

Reti di Calcolatori I

Prof. Roberto Canonico

Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione

Corso di Laurea in Ingegneria Informatica

A.A. 2019-2020

Routing nelle reti a commutazione di pacchetto Routing Link State: introduzione

**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

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

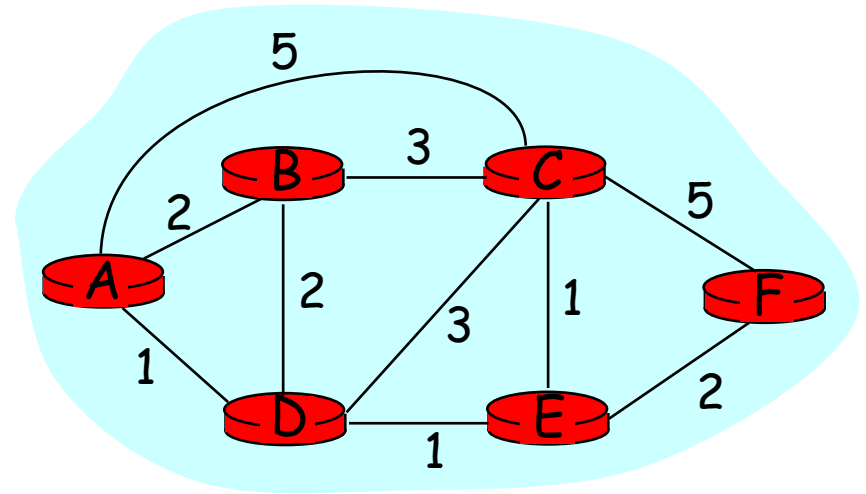
Il livello rete nella pila OSI



Reti di calcolatori e grafi

Rete modellata come grafo:

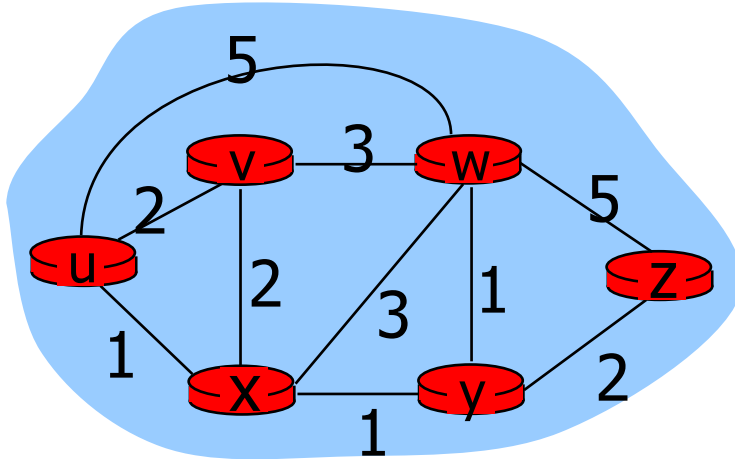
- **nodi** = router
- **archi** = link fisici
 - **costo link:**
 - ritardo,
 - costo trasmissione,
 - congestione,....
- Scelta del cammino:
 - cammino a costo minimo
 - altre possibilità (un cammino calcolato in base a specifici vincoli...)
- Gli algoritmi per la gestione di una rete sono basati sulla teoria dei grafi



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,...)

Graph abstraction: costs



- $c(x,y)$ = cost of link (x,y)
 - e.g., $c(w,z) = 5$
- cost could represent hop count, bandwidth, or 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

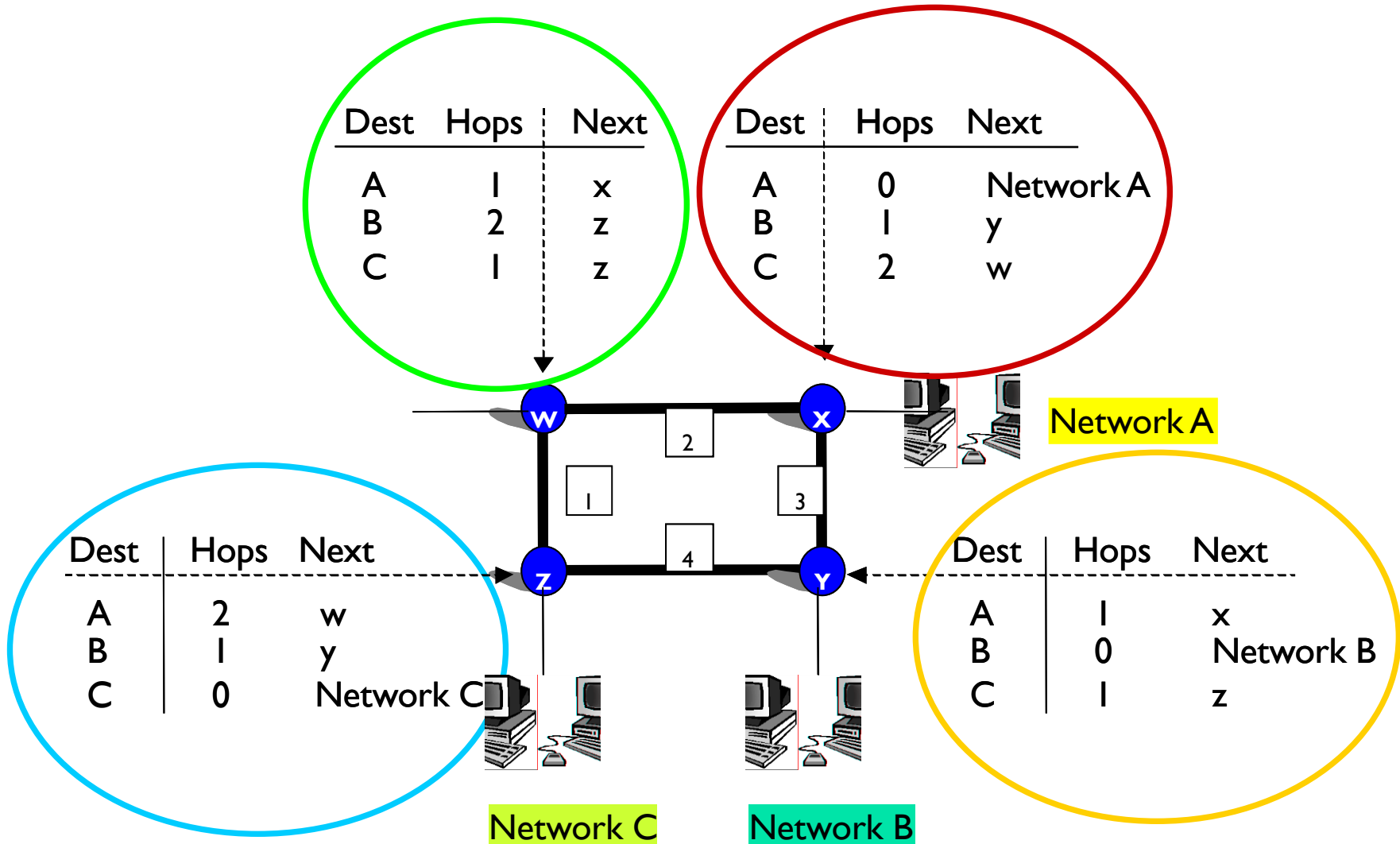
Il processo di routing

- 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 ...

Un esempio di tabelle di routing

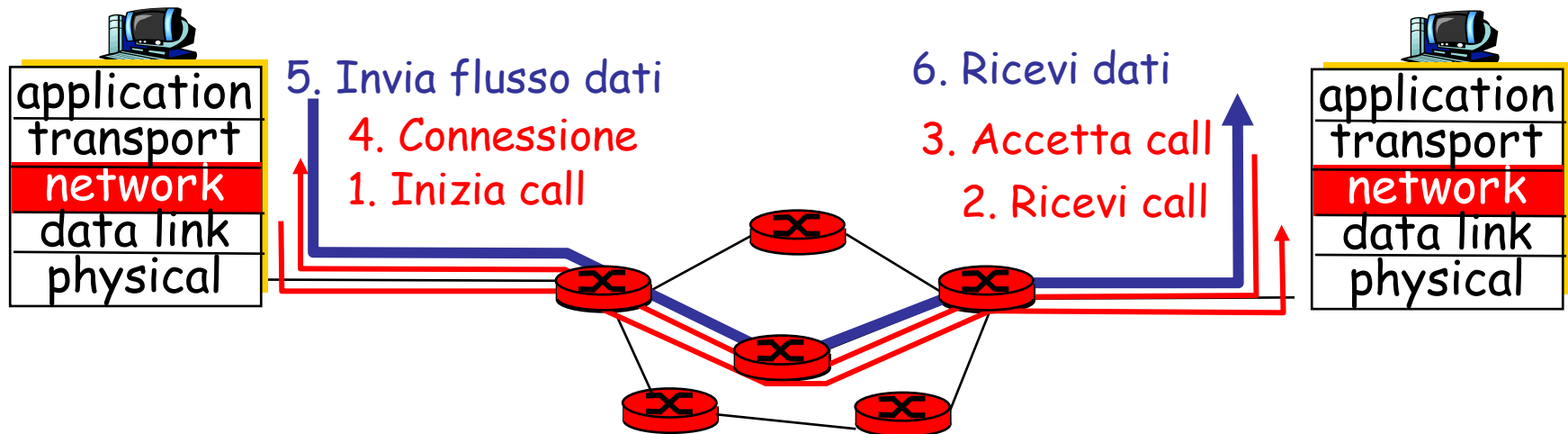


Tecniche di routing

- 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, ...

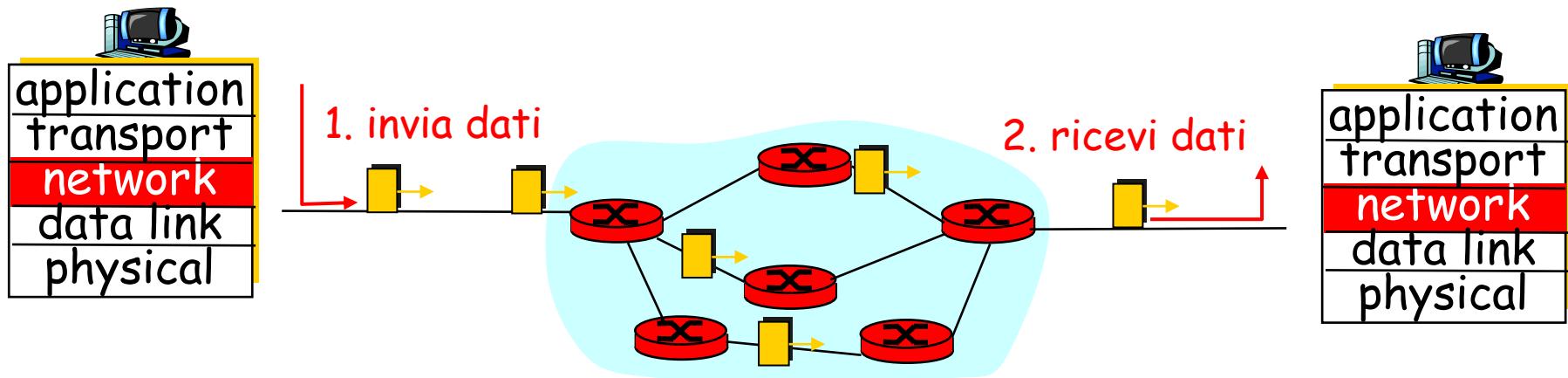
Routing: reti a circuiti virtuali

- Viene aperta una connessione *prima* di inviare dati

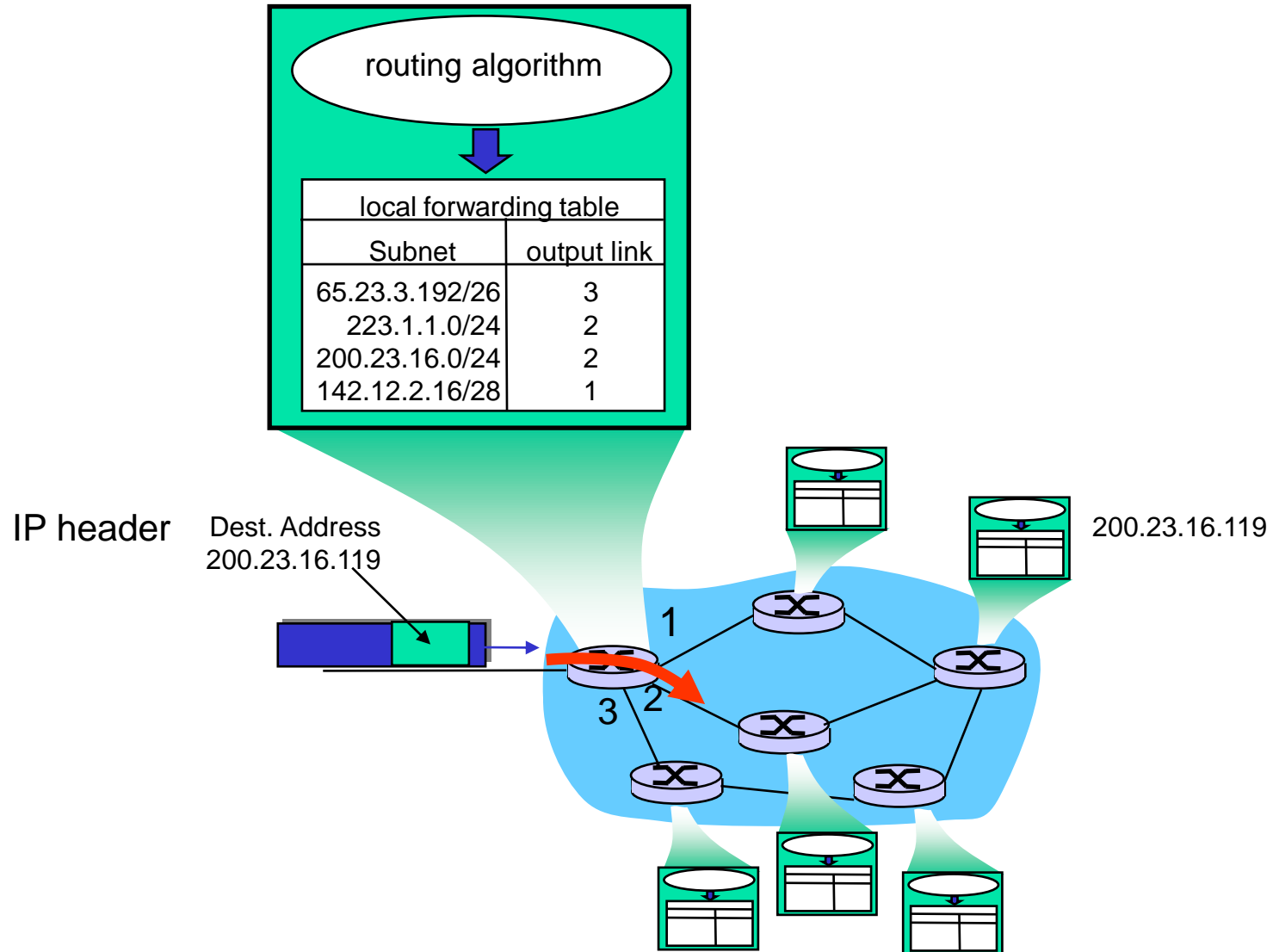


Routing: reti a datagramma

- 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



Un esempio di tabelle di routing per IP



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

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

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**

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

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

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

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.

Algoritmi di routing Link State

- Ogni router:
 - impara il suo ambito locale (linee e nodi adiacenti)
 - trasmette queste informazioni a tutti gli altri router della rete tramite un *Link State Packet (LSP)*
 - memorizza gli LSP trasmessi dagli altri router e costruisce una mappa della rete
 - Calcola, in maniera indipendente, le sue tabelle di instradamento applicando alla mappa della rete l'algoritmo di Dijkstra, noto come *Shortest Path First (SPF)*
- Tale approccio è utilizzato nello standard ISO 10589 (protocollo IS-IS) e nel protocollo OSPF (adottato in reti TCP/IP)

Il processo di *update*

- Ogni router genera un Link State Packet (LSP) contenente:
 - stato di ogni link connesso al router
 - identità di ogni vicino connesso all'altro estremo del link
 - costo del link
 - numero di sequenza per l'LSP
 - checksum
 - Lifetime:
 - la validità di ogni LSP è limitata nel tempo (e.g. un errore sul numero di sequenza potrebbe rendere un LSP valido per anni)

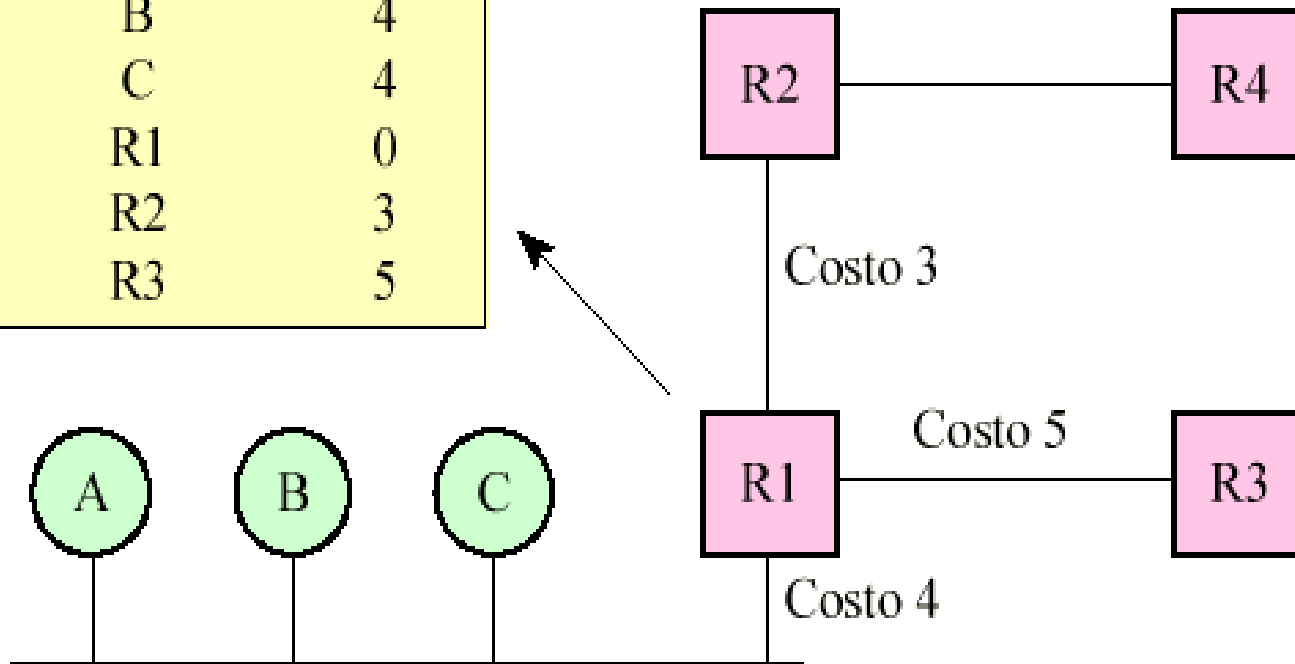
LSP flooding

- Un LSP viene generato periodicamente, oppure quando viene rilevata una variazione nella topologia locale (adiacenze), ossia :
 - Viene riconosciuto un nuovo vicino
 - Il costo verso un vicino e' cambiato
 - Si e' persa la connettività verso un vicino precedentemente raggiungibile
- Un LSP è trasmesso in flooding su tutti i link del router
- I pacchetti LSP memorizzati nei router formano una mappa completa e aggiornata della rete:
 - *Link State Database*

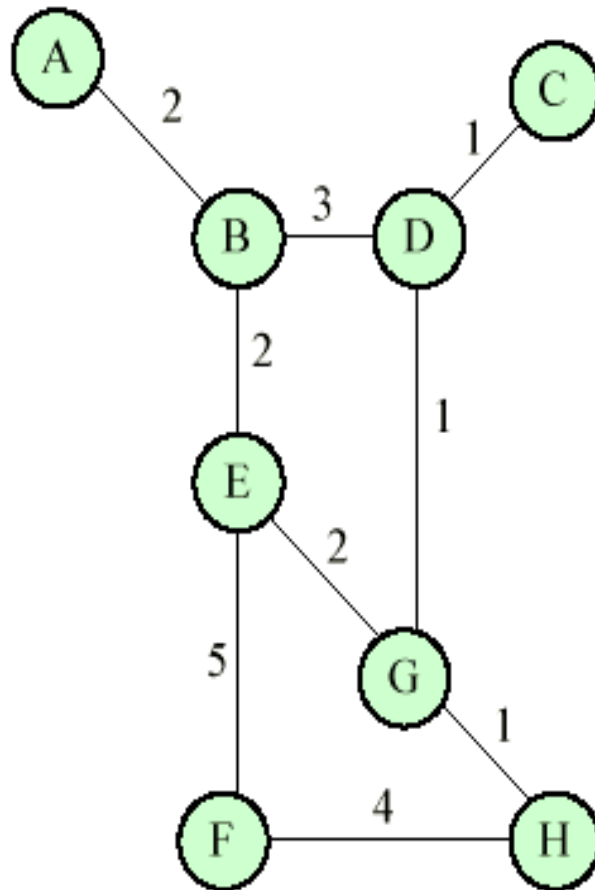
Esempio: trasmissione di un LSP

LSP trasmesso da R1

Adiacente	Costo
A	4
B	4
C	4
R1	0
R2	3
R3	5



Esempio: grafo della rete e LSP-DB



LSP Database

A	B/2		
B	A/2	D/3	E/2
C	D/1		
D	B/3	C/1	G/1
E	B/2	F/5	G/2
F	E/5	H/4	
G	D/1	E/2	H/1
H	F/4	G/1	

(replicato su ogni IS)

LSP database

SORGENTE

↓

	A	B	C	D	E	F	G	H
A	0	2						
B	2	0		3	2			
C			0	1				
D		3	1	0			1	
E		2			0	5	2	
F					5	0		4
G				1	2		0	1
H						4	1	0

DESTINAZIONE →

Questa rappresentazione è quella più appropriata per applicare l'algoritmo di Dijkstra

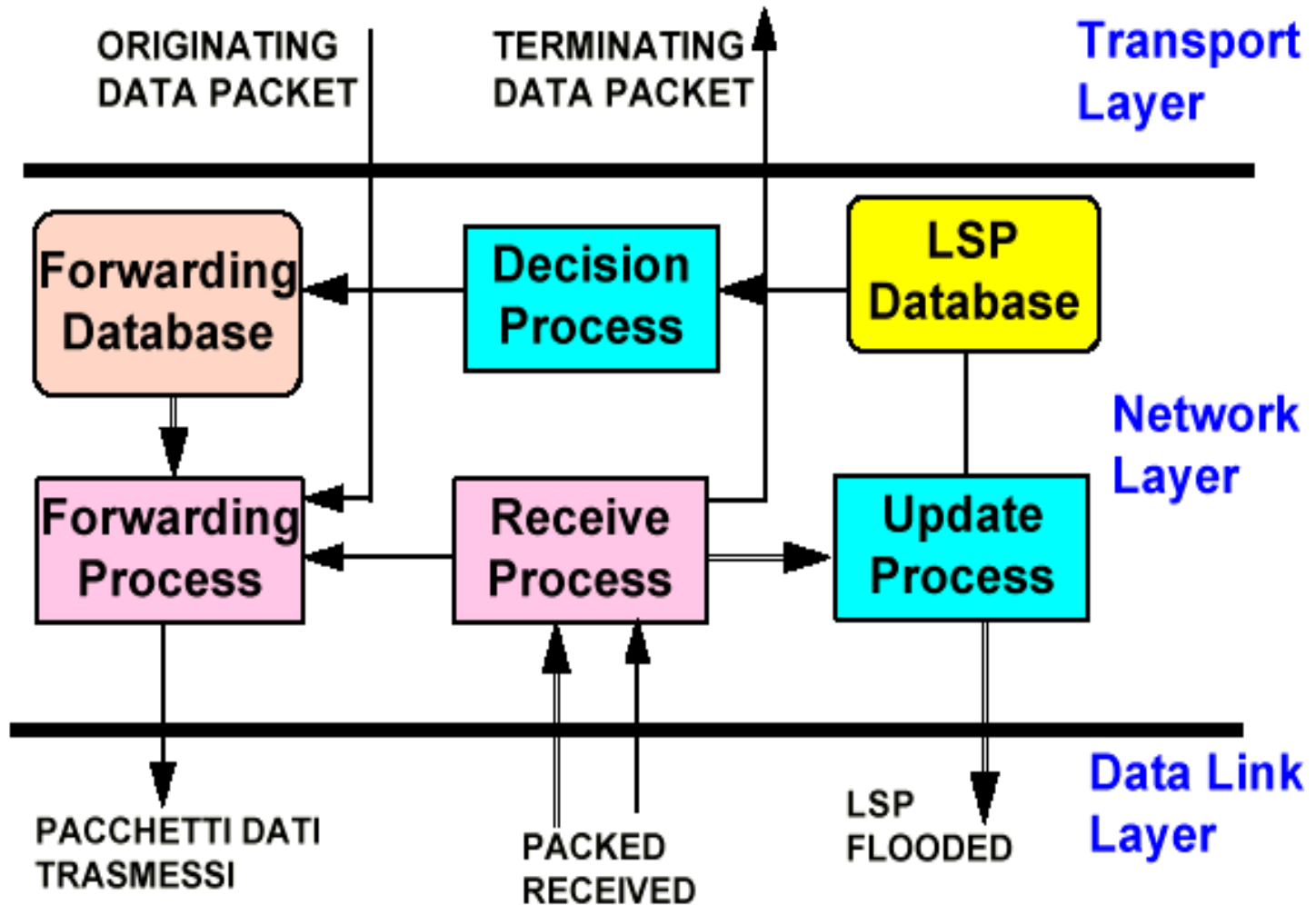
Gestione degli LSP

- All'atto della ricezione di un LSP, il router compie le seguenti azioni:
 1. se non ha mai ricevuto LSP da quel router o se l'LSP è più recente di quello precedentemente memorizzato:
 - memorizza il pacchetto
 - lo ritrasmette in flooding su tutte le linee eccetto quella da cui l'ha ricevuto
 2. se l'LSP ha lo stesso numero di sequenza di quello posseduto:
 - non fa nulla
 3. Se l'LSP è più vecchio di quello posseduto:
 - trasmette al mittente il pacchetto più recente

Routing: decisioni

- Il router elabora il *Link State Database* per produrre il *Forwarding Database*:
 - si pone come radice dello shortest-path tree
 - cerca lo shortest path per ogni nodo destinazione
 - memorizza il vicino (i vicini) che sono sullo shortest path verso ogni nodo destinazione
- Il *Forwarding Database* contiene, per ogni nodo destinazione:
 - l'insieme delle coppie {path, vicino}
 - la dimensione di tale insieme

Architettura di un router Link State



Link State: caratteristiche

- Vantaggi:
 - può gestire reti di grandi dimensioni
 - ha una convergenza rapida
 - difficilmente genera loop, e comunque è in grado di identificarli ed interromperli facilmente
 - facile da capire: ogni nodo ha la mappa della rete
- Svantaggi:
 - Molto complesso da realizzare:
 - Es: la prima implementazione ha richiesto alla Digital 5 anni

Esempio: tabelle di instradamento

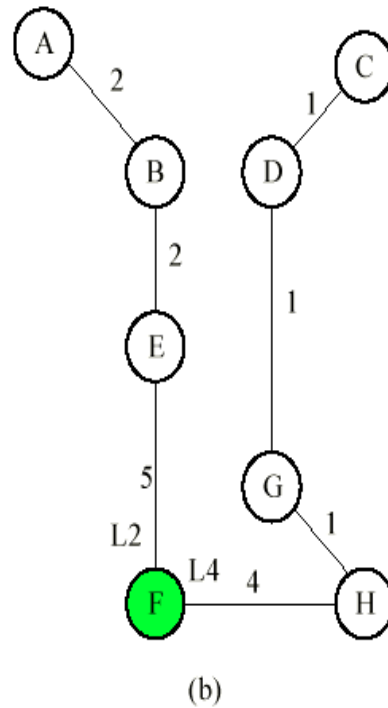
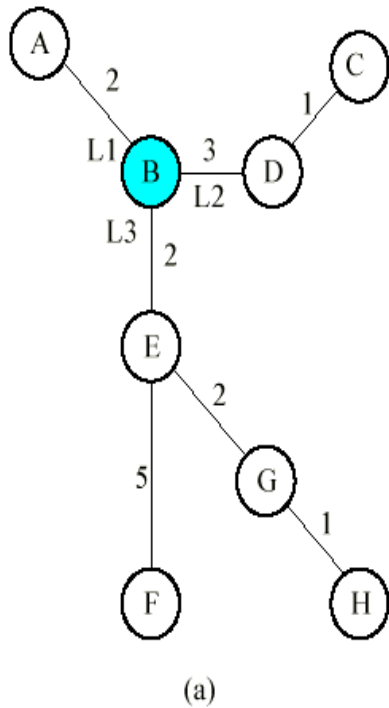


Tabella di B

A	L1
C	L2
D	L2
E	L3
F	L3
G	L3
H	L3

Tabella di F

A	L2
B	L2
C	L4
D	L4
E	L2
G	L4
H	L4