

Strato 3: Instradamento

Gruppo Reti TLC
nome.cognome@polito.it
<http://www.telematica.polito.it/>

Copyright

Quest'opera è protetta dalla licenza *Creative Commons NoDerivs-NonCommercial*. Per vedere una copia di questa licenza, consultare:
<http://creativecommons.org/licenses/nd-nc/1.0/>
oppure inviare una lettera a:
Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

This work is licensed under the *Creative Commons NoDerivs-NonCommercial* License. To view a copy of this license, visit:
<http://creativecommons.org/licenses/nd-nc/1.0/>
or send a letter to
Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Funzioni strato rete

- Instradamento (routing)
 - Effettuato consultando tabelle di instradamento
 - per ogni PDU in rete datagram
 - per ogni connessione in rete a circuito virtuale
 - Tabelle di instradamento contengono informazioni tipo
 - per ogni destinazione next-hop (prossimo router)
 - Tre elementi
 - Protocolli di instradamento (routing protocols)
 - Algoritmi di instradamento (routing algorithms)
 - Procedure di inoltra pacchetti (forwarding)

Funzioni strato rete

- Indirizzamento
 - Indirizzi univoci
 - Risoluzione indirizzi (mapping)
- Tariffazione
 - su rete pubblica
- Controllo di congestione
 - Pre-allocazione memorie
 - Scarto pacchetti
 - Invio segnali di congestione

Instradamento

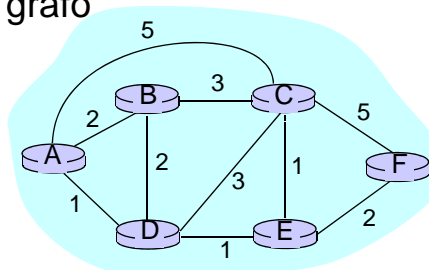
- *Protocollo di instradamento:*
 - definizione delle modalità di scambio di informazioni sullo stato della rete al fine di costruire tabelle di instradamento
- *Algoritmo di instradamento:*
 - operazioni necessarie per scegliere il percorso verso la destinazione, date le informazioni sullo stato della rete
 - crea tabelle di instradamento
- *Procedura di forwarding:*
 - operazioni necessarie per instradare i singoli pacchetti verso la corretta porta di uscita
 - usa tabelle per inoltrare pacchetti

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 5

Algoritmi di instradamento

- Obiettivo degli *algoritmi di instradamento:*
 - determinare un “buon” percorso (sequenza di link o nodi) nella rete da nodo sorgente a nodo destinazione
 - per semplicità si utilizza un solo identificativo per ogni nodo, che rappresenta un aggregato di sorgenti/destinazioni
- Si trasforma la topologia in un grafo
 - nodi sono vertici
 - link fisici sono archi
- Si assegnano costi agli archi
- “Buon” percorso:
 - percorso a costo minimo

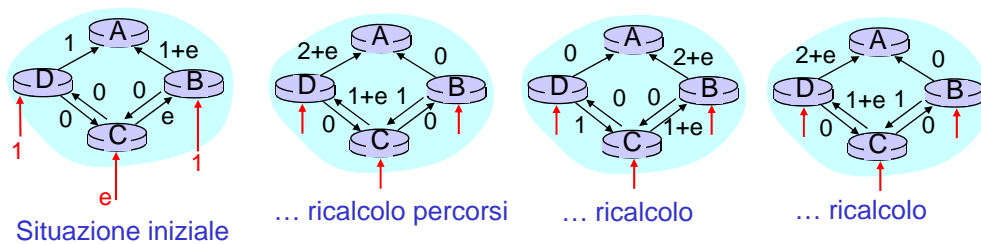


Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 6

Costo

- Distanza, ritardo, euro, livello di congestione
- Statico o dipendente dallo stato della rete
 - Influenza le politiche di aggiornamento
- Possibili oscillazioni:
 - Esempio: costo canali legato al carico trasportato



Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 7

Algoritmi di instradamento: esempi

- Semplici algoritmi senza necessità di coordinamento da parte dei nodi
 - Random
 - Scelgo a caso una porta di uscita
 - Flooding
 - Instrado verso tutte le porte disponibili
 - Deflessione o hot potato
 - Su topologie regolari
 - Instrado verso la porta corretta, se libera. Se occupata, instrado verso una altra porta libera
- Algoritmi complessi per il calcolo del percorso “ottimo”
 - Diversa classificazione

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 8

Classificazione algoritmi di instradamento

- **Calcolo percorso**
 - **Centralizzato:**
 - Un nodo si occupa di raccogliere l'informazione da tutti gli altri nodi
 - Calcola i percorsi
 - Ridistribuisce il risultato del calcolo a tutti gli altri nodi
 - **Distribuito:**
 - Tutti i nodi si scambiano informazione tra loro (utilizzando protocolli di instradamento)
 - Calcolano i percorsi (indipendentemente o in base a quanto fatto dai nodi adiacenti)

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 9

Classificazione algoritmi di routing

- **Centralizzato:**
 - **Vantaggi:**
 - Possibile usare percorsi algoritmi e metriche complesse
 - Tutti i nodi usano piano di instradamento coerente
 - **Svantaggi:**
 - Sensibile al guasto nodo centrale
 - Scambio informazione da/verso nodo centralizzato genera congestione
- **Distribuito:**
 - **Vantaggi:**
 - Robusto ai guasti
 - Scambio informazione uniforme su tutta la rete
 - **Svantaggi:**
 - Richiede intelligenza nei nodi
 - Scambio informazione parziale/errata porta a incongruenze nell'instradamento

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 10

Classificazione algoritmi di routing

- Algoritmi distribuiti - informazione:
 - Globale:
 - Tutti i nodi conoscono la topologia completa, compresi i costi dei canali
 - Scambio informazione tra tutti i nodi
 - Algoritmi link state
 - Parziale:
 - I nodi conoscono i nodi cui sono fisicamente collegati ed i costi dei canali cui sono collegati
 - Scambio di informazione solo con i nodi adiacenti
 - Algoritmi distance vector

Algoritmi Link-State

- Ogni nodo invia informazioni di costo (stato) dei soli suoi canali in (multi)broadcast a tutti gli altri nodi della rete
- Tutti nodi si costruiscono topologia della rete e conoscono i costi di tutti gli archi
- Data la topologia, ogni nodo calcola i percorsi a minimo costo verso tutti gli altri nodi
 - Si ottengono tabelle di routing per questo nodo
- Algoritmo di Dijkstra: usato per determinare cammini ottimi
 - Algoritmo iterativo: dopo k iterazioni si ottengono i cammini a costo minimo per k destinazioni
 - Funziona solo con costi positivi

Algoritmo di Dijkstra

Notazione:

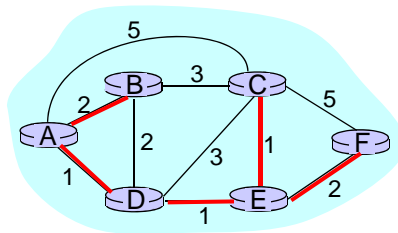
- $c(i,j)$: costo del canale dal nodo i al nodo j
– Infinito se nodo non collegato direttamente al canale
- $D(n)$: costo corrente del percorso migliore da sorgente alla destinazione n
- $p(n)$: nodo che precede n nel percorso da sorgente a destinazione n
- N : insieme di nodi per cui il cammino ottimo è noto

Algoritmo di Dijkstra

- 1 **Inizializzazione (nodo A):**
- 2 $N = \{A\}$
- 3 per tutti i nodi $n \notin N$
- 4 if n adiacente ad A
- 5 then $D(n) = c(A,n)$, $p(n) = A$
- 6 else $D(n) = \text{infinito}$
- 8 **repeat**
- 9 trova nodo $w \notin N$ tale per cui $D(w)$ è minimo
- 10 aggiungi w ad N
- 11 aggiorna $D(n)$ per tutti gli n adiacenti a w , $n \notin N$:
- 12 if ($D(n) > D(w) + c(w,n)$)
- 13 then $D(n) = D(w) + c(w,n)$, $p(n) = w$
- 13 /* il nuovo costo verso n è o il vecchio costo verso n o il
- 14 cammino a minimo costo verso w più costo da w a n */
- 15 **until tutti i nodi in N**

Dijkstra: esempio

Step	start N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
→ 0	A	2,A	5,A	1,A	infinity	infinity
→ 1	AD	2,A	4,D		2,D	infinity
→ 2	ADE	2,A	3,E			4,E
→ 3	ADEB		3,E			4,E
→ 4	ADEBC					4,E
5	ADEBCF					



	Prossimo nodo, costo
B	B,2
C	D,3
D	D,1
E	D,2
F	D,4

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 15

Algoritmo di Dijkstra: proprietà

- Complessità con M nodi
- Ogni iterazione:
 - Controllo tutti i nodi $w \notin N$
 - Aggiungo w a distanza minima
- $M*(M+1)/2$ confronti: $O(M^2)$
- Esistono implementazioni migliori: $O(M \log(M))$

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 16

Algoritmi Distance Vector

- Algoritmi iterativi:
 - continuano fino a quando i nodi non scambiano più informazioni
- Termina in modo autonomo:
 - nessun segnale esplicito di fine algoritmo
- Distribuito:
 - ogni nodo comunica solo con nodi adiacenti

Distance Vector

- Ogni nodo scambia periodicamente con i vicini diretti un vettore contenente:
 - le destinazioni che può raggiungere
 - la distanza dalle destinazioni misurata in costo (ad esempio: numero nodi da attraversare compreso se stesso)
- Il nodo che riceve il vettore lo confronta con la propria RT (Routing Table, tabella di instradamento) ed effettua modifiche:
 - aggiunge nuove destinazioni
 - cambia instradamenti se nuovi sono più brevi
 - modifica costi se usa nodo adiacente come miglior scelta

Distance Vector

- Vantaggi
 - facile da implementare
- Problemi:
 - lento a convergere
 - propaga errori di routing
 - non molto scalabile (le dimensioni dei messaggi scambiati dai nodo crescono al crescere della rete)

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 19

Distance Vector

Implementazione

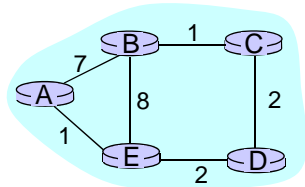
- Struttura dati: tabella distanze
- Ogni nodo possiede la propria
 - Una riga per ogni possibile destinazione
 - Una colonna per ogni nodo adiacente
- Esempio: nel nodo X, per la destinazione Y attraverso nodo adiacente Z:

$$D^X(Y,Z) = c(X,Z) + \min_w \{D^Z(Y,w)\}$$

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 20

Tabella distanze: esempio



$$D^E(C,D) = c(E,D) + \min_w \{D^D(C,w)\} = 2+2 = 4$$

$$D^E(A,D) = c(E,D) + \min_w \{D^D(A,w)\} = 2+3 = 5 \text{ anello!}$$

$$D^E(A,B) = c(E,B) + \min_w \{D^B(A,w)\} = 8+6 = 14 \text{ anello!}$$

		Costo verso destinazione attraverso nodo		
$D^E()$		A	B	D
destinazione	A	1	14	5
	B	7	8	5
	C	6	9	4
	D	4	11	2

Tabella di instradamento a partire da tabella distanze

		Costo verso destinazione attraverso nodo			Prossimo nodo, costo	
$D^E()$		A	B	D		
destinazione	A	1	14	5	A	A,1
	B	7	8	5	D	D,5
	C	6	9	4	D	D,4
	D	4	11	2	D	D,2

Tabella distanze → Tabella di Routing

Instradamento Distance Vector

- Iterativo, asincrono: una iterazione (locale al nodo) causata da:
 - modifica costo canale a cui nodo collegato
 - messaggio ricevuto da nodo adiacente, che causa modifica del cammino ottimo
- Distribuito:
 - ogni nodo avvisa i vicini solo quando il suo cammino migliore verso una certa destinazione è cambiato
 - i vicini avviseranno a loro volta nodi vicini, se necessario

Instradamento Distance Vector

- Ogni nodo esegue un loop infinito
 - aspetta
 - modifica costo canale locale oppure
 - messaggio da nodo adiacente
 - ricalcola tabella distanze
 - se percorso migliore verso qualche destinazione cambiato, avvisa i vicini

Algoritmo Distance Vector

Ad ogni nodo X:

- 1 Inizializzazione:
- 2 per tutti i nodi adiacenti v:
- 3 $D^X(*,v) = \text{infinito}$ /* l'operatore * significa "per ogni riga" */
- 4 $D^X(v,v) = c(X,v)$
- 5 per tutte le destinazioni, y
- 6 invia $\min_w D^X(y,w)$ verso ogni nodo adiacente /* w sono tutti i vicini di X */
- 7

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 25

Algoritmo Distance Vector

```

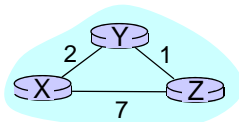
8 loop
9  wait (until I see a link cost change to neighbor V
10    or until I receive update from neighbor V)
11
12  if (c(X,V) changes by d)
13    /* change cost to all dest's via neighbor V by d */
14    /* note: d could be positive or negative */
15    for all destinations y:  $D^X(y,V) = D^X(y,V) + d$ 
16
17  else if (update received from V wrt destination Y)
18    /* shortest path from V to some Y has changed */
19    /* V has sent a new value for its  $\min_w D^V(Y,w)$  */
20    /* call this received new value is "newval" */
21    for the single destination y:  $D^X(Y,V) = c(X,V) + \text{newval}$ 
22
23  if we have a new  $\min_w D^X(Y,w)$  for any destination Y
24    send new value of  $\min_w D^X(Y,w)$  to all neighbors
25
26 forever

```

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 26

Distance Vector: esempio



		cost via	
		Y	Z
d e s t	Y	2	∞
	Z	∞	7

		cost via	
		X	Z
d e s t	X	2	∞
	Z	∞	1

		cost via	
		X	Y
d e s t	X	7	∞
	Y	∞	1

		cost via	
		Y	Z
d e s t	Y	2	8
	Z	3	7

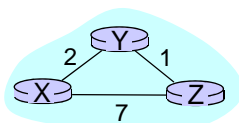
$$D^X(Y,Z) = c(X,Z) + \min_w \{D^Z(Y,w)\}$$

$$= 7 + 1 = 8$$

$$D^X(Z,Y) = c(X,Y) + \min_w \{D^Y(Z,w)\}$$

$$= 2 + 1 = 3$$

Distance Vector: esempio



		cost via	
		Y	Z
d e s t	Y	2	∞
	Z	∞	7

		cost via	
		X	Z
d e s t	X	2	∞
	Z	∞	1

		cost via	
		X	Y
d e s t	X	7	∞
	Y	∞	1

		cost via	
		Y	Z
d e s t	Y	2	8
	Z	3	7

		cost via	
		X	Z
d e s t	X	2	8
	Z	9	1

		cost via	
		X	Y
d e s t	X	7	3
	Y	9	1

		cost via	
		Y	Z
d e s t	Y		
	Z		

		cost via	
		X	Z
d e s t	X		
	Z		

		cost via	
		X	Y
d e s t	X		
	Y		

Algoritmo DV: modifica costo canale

- Nodo riconosce modifica costo canale
- Modifica tabella distanze (passo 15)
- Se modifica implica modifica del cammino migliore avvisa nodi adiacenti (passi 23,24)

“good news travels fast”

	D ^Y X Z				D ^Y X Z				D ^Y X Z				D ^Y X Z		
	x	(4)	6		x	(1)	6		x	(1)	6		x	(1)	3
	D ^Z X Y				D ^Z X Y				D ^Z X Y				D ^Z X Y		
	x	50	(5)		x	50	(5)		x	50	(2)		x	50	(2)

time
 t_0
 t_1
 t_2
→

c(X,Y) change

algoritmo termina

Copyright Gruppo Reti – Politecnico di Torino INTRODUZIONE ALLE RETI TELEMATICHE - 29

Algoritmo DV: modifica costo canale

Modifica costo canale:

- good news travels fast
- bad news travels slow - problema del “count to infinity”!

“good news travels fast”

	D ^Y X Z				D ^Y X Z				D ^Y X Z				D ^Y X Z		
	x	(4)	6		x	(6)	60		x	(6)	60		x	(8)	60
	D ^Z X Y				D ^Z X Y				D ^Z X Y				D ^Z X Y		
	x	50	(5)		x	50	(5)		x	50	(7)		x	50	(7)

time
 t_0
 t_1
 t_2
 t_3
 t_4
→

c(X,Y) change

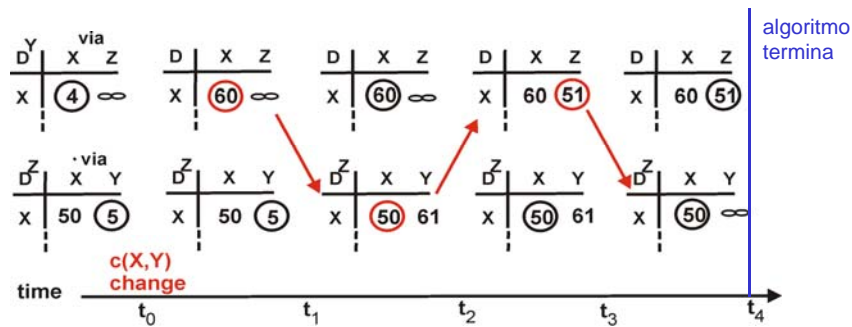
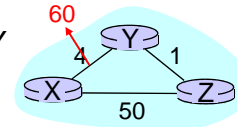
algoritmo prosegue

Copyright Gruppo Reti – Politecnico di Torino INTRODUZIONE ALLE RETI TELEMATICHE - 30

Distance Vector: poisoned reverse

Se Z instrada via Y per raggiungere X:

- Z comunica ad Y sua distanza verso X è infinito (Y non instraderà verso X passando da Z)
- non risolve il problema completamente



Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 31

Confronto tra algoritmi LS e DV

- Complessità messaggi: con M nodi, E canali per nodo
 - LS: ogni nodo invia O(M) messaggi, ciascuno lungo O(E)
 - DV: ogni messaggio contiene tutte le destinazioni O(M), ed è mandato a O(E) vicini

$$O(E M)$$

- Velocità di convergenza
 - LS: ogni volta che un link state è propagato, ho nuova topologia: convergenza immediata
 - DV: scelte nodo dipendono da scelte nodi vicini; si richiedono più scambi di messaggi: tempo di convergenza variabile

Copyright Gruppo Reti – Politecnico di Torino

INTRODUZIONE ALLE RETI TELEMATICHE - 32

Confronto tra algoritmi LS e DV

- Affidabilità: cosa succede se un nodo funziona non correttamente?
- LS: i nodi possono annunciare costi dei canali scorretti
 - Ogni nodo calcola la propria tabella: tutti sbagliano
 - Non si possono creare anelli
 - Al prossimo annuncio tutto si corregge
- DV: i nodi possono annunciare costi dei cammini scorretti
 - Ogni annuncio è usata da tutti i nodi (indirettamente)
 - Gli errori si propagano nella rete
 - Errori di routing creano anelli