

# Nivelul Retea (II)

**Lenuta Alboaie (adria@info.uaic.ro)**  
**Andrei Panu (andrei.panu@info.uaic.ro)**

# Cuprins

- **Nivelul retea**
  - **Activitatea de rutare (dirijare)**
    - Preliminarii
    - Caracterizare
    - Rutare
    - Protocoale de rutare
      - RIP & OSPF
      - BGP & EGP
  - **Congestie – discutii generale**

# Rutare | Preliminarii

- Partea software-ului nivelului retea care alege calea pe care un pachet receptionat trebuie trimis pentru a ajunge la destinatie
- Daca se folosesc datagrame, decizia de rutare trebuie luata pentru fiecare pachet
- Daca se utilizeaza circuite virtuale, decizia de rutare se ia la stabilirea unui nou circuit
- Cerintele pentru un algoritm de rutare: corect, simplu, robust, optim, rapid convergent
- Activitati
  - Determinarea caii optime de rutare (*routing*)
  - Transportarea pachetelor: comutare (*packet switching*)

# Terminologie

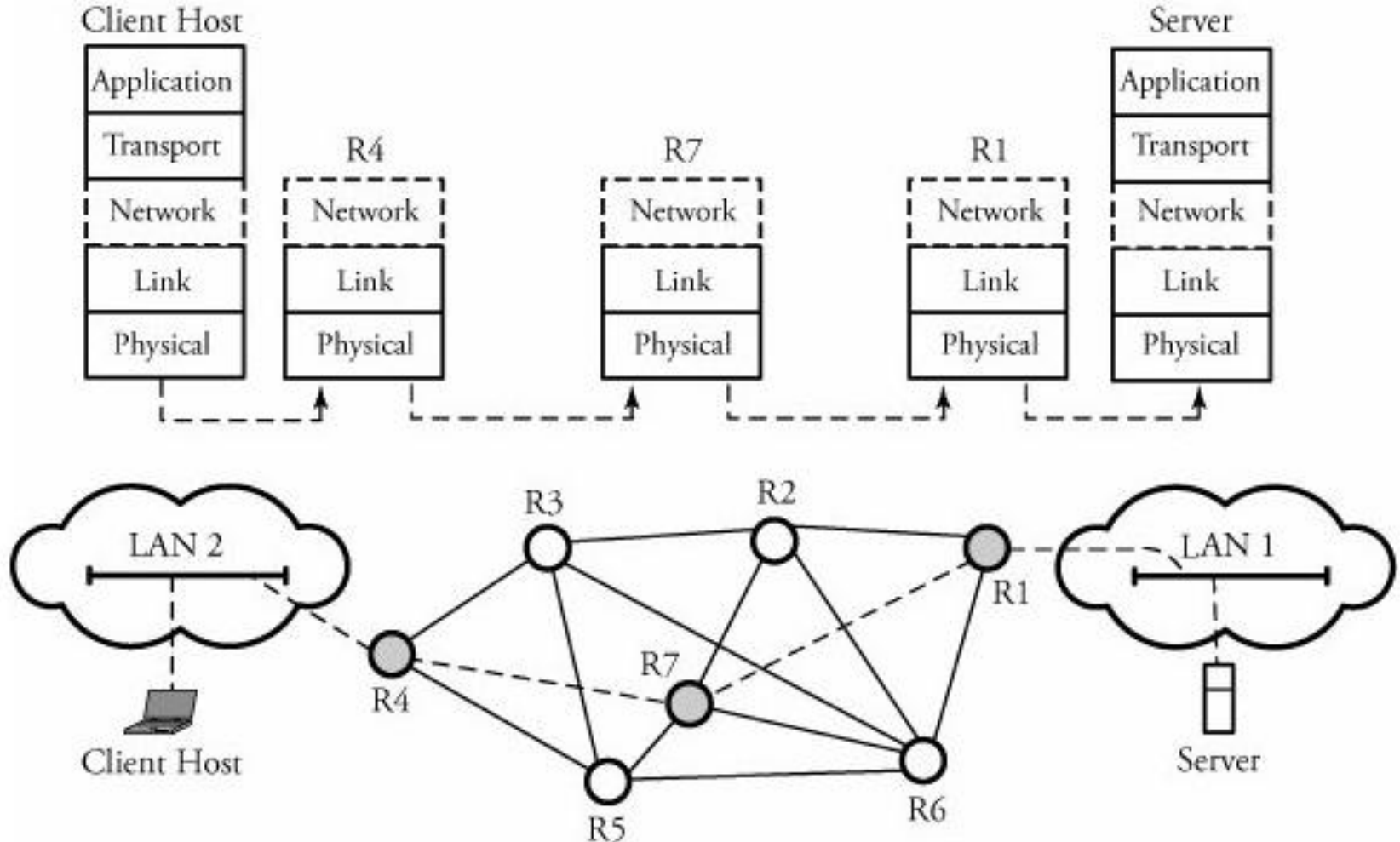
- **end systems** – dispozitive de retea fara capacitati de redirectat pachete catre subretele
- **intermediate systems** – dispozitive de retea avand capacitati de redirectat pachete
  - Intradomain IS – comunicare in cadrul unui domeniu de rutare
  - Interdomain IS – comunicare si intre domenii de rutare
- **sistem autonom** – AS (eng. *Autonomous system*) – colectie de retele care partajeaza aceeasi strategie de dirijare

# Nivelul retea | ...sa ne reamintim

- La nivelul retea, Internetul poate fi vazut ca o colectie de subretele sau sisteme autonome conectate intre ele  
IP-ul este liantul care face posibila aceasta interconectare. (a se vedea Cursul 2)
- Nivelul retea se ocupa cu trimiterea pachetelor de la sursa la destinatie (mecanism care implica trecerea printr-o serie de noduri intermediare) => nivelul retea este nivelul cel mai de jos care se ocupa cu transmisia *end-to-end*

Obs.: Nivelul legaturii de date are rolul de transport a *frame*-urilor de la un punct la altul

# Nivelul retea | ...sa ne reamintim

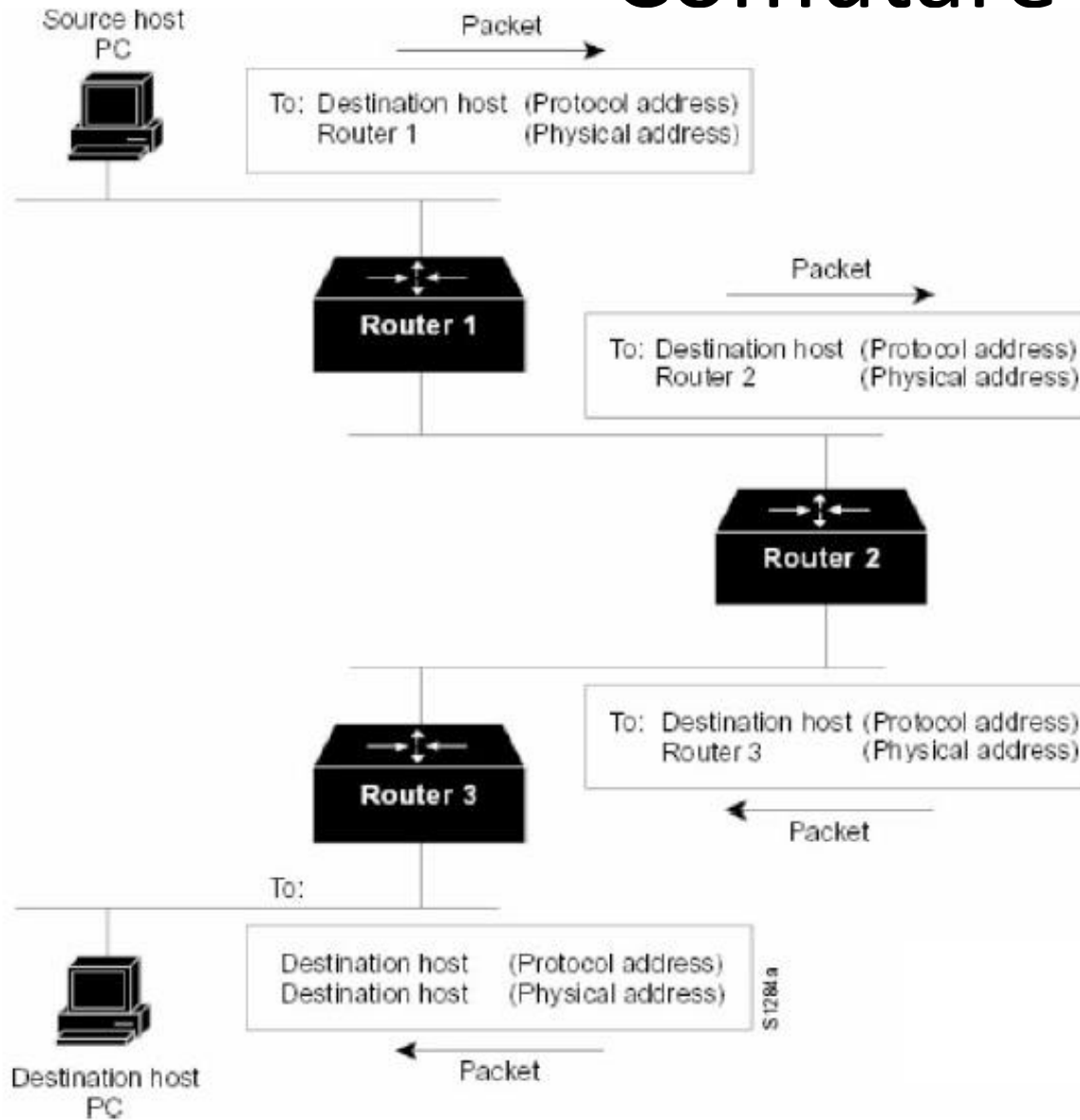


Comunicare *end-to-end* între o gazda client și un server la nivelul retea

# Comutare

- O gazda (eng. *host*) are de trimis un pachet la un alt *host*
- *Host*-ul sursa trimite pachetul la un router, folosind adresa hardware (MAC) a acestuia, un pachet continand adresa de retea a gazdei destinatie
- Routerul examineaza adresa de retea a destinatarului, iar daca nu cunoaste unde sa trimita pachetul, il va distruge
- Altfel, va modifica adresa continuta de pachet in adresa hardware a urmatorului *hop* (punct indermediar de transmitere – *Intermediate System*) si va trimite pachetul spre acesta
- Daca urmatorul *hop* nu este destinatia finala, atunci procesul se repeta pentru un alt router, s.a.m.d.

# Comutare

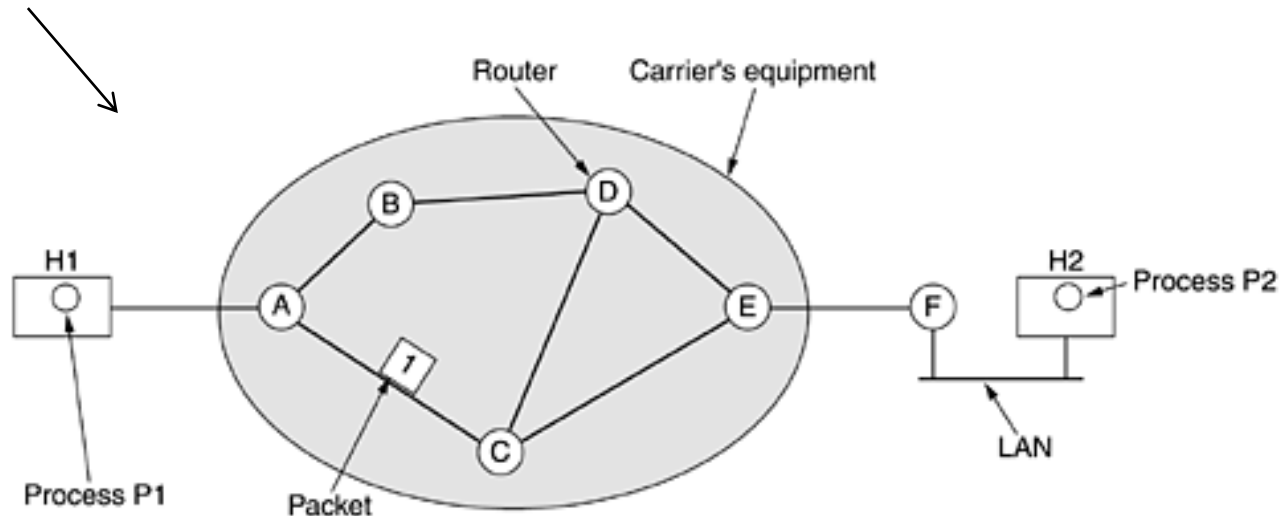


Procesul de  
comutare



# Rutare

Multimea tuturor ruterelor (engl. *communication subnet*)



Context pentru protocoalele de la nivelul retea

## Nivelul retea:

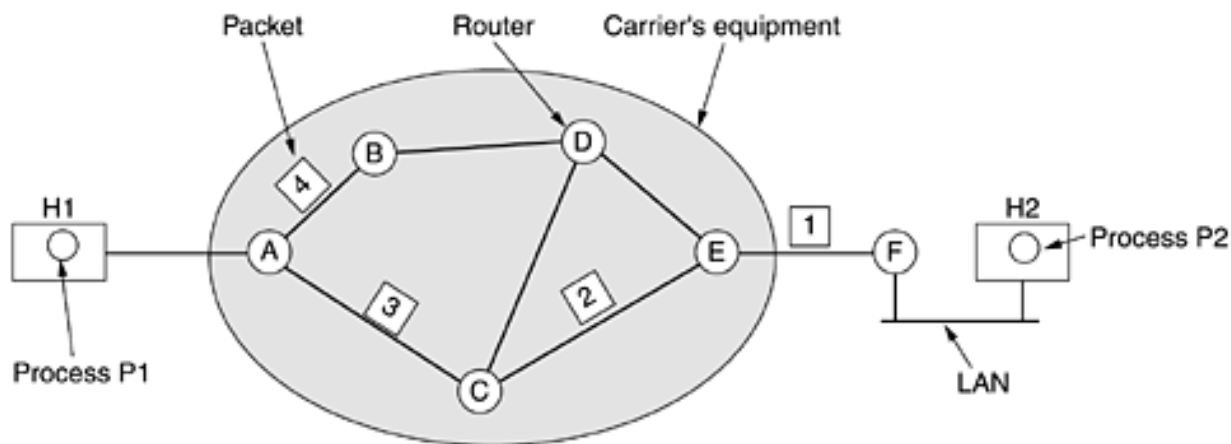
- trebuie sa cunoasca topologia ruterelor si sa aleaga calea pe care un pachet trebuie trimis spre destinatie
- trebuie sa faca alegerea astfel incat sa evite supraincercarea unor linii de comunicatie si a unor rutere (vezi slide-urile urmatoare)

Determinarea  
caii optime de  
rutare

[Computer Networks, 2003  
Andrew S. Tanenbaum]

# Rutare

- In cazul in care la nivelul retea avem servicii neorientate conexiune, pachetele (numite si *datagrame*) sunt trimise individual si sunt rutate in mod independent una de alta



A's table

initially	later
A   -	A   -
B   B	B   B
C   C	C   C
D   B	D   B
E   C	E   B
F   C	F   B
Dest. Line	

C's table

A   A
B   A
C   -
D   D
E   E
F   E

E's table

A   C
B   D
C   C
D   D
E   -
F   F

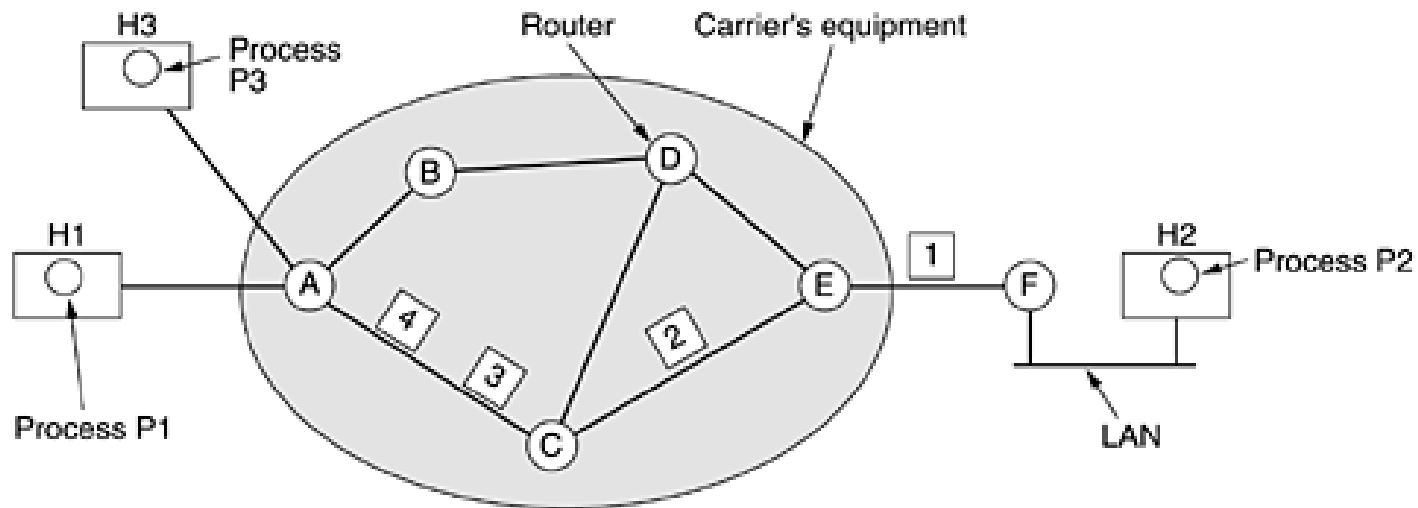
Rutare in interiorul  
unei *datagram subnet*

**Algoritmi de rutare**  
fac managementul  
tabelelor de rutare

[Computer Networks, 2003  
Andrew S. Tanenbaum]

# Rutare

- In cazul in care la nivelul retea avem servicii orientate conexiune se folosesc circuite virtuale si decizia de rutare se ia la stabilirea unui nou circuit



A's table				C's table				E's table			
H1	1	C	1	A	1	E	1	C	1	F	1
H3	1	C	2	A	2	E	2	C	2	F	2
In											

label switching

Rutare in interiorul  
unui *virtual-circuit subnet*  
(*session routing*)

[Computer Networks, 2003  
Andrew S. Tanenbaum]

# Rutare

## Comparatie intre *datagram subnet* si *virtual-circuit subnet*

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

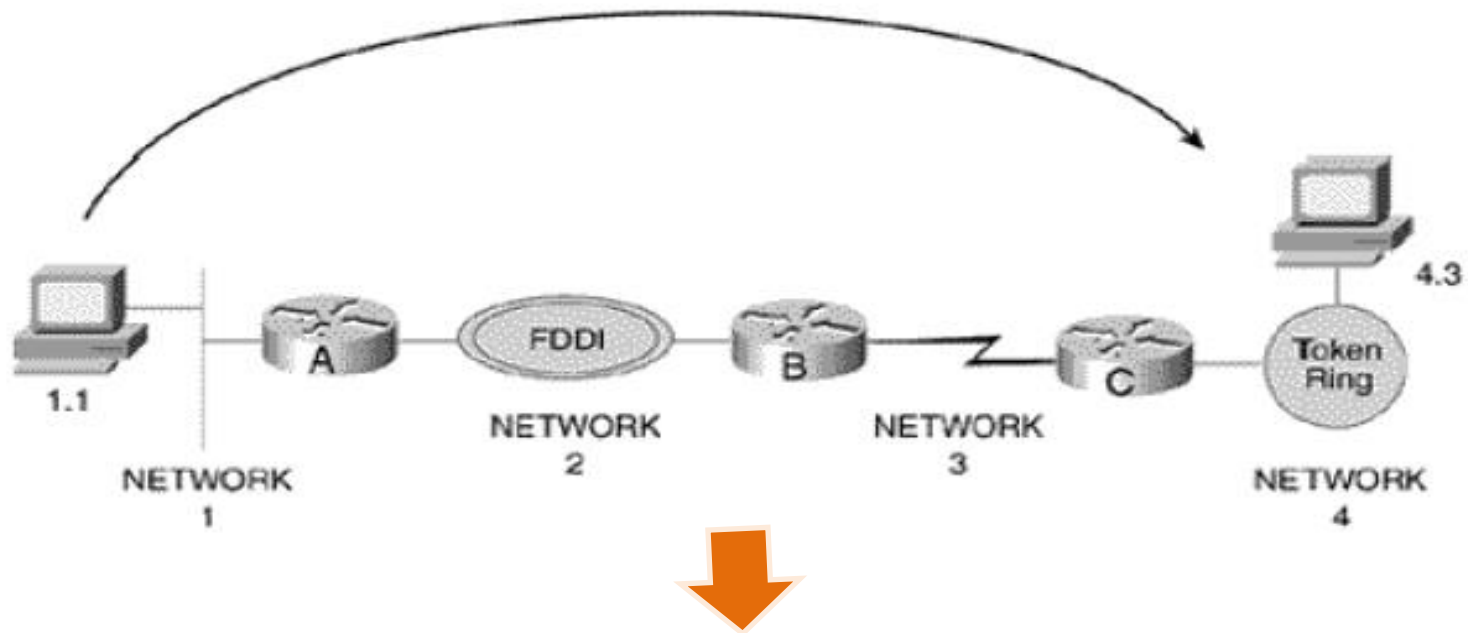
# Rutare

## Determinarea caii de rutare

- Pentru fiecare cale de rutare se determina un cost (metrica)
  - Lungimea caii, siguranta, intarzierea, largimea de banda, incarcarea, costul comunicarii
- Algoritmi de rutare initializeaza si mentin (pentru fiecare gazda) tabele de rutare continand informatii de dirijare
  - Rute catre gazde specificate
  - Rute spre retele specificate
  - O ruta implicita

# Rutare

Un router creeaza o cale logica intre subretele



O aplicatie rulant pe gazda 1.1 nu trebuie sa cunoasca drumul pentru a trimite date aplicatiei de pe gazda 4.3

# Rutare

Algoritmi de rutare - caracteristici:

- **Acuratete** (engl. *Accuracy*) – un algoritm trebuie sa opereze in mod corect si rapid pentru gasirea destinatiei
- **Complexitate redusa** – important pentru rutere cu resurse fizice (soft) limitate
- **Optimalitate** – abilitatea de a gasi ruta optima
- **Robustete** – capacitatea de a functiona corect pentru o perioada lunga de timp, in circumstante diferite
- **Adaptabilitate** – la aparitia unei erori in retea, algoritmul trebuie sa se adapteze (de ex. caderea nodurilor sau coruperea tabelelor de rutare)
- **Convergenta** – algoritmi de rutare trebuie sa converga rapid atunci cand sunt distribuite mesaje de rutare de actualizare
- **Load balancing** – un algoritm de rutare cantareste diferite posibilitati de rutare pentru evitarea legaturilor incete sau a congestiilor

# Rutare

- Abstractizare
  - Retea = graf
  - Dirijarea= gasirea drumului de cost minim de la un nod sursa la un nod destinatie

Tipuri de rutare:

- **Centralizata** – drumul de cost minim poate fi determinat avand disponibile toate informatiile despre retea  
<– **algoritmi folosind starea legaturii**
- **Descentralizata** – drumul de cost minim este determinat in mod iterativ, distribuit (nici un nod nu posedea informatii complete despre costurile legaturilor din retea) <- **algoritmi cu vectori distanta**



# Rutare

- Rutare folosind **starea legaturii**
  - Topologia rețelei & costurile tuturor legaturilor sunt cunoscute
  - Un nod trebuie sa cunoasca identitatile & costurile nodurilor vecine
  - Fiecare nod difuzeaza prin *broadcast* identitatile si costurile tuturor legaturilor de la acel nod la altele

# Rutare

- Rutare **cu vectori distanta**

- Fiecare nod primește informații de la nodurile vecine, realizează calcule și distribuie rezultatele înapoi la vecinii direcți – algoritmul este distribuit și asincron
- Fiecare nod menține o **tabela de distanta** (*distance table*)
- **X**: nodul dorind să realizeze o rutare la nodul **Y** via nodul vecin **Z**
- **$D^X(Y,Z)$** : suma costului legăturii directe între **X** și **Z** ( **$c(X,Z)$** ) plus costul curent al drumului minim de la vecinii lui **Z** la **Y**:  
$$D^X(Y,Z) = c(X,Z) + \min_w \{D^Z(Y,w)\}$$
- Tabela de rutare a unui nod poate fi construită cunoscând tabela de distanta a nodului

# Rutare

## Algoritmi de rutare – clasificare:

- **Statici (neadaptivi)**

- Topologia legaturilor se incarca pentru o perioada de timp in tabelele de rutare a fiecarui nod
- Dezavantaje:
  - Reteaua trebuie sa aiba o dimensiune optima pentru a putea fi controlabila
  - Daca au loc esuuri in retea, nu se poate reactiona imediat

- **Dinamici (adaptivi)**

- Starea retelei este “invatata” din comunicarea rutelor cu vecinii lor; starea fiecarei regiuni din retea este propagata in retea dupa ce toate nodurile isi actualizeaza tabelele de rutare => fiecare ruter poate gasi calea cea mai buna pe baza informatiilor de la nodurile vecine

# Rutare

## Algoritmi de rutare – clasificare:

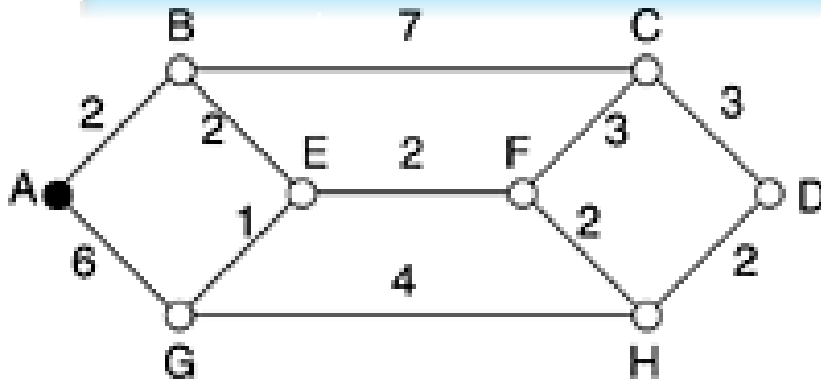
- Statici (neadaptivi)
  - Dirijare pe calea cea mai scurta
  - Inundare (eng. *flooding*)
  - *Deflecting routing* (sau *hot-potato routing*)
- Dinamici (adaptivi)
  - Cu vectori distanta
  - Folosind starea legaturilor
  - Dirijare ierarhica
  - Prin difuziune (*broadcast*)
  - Cu trimitere multipla (*multicast*)

# Rutare

## Algoritmi de rutare – clasificare:

- **Statici** (neadaptivi)
  - **Dirijare pe calea cea mai scurta** (eng. *Shortest path routing*)  
Algoritmul lui Dijkstra (calculeaza drumul de cost minim)
    - Este folosit de protocolul OSPF

“shortest path”: nr. de hopuri => ABC si ABE sunt egale

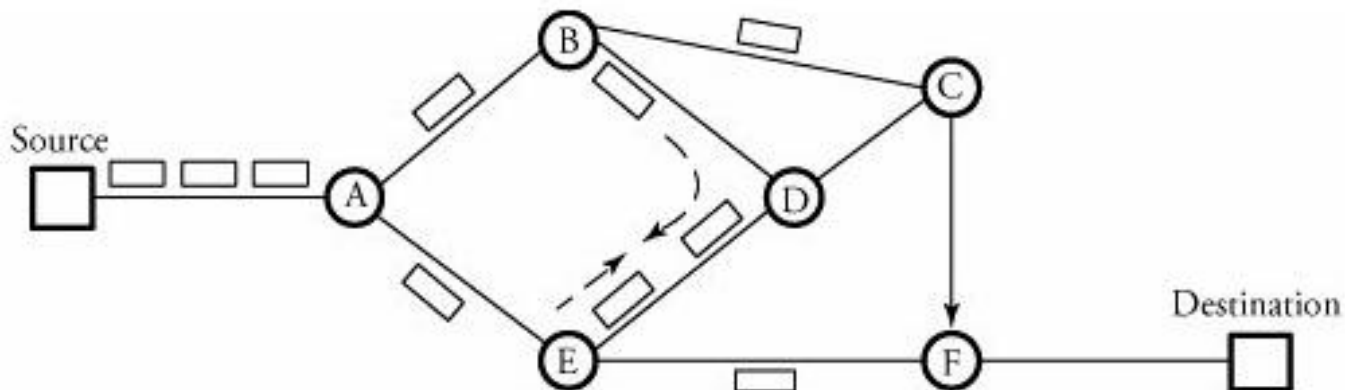


Alte metrice posibile: distanta geografica, largimea de banda, costuri de comunicare etc.

# Rutare

## Algoritmi de rutare – clasificare:

- Statici (neadaptivi)
  - **Inundare** (eng. *flooding*)
    - Un pachet primit este copiat si transmis prin toate legaturile de comunicare (exceptand cea pe unde a venit)
    - Problema: *packet reflection* (un nod poate primi o copie nedorita a unui pachet)
    - Utilizari ale algoritmilor de tip flooding: aplicatii militare, baze de date distribuite, etc.



# Rutare

## Algoritmi de rutare – clasificare:

- Statici (neadaptivi)
  - ***Deflection routing***
    - La fiecare pas un pachet este examinat in raport cu adresa destinatie; daca legatura ceruta este libera pachetul este trimis, altfel este deviat (*deflected*) catre o alta linie de comunicare aleasa aleator;
    - Un pachet are asociat un camp cu o valoare de prioritate care il poate ajuta pe viitor sa castige disputa cu alte pachete

# Rutare

## Algoritmii de rutare – clasificare:

- **Dinamici** (adaptivi)

- **Cu vectori distanta**

- Fiecare router mentine un tabel (vector) cu distanta si linia de comunicare catre destinatie; tabelele sunt actualizate cu informatiile de la vecini

### Algoritmul Bellman-Ford

- Algoritm folosit de protocoalele RIP, BGP, IGRP

### Exemplu:

- Consideram ca metrica: intarzierea (*msec*);
- Routerele vor sti intarzierile asociate vecinilor sai

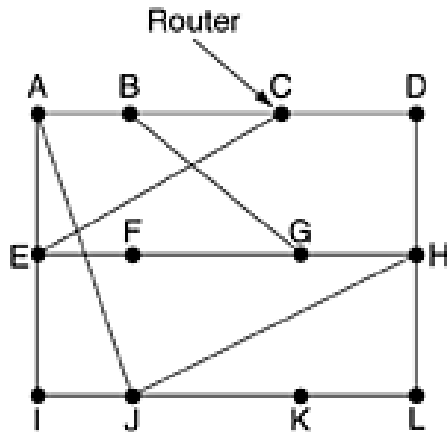


# Rutare

## Algoritmi de rutare – clasificare:

- Dinamici (adaptivi)
  - Cu vectori distanta**

Exemplu:



J doreste sa calculeze ruta catre G

J->A->G = 26 (18+8) msec



J->H->G 18 msec

To	A	I	H	K	New estimated delay from J ↓ Line	
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K

JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6
---------------	----------------	----------------	---------------

Vectors received from J's four neighbors

New routing table for J

[Computer Networks, 2003  
Andrew S. Tanenbaum]

# Rutare

**Problema:** conform algoritmului cu vectori distanta, la fiecare actualizare a rutelor, tabelele de rutare trebuie trimise fiecarui vecin; unele pachete cu informatii legate de dirijare trec pe ruta de pe care deja au venit (*reverse route*)

Intrebare: Pot fi evitate rutele de tip *reverse*?

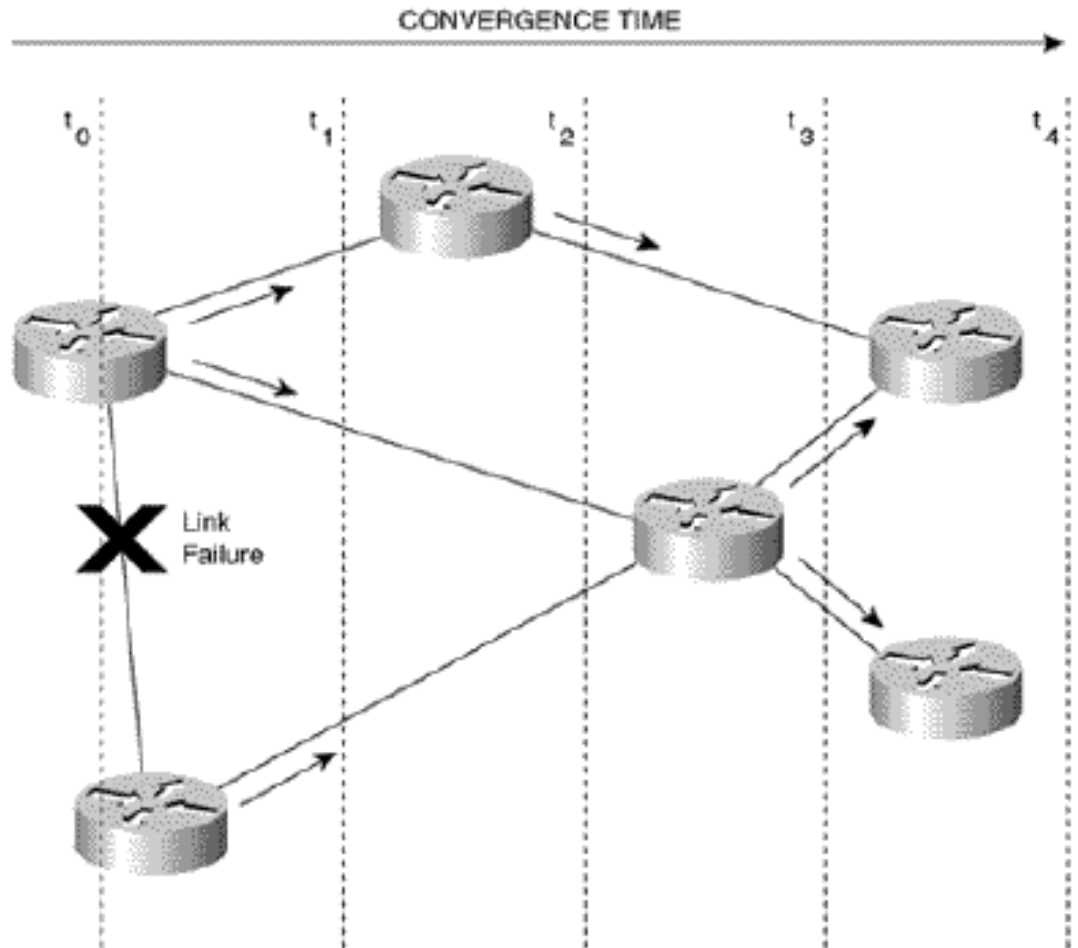
Raspuns: utilizarea tehnicii ***split horizon***

- Cand *router*-ul trimite actualizari de rute folosind o anumita interfata de retea, ele nu vor fi expediate retelelor ale caror rute au fost invatate din actualizari primite via acea interfata

# Rutare

## Problema:

Modificarea  
topologiei  
(deprecierea  
convergentei  
algoritmilor de  
rutare)



[Rețele de calculatoare –  
curs 2007-2008, Sabin Buraga]

# Rutare

## Algoritmii de rutare – clasificare:

- Dinamici (adaptivi)
  - **Folosind starea legaturilor**

Fiecare router trebuie sa:

- Descopere vecinii si sa le “invete” adresele de retea
- Masoare intarzierea sau costul asociat fiecarui vecin
- Construiasca un pachet prin care anunta pe “toti” ceea ce a invatat

Dilema: cand trebuie construite pachetele? (de ex. periodic sau cand apare un eveniment special)

- Trimita pachetul
- Calculeze cea mai scurta cale catre fiecare router

# Rutare

## Algoritmi de rutare – clasificare:

- Dinamici (adaptivi)
  - **Dirijare ierarhica**

Necesitate: in retele de mari dimensiuni nu este fezabil ca un router sa aiba cate o intrare despre fiecare alt router;

Mecanism: Ruterele stiu detalii asociate unei *regiuni*, dar nu stiu detalii despre structura interna a altor regiuni

Obs.:

- Pentru retele de dimensiuni mari, ierarhia de nivel 2 nu este suficienta si atunci regiunile se grupeaza in clustere, clusterelor in zone, zonele in grupuri, etc.
- Care este numarul optim de niveluri?

Pentru un *subnet* cu  $N$  routere numarul optim  **$\ln N$**

[Kamoun&Kleinrock, 1979]

# Rutare

## Algoritmi de rutare – clasificare:

- Dinamici (adaptivi)
  - Prin difuziune (***broadcast routing***)

Utilizare: actualizarea stocurilor (de la bursa de valori), *streaming multimedia*, serviciu de distribuire a rapoartelor despre vreme, etc.

Modalitati:

- **Sursa trimite cate un pachet distinct fiecarui destinatar**

Obs.: Metoda ineficienta: neutilizarea latimii de banda;  
Sursa trebuie sa aiba adresele tuturor destinatarilor

- ***Flooding*** - util cind alte metode nu pot fi aplicate

Problema: Se genereaza prea multe pachete si se consuma multa latime de banda

# Rutare

## Algoritmi de rutare – clasificare:

- Dinamici (adaptivi)
  - **Cu trimitere multipla (*multicast routing*)**

Exemplu de utilizare:

Un proces doreste sa transmita un mesaj unui grup de procese implementand un sistem de baze de date distribuite

Obs.: Se poate face *broadcast*, insa uneori informatia nu este destinata a fi vazuta de oricine

Mecanism: *router*-ul va face periodic o interogare asupra *host*-urilor care apartin unui grup; apoi informatia este propagata catre routere

# Rutare

Exemplu:

```
[adria@thor ~] $ netstat -r
Kernel IP routing table
Destination      Gateway          Genmask          Flags      MSS  Window  irtt  Iface
webapps.vpn      *                255.255.255.255  UH          0  0        0  tun1
172.18.100.2     *                255.255.255.255  UH          0  0        0  tun0
85.122.23.0      *                255.255.255.128  U           0  0        0  eth2
172.18.100.0     172.18.100.2    255.255.255.0    UG          0  0        0  tun0
172.17.17.0      webapps.vpn      255.255.255.0    UG          0  0        0  tun1
default          gw-info-c.uaic.  0.0.0.0          UG          0  0        0  eth2
[adria@thor ~] $ netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask          Flags      MSS  Window  irtt  Iface
172.17.17.1      0.0.0.0          255.255.255.255  UH          0  0        0  tun1
172.18.100.2     0.0.0.0          255.255.255.255  UH          0  0        0  tun0
85.122.23.0      0.0.0.0          255.255.255.128  U           0  0        0  eth2
172.18.100.0     172.18.100.2    255.255.255.0    UG          0  0        0  tun0
172.17.17.0      172.17.17.1     255.255.255.0    UG          0  0        0  tun1
0.0.0.0          85.122.23.126   0.0.0.0          UG          0  0        0  eth2
```

[[http://docstore.mik.ua/oreilly/networking\\_2ndEd/tcp/ch02\\_04.htm](http://docstore.mik.ua/oreilly/networking_2ndEd/tcp/ch02_04.htm)]

Exemplu: Pentru destinatia 172.17.17.0 routerul (gateway-ul) folosit este 172.17.17.1 ; Gateway = 0.0.0.0  
-> interfata de retea locala

Flag-uri: U(*up*) – ruta este operationala; H - indica o ruta catre o anumita gazda; G - ruta utilizeaza un gateway exterior

Crearea tabelor de rutare

- Rute statice: comanda UNIX **route**

- Descoperirea unui router prin ICMP

- Protocol de tip broadcast care descopera routerele unei retele locale



# Rutare

## Protocole de rutare - clasificare

- *Intradomain routing protocol* – realizeaza rutarea pachetelor intr-un domeniu
  - RIP (Routing Information Protocol)
  - OSPF (Open Shortest Path First)
- *Interdomain routing protocol* – realizeaza rutarea pachetelor intre domenii
  - BGP (Border Gateway Protocol)
  - EGP (Exterior Gateway Protocol)
    - RFC 827, 904
    - nu mai este utilizat, fiind inlocuit de BGP


# Rutare

## RIP (Routing Information Protocol)

- RFC 1058, 1723
- Mecanismul de functionare:
  - Se aplica algoritmul Bellman-Ford (pentru *host*-uri si routere)
  - Pentru fiecare router, se creeaza un **vector** continand costul rutei si alte informatii
  - Daca survin modificari intr-un punct, acestea sunt propagate periodic la routerele si *host*-urile vecine cu acel punct

# Rutare

## RIP (Routing Information Protocol)

- Foloseste mesaje IP
  - Fiecare *router* trimite un *broadcast* continand intreaga tabela de rutare a *router*-ului – la fiecare 30 sec.
  - O intrare a tablei de rutare RIP contine:
    - Adresa IP
    - Metrica (numarul de hop-uri: 1-15)
    - Timeout (in secunde)
  - Retelele conectate direct au metrica =1 (un hop)
- 
- Tabela de rutare A: nodul B e la 1 *hop* distanta (conexiune directa), nodul C la 2 *hop*-uri
- Daca o ruta da timeout, metrica devine 16 (nu exista conexiune) si ruta e stearsa dupa 1 minut


# Rutare

## RIP (Routing Information Protocol)

- Daca o informatie de rutare se modifica (de ex. o legatura sau un *router* esueaza), propagarea acestei schimbari are loc foarte lent – RIP sufera de convergenta lenta
- RIP
  - Este un protocol matur, stabil, larg suportat si usor de implementat
  - Este indicat a fi folosit de sistemele autonome de dimensiuni reduse fara rute redundante
  - In practica este inlocuit in majoritatea situatiilor de OSPF

# Rutare

## OSPF (Open Shortest Path First)

- RFC 1247, 2328
  - Fiecare router ce foloseste OSPF cunoaste starea intregii topologii de retea (algoritm folosind starea legaturii) si transmite actualizari la toate routerele
- 
- Conduce la trafic aditional, care poate conduce la congestii  
OSPF permite ca traficul sa fie distribuit pe rute cu costuri similare (*load balancing*)  
OSPF suporta rutarea dupa tipul serviciilor (ToS)
    - protocolul IP contine campul ToS (in general neutilizat)
  - Convergenta mai rapida
  - Oferă suport pentru folosirea mai multor tipuri de metrici

# Rutare

## OSPF (Open Shortest Path First)

- Opereaza intr-o ierarhie de entitati de retea

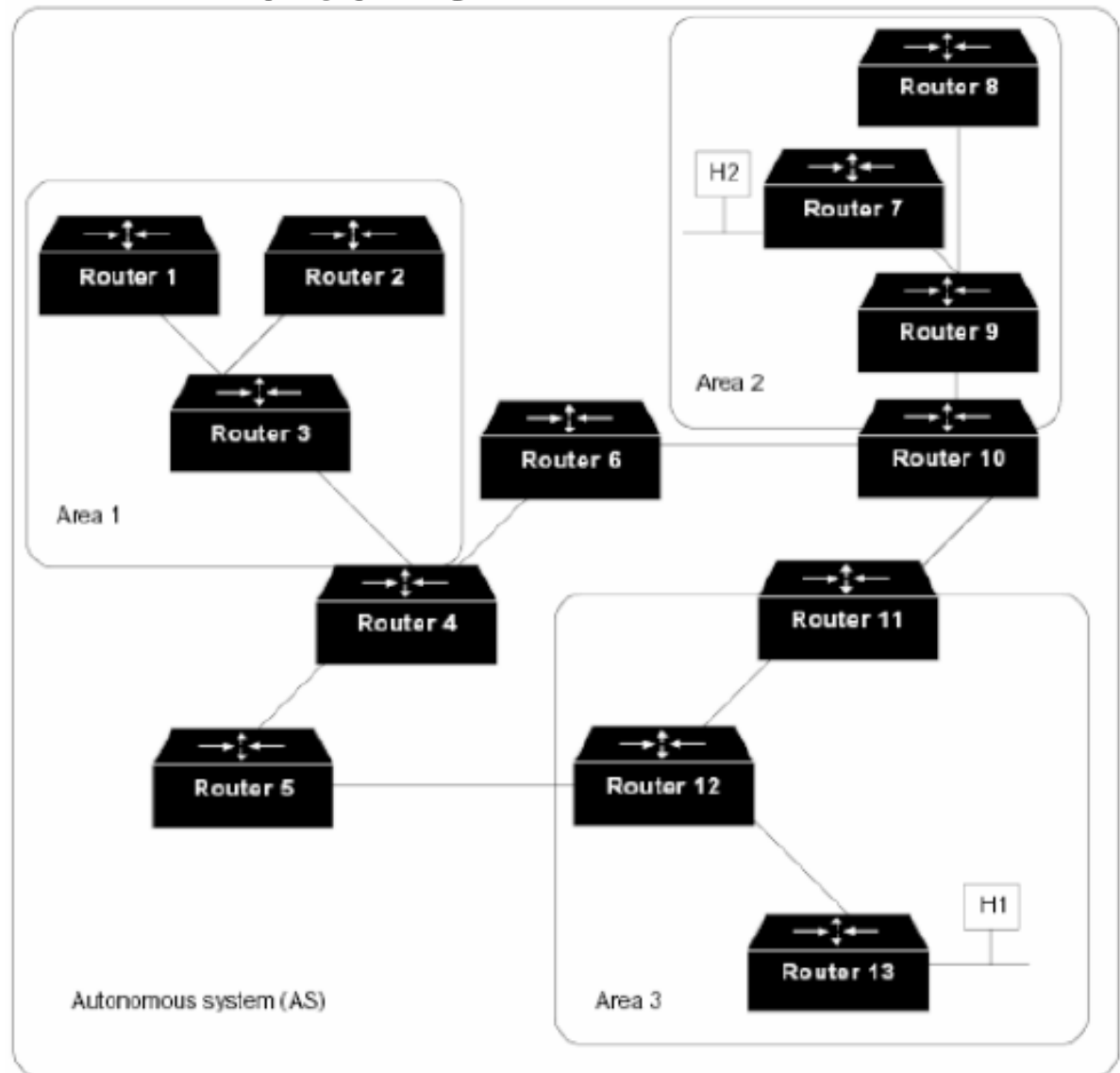
Motivatie: retele de dimensiuni mari => un router nu poate cunoaste intreaga topologie

- **Sistemul autonom (AS)** – colectie de retele care partajeaza aceeasi strategie de dirijare
- Un AS este divizat in **domenii** (engl. **areas**) – grupuri contigue de retele si gazde; routerele au aceeasi informatie privitoare la topologie si ruleaza acelasi algoritm
- **Coloana vertebrala** (*backbone sau area 0*) – responsabila cu distributia informatiilor de rutare intre domenii; orice router conectat la doua sau mai multe domenii face parte din *backbone* (aceste routere vor rula algoritmi corespunzatori pt fiecare domeniu)

# Rutare

OSPF (Open  
Shortest  
Path First)

Un AS si  
domeniile  
sale  
conectate  
via routere



# Rutare

## OSPF (Open Shortest Path First)

Tipuri de mesaje OSPF:

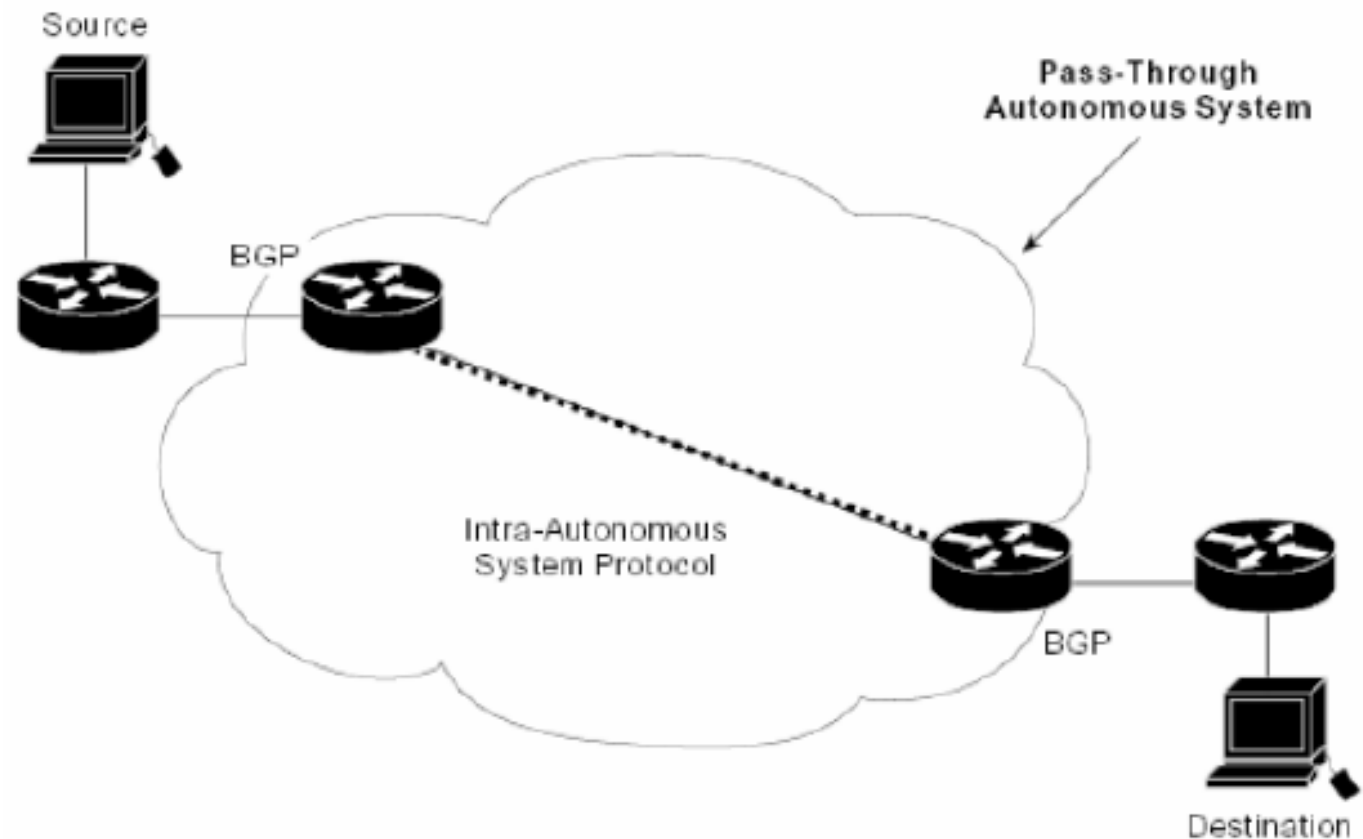
Message type	Description
Hello	Used to discover who the neighbors are
Link state update	Provides the sender's costs to its neighbors
Link state ack	Acknowledges link state update
Database description	Announces which updates the sender has
Link state request	Requests information from the partner

- Cu un mesaj “hello” un router isi afla vecinii (de ex. toate routerele din LAN)
- Fiecare router face *flood* periodic cu un mesaj (ce are asociat un numar de secventa) de tipul *Link state update*; la aceste mesaje se fac confirmari *Link state ack*
- *Database description* furnizeaza numerele de secventa asociate intrarilor legaturilor avute in evidenta de emitator



# Rutare

## BGP (Border Gateway Protocol)



[Rețele de calculatoare –  
curs 2007-2008, Sabin Buraga]

# Rutare

## BGP (Border Gateway Protocol)

- Utilizat pentru comunicarea intre routere aflate in sisteme autonome diferite
- Functii majore:
  - *Neighbor relationship* – se refera la acordul dintre routerele din doua sisteme autonome de a schimba informatii pe baza unor reguli (un router poate refuza stabilirea unei astfel de relatii in functie de: regulile domeniului, supraincarcare etc)
  - *Neighbor maintenance* – routerele isi vor trimite mesaje de tip *keep-alive*
  - *Network maintenance* – fiecare router tine o baza de date cu subretelele existente pentru o rutare eficienta in acea subretea

# Rutare

## BGP (Border Gateway Protocol)

- Exista patru tipuri de pachete BGP:
  - *Open*: folosit pentru stabilirea unei relatii dintre doua routere
  - *Update*: contine informatii actualizate despre rute
  - *Keep-alive*: folosit pentru confirmarea de relatii stabilite anterior
  - *Notification*: folosit atunci cand apar erori
- Perechile de routere BGP comunica intre ele folosind conexiuni TCP
- BGP este un protocol bazat pe vectori distanta cu urmatoarele diferente:
  - nu se pastreaza doar costul asociat unei destinatii, ci se mentine si calea catre acea destinatie
  - nu se furnizeaza vecinilor doar costul estimat, ci si calea exacta
- RFC 1771-1774, 4271

# Rutare

Alte protocoale:

- Interior Gateway Routing Protocol (IGRP)
  - Imbunatatire CISCO a RIP
- Enhanced IGRP (EIGRP)
- Simple Multicast Routing Protocol (SMRP)
  - Rutare de fluxuri multimedia la Apple (via AppleTalk)  
Obs.: Din 2009 AppleTalk este nesuportat, se utilizeaza TCP/IP
- Resource Reservation Protocol (RSVP) (RFC 2205)
  - Nu este un protocol de rutare, dar ofera functionalitati similare
  - Asigura calitatea serviciilor IP

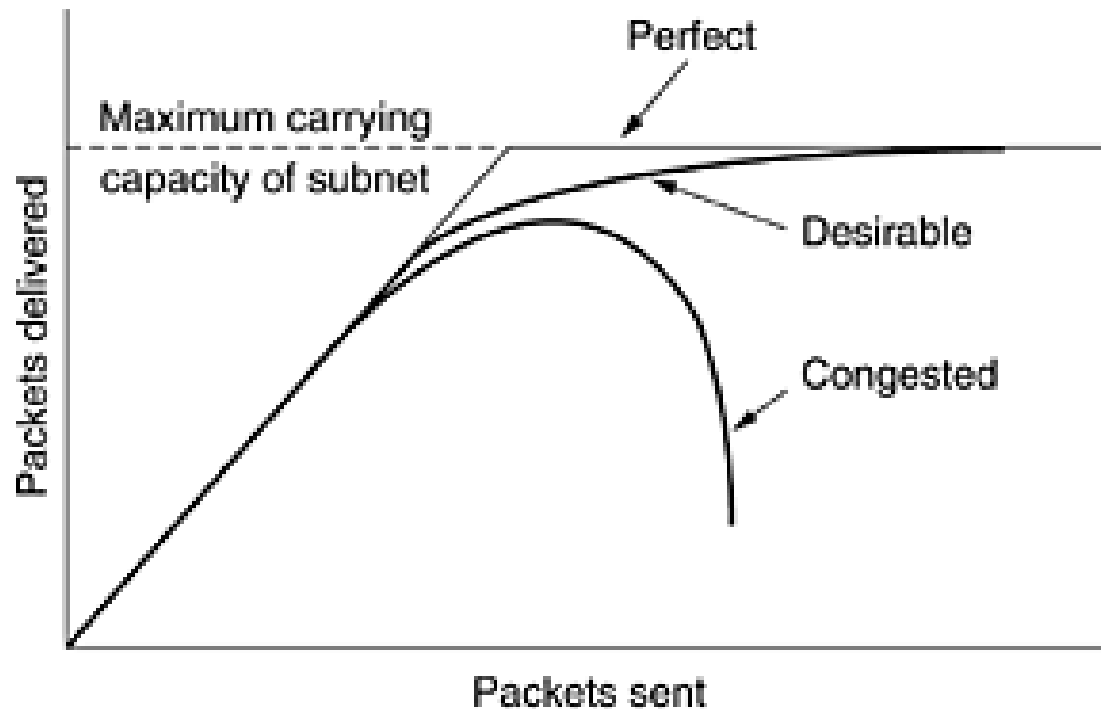
# Rutare | privire de ansamblu

## Rutare interna:

- RIP (Routing Information Protocol)
- IGRP (Interior Gateway Routing Protocol)
- EIGRP (Enhanced IGRP )
- OSPF (Open Shortest Path First)
- IS – IS (Intermediate System to Intermediate System)  
pentru ISO/OSI
- Rutare externa
  - BGP (Border Gateway Protocol)
  - EGP (Exterior Gateway protocol)

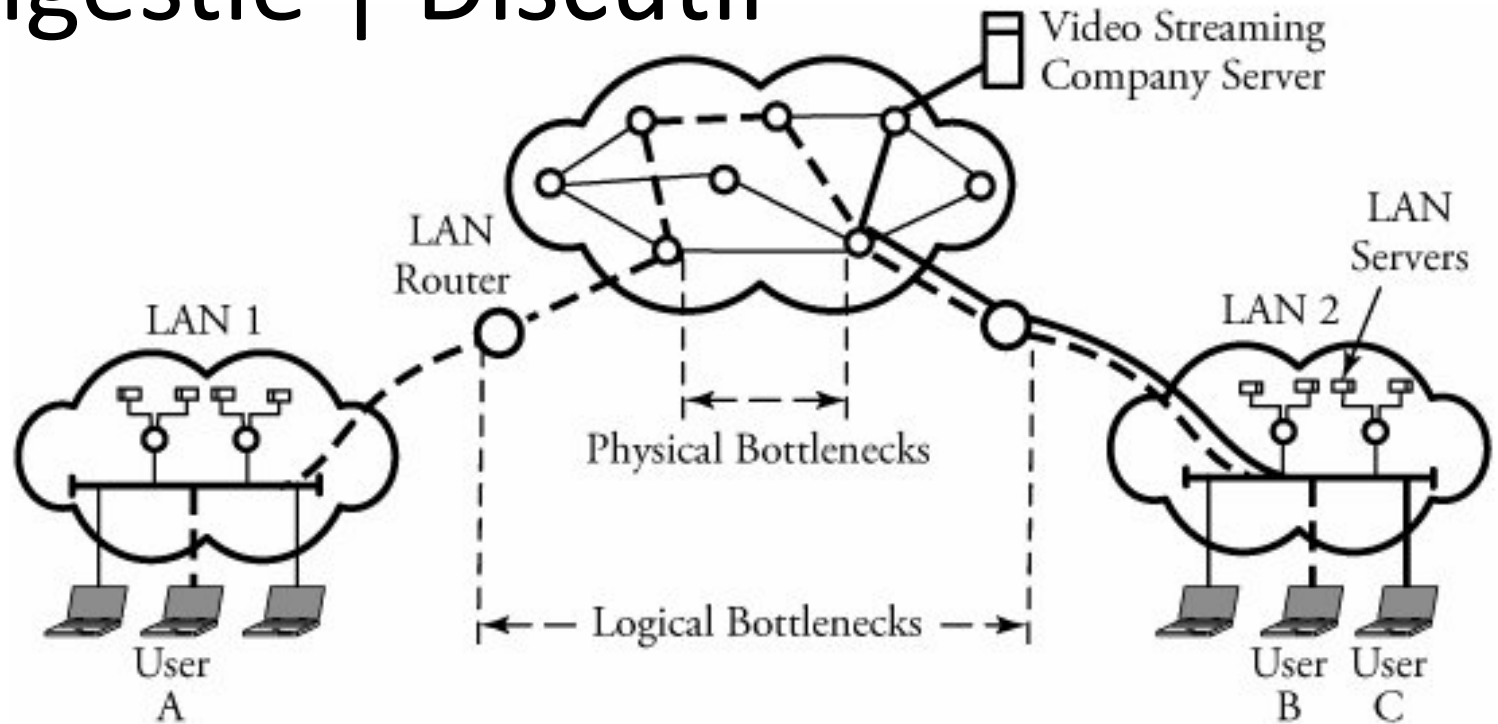
# Congestie | Discutii

- Apare atunci cand se realizeaza supraincarcarea resurselor unei retea



In cazul unui trafic foarte mare, se poate instala congestia si performantele scad brusc

# Congestie | Discutii



Congestia poate aparea:

- La nivelul legaturii de date: cand latimea de banda nu este suficienta
- La nivelul retea: cand coada de pachete de la noduri nu poate fi controlata
- La nivelul transport: cand legatura logica dintre doua rutere aflate intr-o sesiune de comunicare nu mai poate fi controlata

# Congestie | Discutii

- Controlul congestiei – solutii
  - **Open-loop**: rezolvarea inseamna de fapt prevenirea aparitiei congestiilor printr-un design si decizii potrivite
  - **Close-loop**
    - Monitorizarea sistemului pentru detectarea congestiilor

*Metrici*: procentul de pachete eliminate datorita lipsei spatiului in *buffer*, intarzierea pachetelor, etc.
    - Trimiterea acestei informatii la nodurile care pot lua decizii
    - Ajustarea operatiilor pentru corectarea problemei



# Congestie | Discutii

Obs.:

controlul congestiei != controlul fluxului

- Controlul congestiei asigura faptul ca retea are capacitatea de a transporta traficul oferit; implica actiunile tuturor *host*-urilor si a routerelor
- Controlul fluxului se ocupa de comunicarea *point-to-point* dintre un emitator si un receptor si se asigura faptul ca un emitator nu transmite date mai repede decat poate receptorul sa le proceseze

# Rezumat

- **Nivelul retea**
  - **Activitatea de rutare (dirijare)**
    - **Preliminarii**
    - **Caracterizare**
    - **Rutare**
    - **Protocole de rutare**
      - **RIP & OSPF**
      - **BGP & EGP**
  - **Congestie – discutii generale**

# Bibliografie

**Content Networking Fundamentals**, Silvano Da Ros, Publisher: Cisco Press Pub  
Date: March 30, 2006 Print ISBN-10: 1-58705-240-7 Print ISBN-13: 978-1-58705-240-8 Pages: 576

**Computer Networks**, Andrew S. Tanenbaum, Publisher : Prentice Hall

**Computer and Communication Networks**, Nader F. Mir, Publisher: Prentice Hall Pub Date: November 02, 2006 Print ISBN-10: 0-13-174799-1 Print ISBN-13: 978-0-13-174799-9 Pages: 656

<http://www.tuxick.net/linux/ip6routing.html>

<http://www.6diss.org/workshops/see-2/routing-external.pdf>

<http://www.ip6.com/us/book/Chap7.pdf>

[http://www.nanog.org/meetings/nanog44/presentations/Monday/SmithBonica\\_IPv6\\_N44.pdf](http://www.nanog.org/meetings/nanog44/presentations/Monday/SmithBonica_IPv6_N44.pdf)



# Intrebari?

Intrebari?