

GRAFI E RETI

→ DISTANZA FRA DUE ROUTER = RICERCA CAMMINO MINIMO SU UN GRAFO

• MINIMA DISTANZA : SU GRAFO NON PESATO, n° MINIMO DI ARCHI o HOP

• MINIMA LUNGHEZZA : SU GRAFO PESATO

GRAFI

[UN GRAFO È UNA COPPIA ORDINATA DI INSIEMI $G = (V, E)$,]
CON V INSIEME DEI NODI ed E INSIEME DEGLI ARCHI.]

(N.B. $E \subseteq V \times V$)

$$\text{e } |E| \leq |V|^2$$

⁺ CARDINALITÀ INSIEME

↓
COPPIE DI V

ALTERNATIVA: $G = (V, E, f)$

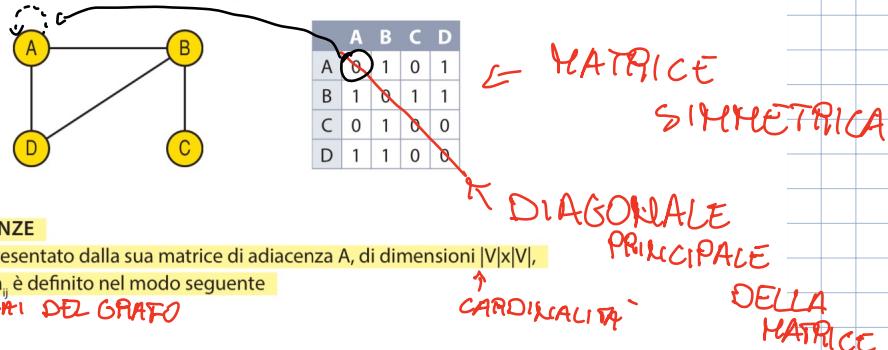
↑

FUNZIONE CHE ASSOCIA UN ARCO $e \in E$
IN DUE VERTICI u e v ($u \in V$ e $v \in V$)

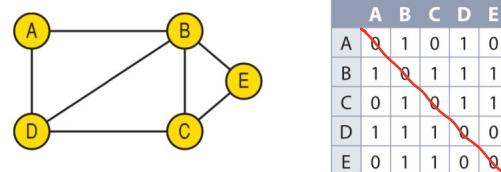
Rappresentazione dei grafi

Tra le possibili rappresentazioni dei grafi ricordiamo la **matrice delle adiacenze**: essendo la matrice una struttura con dimensionamento statico, questa rappresentazione si utilizza nel caso in cui il grafo si riferisca a situazioni dove il numero dei vertici rimane pressoché invariato.

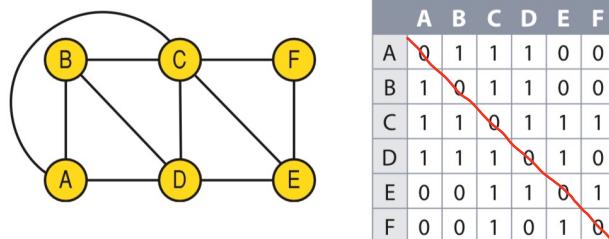
Nella **matrice delle adiacenze** viene riportato, sia in ascissa sia in ordinata, l'elenco dei vertici: nelle celle di incrocio tra due nodi si indica con 1 la presenza di connessione e con 0 l'assenza. Osserviamo che, se il grafo è non orientato, la matrice presenta una simmetria diagonale.



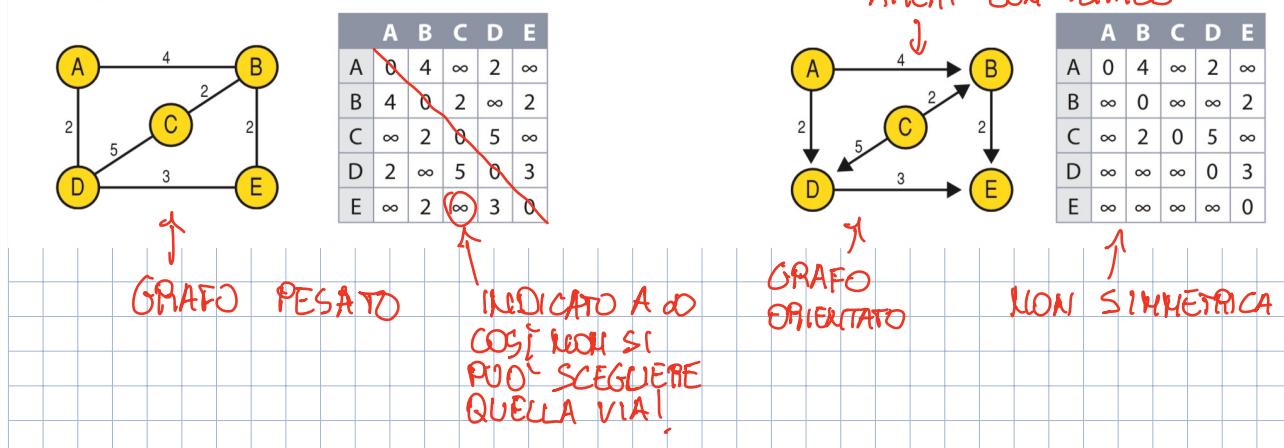
Vediamo un secondo esempio:



Vediamo un terzo esempio:



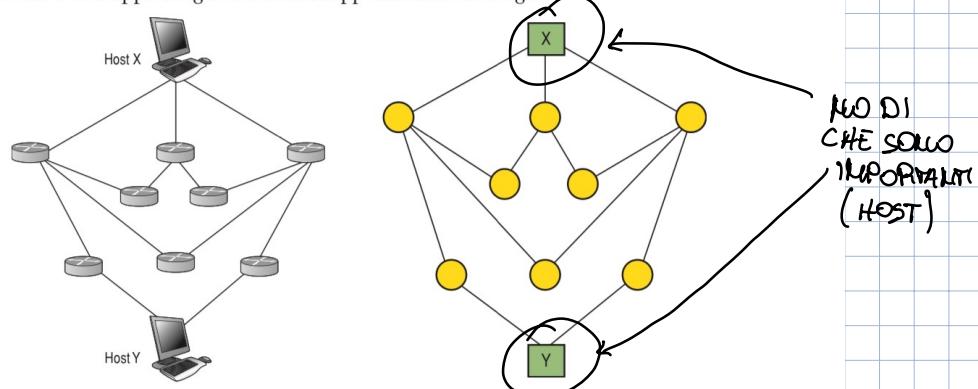
L'utilizzo delle matrici si presta alla rappresentazione dei **grafi pesati**: basta sostituire nelle celle della matrice delle adiacenze al valore 1 il valore del peso tra il vertice rappresentato in ordinata e il vertice adiacente in ascissa, indicando con 0 o ∞ l'assenza di connessione.



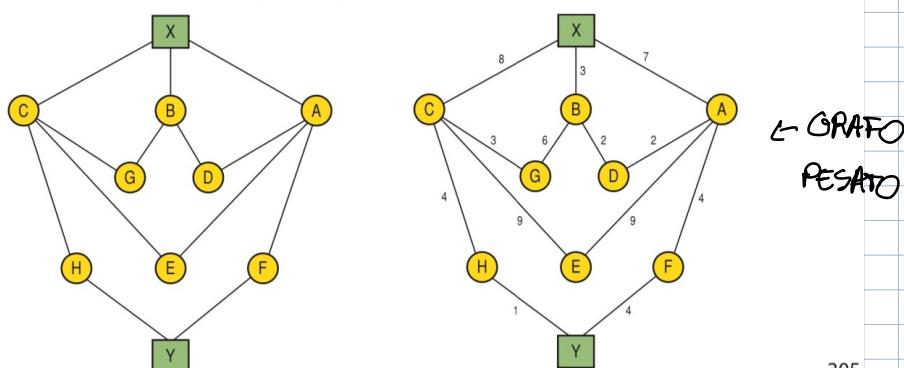
Grafi e reti

PARALLELSIMO DELLA RETE COME UN GRAFO

Vediamo ora come utilizzare i grafi per rappresentare le reti. Supponiamo di dover trasmettere un messaggio dall'host X verso l'host Y che appartengono alla rete rappresentata nella figura:



Come primo passaggio assegniamo un identificatore a ogni router e quindi successivamente li indichiamo come nodi e li collegiamo tra loro come nella seguente figura.

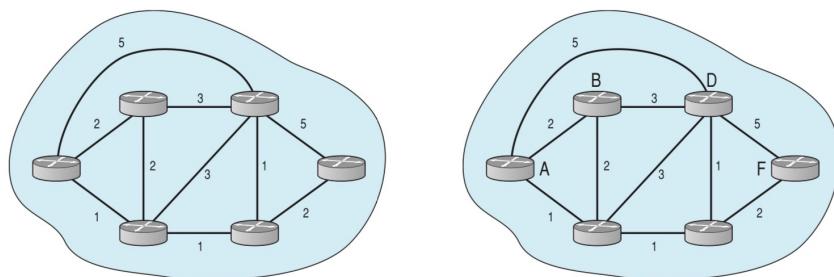


205

Costo metrica

A questo punto aggiungiamo i pesi agli archi e costruiamo la matrice delle adiacenze, dove il peso tra due router è il tempo (oppure il ritardo) per trasmettere un pacchetto tra di essi.

Vediamo un secondo esempio. Alla rete rappresentata nella figura assegniamo una etichetta a ogni router:



e quindi la tabella delle adiacenze.

| | A | B | C | D | E | F |
|---|----------|----------|----------|---|----------|----------|
| A | 0 | 2 | 1 | 5 | ∞ | ∞ |
| B | 2 | 0 | 2 | 3 | ∞ | ∞ |
| C | 1 | 2 | 0 | 3 | 1 | ∞ |
| D | 5 | 3 | 3 | 0 | 1 | 5 |
| E | ∞ | ∞ | 1 | 1 | 0 | 2 |
| F | ∞ | ∞ | ∞ | 5 | 2 | 0 |

Costo infinito perché non c'è una connessione

RICERCA DEL PERCORSO MINIMO (Shortest Path = sp)

DEF SHORTEST PATH (SP)

$$G = (V, E)$$

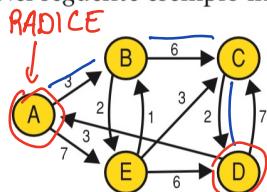
CAMMINO NEL GRAFO

PERCORSO DAL NODO u AL NODO v di V TALE CHE $p = u, v_1, v_2, \dots, v_r$

E' IL CAMMINO CON $w(p)$ MINIMO

PESO DEL CAMMINO

Nel seguente esempio individuiamo su un grafo pesato orientato due cammini tra il nodo A e il nodo D.



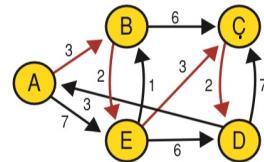
$$p_3(A, B, C, D)$$

$$p_3 = 3 + 6 + 2 = 11$$

FOGLIA
DEL PERCORSO

$$p_1(A \dots D) = (A, E, C, D)$$

Peso di $p_1 = 7 + 3 + 2 = 12$



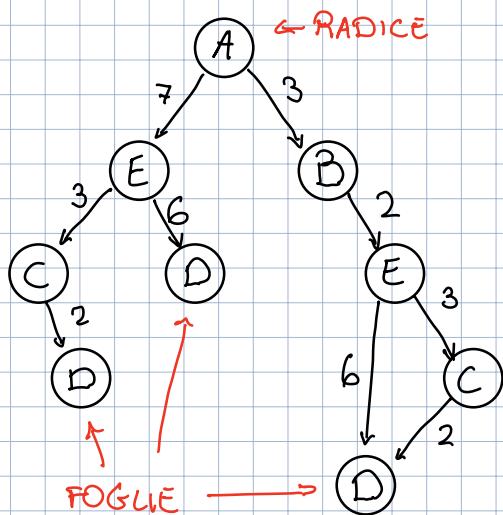
$$p_2(A \dots D) = (A, B, E, C, D)$$

Peso di $p_2 = 3 + 2 + 3 + 2 = 10$

In questo caso il cammino p_2 è il cammino minimo.

COSTO INFERIORE
SE PUR PASSA
PER UN NODO IN
PIÙ!

CAMMINO NEL GRAFO \rightarrow CREA UN ALBERO DI INOLTRO,
(FORWARDING TREE)
 \hookrightarrow ALBERO DI CONSEGNA (DELIVERY TREE)



DISKSTRA

(GRAFO NON ORIENTATO)
 ALGO SU GRAFO CICLICO COM
 PESI NON NEGATIVI)

BELLMAN - FORD

(ALGO SU GRAFO DIRETTO)

PESATO, CON PESI ANCHE NEGATIVI)

ALBERO (TREE)

: GRAFO NON ORIENTATO, CONNESSO E ACICLICO

UNO E
UNO SOLO

ESISTE UN
PERCORSO CHE
VA DA u A v
PER QUALSIASI
 u E v IN V .

NO CICLI /
PERCORSI
CHIUSI

GRAFO

TRASFORMATO

ALBERO

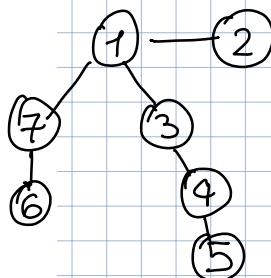
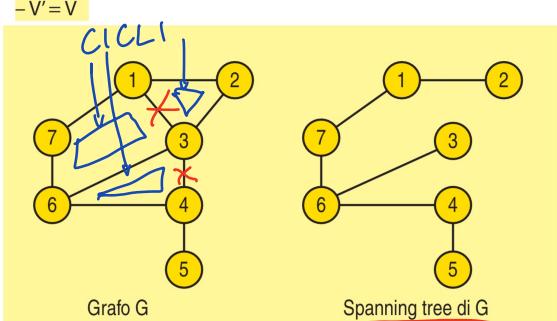
(ELIMINO GLI
ARCHI CHE CREANO → CREO LO SPANNING
CICLI) TREE
(ALBERO DI COPERTURA)

ALBERO DI RICOPRIMENTO

Dato un grafo connesso non orientato $G = (V, E)$, uno **spanning tree** (albero di ricoprimento) di G è un sottografo $G' = (V, T)$ tale che:

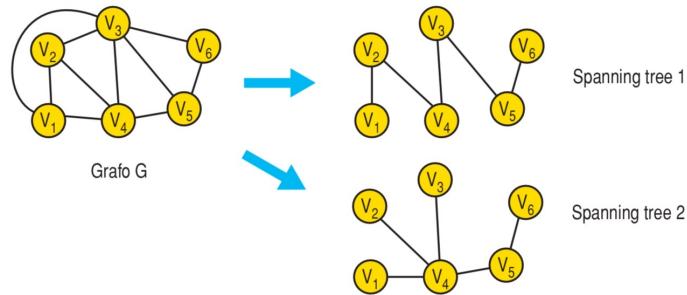
- G' è un albero

- $V' = V$





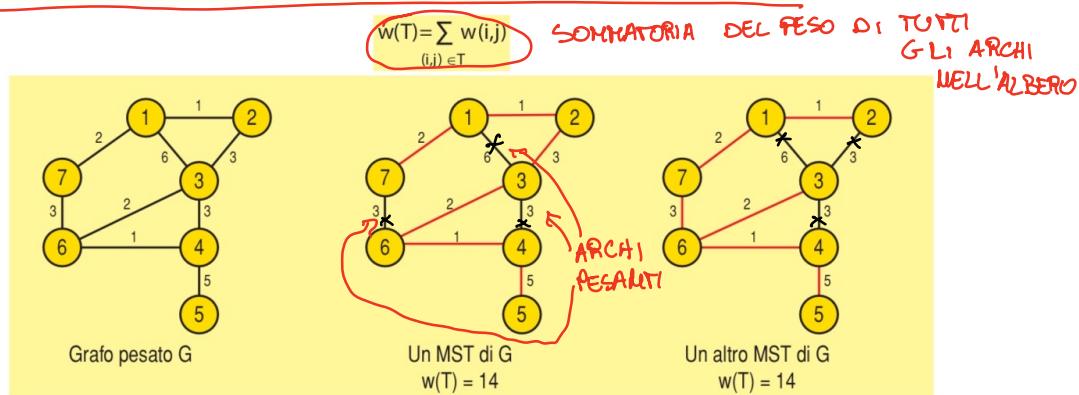
A partire da un grafo è possibile individuare molti alberi ricoprenti e all'aumentare del numero di cicli aumentano anche le differenti combinazioni:



Tra tutti gli alberi ricoprenti individuiamo il **Minimum Spanning Tree (MST)** come l'albero avente somma degli archi minima.

MINIMUM SPANNING TREE

Dato un grafo pesato connesso non orientato $G = (V, E)$, un **minimum spanning tree** (albero di ricoprimento minimo) di G è uno **spanning tree** $G' = (V, T)$ di peso minimo, cioè tale che risulti minimo $w(T)$ dove:



Come è stato riportato nella figura precedente per lo stesso grafo possono esistere diversi **MST**, cioè **Spanning Tree** avanti tutti lo stesso valore (minimo) come somma degli archi.

SPANNING TREE OTTIMO

Definiamo **Spanning Tree Ottimo** lo **Spanning Tree** formato dai cammini a costo minimo (**shortest path**) che uniscono il nodo radice agli altri nodi del grafo (prende anche il nome di **sink tree**).

L'individuazione dei **sink tree** è l'obiettivo di un algoritmo di instradamento in un router e noi analizzeremo a tale scopo, nelle prossime lezioni, l'algoritmo di Dijkstra e di Bellman-Ford.



Sink tree

Sink tree (minimum spanning tree): set of all optimal paths from all sources to a given destination. Can be found using the shortest path algorithm from destination to all sources giving optimal though not necessarily unique route.

ALGO DI ROUTING STATICI

DUE CRITERI ALGO ROUTING

1) INFO SULLA TOPOLOGIA DELLA RETE :

↳ - DISTRIBUITI

- ISOLATI

- CENTRALIZZATI

2) ADATTABILITÀ : - ALGO STATICI (PER CAMBI RARI NELLE RETI)

- ALGO DINAMI (PER CAMBI CONTINUI NELLA RETE)

→ PRIMA DI VEDERE GLI ALGO ABBIANO BISOGNO DI ALCUNI CONCETTI!

• ALBERO DI INSTRADAMENTO

DATO

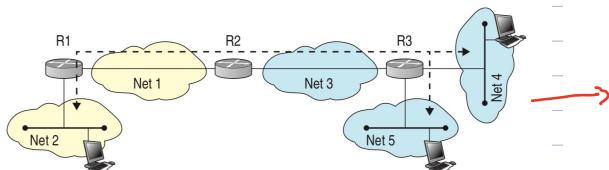
SCHEMA DELLA RETE → "TROVO" IL CAMMINO DAL ROUTER

MITTENTE AL DESTINATARIO

N.B. PER LE RETI SEMPLICI

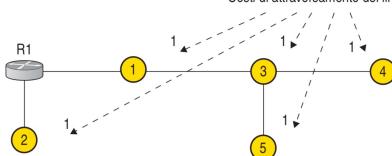
→ PER RETI COMPLESSE USO DI ALGO PER SHORTEST PATH!

Prendiamo come esempio la seguente rete:



dalla rete rappresentata nella figura si ottiene il seguente albero di instradamento

Costi di attraversamento dei link



dove a ogni rete si sostituisce un nodo e a ogni router un arco.

Utilizzando gli alberi di instradamento è inoltre immediato determinare il numero di hop per raggiungere ogni destinazione e quindi compilare la tabella di instradamento.

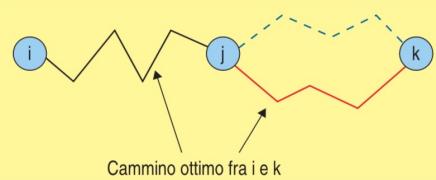


In questo caso il risultato è plausibile perché le connessioni della rete non sono cicliche che porterebbero all'infinito nei percorsi.

• PRINCIPIO DI OTTIMALITÀ → (PER RICERCA DI MST)

PRINCIPIO DI OTTIMALITÀ

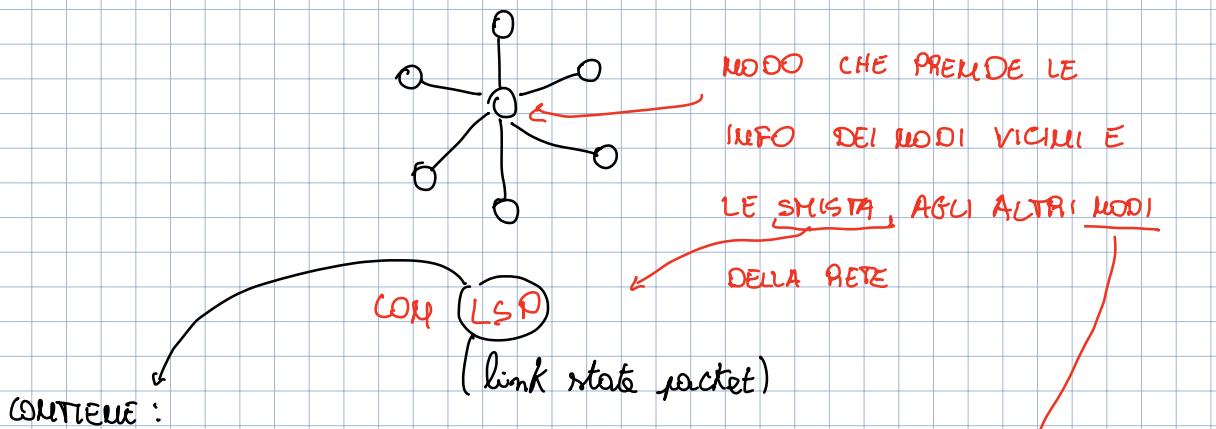
Il principio di ottimalità dice che se il router j è nel cammino ottimo fra i e k, allora anche il cammino ottimo fra j e k è sulla stessa strada:



Dimostriamolo per assurdo, ipotizzando cioè che nell'esempio rappresentato nella figura esista un cammino tra j e k (che nel disegno è tratteggiato in blu) con costo minore di quello indicato in rosso: se così fosse, ci sarebbe un altro cammino fra i e k migliore di quello ottimo (c.v.d.).

• LINK STATE PACKET

ALGO (ROUTING) STATICI



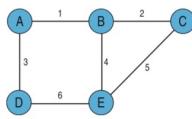
- STATUS LINKS AL ROUTER
- IDENTITÀ DEI LINK CONNESSI
- COSTO LINK
- N° SEQUENZA LSP
- CHECKSUM
- LIFETIME (TTL)

- (1) OGNI NODO HA UN DATABASE CON GLI LSP RICEVUTI E COSTRUISCE I CAMPI MINIMI PER TUTTE LE DESTINAZIONI
- (2)

ESEMPIO

La semplice rete riportata nella figura a lato
è rappresentata dal seguente database:

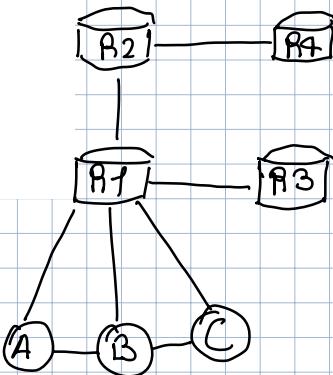
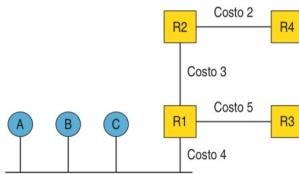
| DA | VERSO | LINK | COSTO | NUMERO DI SEQUENZA |
|----|-------|------|-------|--------------------|
| A | B | 1 | 1 | 1 |
| A | D | 3 | 1 | 1 |
| B | A | 1 | 1 | 1 |
| B | C | 2 | 1 | 1 |
| B | E | 4 | 1 | 1 |
| C | B | 2 | 1 | 1 |
| C | E | 5 | 1 | 1 |
| D | A | 3 | 1 | 1 |
| D | E | 6 | 1 | 1 |
| E | B | 4 | 1 | 1 |
| E | C | 5 | 1 | 1 |
| E | D | 6 | 1 | 1 |



Con questa tabella ogni nodo può calcolare il percorso più breve verso tutti gli altri nodi.

ESEMPIO

Vediamo ora un secondo semplice esempio di come viene realizzato un LS database.



Il router R1 ha sei host direttamente collegati e di ciascuno conosce il peso del collegamento e costruisce la seguente tabella:

| ADIACENZE R1/x | COSTO |
|----------------|-------|
| A | 4 |
| B | 4 |
| C | 4 |
| R1 | 0 |
| R2 | 3 |
| R3 | 5 |

Quindi la invia in flooding su tutti i link del router e i dati vengono memorizzati nei diversi router formando una mappa completa e aggiornata della rete.

Analogamente il router 2 invia un messaggio così composto:

| ADIACENZE R2/x | COSTO |
|----------------|-------|
| R1 | 3 |
| R2 | 0 |
| R4 | 2 |

Lo stesso discorso vale per i router R3 e R4

| ADIACENZE R3/x | COSTO |
|----------------|-------|
| R1 | 5 |
| R3 | 0 |

| ADIACENZE R4/x | COSTO |
|----------------|-------|
| R2 | 2 |
| R4 | 0 |

Ricapitolandoabbiamo:

| | | | | | |
|----|-----|------|------|------|------|
| R1 | A/4 | B/4 | C/4 | R2/3 | R3/5 |
| R2 | | R1/3 | R4/2 | | |
| R3 | | | R1/5 | | |
| R4 | | | | R2/2 | |

Dall'unione di tutte queste informazioni è possibile costruire la seguente tabella delle adiacenze:

| | A | B | C | R1 | R2 | R3 | R4 |
|----|---|---|---|----|----|----|----|
| A | 0 | 0 | 0 | 4 | | | |
| B | 0 | 0 | 2 | 4 | | | 2 |
| C | 0 | 2 | 0 | 4 | | | |
| R1 | 4 | 4 | 4 | 0 | 3 | | |
| R2 | | | | 3 | 0 | | |
| R3 | 5 | | | | | 0 | |
| R4 | | 2 | | | | 5 | 0 |