

Unità didattica: algoritmi di visita di un grafo

[4-T]

Titolo: Algoritmi di visita di una struttura reticolare

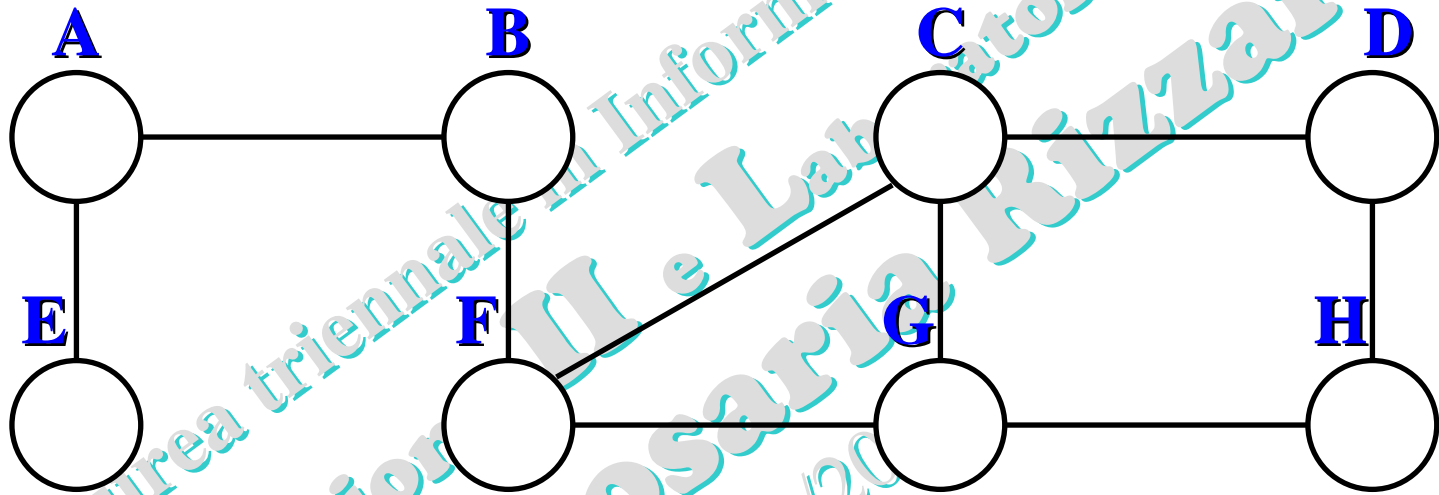
Argomenti trattati:

- ✓ Visita di un grafo: algoritmo di visita in ampiezza (Breadth First Search)

Prerequisiti richiesti: coda, rappresentazione di un grafo

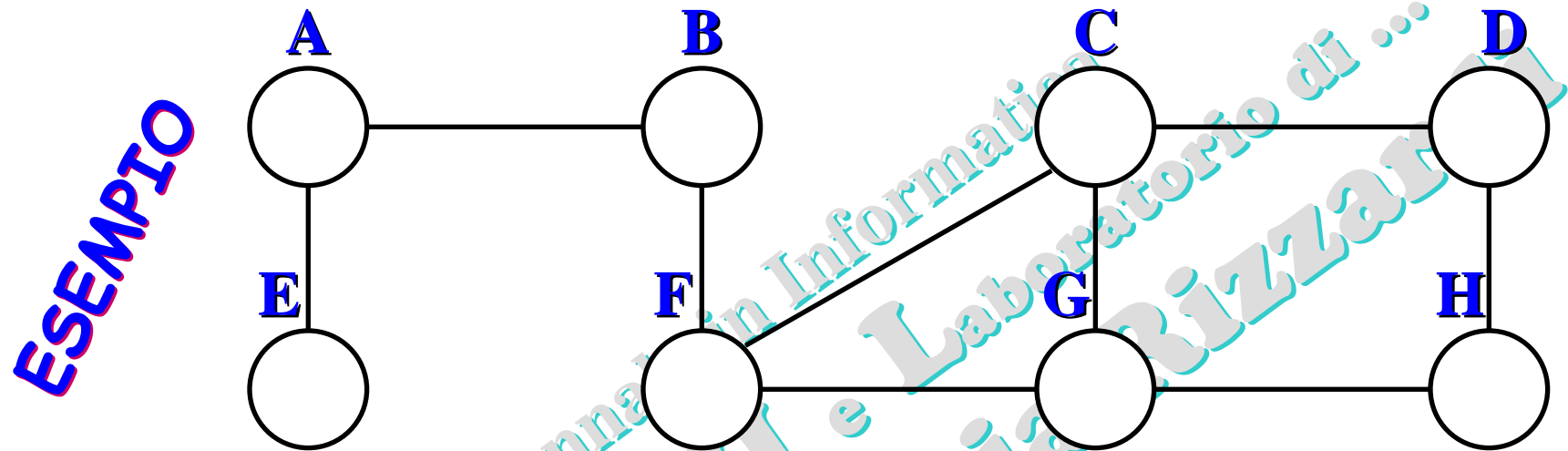
Algoritmi di visita di un grafo: visita in ampiezza (BFS - Breadth First Search)

ESEMPIO



La **visita in ampiezza** viene normalmente implementata utilizzando una **coda**: ogni volta che si visita un nodo, lo si mette nella coda. Ad ogni passo si estrae un nodo dalla coda e si controllano i nodi ad esso adiacenti: se non sono ancora stati visitati, li si visita.

Algoritmi di visita di un grafo: visita in ampiezza



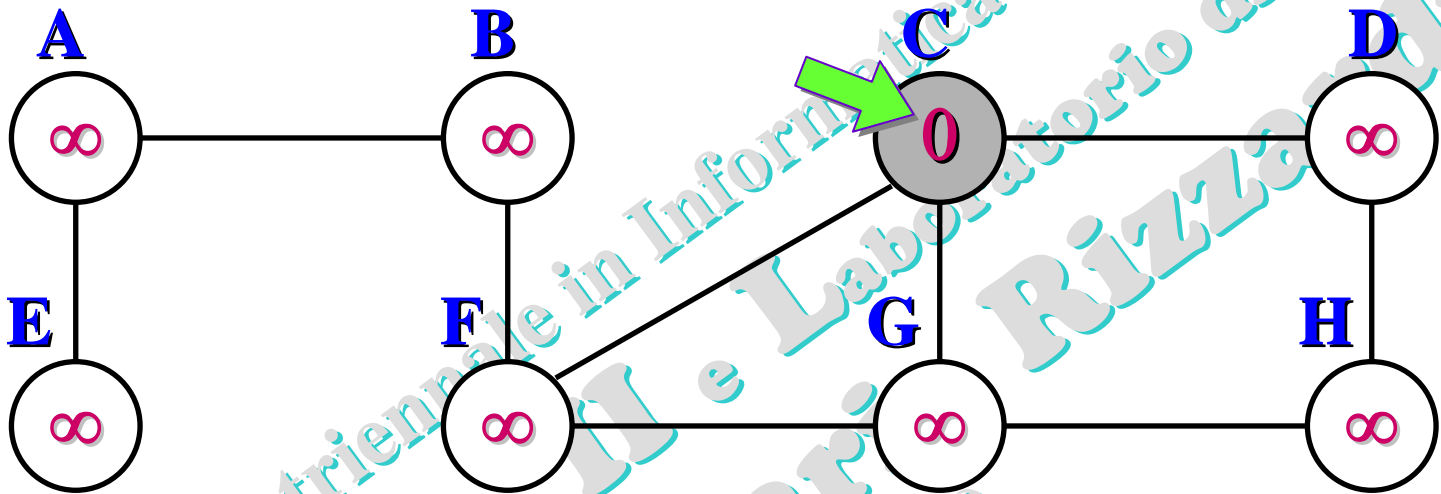
Per simulare il funzionamento dell'algoritmo si utilizza una procedura consistente nella **colorazione** dei vertici con le seguenti regole:

- Inizialmente tutti i vertici sono bianchi.
- Un vertice è colorato in grigio quando viene raggiunto per la prima volta.
- Un vertice è colorato in nero quando tutti i vertici ad esso adiacenti e non ancora visitati sono stati inseriti nella coda.

Inoltre in ogni vertice, oltre al colore, viene indicato il suo livello che sarà pari a quello del vertice del livello precedente +1.

Visita in ampiezza

idea algoritmo [1]

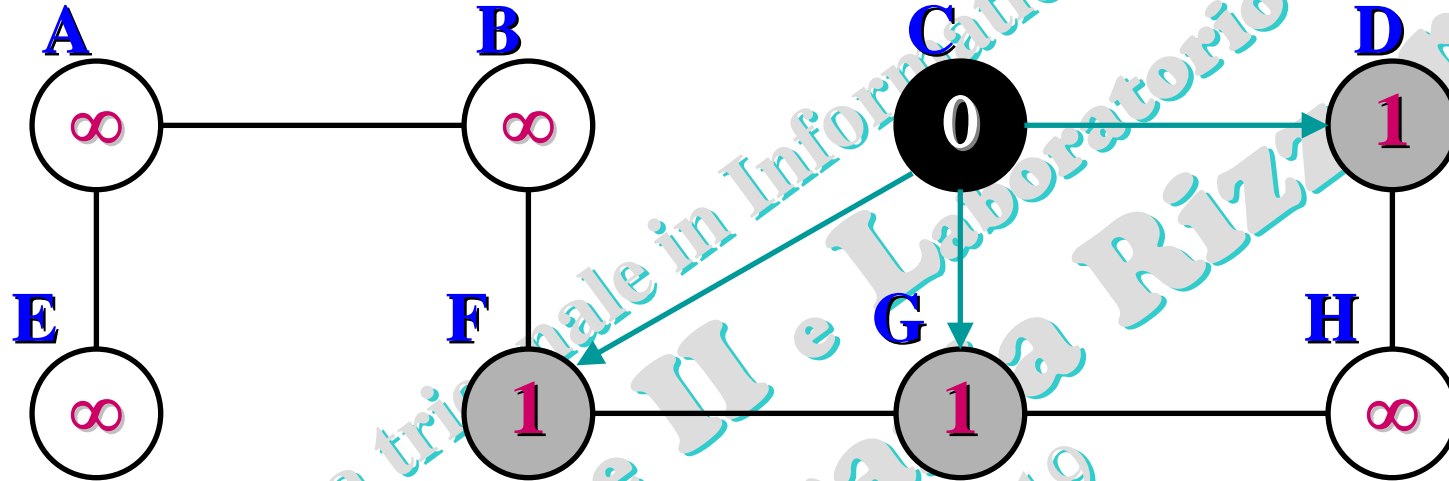


- Inizialmente viene assegnato il valore ∞ al livello di tutti i vertici.
- Si parte, ad es., dal nodo **C** come vertice sorgente: viene visitato, colorato di grigio e gli viene assegnato il livello 0.
- **C** viene inserito nella coda.

coda **C**

visita

C



coda **C**

- Estrae nodo dalla coda (**C**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti a **C**: li visita, li colora di grigio ed assegna loro il livello di **C** incrementato di 1.

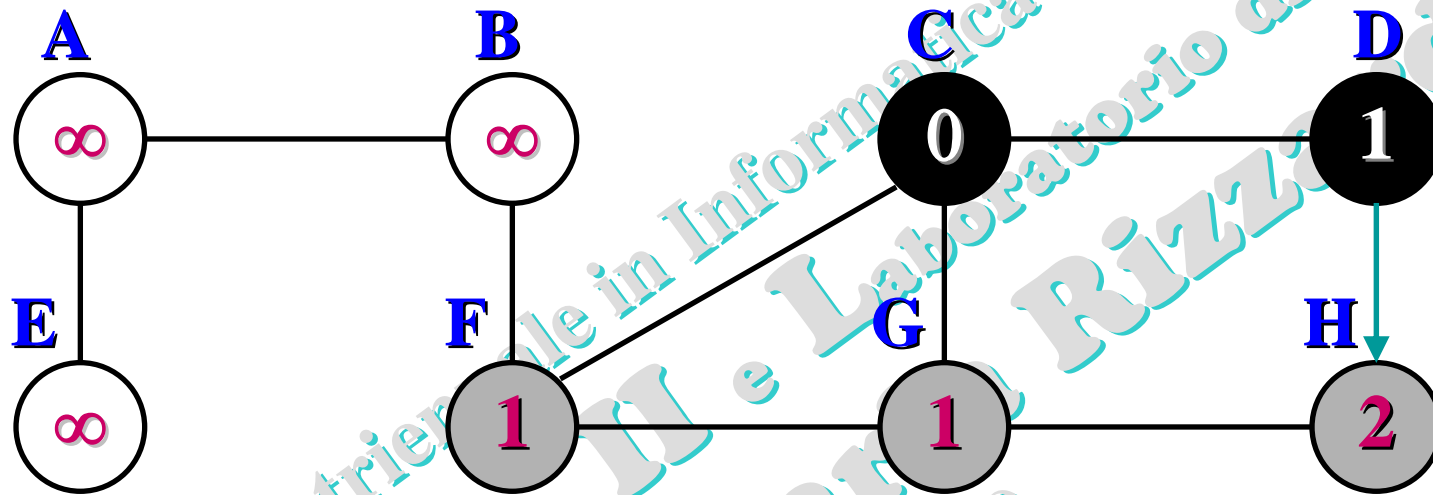
coda **D G F**

visita

C
D
G
F

Visita in ampiezza

idea algoritmo [3]



coda **D** **G** **F**

- Estrae nodo dalla coda (**D**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti al nodo (**D**) non ancora visitati (bianchi o di livello ∞): li visita, li colora di grigio ed assegna loro il livello di **D** incrementato di 1.

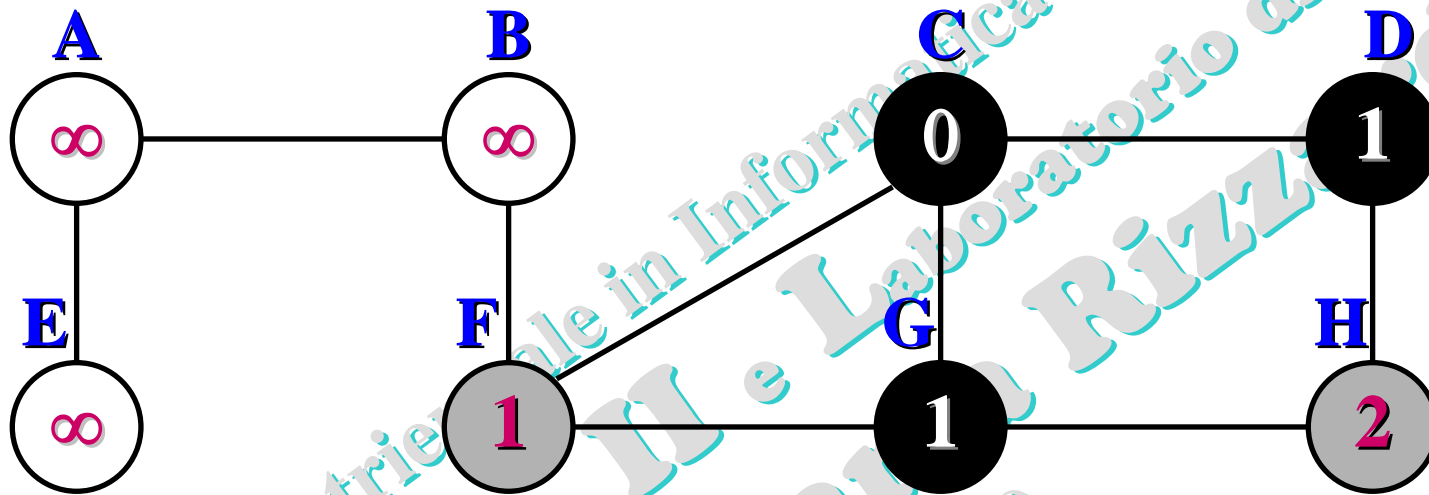
coda **G** **F** **H**

visita

C
D
G
F
H

Visita in ampiezza

idea algoritmo [4]



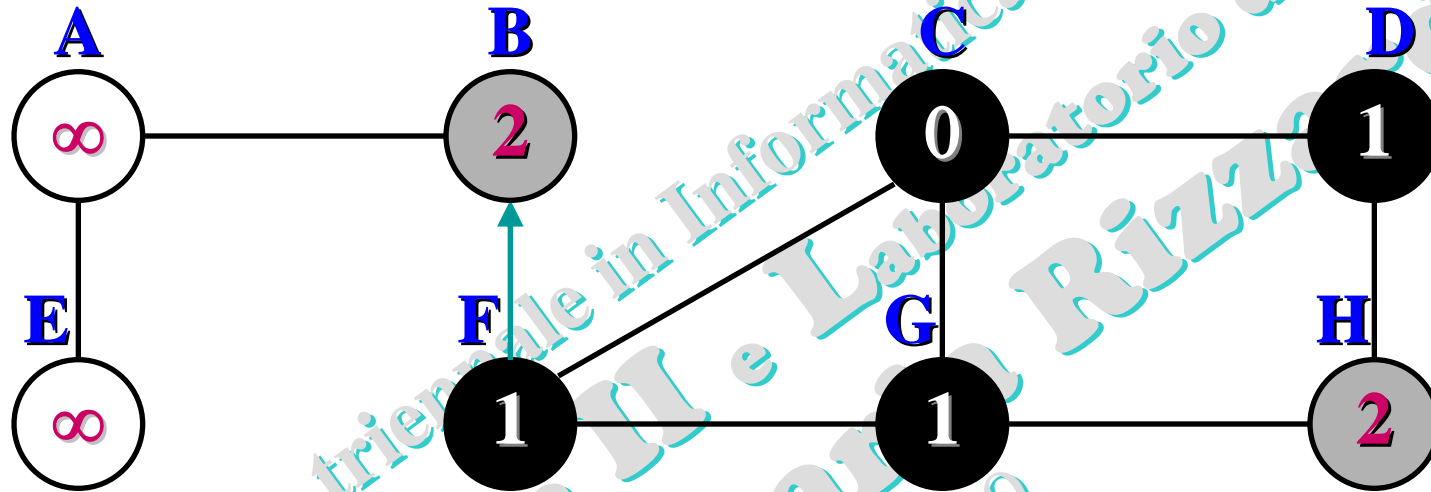
coda **G** **F** **H**

- Estrae nodo dalla coda (**G**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti a **G** non ancora toccati (bianchi o di livello ∞)...[in questo caso non ce ne sono!].

coda **F** **H**

Visita in ampiezza

idea algoritmo [5]



coda **F** **H**

- Estrae nodo dalla coda (**F**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti ad **F** non ancora visitati (bianchi o di livello ∞): li visita, li colora di grigio ed assegna loro il livello del nodo incrementato.

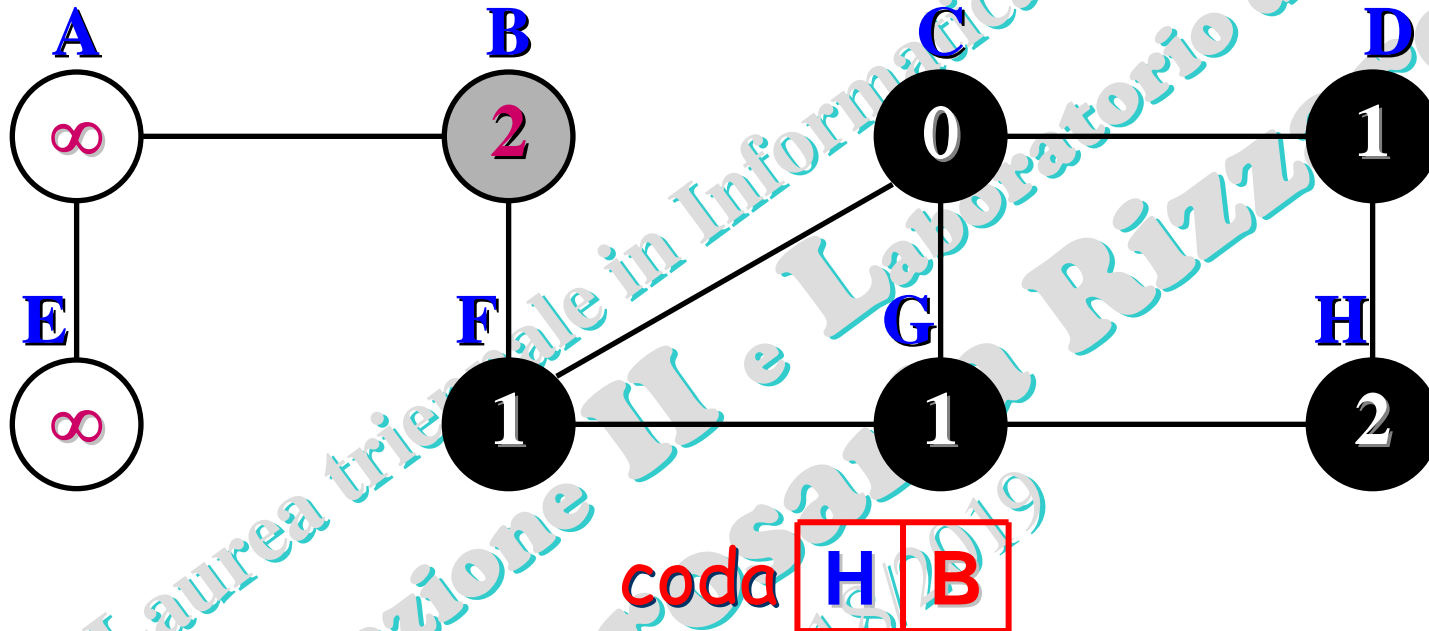
coda **H** **B**

visita

C
D
G
F
H
B

Visita in ampiezza

idea algoritmo [6]

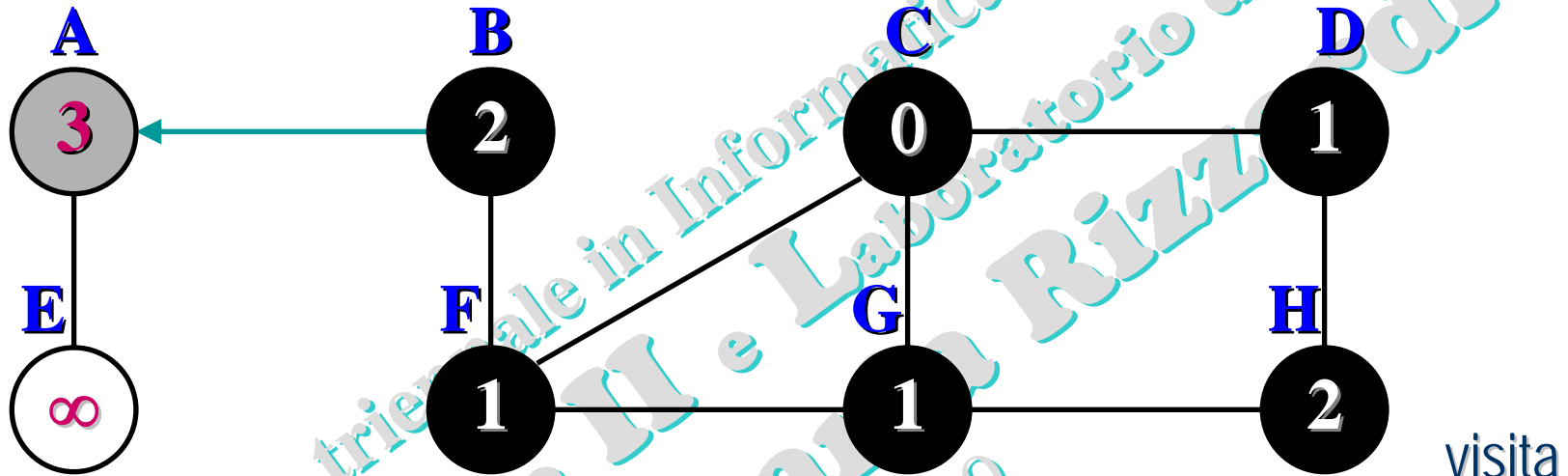


- Estrae nodo dalla coda (**H**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti ad **H** non ancora visitati (bianchi o di livello ∞)...[in questo caso non ce ne sono!].

coda **B**

Visita in ampiezza

idea algoritmo [7]



- Estrae nodo dalla coda (**B**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti a **B** non ancora visitati (bianchi o di livello ∞): li visita, li colora di grigio ed assegna loro il livello del nodo incrementato.

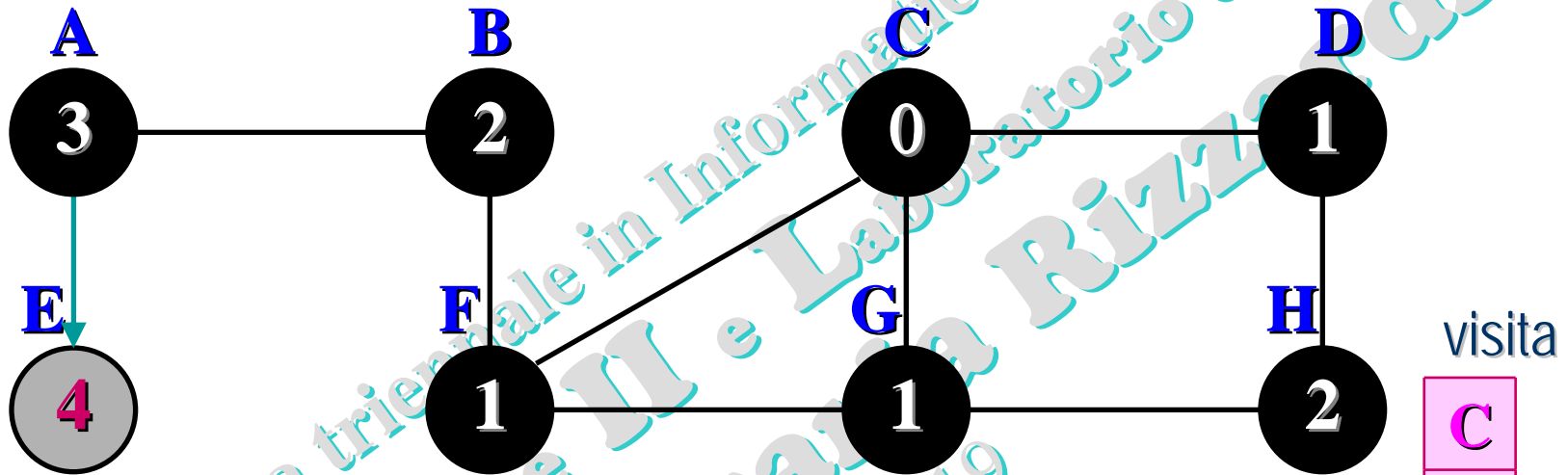
coda **A**

visita

C
D
G
F
H
B
A

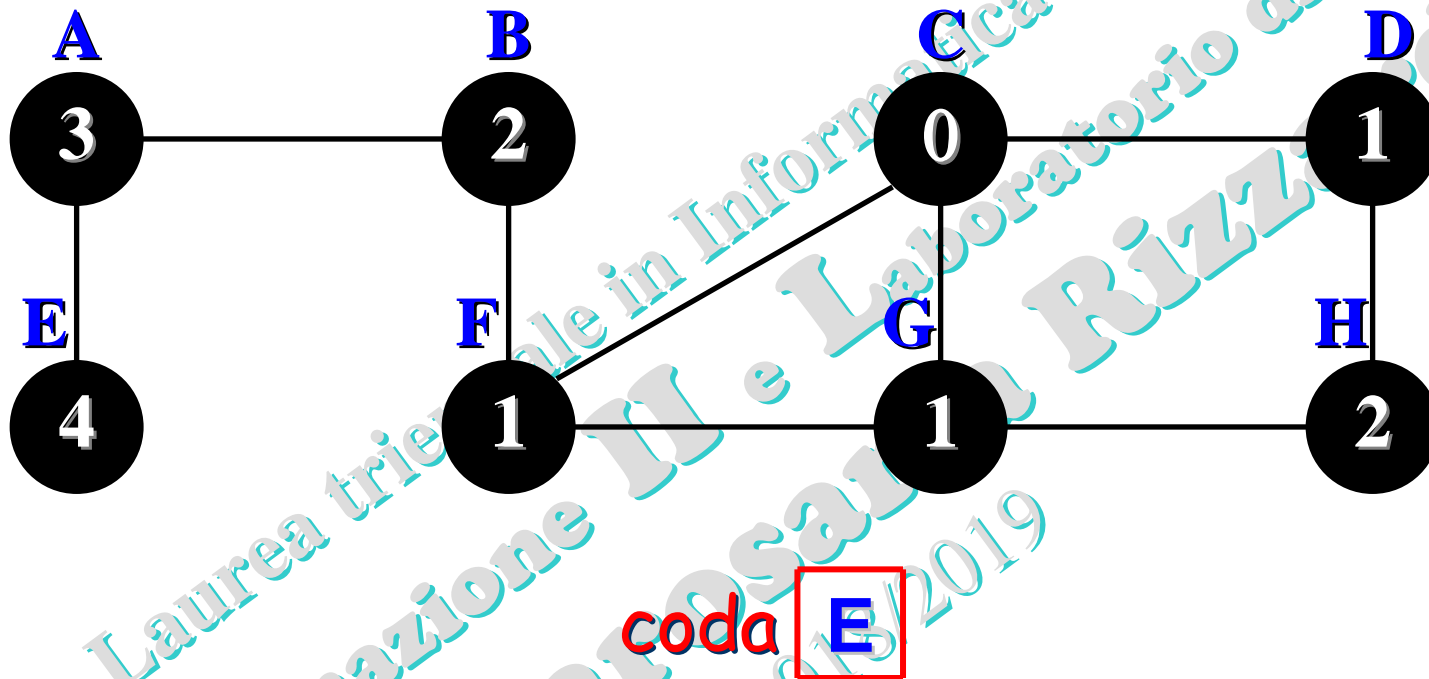
Visita in ampiezza

idea algoritmo [8]



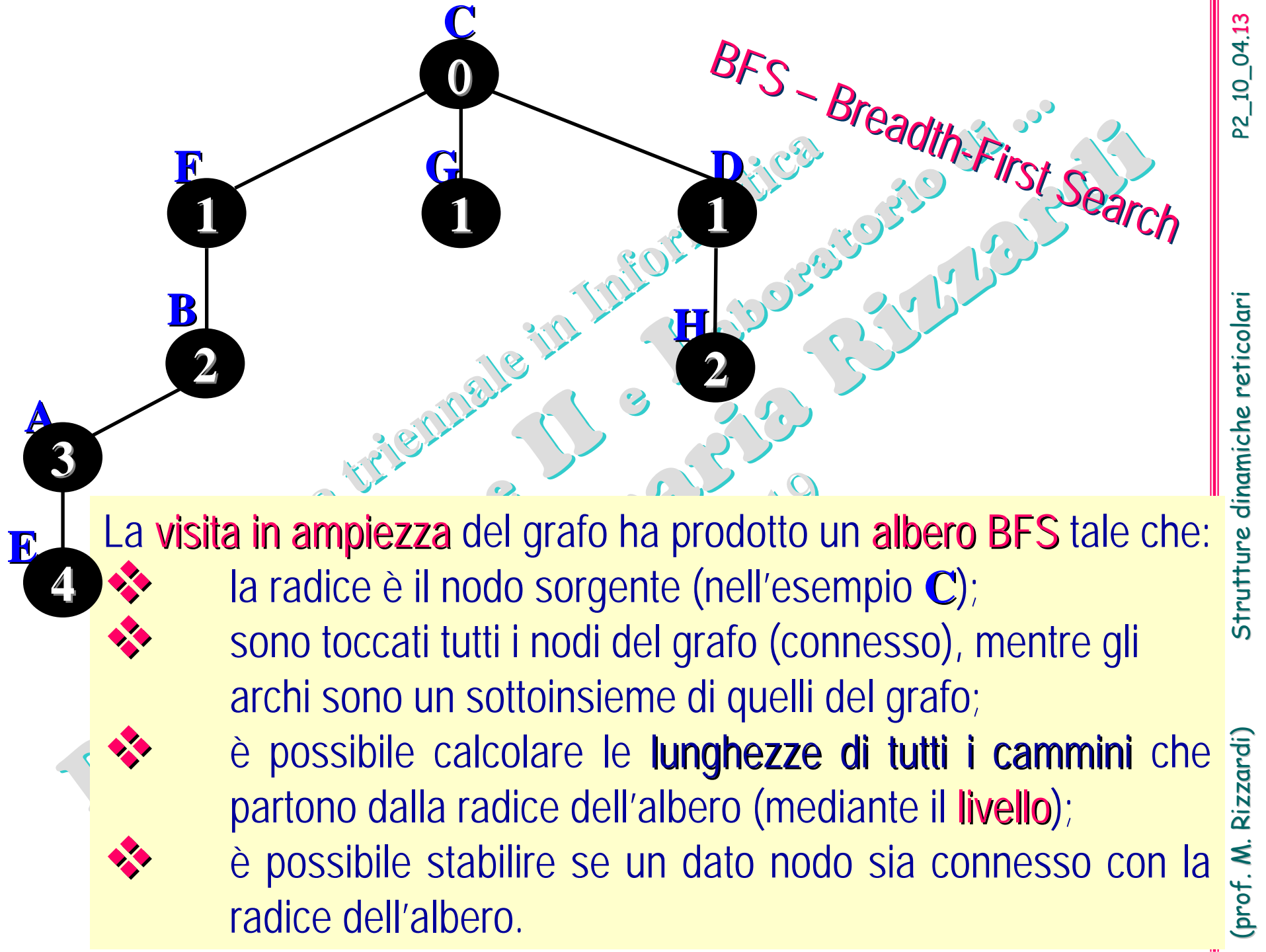
- Estrae nodo dalla coda (**A**) e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti ad **A** non ancora visitati (bianchi o di livello ∞): li visita, li colora di grigio ed assegna loro il livello del nodo incrementato.

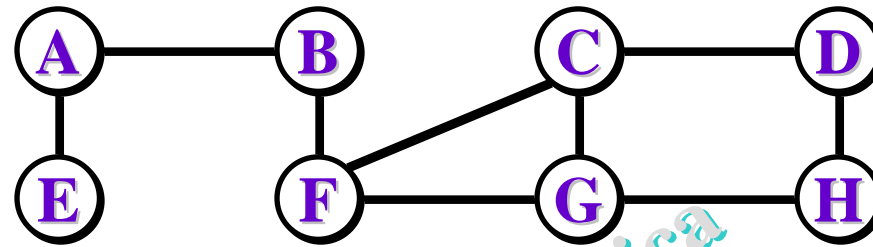
codas **E**



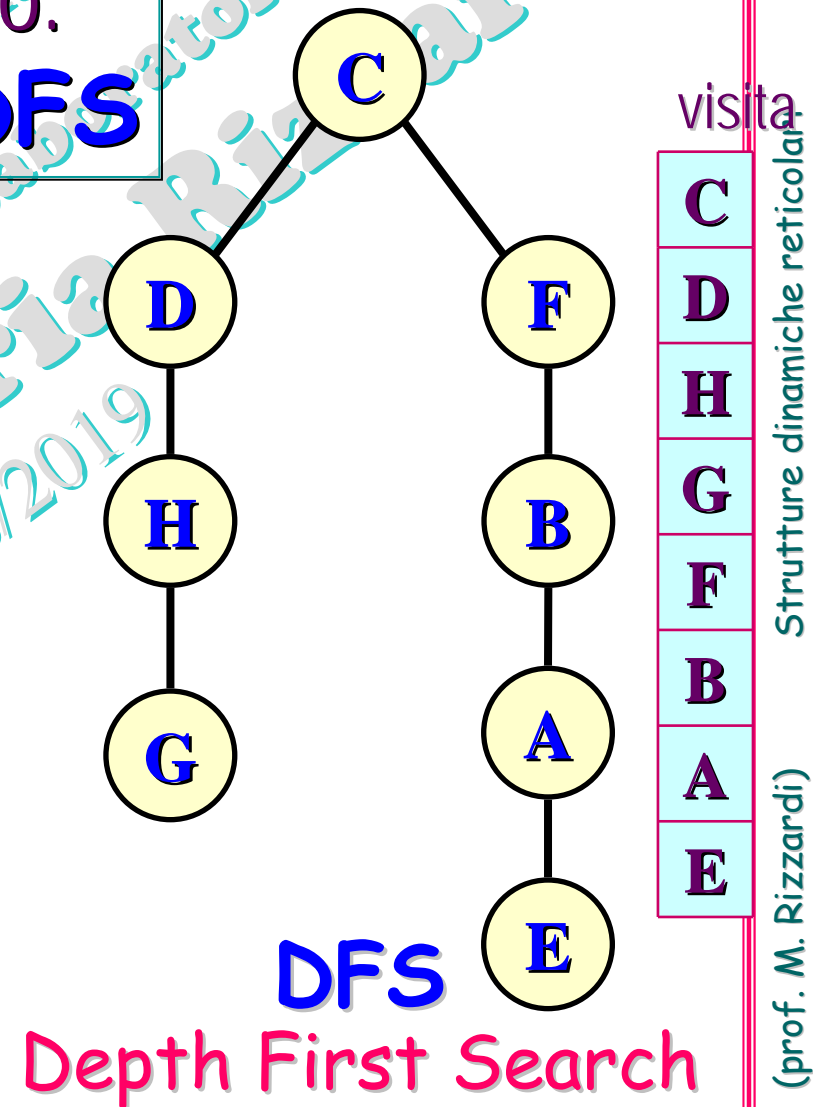
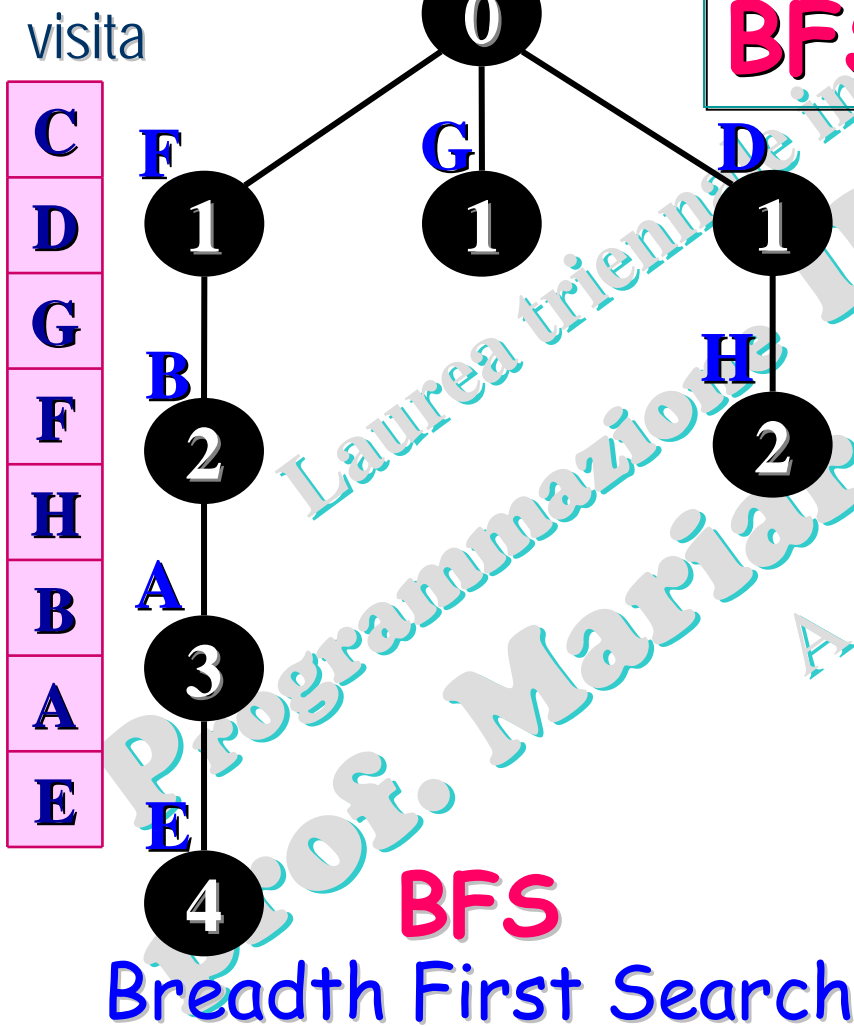
- Estrae nodo dalla coda (**E** e lo colora di nero.
- Inserisce nella **coda** i vertici adiacenti al nodo non ancora visitati (bianchi o di livello ∞)...[in questo caso non ce ne sono!].

coda $\emptyset \Rightarrow$ fine algoritmo





confronto:
BFS vs **DFS**



Esercizio

1

Scrivere *function C* per la visita in ampiezza (BFS – Breadth First Search) di un grafo: applicare questo algoritmo per stabilire se esiste un cammino che unisce due nodi qualsiasi di un grafo dati in input. [liv. 3]