Università di Genova
Facoltà di Ingegneria

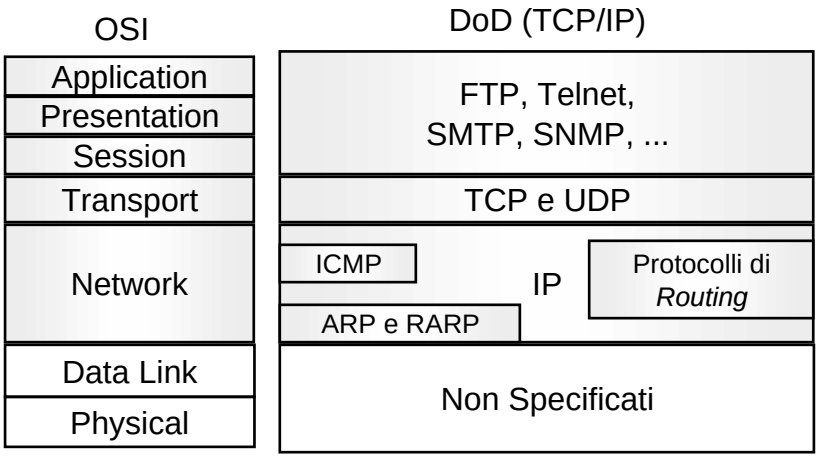
Telematica

6. TCP/IP - Introduzione ed IP

Prof. Raffaele Bolla

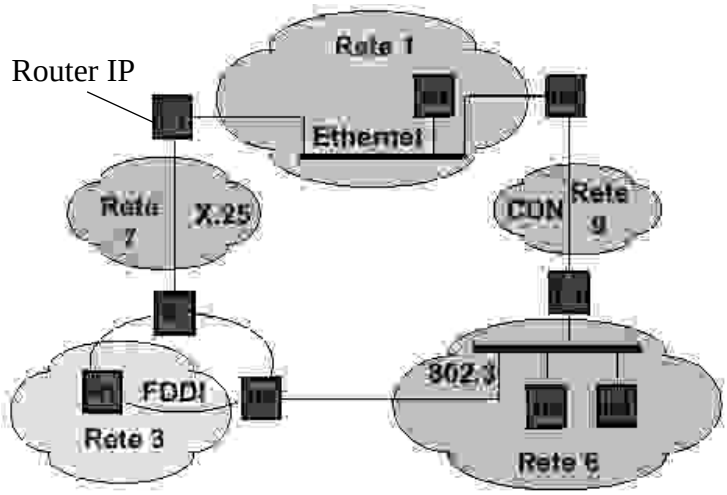
dist

Architettura



6.2

Struttura



6.3

Internetworking Protocol (IP)

- È il livello Network di TCP/IP ed è il protocollo principale di questa architettura
- Offre un servizio non connesso.
- Semplice protocollo di tipo Datagram.
- E` specificato nel RFC (Request For Comments) 791.
- La versione attuale è la 4 (IPv4), anche se quella successiva è già stata completamente definita come 6 (IPv6).

6.4

IP - Datagram

- I pacchetti viaggiano su percorsi indipendenti
- *Out of order delivery*
- Gestione della banda difficoltoso
  - riservare e garantire banda
  - rifiutare connessioni (*Call Acceptance Control*)
- Meno complesso: non richiede negoziazione né lato utente, nè all'interno della rete
- Robusto: si adatta a variazioni di traffico,topologia, guasti
- Adatto al traffico dati (*bursty*)

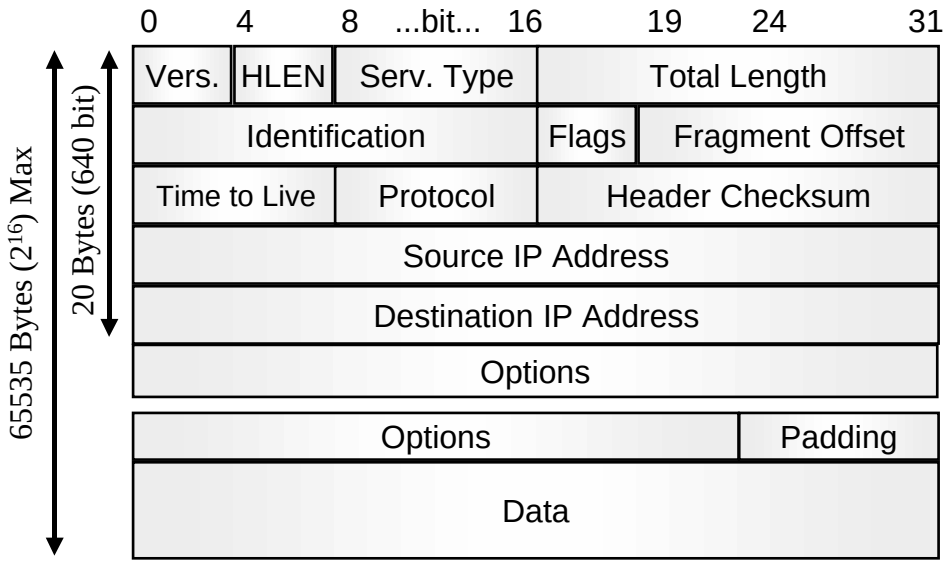
6.5

IP

- Gestione indirizzi a 32 bit a livello di rete e di *host*
- *Routing*
- Frammentazione e riassemblaggio dei pacchetti
- Rivelazione (ma non correzione) di errori (sull'intestazione)

6.6

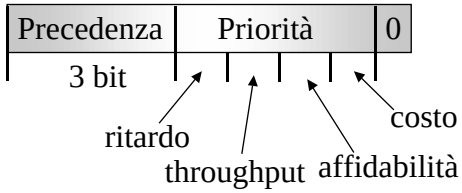
IP Datagram



6.7

IP Datagram

- Versione
- Header Length
  - lunghezza dell'header, in blocchi di 4 byte (max 64 byte)
- Service Type
  - tipo di servizio, attualmente non usato ma importante in prospettiva.



6.8

IP Datagram

- Total Length (16 bit)
  - lunghezza globale del pacchetto corrente (non quello prima della frammentazione), max 2^{16}-1 = 65.535 ottetti (576 ottetti il minimo che un router deve saper gestire senza frammentazione).
- Identification (16 bit)
  - ID univoco del pacchetto (costante nel caso di frammentazione), necessario per la deframmentazione
- Flags (3 bit)
  - 0: posto a zero
  - DF: Don't Fragment
  - MF: More Fragments (0 sull'ultimo frammento)

6.9

IP Datagram

- Fragment Offset (13 bit)
  - In multipli di 8 byte, quindi è in grado di rappresentare fino a 65 535 byte.
- Time To Live (8 bit)
  - contatore decrementato ad ogni hop (una unità in meno per ogni secondo di attesa nel router).
- Protocol (8 bit)
  - TCP (6), UDP (17), ICMP, ...
- Header Checksum
- Source - Destination IP Address

6.10

IP Datagram

Copy	Class	Number
1 bit	2 bit	5 bit

- Il *copy bit* decide se le opzioni vanno copiate in tutti i pacchetti in caso di frammentazione
- Classe 0 denota controllo della rete o del datagram, classe 2 *debugging* o misure.
- Le opzioni possibili sono:
  - *Loose and strict source routing*
  - *Record / trace route*
  - *Timestamp*

6.11

IP Datagram

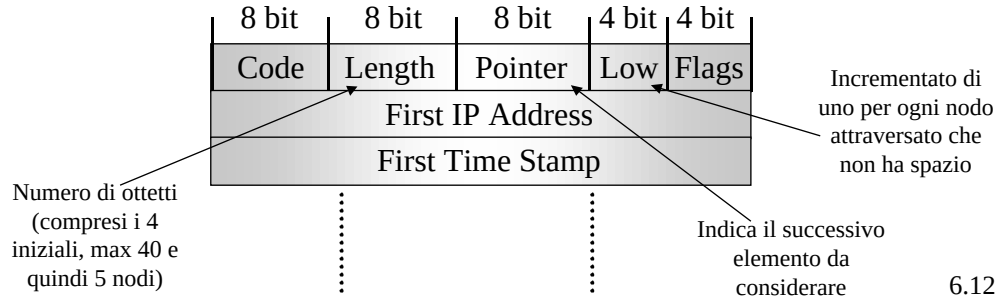
Opzione Timestamp

Tre tipi di comportamento:

Flag = 0: Registra solo i *timestamp*

Flag = 1: Registra *timestamp* e gli indirizzi IP

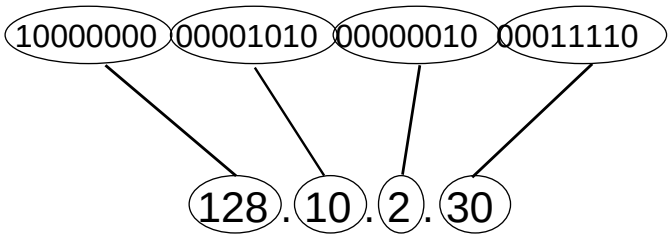
Flag = 3: La lista degli indirizzi viene inserita dalla sorgente, i *timestamp* vengono inseriti solo dai nodi attraversati presenti nella lista



6.12

Indirizzi IP

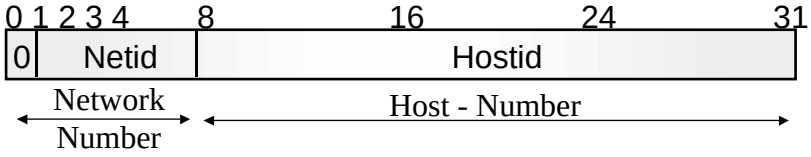
- Gli indirizzi IP sono indirizzi univoci, assegnati da una autorità centrale, e hanno una lunghezza di 32 bit.
- Tali indirizzi sono composti da due parti:
  - l'indirizzo della rete (netid)
  - l'indirizzo del *host* (hostid)
- L'indirizzo è legato alle interfacce di rete.



6.13

Indirizzi IP

Classe A - /8



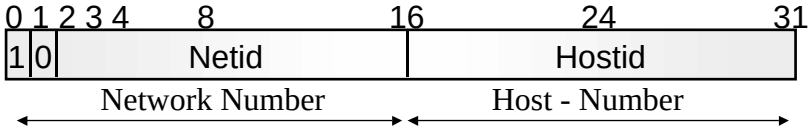
Netid validi	Hostid validi
1.x.x.x - 127.x.x.x	**x.0.0.1 - x.255.255.254**
Max num. di reti	Max numero di host
126 (2^7-2)	**16.777.214** (2^{24}-2)
(non va contata la rete 127 e la 0)	(non va contato l'ind. tutti 0 e quello tutti 1)

Considerando che lo spazio complessivamente disponibile è di 4.294.967.296 indirizzi (2^{32}) le reti di classe A coprono circa il 50% dello spazio di indirizzamento disponibile)

6.14

Indirizzi IP

Classe B - /16



Netid validi
128.1.x.x - 191.254.x.x
Max num. di reti
16.384 (2^{14})

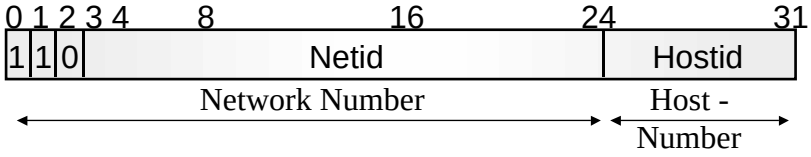
Hostid validi
x.x.0.1 - x.x.255.254
Max numero di host
65.534 (2^{16}-2)

Comprendono circa 1.073.741.824 (2^{30}) indirizzi, la classe B rappresenta circa il 25% dello spazio di indirizzamento complessivo

6.15

Indirizzi IP

Classe C - /24



Netid validi
192.0.1.x - 223.255.254.x
Max num. di reti
2.097.152 (2^{21})

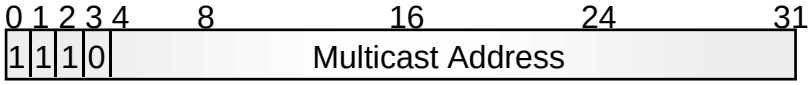
Hostid validi
x.x.x.1 - x.x.x.254
Max numero di host
254 (2^{8}-2)

Comprendono circa 536.870.912 (2^{29}) indirizzi, la classe C rappresenta circa il 12,5% dello spazio di indirizzamento complessivo

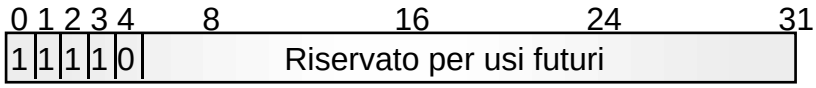
6.16

Indirizzi IP

Classe D **(224.0.0.1 - 239.255.255.254)**

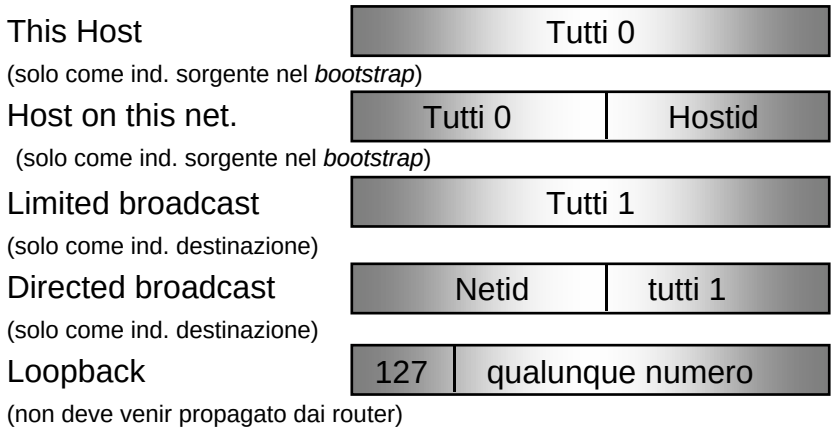


Classe E **(240.0.0.1 - 255.255.255.254)**



6.17

Indirizzi IP particolari



6.18

Indirizzi IP - Netid particolari

- Alcuni netid sono riservati per essere usati su reti private
- Non sono “annunciati” su Internet, quindi non sono raggiungibili direttamente

172.16.0.0 - 172.31.255.255	16 indirizzi di classe B
10.0.0.0 - 10.255.255.255	1 network di classe A
192.168.0.0 - 192.168.255.255	256 indirizzi di classe C

6.19

IP

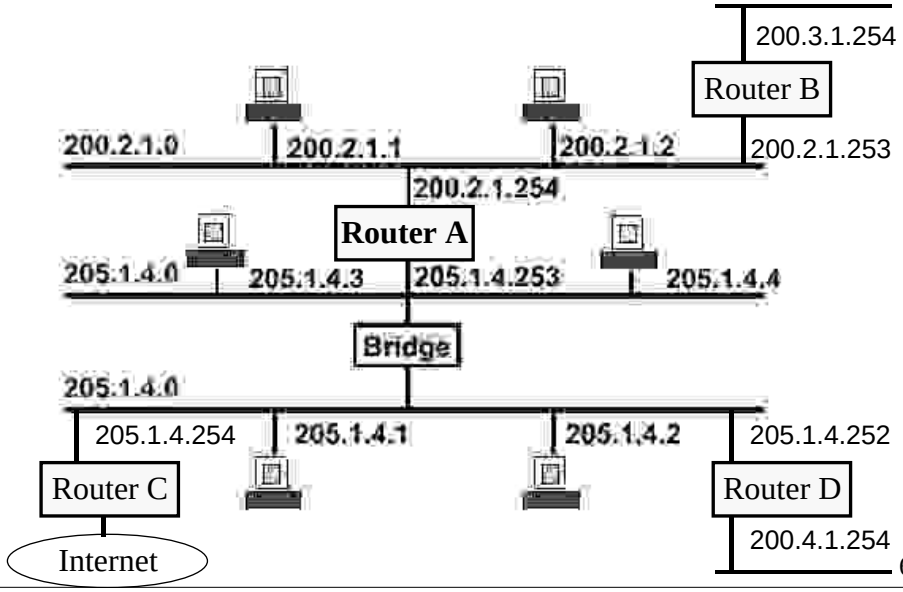
Instradamento

- IP assume una corrispondenza biunivoca tra reti fisiche (intese come dominii di *broadcasting* a livello 2) e logiche.
  - Si osservi che
    - » le realizzazioni moderne ammettono
      - più reti logiche sulla stessa rete fisica.
      - più reti fisiche nella stessa rete logica (Proxy ARP)
    - » Una conseguenza del fatto che l’indirizzo contenga l’identificatore di una rete è che quando una macchina viene fisicamente spostata il suo indirizzo deve essere modificato.

6.20

IP - Instradamento

Esempio



6.21

IP - Instradamento

Reti fisiche e reti logiche

- Punto-punto
  - le interfacce possono essere "*unnumbered*" (es.: linee dedicate o dial-up)
- Multiaccesso con possibilità di *broadcast*
  - gli host possono comunicare direttamente senza passare per router intermedi (es. : le LAN)
- Multiaccesso senza possibilità di broadcast
  - gli host possono comunicare direttamente senza passare per router intermedi (es. : reti a pacchetto commutate)

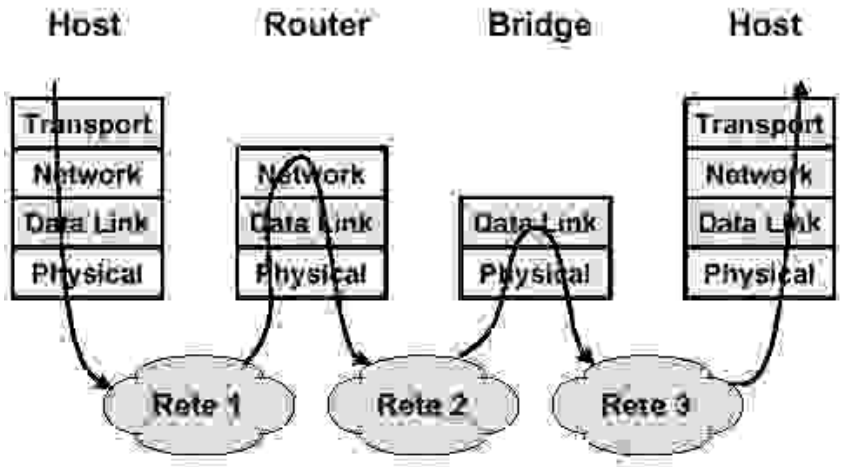
6.22

IP
Instradamento

- L'instradamento viene quindi realizzato:
  - all'interno di una rete logica, in modo implicito, direttamente dalla stazione sorgente
    - » ossia deve essere realizzato dal livello 2 e quindi si deve fornire la "traduzione" dell'indirizzo IP di destinazione in indirizzo di livello due (ARP -RARP)
  - Tra reti logiche diverse è gestito esplicitamente dai router, ossia è il router che deve ricevere il pacchetto e che quindi instradarlo verso la sottorete opportuna
    - » La stazione sorgente deve quindi indirizzare il pacchetto al router, questo implica che:
      - Deve essercene almeno uno direttamente connesso alla rete fisica
      - La stazione deve conoscerne l'indirizzo (default router)

6.23

IP
Instradamento



6.24

IP - Instradamento

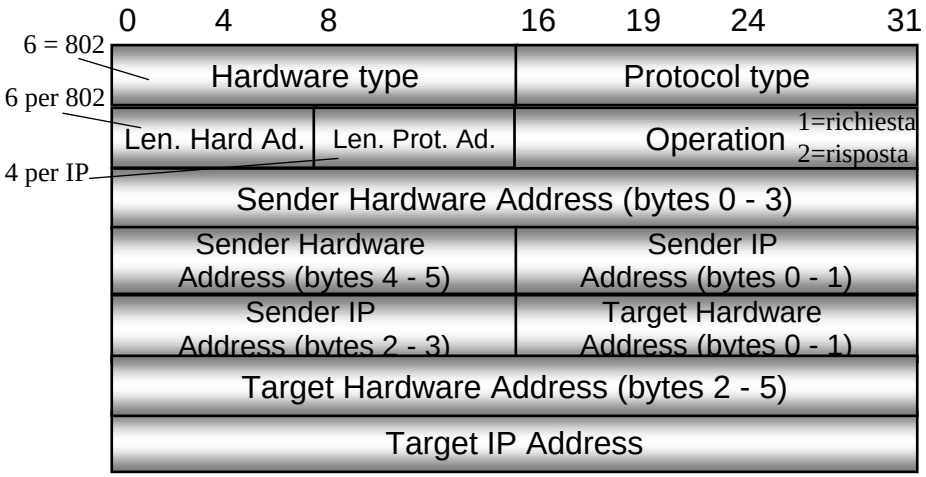
(R)ARP

- I protocolli ARP (*Address Resolution Protocol*) e RARP (*Reverse Address Resolution Protocol*) servono per definire in modo automatico le corrispondenze fra indirizzi di livello 2 ed indirizzi IP e viceversa.
  - ARP viene usato tutte le volte che una stazione vuole inviare un pacchetto ad un'altra stazione sulla sottorete, di cui conosce solo l'indirizzo IP.
  - RARP viene usato dalle stazioni non dotate di memoria di massa (*diskless*) per reperire il propio indirizzo IP all'avvio (*bootstrap*).
- Si appoggiano direttamente sui protocolli di livello 2 della sottorete e non su IP.

6.25

IP - Instradamento

ARP



6.26

IP - Instradamento

ARP

- La stazione A manda in *broadcast* un pacchetto ARP contenente l'indirizzo IP di cui vuol conoscere il corrispondente indirizzo di livello 2.
- La stazione B che riconosce il proprio ind. IP risponde fornendo il suo indirizzo di livello 2.
- Con il primo pacchetto ARP la stazione A fornisce anche il proprio indirizzo di livello 2, così che B può risponderle senza usare un *broadcast*.
- La corrispondenza resta memorizzata in una memoria di cache per un certo periodo.

6.27

IP - Instradamento

Instradamento nei router

- Nei router l'intradamento è realizzato tramite una tabella di instradamento (*Routing Table*, RT) del tipo:
  - **Destinazione** (rete o host) — **Indirizzo di invio**

200.2.1.0	Invio diretto sulla interf. 1
205.1.4.0	Invio diretto sulla interf. 2
200.3.1.0	200.2.1.253
200.4.1.0	205.1.4.252
default	205.1.4.254

6.28

IP - Instradamento

Instradamento nei router

- Due sono quindi gli aspetti di cui si compone l'instradamento:
  - Esecutivo: la scelta della direzione di uscita tramite la tabella
  - Algoritmico: la compilazione / aggiornamento della tabella
- Il secondo aspetto si realizza tramite
  - il calcolo del percorso migliore eseguito secondo un qualche algoritmo
  - Lo scambio di informazioni fra i router per eseguire tale calcolo

6.29

IP - Instradamento

Instradamento nei router

- Per rendere l'instradamento efficiente si deve mantenere le RT di piccole dimensioni.
- Tabelle grandi:
  - Richiedono più tempo per l'individuazione della corretta direzione di uscita (*next hop*)
  - Sono di difficile gestione in fase di calcolo e di aggiornamento.
- La suddivisione net e host, crea una gerarchia che ha l'obiettivo di ridurre la dimensione delle RT.
- Lo stesso vale per la presenza del "default router"

6.30

IP - Instradamento

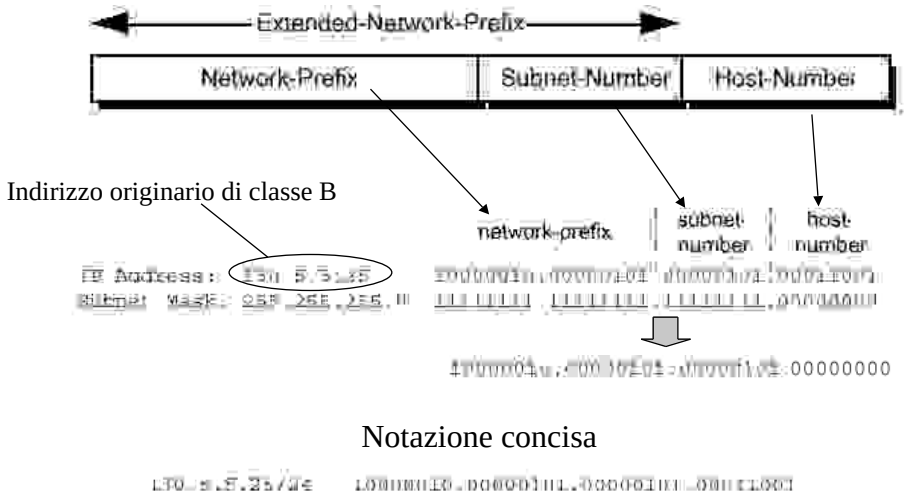
Routing Table

- Tipiche informazioni contenute nelle RT per ciascune delle reti destinazione sono
  - Indirizzo della rete destinazione
  - Maschera di *subnet*
  - Indirizzo IP del successivo *router* da attraversare (*next hop*) o sul fatto che la destinazione è direttamente raggiungibile
  - Porta di uscita del "*next hop*"
  - Metrica (anche più di una)
  - Identificatore della sorgente dell'instradamento (manuale, locale, ICMP, uno degli algoritmi di instradamento)

6.31

IP - Instradamento

Subnetting



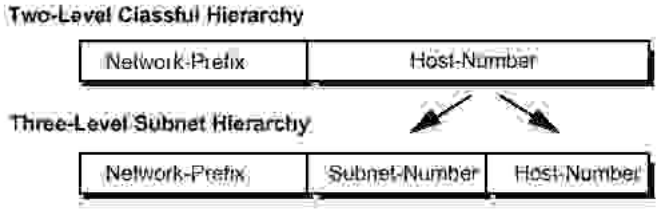
Notazione concisa

6.32

IP - Instradamento

Subnetting

- Il *subnetting* aggiunge un livello di gerarchia all'indirizzamento, contribuendo a ridurre la dimensione delle RT.

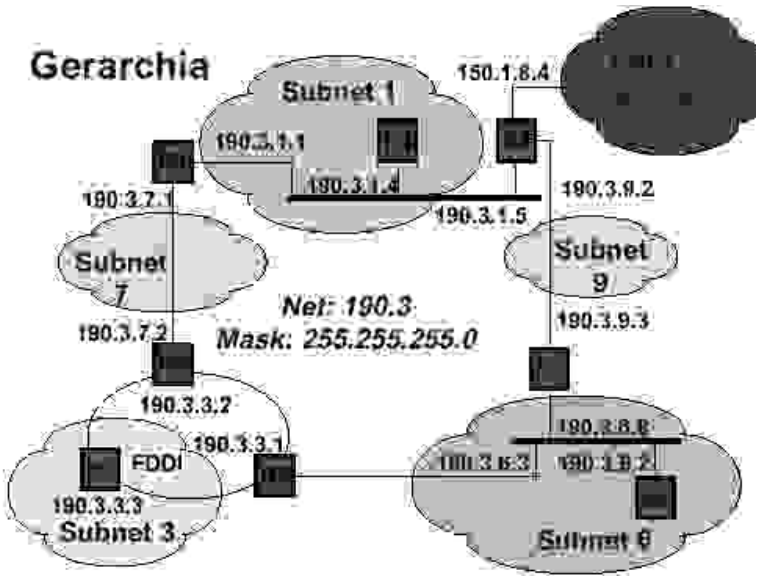


- All'interno di una classe di indirizzi la destinazione è ora individuata dalla coppia (*IP-address*, *Netmask*)

6.33

IP - Instradamento

Subnetting



6.34

IP - Instradamento

Subnetting

- Il router R5 avrà una tabella del tipo

Destinazione		next hop
190.3.6.0/24	255.255.255.0	diretta
190.3.3.0/24	255.255.255.0	diretta
190.3.1.0/24	255.255.255.0	190.3.3.2
190.3.7.0/24	255.255.255.0	190.3.3.2
190.3.9.0/24	255.255.255.0	190.3.6.8
150.1.0.0/16	255.255.0.0	190.3.6.8
Default		190.3.6.8

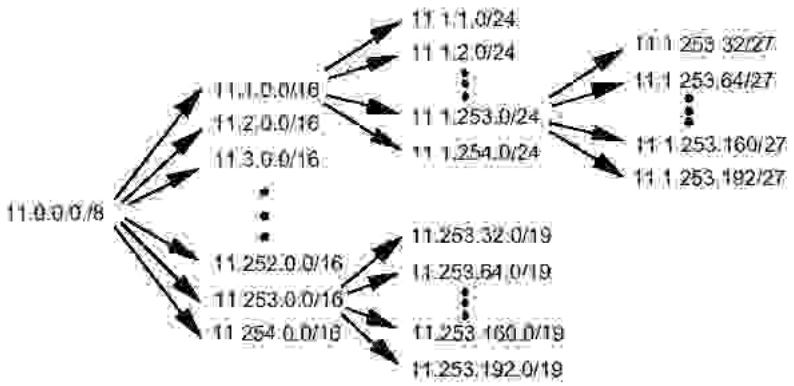
- Gli host dovranno conoscere il proprio ind. IP, la Netmask e l'indirizo del router di default. ad es. per l'host H3 avrà:
  - Ind. IP = 190.3.6.2, NetMask = 255.255.255.0 (ff.ff.ff.0), default router = 190.3.6.8.

6.35

IP - Instradamento

Variable Length Subnetting

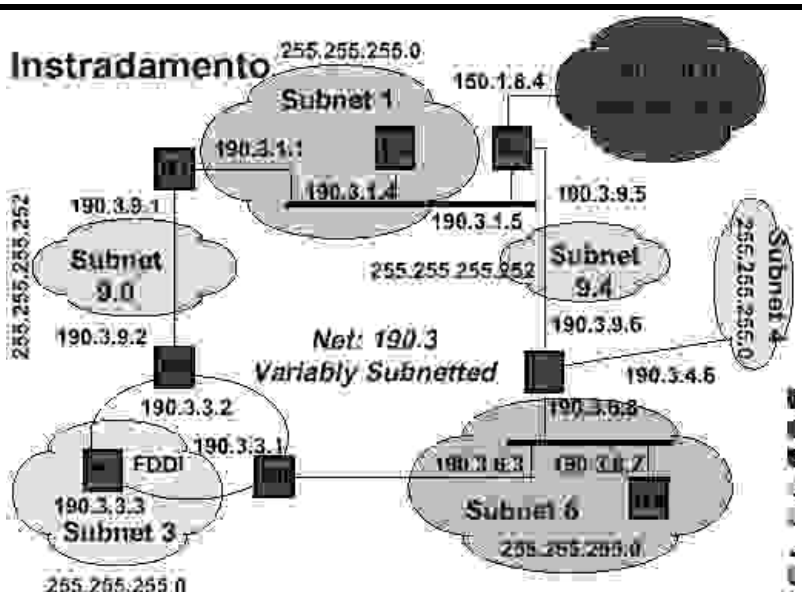
- All'interno di una singola classe si possono realizzare gerarchie multiple:



6.36

IP - Instradamento

Esempio



6.37

IP - Instradamento

Variable Length Subnetting

- La maschera variabile permette di sfruttare meglio lo spazio di indirizzamento e ridurre ulteriormente le RT.
- In presenza di più di una scelta per una destinazione si sceglie quella con *subnet mask* più lunga
- Si consideri ad es. la RT di R5

Destinazione		***Next hop***
190.3.6.0/24	255.255.255.0	diretta
190.3.3.0/24	255.255.255.0	diretta
190.3.9.0/24	255.255.255.0	190.3.3.2
190.3.9.4/30	255.255.255.252	190.3.6.8
190.3.4.0/24	255.255.255.0	190.3.6.8
190.3.0.0/16	255.255.0.0	190.3.3.2
150.1.0.0/16	255.255.0.0	190.3.6.8
Default		190.3.6.8

6.38

IP - Instradamento

Proxy ARP

- E' una tecnica che permette la corrispondenza di più reti fisiche ad una sola rete o sottorete logica .
- Un *router* risponde agli ARP verso indirizzi IP che sa non appartenere a quella rete fisica con il proprio indirizzo di livello 2.
- L'appartenenza o meno ad una rete viene ricavata attraverso un *subnetting* che solo il *router* è obbligato a conoscere.

6.39

IP - Instradamento

Classless InterDomain Routing (CIDR)

- La crescita degli utenti nella rete ha velocemente portato verso l'esaurimento lo spazio di indirizzamento disponibile.
- La ragione principale è legata al fatto che in molte situazioni le reti di classe C sono troppo piccole, quindi viene richiesto un indirizzo di classe B di cui però va sprecato gran parte dello spazio di indirizzamento.
- Per cui si è definito un meccanismo di "supernetting" che consiste nel accorpare indirizzi di classe C contigui in un unico spazio di indirizzamento creando suddivisioni netid-hostid ad hoc.

6.40

IP - Instradamento

Classless InterDomain Routing (CIDR)

- Il CIRD trasforma lo spazio di indirizzamento della classe C in un unico spazio “senza classe” che viene suddiviso usando come “quanti” le reti di classe C con un meccanismo di subnetting.
- I router in grado di gestire tale meccanismo, operano usando la coppia indirizzo IP - netmask per identificare il “*next-hop*”
- Quindi “annunciano” la coppia ed in presenza della netmask ignorano la definizione di classe C (effettuano un *supernetting*)

6.41

IP - Instradamento

Classless InterDomain Routing (CIDR)

- Si supponga che ad un ISP sia stato assegnato il blocco di indirizzi 206.0.64.0 / 18.
- Questo blocco rappresenta 16.384 IP *address* che possono essere interpretati come 64 reti da x / 24.
- Se un cliente richiede 800 *host addresses*, invece che assegnargli una classe B (e perdere 64.700 indirizzi che non ha) o quattro Classi C (introducendo 4 nuove reti nelle *routing table* di Internet), l’ ISP può assegnare al cliente il blocco 206.0.68.0 / 22, con 1.024 indirizzi IP

```
ISP:        206.0.64.0/18 11001110.00000000.01000000.00000000
Client:     206.0.68.0/22 11001110.00000000.01000100.00000000
Class C #0: 206.0.68.0/24 11001110.00000000.01000100.00000000
Class C #1: 206.0.69.0/24 11001110.00000000.01000101.00000000
Class C #2: 206.0.70.0/24 11001110.00000000.01000110.00000000
Class C #3: 206.0.71.0/24 11001110.00000000.01000111.00000000
```

6.42

IP - Instradamento

Classless InterDomain Routing (CIDR)

CIDR prefix length	Dotted-Decimal	# Individual Addresses	# of Classful Networks
/13	255.248.0.0	512 K	8 Bs or 2048 Cs
/14	255.252.0.0	256 K	4 Bs or 1024 Cs
/15	255.254.0.0	128 K	2 Bs or 512 Cs
/16	255.255.0.0	64 K	1 B or 256 Cs
/17	255.255.128.0	32 K	128 Cs
/18	255.255.192.0	16 K	64 Cs
/19	255.255.224.0	8 K	32 Cs
/20	255.255.240.0	4 K	16 Cs
/21	255.255.248.0	2 K	8 Cs
/22	255.255.252.0	1 K	4 Cs
/23	255.255.254.0	512	2 Cs
/24	255.255.255.0	256	1 C
/25	255.255.255.128	128	1/2 C
/26	255.255.255.192	64	1/4 C
/27	255.255.255.224	32	1/8 C

6.43

IP - Instradamento

Network Address Translation (NAT)

- Consiste nell'usare nella rete una delle classi assegnate all'uso privato e quindi non "annunciate" e far tradurre al router di accesso ad Internet gli indirizzi verso l'esterno (usando uno spazio di indirizzi "ufficiale" ridotto) in modo dinamico.
- Esistono dei limiti di questo metodo:
  - Con UDP e TCP si deve ricalcolare il *checksum*
  - FTP ha il numero IP scritto in ASCII nel suo interno, cambiarlo può cambiare la lunghezza del pacchetto e avere effetti sul TCP.
  - ICMP ha l'indirizzo IP nella parte dati
  - Tutte le applicazioni che trasportano l'indirizzo IP possono aver problemi.

6.44

Instradamento

- Requisiti
  - Minimizzare lo spazio occupato dalle RT per:
    - » Velocizzare la commutazione
    - » Semplificare i router (meno cari)
    - » Ridurre l'informazione necessaria all'aggiornamento
  - Minimizzare il traffico di controllo
  - Essere robusto, ossia evitare:
    - » Cicli
    - » Buchi neri
    - » Oscillazioni
  - Ottimizzare i percorsi (dal punto di vista della distanza, del ritardo, del costo economico, …)

6.45

Instradamento

Alternative

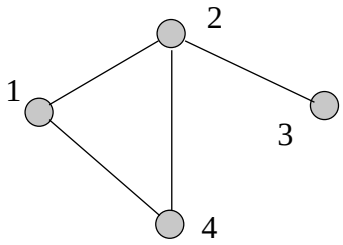
- Centralizzato o **distribuito** (o isolato)
- Basato sulla sorgente o ***"hop-by-hop"***
- **Deterministico** o stocastico
- **Singolo percorso** o multi-percorso
- Dipendente dallo stato (**dinamico**) o indipendente dallo stato (**statico**)
- ***Distance Vector*** o ***Link State***.

6.46

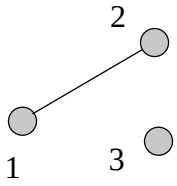
Algoritmi di instradamento

Grafo

- Un **grafo** (*graph*) $G = \{N, A\}$ e un insieme non vuoto N di nodi e un insieme A di coppie di nodi appartenenti ad N detti **archi** (*arc*).



N = {1, 2, 3, 4}
A ={(1, 2), (1, 4), (2, 3), (2, 4)}



N = {1, 2, 3}
A ={(1, 2)}

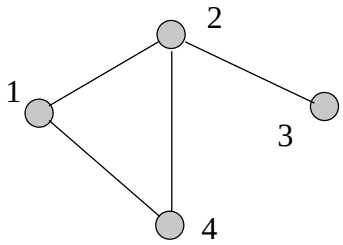
N = {1}
A ={ }

6.47

Algoritmi di instradamento

Cammino e percorso

- Un **cammino** (*walk*) è una sequenza di nodi (n_1, n_2, …, n_L) tali che (n_l, n_{l+1}), l = 1,…, L-1 sono archi di G.
- Un cammino (n_1, …, n_L) con $L \geq 3$ e $n_1 = n_L$ è detto **ciclo**.
- Un cammino senza nodi ripetuti è detto **percorso** (*path*).



Es. di cammini:
(1, 4, 2, 1, 4, 1), (2, 3, 2), (1, 4, 2)

Es. percorsi (*path*):
(1, 4, 2, 3), (1, 2, 3)

6.48

Algoritmi di instradamento

Alberi

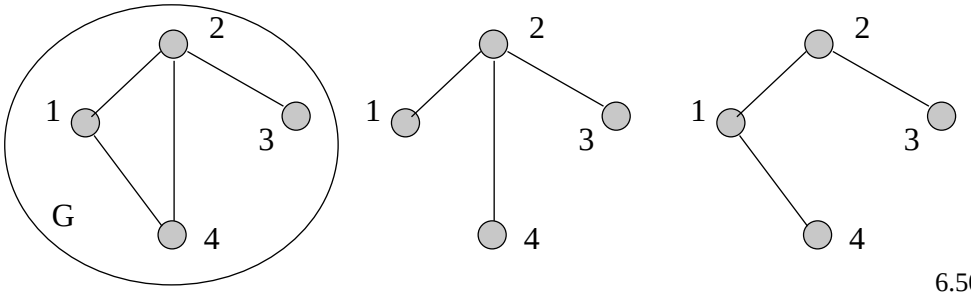
- Un grafo è detto **connesso** se per ogni nodo i esiste un percorso ($i = n_1,\ldots,n_L = j$) verso ogni nodo j.
- **Lemma**
  Sia *G* un grafo connesso e sia *S* un sottoinsieme di *N*, allora esiste almeno un arco (i, j) con i *S* e j *S*.
- *G*' ={*N*', *A*'} è un **sottografo** (*subgraph*) di *G* se *N*' *N* e *A*' *A*.

6.49

Algoritmi di instradamento

Alberi

- Un **albero** (*tree*) è un grafo connesso che non contiene cicli.
- Uno ***spanning tree*** di un grafo *G* è un sottografo di *G* che è un albero e contiene tutti i nodi di *G* (*N*' = *N*).



6.50

Algoritmi di instradamento

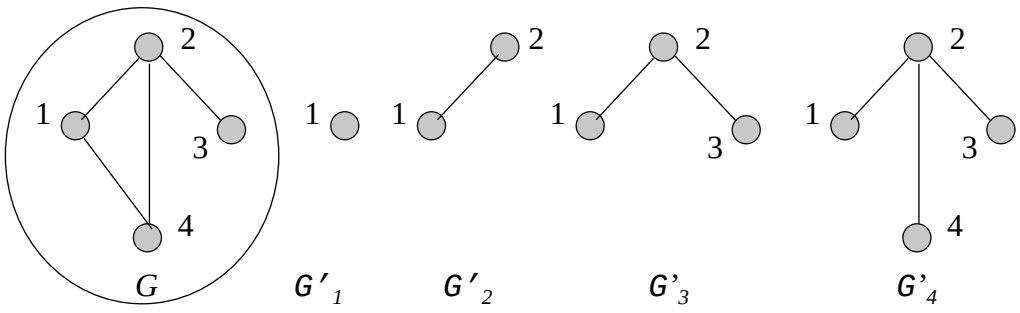
Spanning Tree

- Si può costruire uno *spanning tree* con il seguente algoritmo:
  - Sia N = {n}, con n un nodo arbitrario di N, e A' un insieme vuoto.
  - Se $N' = N$, allora $G' = (N', A')$ è uno *spanning tree*, altrimenti si esegua il passo successivo
  - Sia (i, j) A un arco con i N', j N-N', si ponga
    $N' = N' \cup \{j\}$
    $A' = A' \cup \{(i, j)\}$
    e si torni al passo precedente

6.51

Algoritmi di instradamento

Spanning Tree



- Sia G un grafo connesso con N nodi e A archi allora
  - G contiene almeno uno *spanning tree*;
  - A ≥ N-1;
  - G è un albero solo se A = N-1;

6.52

Algoritmi di instradamento

Minimum weight Spanning Tree (MST)

- Se si associa ad ogni arco (i, j) un **peso** (*weight*) $\mathbf{w_{ij}}$ che riassume il "costo" di una trasmissione lungo di esso, la somma dei costi degli archi di un SP rappresenta il costo del *broadcast* di un messaggio sull'albero.
- Allora potrebbe essere conveniente cercare lo *spanning tree* la cui somma dei pesi degli archi sia minima, ossia il

  Minimum weight Spanning Tree **(MST).**

6.53

Algoritmi di instradamento

Minimum weight Spanning Tree (MST)

- Si chiami frammento ogni sotto-albero (*sub-tree*) di un MST.

Proposizione

Dato un frammento *F* ed essendo α = (i, j) l'arco con peso minimo per cui i appartiene ad *F* e j no, allora *F* esteso con l'arco (i, j) e il nodo j è a sua volta un frammento.

- Questa proposizione può essere usata per realizzare algoritmi che trovino un MST.

6.54

Algoritmi di instradamento

Grafi orientati

- Consideriamo ora un **Grafo Orientato** (*directed graph*), i cui archi sono orientati ossia sono coppie ordinate di nodi.
- Tutte le definizioni date per il grafo, (ossia cammino, ciclo, percorso, albero) possono essere ripetute per il grafo orientato.
- Si associ ad ogni arco (i, j) un valore d_{ij} genericamente indicato come distanza. Dato un **percorso diretto** (*direct path*) $p = (n_1, n_2, \ldots, n_L)$ fra n_1 e n_L, definiamo come lunghezza del percorso

$$D = \sum_{r=1}^{L-1} d_{n_r n_{r+1}}$$

- Si osservi che se $d_{ij} = 1$, la lunghezza del percorso corrisponde al numero di archi che lo compone.

6.55

Algoritmi di instradamento

Shortest path

- Il problema del "**percorso minimo**" o ***shortest path*** è quello di trovare il percorso p fra i e j tale che D sia minimo.
- Esistono diversi metodi per risolvere questo problema, uno di questi prende il nome di **Algoritmo di Bellman-Ford**.
- Tale algoritmo fissata una destinazione, trova il percorso minimo da ogni nodo a tale destinazione nell'ipotesi non ci siano distanze negative ($d_{ij} \geq 0$).

6.56

Algoritmi di instradamento

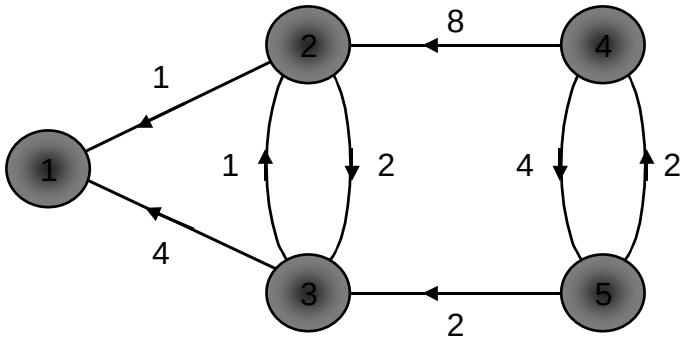
Shortest path - Bellman Ford

- Definendo
  - 1 come il nodo destinazione
  - $d_{ij} = \infty$ se (i, j) non è un arco (i nodi i e j non sono direttamente connessi),
  - D_i^h come la lunghezza del percorso più corto fra il nodo i ed il nodo 1, contenente al massimo h archi
  - $D_i^h = \infty$ se tale percorso non esiste
  - $D_1^h = 0, \quad h$, per convenzione
  - $D_i^0 = \infty$ per tutti gli $i \neq 1$
- l'iterazione dell'algoritmo di **Bellman-Ford** è
  $$D_i^{h+1} = \min_j \{d_{ij} + D_j^h\} \quad \text{per ogni } i \neq 1$$
- L'algoritmo ha termine quando $D_i^{h+1} = D_i^h \quad i$

6.57

Algoritmi di instradamento

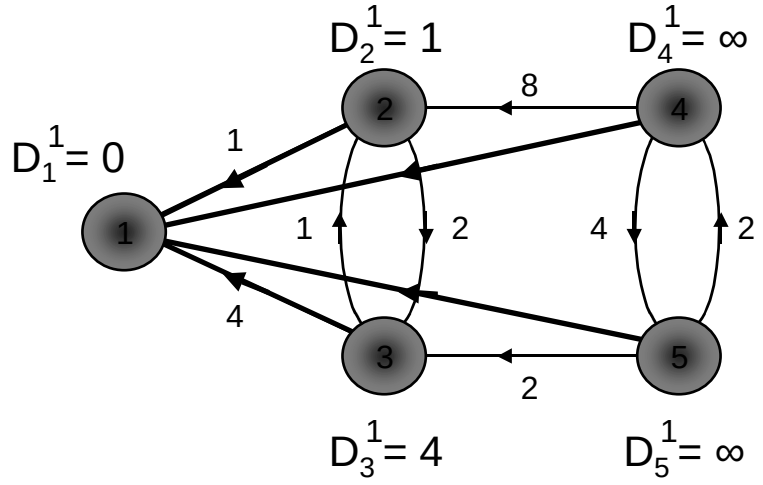
Bellman-Ford (Es.)



6.58

Algoritmi di instradamento

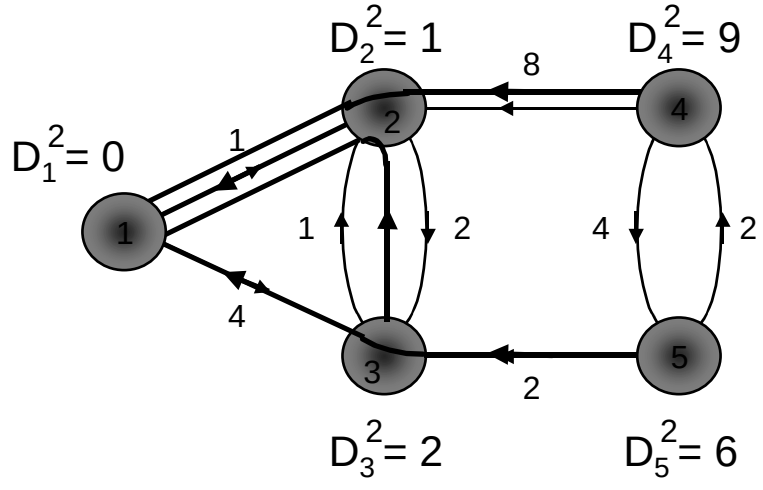
Bellman-Ford (Es.)



6.59

Algoritmi di instradamento

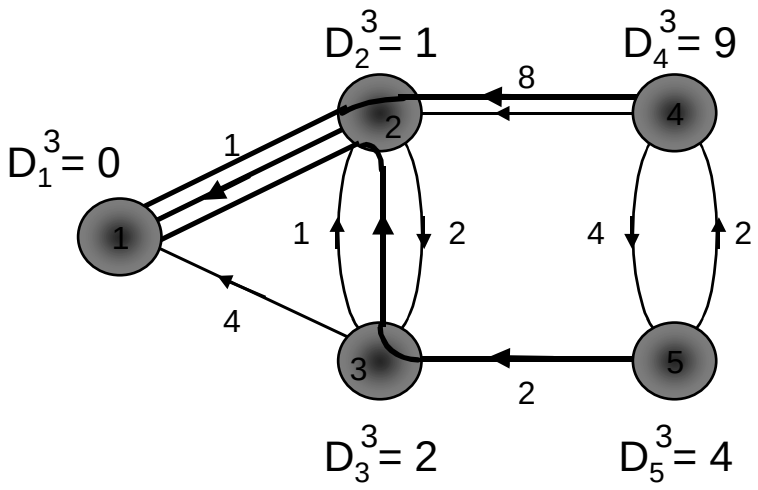
Bellman-Ford (Es.)



6.60

Algoritmi di instradamento

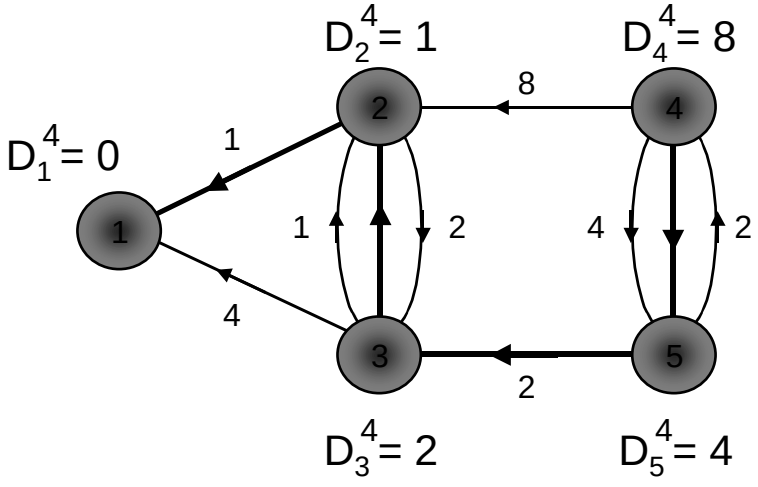
Bellman-Ford (Es.)



6.61

Algoritmi di instradamento

Bellman-Ford (Es.)



6.62

Algoritmi di instradamento

Shortest path Spanning Tree

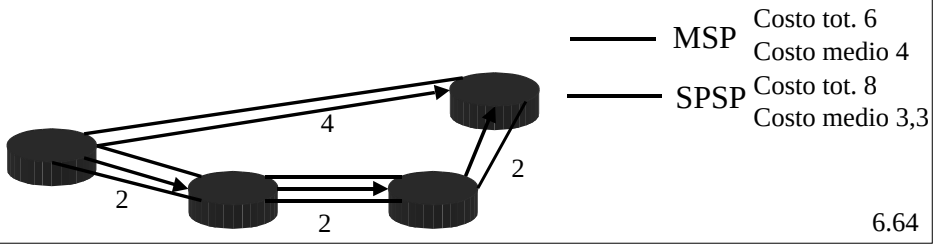
- L'applicazione equazione di Bellman seleziona un solo arco uscente da ogni nodo i (eccetto la dest. 1), cioè quello per cui la somma $d_{ij} + D_j^h$ è minima.
- Quindi in sostanza gli archi scelti dall'algoritmo ed i nodi formano uno *spanning tree* perché:
  - Comprendono tutti i nodi per definizione
  - non possono formare cicli (essendo le lunghezze positive)
- Tale spanning tree viene chiamato ***Shortest Path Spanning Tree*** (SPST) ed il nodo destinazione è chiamato ***root*** (radice).

6.63

Algoritmi di instradamento

Shortest path Spanning Tree

- Un grafo non orientato può essere rappresentato come un grafo orientato a cui ad ogni arco non orientato corrispondono due archi, uno per direzione, con eguale peso.
- In generale però il MSP e il SPSP sono diversi:
  - Il MSP minimizza il costo di un broadcasting;
  - Il SPSP invece minimizza il costo delle comunicazioni fra un qualunque nodo e la *root*.



6.64

Instradamento

Distance vector

- L'algoritmo di Bellman Ford può essere realizzato in modo distribuito ed in questo caso viene chiamato ***Distance Vector Routing***.
- Ogni nodo (router) conosce l'identità di tutti nodi della rete e i nodi a lui direttamente connessi (vicini).
- Ogni nodo mantiene un *Distance Vector*, ossia una lista di coppie (destinazione, costo) per tutte le possibili destinazioni.
- Il costo è la somma stimata dei costi sui singoli link sul percorso "più corto" (*shortest path*) verso quella destinazione.
- Ogni nodo inizializza i costi relativi a destinazioni "lontane" ad una valore alto, convenzionalmente indicato infinito.

6.65

Instradamento

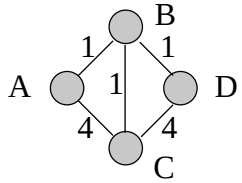
Distance vector

- Periodicamente ogni nodo invia ai propri vicini il proprio DV.
- Quando un *router* A riceve un DV da B (suo vicino), verifica quali sarebbero i costi per le varie destinazioni usando come transito B; per le destinazioni in cui tali costi risultano minori di quelli attuali, sostituisce il costo vecchio con quello calcolato e lo stesso fa con il next-hop nella RT.

6.66

Instradamento

Distance vector



Situazione Iniziale

	A	B	C	D
A	0	1	4	∞
B	1	0	1	1
C	4	1	0	4
D	∞	1	4	0

1	Costo per raggiungere B da A
+	
1 0 1 1	Costi da B agli altri
=	
2 1 2 2	Costi passando per B
0 1 4 ∞	Costi attuali in A
min	
0 1 2 2	Nuovo DV
- - B B	*Next hop*

6.67

Instradamento

Distance vector

- Questo procedimento corrisponde a realizzare in modo distribuito e asincrono l'algoritmo di Bellman-Ford, perché ogni nodo *i* esegue l'iterazione

$$D_i \leftarrow \min_{j \in N(i)} \{d_{ij} + D_j\}$$

  (dove $N(i)$ è l'insieme dei nodi adiacenti ad i), usando le stime D_j più recenti ricevute dai vicini e trasmettendo D_i ai propri vicini.
- Si dimostra che non è necessaria una inizializzazione con particolari valori di D_j.

6.68

Instradamento

Distance vector

Sia A il numero di archi e N quello dei nodi

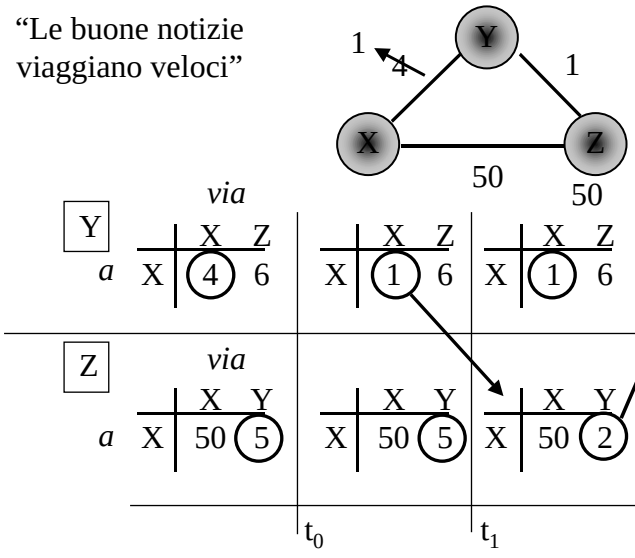
- Nel caso peggiore, l'algoritmo di Bellman-Ford centralizzato compie N-1 iterazioni, ciascuna su N-1 nodi, con al più N-1 alternative per nodo, il che porterebbe a complessità **$O(N^3)$**.
- Si può mostrare che la complessità è **$O(mA)$**, con m numero di iterazioni per la convergenza. Questo porta una complessità generalmente compresa fra **$O(N^2)$** e **$O(N^3)$.**
- Nel caso distribuito, se le iterazioni fossero eseguite in modo sincrono (simultaneamente ad ogni nodo), scambiando ad ogni iterazione i risultati con i vicini, partendo dalle condizioni iniziali $D_i^0 = \infty$ per tutti gli $i \neq 1$ e $D_1^0 = 0$, l'algoritmo convergerebbe in al più N-1 passi.

6.69

Instradamento

Distance vector

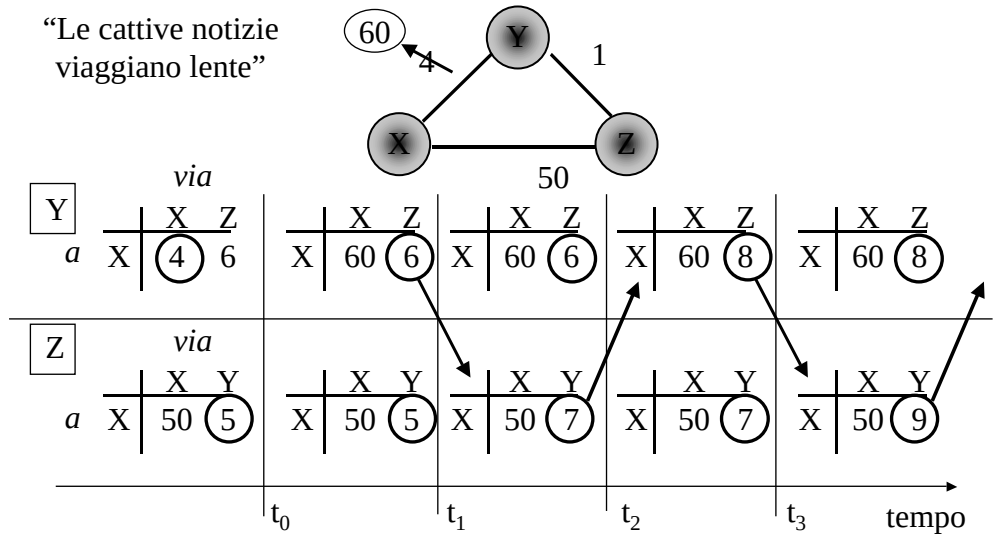
"Le buone notizie viaggiano veloci"



6.70

Instradamento

Distance vector

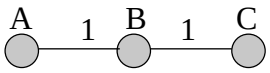


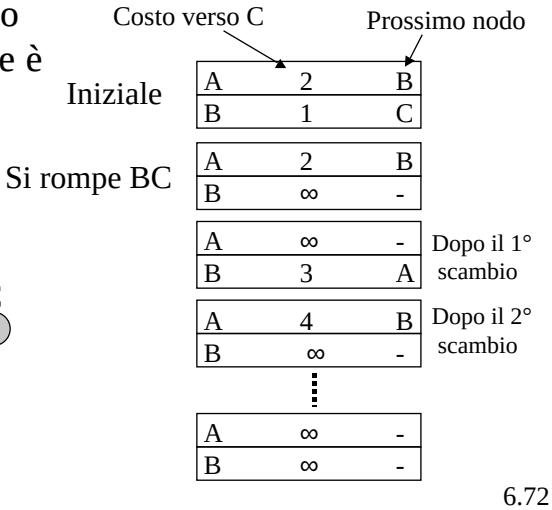
6.71

Instradamento

Distance vector

- Questo tipo di algoritmo ha un problema legato all'aggiornamento che è chiamato ***Count-to-infinity*** :





6.72

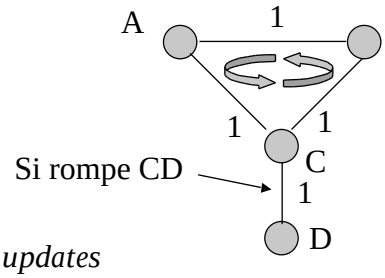
Instradamento
Distance vector

- **Ci sono diverse possibili soluzioni al count to infinity**
  - ***Path vector***
    - » oltre al costo si trasmette il percorso (*path-vector*), in questo modo i nodi possono capire quando non esiste più un percorso valido verso una certa destinazione. (BGP)
  - *Split horizon*
    - » Non viene passato il costo per una certa destinazione ad un vicino se questi è il *next hop* per quella destinazione.
    - » Una versione più complessa detta s*plit horizon with poisonous reverse*, invece di non passare costi passa un costo infinito, questo a volte accelera la convergenza. (RIP)
    - » Nel caso precedente, A non invia a B un costo (o lo invia infinito) verso C. Il ciclo quindi non si crea.

6.73

Instradamento
Distance vector

» Se ho ,più nodi coinvolti nella rottura direttamente non si riesce a bloccare il conto ad infinito.



B e A hanno sempre mandato a C un costo infinto verso D, ma non l'uno all'altro; questo innesca un ciclo che coinvolge A, B e C.

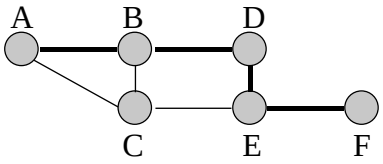
– *Triggered updates*
  - » Normalmente, per evitare un numero eccessivo di aggiornamenti delle tabelle e di traffico di controllo, si limita il ritardo minimo fra due aggiornamenti consecutivi (per es. 30 sec.). Nel caso di collegamento caduto, gli aggiornamenti sono fatti immediatamente (riduce il tempo di convergenza). (RIP)

6.74

Instradamento

Distance vector

– *Source tracing*
  » Insieme al costo i nodi si scambiano anche il nodo da attraversare immediatamente prima della destinazione. Con questa informazione aggiuntiva è possibile ricavare direttamente dalla tabella il percorso complessivo e quindi, mantenendo la RT più piccola si ottiene lo stesso del *path vector*.



RT di A

Dest.	Succ.	Ultimo
A	-	
B	B	A
C	C	A
D	B	B
E	B	D
F	B	E

6.75

Instradamento

Link-state

- La filosofia del ***Link State (LS) routing*** è quella di distribuire a tutti i nodi della rete l'intera sua topologia ed i costi di ogni *link* che la compone.
- Con questa informazione ogni *router* è in grado ci calcolarsi i propri percorsi ottimi verso ogni destinazione.
- Se tutti vedono gli stessi costi e tutti usano lo stesso algoritmo, i percorsi saranno liberi da cicli.
- Quindi sono due gli aspetti caratterizzanti questo metodo
  – Il modo in cui la topologia della rete viene diffusa fra i nodi.
  – Il modo in cui ogni nodo calcola i percorsi ottimi.

6.76

Instradamento

Link-state - Dijkstra (Cont.)

- Nel caso LS ogni nodo applica l'algoritmo di **Dijkstra**.
- A differenza dell'algoritmo di Bellman-Ford che itera sul numero di archi attraversati da un percorso, l'algoritmo di Dijkstra itera sulla lunghezza del percorso.
- Nel caso peggiore la sua complessità è $\mathbf{O(N^2)}$, in media si colloca intorno a $\mathbf{O(A \log A)}$ con A numero degli archi .

6.77

Instradamento

Link-state - Dijkstra

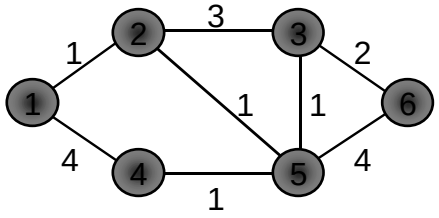
- Sia P un insieme di nodi e D_i la distanza minima "stimata" dal nodo 1.
- Fissando inizialmente
  $P = \{1\}$, $D_1 = 0$, $D_j = d_{j1}$ per tutti i $j \neq 1$
- I passi dell'algoritmo di Dijkstra sono
  1. Trova $i \notin P$ tale che
     $D_i = \min_{j \notin P} \{D_j\}$
     e poni $P \leftarrow P \cup \{i\}$.
     Se P contiene tutti i nodi l'algoritmo è completato.
  2. Per tutti i $j \notin P$, poni $D_j \leftarrow \min\{D_j, d_{ji} + D_i\}$
     e torna al passo 1

6.78

Instradamento

Link-state - Dijkstra

Bellman-Ford



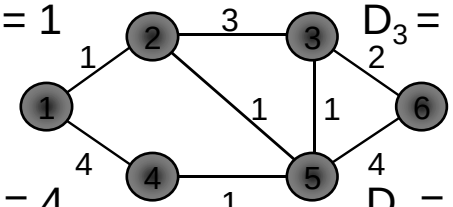
$d_{ij} = d_{ji}$

(i, j)

P = {1, 2}

Dijkstra

$D_2 = 1$ $D_3 = \infty$



$D_6 = \infty$

$D_4 = 4$ $D_5 = \infty$

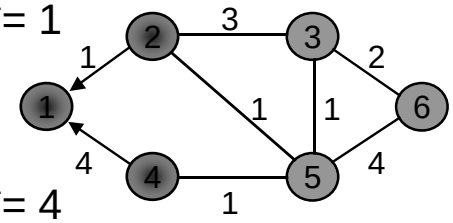
6.79

Instradamento

Link-state - Dijkstra

Bellman-Ford

$D_2^1 = 1$

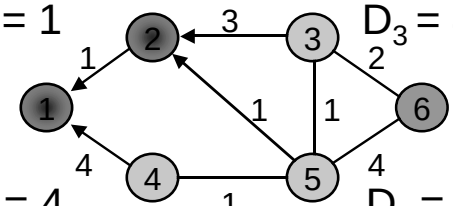


$D_4^1 = 4$

Dijkstra

$D_2 = 1$ $D_3 = 4$

P = {1, 2}



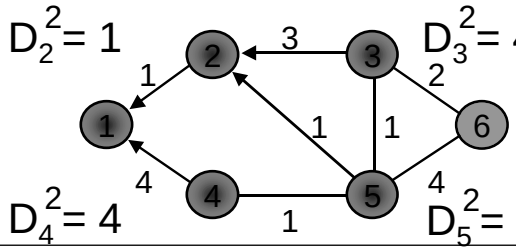
$D_6 = \infty$

$D_4 = 4$ $D_5 = 2$

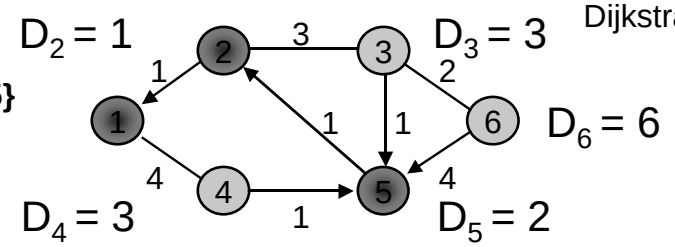
6.80

Instradamento

Link-state - Dijkstra



Bellman-Ford



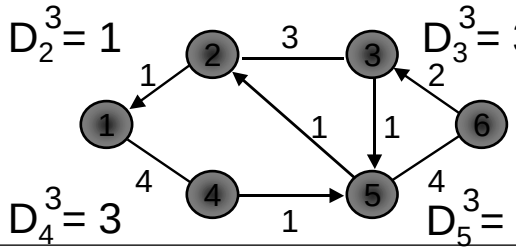
Dijkstra

P = {1, 2, 5}

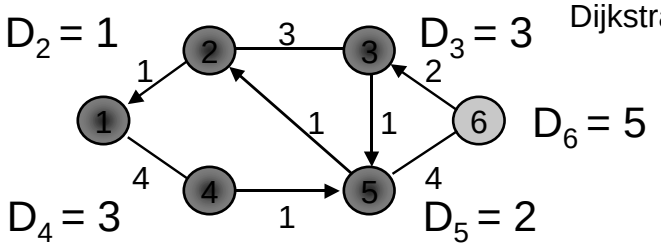
6.81

Instradamento

Link-state - Dijkstra



Bellman-Ford



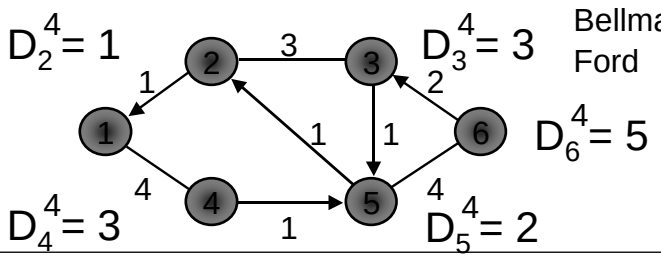
Dijkstra

P = {1, 2, 3, 4, 5}

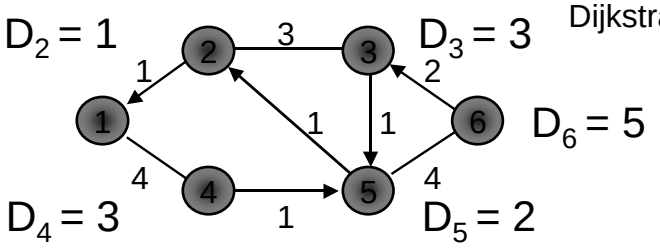
6.82

Instradamento

Link-state - Dijkstra



P = {1, 2, 3, 4, 5, 6}

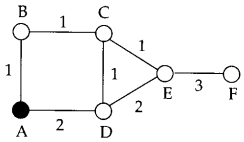


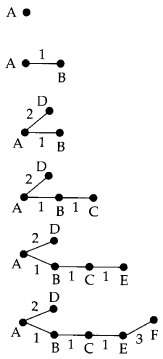
6.83

Instradamento

Link-state - Dijkstra

C(B,1) significa che C è raggiunto tramite B, con costo 1





Definitivo (P)	Temporaneo	Commenti
A	B(A,1),D(A,2)	La radice e i suoi vicini
A, B(A,1)	D(A,2), C(B,2)	Aggiunto C
A, B(A,1), D(A,2)	E(D,4), C(B,2)	Scartato C(D,3)
A, B(A,1), D(A,2), C(B,2)	E(C,3)	Scartato D(D,4)
A, B(A,1), D(A,2), C(B,2), E(C,3)	F(E,6)	
A, B(A,1), D(A,2), C(B,2), E(C,3), F(E,6)	Vuoto	Stop

6.84

Instradamento
Link-state

- Disseminazione della topologia
  - Ogni nodo crea un insieme di ***Link-State-Packet*** (LSP) che descrivono le sue linee in uscita.
  - Ogni LSP contiene l'indirizzo del nodo, quello dei nodi vicini, ed il costo delle linee verso i nodi vicini.
  - Ogni LSP viene distribuito a tutti i nodi tramite un ***controlled flooding***
    - » Ogni nodo che riceve un LSP lo memorizza in un database e invia una copia su tutte le proprie linee in uscita, tranne quella da cui l'ha ricevuto. Si può dimostrare che nessun LSP passa due volte per lo stesso *link* lungo la stessa direzione, quindi un LSP viene distribuito in al più 2L invii, dove L è il numero dei *link*.

6.85

Instradamento
Link-state

- Numero di sequenza
  - Per poter decidere se un LSP ricevuto è significativo (contiene una informazione più recente di quella attualmente nel nodo) ogni LSP deve contenere un numero di sequenza progressivo.
  - Il numero di sequenza ha valore locale per ogni tipo di LSP (identificato da coppia ordinata di nodi collegati da una linea)
  - Ogni volta che un nodo riceve un LSP più vecchio di quello in memoria, lo elimina senza propagarlo.

6.86

Instradamento
Link-state

- Le sequenze realmente utilizzabili sono di lunghezza finita e quindi soggetta ad “avvolgersi” (*wrapping*) bloccando l’aggiornamento.
- *Wrapped sequence number*
  - Per evitare il problema si può prendere una sequenza molto grande (32 bit => 4.295.967295) e decidere che quando due numeri distano troppo, il più piccolo sia anche il più giovane. Per esempio supponendo che N sia lunghezza della sequenza, allora ***a*** è più vecchio di ***b*** se
    - » $a < b$ e $|b - a| < N/2$
      oppure se
    - » $a > b$ e $|b - a| \geq N/2$

6.87

Instradamento
Link-state

- La presenza di un numero di sequenza pone il problema dell’inizializzazione della sequenza quando un nodo si (ri)attiva. Due sono i meccanismi adottati
  - Invecchiamento (*Aging*)
  - *Lollipop sequence space*

6.88

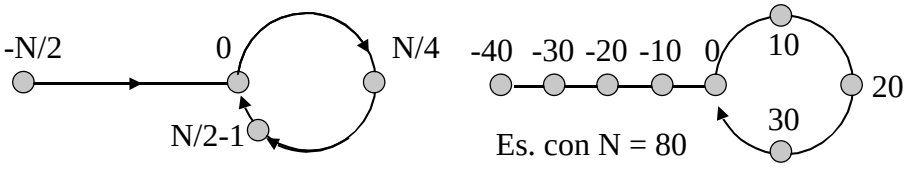
Instradamento
Link-state

- Invecchiamento
  - Prevede l'inserimento di un campo di di anzianità nel LSP, che viene inizializzato ad un valore (MAX_AGE) dal creatore del pacchetto.
  - Ogni nodo copia in un contatore C_AGE il valore MAX_AGE e lo decrementa periodicamente.
  - Quando in un *router* C_AGE raggiunge zero, la corrispettiva informazione viene eliminata dal DB e viene generato un LSP con anzianità zero, per forzare la stessa operazione sugli altri *router*.
  - E' difficile fissare un valore ottimale per MAX_AGE (troppo corto: scade prima di essersi propagato; troppo lungo: un nodo che riparte deve attendere a lungo perché i nuovi pacchetti diventino significativi)

6.89

Instradamento
Link-state

- *Lollipop sequence space*
  - Si tratta di una sequenza che si "avvolge" in modo particolare (al *boot* la macchina riparte da -N / 2)



  - In questo caso **a** è più vecchio di **b** se:
    - » **a** < ***0*** e **a** < **b** o
    - » **a** > ***0,*** **a** < **b** e | **b** - **a** | < N / 4, o
    - » **a** > ***0,*** **b** > ***0,*** **a** > **b** e | **b** - **a** | > N / 4

6.90

Instradamento
Link-state

- *Lollipop sequence space*
  - Quando un nodo riceve un LSP con un numero di sequenza più vecchio di quello nel DB, lo comunica a chi gli ha inviato il pacchetto fornendo anche l'ultimo valore di sequenza che aveva memorizzato.
  - Un nodo che riparte genera sempre un numero di sequenza più vecchio degli altri e quindi i nodi vicini gli inviano l'ultimo valore da lui usato da cui può ripartire aggiungendogli 1.
  - In pratica i *router* vicini si comportano come una sorta di memoria distribuita.

6.91

Instradamento
Link-state

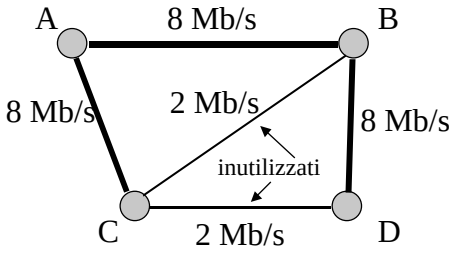
- Ci sono alcune altre considerazioni di criticità da fare
  - Se la rete di partiziona per la caduta di una o più linee e le singole parti evolvono indipendentemente, quando si ricollegano possono crearsi problemi (*loop*). (Soluzione: scambio fra nodi vicini di parti di DB)
  - Se invece di una linea, si rompe un nodo, non c'è nessuno che propaga l'informazione. (Soluzione: pacchetti di Hello e anzianità massima degli LSP nel DB)
  - Bisogna proteggere gli LSP da "corruzioni" casuali o volute.

6.92

Instradamento

Metriche e dinamica

- Metriche statiche
  - In genere si tratta di valori inversamente proporzionali alla capacità del *link*. La staticità fa si che le linee a minor velocità tendano ad essere sotto-utilizzate.



Pesi
8 Mb/s = 1
2 Mb/s = 4

6.93

Instradamento

Metriche e dinamica

- Dinamiche
  - Le metriche dipendenti dal traffico sono sicuramente più efficaci, ma comportano alcuni problemi.
  - Consideriamo la sperimentazione avvenuta su ARPAnet, dove in origine si era usata una metrica proporzionale alla lunghezza delle code di uscita dei *router*, per fare alcune osservazioni:
    - » La lunghezza (metrica) derivava da una media su un orizzonte (10 s). La durata dell'orizzonte è critica:
      - Corta: troppi transienti;
      - Lunga: rete converge lentamente;
      - La durata ottima non è omogenea sulla rete: dipende dalle capacità dei *link*

6.94

Instradamento

Metriche e dinamica

» La dinamica del costo non deve essere alta: altrimenti alcuni percorsi vengono completamente ignorati

» La lunghezza della coda è usata come "predittore" della situazione futura del *link*: ma linee con code lunghe non verranno scelte nel futuro e quindi si "scaricheranno" (specialmente quelle ad alta capacità) e viceversa.

» La mancanza di restrizioni fra valori successivi dei costi può generare oscillazioni significative.

» Il ricalcolo quasi-sincrono delle tabelle tende a raccogliere traffico su alcune linee.

6.95

Instradamento

Metriche e dinamica

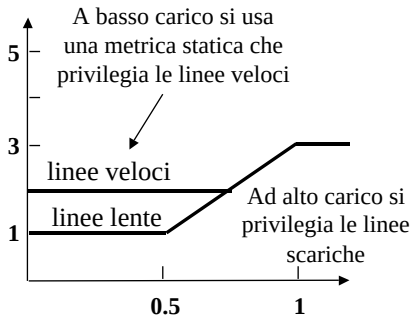
- La soluzione scelta per ARPAnet è stata:
  - Metrica mista capacità-coda (statica dinamica) dove a carico basso prevale la capacità, carico alto la coda.
  - Costi con una dinamica ridotta: valori da 1 a 3.
  - Massima variazione permessa fra due successivi ricalcoli: 1/2.

6.96

Instradamento

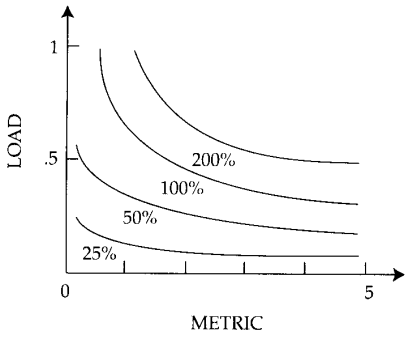
Metriche e dinamica

Mappa della metrica



Esprime il valore del costo in funzione del carico sulla linea

Risposta della rete

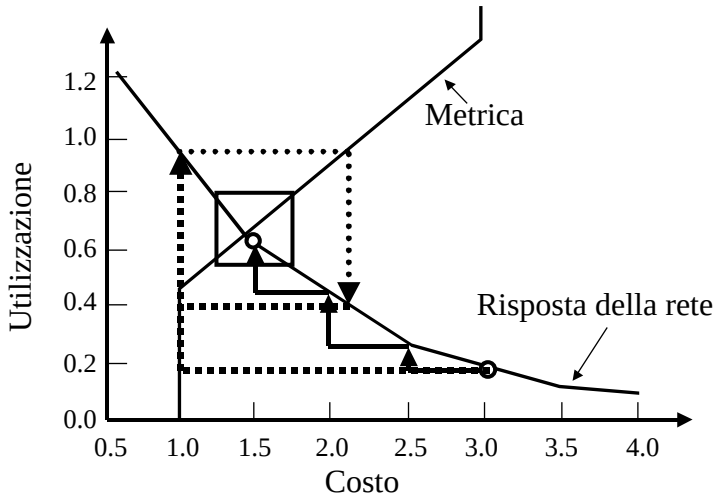


Carico medio in funzione del valore del costo su una singola linea (una curva per ogni valore del carico complessivo offerto alla rete)

6.97

Instradamento

Metriche e dinamica



6.98

Instradamento

DVR e LSR

- Il confronto DV *Routing* (DVR) e LS *Routing* (LSR) è complesso, proviamo a distinguere diversi aspetti:
  - Velocità di convergenza
    - » In genere si tende a supporre che gli LSR convergano più rapidamente dei DVR, in pratica la velocità di convergenza dipende molto dalla topologia della rete e dalle caratteristiche del traffico.
  - Volume di messaggi di controllo
    - » LS : con N nodi e A archi richiedono lo scambio di O(NA) messaggi per ciascun nodo.
    - » DV: deve solo scambiare i messaggi con i propri vicini.
  - Robustezza; se un nodo comincia a funzionare male:
    - » LS: il nodo propaga un costo sbagliato, ogni nodo si calcola separatamente la propria tabella
    - » DV: il nodo propaga un percorso sbagliato, ogni tabella viene calcolata facendo uso delle altre

6.99

Instradamento

DVR e LSR

- I DVR sono non escludono la presenza di cicli a priori, ma con le opportune modifiche gli possono evitare efficacemente.
- Gli LSR, per contro, sono più complessi, devono fare uno sforzo significativo per mantenere i DB congruenti (generando anche un traffico di controllo più elevato) ed hanno *Distance Table* più grandi.
- Gli LSR possono usare più metriche diverse contemporaneamente.
- Gli LSR si prestano ad essere estesi per supportare con le stesse tabelle rounting *unicast* e *multicast*

6.100

Instradamento
Gerarchia

Ci sono due ragioni importanti per le quali nelle reti di una certa dimensione si tende ad usare meccanismi di instradamento gerarchici:

- **La scalabilità**
  - Per un numero di nodi elevato (WAN), indipendentemente dal tipo di algoritmo, la complessità dell'instradamento e la dimensione delle RT diventano comunque eccessive (oltre al traffico di segnalazione).
  - Per esempio, nel caso LS, con tanti archi quanti nodi, si ha una complessità di circa O(NlogN) ed una RT con con dimensione O(N), quindi

# nodi	RT	Calcoli
1000	1000	O(3000)
1.000.000	1.000.000	O(6.000.000)

- **L'autonomia amministrativa**

6.101

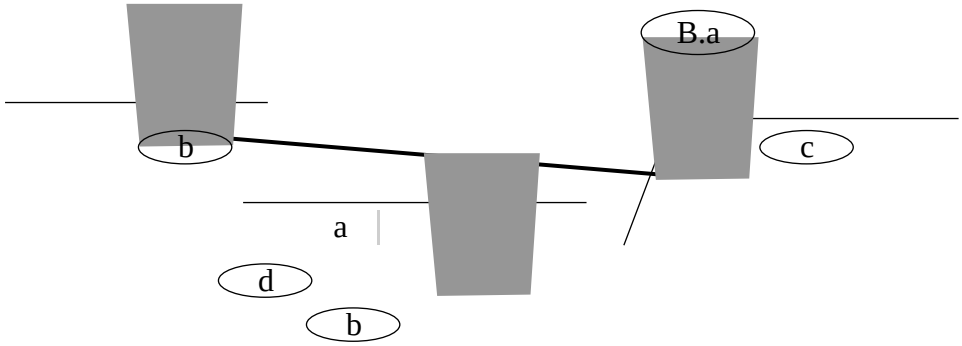
Instradamento
Gerarchia

- Internet distingue tre livelli gerarchici principali:
  - Sottoreti o singoli domini di *broadcast*
    - » All'interno dei quali l'instradamento fa uso dell'ARP.
  - *Autonomous System* (AS)
    In cui i protocolli di instradamento prendono il nome di ***Interior Gatway Protocol*** (IGP) e sono: RIP, IGRP e OSPF.
  - *Backbone*
    In cui i protocolli di instradamento prendono il nome di ***Exterior Gateway Protocol*** (EGP) e sono: EGP e BGP.
- Ulteriori livelli possono essere inseriti tramite alcuni protocolli (OSPF), o sfruttando la "*route aggregation*".

6.102

Instradamento

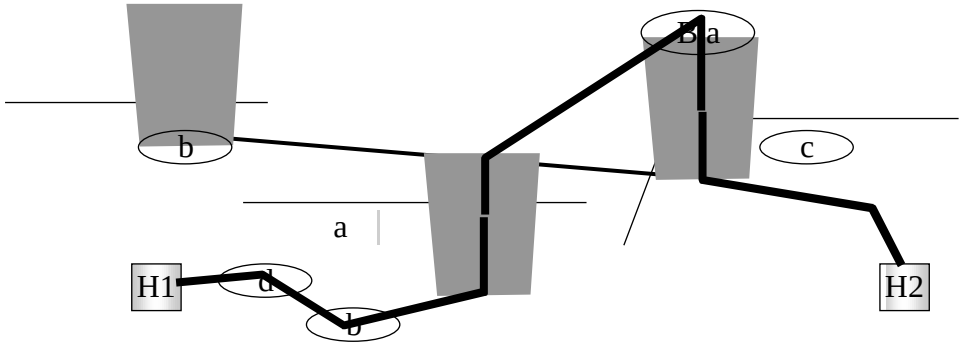
Gerarchia - *Autonomous Systems*



6.103

Instradamento

Gerarchia - *Autonomous Systems*



6.105

Instradamento

Gerarchia

- Il partizionamento è realizzato grazie alla gerarchizzazione degli indirizzi.
- L'obiettivo è avere relativamente "pochi" nodi per ogni livello (e area).
- Potenzialmente ogni livello può usare algoritmi diversi
- La gerarchia non è stretta, ossia il collegamento un area di un livello e il livello superiore può avvenire tramite più nodi (*Border Gatway*, BG)
- Ci sono dei router che partecipano all'instradamento di livelli differenti.
- Alcuni indirizzi possono non essere omogenei con lo spazio di indirizzamento dell'area / livello (questo diminuisce l'efficacia della gerarchia).

6.106

Instradamento

Gerarchia

- I diversi livelli non possono nascondersi reciprocamente tutte le informazioni:
  - Ad es., per poter calcolare l'instradamento più opportuno, un nodo del livello 3 deve conoscere i costi per raggiungere i nodi del livello superiore.
  - Allo stesso modo, un nodo di livello 4 deve conoscere i costi verso i nodi del livello 3.
  - Queste conoscenze sono fornite tramite LSP particolari (detti *external records* e *summary records*) che contengono solo le destinazioni ed i costi per raggiungerle (non la topologia). In pratica le reti dei livelli superiori/inferiori vengono rappresentate come se i loro nodi fossero direttamente collegati ai BG.

6.107

Instradamento

Routing Information Protocol (RIP)

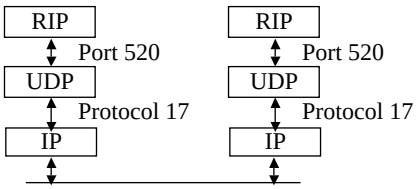
- E' un IGP originariamente progettato dalla Xerox per la propria rete, introdotto dall'Università di Berkley nella propria implementazione di TCP/IP (BSD)
- E' stato standardizzato con RFC 1058 nel 1988, la versione 2 è descritta nel RFC 1723.
- E' un DVR ed usa una metrica statica: il costo di un percorso è il numero di hop (ossia di linee) di cui è composto (ossia ogni linea a costo 1).
- Utilizza lo *split horizon with poisonous reverse*, e *i triggered update*.
- Aggiorna la RT (tramite RIP *response message* o RIP *advertisement*) ogni 30 s. e elimina ogni vettore non aggiornato per 180 s consecutivi (considerando la corrispondente linea non più disponibile).

6.108

Instradamento

Routing Informatio Protocol (RIP)

- Ha come valore massimo del costo 15, 16 corrisponde ad infinito. Quindi non permette reti con percorsi con più di 15 *router* attraversati.
- La limitazione di cui sopra è legata al fatto che per reti più grandi è troppo lento a convergere (non alla dimensione del campo costo).
- Ne esistono due versioni, la seconda (RIPv2) consente l'uso del CIDR, ossia la "*route aggregation*" , e la "*default route*".
- Lo scambio di informazioni avviene attraverso un protocollo di livello 4 (UDP)



6.109

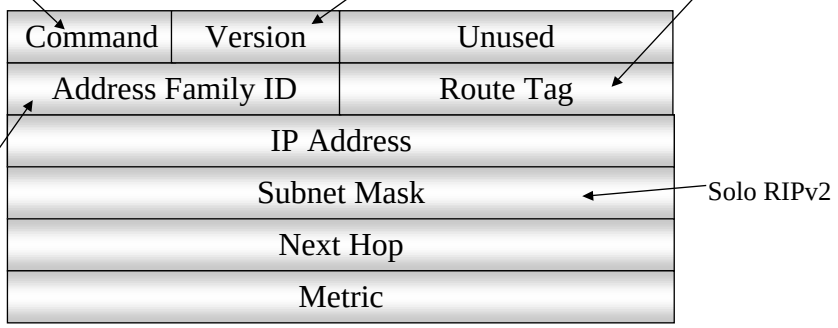
Instradamento

Routing Information Protocol (RIP)

- Sollecito per un DV
- Messaggio di *update* (anche su sollecito)

1 o 2

Usato solo in RIPv2 per distinguere fra percorsi interni all'AS ed esterni.



- Usato per autenticazione
- Nel RIPv2 viene posto a FFFF e viene aggiunto successivamente un campo *password*

6.110

Instradamento

Interior Gateway Routing Protocol (IGRP)

- E' nuovamente un DVR, ma di tipo proprietario; infatti è stato sviluppato dalla CISCO verso la metà degli anni '80 ed è disponibile solo sui suoi prodotti.
- Usa una metrica dinamica e sofisticata (considera ritardo, banda, affidabilità, lunghezza del pacchetto ed il carico) in cui il costo della linea viene composto tramite una somma pesato, i cui pesi sono impostabili dal gestore.
- Permette la suddivisione del carico su più linee (multipercorso).
- Usa un meccanismo sofisticato per accelerare la convergenza ed evitare i cicli.

6.111

Instradamento

Open Shortest Path First (OSPF)

- Nasce nel 1990 con l'RFC 1247 per sostituire il RIP
- E' un protocollo di tipo ***Link State***
- Quindi ogni nodo costruisce al proprio interno la topologia di tutta la rete e invia in *flooding* i LSP, contenenti i costi dei *link* ad esso connessi.

6.112

Instradamento

Open Shortest Path First (OSPF)

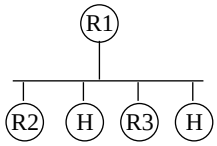
- E' stato realizzato per rispondere a diverse esigenze:
  - ***Open*: ossia aperto e non proprietario.**
  - **Sicurezza**: gli scambi fra *router* vengono autenticati, per proteggere gli aggiornamenti.
  - **Multi-metrica**: permette l'uso di più metriche anche dinamiche e instradamenti differenziati a seconda del campo TOS.
  - **Multi-percorso**: permette il bilanciamento dei flussi su percorsi a costo uguale.
  - **Gerarchico**: supporta una gerarchia interna.
  - **Multicast**: supporta il multicast (M-OSPF)

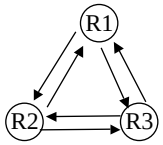
6.113

Instradamento

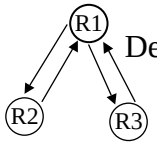
Open Shortest Path First (OSPF)

- Supporta tre tipi di connessione e reti
  - Punto - punto fra router
  - Reti multiaccesso con *broadcast* (LAN)
  - Reti multiaccesso senza *broadcast* (WAN a pacchetto)
- Nel caso di LAN a cui sono connessi più router identifica un *router* di riferimento (*Designated Router*, DR) per ridurre il traffico di LSP in broadcast sulla LAN.





Senza DR
(n x (n-1) broadcast)



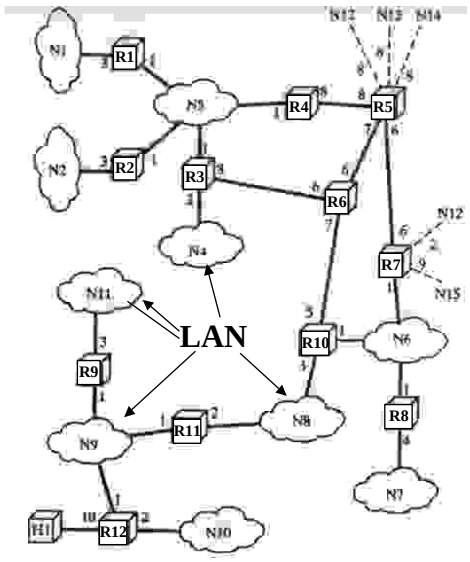
Designated Router
(Backup DR)

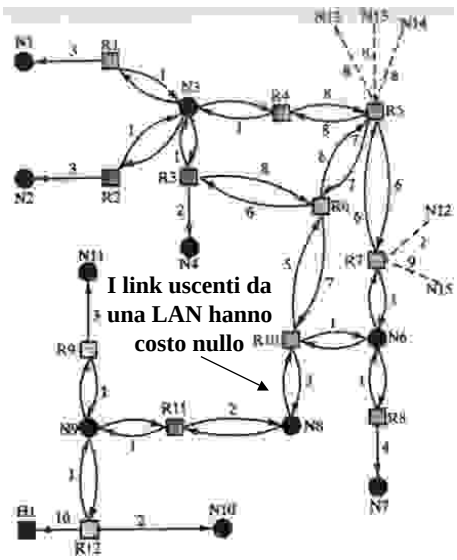
Con DR
(minor # di broadcast)

6.114

Instradamento

Open Shortest Path First (OSPF)





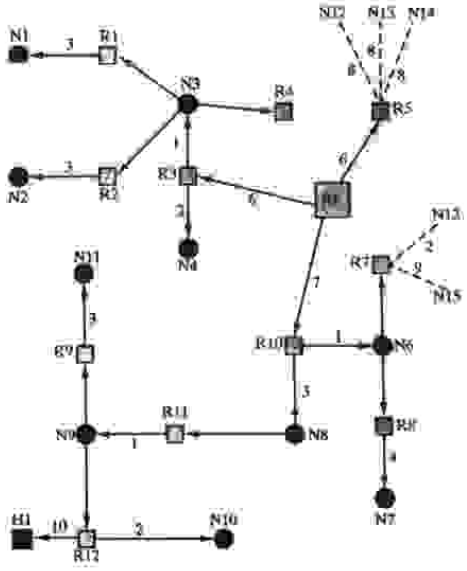
6.115

Instradamento

Open Shortest Path First (OSPF)

TABLE 16.5 Routing Table for RT6

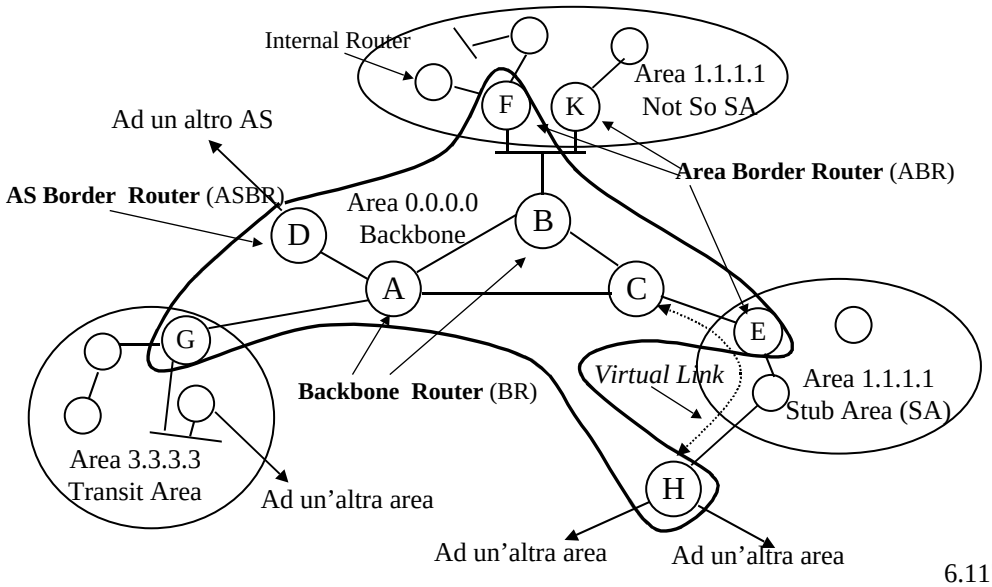
Destination	Next hop	Distance
N1	RT3	10
N2	RT3	10
N3	RT3	7
N4	RT3	8
N6	RT10	8
N7	RT10	12
N8	RT10	10
N9	RT10	11
N10	RT10	13
N11	RT10	14
H1	RT10	21
RT5	RT5	6
RT7	RT10	8
N12	RT10	10
N13	RT5	14
N14	RT5	14
N15	RT10	17



6.116

Instradamento

Open Shortest Path First (OSPF)



6.117

Instradamento

Open Shortest Path First (OSPF)

- Le *Stub Area* non propagano informazioni interne o esterne ed accedono al *backbone* tramite un *router* di *default*.
- L'instradamento fra due aree viene realizzato in tre parti:
  - Il percorso nell'area sorgente fra la sorgente stessa ed un *Area Border Router.*
  - Il percorso fra i due ABR delle due aree tramite il *backbone*
  - Il percorso nell'area destinazione fra l'ABR che riceve il pacchetto dal *backbone* e la destinazione.
- In pratica si forza un instradamento a stella in cui il backbone rappresenta il centro stella.

6.118

Instradamento

EGP- “EGP”

- Al più vecchio dei protocolli EGP è stato assegnato lo stesso nome che distingue la categoria: EGP.
- E’ un protocollo di stile DV che però non propaga costi ma solo informazioni di raggiungibilità.
- Non è in grado di evitare cicli e quindi non può essere usato in topologie magliate ma solo ad albero.
- La sua struttura di riferimento è composta da “*Core Router*” (CR) collegati fra loro ad albero.
- Ogni AS può essere collegato ad un unico CR e quindi ogni CR fa da centro stella per un gruppo di AS

6.119

Instradamento

EGP - Border Gateway Protocol (BGP)

- E’ il protocollo EGP relativamente recente, definito dal RFC 1654.
- La versione in uso attualmente è la 4 (BGP4).
- Permette la cooperazione fra *router* di AS diversi (chiamati *gateway*) per la realizzazione dell’instradamento fra AS.
- Per lo scambio di informazioni fra i nodi usa il TCP (porta 179).
- Non propaga vere e proprie metriche, ma lascia che la scelta dei percorsi venga determinata tramite “politiche” impostate dai singoli gestori.

6.120

Instradamento

EGP - Border Gateway Protocol (BGP)

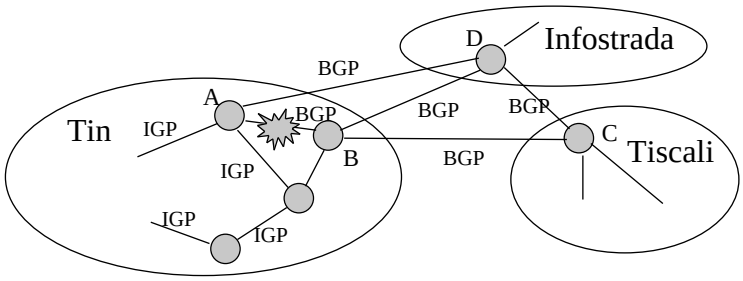
- Opera in tre passi:
  - Identificazione dei nodi adiacenti (*neighbor*)
  - Raggiungibilità dei nodi adiacenti
  - Raggiungibilità delle reti
- Utilizza un algoritmo DV, ed in particolare usa un *Path Vector*.
- Distingue tre tipi di reti
  - *Stub:* che hanno un'unica connessione con il *backbone* e non possono venir usate come transito
  - *Multiconnected*: che potenzialmente possono essere usate per transito (se lo permettono)
  - *Transit*: costruite per realizzare il transito.

6.121

Instradamento

EGP - Border Gateway Protocol (BGP)

- Anche se tutti i *router* all'interno di un AS sono fra loro cooperativi, non è detto che i nodi che interconnettono due AS si "fidino" l'uno dell'altro.
- Questo accade in quanto gli AS sono in genere controllati o posseduti da organizzazioni diverse e quindi l'instradamento deve dipendere anche dagli accordi fra i diversi AS (transito).



6.122

Instradamento

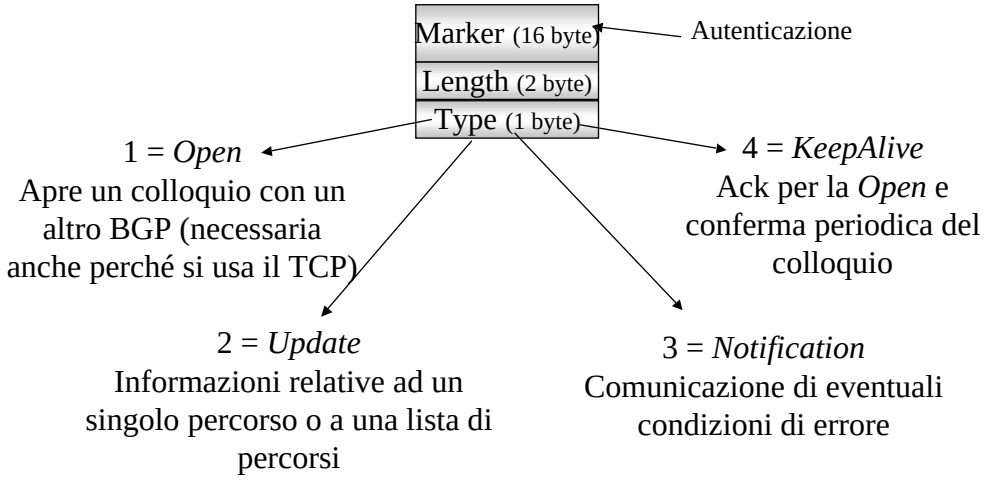
EGP - Border Gateway Protocol (BGP)

- In quanto protocollo di tipo EGP, il BGP differisce dai protocolli IGP in diversi aspetti:
  - Politica: fra AS nella scelta del percorso la "politica" domina, ossia le scelte dipendono da più fattori che coinvolgono considerazioni strategiche, economiche e di sicurezza più che tecniche. Queste considerazioni sono specifiche di ogni AS, quindi la scelta è principalmente sotto il controllo amministrativo.
  - Scala: la scalabilità è molto importante perché le "reti" coinvolte sono generalmente grandi. Invece se un AS cresce eccessivamente lo si può sempre dividere in due.
  - Prestazioni: le prestazioni tecniche contano relativamente poco in un EGP.

6.123

Instradamento

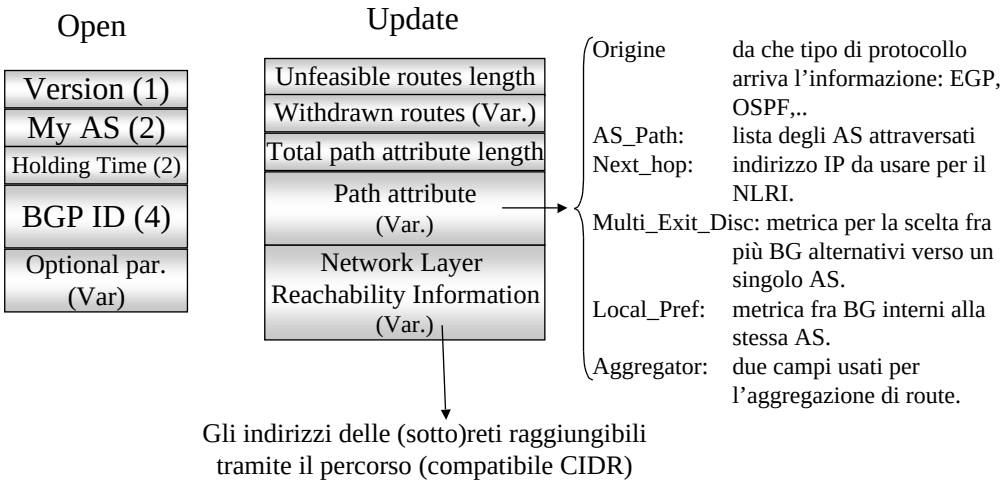
EGP - Border Gateway Protocol (BGP)



6.124

Instradamento

EGP - Border Gateway Protocol (BGP)



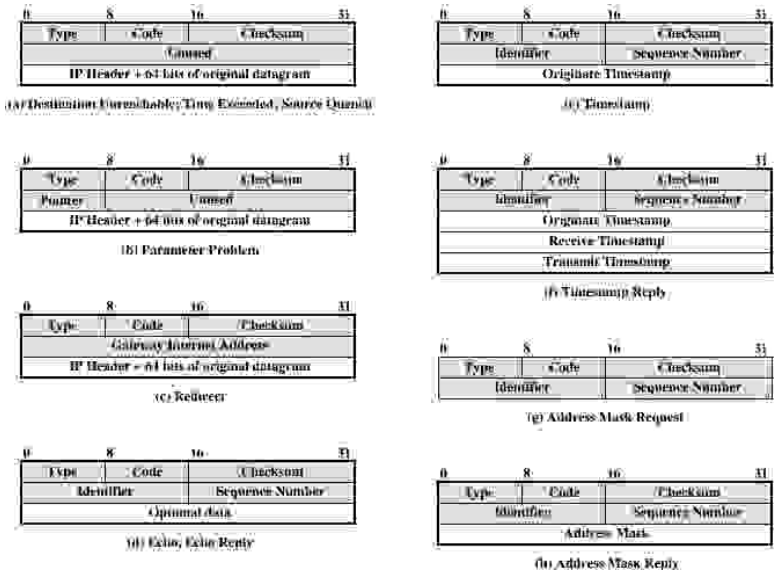
6.125

ICMP - Internet Control Message Protocol

- E' definito nel RFC 792
- Permette il trasferimento di messaggi di controllo tra router, tra router e *host* e tra *host*.
- Permette di fornire una retroazione in presenza di problemi
  - Per esempio, lo scadere del TTL
- I messaggi sono direttamente incapsulati in IP datagram
  - Quindi non è affidabile

6.126

ICMP


0 8 16 31
Type Code Checksum
Unused
IP Header + 64 bits of original datagram
(a) Destination Unreachable, Time Exceeded, Source Quench
0 8 16 31
Type Code Checksum
Identifier Sequence Number
Originate Timestamp
(c) Timestamp
0 8 16 31
Type Code Checksum
Pointer Unused
IP Header + 64 bits of original datagram
(b) Parameter Problem
0 8 16 31
Type Code Checksum
Identifier Sequence Number
Originate Timestamp
Receive Timestamp
Transmit Timestamp
(d) Timestamp Reply
0 8 16 31
Type Code Checksum
Gateway Internet Address
IP Header + 64 bits of original datagram
(e) Redirect
0 8 16 31
Type Code Checksum
Identifier Sequence Number
(g) Address Mask Request
0 8 16 31
Type Code Checksum
Identifier Sequence Number
Optional data
(a) Echo, Echo Reply
0 8 16 31
Type Code Checksum
Identifier Sequence Number
Address Mask
(h) Address Mask Reply

6.127