

Corso di Laurea in Ingegneria Informatica



**Corso di Reti di Calcolatori
(a.a. 2010/11)**

Roberto Canonico (roberto.canonico@unina.it)

Giorgio Ventre (giorgio.ventre@unina.it)

**Routing: introduzione
Routing Distance Vector: introduzione**

11 novembre 2010

**I lucidi presentati al corso sono uno strumento didattico
che NON sostituisce i testi indicati nel programma del corso**

Nota di copyright per le slide COMICS



Nota di Copyright

Questo insieme di trasparenze è stato ideato e realizzato dai ricercatori del Gruppo di Ricerca COMICS del Dipartimento di Informatica e Sistemistica dell'Università di Napoli Federico II. Esse possono essere impiegate liberamente per fini didattici esclusivamente senza fini di lucro, a meno di un esplicito consenso scritto degli Autori. Nell'uso dovranno essere esplicitamente riportati la fonte e gli Autori. Gli Autori non sono responsabili per eventuali imprecisioni contenute in tali trasparenze né per eventuali problemi, danni o malfunzionamenti derivanti dal loro uso o applicazione.

Autori:

Simon Pietro Romano, Antonio Pescapè, Stefano Avallone,
Marcello Esposito, Roberto Canonico, Giorgio Ventre

Nota: alcune delle slide di questa lezione sono direttamente prese dal materiale didattico preparato dagli autori del libro di testo Kurose e Ross

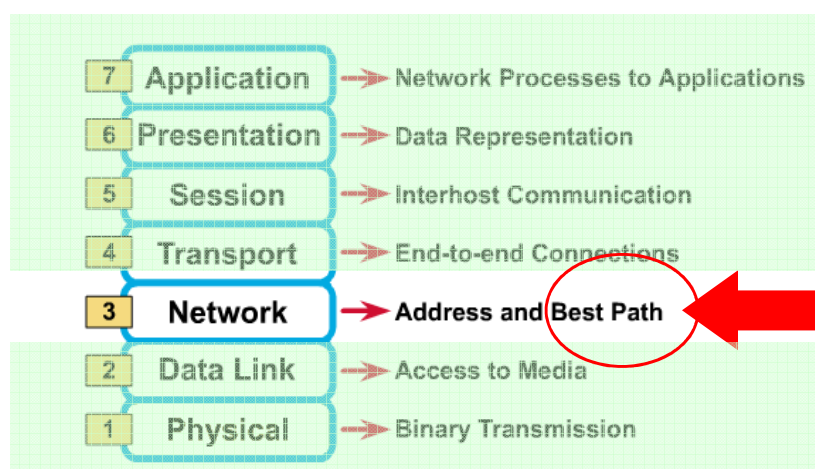
Il ruolo dei livelli OSI



Dobbiamo Pavimentare le strade		Livello Fisico Cablaggio Strutturato
Dobbiamo scegliere il tipo di strada (Autostrada, Provinciale, Urbana,...)		Livello Data Link
Dobbiamo scegliere le indicazioni della prossima rotonda		Livello Rete
Dobbiamo scegliere come trasportare		Livello Trasporto

3

Il livello rete nella pila OSI



4

Reti di calcolatori e grafi

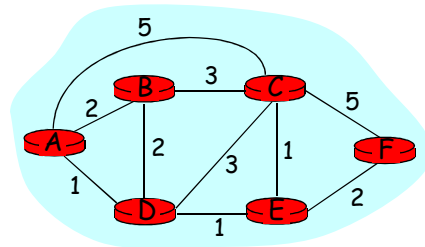


Rete modellata come grafo:

- **nodi** = router
- **archi** = link fisici

- **costo link:**

- ritardo,
- costo trasmissione,
- congestione,...



- Scelta del cammino:
 - cammino a costo minimo
 - altre possibilità (un cammino calcolato in base a specifici vincoli...)
- Gli algoritmi per la gestione di una rete sono basati sulla teoria dei grafi

5

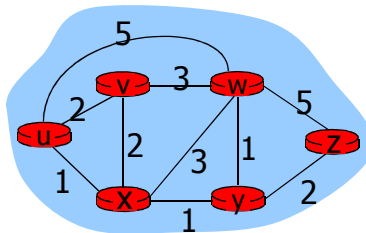
Parametri del processo decisionale



- **Bandwidth**
 - capacità di un link, tipicamente definita in bit per secondo (bps)
- **Delay**
 - il tempo necessario per spedire un pacchetto da una sorgente ad una destinazione
- **Load**
 - una misura del carico di un link
- **Reliability**
 - riferita, ad esempio, all'error rate di un link
- **Hop count**
 - il numero di router da attraversare nel percorso dalla sorgente alla destinazione
- **Cost**
 - un valore arbitrario che definisce il costo di un link
 - ad esempio, costruito come funzione di diversi parametri (tra cui bandwidth, delay, packet loss, MTU,...)

6

Graph abstraction: costs



- $c(x,y)$ = cost of link (x,y)

- e.g., $c(w,z) = 5$

- cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

Cost of path $(x_1, x_2, x_3, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Il processo di routing



- Il processo di routing è un processo decisionale
- Ogni entità che partecipa a questo processo:
 - mantiene delle informazioni
 - in base ad uno specifico algoritmo ed in funzione di determinate metriche:
 - definisce il procedimento di instradamento verso le possibili destinazioni
 - può spedire informazioni di aggiornamento alle altre entità coinvolte, secondo diversi paradigmi

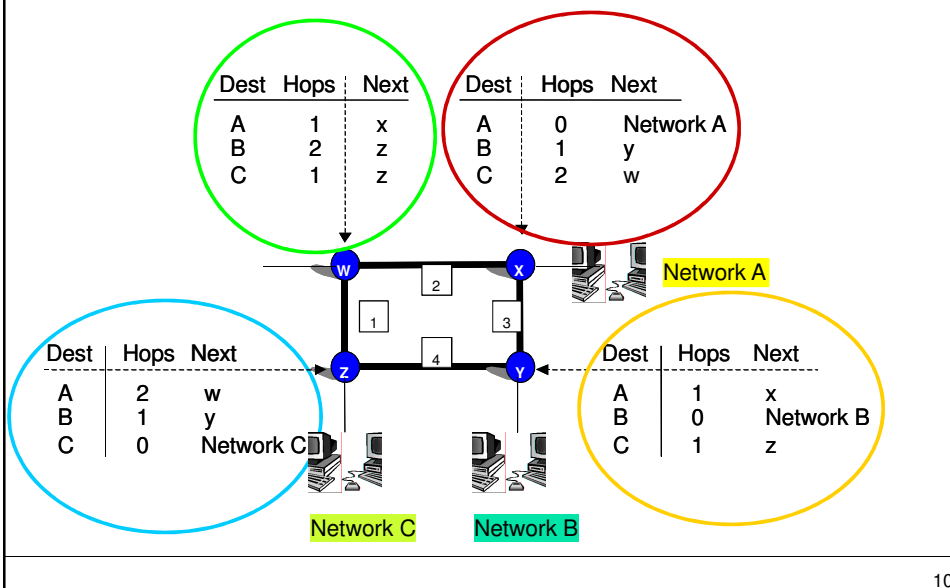
Il routing e la funzione di un router



- La funzione principale di un router è quella di determinare i percorsi che i pacchetti devono seguire per arrivare a destinazione, partendo da una data sorgente:
 - ogni router si occupa, quindi, del processo di ricerca di un percorso per l'instradamento di pacchetti tra due nodi qualunque di una rete
- Problemi da risolvere:
 - Quale sequenza di router deve essere attraversata?
 - Esiste un percorso migliore (più breve, meno carico, ...)?
 - Cosa fare se un link si guasta ?
 - Trovare una soluzione robusta e scalabile ...

9

Un esempio di tabelle di routing



10

Tecniche di routing



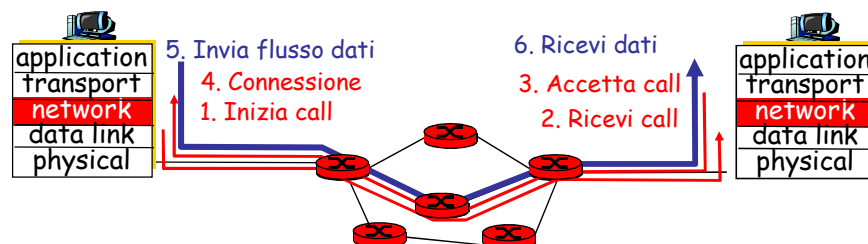
- Routing by Network Address
 - ogni pacchetto contiene l'indirizzo del nodo destinatario, che viene usato come chiave di accesso alle tabelle di instradamento
 - usato tipicamente nei protocolli non orientati alla connessione:
 - IPv4 e IPv6, bridge trasparenti, OSI CLNP, ...
- Label Swapping
 - ogni pacchetto è marcato con una *label* (etichetta) che:
 - identifica la connessione
 - viene usata come chiave per determinare l'instradamento
 - generalmente usato nei protocolli orientati alla connessione:
 - X.25, ATM, MPLS, ...

11

Routing: reti a circuiti virtuali



- Viene aperta una connessione *prima* di inviare dati

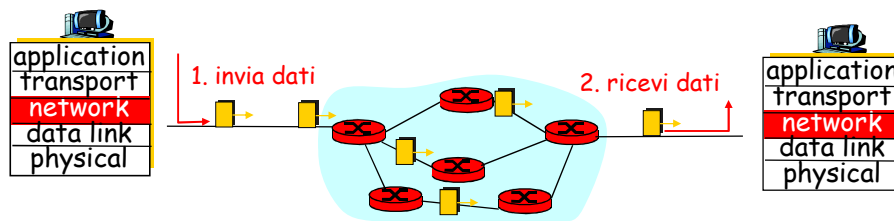


12

Routing: reti a datagramma



- Non esiste la fase di *call setup* a livello rete
- Nei router non esiste il concetto di connessione
- I pacchetti sono indirizzati usando un ID di destinazione:
 - pacchetti fra la stessa coppia sorgente-destinazione possono seguire strade diverse



13

Tipologie di routing



- La scelta del percorso di instradamento può essere realizzata mediante due approcci:
 - centralizzato
 - più semplice, ma non scalabile
 - distribuito
 - più complesso, ma scalabile e robusto
- Lo scopo ultimo di un protocollo di routing consiste nel creare una **tabella di instradamento** in ciascun nodo della rete:
 - ciascun nodo deve prendere una decisione locale sulla base della conoscenza dello stato dell'intera rete
 - Questa è, probabilmente, la difficoltà principale del routing

14

Routing centralizzato



- Esiste un nodo centrale che calcola e distribuisce le tabelle
 - tale nodo riceve informazioni sullo stato della rete da tutti gli altri e calcola le nuove tabelle
- Ottimizza le prestazioni, ma è poco robusto
 - aggiornamenti parziali delle tabelle dovuti a guasti possono generare loop
 - induce un notevole carico sulla rete, specialmente in prossimità del nodo centrale

15

Routing distribuito



- Ogni router calcola le sue tabelle dialogando con gli altri router:
 - Ogni router informa gli altri riguardo le “rotte” che conosce
- Il dialogo tra router avviene tramite dei protocolli ausiliari di livello rete
- Comprende due approcci principali:
 - Algoritmi *distance vector*
 - Algoritmi *link state*
- Utilizzato in varie reti proprietarie, in OSI, ed in **Internet**

16

Problematiche associate al routing



- Un router deve opportunamente sintetizzare le informazioni rilevanti utili alle proprie decisioni:
 - per prendere correttamente decisioni locali bisogna avere almeno una conoscenza parziale dello stato globale della rete
 - lo stato globale della rete è difficile da conoscere in quanto si può riferire ad un dominio molto esteso e che cambia in maniera estremamente dinamica
- Le tabelle di routing devono essere memorizzate all'interno dei router:
 - bisogna minimizzare l'occupazione di spazio e rendere rapida la ricerca
 - Bisogna minimizzare il numero di messaggi che i router si scambiano
- Si deve garantire la robustezza dell'algoritmo

17

Scambio delle informazioni di *update*



- Broadcast periodico
 - i router possono trasmettere agli altri router informazioni circa la raggiungibilità delle reti (destinazioni) di propria competenza ad intervalli regolari di tempo
 - questa tecnica risulta inefficiente, in quanto si spediscono informazioni anche quando non è cambiato nulla rispetto all'update precedente
- Event-driven
 - in questo caso gli update sono inviati solo quando è cambiato qualcosa nella topologia oppure nello stato della rete
 - questa tecnica garantisce un uso più efficiente della banda disponibile

18

Scelta dell'algoritmo di routing: problematiche



- Possono esistere più criteri di ottimalità contrastanti:
 - Es: “minimizzare il ritardo medio di ogni pacchetto” vs “massimizzare l'utilizzo dei link della rete”
- Il numero di nodi può essere elevato
- La topologia può essere complessa
- Algoritmi troppo complessi, operanti su reti molto grandi, potrebbero richiedere tempi di calcolo inaccettabili
- Vincoli di tipo amministrativo

19

Scelta dell'algoritmo di routing: parametri



- **Semplicità**
 - I router hanno CPU e memoria finite
- **Robustezza**
 - Adattabilità alle variazioni (di topologia, di carico, ...)
- **Stabilità**
 - L'algoritmo deve convergere in tempo utile
- **Equità**
 - Stesso trattamento a tutti i nodi
- **Metrica da adottare**
 - Numero di salti effettuati, somma dei costi di tutte le linee attraversate, ecc.

20

Distance Vector



- Ogni router mantiene una tabella di tutti gli instradamenti a lui noti
 - inizialmente, solo le reti a cui è connesso direttamente
- Ogni entry della tabella indica:
 - una rete raggiungibile
 - il *next hop*
 - il numero di hop necessari per raggiungere la destinazione
- Periodicamente, ogni router invia a tutti i vicini (due router sono vicini se sono collegati alla stessa rete fisica):
 - un messaggio di aggiornamento contenente tutte le informazioni della propria tabella (**vetto**re delle distanze – **distance vector**)
- I router che ricevono tale messaggio aggiornano la tabella nel seguente modo:
 - eventuale modifica di informazioni relative a cammini già noti
 - eventuale aggiunta di nuovi cammini
 - eventuale eliminazione di cammini non più disponibili

21

Distance Vector: un esempio



Destin.	Dist.	Route
net 1	0	direct
net 2	0	direct
net 4	8	router L
net 17	5	router M
net 24	6	router A
net 30	2	router Q
net 42	2	router A

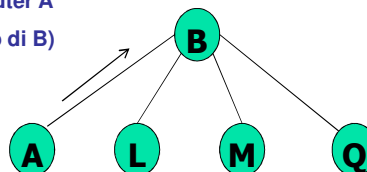
Tabella del router B

Destin.	Dist.
net 1	2
net 4	3
net 17	6
net 21	4
net 24	5
net 30	10
net 42	3

Messaggio di
aggiornamento
del router A
(vicino di B)

Destin.	Dist.	Route
net 1	0	direct
net 2	0	direct
net 4	4	router A
net 17	5	router M
net 24	6	router A
net 30	2	router Q
net 42	4	router A
net 21	5	router A

Tabella aggiornata del router B



22

Distance Vector Algorithm



Bellman-Ford Equation (dynamic programming)

Define

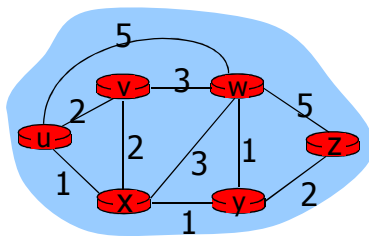
$d_x(y) := \text{cost of least-cost path from } x \text{ to } y$

Then

$$d_x(y) = \min_v \{ c(x,v) + d_v(y) \}$$

where min is taken over all neighbors v of x

Bellman-Ford example



Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

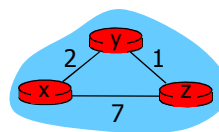
$$\begin{aligned} d_u(z) &= \min \{ c(u,v) + d_v(z), \\ &\quad c(u,x) + d_x(z), \\ &\quad c(u,w) + d_w(z) \} \\ &= \min \{ 2 + 5, \\ &\quad 1 + 3, \\ &\quad 5 + 3 \} = 4 \end{aligned}$$

Node that achieves minimum is next
hop in shortest path → forwarding table

		cost to		
		x	y	z
from	x	∞	∞	
	y	∞	∞	∞
	z	7	1	0

		cost to		
		x	y	z
from	x	0	2	3
	y	2	0	1
	z	7	1	0

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} \\ = \min\{2+1, 7+0\} = 3$$



time

from

	x	y	z
x	∞	∞	
y		∞	∞
z	7	1	0

from

	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

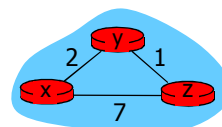
cost to

	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

		cost to		
		x	y	z
from	x	0	2	3
	y	2	0	1
	z	3	1	0

		cost to		
		x	y	z
from	x	0	2	3
	y	2	0	1
	z	3	1	0

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} = \min\{2+1, 7+0\} = 3$$



→ time

Distance Vector: analisi



- Vantaggi:
 - facile da implementare
- Svantaggi
 - ogni messaggio contiene un'intera tabella di routing
 - lenta propagazione delle informazioni sui cammini:
 - se lo stato della rete cambia velocemente, le rotte possono risultare inconsistenti