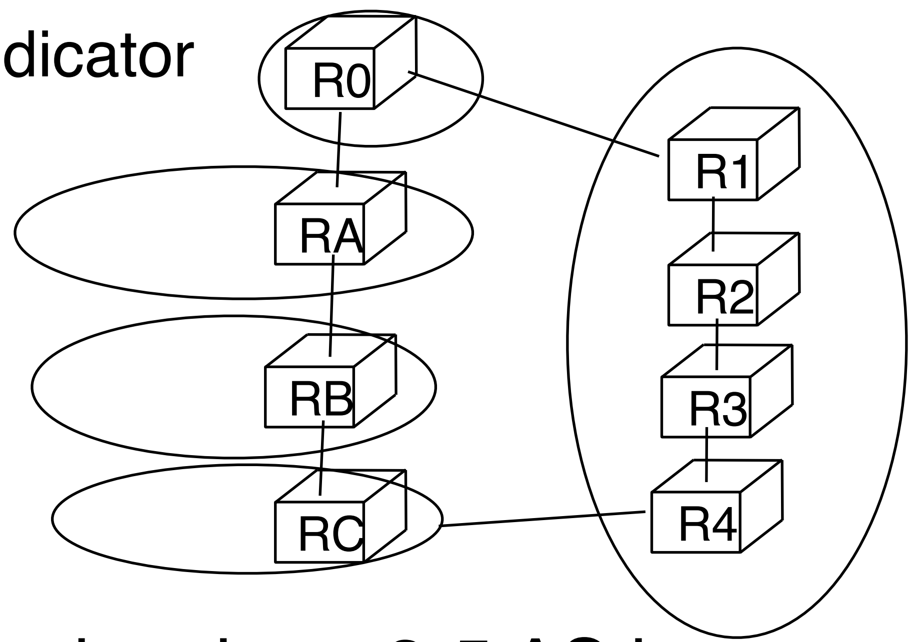


BGP Decision Process

- *Ignore routes with unreachable nexthop*
- Prefer routes with highest local-pref
- Prefer routes with shortest ASPath
- Prefer routes with smallest MED
- Prefer routes learned via eBGP over routes learned via iBGP
- Prefer routes with closest next-hop
- Tie breaking rules
  - Prefer Routes learned from router with lowest router id

The shortest `AS-Path` step in the BGP decision process

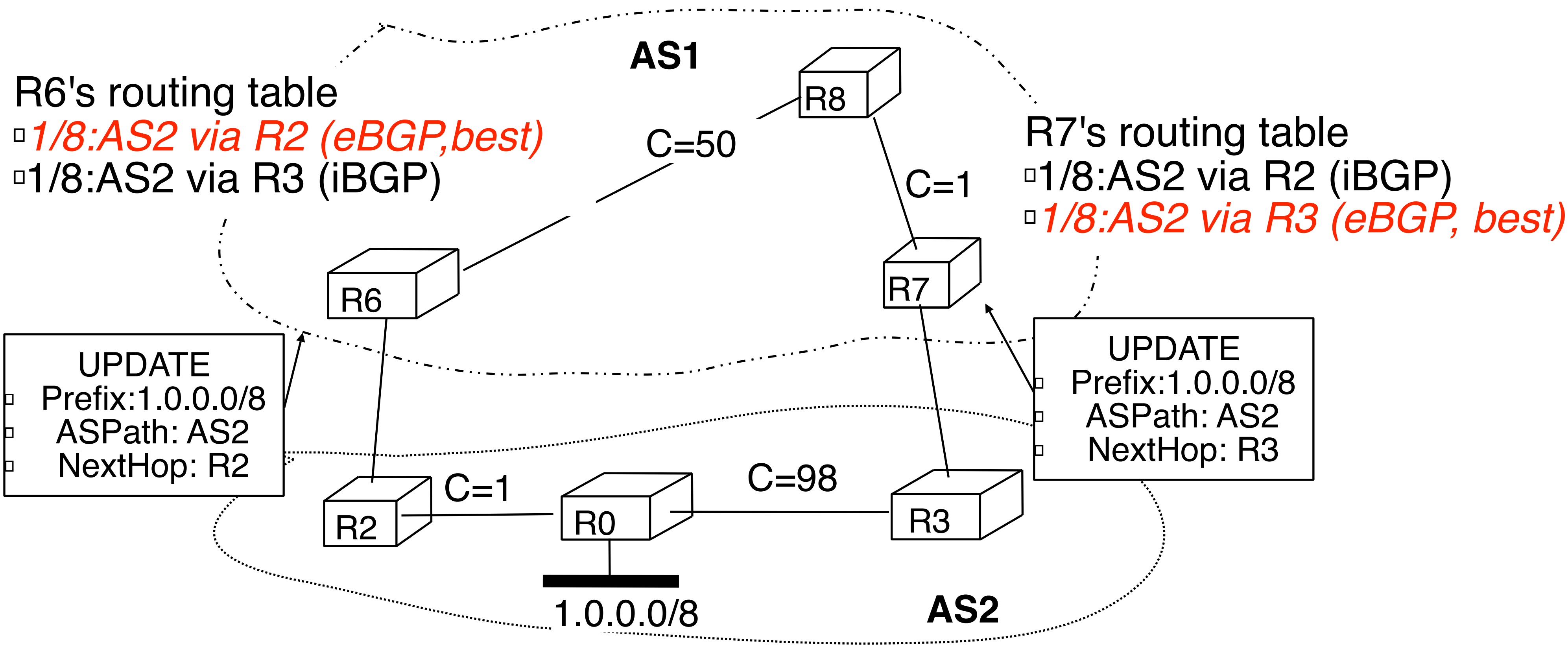
- Motivation
  - BGP does not contain a real “ metric”
  - Use length of AS-Path as an indication of the quality of routes
    - Not always a good indicator



- Consequence
  - Internet paths tend to be short, 3-5 AS hops
  - Many paths converge at Tier-1 ISPs and those ISPs carry lots of traffic

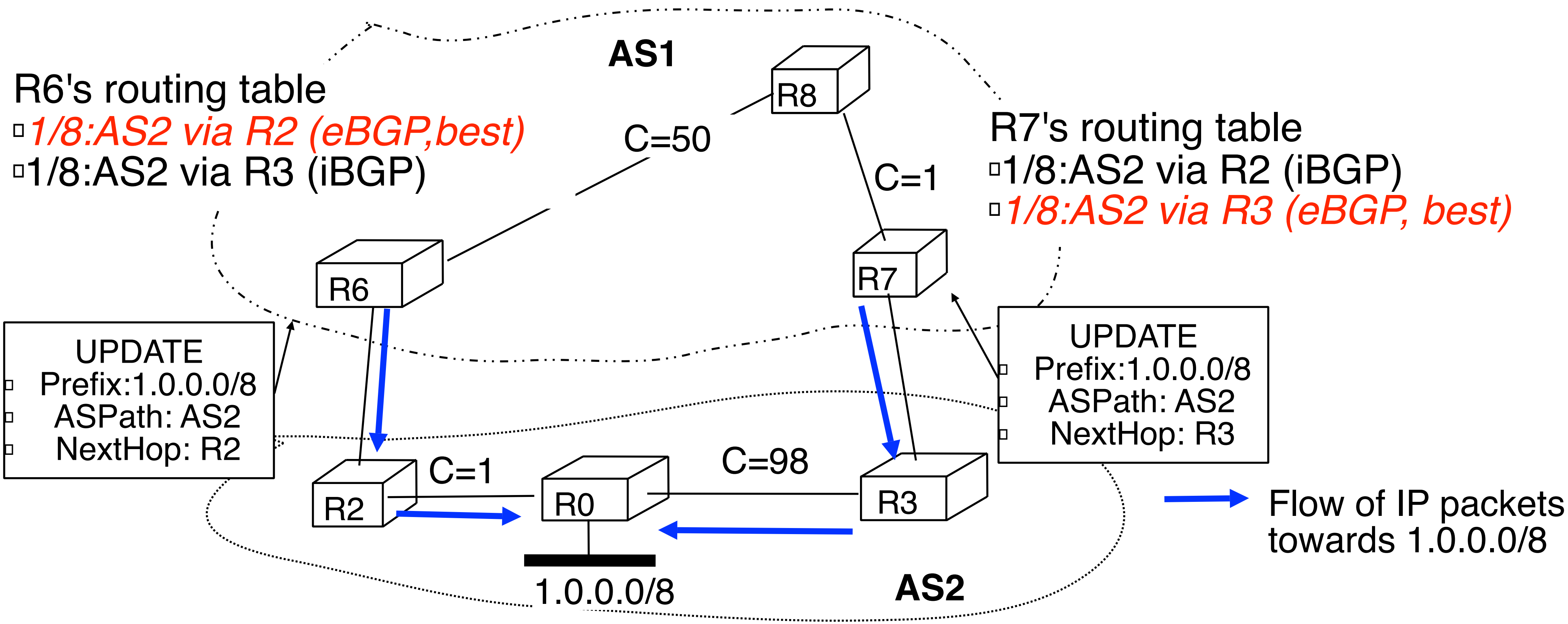
The prefer eBGP over iBGP step in the BGP decision process

- Motivation : hot potato routing
  - A router should try to get rid of packets sent to external domains as soon as possible



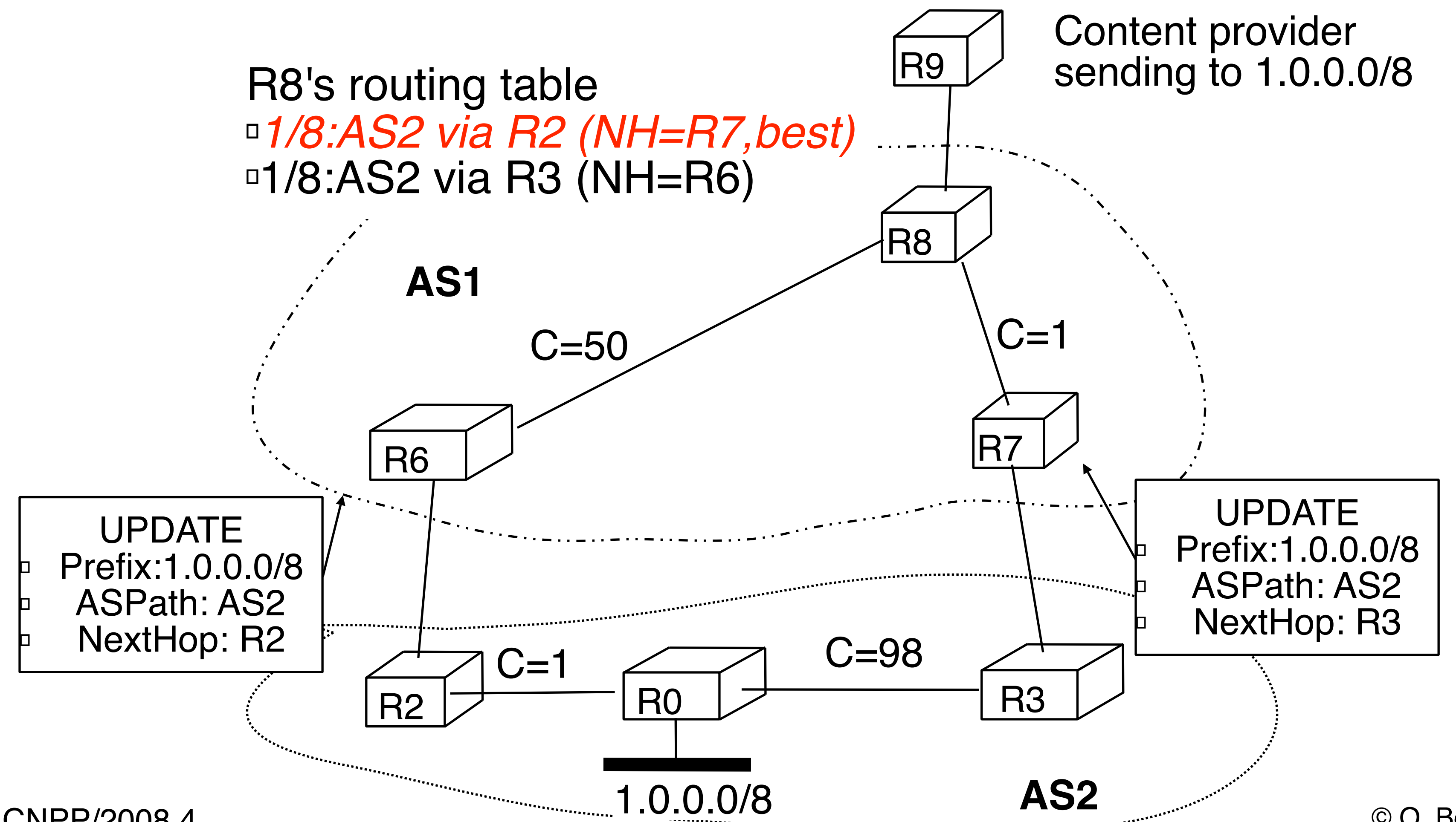
The prefer eBGP over iBGP step in the BGP decision process

- Motivation : hot potato routing
  - A router should try to get rid of packets sent to external domains as soon as possible



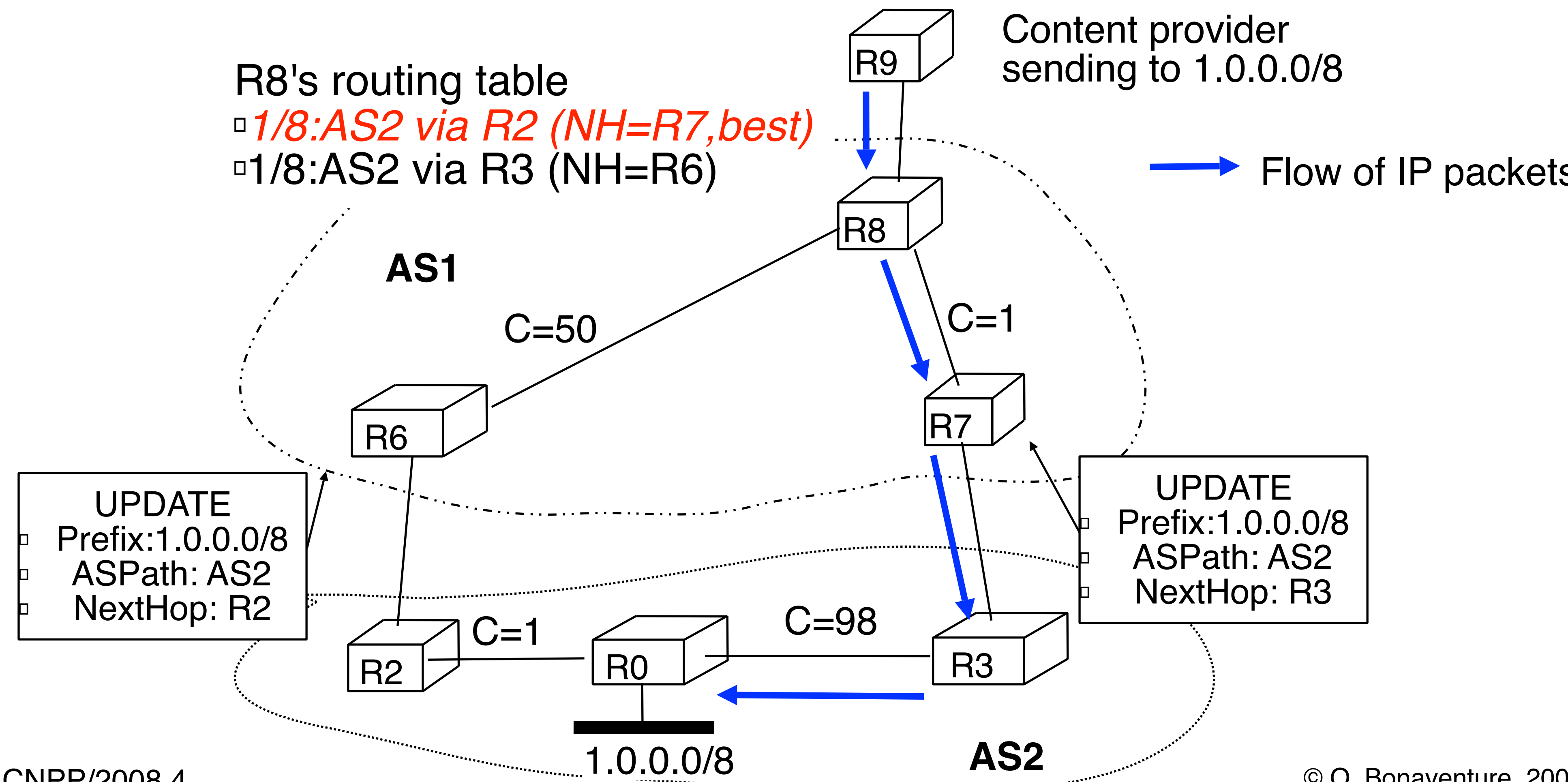
The closest `nexthop` step in the BGP decision process

- Motivation : hot potato routing
  - A router should try to get rid of packets sent to external domains as soon as possible



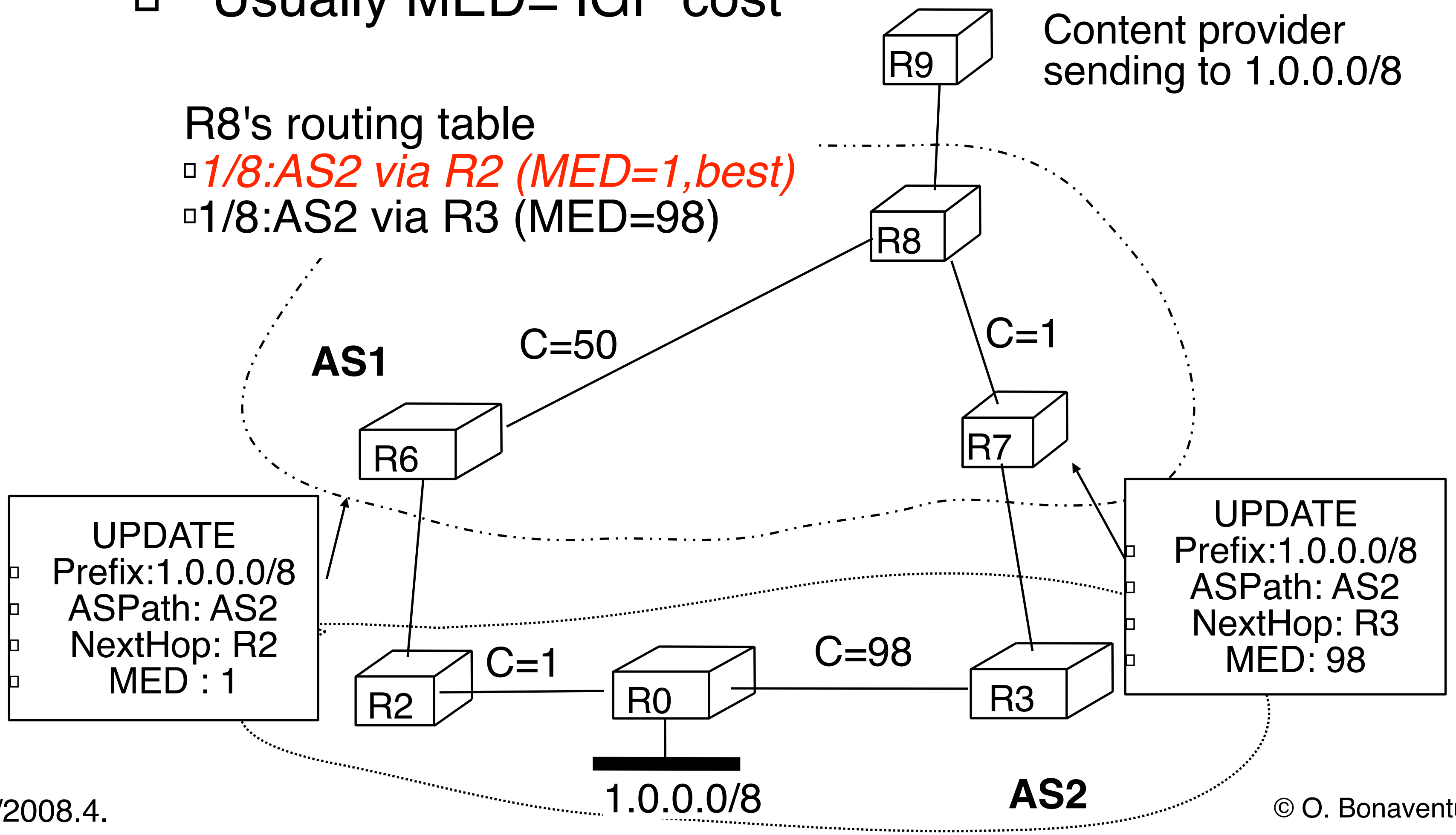
The closest `nexthop` step in the BGP decision process

- Motivation : hot potato routing
  - A router should try to get rid of packets sent to external domains as soon as possible



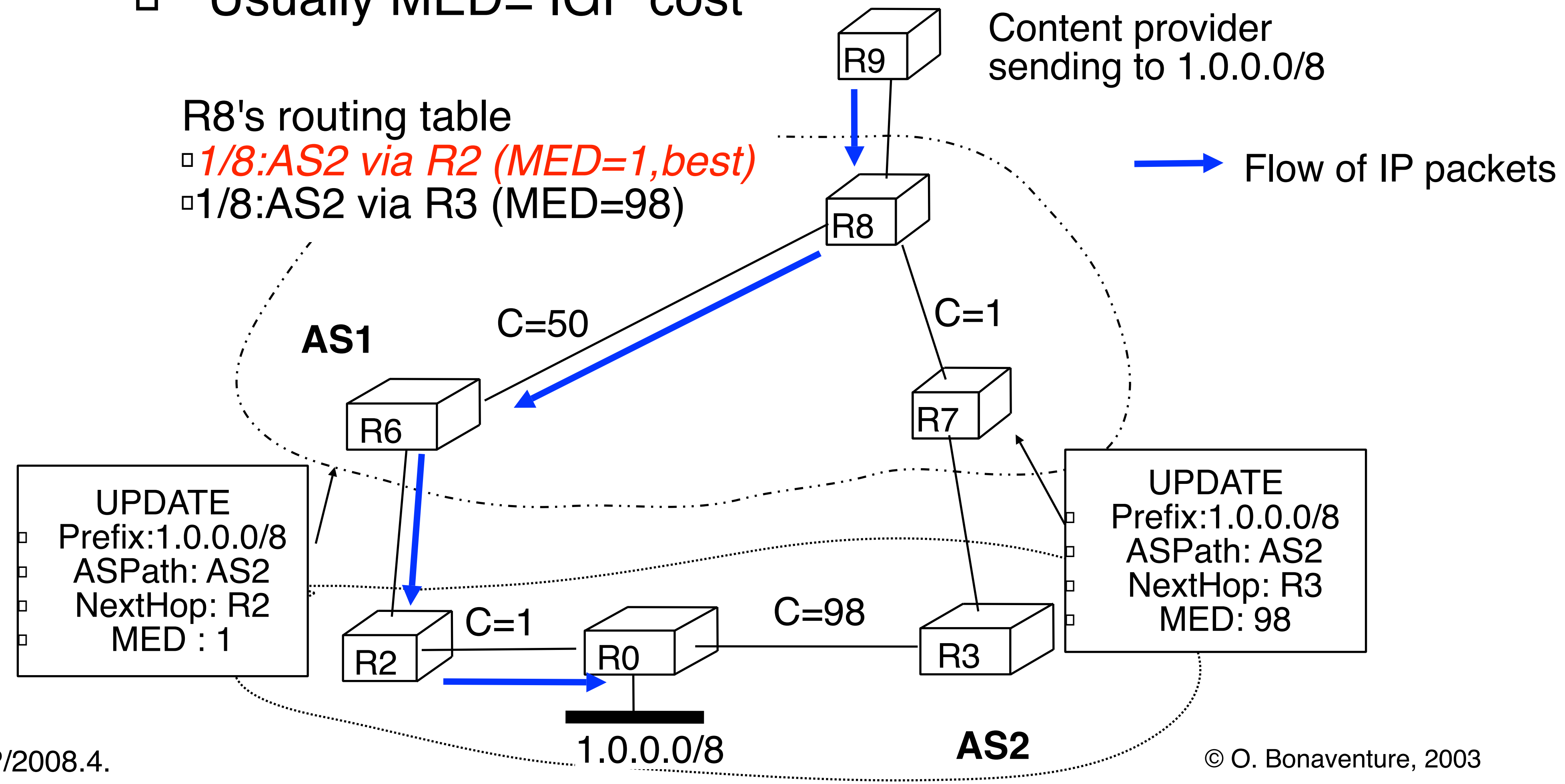
The lowest `MED` step in the BGP decision process

- Motivation : cold potato routing
  - In a multi-connected AS, indicate which entry border router is closest to the advertised prefix
    - Usually MED= IGP cost



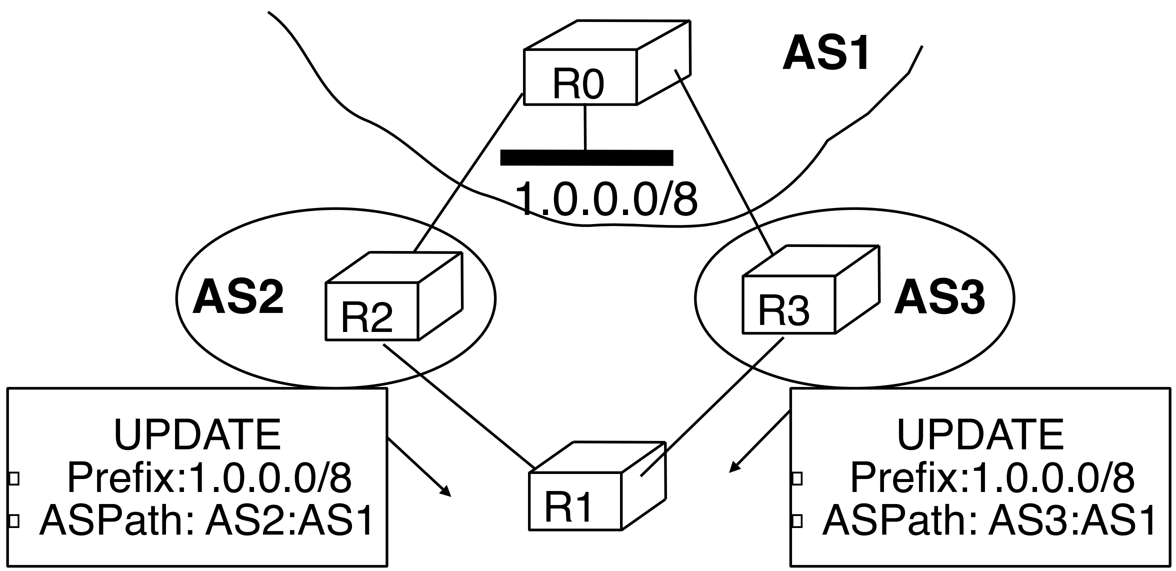
The lowest MED step in the BGP decision process

- Motivation : cold potato routing
  - In a multi-connected AS, indicate which entry border router is closest to the advertised prefix
    - Usually MED= IGP cost



The lowest router id step in the BGP decision process

- Motivation
  - A router must be able to determine *one* best route towards each destination prefix
    - A router may receive several routes with comparable attributes towards one destination



- Consequence
  - A router with a low IP address will be preferred

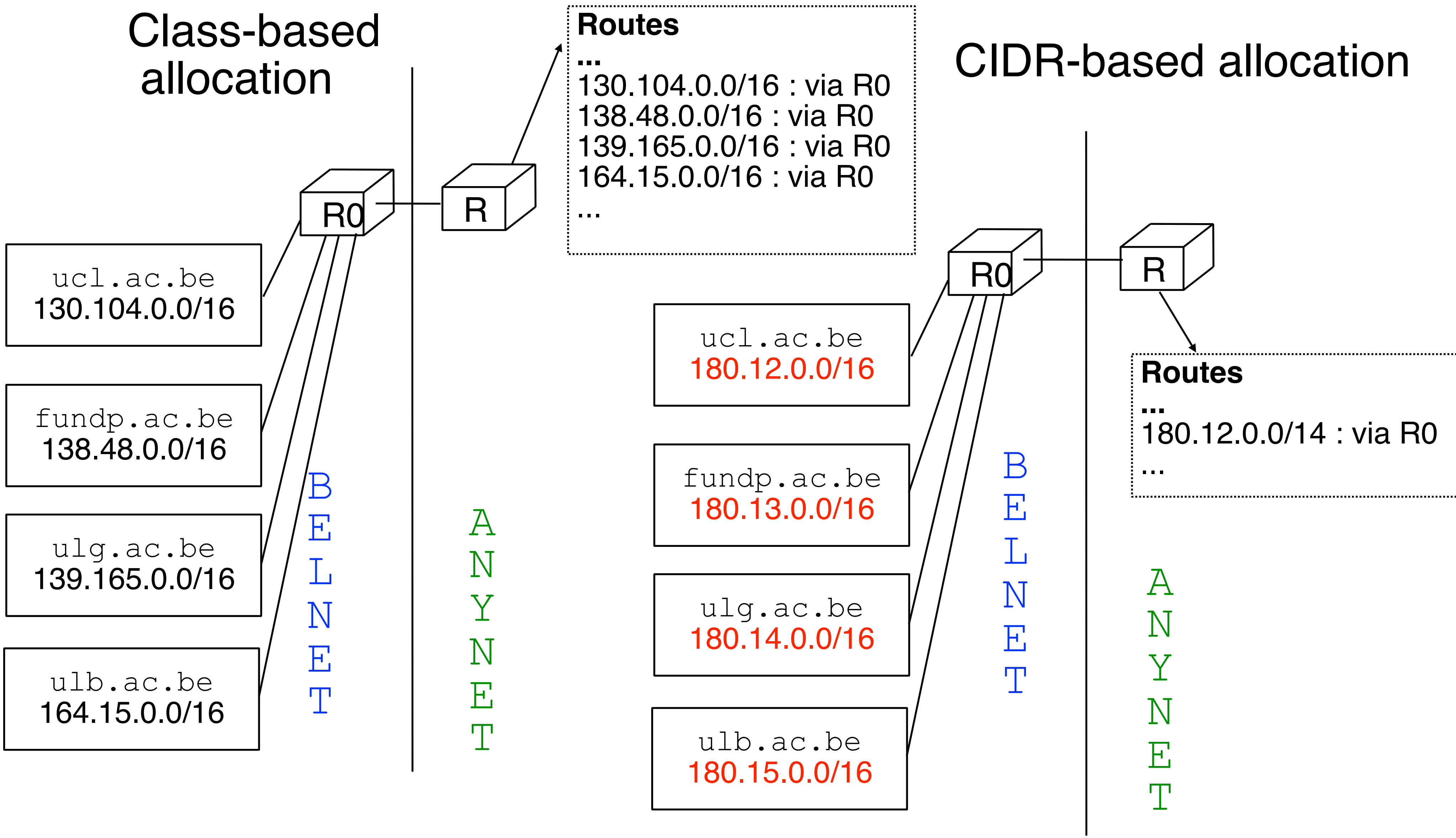
Allocation of IP addresses

- How to allocate IP addresses
- First solution
  - Objective : Ensure that IP addresses are unique
  - Rule used by registries
    - Any organisation can be allocated a unique IP subnet on a FCFS basis
  - Size of the allocated subnet : three classes
    - Class A : subnet with 8 bits mask
    - Class B : subnet with 16 bits mask
    - Class C : subnet with 24 bits mask
  - Drawbacks
    - Too rigid
      - Class A is too large for most networks and Class C too small
        - address waste !
    - Difficult to aggregate prefixes

Allocation of IP addresses (2)

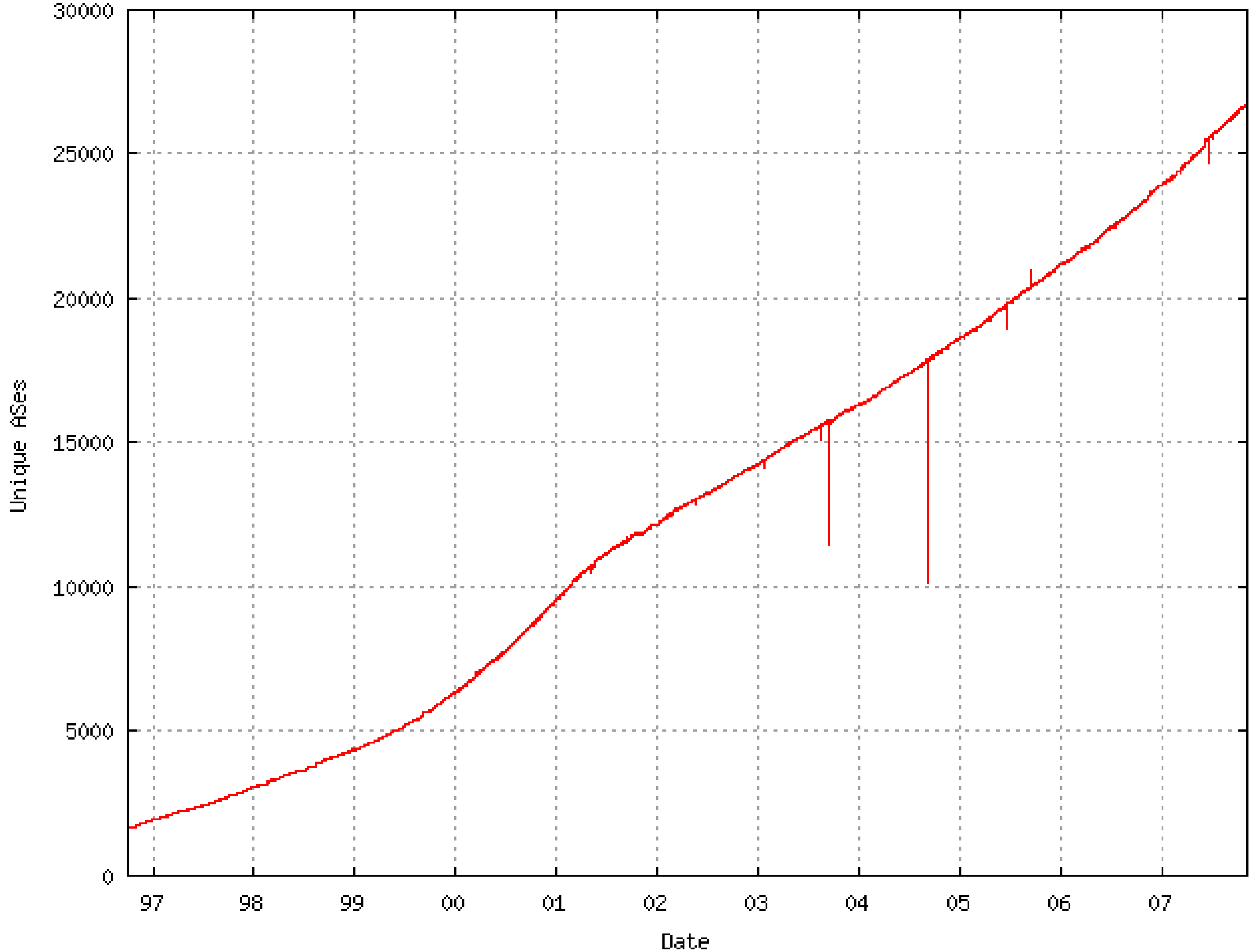
- CIDR
  - Goals
    1. Ensure that IP addresses are unique
    2. Allow BGP routers to advertise aggregated prefixes
  - Rules used by registries
    - Only Internet Service Providers (and large companies) can obtain IP subnets
      - Size of allocated subnet is function of current and expected number of customers
    - An organisation willing to be connected to the Internet must obtain IP addresses from its ISP
  - Advantage
    - Improved aggregation of addresses
  - Drawback
    - If a company switches from one provider to another, it will need to renumber its IP network - a real pain !

Allocation of IP addresses (3)



Internet evolution

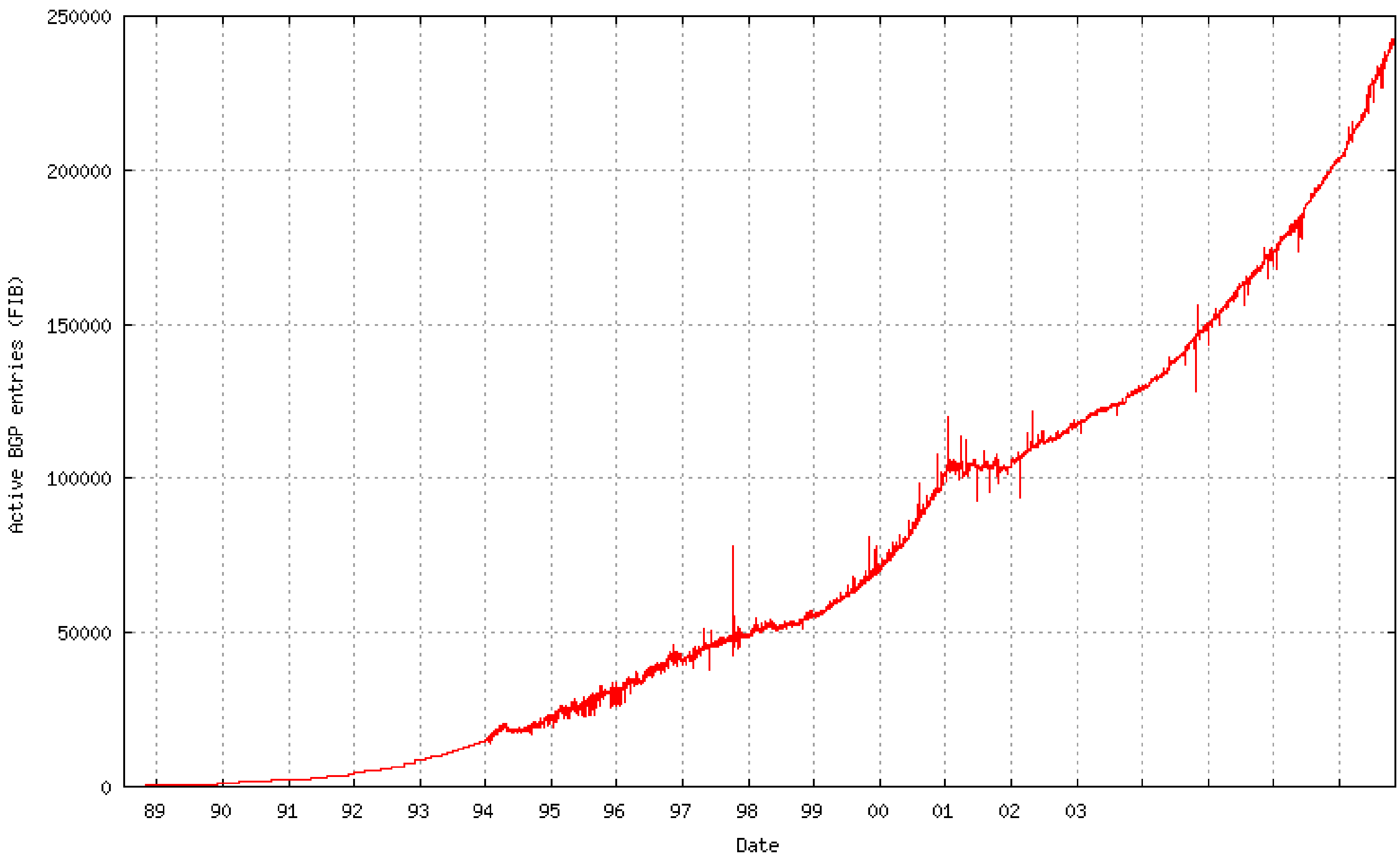
- Number of Autonomous Systems



Source : http://bgp.potaroo.net

Internet evolution (2)

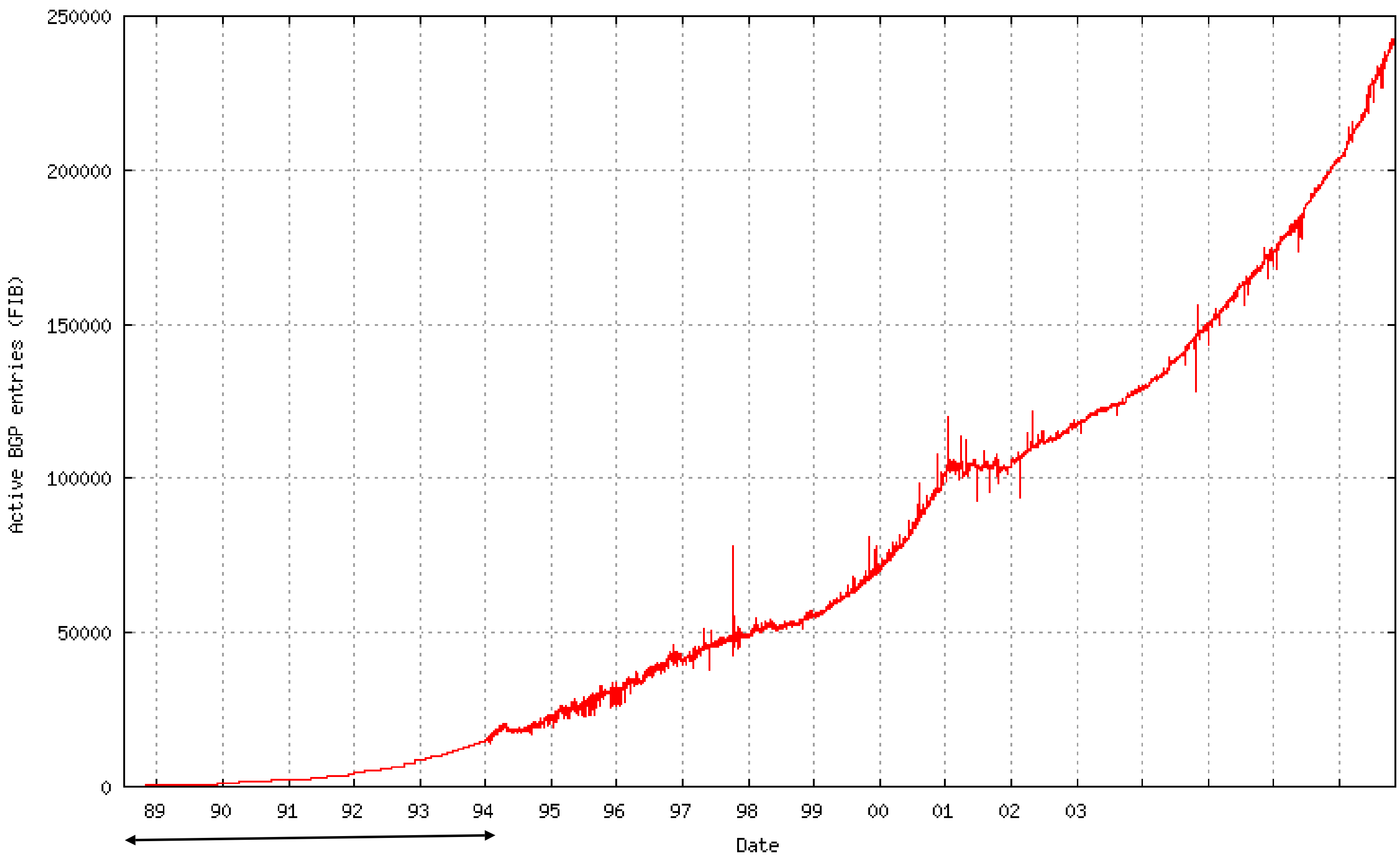
- Size of the BGP routing tables
  - Number of IPv4 prefixes in default-free routers



Source : http://bgp.potaroo.net

Internet evolution (2)

- Size of the BGP routing tables
  - Number of IPv4 prefixes in default-free routers

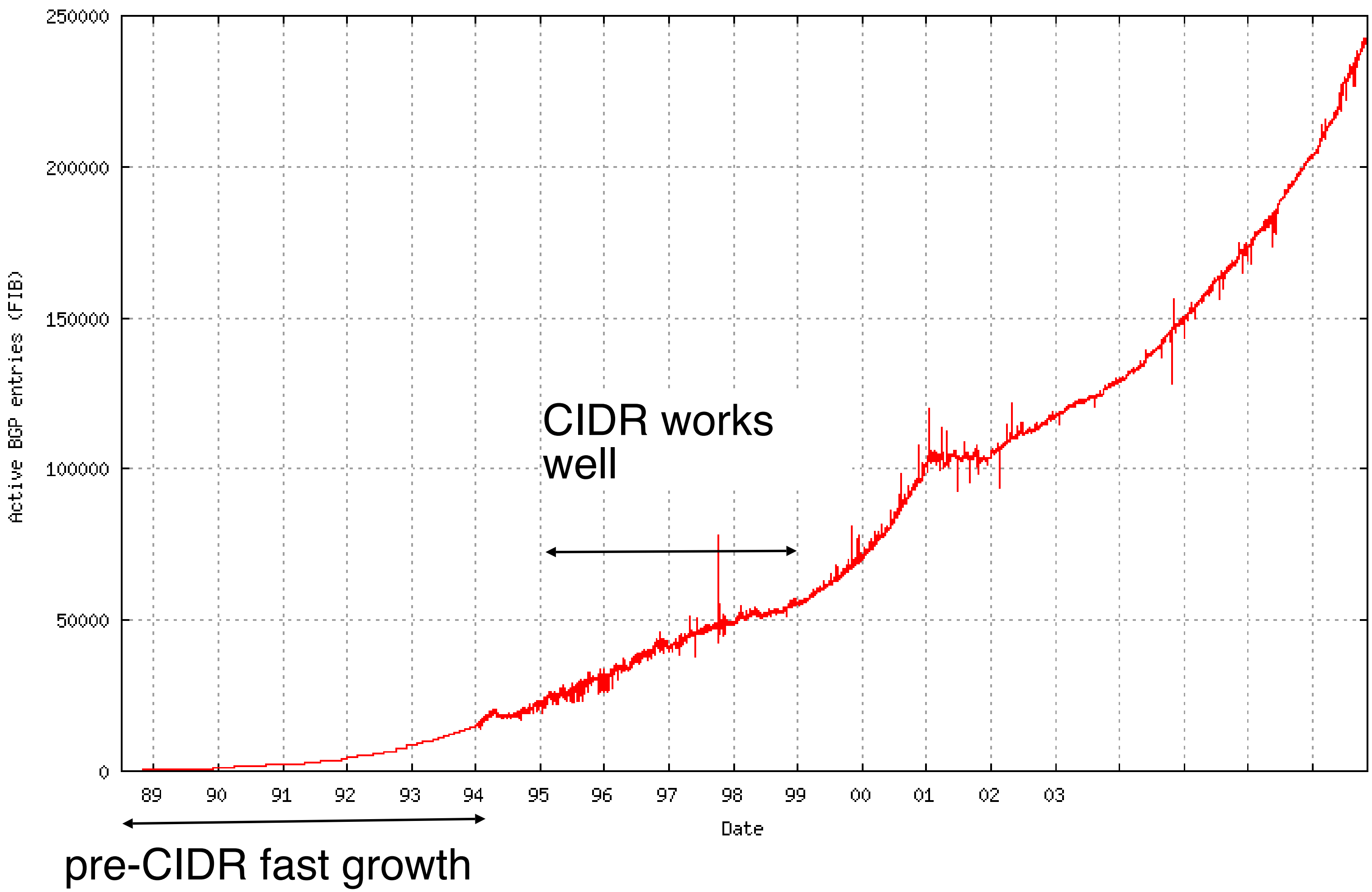


pre-CIDR fast growth

Source : http://bgp.potaroo.net

Internet evolution (2)

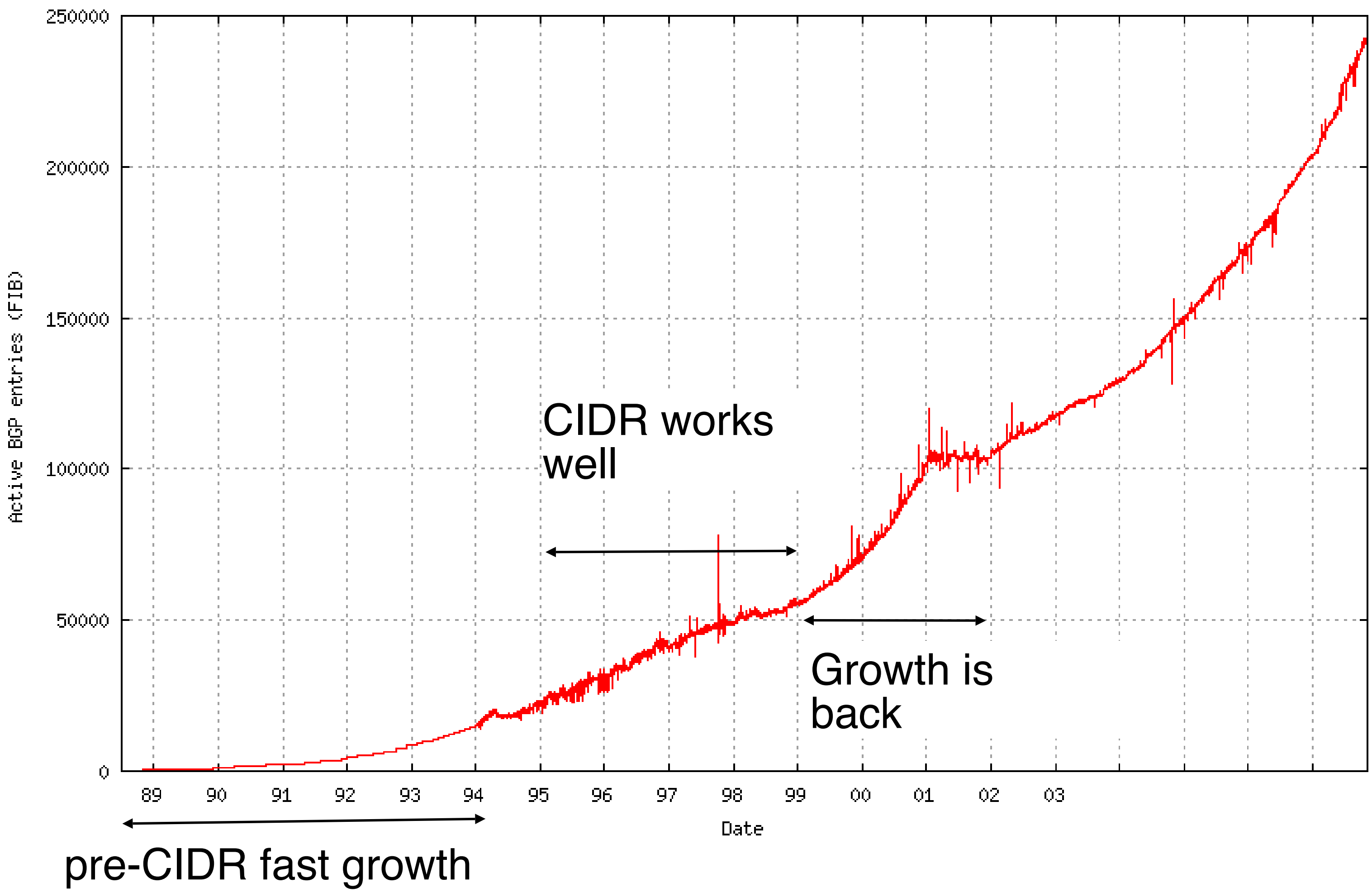
- Size of the BGP routing tables
  - Number of IPv4 prefixes in default-free routers



Source : http://bgp.potaroo.net

Internet evolution (2)

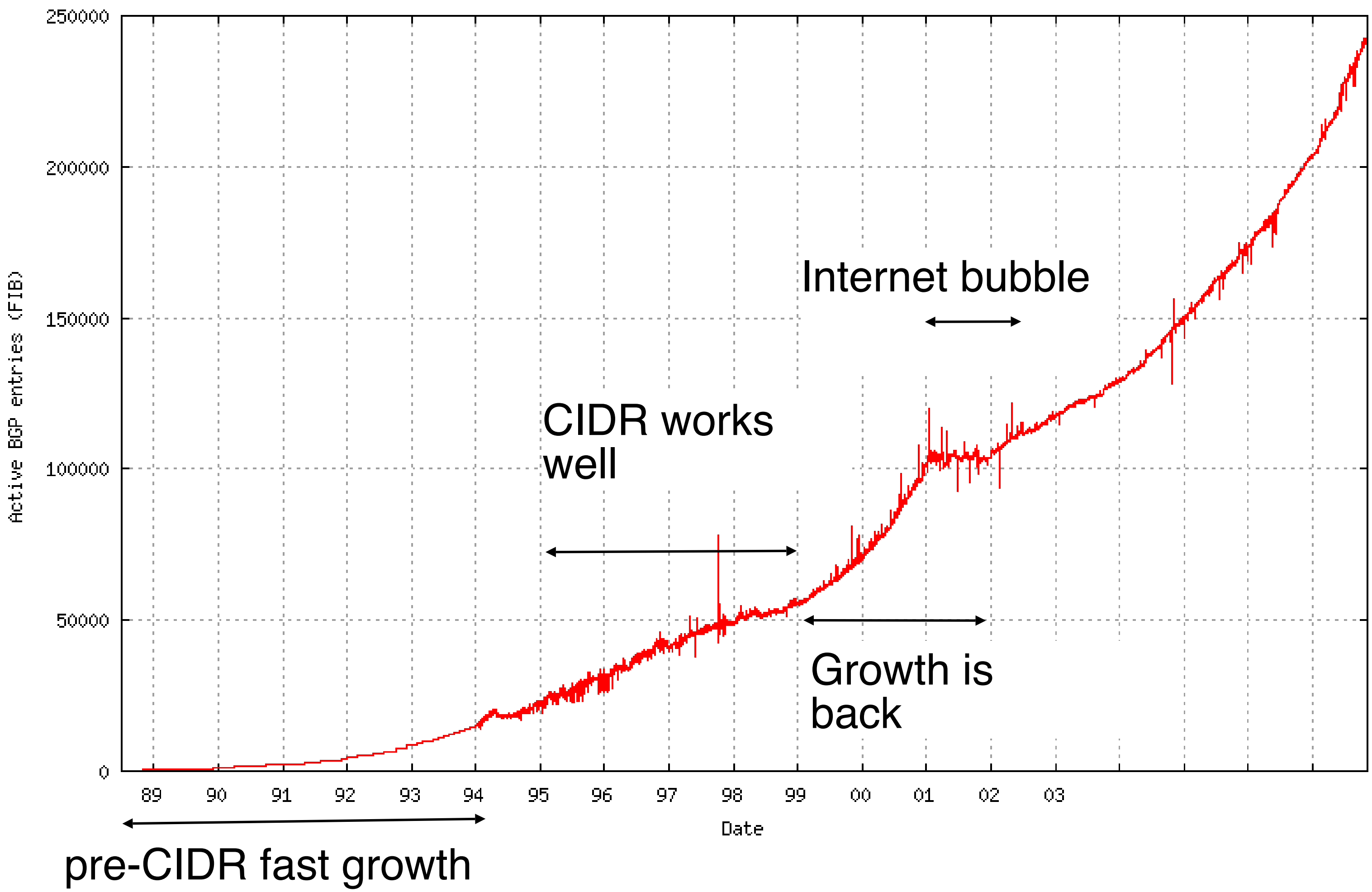
- Size of the BGP routing tables
  - Number of IPv4 prefixes in default-free routers



Source : http://bgp.potaroo.net

Internet evolution (2)

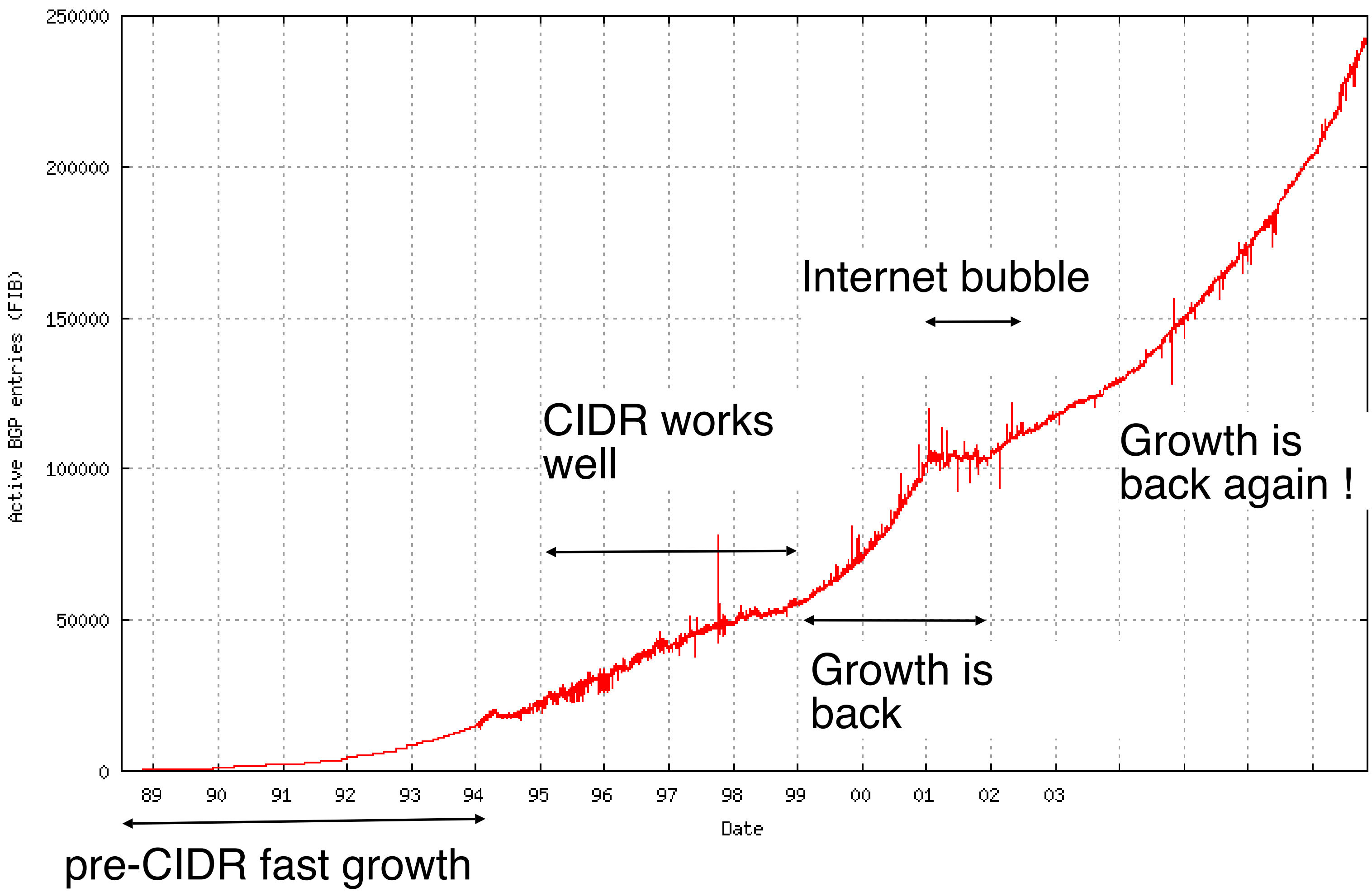
- Size of the BGP routing tables
  - Number of IPv4 prefixes in default-free routers



Source : http://bgp.potaroo.net

Internet evolution (2)

- Size of the BGP routing tables
  - Number of IPv4 prefixes in default-free routers



Source : http://bgp.potaroo.net