Livello di Rete: Routing, protocollo RIP

Prof.ssa Gaia Maselli
maselli@di.uniroma1.it

Parte di queste slide sono state prese dal materiale associato ai libri:

1) B.A. Forouzan, F. Mosharraf – Reti di calcolatori. Un approccio top-down. Copyright © 2013 McGraw-Hill Education Italy srl. Edizione italiana delle slide a cura di Gabriele D'Angelo e Gaia Maselli
2) Computer Networking: A Top Down Approach , 6th edition. All material copyright 1996-2009 J.F Kurose and K.W. Ross, All Rights Reserved

Forwarding datagrammi IP

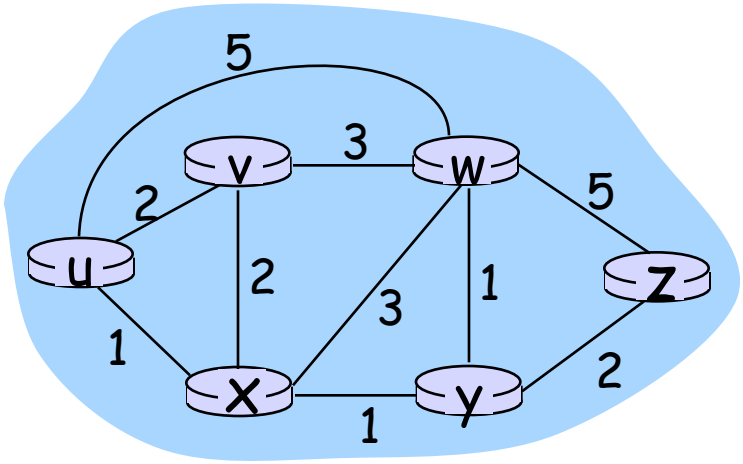
- Inoltrare significa collocare il datagramma sul giusto percorso (porta di uscita del router) che lo porterà a destinazione (o lo farà avanzare verso la destinazione)
- Inviare il datagramma al prossimo hop
- Quando un host ha un datagramma da inviare lo invia al router della rete locale
- Quando un router riceve un datagramma da inoltrare, accede alla tabella di routing per trovare il successivo hop a cui inviarlo
- L'inoltro richiede una riga nella tabella per ogni blocco di rete

Routing

- Quale percorso deve seguire un pacchetto che viene instradato da un router sorgente a un router destinazione?
- Sono disponibili più percosi… quale si sceglie?
- Il routing si occupa di trovare il miglior percorso (o rotta, path, route) e inserirlo nella tabella di routing (o forwarding)

Il routing costruisce le tabelle, il forwarding le usa!

Grafo di una rete di calcolatori



Grafo: G = (N,E)

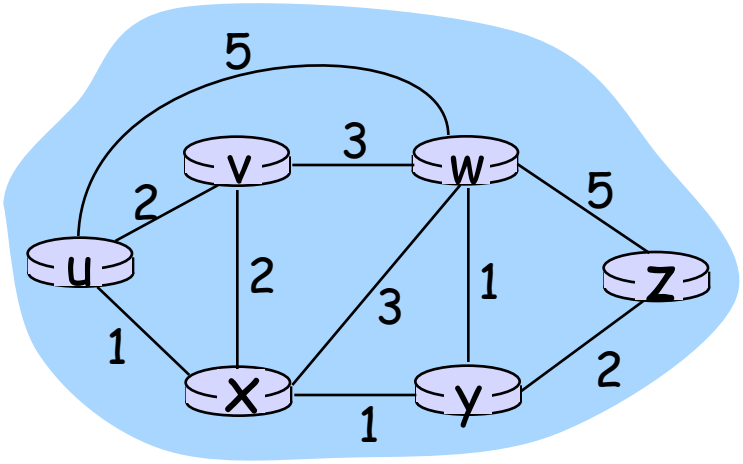
N = insieme di nodi (router) = { u, v, w, x, y, z }

E = insieme di archi (collegamenti) ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

N.B.: Il grafo è un'astrazione utile anche in altri contesti di rete

Esempio: P2P, dove N è un insieme di peer ed E è un insieme di collegamenti TCP

Grafo di una rete : costi



- c(x,x') = costo del collegamento (x,x')

  - es., c(w,z) = 5

- il costo di un cammino è semplicemente la somma di tutti i costi degli archi lungo il cammino

Costo di un cammino $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Domanda: Qual è il cammino a costo minimo tra u e z ?

Algoritmo d'instradamento: determina il cammino a costo minimo.

Algoritmo d'instradamento con vettore distanza (Distance Vector -DV)

Distribuito: ogni nodo riceve informazione dai vicini e opera su quelle

Asincrono: non richiede che tutti i nodi operino al passo con gli altri

Si basa su:

1. Equazione di Bellman-Ford
2. Concetto di vettore di distanza

Equanzione di Bellman-Ford

Formula di Bellman-Ford

Definisce

$D_x(y)$:= il costo (o la distanza) del percorso a costo minimo dal nodo *x* al nodo *y*.

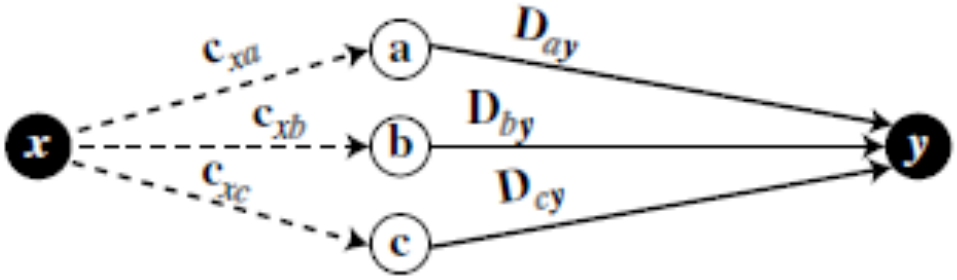
Allora:

$$D_x(y) = \min_v \{c(x,v) + D_v(y)\}$$

dove min_v riguarda tutti i vicini di *x*.

Rappresentazione grafica

I percorsi (a➔y), (b➔y), (c➔y) sono percorsi a costo minimo precedentemente stabiliti e (x➔y) è un nuovo percorso a costo minimo

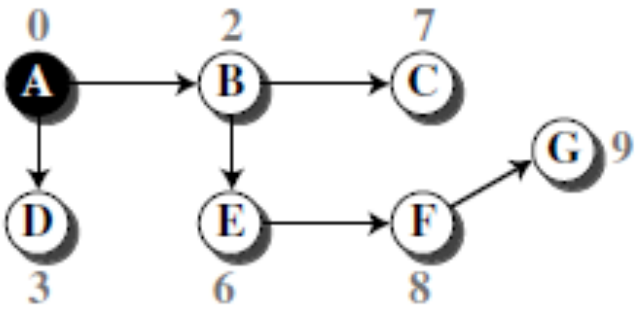


a. Caso generale con tre nodi intermedi

$$D_{xy} = \min\left\{\left(c_{xa} + D_{ay}\right), \left(c_{xb} + D_{by}\right), \left(c_{xc} + D_{cy}\right)\right\}$$

Vettore distanza

- Un albero a costo minimo è una combinazione di percorsi a costo minimo dalla radice dell'albero verso tutte le destinazioni
- Il vettore di distanza è un array monodimensionale che rappresenta l'albero
- Un vettore di distanza non fornisce il percorso da seguire per giungere alla destinazione ma solo i costi minimi per le destinazioni



a. Albero per il nodo A

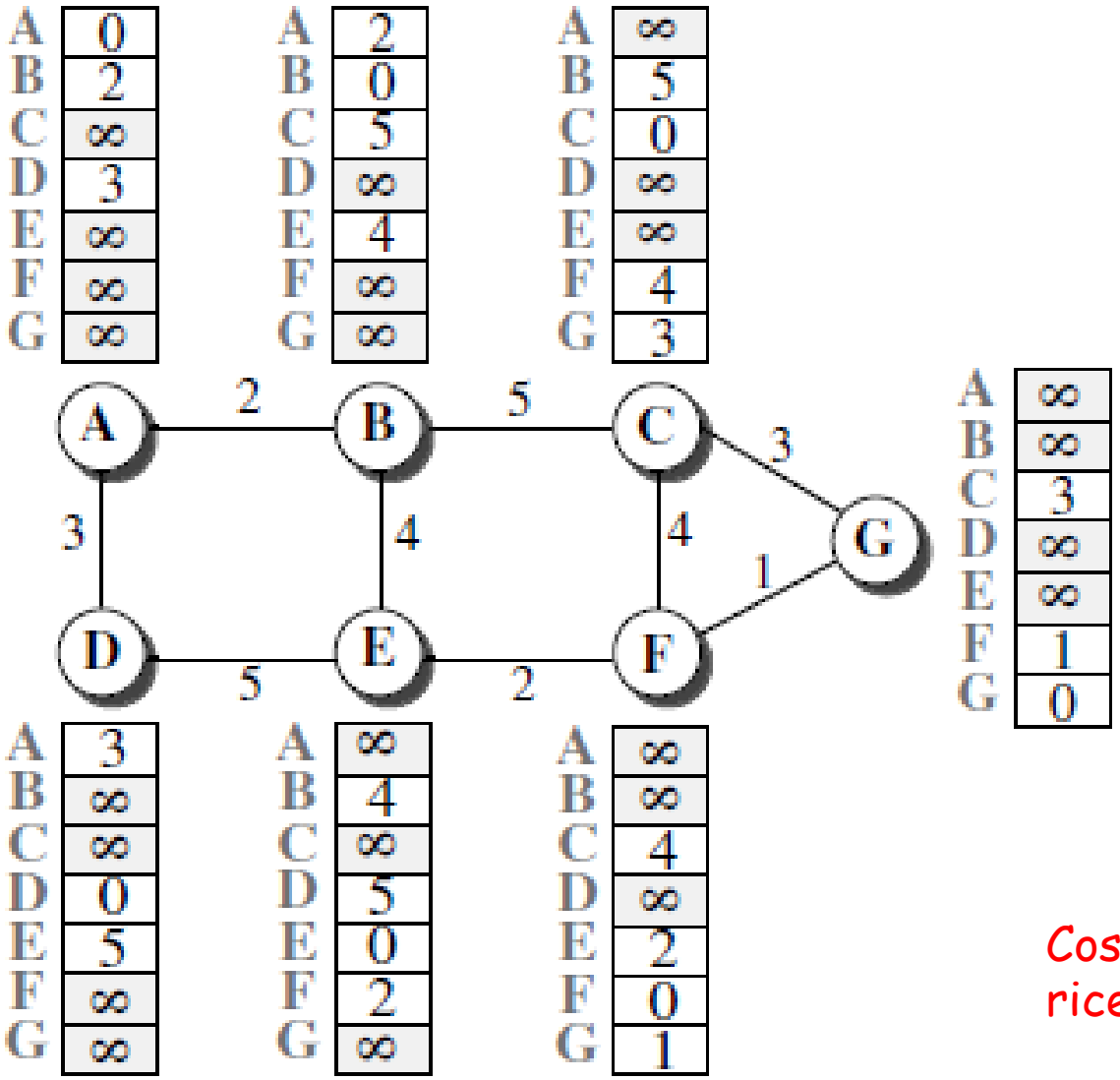
	A
A	0
B	2
C	7
D	3
E	6
F	8
G	9

b. Vettore distanza per il nodo A

Come viene creato il vettore distanza?

- Ogni nodo della rete quando viene inizializzato crea un vettore distanza iniziale con le informazioni che il nodo riesce a ottenere dai proprio vicini (nodi a cui è direttamente collegato)
- Per creare il vettore dei vicini invia messaggi di hello attraverso le sue interfacce (e lo stesso fanno i vicini) e scopre l'identità dei vicini e la sua distanza da ognuno di essi
- Il vettore iniziale rappresenta il vettore a costo minimo verso i vicini
- Dopo che ogni nodo ha creato il suo vettore ne invia una copia ai suoi vicini
- Quando un nodo riceve un vettore distanza da un vicino provvede ad aggiornare il suo vettore distanza applicando l'equazione di Bellman-Ford

Esempio

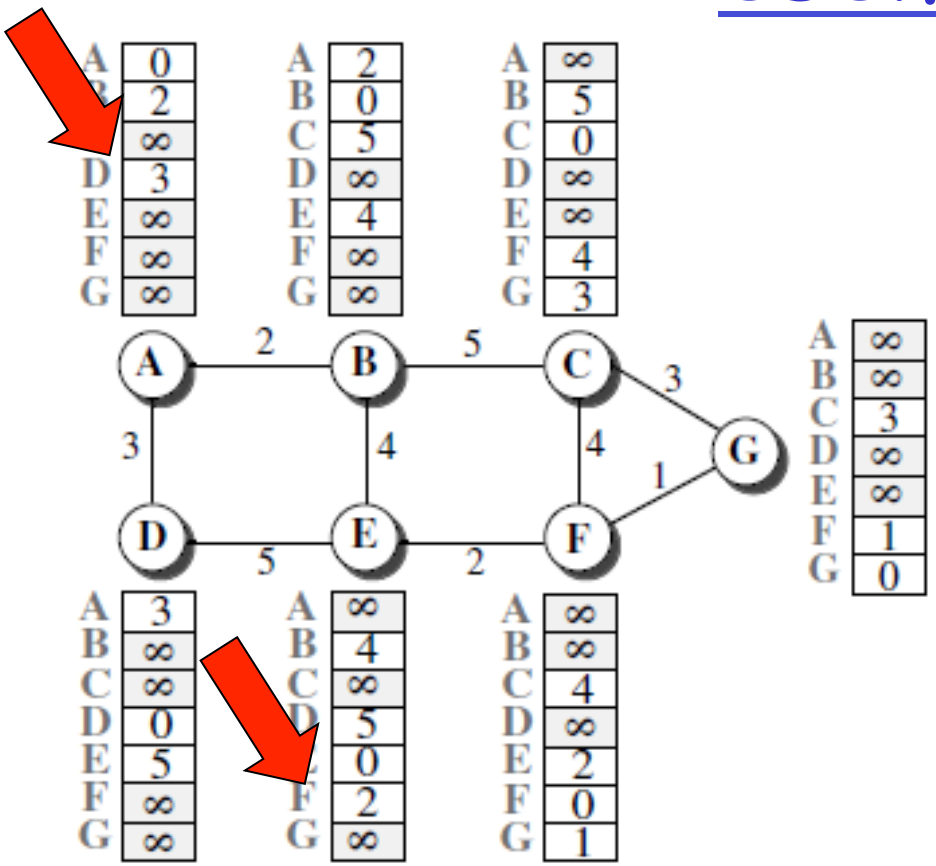


Cosa succede quando B riceve una copia di A?

Vettori distanza iniziali dopo messaggi di hello

esempio

Nota:
X[ ]: l'intero vettore



	Nuovo B	Vecchio B	A
A	2	2	0
B	0	0	2
C	5	5	∞
D	**5**	∞	3
E	4	4	∞
F	∞	∞	∞
G	∞	∞	∞

B[ ] = min (B[ ], 2 + A[ ])

a. Primo evento: B riceve una copia del vettore di A.

	Nuovo B	Vecchio B	E
A	2	2	∞
B	0	0	4
C	5	5	∞
D	5	5	5
E	4	4	0
F	**6**	∞	2
G	∞	∞	∞

B[ ] = min (B[ ], 4 + E[ ])

b. Secondo evento: B riceve una copia del vettore di E.

E se ora B riceve una copia di E?

Algoritmo con vettore distanza

Idea di base:

- Ogni nodo invia una copia del proprio vettore distanza a ciascuno dei suoi vicini.
- Quando un nodo x riceve un nuovo vettore distanza, DV, da qualcuno dei sui vicini, lo salva e usa la formula B-F per aggiornare in proprio vettore distanza come segue:

$$D_x(y) \leftarrow min_v\{c(x,v) + D_v(y)\} \quad per\ ciascun\ nodo\ y\ in\ N.$$

- Se il vettore distanza del nodo *x* è cambiato per via di tale passo di aggiornamento, il nodo *x* manderà il proprio vettore distanza aggiornato a ciascuno dei suoi vicini, i quali a loro volta aggiornano il loro vettore distanza

Algoritmo con vettore distanza

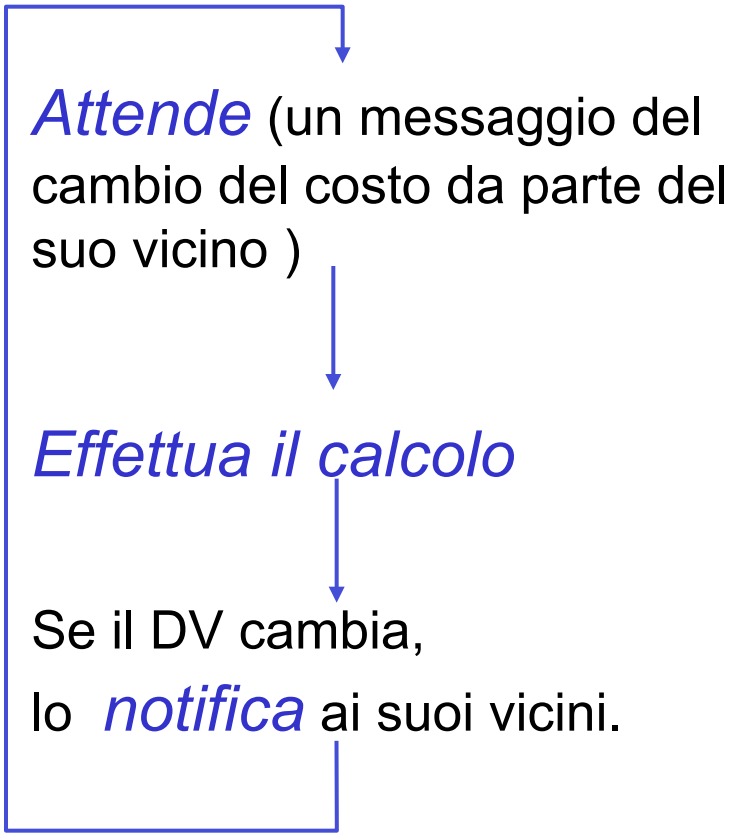
Iterativo, asincrono: ogni iterazione locale è causata da:

- cambio del costo di uno dei collegamenti locali
- ricezione da qualche vicino di un vettore distanza aggiornato.

Distribuito:

- Ogni nodo aggiorna i suoi vicini *solo* quando il suo DV cambia.
  - i vicini avvisano i vicini solo se necessario.

Ciascun nodo:



Algoritmo Distance Vector

A ciascun nodo x:

Inizializzazione

per tutte le destinazioni y in N:

    se y è un vicino
        $D_x(y) = c(x,y)$
    else $D_x(y) = \infty$

per ciascun vicino w

    invia il vettore distanza $\mathbf{D}_x = [D_x(y): y \text{ in } N]$ a w

Ciclo

attendi (finchè vedi cambiare il costo di un collegamento verso qualche vicino w o ricevi un vettore distanza da qualche vicino w)

per ogni y in N:

    $D_x(y) = \min_v \{c(x,v) + D_v(y)\}$

se qualche $D_x(y)$ è cambiato invia vettore distanza $\mathbf{D}_x$ a tutti i vicini

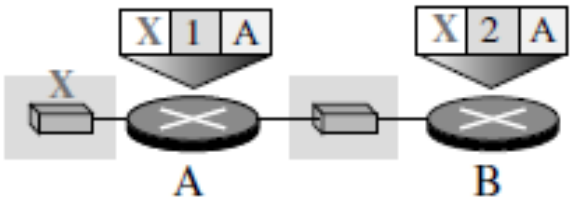
Algoritmo con vettore distanza: modifica dei costi e problema

Modifica dei costi:

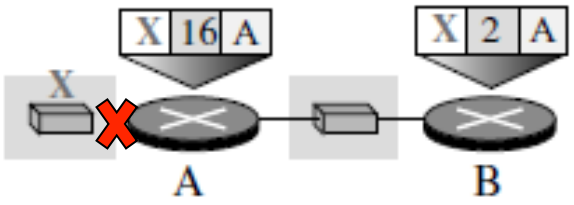
- Un nodo rileva un cambiamento nel costo dei collegamenti.
- Aggiorna il proprio vettore distanza.
- Se si verifica un cambiamento nel costo, trasmette ai suoi vicini il nuovo DV.

Problema del conteggio all'infinito: *Le buone notizie viaggiano in fretta Le cattive notizie si propagano lentamente*

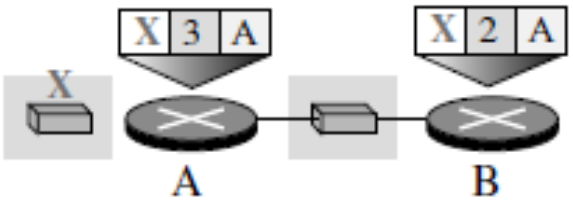
Esempio: si guasta il collegamento tra A e X



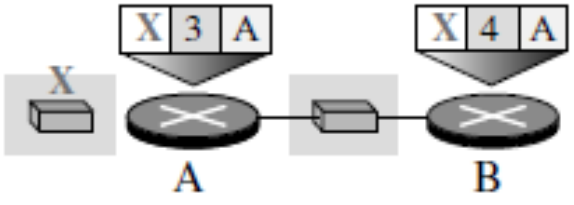
a. Prima del guasto



b. Dopo il guasto del collegamento

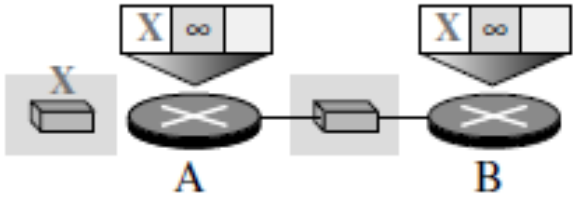


c. Dopo che A è aggiornato da B



d. Dopo che B è aggiornato da A

I pacchetti rimbalzano tra A e B creando un ciclo a due nodi. Il sistema diventa stabile dopo molti aggiornamenti



e. Alla fine

Soluzioni

Split horizon

- Invece di inviare la tabella attraverso ogni interfaccia, ciascun nodo invia solo una parte della sua tabella tramite le interfacce
- Se il nodo B ritiene che il percorso ottimale per raggiungere X passi attraverso A, allora non deve fornire questa informazione ad A (L'informazione è arrivata da A e quindi la conosce già)
- Nell'esempio B elimina la riga di X dalla tabella prima di inviarla ad A

Poisoned reverse (inversione avvelenata)

- Si pone a ∞ il valore del costo del percorso che passa attraverso il vicino a cui si sta inviando il vettore
- Nell'esempio B pone a ∞ il costo verso X quando invia il vettore ad A

Algoritmo Distance vector

Protocollo RIP

RIP (Routing Information Protocol)

- È un protocollo a vettore distanza.
- È tipicamente incluso in UNIX BSD dal 1982.
- Metrica di costo: distanza misurata in hop (max = 15 hop, il valore 16 indica l'infinito)
  - Ogni link ha costo unitario

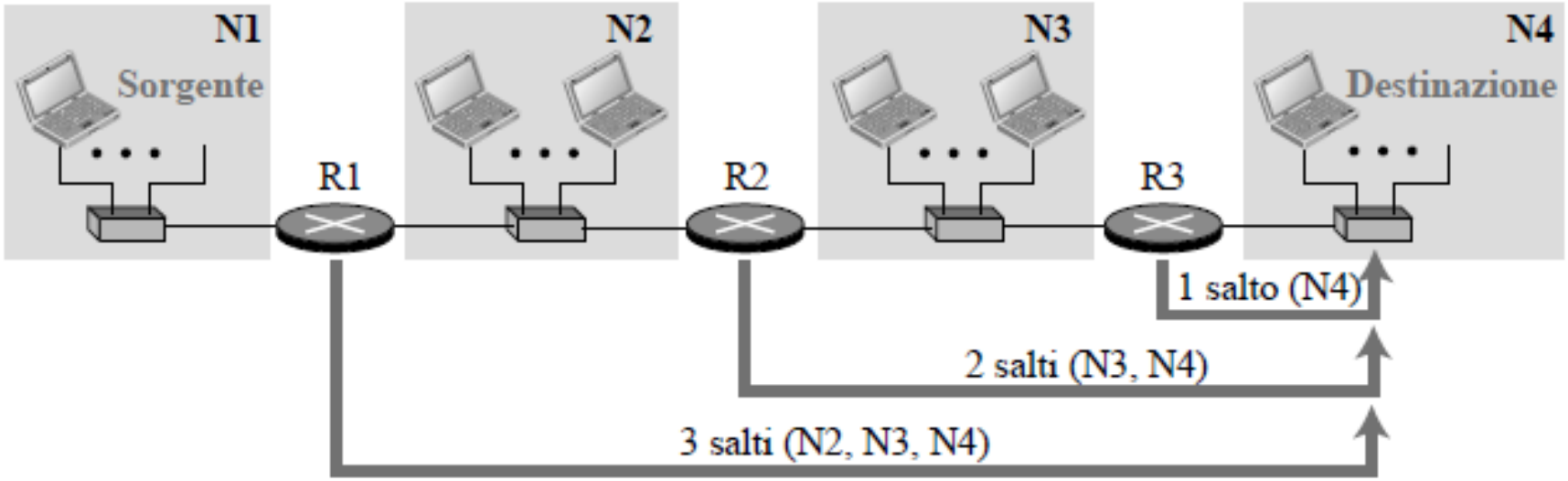


Tabelle di routing

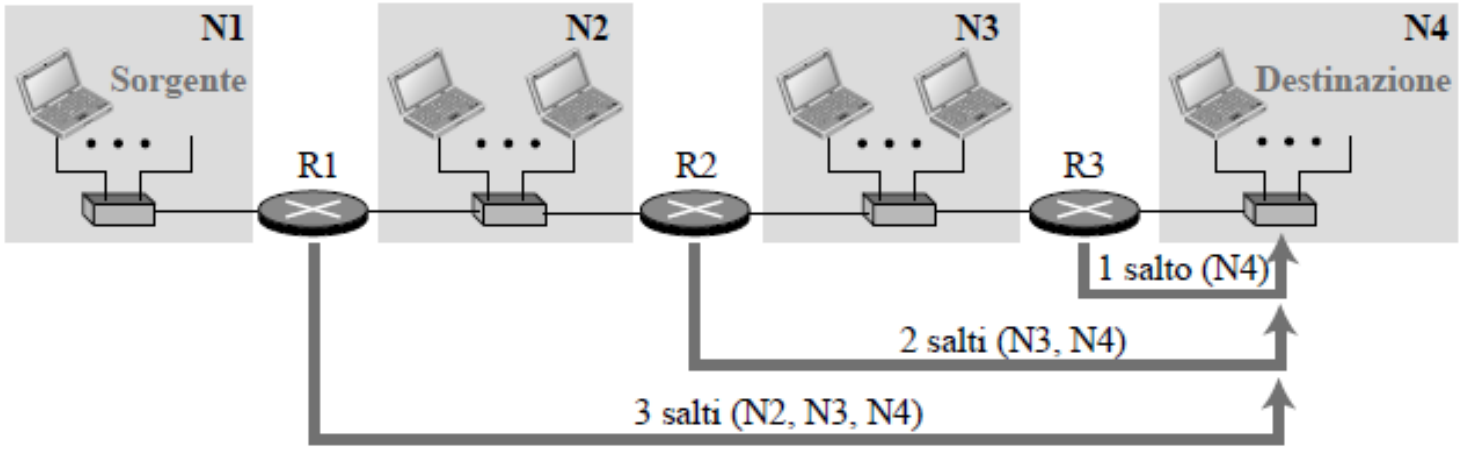


Tabella d'inoltro per R1

Rete di destinazione	Prossimo router	Costo (in hop)
N1	——	1
N2	——	1
N3	R2	2
N4	R2	3

Tabella d'inoltro per R2

Rete di destinazione	Prossimo router	Costo (in hop)
N1	R1	2
N2	——	1
N3	——	1
N4	R3	2

Tabella d'inoltro per R3

Rete di destinazione	Prossimo router	Costo (in hop)
N1	R2	3
N2	R2	2
N3	——	1
N4	——	1

L'informazione nella tabella di routing è sufficiente per raggiungere la destinazione

RIP protocol

- ***RIP Route Determination Algorithm***

  On a regular basis (= peridically), each router running RIP will send out its routing table entries (distance vector) to provide information to other routers about the networks and hosts it knows how to reach. Any routers on the same network as the one sending out this information will be able to update their own tables based on the information they receive. Any router that receives a message from another router on the same network saying it can reach network *X* at a cost of *N*, knows it can reach network *X* at a cost of *N+1* by sending to the router it received the message from.

N.B. Invece di inviare solo vettori di distanza, i router inviano l'intero contenuto della tabella di routing

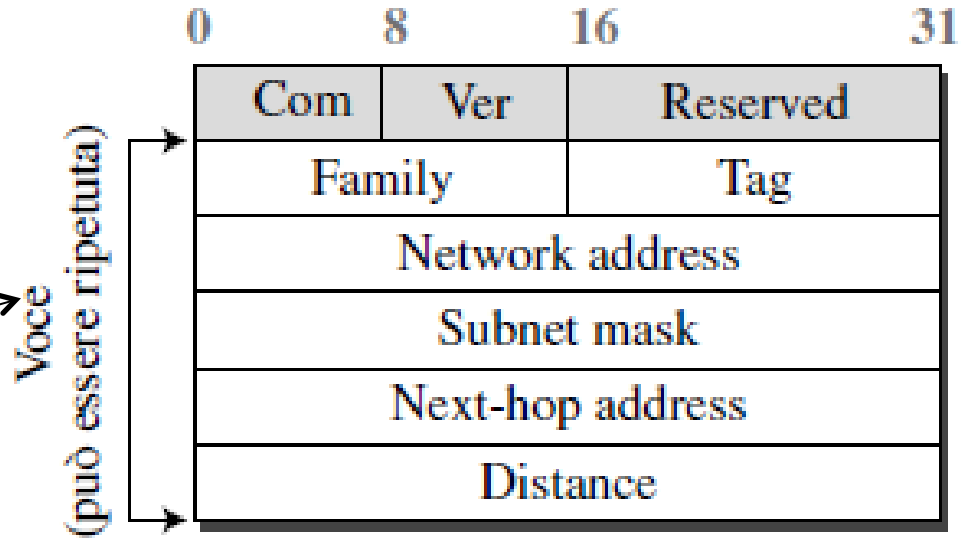
Messaggi RIP

- RIP si basa su una coppia di processi client-server e sul loro scambio di messaggi
- RIP Request:
  - Quando un nuovo router viene inserito nella rete invia una RIP Request per ricevere immediatamente informazioni di routing
  - Fini diagnostici (richiedere una voce specifica)
- RIP Response (o advertisements):
  - In risposta a una Request (solicited response)
  - Peridicamente ogni 30 sec (unsolicited response)

- Ogni messaggio contiene un elenco comprendente fino a 25 sottoreti di destinazione all'interno del sistema autonomo nonché la distanza del mittente rispetto a ciascuna di tali sottoreti.

Struttura messaggi RIP

Corrisponde a una entry della tabella di routing



Campi

Com: comando, richiesta(1), risposta (2)
Ver: versione, la versione corrente è 2
Family: famiglia del protocollo, per il TCP/IP il valore è 2
Tag: informazioni sul sistema autonomo
Network address: indirizzo di destinazione
Subnet mask: maschera di sottorete (lunghezza del prefisso)
Next-hop address: indirizzo del prossimo hop
Distance: numero di hop fino alla destinazione

Timer RIP

- *Timer periodico*
  - Controlla invio messaggi di aggiornamento (25-35 secondi)
- *Timer di scadenza*
  - Regola la validità dei percorsi (180 secondi)
  - Se entro lo scadere del timer non si riceve aggiornamento, il percorso viene considerato scaduto e il suo costo impostato a 16
- *Timer per garbage collection*
  - Elimina percorsi dalla tabella (120 secondi)
  - Quando le informazioni non sono più valide, il router continua ad annunciare il percorso con costo pari a 16, e allo scadere del timer rimuove il percorso

RIP: guasto sul collegamento e recupero

Se un router non riceve notizie dal suo vicino per 180 sec --> il nodo adiacente/il collegamento viene considerato spento o guasto.

- RIP modifica la tabella d'instradamento locale
- Propaga l'informazione mandando annunci ai router vicini.
- I vicini inviano nuovi messaggi (se la loro tabella d'instradamento è cambiata).
- L'informazione che il collegamento è fallito si propaga rapidamente su tutta la rete.
- L'utilizzo dell'inversione avvelenata evita i loop (distanza infinita = 16 hop)

Caratteristiche di RIP

- *Split horizon with poisoned reverse* (inversione avvelenata)
  - Serve per evitare che un router invii rotte non valide al router da cui ha imparato la rotta (evitare cicli).
  - Si mette a infinito (16) il costo della rotta che passa attraverso il vicino a cui si manda advertisement
- *Triggered updates*
  - Riduce il problema della convergenza lenta
  - Quando cambia una rotta si inviano immediatamente informazioni ai vicini senza attendere il timeout.
- *Hold-down*
  - Fornisce robustezza
  - Quando si riceve una informazione di una rotta non più valida, si avvia un timer e tutti gli advertisement riguardanti quella rotta che arrivano entro il timeout vengono tralasciati

Implementazione di RIP

- Implementato come applicazione sopra UDP porta 520
- Un processo chiamato routed (route daemon) esegue RIP, ossia mantiene le informazioni d'instradamento e scambia messaggi con i processi routed nei router vicini.
- Poiché RIP viene implementato come un processo a livello di applicazione, può inviare e ricevere messaggi su una socket standard e utilizzare un protocollo di trasporto standard.