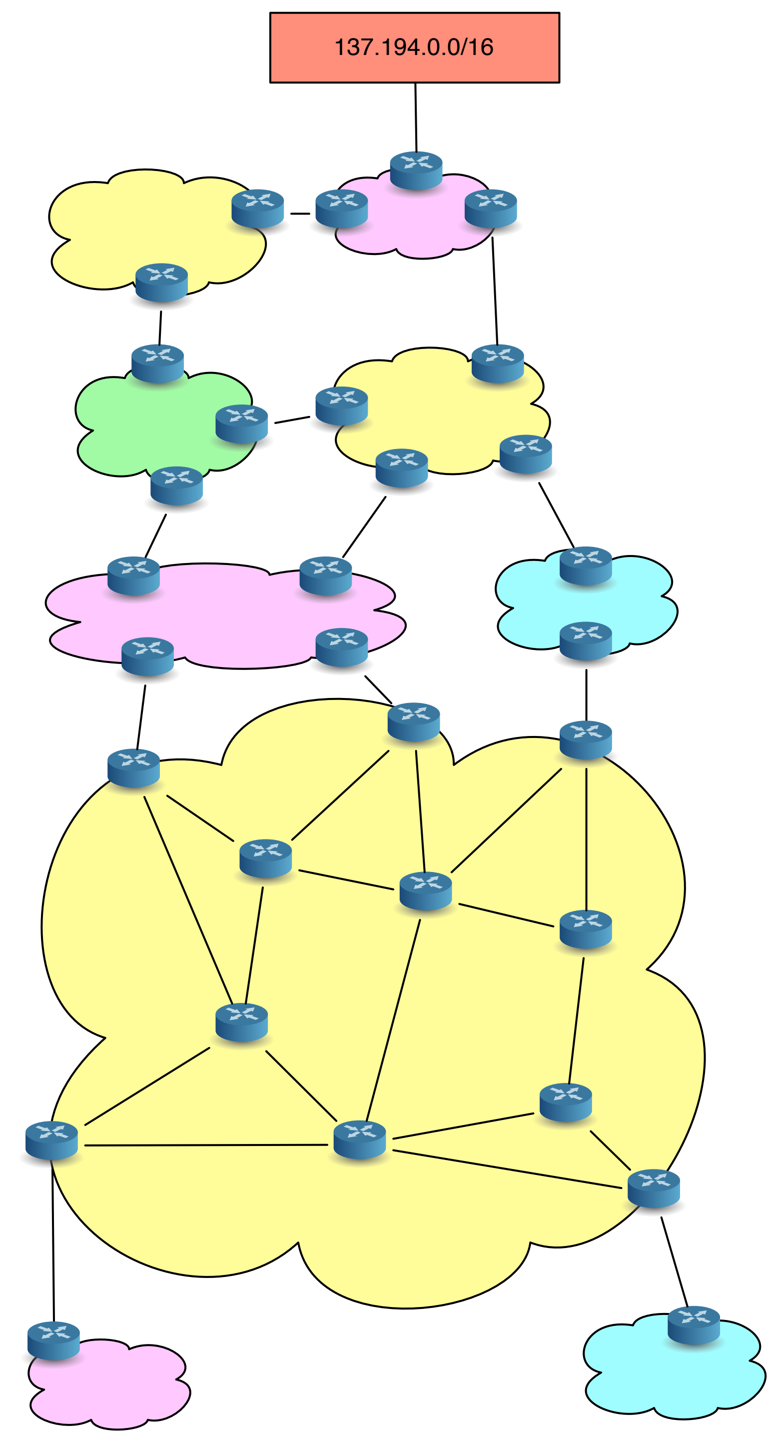
Dynamic routing

Claude Chaudet

Routing purpose

- **Configure tables in routers so that packets take the best possible path from any source to any destination**
  - One route per IP prefix
  - Shortest path with respect to a cost metric
    - number of traversed routers (hops)
    - delay
    - financial cost (peering vs. transit)
    - etc.



Routing Tables updates

Frequent changes in the routing table

- Devices addition / failures
- Routes cost evolution

How to fill/update routing tables?

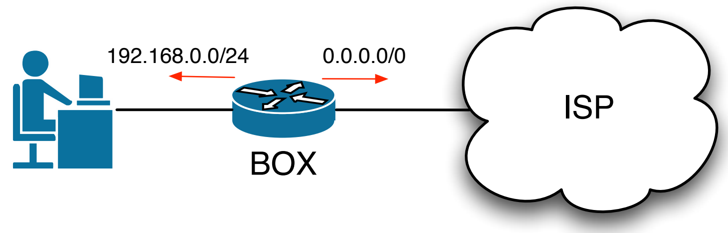
- Static routing: manual configuration by a system administrator
- Dynamic routing: automatic configuration - routers exchange data
- Each domain (Autonomous System) is free of its *internal* routing strategy

What is an efficient routing protocol?

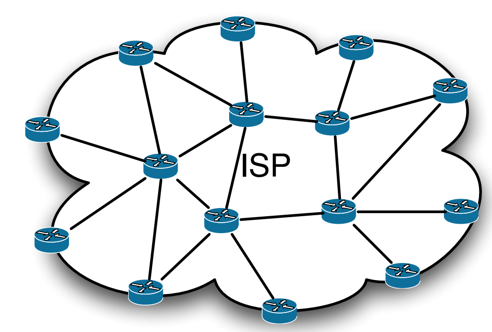
- Routes quality (length, delay, congestion, ...)
- Reaction to the topology changes (convergence speed)
- Overhead (amount of control messages necessary)
- Simplicity (easy to implement, light CPU load on routers)

Static routing

- **Base idea: each router is configured manually (cf. lab)**
  - `route add ...`
  - Once the routes are configured, they are not updated
- **Well suited for a simple network or for terminals**
  - e.g.: set-top-box: 1 router and two routes (interior network vs. rest of the world)



- **Not well adapted for a large and dynamic network (many routers, many alternate paths, ...)**
  - example: operator-level routing
    - many routers (hundreds)
    - each router contains a lot of routes (~ 200 000)



Dynamic routing

- **Automatic update of the routing tables**
  - The domain administrator defines the global policy (cost expression) and lets the network calculate routes autonomously
  - Each network modification can be detected by routers
    - Links up/down by hardware detection
    - Paths update by a dedicated protocol
  - Routers notify other routers of the changes and update routes accordingly

- **Two main routing protocols families**
  - Distance-vector routing: based on Bellman-Ford algorithm
  - Link-state routing: based on Dijkstra algorithm

Distance Vector Routing

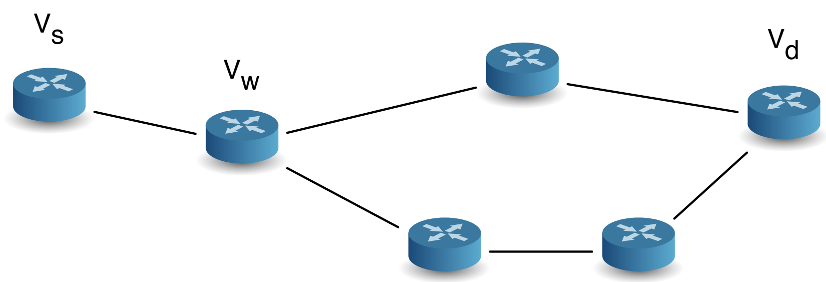
Distance vector routing

- **A router only communicates with its direct neighbors**
  - Regular emission of couples (destination; distance)
  - When receiving such a message, a router compares its own paths with the newly announced ones
  - If a better path is announced, replace the entry in the routing table
  - Infinite iteration of the process

- **Based on Bellman-Ford algorithm**
  - Complexity: O(n.m) — n = number of devices; m = number of links

Algorithmic basis: Bellman-Ford algorithm

- **G=(V,E): weighted graph that represents the network**
  - V: vertices (i.e. routers)
  - E: edges (i.e. links)
  - $w(V_a, V_b)$: weight (length, delay,...) of the edge (V_a,V_b)



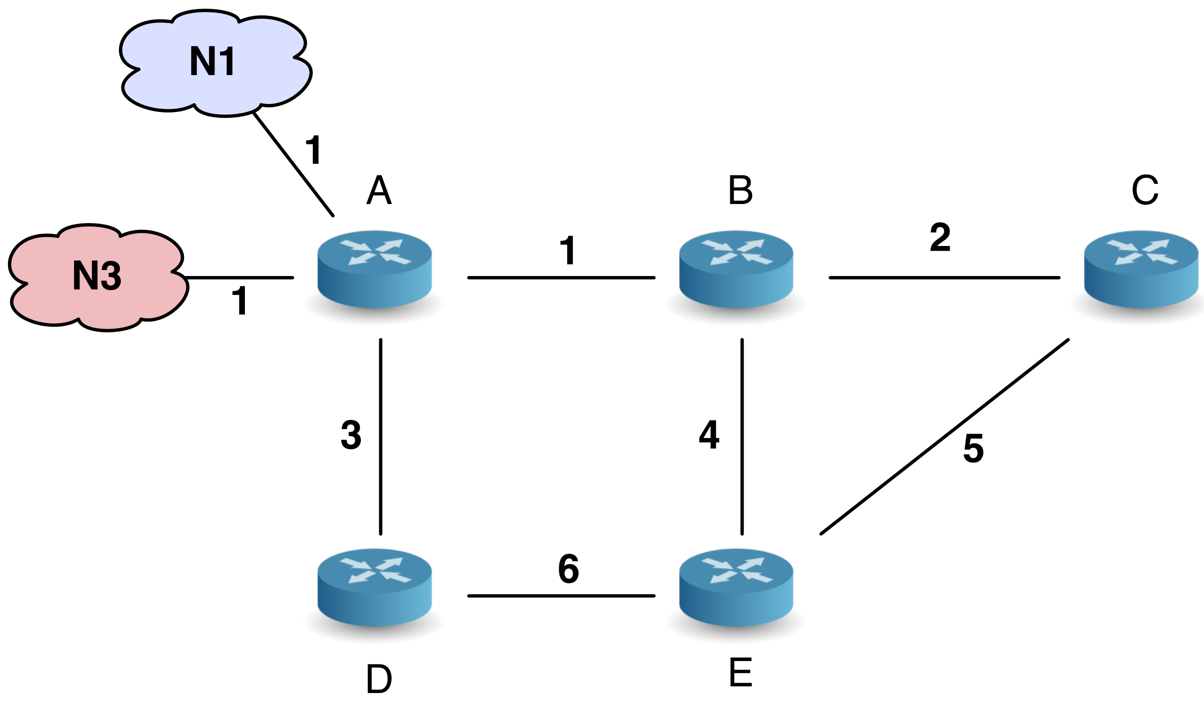
- **On router $V_s \in V$, $OPT_s(i, V_d)$ is the <u>minimal cost</u> of a path from V_s to V_d that passes through <u>at most</u> *i* edges**
  - Let P be an optimal path from V_s to V_d
  - If P uses at most $i-1$ edges, $OPT_s(i, V_d) = OPT_s(i-1, V_d)$;
  - If P uses exactly *i* edges, $\exists V_w \in V$, neighbor of V_s s.t.:
    $$OPT_s(i, V_d) = w(V_s, V_w) + OPT_w(i-1, V_d)$$
  - Finally [1]:
    - $OPT_s(i, V_d) = \min\{OPT_s(i-1, V_d), \min_{w \in V} [w(V_s, V_w)+OPT_w(i-1, V_d)]\}$ (1)
  - We start with $OPT_s(n-1, V_d)= +\infty$ and we minimize iteratively the value with equation (1).

[1] Richard Bellman: On a Routing Problem, in Quarterly of Applied Mathematics, 16(1), pp.87-90, 1958.
[2] Lestor R. Ford jr., D. R. Fulkerson: Flows in Networks, Princeton University Press, 1962.

TELECOM ParisTech

Distance Vector routing — example

- **5 routers, 6 links, heterogeneous costs**
  - 2 networks (N1 & N3) connected to router A



- **Initial routing tables:**

A		
Network	**Cost**	**Next**
N1	1	Loc
N3	1	Loc

B		
Network	**Cost**	**Next**

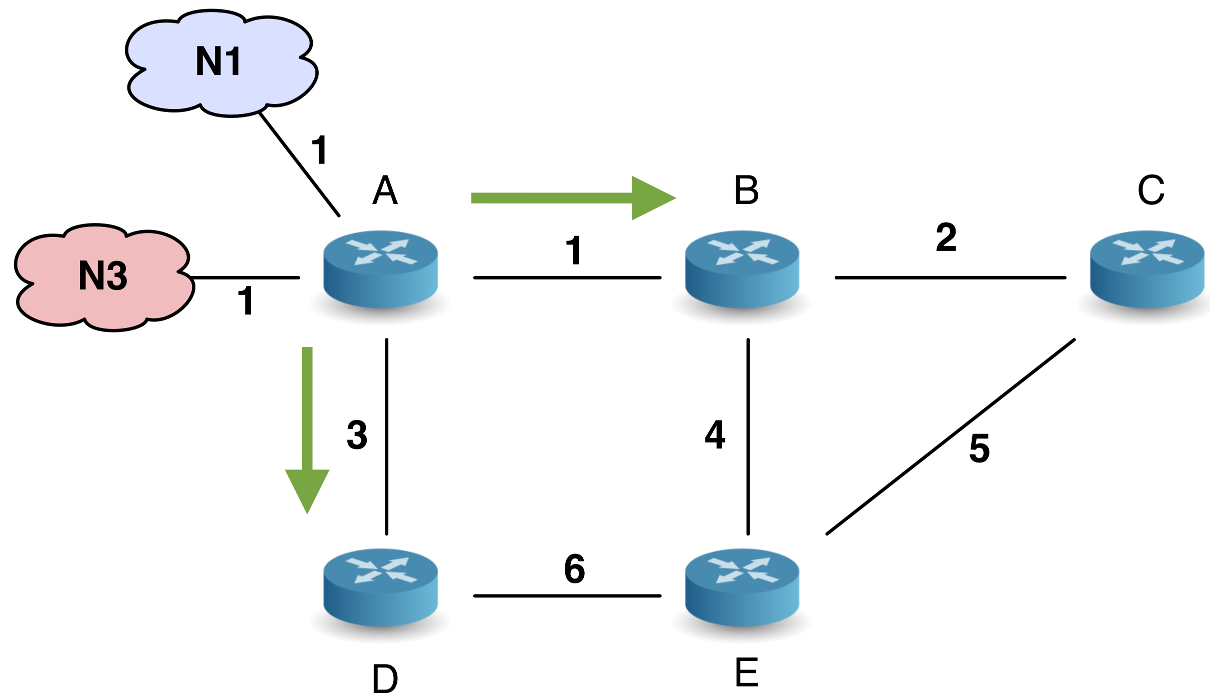
C		
Network	**Cost**	**Next**

D		
Network	**Cost**	**Next**

E		
Network	**Cost**	**Next**

Distance Vector routing — example

- **First communication step**



A		
Network	**Cost**	**Next**
N1	1	Loc
N3	1	Loc

B		
Network	**Cost**	**Next**
N1	2	A
N3	2	A

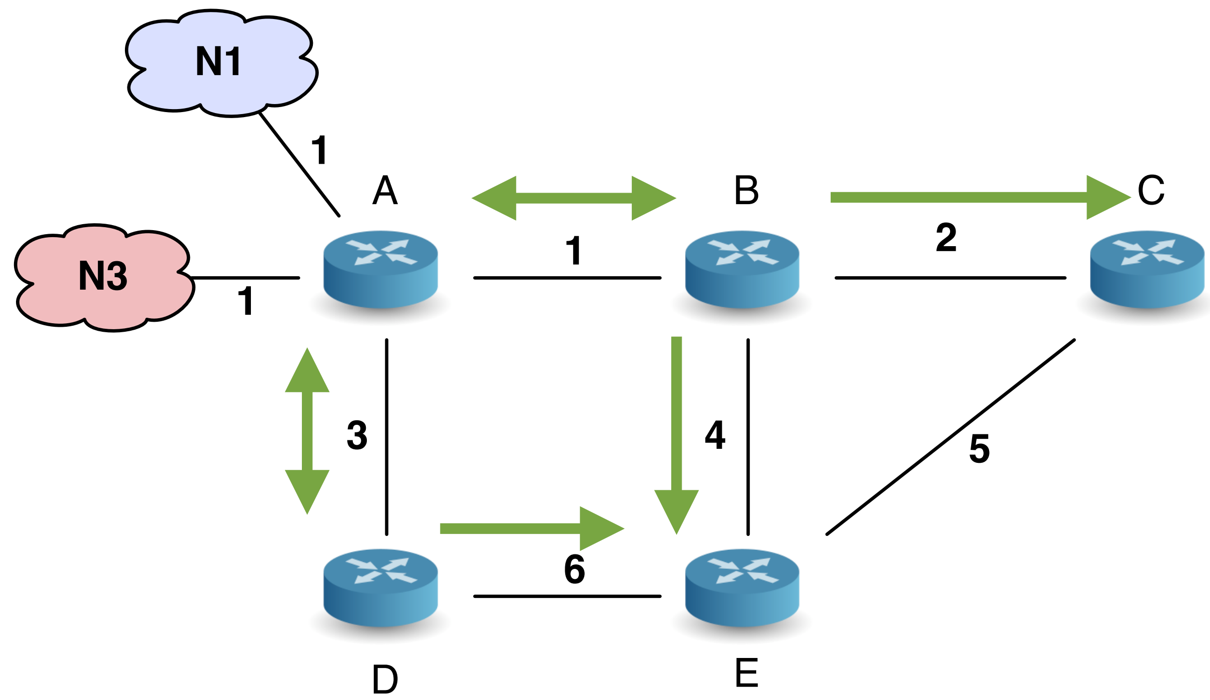
C		
Network	**Cost**	**Next**

D		
Network	**Cost**	**Next**
N1	4	A
N3	4	A

E		
Network	**Cost**	**Next**

Distance Vector routing — example

- **Third communication step**



A		
Network	**Cost**	**Next**
N1	1	Loc
N3	1	Loc

B		
Network	**Cost**	**Next**
N1	2	A
N3	2	A

C		
Network	**Cost**	**Next**
N1	4	B
N3	4	B

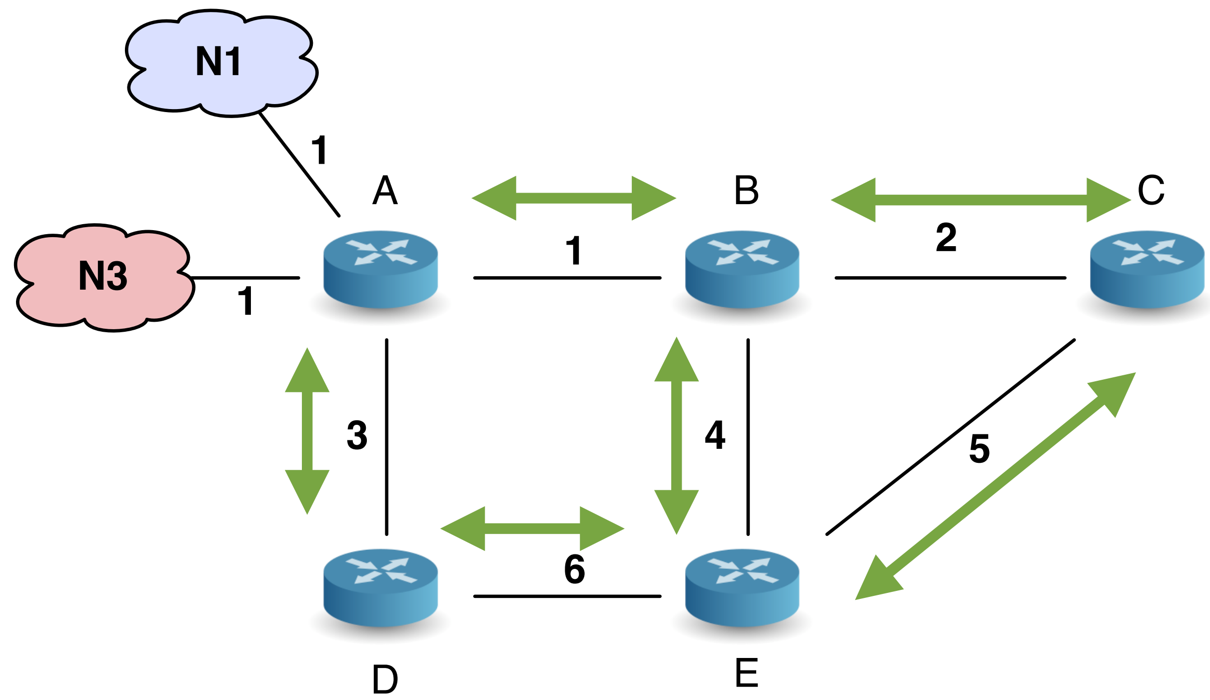
D		
Network	**Cost**	**Next**
N1	4	A
N3	4	A

E		
Network	**Cost**	**Next**
N1	6	B
N3	6	B

Distance Vector routing — example

Fourth communication step

- No modification
- The routing process has converged



A		
Network	**Cost**	**Next**
N1	1	Loc
N3	1	Loc

B		
Network	**Cost**	**Next**
N1	2	A
N3	2	A

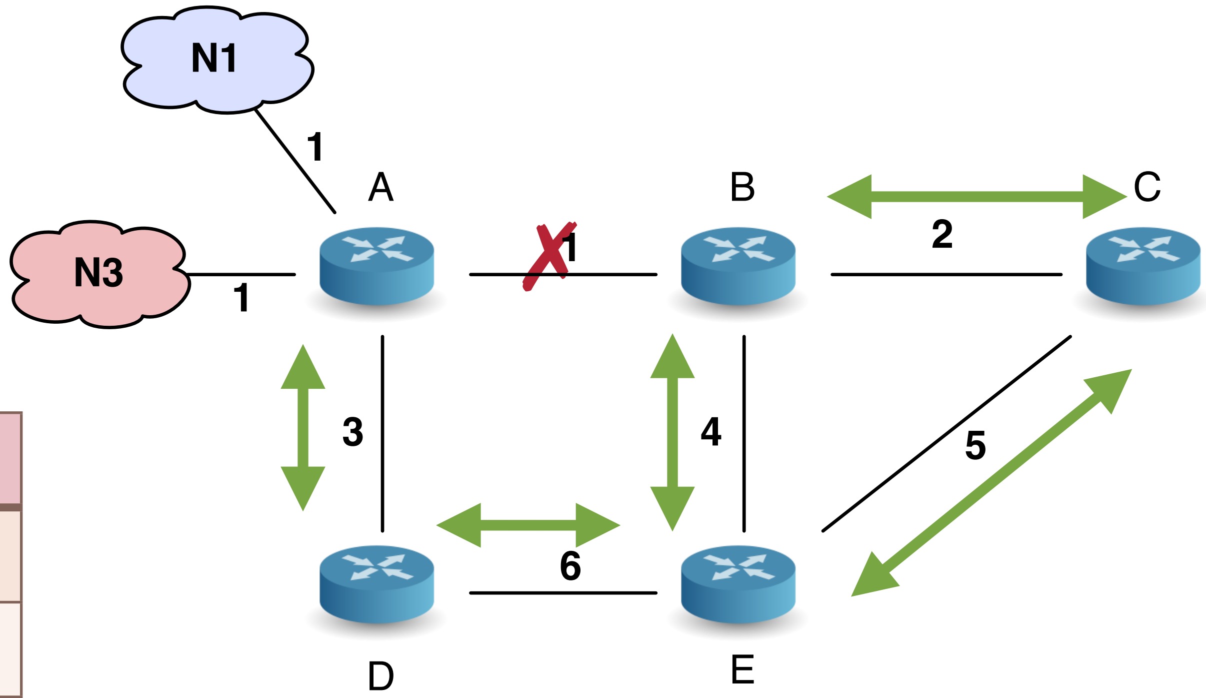
C		
Network	**Cost**	**Next**
N1	4	B
N3	4	B

D		
Network	**Cost**	**Next**
N1	4	A
N3	4	A

E		
Network	**Cost**	**Next**
N1	6	B
N3	6	B

When a link breaks

- **B detects the failure**
  - It associates an infinite cost to N1 & N3



A		
Network	**Cost**	**Next**
N1	1	Loc
N3	1	Loc

B		
Network	**Cost**	**Next**
N1	∞	—
N3	∞	—

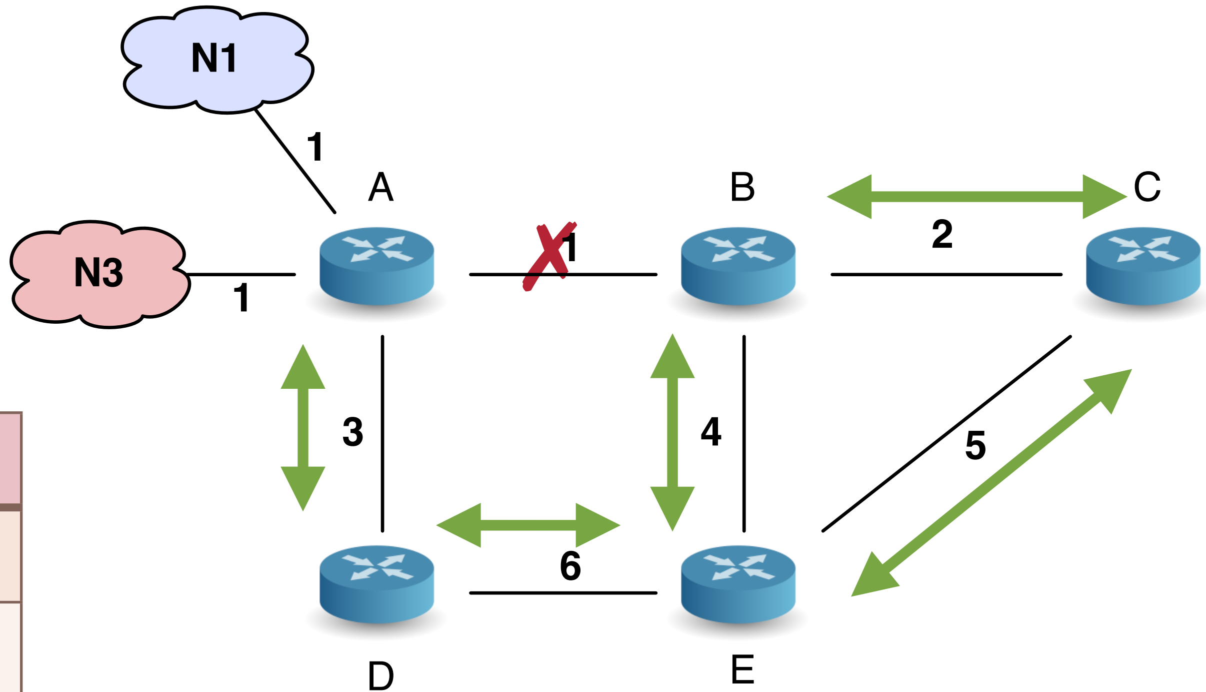
C		
Network	**Cost**	**Next**
N1	4	B
N3	4	B

D		
Network	**Cost**	**Next**
N1	4	A
N3	4	A

E		
Network	**Cost**	**Next**
N1	6	B
N3	6	B

When a link breaks

- **If the E → B message is emitted before the B → E one**
  - Potentially slow re-convergence



A		
Network	**Cost**	**Next**
N1	1	Loc
N3	1	Loc

B		
Network	**Cost**	**Next**
N1	**10**	**E**
N3	**10**	**E**

C		
Network	**Cost**	**Next**
N1	**12**	**B**
N3	**12**	**B**

D		
Network	**Cost**	**Next**
N1	4	A
N3	4	A

E		
Network	**Cost**	**Next**
N1	10	E
N3	10	E

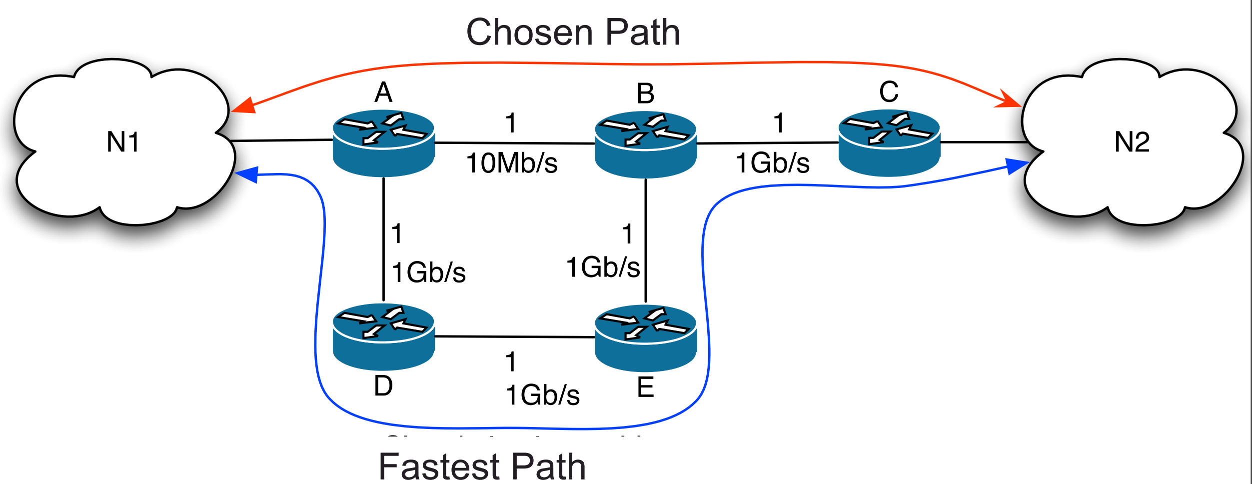
Count to Infinity problem

- **Two routers consider each other as the best next hop to a destination**
  - The algorithm shall wait until it reaches a large cost value to conclude that a route has failed

- **Some solutions (non-exhaustive list)**
  - Limit the maximal cost (limit usually quite low; e.g: 15 hops)
    - What happens in presence of routes effectively longer? Important calibration.
  - Exchange next hop address in messages
    - If a router sees itself as next hop, it will not consider the route
    - Increases messages size, hence traffic
  - Do not announce a route to a neighbor if the route passes through this neighbor (shared horizon)
    - Requires to distinguish neighbors instead of using broadcast transmission

Implémentation: RIP (*Routing Information Protocol*)

- **RFC 2453 (RIPng; 1998)**
  - A message is sent every 30 seconds
  - Maximum 25 routes in a message
  - Send to an IP multicast address (224.0.0.9)
- **In case of failure:**
  - Detection time: 180 seconds
  - Convergence time: a few minutes
- **Links weights = 1**
  - Routes are selected based on the number of hops to the destination
  - Link throughput is not considered
- **Maximum number of hops: 15**
  - Avoids loops



Link State Routing

Link State routing

- **Every router discovers its neighbors**
  - *Hello* packets exchanged regularly with neighbors

- **It creates a *Link State Packet* (LSP) containing this list alongside with associated costs**

- **LSP is transmitted to every other router who keeps the most recent update from every other node**
  - Vision of the global topology of the network

- **Transmission on particular events only**
  - New neighbor; neighbor disappeared; change of cost; ...
  - Few messages generated in a stable network
  - Every 30 minutes if nothing happened

Link State routing

- **A router knows <u>the whole network topology</u>**
  - Hello messages to identify neighbors and to measure links costs
  - Topology update messages to transmit neighborhood to all routers (usually through a multicast address)

- **Shortest path calculation algorithm (Dijkstra)**

[1] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

Algorithmic basis: Dijkstra's algorithm

G = (V,E): weighted graph that represents the network

- V: Vertices (routers)
- E: edges (links)
- w(s1,s2): weight of the edge between V1 and V2
- The weight of a path is the sum of the weights of the edges that compose the path.

For each router V_s

- V_s places itself as the root of a tree *P* (cycle-free sub-graph of G)
- V_s identifies the 1-hop neighbor that can be reached through the minimal cost edge
- This edge and this vertex are marked as selected
- The process is repeated by selecting at each step the minimal cost edge linking a selected and an unselected vertex
- The process stops when all vertices are selected (i.e. |V| steps)

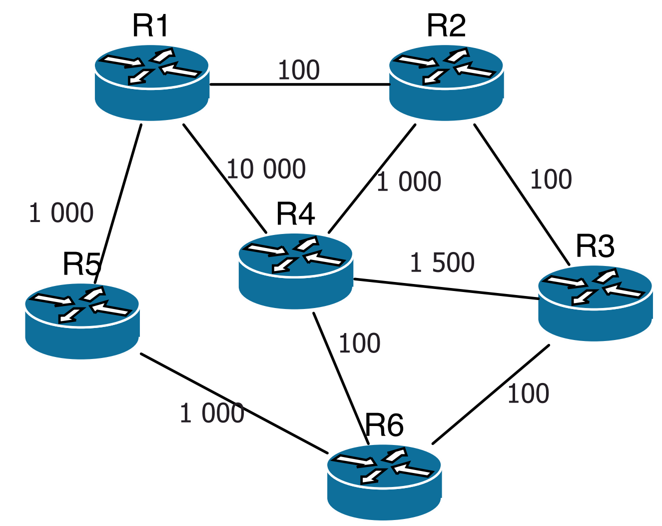
[1] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

Dijstra's algorithm - example (1)

Complexity: $O(n^2)$

- Only requires local calculation, few messages
- Fast convergence
- Good scalability

R1	R2	R3	R4	R5	R6
0	.	.	.	.	.



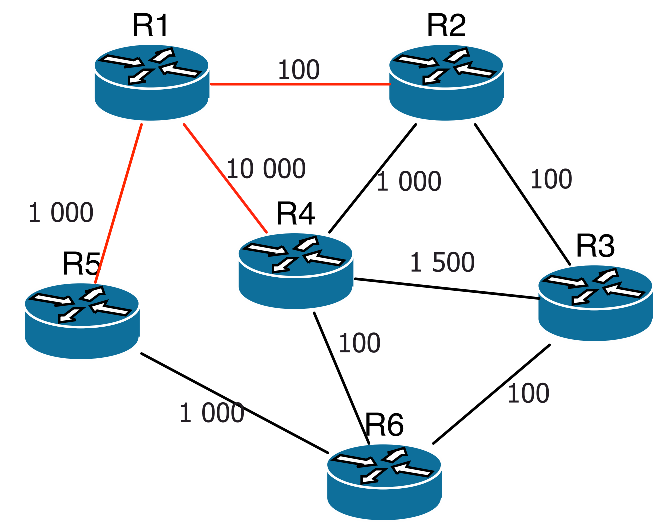
[1] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

Dijstra's algorithm - example (2)

Complexity: $O(n^2)$

- Only requires local calculation, few messages
- Fast convergence
- Good scalability

R1	R2	R3	R4	R5	R6
0	.	.	.	.	.
	100 (R2)	.	10000 (R4)	1000 (R5)	.



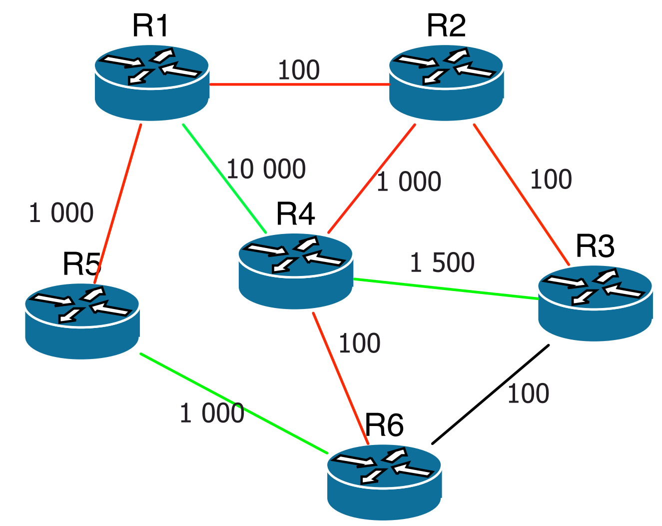
[1] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

Dijstra's algorithm - example (3)

Complexity: $O(n^2)$

- Only requires local calculation, few messages
- Fast convergence
- Good scalability

R1	R2	R3	R4	R5	R6
0	.	.	.	.	.
.	100 (R2)	.	10000 (R4)	1000 (R5)	.
.	.	200 (R2)	1 100 (R2)	.	1 200 (R2)



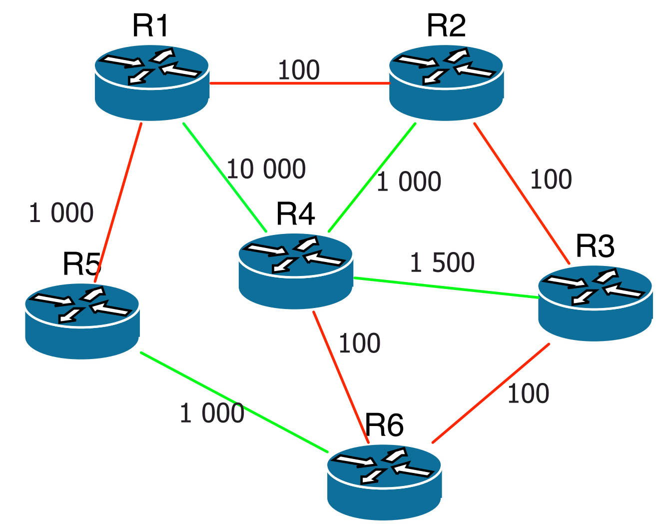
[1] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

Dijstra's algorithm - example (4)

Complexity: O(n²)

- Only requires local calculation, few messages
- Fast convergence
- Good scalability

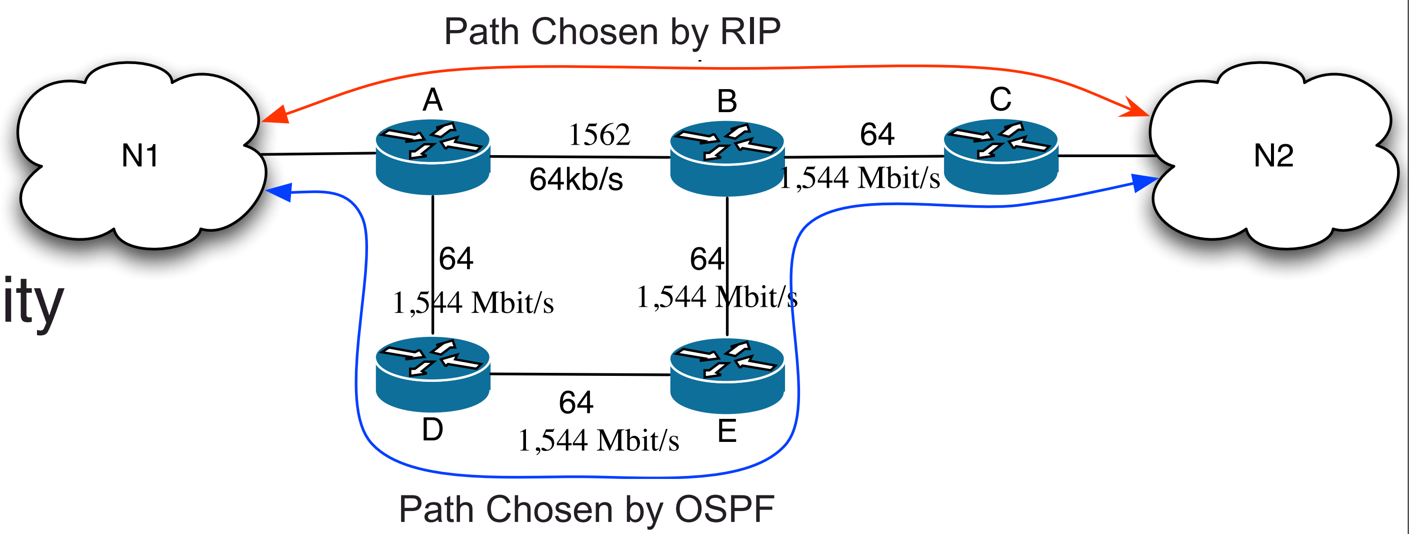
R1	R2	R3	R4	R5	R6
0	.	.	.	.	.
.	100 (R2)	.	10000 (R4)	1000 (R5)	.
.	.	200 (R2)	1 100 (R2)	.	1 200 (R2)
.	.	.	400 (R6)	.	300 (R3)



[1] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269–271, 1959.

Example: OSPF (Open shortest path first)

- **RFC 3740 (OSPF v3 - 1999)**
- **In case of failure:**
  - Convergence time around 1 sec (depends on the flooding time)



- **Links weights**
  - Depends on the link capacity
  - Weight = Reference capacity / true capacity
- **Maximum number of hops**
  - No limit
  - Each router knowns the full topology.
- **Complexity larger than RIP**
  - The network is divided in *areas* (divide and conquer)

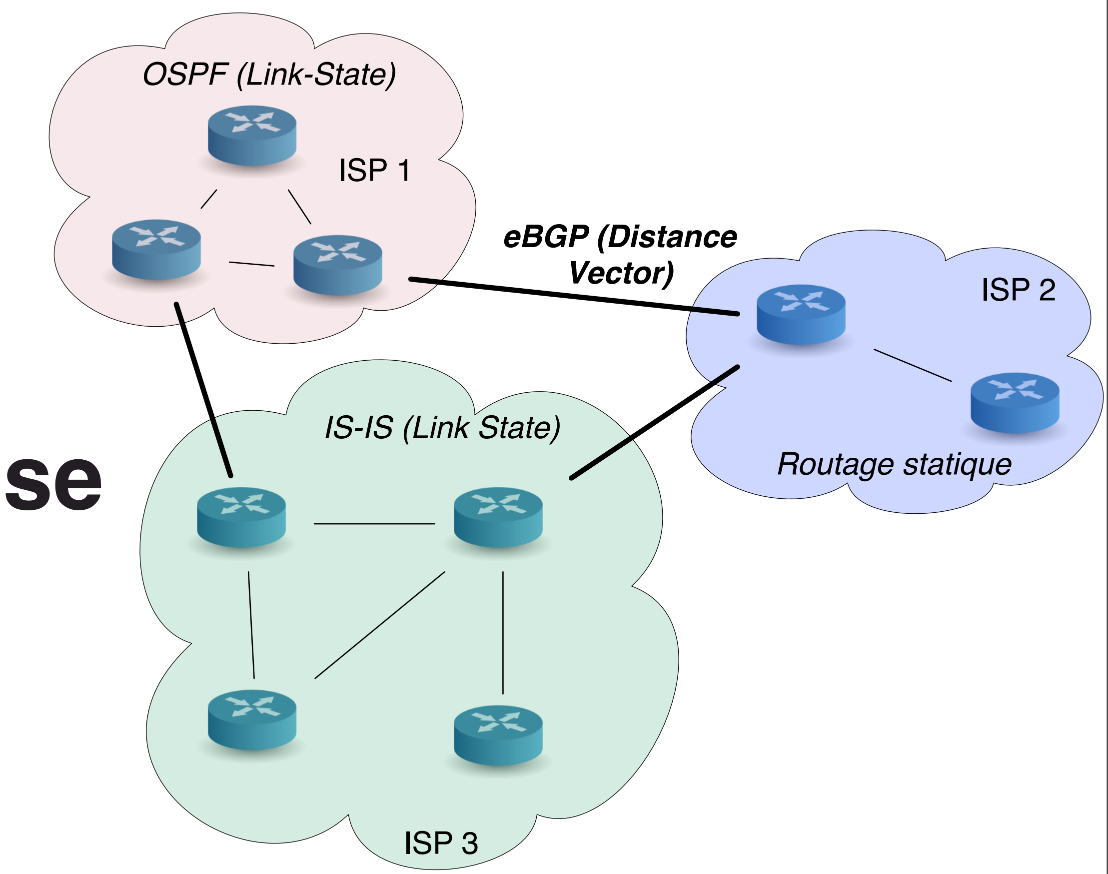
Internal routing protocols (IGPs)

- **RIP**
  - Distance vector routing ; CIDR-compatible (VLSM)
- **OSPF**
  - Link-state routing ; CIDR-compatible (VLSM)
  - More popular in large companies
- **IS-IS**
  - Link-state routing ; CIDR-compatible (VLSM)
  - Hierarchization of routers (inside an area vs. between areas)
  - More popular at ISPs
  - Multi-protocol (not limited to IP and does not use IP for control packets transmission)
- **EIGRP (Cisco)**
  - hybrid protocol (distance vector basis with distinction for the neighbor routers)
  - Compound metric (mixes delay, capacity, reliability, load)

Inter-domain routing

BGP: Inter-domain routing

- **RIP et OSPF are IGP (*Interior Gateway Protocol*)**
  - Their usage is bounded to a domain (AS)



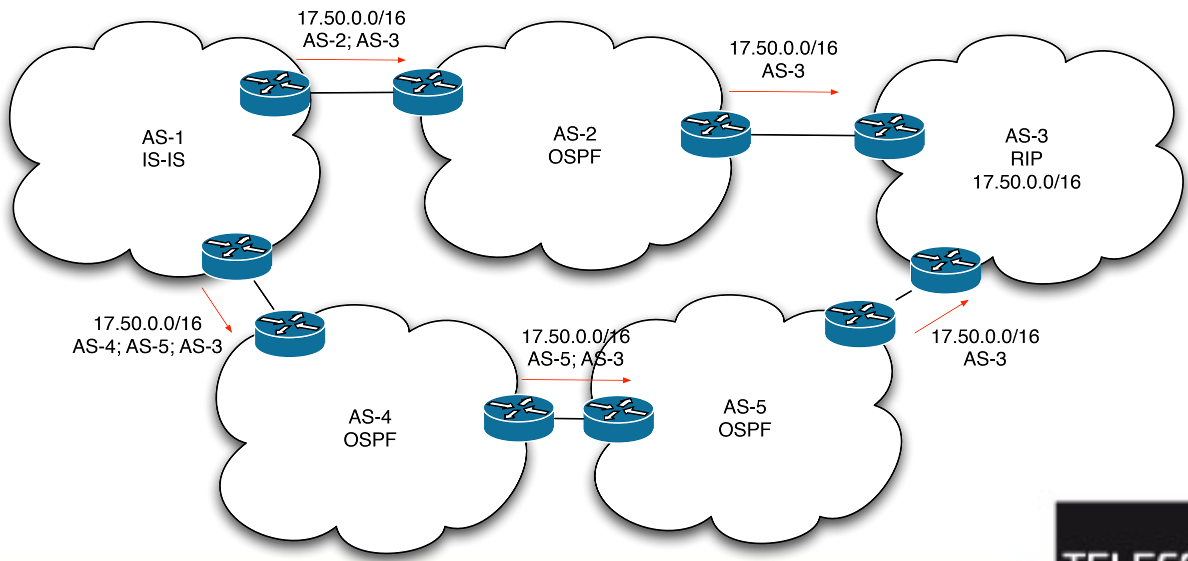
- **ISPs & companies are free to choose their internal routing policy**
  - Most often, static or link-state routing (OSPF, IS-IS)
- **For external routing, (between ISP), a single standard protocol: BGP 4 (RFC 4271)**
  - Path vector routing
  - Implemented between AS (eBGP) and between edge routers of a single AS (iBGP)

Between AS: eBGP

- **eBGP:**
  - point-to-point connection (unicast) between close routers
  - A router announces the accessible prefixes (i.e. the ones the AS accepts to route) and the AS-paths (list of traversed AS)

- **eBGP does not utilize explicit costs**
  - Choice of the best route for a destination based on:
    - Routing policies (e.g. prefer peering links over transit links)
    - AS-path: pass by the lowest number of AS

- **The IGP is hidden from other AS**
  - Mutual trust between close AS

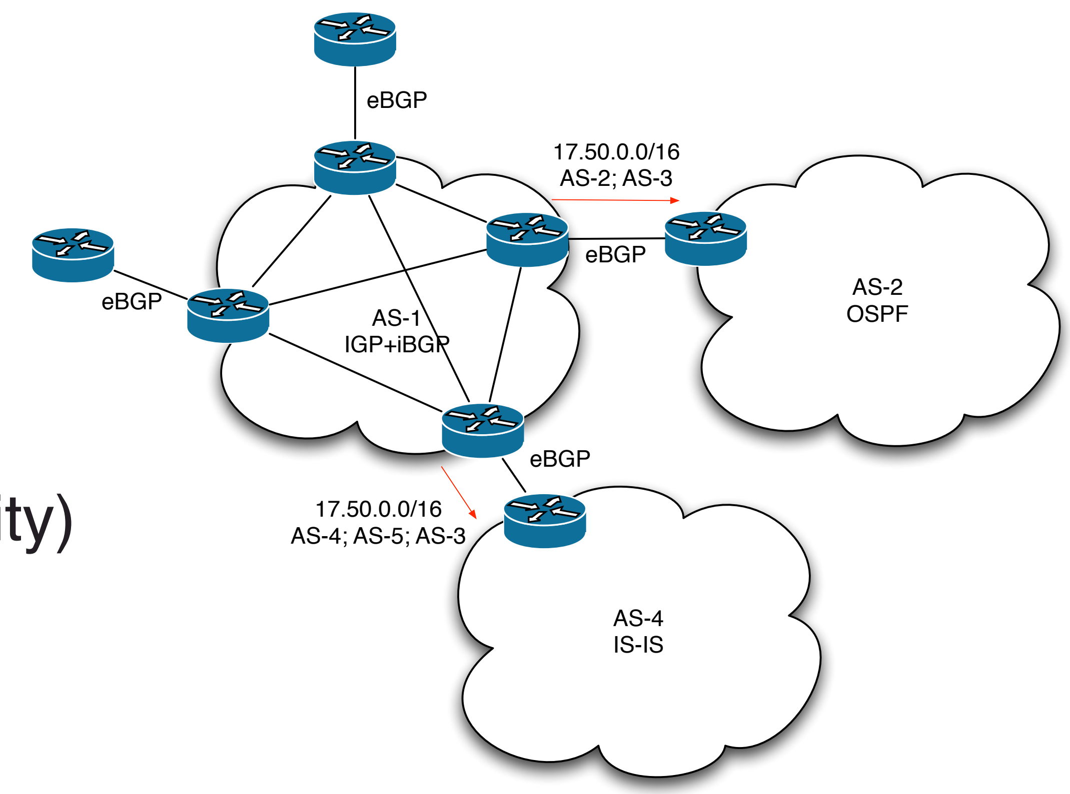


Inside an AS: iBGP

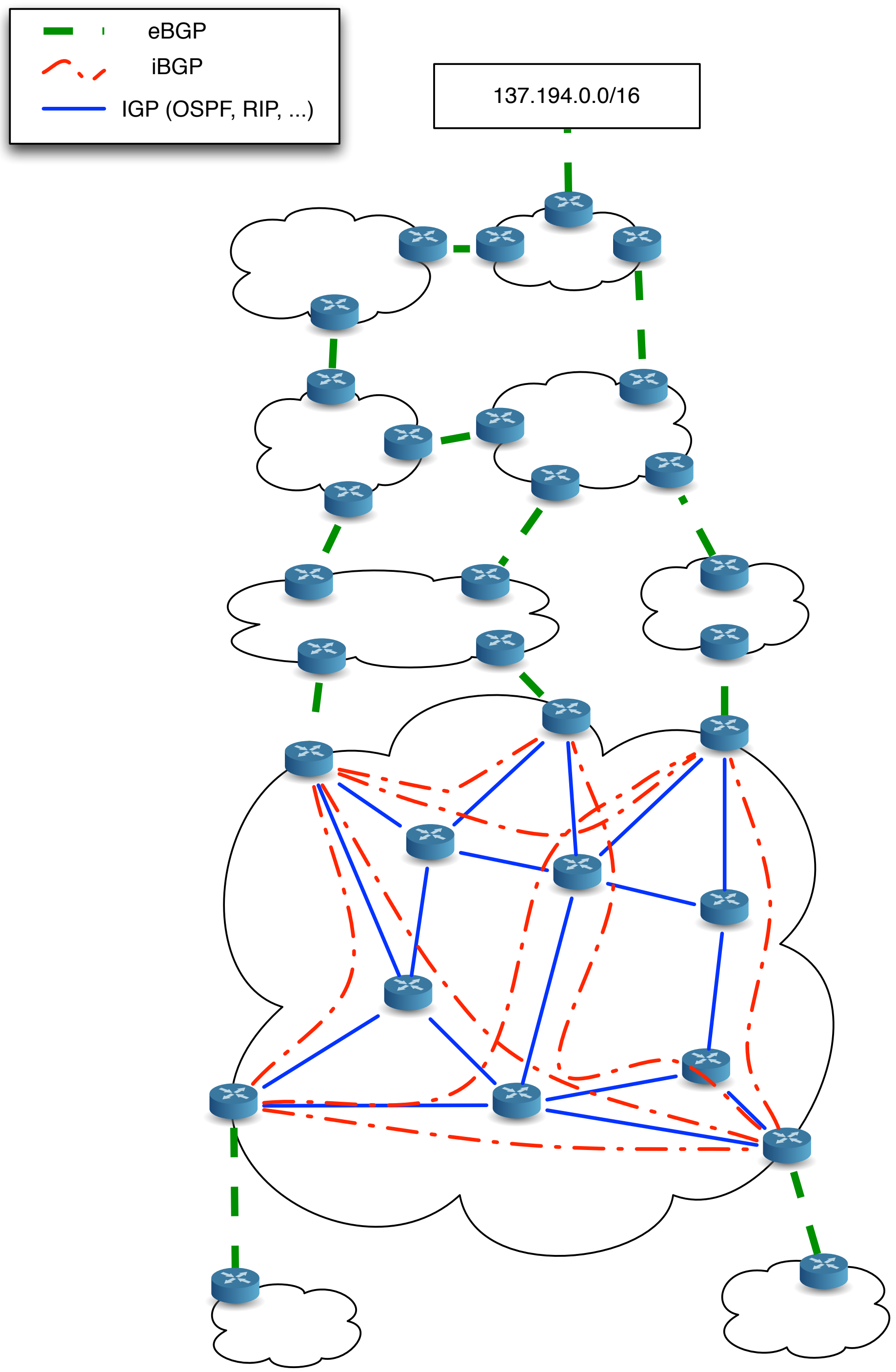
- **iBGP: border routers belonging to the same AS exchange routing information**
  - Share information to take a decision on how to reach any IP prefix at the AS level

- **Differences between iBGP and eBGP**
  - eBGP works over dedicated links
  - iBGP routers are separated by a whole network (switches, routers, ...)

  - Unicast connections in both cases
    - eBGP: single link, single peer
    - iBGP: Mesh network of all routers (strong connexity)



Conclusion



- **Internet routing happens at multiple scales**
  - eBGP between AS
  - iBGP between border routers of an AS, for external prefixes
  - arbitrary IGP (RIP, OSPF, static, ...) inside an AS
- **IGP: various strategies**
  - Distance vector vs. link state vs...