



*Corso di Laurea Triennale in Informatica
Università degli Studi della Basilicata*

Reti di Calcolatori

Docente: Ugo Erra

ugo.erra+reti@unibas.it

11° Lezione – Livello di rete– III° parte

Sommario



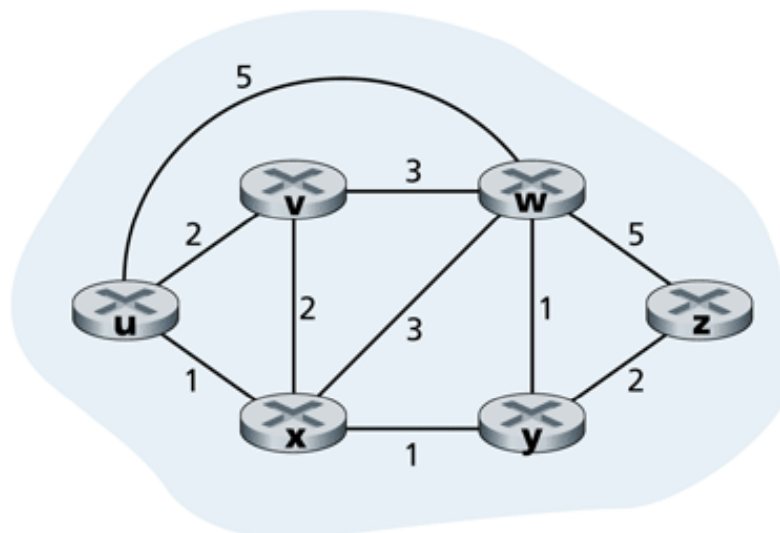
- **Algoritmi di instradamento**
 - Algoritmo di Dijkstra
 - Algoritmo vettore distanza

Algoritmi di instradamento



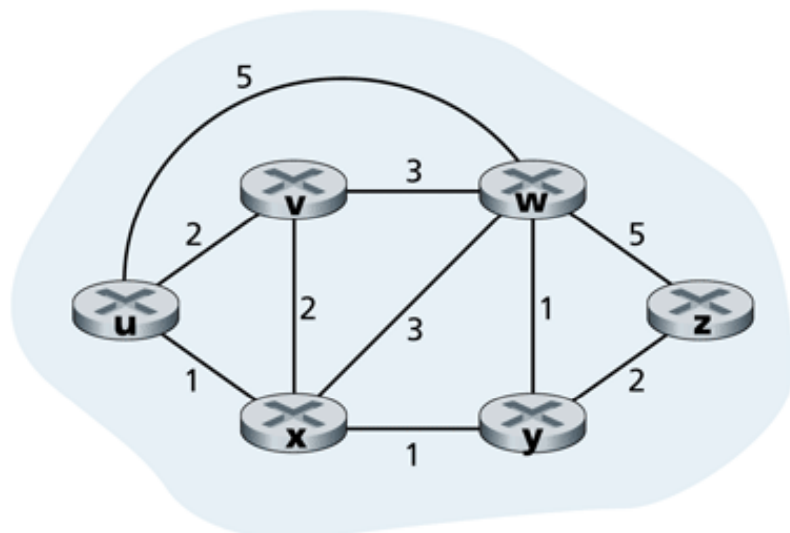
- Lo scopo degli algoritmi di instradamento è determinare buoni percorsi attraverso i router della rete
- Il default router collegato all'host è definito **router di primo hop**
- Il router predefinito dell'host di destinazione sarà il **router destinazione**
- Il problema dell'instradamento è determinare un percorso tra questi due router

Grafo di una rete di calcolatori



- Definiamo un **grafo**: $G = (N, E)$
 - ▣ Insieme di nodi $N = \{ u, v, w, x, y, z \}$
 - ▣ Insieme di archi $E = \{ (u, v), (u, x), (v, x), (v, w), (x, w), (x, y), (w, y), (w, z), (y, z) \}$
- Il grafo è un'astrazione utile per illustrare il funzionamento degli algoritmi di instradamento

Grafo di una rete:costi



- Definiamo il costo del collegamento tra il nodo x ed il nodo y come $c(x,y)$
 - Ad esempio $c(u,x)=1$, $c(w,z)=5$
 - Se un arco (u,v) non è presente nel grafo allora $c(u,v)=\infty$
-
- Definiamo un **cammino** in un grafo una sequenza di nodi $(x_1, x_2, x_3, \dots, x_p)$ tale che per ogni coppia (x_{p-1}, x_p) esiste un arco in G
 - Il costo di un cammino $(x_1, x_2, x_3, \dots, x_p)$ è semplicemente la somma di tutti i costi degli archi lungo il cammino
$$c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$$
 - Ad esempio $c(u, x, y, z) = c(u, x) + c(x, y) + c(y, z) = 1 + 1 + 2 = 4$
 - Un **algoritmo di instradamento determina un cammino di costo minimo**

Classificazione degli algoritmi di instradamento

- Globale:
 - ▣ L'algoritmo riceve in ingresso tutti i collegamenti tra i nodi e i loro costi
 - ▣ Algoritmi a stato del collegamento (LS, link-state algorithm)
- Decentralizzato:
 - ▣ Ogni nodo elabora un vettore di stima dei costi (distanze) verso tutti gli altri nodi nella rete
 - ▣ Il cammino a costo minimo viene calcolato in modo distribuito e iterativo
 - ▣ Algoritmo a vettore distanza (VC, distance-vector algorithms)

Sommario



- Algoritmi di instradamento
 - ▣ **Algoritmo di Dijkstra**
 - ▣ Algoritmo vettore distanza

Algoritmo d'instradamento a stato del collegamento (LS)

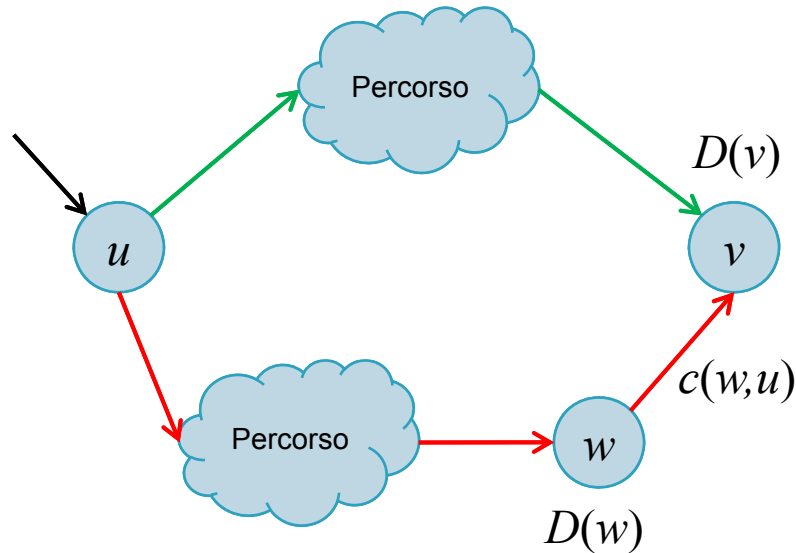
- L'**algoritmo di Dijkstra** prende in input la topologia di rete e tutti i costi dei collegamenti
- Calcola il cammino a costo minimo da un nodo origine u a tutti gli altri nodi della rete
- Attraverso il "link-state broadcast" tutti i nodi dispongono delle stesse informazioni e l'output sarà lo stesso per tutti i nodi
 - Per ogni nodo avremo una tabella d'inoltro

Algoritmo di Dijkstra: un po' di notazione

- L'algoritmo è iterativo: dopo la k -esima iterazione i cammini a costo minimo sono noti a k nodi di destinazione
- $c(x,y)$: costo dei collegamenti dal nodo x al nodo y ; $= \infty$ se non sono adiacenti
- $D(v)$: costo del cammino dal nodo origine u alla destinazione v per quanto riguarda l'iterazione corrente
- $p(v)$: immediato predecessore di v lungo il cammino
- N' : sottoinsieme di nodi per cui il cammino a costo minimo dall'origine è definitivamente noto

Operazione di rilassamento

- Durante l'interazione utilizzeremo la seguente operazione di rilassamento da applicare ai vertici v adiacenti al vertice appena esplorato w



Ad ogni interazione si può scegliere come arrivare al nodo v tra due percorsi con i seguenti costi:

1. $D(v)$
2. $D(w) + c(w,v)$

Quale considero?

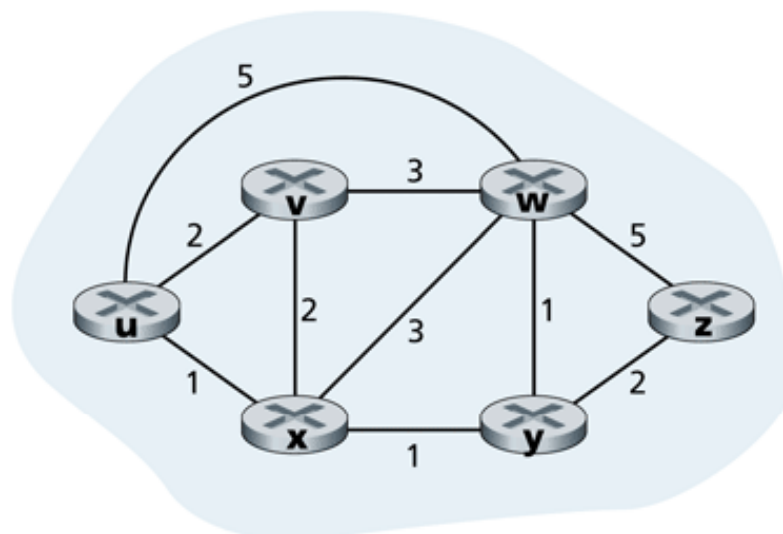
$$D(v) = \min \{D(v), D(w) + c(w,v)\}$$

Algoritmo di Dijkstra

```
1  Inizializzazione:
2    N' = {u}
3    per tutti i nodi v
4      se v è adiacente a u
5        allora D(v) = c(u,v)
6        altrimenti D(v) = ∞
7
8  Ciclo
9    determina un w non in N' tale che D(w) sia minimo
10   aggiungi w a N'
11   aggiorna D(v) per ciascun nodo v adiacente a w e non in N' :
12     D(v) = min( D(v), D(w) + c(w,v) )
13   /* il nuovo costo verso v è il vecchio costo verso v oppure
14     il costo del cammino minimo noto verso w più il costo da w a v
15   */
15  Finché N' = N
```

Algoritmo di Dijkstra: esempio

Passo	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u		∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Sommario



- Algoritmi di instradamento
 - ▣ Algoritmo di Dijkstra
 - ▣ **Algoritmo vettore distanza**

Algoritmo d'instradamento con vettore distanza (DV)

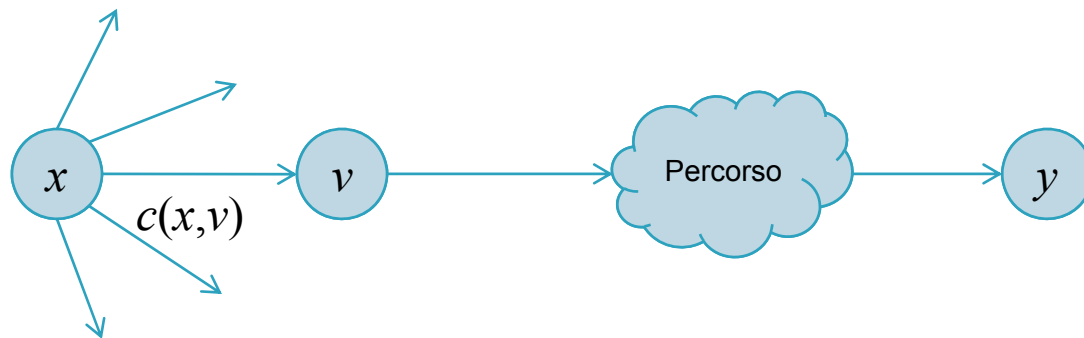
- L'algoritmo con vettore delle distanza ha le seguenti proprietà
 - ▣ *Distribuito*: ogni nodo riceve parte dell'informazione dai suoi vicini
 - ▣ *Iterativo*: il processo si ripete fino a quando non avviene un ulteriore scambio di informazioni con i vicini
 - ▣ *Asincrono*: i passi eseguiti all'interno di un nodo non sono sincronizzati con i passi degli altri nodi

Algoritmo d'instradamento con vettore distanza (DV)

- Definiamo $d_x(y) :=$ il costo del percorso a costo minimo dal nodo x al nodo y
- La Formula di Bellman-Ford definisce

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

dove \min_v riguarda tutti i vicini di x



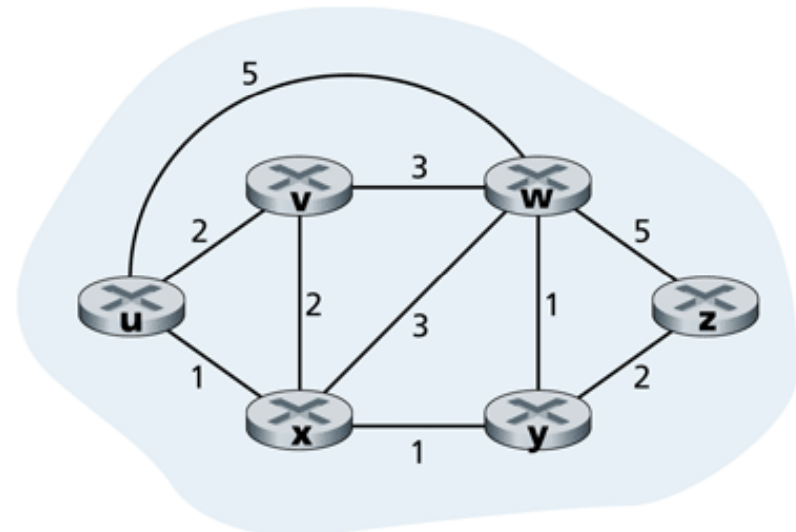
Formula di Bellman-Ford: esempio

- Sorgente u , destinazione z , da cui:

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

- L'equazione di Bellman-Ford definisce:

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



Algoritmo con vettore distanza

- $D_x(y)$ = stima del costo del percorso a costo minimo da se stesso al nodo y
- Vettore distanza: $\mathbf{D}_x = [D_x(y): y \in N]$

- Ogni nodo x mantiene le seguenti informazioni:
 - Il costo $c(x, v)$ per ciascun vicino v di x
 - Il vettore distanza $\mathbf{D}_x = [D_x(y): y \in N]$
 - I vettori distanza di ciascuno dei suoi vicini, ovvero per ciascun vicino v , x mantiene $\mathbf{D}_v = [D_v(y): y \in N]$

Algoritmo con vettore distanza: idea di base

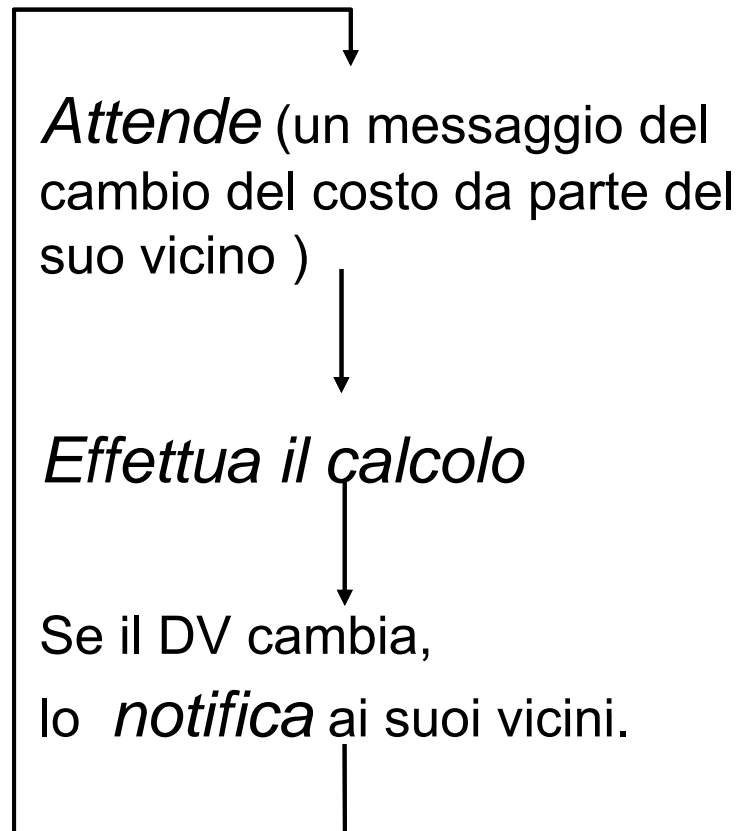
- Ogni nodo invia una copia del proprio vettore distanza a ciascuno dei suoi vicini
- Quando un nodo x riceve da qualcuno dei suoi vicini un nuovo vettore distanza D_v , lo salva e usa la formula BF per aggiornare il proprio vettore distanza come segue

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

per ciascun nodo y in N

- Finché tutti i nodi continuano a cambiare i propri D_v in maniera asincrona, ciascuna stima dei costi $D_x(y)$ converge a $d_x(y)$

Algoritmo con vettore distanza:passi di ogni nodo



Algoritmo con vettore distanza

Per ogni nodo x

1 **Inizializzazione**

2 per tutte le destinazioni y in N

3 $D_x(y) = c(x,y)$ /*se y non è adiacente, allora $c(x,y)=\infty$ */

4 per ciascun vicino w

5 $D_w(y) =$ per tutte le destinazioni y in N

6 per ciascun vicino w

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

8

9 **ciclo**

10 **attendi** {un messaggio del cambio del costo da parte del suo
11 vicino}

12 per ogni y in N :

13 $D_x(y) = \min_v \{c(x,v) + D_v(y)\}$

14

15 **se** $D_x(y)$ è cambiato per qualche destinazione y

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

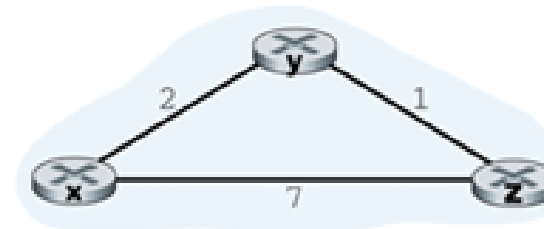
17 **continua**

Algoritmo con vettore distanza: esempio

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

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

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



- Il nodo y invia il vettore $D_y=[2,0,1]$ ad x (ed a z)
- Il nodo z invia il vettore $D_z=[7,1,0]$ ad x (ed a y)
- Il nodo x è in grado di calcolare il suo nuovo vettore delle distanze D_x

Vettore delle distanze D_x

$$D_x(x) = 0$$

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

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

Confronto tra gli algoritmi LS e DV

Complessità dei messaggi:

- LS: con n nodi, E collegamenti, implica l'invio di $O(nE)$ messaggi
- DV: richiede scambi tra nodi adiacenti
 - ▣ Il tempo di convergenza può variare

Velocità di convergenza:

- LS: l'algoritmo $O(n^2)$ richiede $O(nE)$ messaggi
 - ▣ Ci possono essere oscillazioni di velocità
- DV: può convergere lentamente
 - ▣ Può presentare cicli d'instradamento
 - ▣ Può presentare il problema del conteggio all'infinito

Robustezza: cosa avviene se un router funziona male?

- LS:
 - ▣ Un router può comunicare via broadcast un costo sbagliato per uno dei suoi collegamenti connessi (ma non per altri)
 - ▣ I nodi si occupano di calcolare soltanto le proprie tabelle
- DV:
 - ▣ Un nodo può comunicare cammini a costo minimo errati a tutte le destinazioni
 - ▣ La tabella di ciascun nodo può essere usata dagli altri.
 - Un calcolo errato si può diffondere per l'intera rete