Δίκτυα Υπολογιστών

Πρόγραμμα Μεταπτυχιακών Σπουδών:
Τεχνο – Οικονομικά Συστήματα

Καθηγητής Συμεών Παπαβασιλείου

Εθνικό Μετσόβιο Πολυτεχνείο
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής & Συστημάτων Πληροφορικής

(E-mail: papavass@mail.ntua.gr Τηλ: 210 772-2550
Γραφείο: Β.3.15 Νέο Κτίριο Ηλεκτρολόγων)

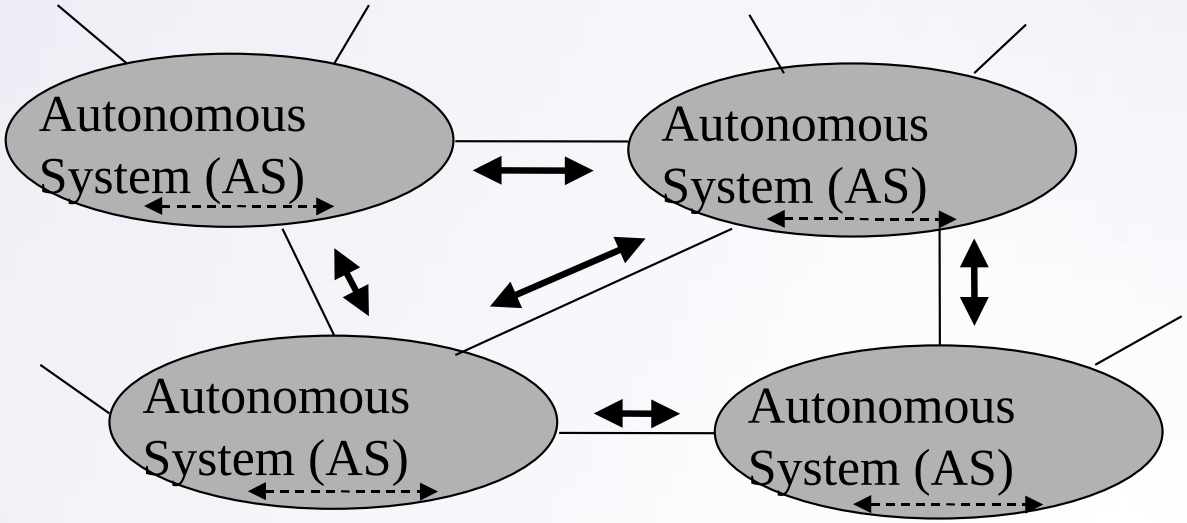
Βιβλίο Αναφοράς: TCP/IP Illustrated, Vol. I, by W.R. Stevens (Addison Wesley)

ΔΡΟΜΟΛΟΓΗΣΗ ΣΤΟ ΔΙΑΔΙΚΤΥΟ – ΑΛΓΟΡΙΘΜΟΙ

Routing

- Internet Routing Hierarchy and Autonomous Systems
- IP Routing is done on hop-by-hop basis
- Routing Mechanism: procedure to search the routing table
- Routing Policy: procedure to determine which routes go into the routing table (Routing Protocols)
- Here we emphasize on how to calculate and set up the routing tables
- Routing Algorithms
  - Distance Vector
  - Link State
- IP Routing Protocols (RIP, OSPF)

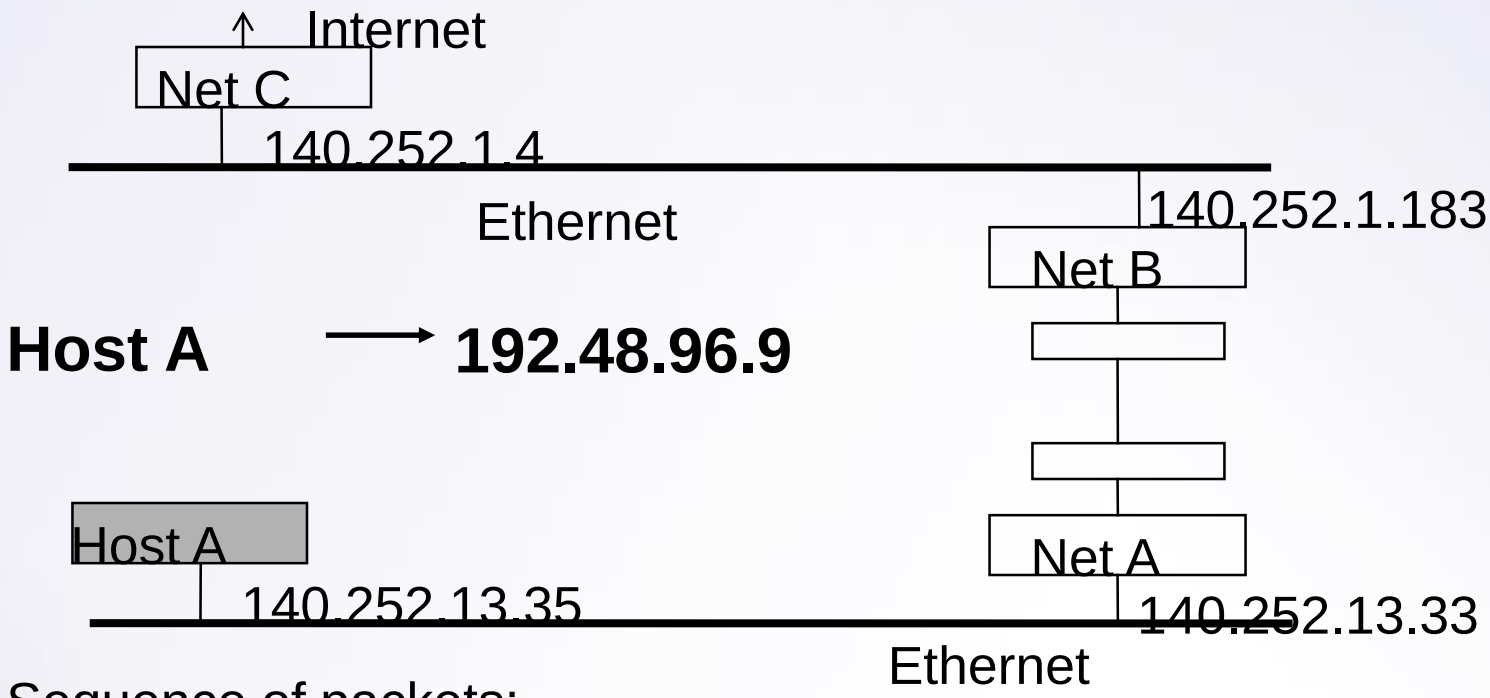
Routing Hierarchy



Interior Routing Protocol (intradomain routing)
Exterior Routing Protocol (interdomain routing)

An Autonomous System is a region of the Internet that is administered by a single entity. Routing is done differently within an autonomous system (intradomain routing) and between autonomous systems (interdomain routing)

IP Routing Example



Sequence of packets:

HostA ---> NetA	Dest_IP_addr=192.48.96.9 Dest_Eth_addr=Eth_addr(NetA)
NetA ---> NetB	Dest_IP_addr=192.48.96.9
NetB ---> NetC	Dest_IP_addr=192.48.96.9 Dest_Eth_addr=Eth_addr(NetC)

Intradomain and Interdomain Routing

- **Intradomain Routing**
  - Routing with an AS
  - Ignores the internet outside the AS
  - Interior Gateway Protocols (IGPs) (such as RIP and OSPF)

Interdomain Routing

- Routing between ASs

- Assumes that the Internet is a collection of interconnected ASs

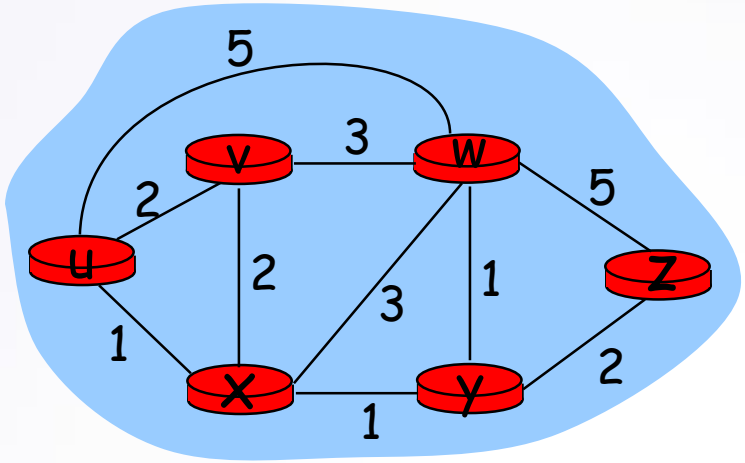
- Exterior Gateway Protocols (EGPs)

- Usually there is one (or two for backup purposes) that handles interdomain traffic in each AS

Αλγόριθμοι δρομολόγησης

Γράφος δικτύου

Για τη διατύπωση του αλγορίθμου δρομολόγησης χρησιμοποιείται ο γράφος του δικτύου.



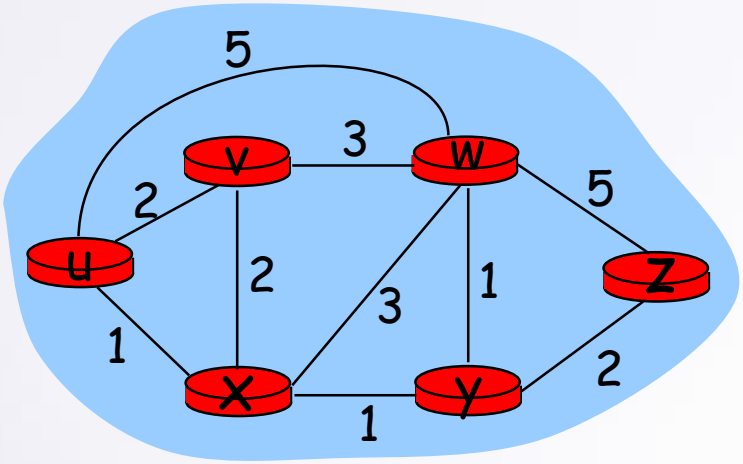
Γράφος: G = (N,E)

Σύνολο δρομολογητών N = { u, v, w, x, y, z }

Σύνολο ζεύξεων E ={ (u,v),(u,x),(u,w),(v,x),(v,w),(x,w),(x,y),(w,y),(w,z),(y,z) }

Αλγόριθμοι δρομολόγησης

Γράφος δικτύου: κόστη



- c(x,x′) = κόστος ζεύξης (x,x′)

  - π.χ., c(w,z) = 5

- το κόστος μπορεί να είναι πάντα 1, ή αντιστρόφως ανάλογο του εύρους ζώνης, ή ανάλογο της συμφόρησης

Κόστος διαδρομής $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Ποια είναι η διαδρομή ελάχιστου κόστους μεταξύ u και z ;

Ο αλγόριθμος δρομολόγησης βρίσκει τη διαδρομή ελαχίστου κόστους.

Αλγόριθμοι δρομολόγησης

Ταξινόμηση

Προέλευση της πληροφορίας δρομολόγησης:

Τοπική:

- Οι διαδρομές υπολογίζονται μόνο με βάση την τοπική τοπολογία και κίνηση.
- Ο δρομολογητής γνωρίζει τους φυσικά συνδεδεμένους γείτονες και το κόστος των ζεύξεων προς τους γείτονες.
- Επαναληπτική διαδικασία υπολογισμού, ανταλλαγή πληροφοριών με τους γείτονες.
- Αλγόριθμοι διανύσματος αποστάσεων

Καθολική:

- Οι διαδρομές υπολογίζονται με βάση την συνολική τοπολογία.
- Όλοι οι δρομολογητές ξέρουν την πλήρη τοπολογία και το κόστος των ζεύξεων.
- Αλγόριθμοι κατάστασης ζεύξεων

Routing Algorithms

- Routing algorithms calculate the route with the “least cost”
- Main components:
  - Calculate Cost (cost could be function of: hops, delay, throughput etc.)
  - Disseminate routing information (nodes disseminate their costs measures to other nodes)
  - Calculate routing tables (nodes run some form of least cost routing to (re)calculate routes)

Routing Protocols

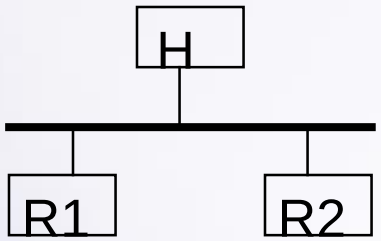
- Routers need to understand the topology of the network in order to route packets properly
- Routers need to tell to each other which paths are “up” and what costs are involved
- Routers communicate with each other using routing protocols
- Two broad classes of routing policies:
  - Static: All routes are listed explicitly; the router does not communicate with other routers
  - Dynamic: routing protocols are used to enable routers to react to changes in network topology without manual intervention

Static Routing

- When an interface is initialized the direct routes are automatically created
- Routes to hosts and networks not directly connected must be entered in the routing table (i.e. manually by command route, e.g. route add host_A router_B)
- Static routing is good in small networks
- Networks should usually not mix static and dynamic routing
  - Dynamic routing information may conflict with static routing
  - Point-to-point links may be routed with static routes

ICMP Redirect Messages

- An ICMP Redirect error is sent by a router to the sender of an IP datagram when the datagram should have been sent to a different router.
- This procedure helps hosts to build better routing tables by time (they can start just with default routing)



a. H --> R1 (i.e. default)
b. R1 receives and: R1 --> R2
c. R1 detects that it sends the packet on the same interface that received it
d. R1 sends ICMP Redirect to H informing it to send future packets to R2 instead

Dynamic Routing Protocols

- Two classes of dynamic routing protocols
- **Distance Vector (DV):** Routers know local information and broadcast it to other neighboring routers
  - Each node knows the distance to all other nodes
  - A node sends a list to its neighbors with the shortest distances to all nodes
  - If all nodes update their distances, the routing tables eventually converge

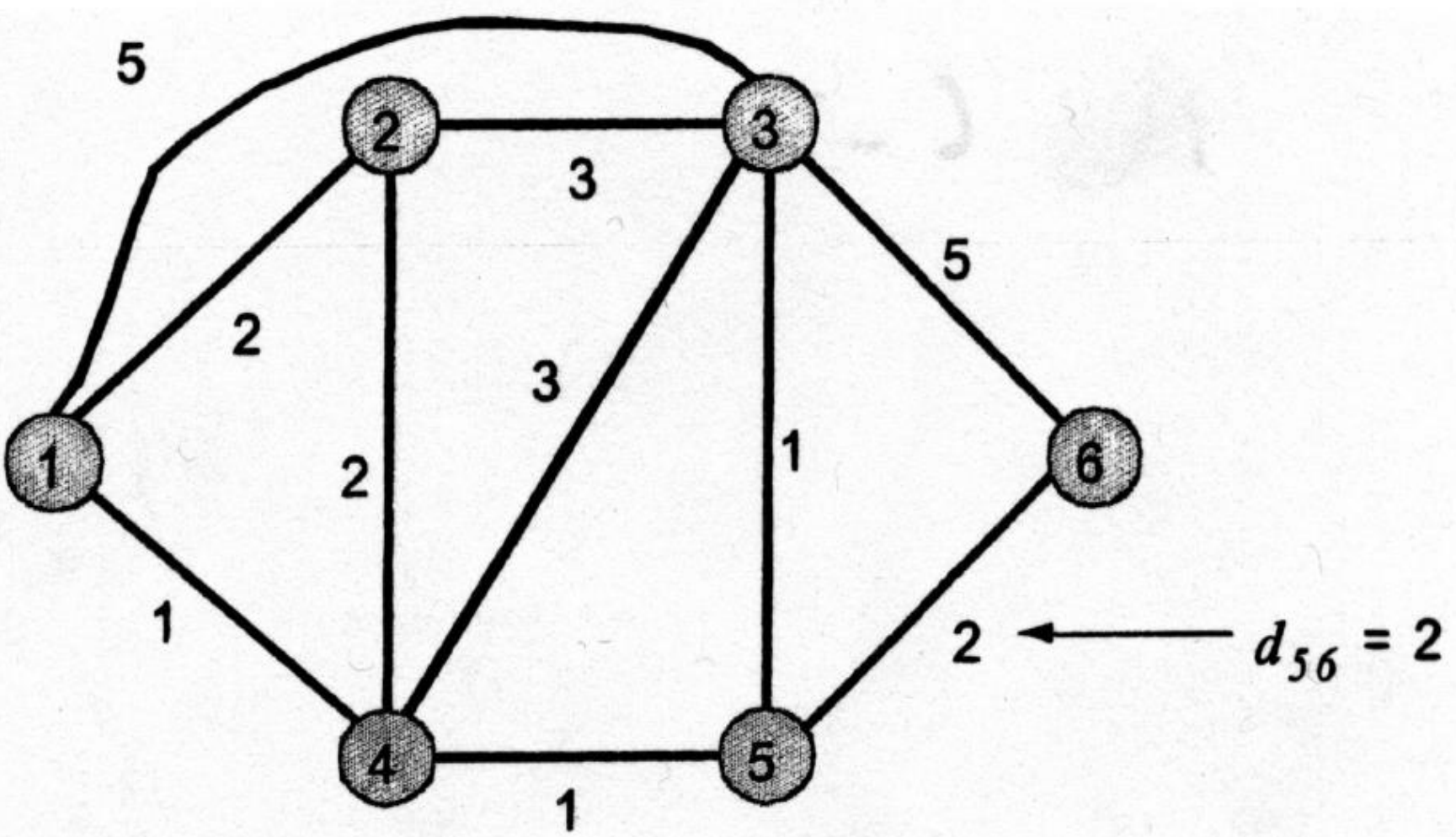
Dynamic Routing Protocols Cont…

- **Link State (LS): Routers know state of the entire network**
  - Link state information is broadcast to all nodes
  - Each node knows the topology of the entire network
  - Each node calculates the routing tables independently (by using a shortest-path algorithm on network topology)

Parameters

- Parameters:
  - dij: Cost of link between node i and node j
  - dij=∞ if nodes i and j are not connected
  - dii=0
  - N: set of nodes
- Goal: Given a network where each link between two nodes i and j is assigned a cost, find the path with the least cost between nodes s (source) and d (destination)

Network Example



Distance Vector Routing

- Distance Vector routing requires each router to maintain the distance from itself to every possible destination.
- Each node maintains two tables: **Distance table** (cost to each node via each outgoing link) and **Routing Table** (*minimum* cost to each node and next hop node)
- For Routing Table each router maintains a list of routes in the form (D,A)
  - D is the distance to the listed network or host
  - A is the destination host or net address
- Each router sends a vector of distance information to its neighbors. The distances are computed using the information in neighbors' distance vectors
- Depending on implementation, distances may be hops, or some other measure of distance

Distance Vector Algorithm Operation

Each router:

- is configured with its own ID and a number to use as the cost of each link
- starts with a distance vector consisting of zero for itself and the value infinity for every other destination
- transmits its distance vector to each of its neighbors whenever the information changes (or at regular updates)
- saves the most recently received distance vector from each of its neighbors
- calculates its own distance vector, based on minimizing the cost to each destination, by examining the cost to that destination reported by each neighbor and then adding the cost of its link to that neighbor

Distance Vector Algorithm: Tables

NODE V

DISTANCE TABLE

to \ via	w	n
d	$C_d(v,w)$	$C_d(v,n)$

ROUTING TABLE

to	via	cost
d	n	$D_d(v)$

$l(w,v)$ cost of link (w,v)

$C_d(v,w)$ cost from v to d via w

$D_d(v)$ minimum cost from v to d

Distance and Routing Tables Explanation

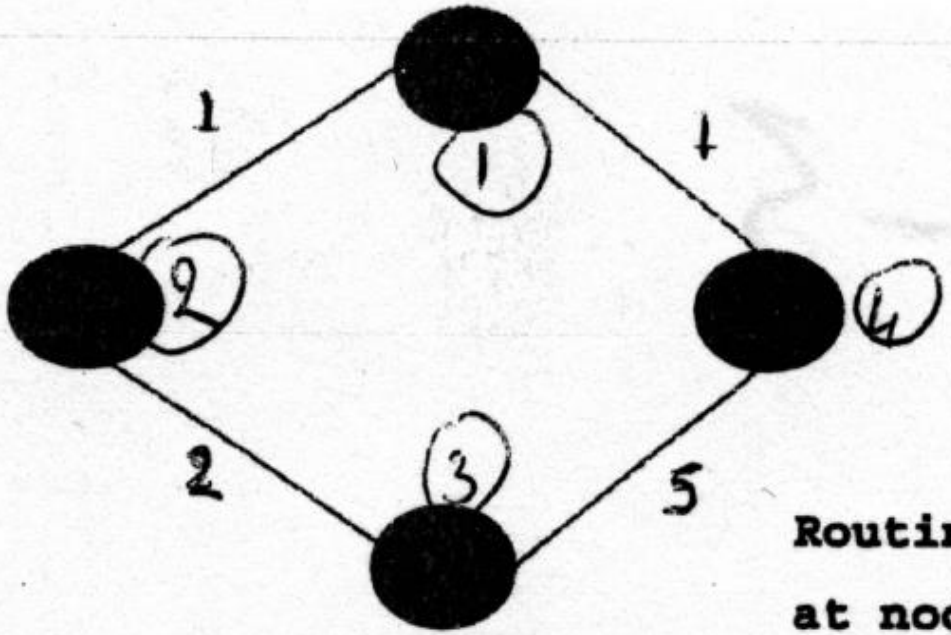
Distance table at a node N

- Not counting header row, this table has as many rows as there are nodes in the network -1
- Not counting the first column (which just lists all other nodes in the network), the number of columns is equal to the number of neighbors of node N

Routing table at node N

- Not counting header row, this table has as many rows as there are nodes in the network -1
- Not counting the first column, the number of columns is always 2 (one for cost and the other for the via node)

Example



Distance table at node 2

via / to	1	3
1	1	5
3	4	2
4	2	6

Routing table at node 2

to	via	cost
1	1	1
3	3	2
4	1	2

Messages

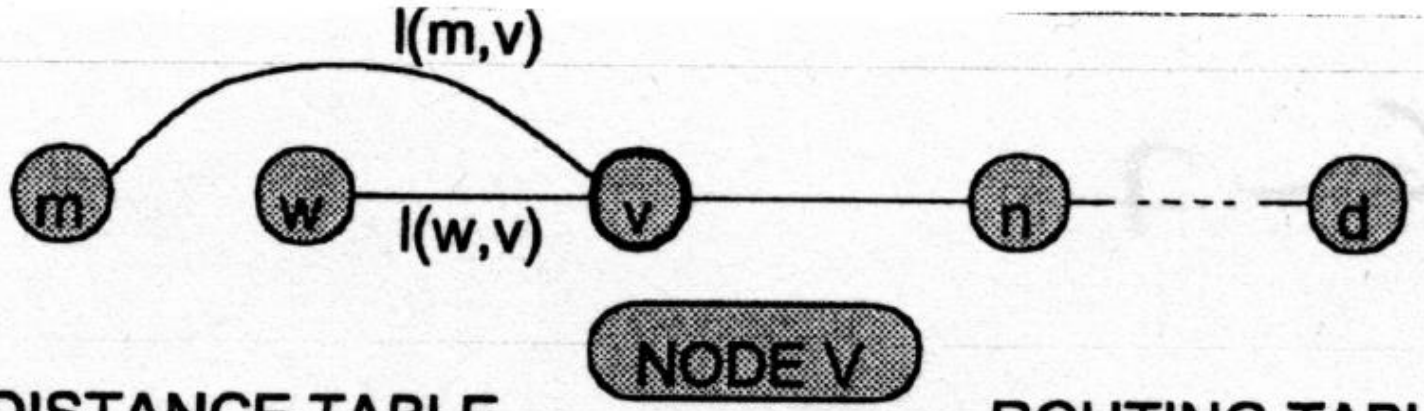
- Nodes exchange messages with their neighbors.
- If node v sends a message of the form:

$$[m , D_m (v)]$$

I can go to node m with minimum cost $D_m (v)$

- This message is only of interest to neighbors of v.

Insertion of new link with cost l(m,v)



DISTANCE TABLE

to \ via	w	n	m
m	$C_m(v,w)$	$C_m(v,n)$	$I(m,v)$

New column

ROUTING TABLE

to	via	cost
m	m	$D_m(v)$

Node's v operations when new link (m,v) is inserted

1. Add a new column in the distance table for node m
2. Add "l(m,v)" to distance table in the entry corresponding to row m/col. m.
3. Compute $\min_w C_m(v,w)$: for all nodes w

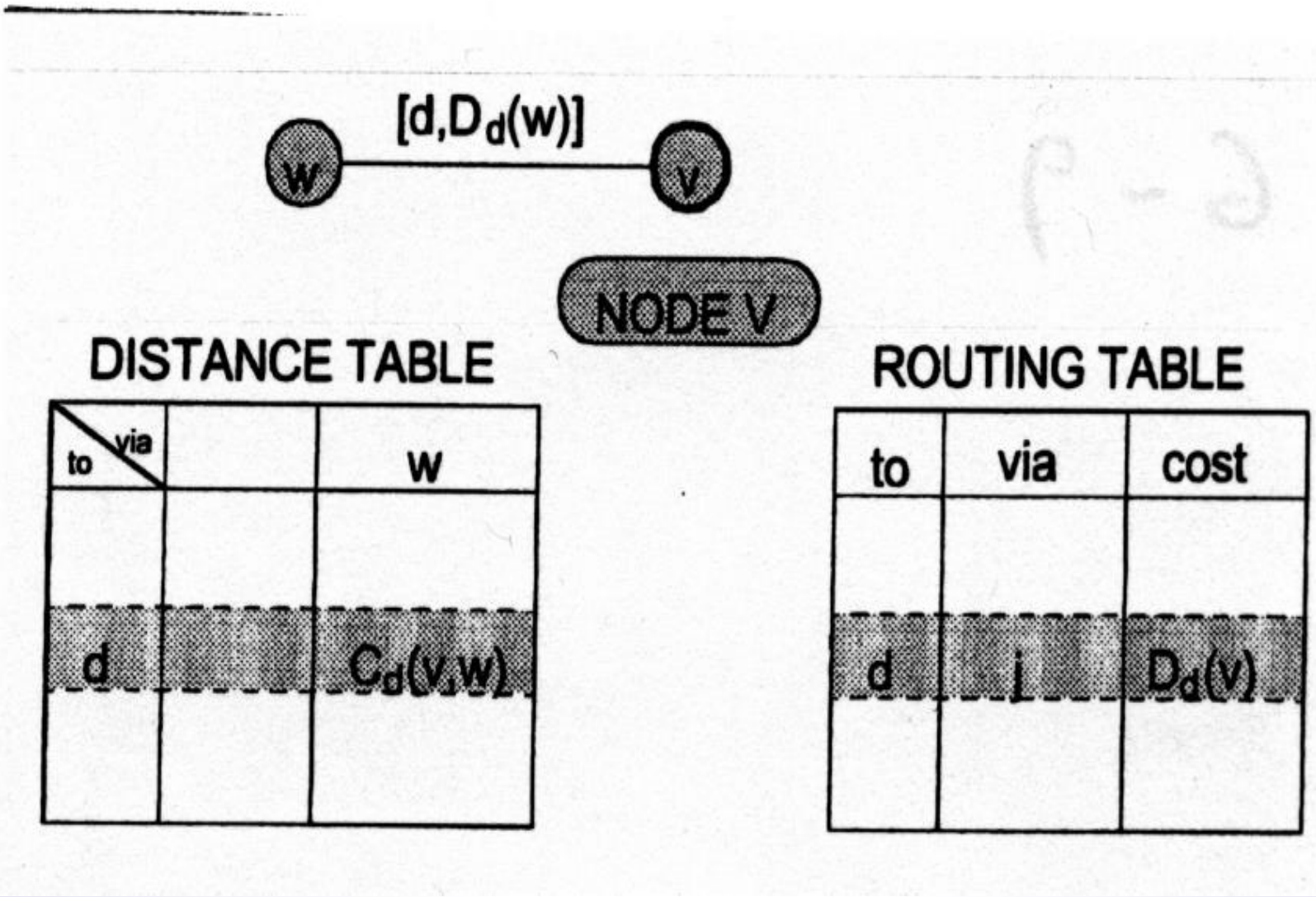
   (a)If no changes to previous value of $\min_w C_m(v,w)$:

   Do nothing

   (b) If $C_m(v, m) = \min_w C_m(v,w)$

   $D_m(v)=C_m(v,m)$ change entry in m-th row of routing table to $(m,C_m(v,m)$ and send message $[m, D_m(v)]$ to all neighbors.
4. Also: Since v is a neighbor of m, it sends the contents of its routing table to m: $[a, D_a(v)]$, $[b, D_b(v)]$, ..., $[z, D_z(v)]$

Reception of message $[d,D_d(w)]$ by node v

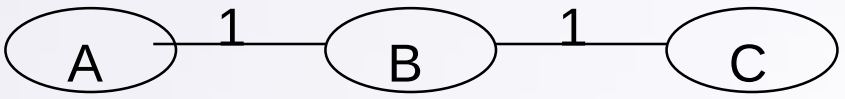


Operations at node v due to reception of message [d,$D_d(w)$]

Operations at node v

1. If d = v then ignore the message.
2. If d ≠ v then
   - $C_d(v, w) = D_d(w) + l(w,v)$
   - Compute $\min_x C_d(v,x)$ for all nodes x
   - If $C_d(v,w)=\min_x C_d(v,x)$, then change entry in *d*-th row of routing table to (d, $C_d(v, w)$) and send message [d, $C_d(v,w)$] to all neighbors.

Count-to-Infinity Problem with Distance Vector Routing



Distances to C:
A --> C is 2
B --> C is 1

Assume that link (B,C) goes down, B does not know that C is unreachable and thinks that it can reach C through A at cost: (2+1)=3. Then A recalculates its distance vector and finds that C is at distance (3+1)=4 (through B) etc.

The **critical issue** is: the fact that A's path to C depends on the (B,C) link is not considered.

Fixing the Count-to-Infinity problem

- Apply a maximum value on the cost (i.e. 20 hops)
  - slow convergence
- Report the entire path to the destination in addition to the cost to the destination
  - very expensive solution
- Split horizon technique: If router R forwards traffic for destination D through neighbor N then R reports to N that R's distance to D is infinity. Because R is routing traffic to D through N, R's real distance to D cannot possibly matter to N.

Link State Routing

- Link State routing is an alternative to Distance Vector routing
- Link State routers do not exchange distance information as Distance Vector routers.
- They only exchange the state of the link to each neighbor
- OSPF (Open Shortest Path First) is the most popular link state routing algorithm in IP

Link State Routing Algorithm Operation

- **Construction of Link State Packet (LSP)**: Each router constructs a packet known as Link State Packet that contains a list of the names and cost to each neighbor
- **Distribution of LSPs**: The LSP is transmitted to all the routers and each router stores the **“most recently” generated LSP** from each other router
- **Calculation of routes**: Each router armed with a complete map of the topology computes routes to each destination (i.e. compute shortest paths)

Distribution of LSPs

- Flooding: each packet received is transmitted to each neighbor except the one from which it was received
  - use of max # of hops to prevent a single packet's spawning an infinite number of offspring
  - exponential number of copies for each packet
- If an LSP (from source S) is received from neighbor N and the LSP is identical to the one from S that is stored, then ignore it (it is duplicate). Otherwise store the received LSP and transmit it to all neighbors except N. The LSP travels each link in the network only once.

Most Recently Generated LSP?

- A router R can not always assume that the most recently received LSP from S is the most recently generated by S (two LSP's may travel different paths).
- Possible solutions:
  - use of timestamp
  - use of sequence numbers
  - use of AGE field: starts at some initial value and gets decremented as it is held in memory. If it reaches zero the LSP is considered too old and an LSP with a non-zero age field is accepted as newer (regardless of sequence number etc.)

Computing shortest paths (Dijkstra's Algorithm)

Shortest Path Routing (Dijkstra's Algorithm)

Let $D(v)$ be the distance (sum of link weights along a given path) from source 1 to node v. Let $\ell(i,j)$ be the (given) cost between node i and node j. There are then two parts to the algorithm: an initialization step, and a step to be repeated until the algorithm terminates:

1. *Initialization.* Set $N = \{1\}$. For each node v not in N, set $D(v) = \ell(1,v)$. (We use ∞ for nodes not connected to 1; any number larger than the maximum cost or distance in the network would suffice.)

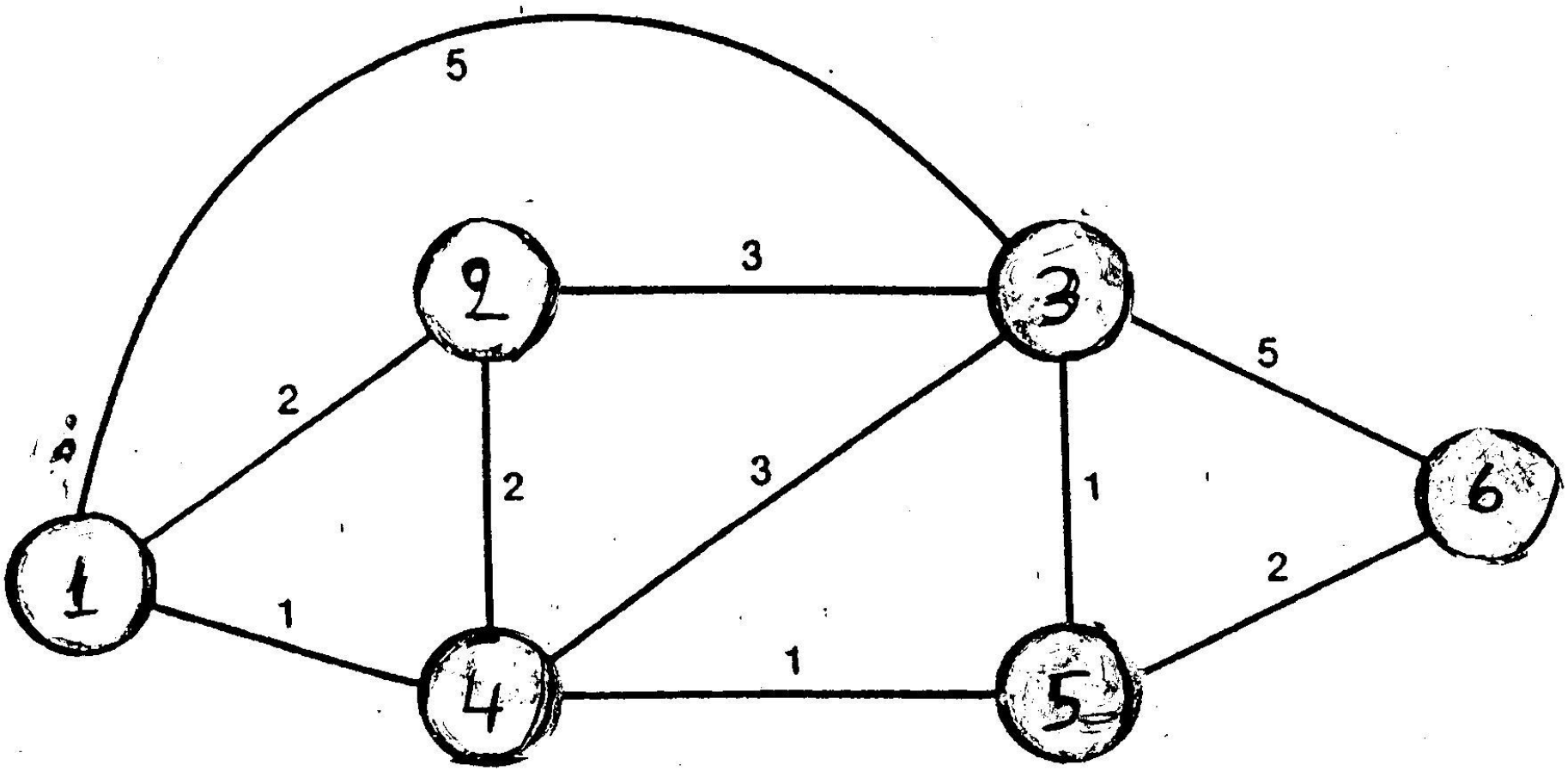
2. *At each subsequent step.* Find a node w not in N for which $D(w)$ is a minimum, and add w to N. Then update $D(v)$ for all nodes remaining that are not in N by computing

$$D(v) \leftarrow \text{Min}[D(v), D(w) + \ell(w,v)]$$

Step 2 is repeated until all nodes are in N.

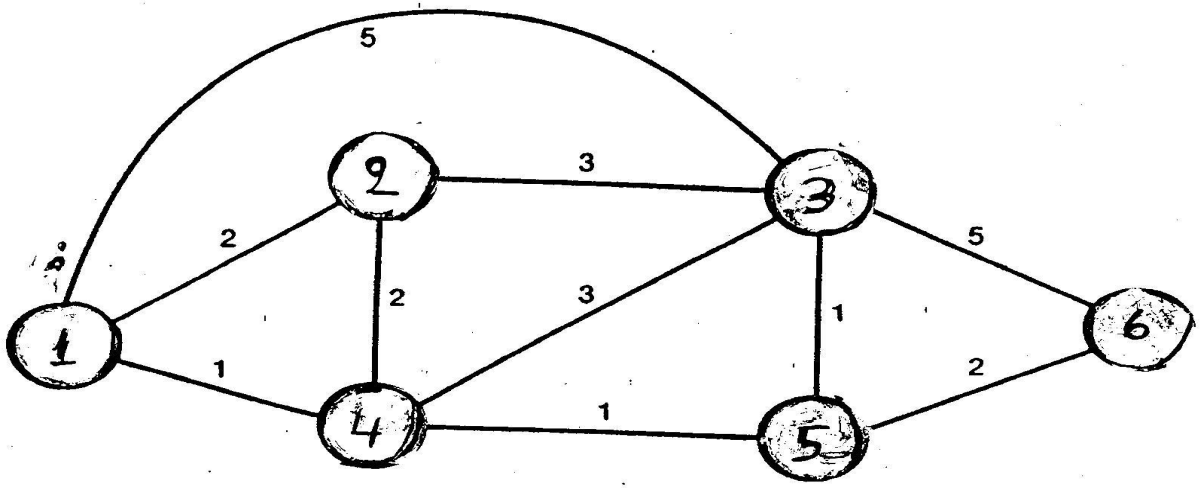
An Example:

- **For the network below, calculate the shortest paths from node 1 to all other nodes, using Dijkstra's algorithm**



Final Calculation

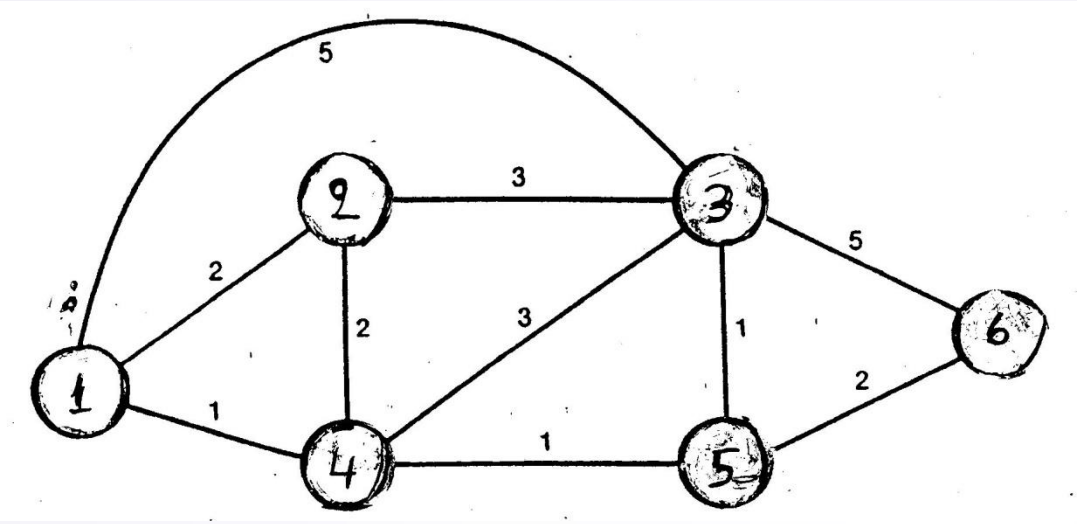
Example:
Dijkstra's Algorithm



Example network

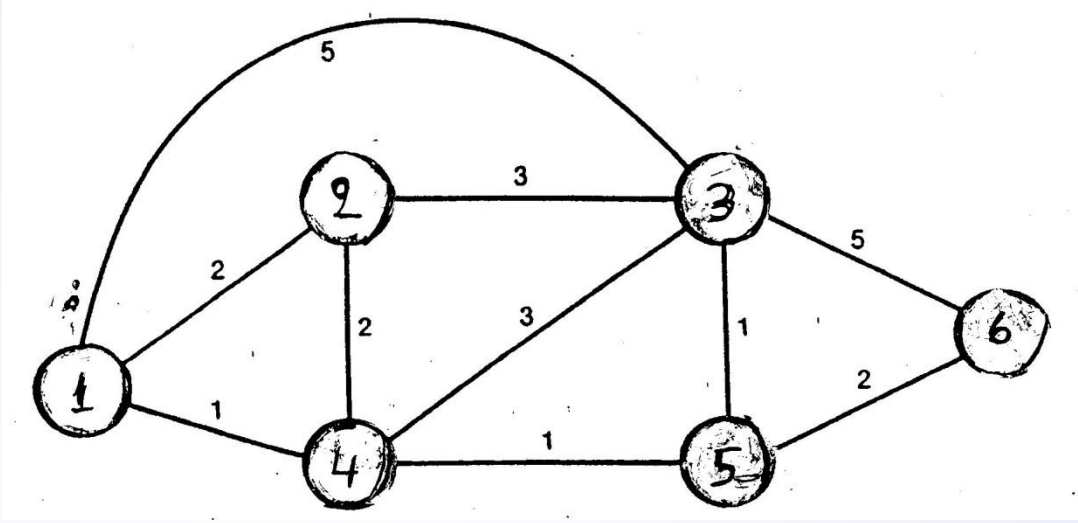
Step	N	D(2)	D(3)	D(4)	D(5)	D(6)
Initial	{1}	2	5	1	∞	∞
1	{1,4}	2	4	①	2	∞
2	{1,4,5}	2	3	1	②	4
3	{1,2,4,5}	②	3	1	2	4
4	{1,2,3,4,5}	2	③	1	2	4
5	{1,2,3,4,5,6}	2	3	1	2	④

Step by step calculation: Initialization phase



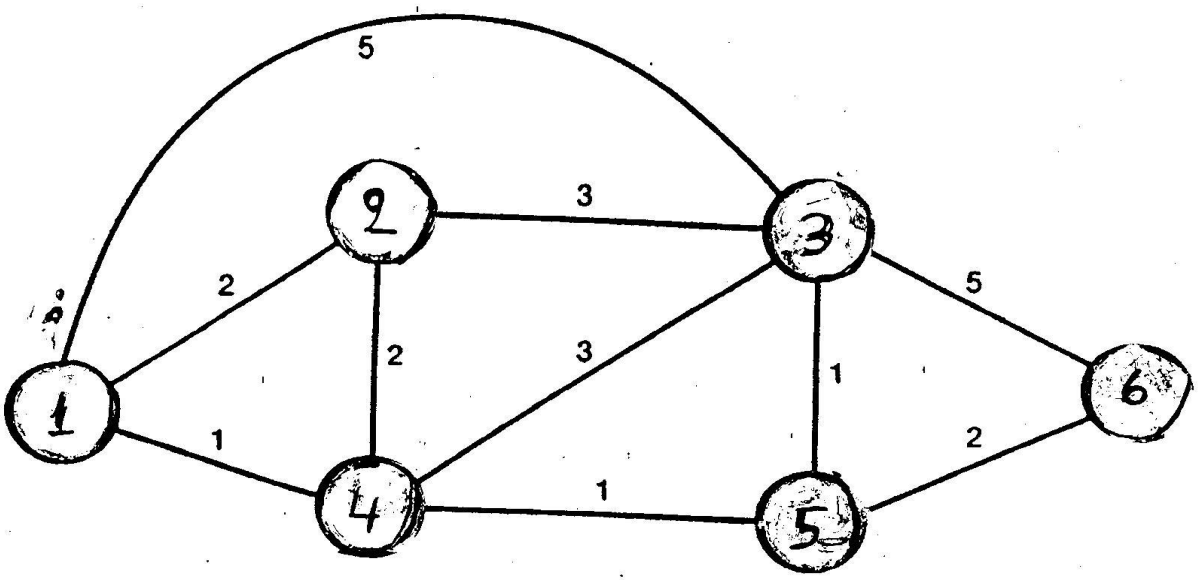
- Initialization step: N={1}
- For each node v not in N (e.g. v={2,3,4,5,6}): D(v)=l(1,v)
- D(2)=l(1,2)=2; D(3)=l(1,3)=5; D(4)=l(1,4)=1;
- D(5)=l(1,5)=inf; D(6)=l(1,6)=inf
- These values are placed in the first row of the table (see previous figure)

Step 1



- Find node w not in N, such that D(w) is minimum. In this step: w=4, since D(4)=1. (see first row of the table in previous figure). Now: N={1,4}
- Update D(v) for all nodes v not in N (e.g. v={2,3,5,6}), using:
- D(v)=Min{D(v),D(w)+l(w,v)}
- Example:
- For v=2; D(2)=Min{D(2),D(4)+l(4,2)}=Min{2,1+2}=2
- For v=3; D(3)=Min{D(3),D(4)+l(4,3)}=Min{5,1+3}=4
- For v=5; D(5)=Min{D(5),D(4)+l(4,5)}=Min{inf,1+1}=2
- For v=6; D(6)=Min{D(6),D(4)+l(4,6)}=Min{inf,1+inf}=inf
- Based on this calculation Step 1 row (second row of the final table) is calculated

Example:
Dijkstra's Algorithm



Example network

Step	N	D(2)	D(3)	D(4)	D(5)	D(6)
Initial	{1}	2	5	1	∞	∞
1	{1,4}	2	4	①	2	∞
2	{1,4,5}	2	3	1	②	4
3	{1,2,4,5}	②	3	1	2	4
4	{1,2,3,4,5}	2	③	1	2	4
5	{1,2,3,4,5,6}	2	3	1	2	④

Following steps

- Same procedure as before is repeated in order to obtain the new updated calculation (Step 2 row in the final table). In step 2 we have to find node w not in N, such that D(w) is minimum (see second row on the table). In this step we choose: w=5, since D(5)=2. Please note that D(2)=D(5)=2 in this step. In the case of tie as here, among those nodes that tie, we choose one node randomly. Here we chose w=5. Therefore: N={1,4,5}.
- In this step we update D(v) for all nodes v not in N (e.g. v={2,3,6}), using: D(v)=Min{D(v),D(w)+l(w,v)}. After completing this step, Step 2 row (third row in the final table) is completed, and so on, until you exhaust all the nodes..

Comparison of Link State and Distance Vector Routing

- Memory: Assume n-nodes in the network and that each node has k-neighbors:
  - Distance Vector: Each DV is O(n)and keeps distance vector from each of its k neighbors. Therefore O(k x n).
  - Link State: Each node keeps n LSPs (one from each node in the network). Each LSP is proportional to k. Therefore: O(n x k).
- Bandwidth Consumption: Depends heavily on the topology.

Comparison of Link State and Distance Vector Routing (cont.)

  - Distance Vector fans: a link change only propagates control messages as far as the link change's routing effect (i.e. in case of two parallel links where one fails and recovers)
  - Link Sate fans:a link change can cause the transmission of multiple control packets over a single link under distance Vector (i.e. count to infinity problem. Under LS each LSP travels only once on each link.
- Speed of convergence: LSP converges faster than the DV.

Ιεραρχική δρομολόγηση

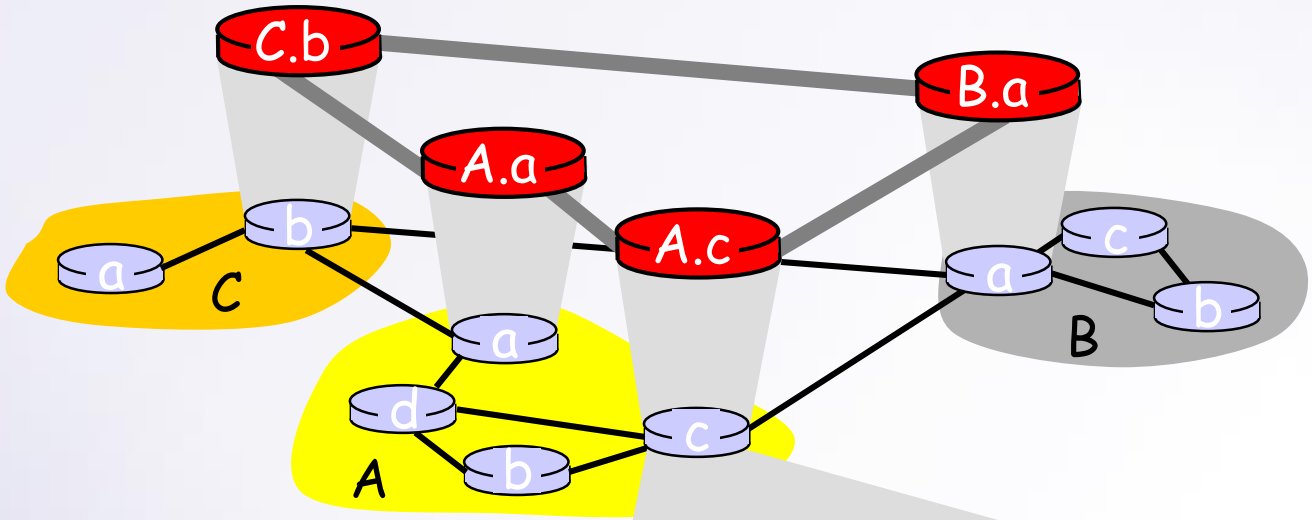
- Το Internet αποτελείται από Αυτόνομα Συστήματα (AS) διασυνδεδεμένα μεταξύ τους:
- **Stub AS**: μικρή επιχείρηση, μία σύνδεση με άλλο AS.
- **Multihomed AS**: μεγάλη επιχείρηση (όχι transit), πολλαπλές συνδέσεις με άλλα AS.
- **Transit AS**: πάροχος, διασυνδέει πολλά AS.
- Δρομολογητές στο ίδιο AS τρέχουν το ίδιο "intra-AS" πρωτόκολλο δρομολόγησης.
- Δρομολογητές σε διαφορετικά AS μπορεί να τρέχουν διαφορετικό intra-AS πρωτόκολλο δρομολόγησης.

Δρομολογητές πύλες

- Ειδικοί δρομολογητές στο AS.
- *Υπεύθυνοι* για δρομολόγηση προς προορισμούς εκτός AS.
  - Τρέχουν πρωτόκολλο δρομολόγησης *inter-AS* με τους άλλους δρομολογητές πύλες.

Ιεραρχική δρομολόγηση

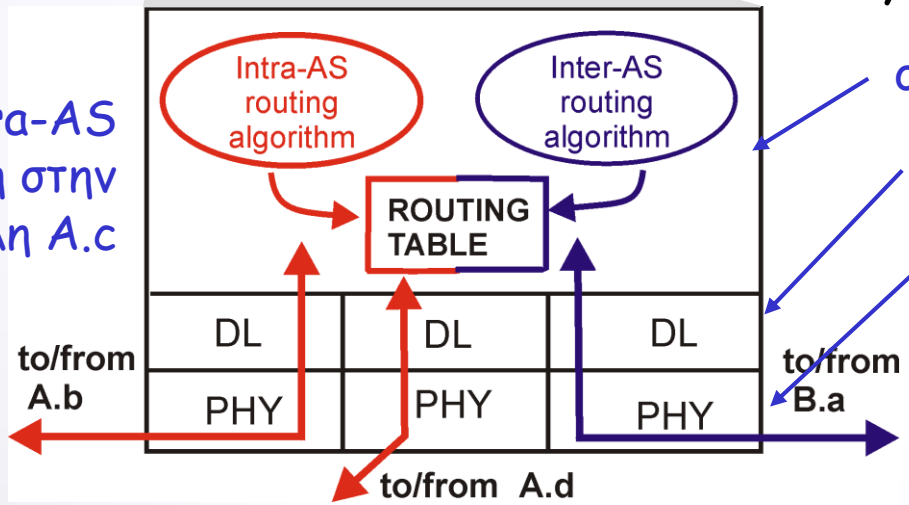
Δρομολόγηση Intra-AS και Inter-AS



Πύλες:

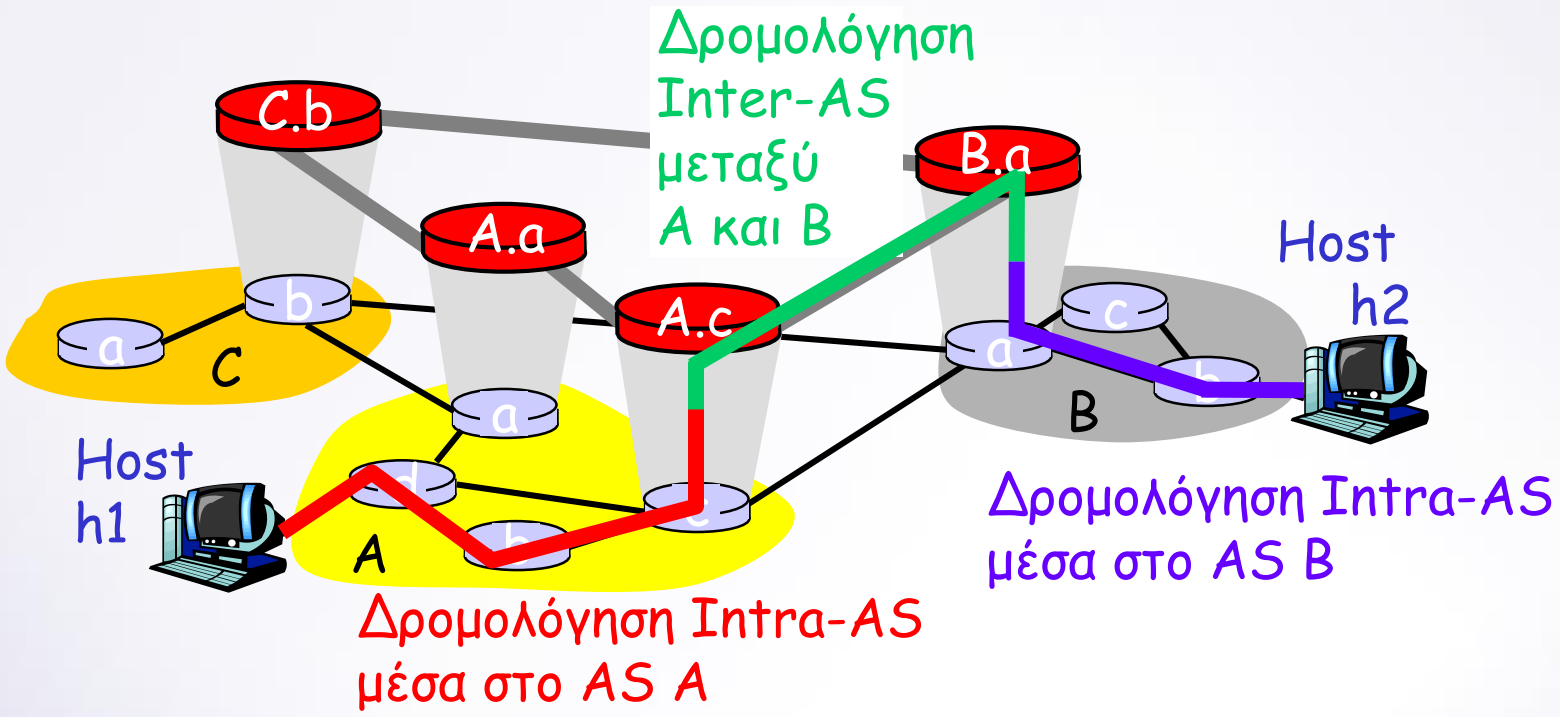
- Πραγματοποιούν δρομολόγηση inter-AS μεταξύ τους.
- Πραγματοποιούν δρομολόγηση intra-AS με τους άλλους δρομολογητές στο AS τους.

inter-AS & intra-AS δρομολόγηση στην πύλη A.c



Ιεραρχική δρομολόγηση

Δρομολόγηση Intra-AS και Inter-AS



Ιεραρχική δρομολόγηση

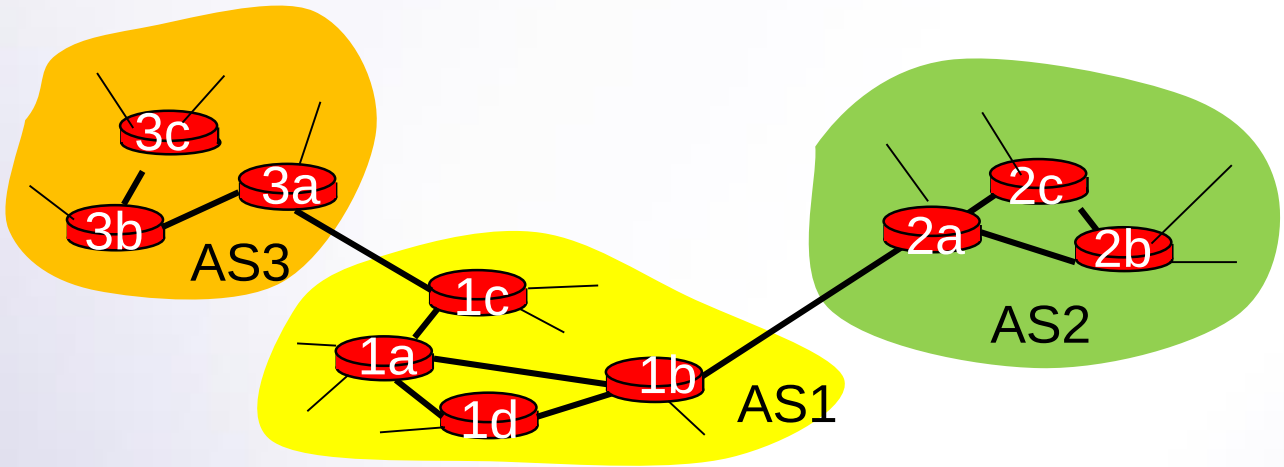
Δρομολόγηση Inter-AS

- Έστω ότι κάποιος router στο AS1 λαμβάνει πακέτο με προορισμό εκτός του AS1:
  - Ο router θα πρέπει να προωθήσει το πακέτο στον gateway router, αλλά ποιον;

Το AS1 πρέπει:

1. Να μάθει ποιοι προορισμοί είναι προσβάσιμοι μέσω του AS2 και ποιοι μέσω του AS3.
2. Να διαδώσει αυτή την πληροφορία πρόσβασης σε όλους τους routers στο AS1.

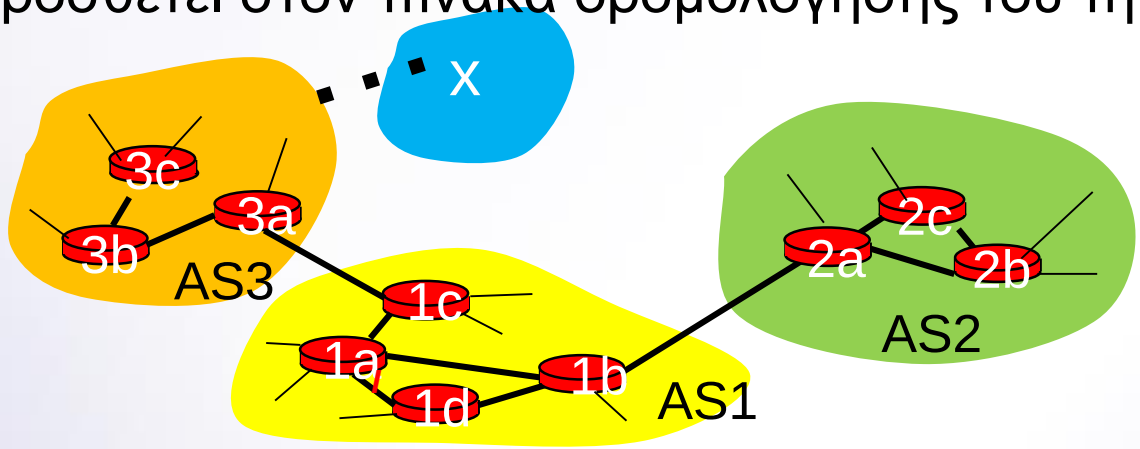
Έργο της δρομολόγησης inter-AS!



Ιεραρχική δρομολόγηση

Παράδειγμα διάρθρωσης πίνακα δρομολόγησης

- Έστω ότι το AS1 μαθαίνει (μέσω του πρωτοκόλλου inter-AS) ότι υποδίκτυο *X* είναι προσβάσιμο μέσω του AS3 (gateway 1c) αλλά όχι μέσω του AS2 (gateway 1b).
- Το πρωτόκολλο inter-AS διαδίδει την πληροφορία πρόσβασης σε όλους τους εσωτερικούς δρομολογητές.
- Ο δρομολογητής 1d γνωρίζει από τη δρομολόγηση intra-AS ότι η *I* διεπαφή του είναι στην διαδρομή ελαχίστου κόστους προς τον 1c.
  - προσθέτει στον πίνακα δρομολόγησής του την εγγραφή *(X,I)*

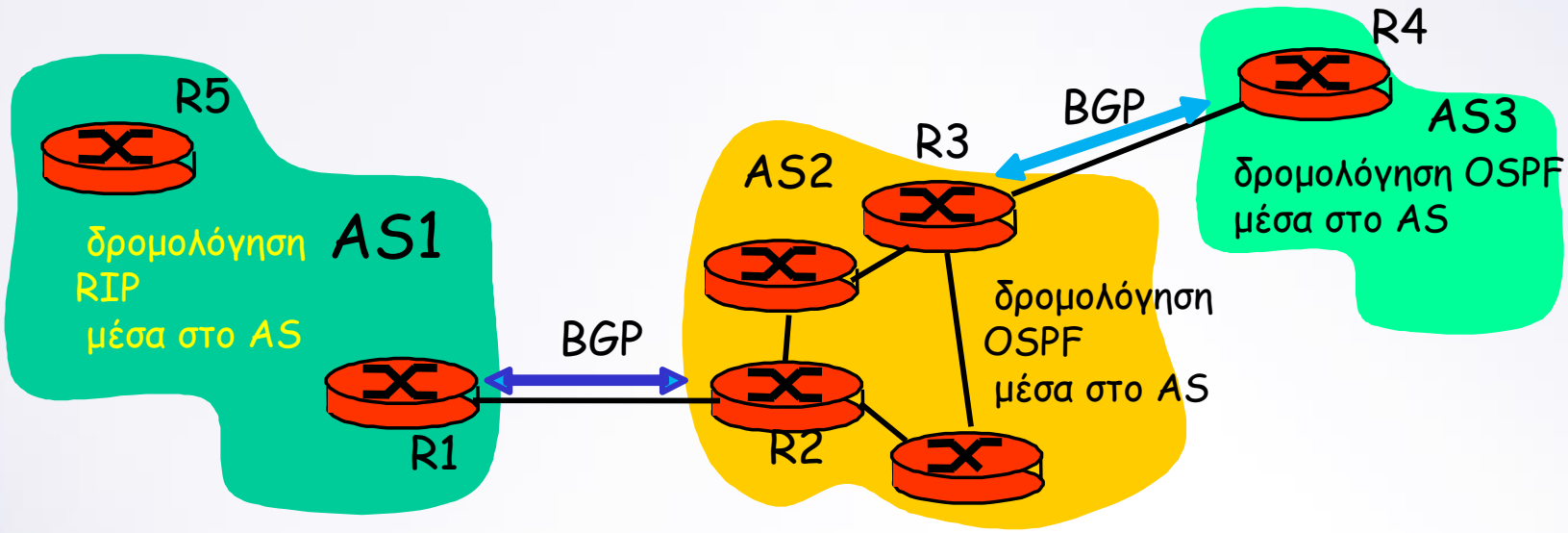


Πρωτόκολλα Δρομολόγησης Intra-AS

- Γνωστά και ως Interior Gateway Protocols (IGP).
- Τα γνωστότερα πρωτόκολλα δρομολόγησης Intra-AS:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (ιδιωτικό της Cisco)

Δρομολόγηση μεταξύ AS

BGP (Border Gateway Protocol)



BGP

- Είναι το πρότυπο που ισχύει για δρομολόγηση μεταξύ AS.
- Παρέχει σε κάθε AS τα μέσα για:
  - Να αποκτήσει πληροφορίες προσβασιμότητας από τα γειτονικά AS.
  - Να διαδίδει τη πληροφορία προσβασιμότητας σε όλους τους εσωτερικούς δρομολογητές του.
  - Να καθορίζει “καλές” διαδρομές προς τα άλλα AS βάσει της πληροφορίας προσβασιμότητας και της πολιτικής δρομολόγησης.
- Επιτρέπει σε ένα AS να αναγγείλει την ύπαρξή του στο υπόλοιπο Internet.

Γιατί δρομολόγηση Intra- και Inter-AS;

Πολιτική:

- Inter-AS: ο διαχειριστής θέλει να ελέγχει πώς δρομολογείται η κίνηση του δικτύου του και ποιος δρομολογεί μέσω του δικτύου του.
- Intra-AS: μία επικράτεια διαχείρισης, οπότε δεν χρειάζεται πολιτική δρομολόγησης.

Κλίμακα:

- Η ιεραρχική δρομολόγηση εξοικονομεί μέγεθος πινάκων, περιορίζει το φορτίο ενημέρωσης.

Επίδοση:

- Intra-AS: μπορεί να εστιάσει στην επίδοση.
- Inter-AS: η πολιτική μπορεί να επισκιάζει την επίδοση.