Progetto Python: Distance Vector Routing

Jacopo Turchi

 $12\ \mathrm{dicembre}\ 2024$

Indice

1	Inti	roduzione	2			
2	Des	scrizione dei File	3			
	2.1	main.py	3			
	2.2	network_setup.py	_			
	2.3	router_management.py	3			
3	Funzionamento dello Script					
	3.1	Creazione della Rete Iniziale	5			
	3.2	Modifica della Rete	7			
	3.3	Ulteriore Modifica della Rete	10			
4	Cor	nclusione	13			

Introduzione

Ho scelto di seguire la seconda traccia, realizzando uno script Python che simula il protocollo *Distance Vector Routing*. Lo script è composto da tre file principali: main.py, network_setup.py, e router_management.py.

Descrizione dei File

2.1 main.py

Il file main.py è il punto di ingresso dello script. Esso crea una rete iniziale utilizzando le funzioni definite in network_setup.py e simula il protocollo di Distance Vector Routing utilizzando le funzioni definite in router_management.py. Inoltre, modifica dinamicamente la rete e testa il corretto funzionamento in diverse condizioni.

2.2 network_setup.py

Il file network_setup.py contiene funzioni per la creazione e la modifica della rete. Le funzioni principali includono:

- add_node(network, node): Aggiunge un nuovo nodo alla rete.
- add_edge(network, node1, node2, cost): Aggiunge un collegamento tra due nodi con un costo specificato.
- remove_edge(network, node1, node2): Rimuove il collegamento tra due nodi.
- remove_node(network, node): Rimuove un nodo e i suoi collegamenti dalla rete.
- create_network(): Crea una rete iniziale di partenza aggiungendo nodi e collegamenti con costi specificati.

2.3 router_management.py

Il file router_management.py contiene funzioni per la gestione delle tabelle di routing e la simulazione del protocollo Distance Vector. Le funzioni principali includono:

- initialize_routing_table(network): Inizializza le tabelle di routing per ogni nodo nella rete.
- update_routing_table(node, neighbors, routing_table): Aggiorna la tabella di routing di un nodo usando l'algoritmo di Bellman-Ford.
- propagate_updates(network, routing_table): Propaga gli aggiornamenti delle tabelle di routing fino alla convergenza.
- print_routing_tables(routing_table): Stampa le tabelle di routing in formato tabellare.
- simulate_distance_vector_routing(network): Esegue la simulazione del Distance Vector Routing.

Funzionamento dello Script

3.1 Creazione della Rete Iniziale

Lo script inizia creando una rete iniziale con tre nodi (A, B, C) e collegamenti con costi specificati. La rete iniziale è rappresentata nella Figura 3.1.

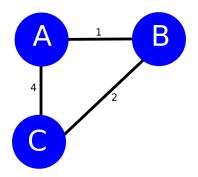


Figura 3.1: Rete iniziale con nodi A, B, C.

Successivamente, esegue la simulazione del Distance Vector Routing, che consiste nell'inizializzare le tabelle di routing e propagare gli aggiornamenti fino alla convergenza La tabella di routing iniziale è mostrata nella Figura 3.2. La tabella finale è mostrata nella Figura 3.3

Iteration 1: Routing table for	r A:				
Destination	Cost	Next Hop			
A		-====================================			
B	1	B			
i c	4	C			
Routing table for	r B:				
Destination	Cost	Next Hop			
A	1	A			
B	0	B			
į c	2	i c i			
Routing table for C:					
Destination	Cost	Next Hop			
A	3	B			
B	2	B			
i c		C			
+		+			

Figura 3.2: Tabella di routing iniziale per i nodi A, B, C.

Iteration 2: Routing table	for	A:	·		
Destination	ļ	Cost	Next Hop		
A	ļ	0	A		
В	į	1	В [
C	į	3	B		
Routing table	for	B:			
Destination	į	Cost	Next Hop		
A	+ !	1	A		
В	į	0	В [
. c	į	2	C .		
Routing table for C:					
Destination	ļ	Cost	Next Hop		
A		3	B		
B	į	2	B		
C	İ	0	C		
'					

Figura 3.3: Tabella di routing finale per i nodi A, B, C.

3.2 Modifica della Rete

Dopo la prima simulazione, lo script modifica dinamicamente la rete aggiungendo un nuovo nodo (D), collegandolo ai nodi esistenti con costi specificati, e rimuovendo il collegamento tra B e C. La rete modificata è mostrata nella Figura 3.4.

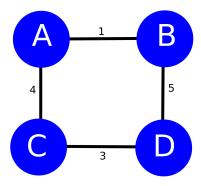


Figura 3.4: Rete modificata con l'aggiunta del nodo D e la rimozione del collegamento tra B e C.

La simulazione viene quindi rieseguita per riflettere i cambiamenti. La tabella di routing iniziale è mostrata nella Figura 3.5. La tabella finale è mostrata nella Figura 3.6

Routing table fo				
+ Destination	-+ Cost	 Next Hop		
+=====================================	:+======- 0	+======		
+ В	1	+ В		
C	4	C		
D	inf	- -		
Routing table for B:				
Destination	Cost	Next Hop		
т А	1	A		
B	[0	B		
, с		A		
		<u></u>		
D	†5	D		
D D 	· +	D		
+	· +	D		
Routing table fo	i or C:			
Routing table for the control of the	or C: + Cost -+	 Next Hop		
Routing table for the state of	r C: Cost 4	 Next Hop 		
Routing table for the state of	cr C: Cost 4	Next Hop A		
Routing table for the state of	Cr C: Cost 4 5 0	Next Hop A A		
Routing table for the continuity of the continui	Cr C: Cost 4 5 0	Next Hop A A		
Routing table for the continuous	Cost Cost 4 5 0 3	Next Hop		
Routing table for the state of	Cost Cost Cost Cost Cost Cost Cost Cost	Next Hop		
Routing table for the continuity of the continui	Cost Cost Cost Cost Cost Cost Cost Cost Cost	Next Hop A A C D Next Hop		

Figura 3.5: Tabella di routing iniziale per i nodi A, B, C, D.

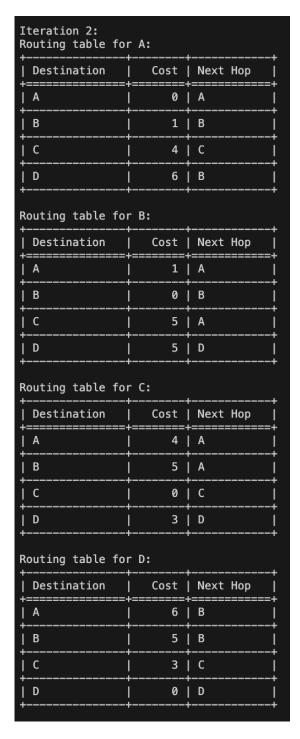


Figura 3.6: Tabella di routing finale per i nodi A, B, C, D.

3.3 Ulteriore Modifica della Rete

Infine, lo script modifica il costo di un collegamento esistente e riesegue la simulazione per aggiornare le tabelle di routing. La rete con il costo modificato è mostrata nella Figura 3.7.

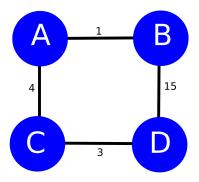


Figura 3.7: Rete con il costo del collegamento tra B e D modificato.

La simulazione viene quindi rieseguita per riflettere i cambiamenti causati dalla modifica del costo dell'arco. La tabella di routing iniziale è mostrata nella Figura 3.8. La tabella finale è mostrata nella Figura 3.9

Destination	Cost	Next Hop
 A	+======================================	+======= A
 В	+ 1	+ В
 C	+ 4	+
D	+ inf	+ -
outing table fo	r R.	+
outing table fo Destination	гв: + Cost	+ Next Hop
-=====================================	+======= 1	+======= A
 В	 0	, + В
 C	5	 A
D 	+	D
	÷	D Next Hop
outing table fo	i r C: +	; +
outing table fo Destination	+ r C: + Cost +======	 Next Hop
outing table fo Destination	r C: 	
Duting table fo Destination A	r C: Cost Cost 4	
Duting table fo	r C: Cost 4 5 0	Next Hop A A C
Destination A B C	r C: Cost 4 5 0	Next Hop A A C
Destination A B C D D D D D D D D D D D D	r C: Cost 	Next Hop
Destination B C Destination B C D Destination	r C: Cost 	Next Hop
Destination A B C D Duting table for the second secon	r C: Cost 4 5 0 3 TD: Cost	Next Hop

Figura 3.8: Tabella di routing iniziale per i nodi A, B, C, D dopo il cambio di costo.

Thoration 2.					
<pre>Iteration 2: Routing table for A: +</pre>					
Destination	Cost	Next Hop			
A	0	A			
B	1	B			
C	4	C			
D	7	C			
Routing table for B:					
Destination	Cost	Next Hop			
A	1	A			
B	0	B			
C	5	A			
D	8	A			
Routing table for	Routing table for C:				
Destination	Cost	Next Hop			
A	4	+ A			
B	5	A			
j c	0	C			
D	3	D			
Routing table for D:					
Destination	Cost	Next Hop			
Α	7	C			
B	8	C			
C	3	C			
D	0	D			
+		+			

Figura 3.9: Tabella di routing finale per i nodi A, B, C, D dopo il cambio di costo.

Conclusione

Questo script fornisce una semplice ma efficace simulazione del protocollo di routing Distance Vector, permettendo di osservare come le tabelle di routing vengono aggiornate in risposta ai cambiamenti nella topologia della rete. Le funzioni modulari e ben definite rendono lo script facilmente estendibile per ulteriori esperimenti e simulazioni.