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)

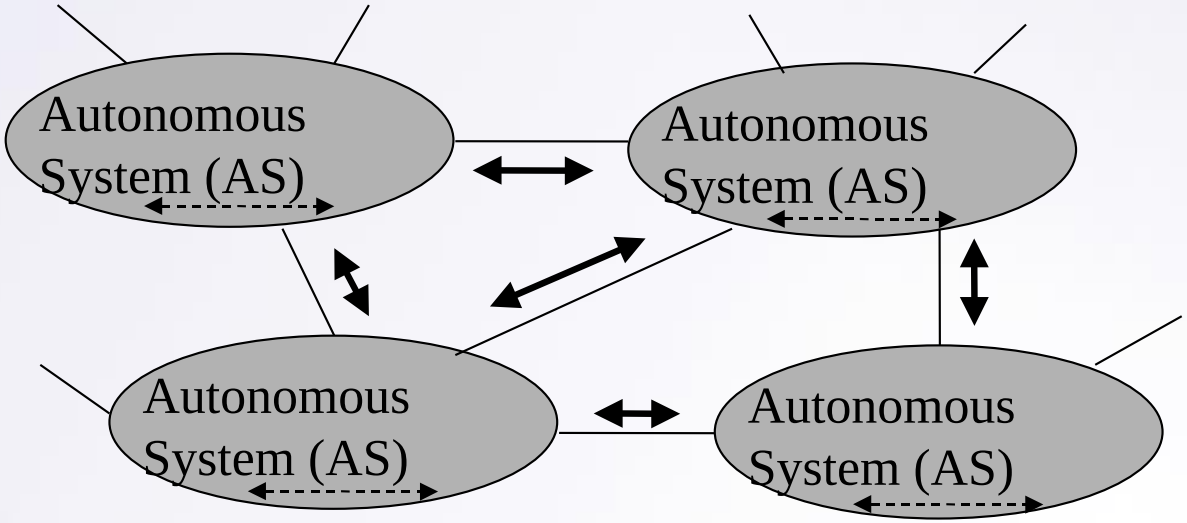
Βιβλίο Αναφοράς: TCP/IP Illustrated, Vol. I, by W.R. Stevens (Addison Wesley)

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)

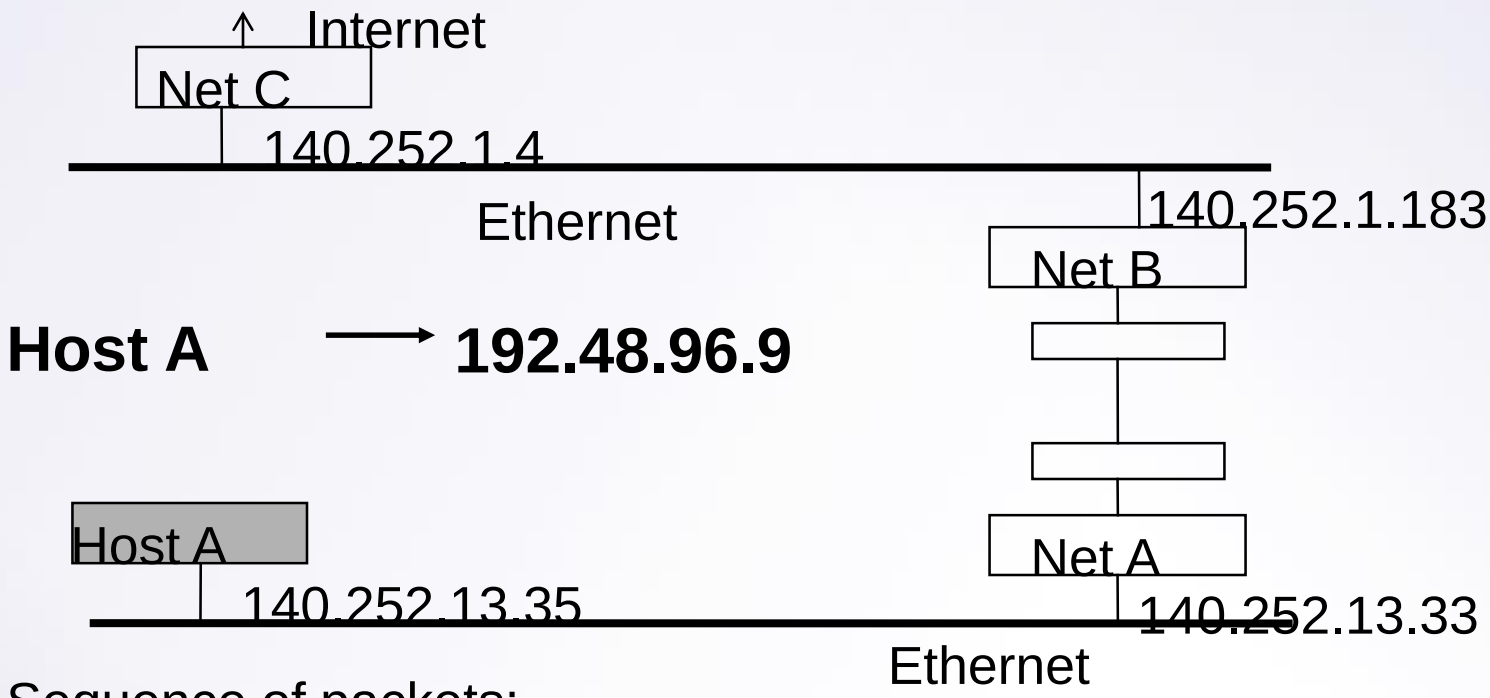
Routing Hierarchy



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

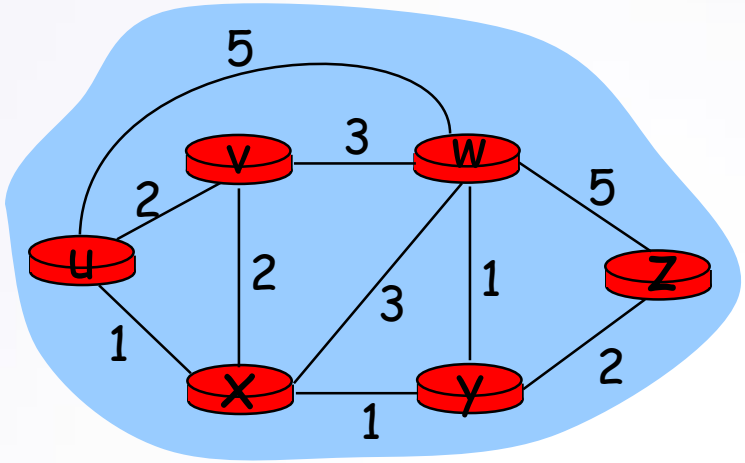
- Exterior Gateway Protocols (EGPs)

- 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.



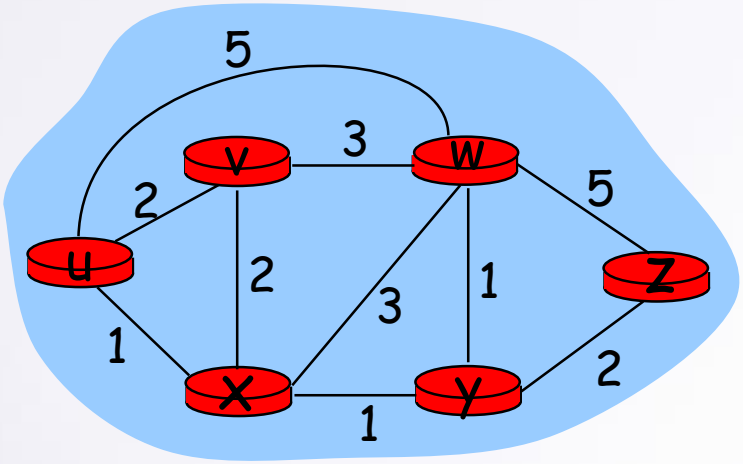
Γράφος: G = (N,E)

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

Στατική ή δυναμική δρομολόγηση;

Στατική:

- Εκ των προτέρων καθορισμένες διαδρομές.
- Οι διαδρομές αλλάζουν αργά με το χρόνο.

Δυναμική:

- Οι διαδρομές αλλάζουν πιο συχνά.
- Επανακαθορισμός και αναδιάταξη των διαδρομών:
  - Περιοδική ενημέρωση.
  - Ενημέρωση, όταν αλλάζουν τα κόστη των ζεύξεων.

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

Αλγόριθμος διανύσματος αποστάσεων

- Διάνυσμα αποστάσεων: Distance vector
- Κάθε κόμβος ξέρει την απόσταση (κόστος) προς κάθε άμεσα συνδεόμενο με αυτόν γείτονα.
- Κάθε κόμβος στέλνει περιοδικά το δικό του διάνυσμα αποστάσεων στους γείτονές του.
- Όταν ένας κόμβος X λάβει νέο διάνυσμα αποστάσεων από γείτονα *V*, ενημερώνει το δικό του διάνυσμα σύμφωνα με τη σχέση:

$$d_X(Y) = \min_V \{c(X,V) + d_V(Y)\}, \quad \forall Y \in N$$

- Μετά από μερικές αλλαγές οι αποστάσεις συγκλίνουν στις ελαχίστου κόστους.

Αλγόριθμος διανύσματος αποστάσεων

Επαναληπτικός, ασύγχρονος:

- Κάθε τοπικός επανυπολογισμός προκαλείται από:
  - Τοπική αλλαγή κόστους ζεύξης.
  - Νέο διάνυσμα αποστάσεων από τον γείτονα: αλλαγή διαδρομής ελαχίστου κόστους από τον γείτονα.
- *Τερματίζει μόνος του*, δεν απαιτείται εντολή για να σταματήσει.

Κατανεμημένος:

- Κάθε κόμβος ειδοποιεί τους γείτονες, *μόνον* όταν η ελαχίστου κόστους διαδρομή προς *οποιαδήποτε* κατεύθυνση αλλάζει.
  - Οι γείτονες τότε ειδοποιούν τους γείτονές τους, αν είναι αναγκαίο.

Κάθε κόμβος:

Περιμένει για (αλλαγή κόστους τοπικής ζεύξης ή μήνυμα από γείτονα).

↓

Ξαναϋπολογίζει τον πίνακα αποστάσεων.

↓

Αν η διαδρομή ελαχίστου κόστους προς οιονδήποτε προορισμό έχει αλλάξει, *ειδοποιεί* τους γείτονες.

Αλγόριθμος διανύσματος αποστάσεων

Σε κάθε κόμβο διατηρείται πίνακας αποστάσεων στον οποίο υπάρχει:

- Μία σειρά για κάθε δυνατό προορισμό
- Μία στήλη για κάθε άμεσα συνδεόμενο γειτονικό κόμβο
- Παράδειγμα: στον κόμβο X, η καταχώρηση για τον προορισμό Y, υπολογίζεται ως εξής:

$$d_X(Y) = min\{c(X,V) + d_V(Y)\}$$

- όπου το min λαμβάνεται για όλους τους γείτονες V του X.

Ο γειτονικός κόμβος V για τον οποίο ισχύει το min είναι ο next hop στον πίνακα προώθησης του X για τον προορισμό Y.

Αλγόριθμος διανύσματος αποστάσεων

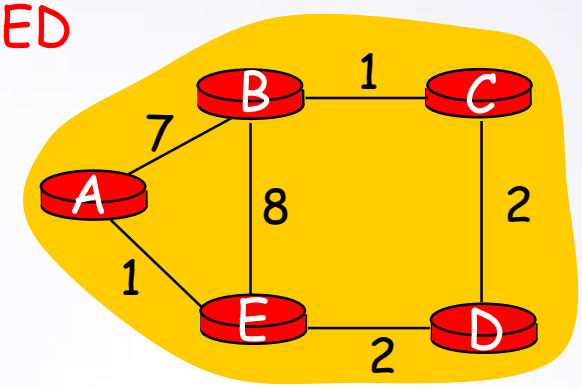
Πίνακας αποστάσεων: παράδειγμα

E	απόσταση μέσω A	B	D
προορισμός A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

(D,E,A)+ED

(B,C,D,E,A)+EB

(A,E,D,C,B)+EA



Κάθε κόμβος έχει τη δική του:

- σειρά, για κάθε δυνατό προορισμό,
- στήλη, για κάθε άμεσα συνδεδεμένο γειτονικό κόμβο.

Αλγόριθμος διανύσματος αποστάσεων

Από τον πίνακα αποστάσεων προκύπτει ο πίνακας δρομολόγησης.

προορισμός \ Ε	απόσταση μέσω A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

Πίνακας αποστάσεων ⟶

προορισμός	Ε — Εξερχόμενη ζεύξη προς χρήση
A	A,1
B	D,5
C	D,4
D	D,2

Πίνακας δρομολόγησης

Αλγόριθμος διανύσματος αποστάσεων

	A	
A		
B	1	B
C	1	C
D	2	C
E	3	C
F	4	C

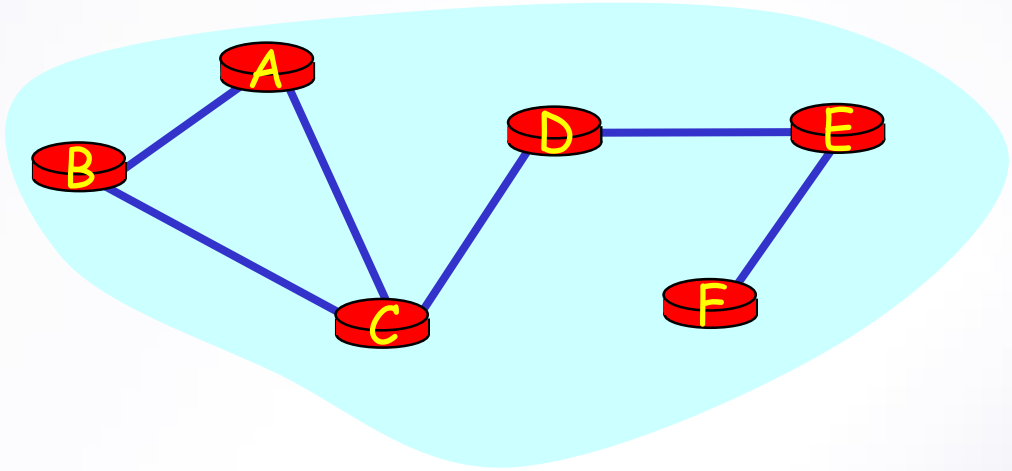
	B	
A	1	A
B		
C	1	C
D	2	C
E	3	C
F	4	C

	C	
A	1	A
B	1	B
C		
D	1	D
E	2	D
F	3	D

	D	
A	2	C
B	2	C
C	1	C
D		
E	1	E
F	2	E

	E	
A	3	D
B	3	D
C	2	D
D	1	D
E		
F	1	F

	F	
A	4	E
B	4	E
C	3	E
D	2	E
E	1	E
F		



Αλγόριθμος διανύσματος αποστάσεων

Ενημέρωση πινάκων

- Οι κόμβοι στέλνουν ενημερώσεις των διανυσμάτων αποστάσεων στους γείτονες.
- Κάθε κόμβος ενημερώνει τον πίνακά του υπολογίζοντας τη συντομότερη διαδρομή.

	A	
A		
B	1	B
C	1	C
D	2	C
E	3	C
F	4	C

	B	
A	1	A
B		
C	1	C
D	2	C
E	3	C
F	4	C

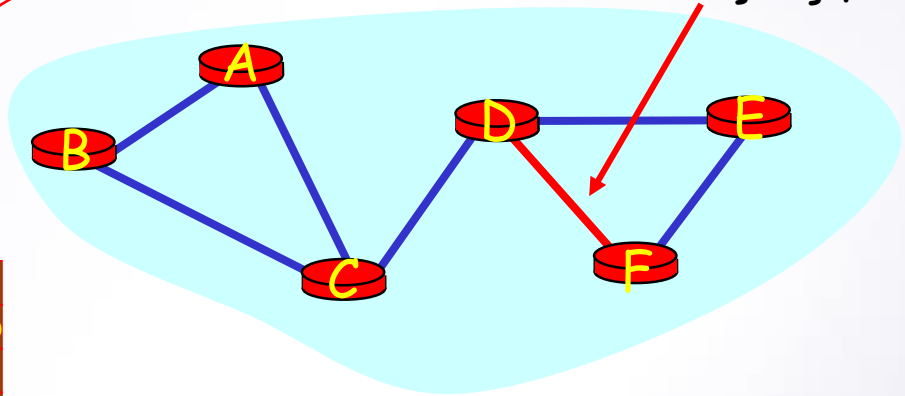
	C	
A	1	A
B	1	B
C		
D	1	D
E	2	D
F	3	D

	D	
A	2	C
B	2	C
C	1	C
D		
E	1	E
F	2	E

	E	
A	3	D
B	3	D
C	2	D
D	1	D
E		
F	1	F

	F	
A	4	E
B	4	E
C	3	E
D	2	E
E	1	E
F		

Νέα ζεύξη



Ενημέρωση πίνακα δρομολόγησης

	F	
A	4	E
B	4	E
C	3	E
D	2	E
E	1	E
F		

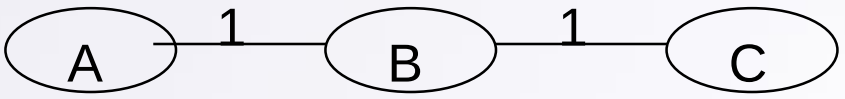
+

	D	
A	2	
B	2	
C	1	
D		
F	1	
F	1	

→

	F	
A	3	D
B	3	D
C	2	D
D	1	D
E	1	E
F		

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.)

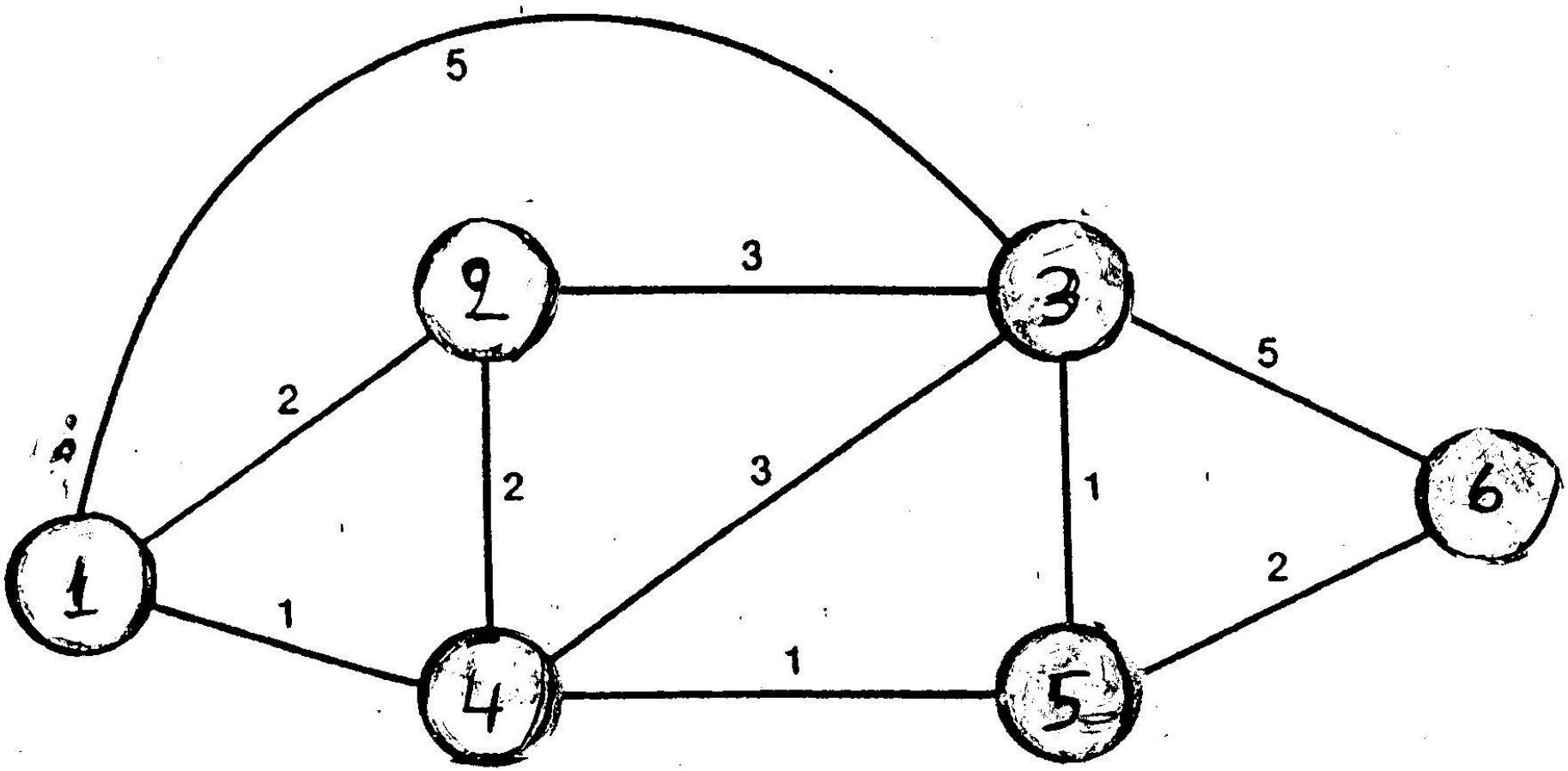
2. *At each subsequent step.* Find a node w not in N for which $D(w)$ is a minimum, and add w to N. Then update $D(v)$ for all nodes remaining that are not in N by computing

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

Step 2 is repeated until all nodes are in N.

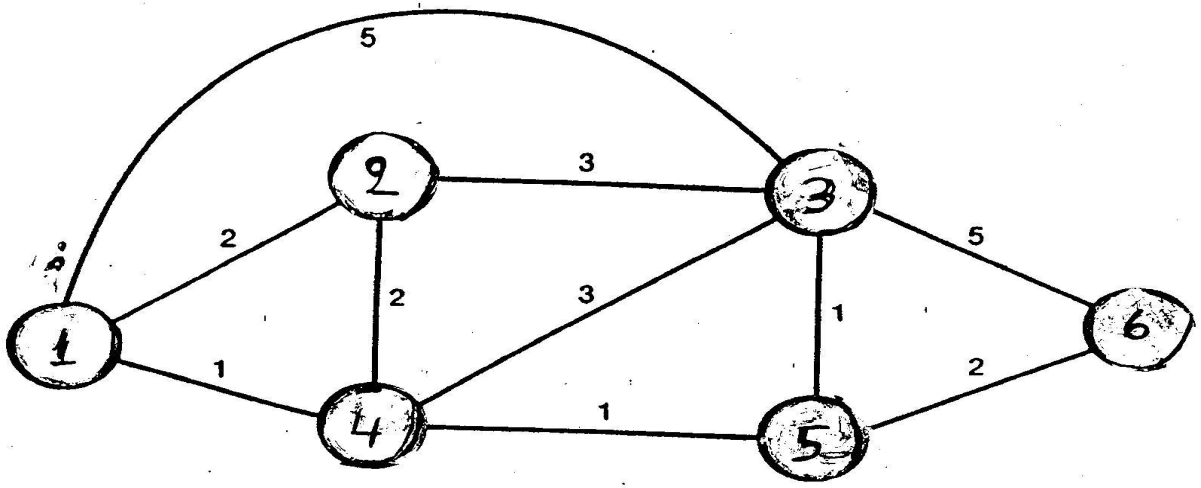
An Example:

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



Final Calculation

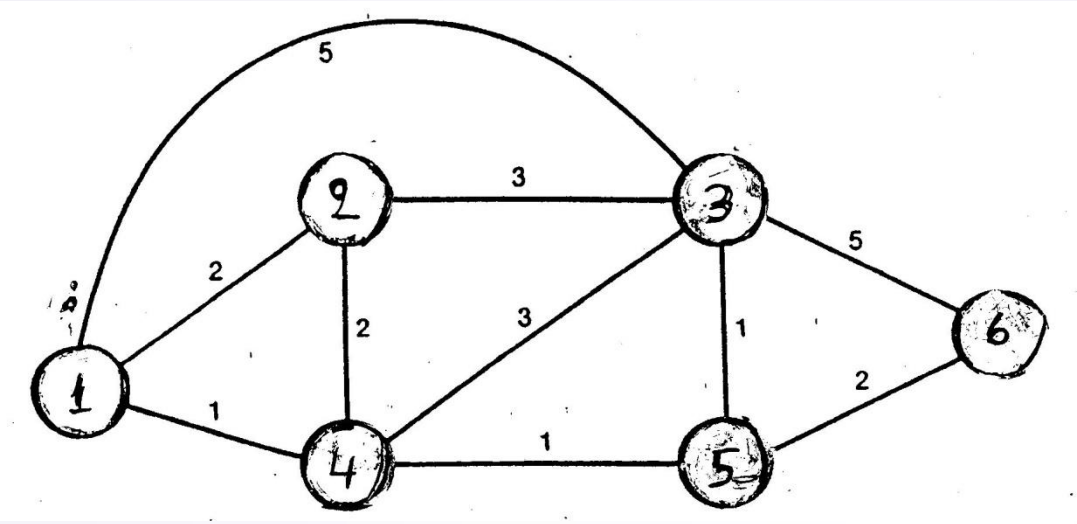
Example:
Dijkstra's Algorithm



Example network

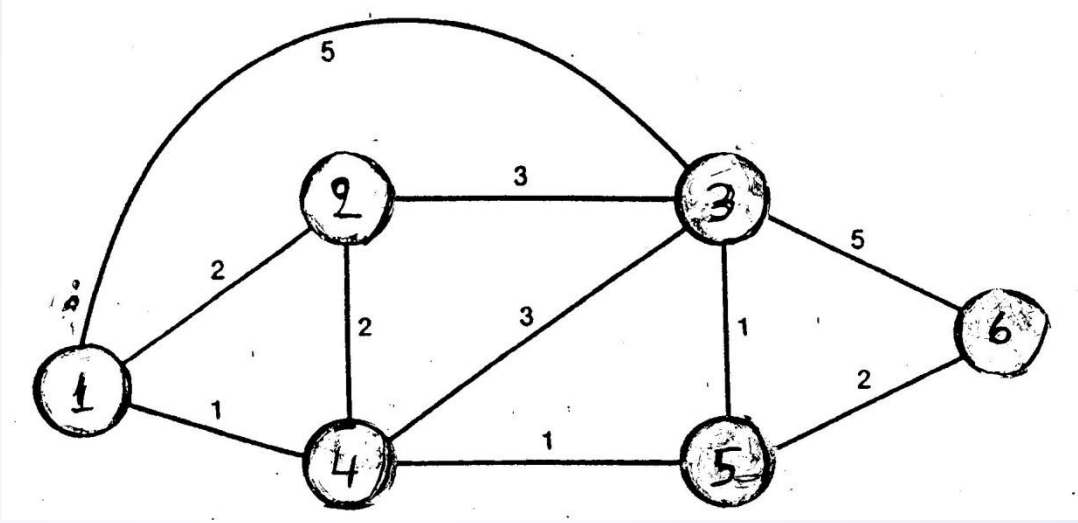
Step	N	D(2)	D(3)	D(4)	D(5)	D(6)
Initial	{1}	2	5	1	∞	∞
1	{1,4}	2	4	①	2	∞
2	{1,4,5}	2	3	1	②	4
3	{1,2,4,5}	②	3	1	2	4
4	{1,2,3,4,5}	2	③	1	2	4
5	{1,2,3,4,5,6}	2	3	1	2	④

Step by step calculation: Initialization phase



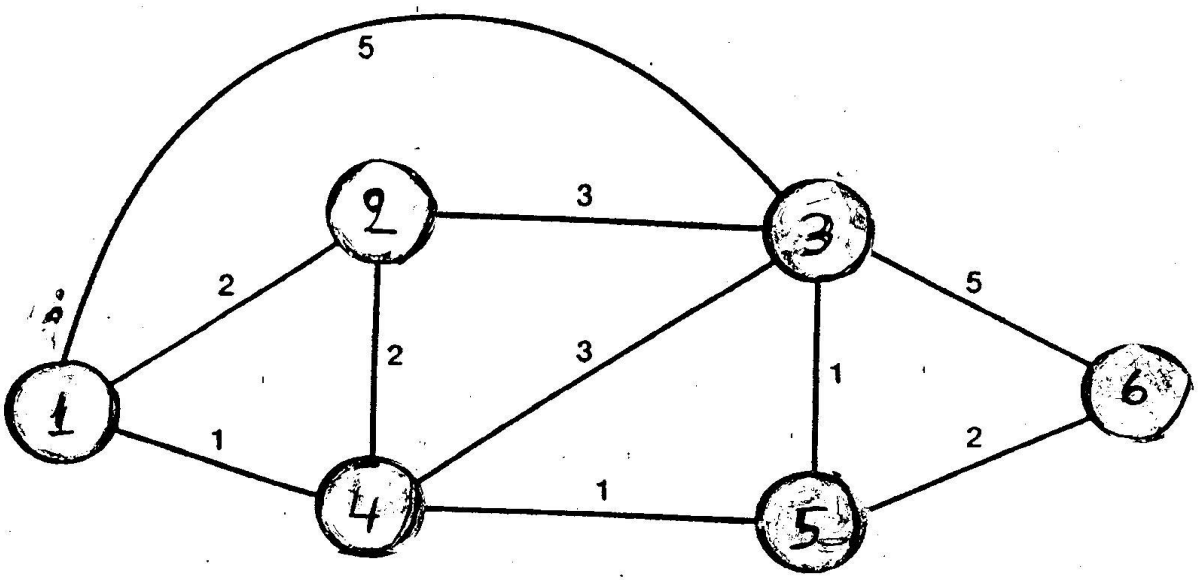
- Initialization step: N={1}
- For each node v not in N (e.g. v={2,3,4,5,6}): D(v)=l(1,v)
- D(2)=l(1,2)=2; D(3)=l(1,3)=5; D(4)=l(1,4)=1;
- D(5)=l(1,5)=inf; D(6)=l(1,6)=inf
- These values are placed in the first row of the table (see previous figure)

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

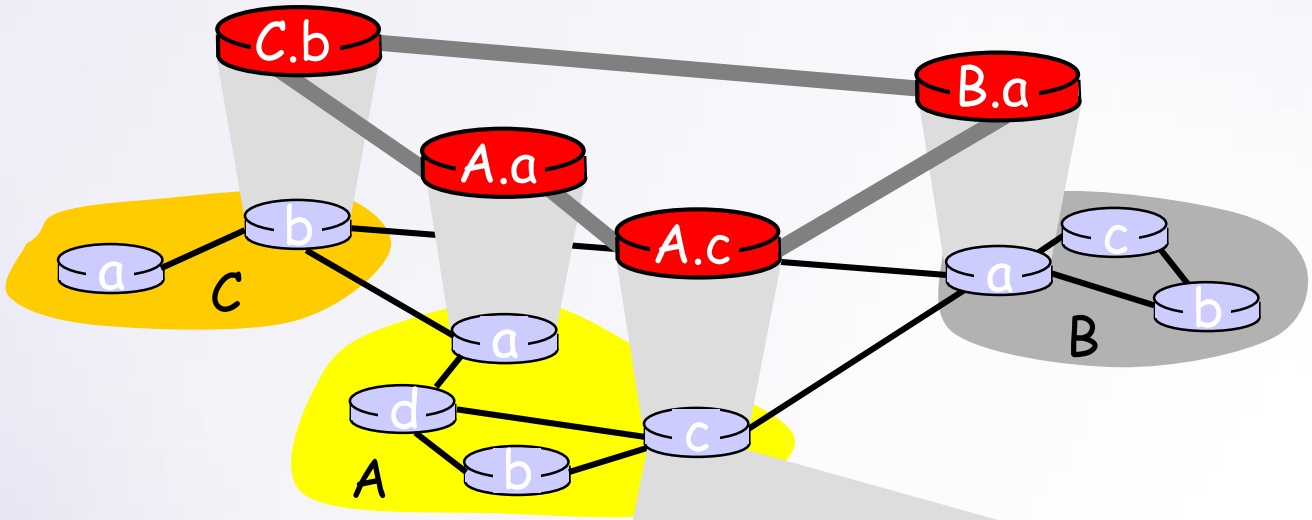
- Το Internet αποτελείται από Αυτόνομα Συστήματα (AS) διασυνδεδεμένα μεταξύ τους:
- **Stub AS**: μικρή επιχείρηση, μία σύνδεση με άλλο AS.
- **Multihomed AS**: μεγάλη επιχείρηση (όχι transit), πολλαπλές συνδέσεις με άλλα AS.
- **Transit AS**: πάροχος, διασυνδέει πολλά AS.
- Δρομολογητές στο ίδιο AS τρέχουν 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

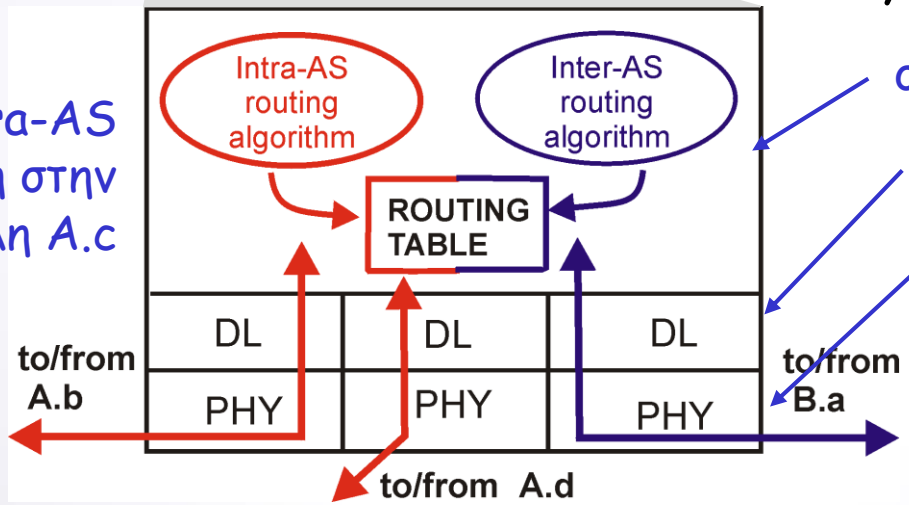
Δρομολόγηση Intra-AS και Inter-AS



Πύλες:

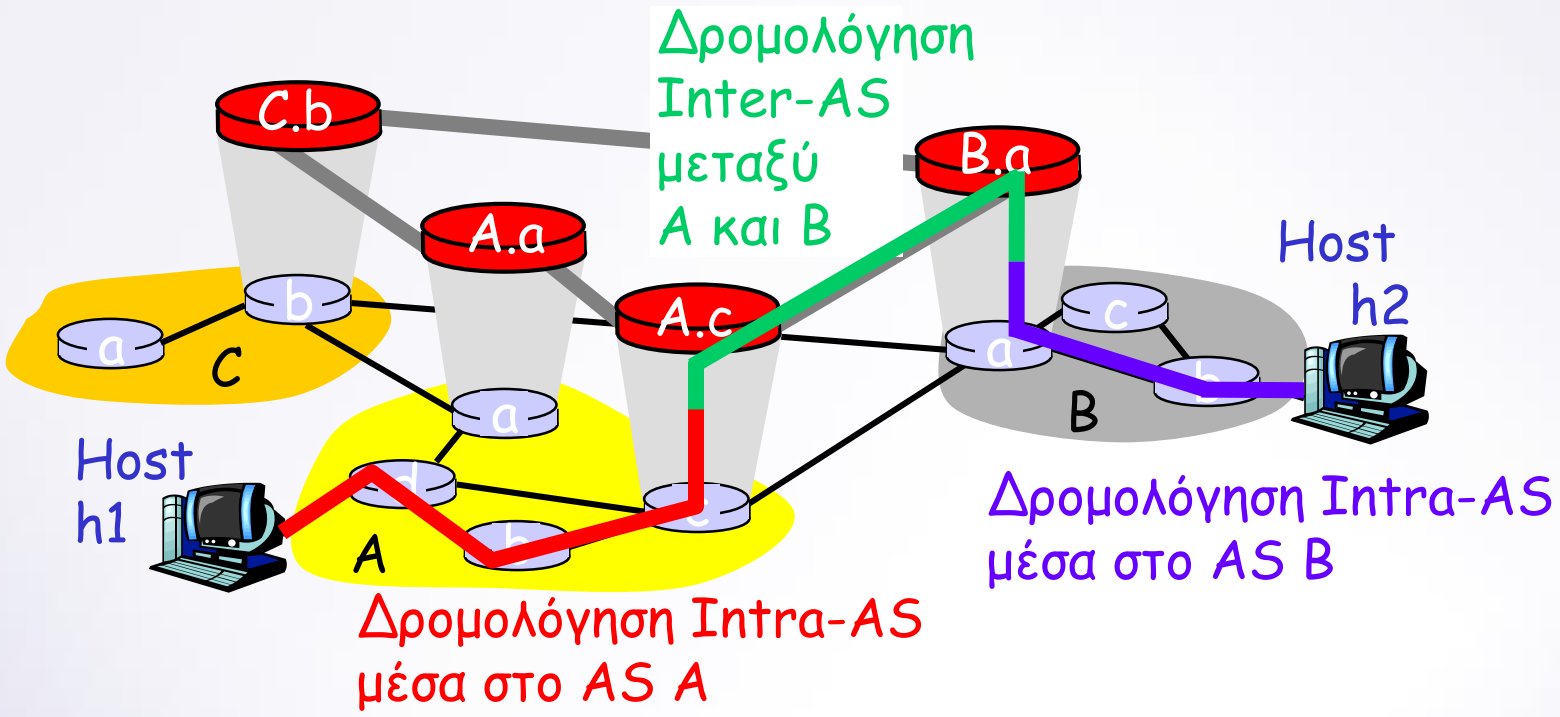
- Πραγματοποιούν δρομολόγηση inter-AS μεταξύ τους.
- Πραγματοποιούν δρομολόγηση intra-AS με τους άλλους δρομολογητές στο AS τους.

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



Ιεραρχική δρομολόγηση

Δρομολόγηση Intra-AS και Inter-AS



Ιεραρχική δρομολόγηση

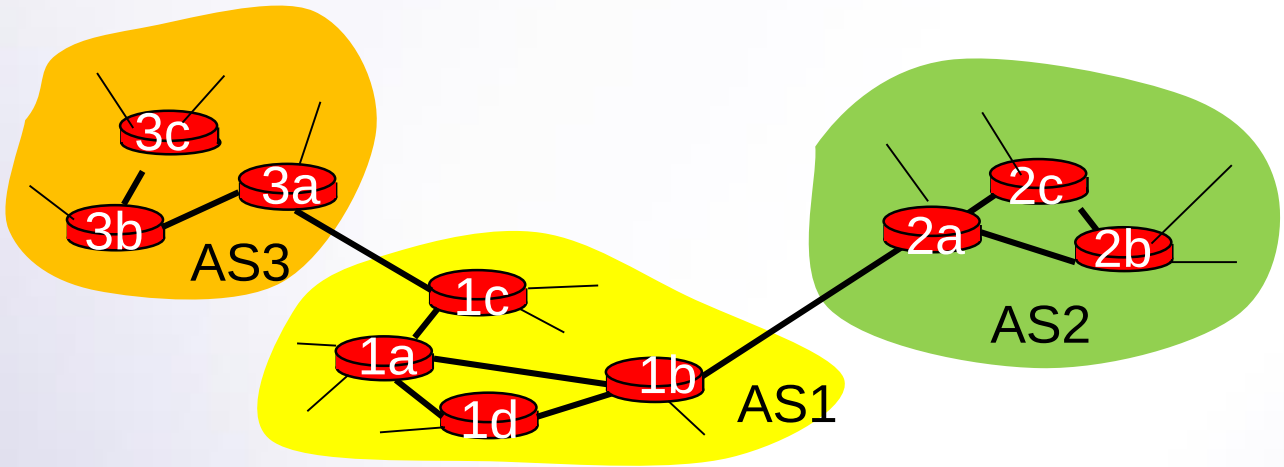
Δρομολόγηση Inter-AS

- 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.

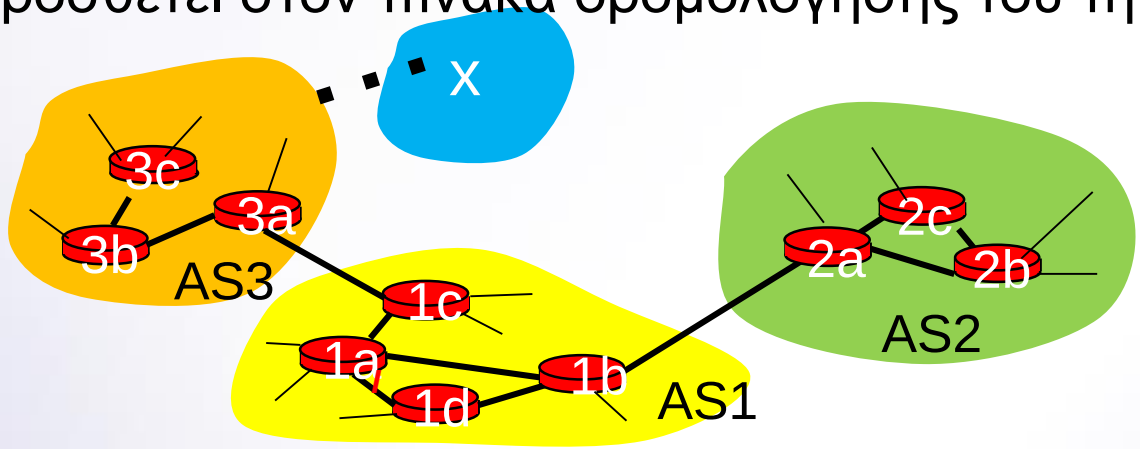
Έργο της δρομολόγησης inter-AS!



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)*

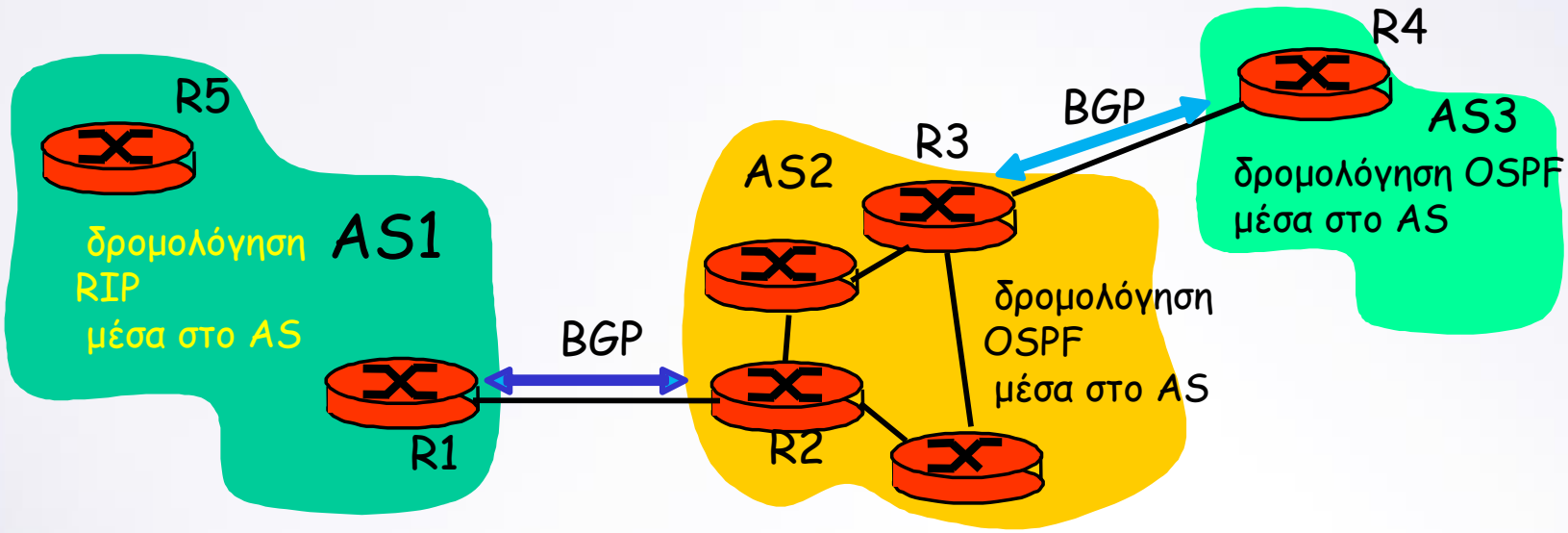


Πρωτόκολλα Δρομολόγησης Intra-AS

- 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.