

# Grafi

Progettazione di Algoritmi a.a. 2019-20

Matricole congrue a 1

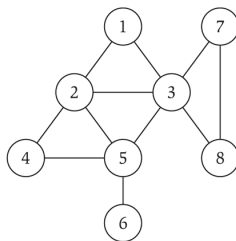
Docente: Annalisa De Bonis

1

1

## Grafi non direzionati

- **Grafi non direzionati.**  $G = (V, E)$ 
  - $V$  = insieme nodi.
  - $E$  = insieme archi.
  - Esprime le relazioni tra coppie di oggetti.
  - Parametri del grafo:  $n = |V|$ ,  $m = |E|$ .



$$V = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

$$E = \{1-2, 1-3, 2-3, 2-4, 2-5, 3-5, 3-7, 3-8, 4-5, 5-6, 7-8\}$$

$$n = 8$$

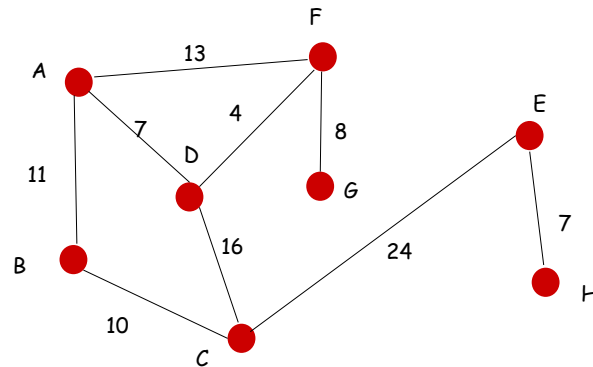
$$m = 11$$

2

2

### Esempio di applicazione

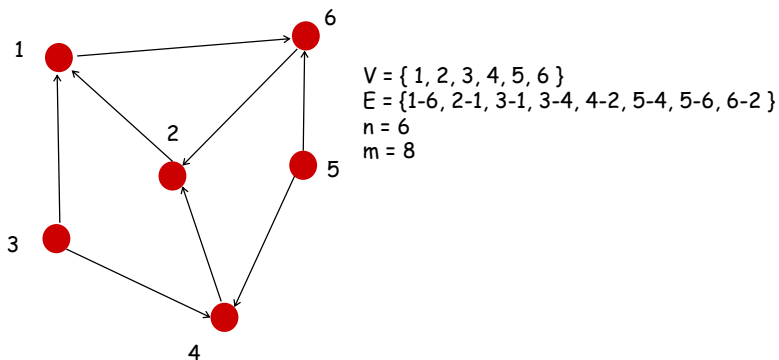
- Archi: strade (a doppio senso di circolazione)
- Nodi: intersezioni tra strade
- Pesi archi: lunghezza in km



3

### Grafi direzionati

- Gli archi hanno una direzione
  - L'arco  $(u,v)$  è diverso dall'arco  $(v,u)$
  - Si dice che l'arco  $e=(u,v)$  lascia  $u$  ed entra in  $v$
  - e che  $u$  è l'origine dell'arco e  $v$  la destinazione dell'arco

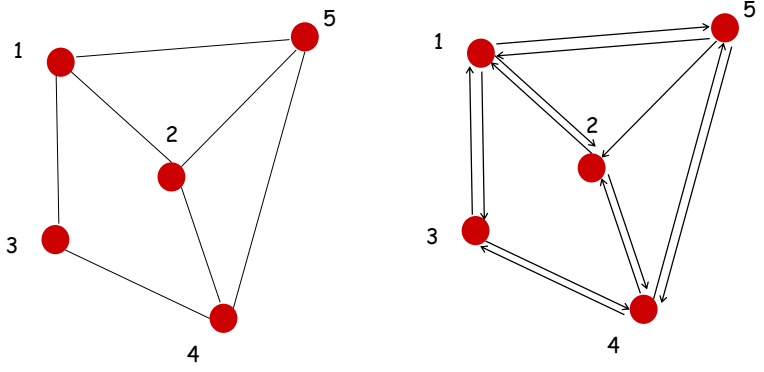


Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

4

### Grafì direzionati

- Grafì non direzionati  $G = (V, E)$  possono essere visti come un caso particolare degli archi direzionati in cui per ogni arco  $(u,v)$  c'è l'arco di direzione opposta  $(v,u)$

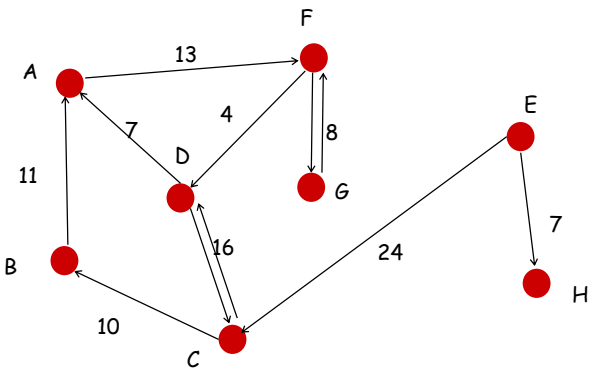


Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

5

### Esempio di applicazione

- Archi: strade (a senso unico di circolazione)
- Nodi: intersezioni tra strade
- Pesi archi: lunghezza in km



6

### Alcune applicazioni dei grafi

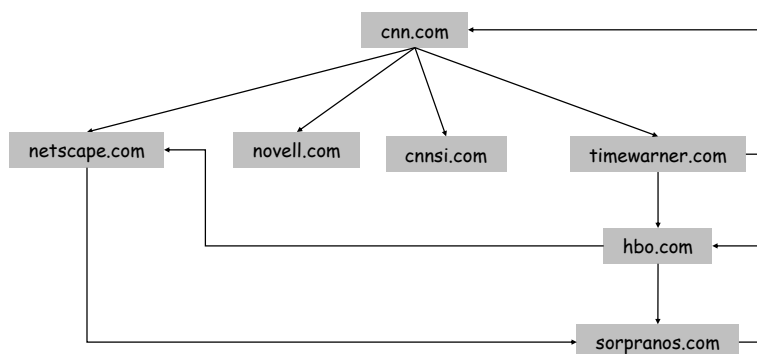
- Rete di amicizia su un social network: ogni utente è un nodo; ogni volta che due utenti diventano amici, si crea un arco del grafo.
- Google maps: i nodi rappresentano città, intersezioni di strade, siti di interesse, ecc. e gli archi rappresentano le connessioni dirette tra i nodi.
  - La rappresentazione mediante un grafo permette di trovare il percorso più corto per andare da un posto all'altro mediante un algoritmo.
- World Wide Web: le pagine web sono i nodi e il link tra due pagine è un arco. Google utilizza questa rappresentazione per esplorare il World Wide Web

7

7

### World Wide Web

- **Web graph.**
  - Nodo: pagina web.
  - Edge: hyperlink da una pagina all'altra.



Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

8

8

### Ecological Food Web

- Food web graph.
  - Nodo = specie
  - Arco dalla preda al predatore.

Reference: <http://www.twingroves.district96.k12.il.us/Wetlands/Salamander/SalGraphics/salfoodweb.gif>

Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

9

### Alcune applicazioni dei grafi

<i>Graph</i>	<i>Nodi</i>	<i>Archi</i>
trasporto	intersezioni di strade	strade
trasporto	aeroporti	voli diretti
comunicazione	computer	cavi di fibra ottica
World Wide Web	web page	hyperlink
rete sociale	persone	relazioni
catena del cibo	specie	predatore-preda
scheduling	task	vincoli di precedenza
circuiti	gate	wire

Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

10

### Terminologia

- Consideriamo due nodi  $u$  e  $v$  di un grafo  $G$  connessi dall'arco  $e = (u,v)$
- Si dice che
  - $u$  e  $v$  sono adiacenti
  - $u$  e  $v$  sono le estremità dell'arco  $(u,v)$
  - l'arco  $(u,v)$  incide sui vertici  $u$  e  $v$
  - $u$  è un nodo vicino di  $v$
  - $v$  è un nodo vicino di  $u$
- Dato un vertice  $u$  di un grafo  $G$ 
  - grado di  $u$  = numero archi incidenti su  $u$ 
    - è indicato con  $\text{deg}(u)$

11

11

### Numero di archi di un grafo non direzionato

$m$  = numero di archi di  $G$ ;

$n$  = numero di nodi di  $G$  .

1. La somma di tutti i gradi dei nodi di  $G$  è  $2m$  :  $\sum_{u \in V} \text{deg}(u) = 2m$

Degree (grado) = numero di vicini di  $u$

**Dim.** Ciascun arco incide su due vertici e quindi viene contato due volte nella sommatoria in alto. L'arco  $(x,y)$  è contato sia in  $\text{deg}(x)$  che in  $\text{deg}(y)$ .

2. Il numero  $m$  di archi di un grafo  $G$  non direzionato è al più  $n(n-1)/2$  .

**Dim.** Il numero di coppie **non ordinate** distinte che si possono formare con  $n$  nodi è  $n(n-1)/2$  .

Posso scegliere il primo nodo dell'arco in  $n$  modi e il secondo in modo che sia diverso dal primo nodo, cioè in  $n-1$  modi. Dimezzo in quanto l'arco  $(u,v)$  è uguale all'arco  $(v,u)$

12

12

### Numero di archi di un grafo direzionato

$m$  = numero di archi di  $G$ ;

$n$  = numero di nodi di  $G$

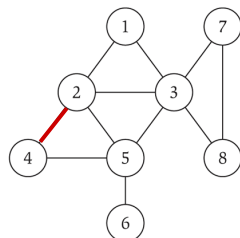
Il numero  $m$  di archi di  $G$  è al più  $n^2$

**Dim.** Il numero di coppie **ordinate** distinte che si possono formare con  $n$  nodi è  $n^2$ . Posso scegliere il primo nodo dell'arco in  $n$  modi e il secondo in altri  $n$  modi (se ammettiamo archi con entrambe le estremità uguali).

13

### Graph Representation: Adiacenza Matrix

- **Matrice di adiacenza.** Matrice  $n \times n$  con  $A_{uv} = 1$  se  $(u, v)$  è un arco.
  - Due rappresentazioni di ciascun arco.
  - Spazio proporzionale a  $n^2$ .
  - Controllare se  $(u, v)$  è un arco richiede tempo  $\Theta(1)$ .
  - Identificare tutti gli archi richiede tempo  $\Theta(n^2)$ .

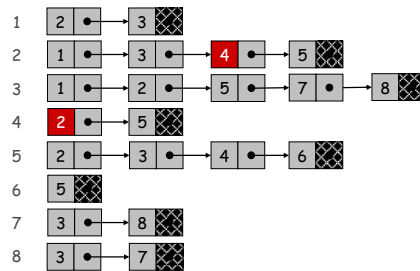
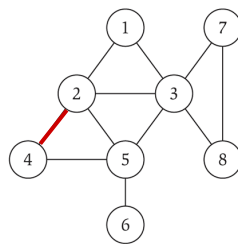


	1	2	3	4	5	6	7	8
1	0	1	1	0	0	0	0	0
2	1	0	1	1	0	0	0	0
3	1	1	0	0	1	0	1	1
4	0	1	0	0	1	0	0	0
5	0	1	1	1	0	1	0	0
6	0	0	0	0	1	0	0	0
7	0	0	1	0	0	0	0	1
8	0	0	1	0	0	0	1	0

14

### Rappresentazione di un grafo: liste di adiacenza

- **Liste di adiacenza.** Array di liste in cui ogni lista è associata ad un nodo.
  - Ad ogni arco corrisponde un elemento della lista.
  - Se esiste l'arco  $(u,v)$  allora la lista associata ad  $u$  contiene  $v$
  - In un grafo non direzionato l'arco  $(u,v)$  corrisponde ad un elemento della lista associata ad  $u$  e ad un elemento della lista associata a  $v$
  - Spazio proporzionale a  $m + n$ .
  - Controllare se  $(u, v)$  è un arco richiede tempo  $O(deg(u))$ .
  - Individuare tutti gli archi richiede tempo  $\Theta(m + n)$ .



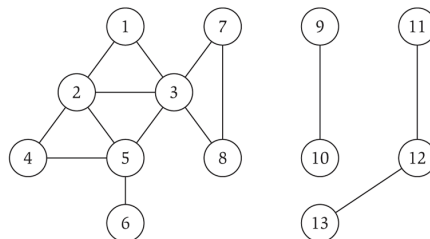
Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

15

15

### Percorsi e connettività

- **Def.** Un percorso in un grafo non direzionato  $G = (V, E)$  è una sequenza  $P$  di nodi  $v_1, v_2, \dots, v_{k-1}, v_k$  con la proprietà che ciascuna coppia di vertici consecutivi  $v_i, v_{i+1}$  è unita da un arco in  $E$ .
- **Def.** Un percorso è **semplice** se tutti i nodi sono distinti.
- **Def.** Un grafo non direzionato è **connesso** se per ogni coppia di nodi  $u$  e  $v$ , esiste un percorso tra  $u$  e  $v$ .



Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

16

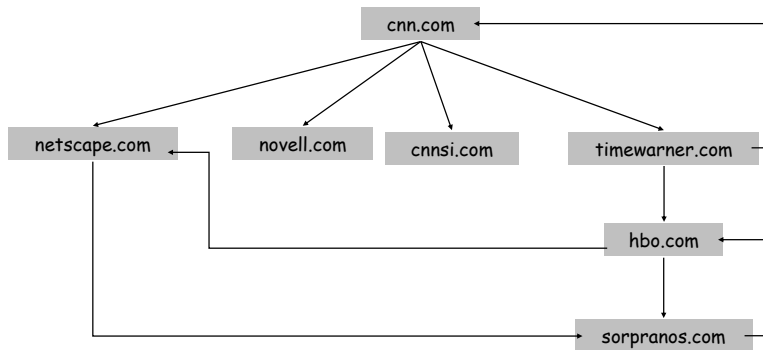
16



### Applicazione del concetto di percorso

- **Esempi:**

Web graph. Voglio capire se è possibile, partendo da una pagina web e seguendo gli hyperlink nelle pagine via via attraversate, arrivare ad una determinata pagina



Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

17

17

### Applicazione del concetto di percorso

In alcuni casi può essere interessante scoprire il percorso più corto tra due nodi.

**Esempio:**

- Grafo : rete di trasporti dove i nodi sono gli aeroporti e gli archi i collegamenti diretti tra aeroporti.
- Voglio arrivare da Napoli a New York facendo il minimo numero di scali.

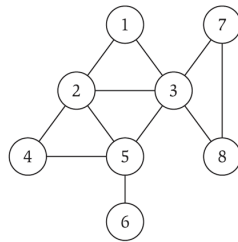
Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

18

18

## Cicli

- **Def.** Un ciclo è un percorso  $v_1, v_2, \dots, v_{k-1}, v_k$  in cui  $v_1 = v_k$ ,  $k > 2$ .
- **Def.** Un ciclo  $v_1, v_2, \dots, v_{k-1}, v_1$  è **semplice** se i primi  $k-1$  nodi del ciclo sono tutti distinti tra di loro



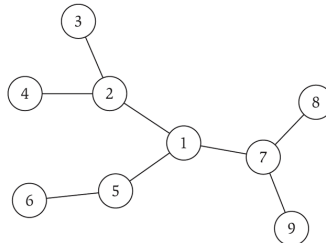
ciclo (semplice)  $C = 1-2-4-5-3-1$

ciclo (non semplice)  $C' = 1-3-7-8-3-5-2-1$

19

## Alberi

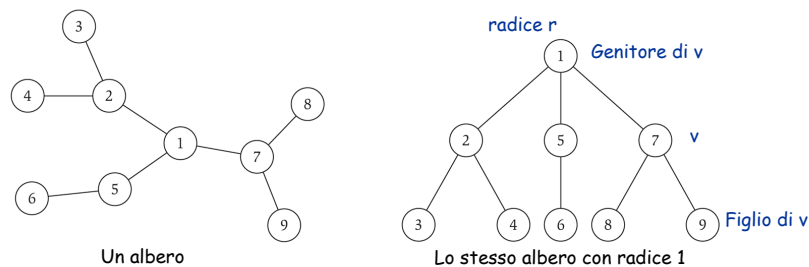
- **Def.** Un grafo non direzionato è un **albero (tree)** se è connesso e non contiene cicli
  - **Teorema.** Sia  $G$  un grafo direzionato con  $n$  nodi. Ogni due delle seguenti affermazioni implica la restante affermazione.
    - $1 \text{ e } 2 \implies 3$ ;  $1 \text{ e } 3 \implies 2$ ;  $2 \text{ e } 3 \implies 1$
1.  $G$  è connesso.
  2.  $G$  non contiene cicli.
  3.  $G$  ha  $n-1$  archi.



20

### Alberi con radice

- **Albero con radice.** Dato un albero  $T$ , si sceglie un nodo radice  $r$  e si considerano gli archi di  $T$  come orientati a partire da  $r$
- Dato un nodo  $v$  di  $T$  si dice
  - **Genitore di  $v$ :** il nodo che  $w$  precede  $v$  lungo il percorso da  $r$  a  $v$  ( $v$  viene detto figlio di  $w$ )
  - **Antenato di  $v$ :** un qualsiasi nodo  $w$  lungo il percorso che va da  $r$  a  $v$  ( $v$  viene detto discendente di  $w$ )
- **Foglia:** nodo senza discendenti

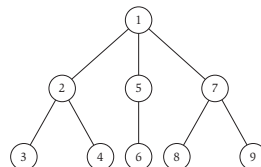


Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

21

### Alberi con radice

- Scegliere un nodo come radice, rende più semplice dimostrare la seguente affermazione
- se  $G$  è connesso e non contiene cicli, in altre parole, se  $G$  è un albero allora il numero di archi è  $n-1$  (dove  $n$  è il numero di nodi).
- **Dim.**
- Ogni nodo diverso dalla radice ha esattamente un arco che lo connette al proprio padre  $\rightarrow$  c'è un arco distinto per ogni nodo non radice  $\rightarrow$  numero di archi  $\geq n-1$
- Ogni arco connette esattamente un nodo non radice al proprio padre  $\rightarrow$  c'è un distinto nodo non radice per ogni arco  $\rightarrow$  numero di archi  $\leq n-1$
- le due disequaglianze  $\rightarrow$  numero di archi =  $n-1$

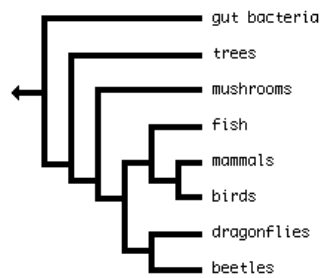


Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

22

### Importanza degli alberi: rappresentano strutture gerarchiche

- **Alberi filogenetici.** Descrivono la storia evolutiva delle specie animali.



La filogenesi afferma l'esistenza di una specie ancestrale che diede origine a mammiferi e uccelli ma non alle altre specie rappresentate nell'albero (cioè, mammiferi e uccelli condividono un antenato che non è comune ad altre specie nell'albero). La filogenesi afferma inoltre che tutti gli animali discendono da un antenato non condiviso con i funghi, gli alberi e i batteri, e così via.

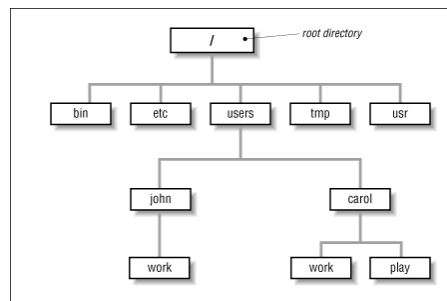
Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

23

23

### Importanza degli alberi: rappresentano strutture gerarchiche

- **File system.** Un file system tipicamente consiste di file organizzati in gruppi chiamati directory.
  - Una directory può contenere file e altre directory,
  - Un file system gerarchico è organizzato secondo una struttura gerarchica ad albero con radice
    - nodi interni: directory
    - foglie: file



Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

24

24

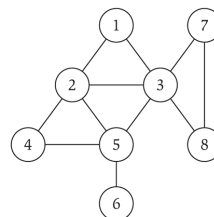
## Visite di grafi

Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

25

## Connettività

- **Problema della connettività tra  $s$  e  $t$ .** Dati due nodi  $s$  e  $t$ , esiste un percorso tra  $s$  e  $t$ ?
- **Problema del percorso più corto tra  $s$  e  $t$ .** Dati due nodi  $s$  e  $t$ , qual è la lunghezza del percorso più corto tra  $s$  e  $t$ ?
- **Applicazioni.**
  - Attraversamento di un labirinto.
  - Erdős number.
  - Minimo numero di dispositivi che devono essere attraversati dai dati in una rete di comunicazione per andare dalla sorgente alla destinazione



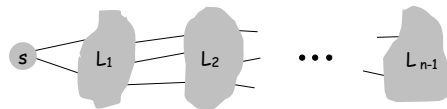
Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

26

26

### Breadth First Search (visita in ampiezza)

- **BFS.** Esplora il grafo a partire da una sorgente  $s$  muovendosi in tutte le possibile direzioni e visitando i nodi livello per livello (N.B.: il libro li chiama layer e cioè strati).



- **BFS algorithm.**
  - $L_0 = \{ s \}$ .
  - $L_1 =$  tutti i vicini di  $s$ .
  - $L_2 =$  tutti i nodi che non appartengono a  $L_0$  or  $L_1$ , e che sono uniti da un arco ad un nodo in  $L_1$ .
  - $L_{i+1} =$  tutti i nodi che non appartengono agli strati precedenti e che sono uniti da un arco ad un nodo in  $L_i$ .

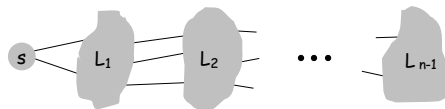
Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

27

27

### Breadth First Search

- **distanza tra  $u$  e  $v$**  = lunghezza del percorso piu' corto tra  $u$  e  $v$
- **Teorema.** Per ogni  $i$ ,  $L_i$  consiste di tutti i nodi a distanza  $i$  da  $s$ . C'è un percorso da  $s$  a  $t$  se e solo  $t$  appare in qualche livello.



$L_1$ : livello dei nodi a distanza 1 da  $s$   
 $L_2$ : livello dei nodi a distanza 2 da  $s$   
 $\dots$   
 $L_{n-1}$ : livello dei nodi a distanza  $n-1$  da  $s$

Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

28

28

### Breadth First Search

**Pseudocodice**

1. BFS(s)
2.  $L_0 = \{s\}$
3. **For**( $i=0; i \leq n-2; i++$ )
4.  $L_{i+1} = \emptyset$ ;
5. **Foreach** nodo  $u$  in  $L_i$
6.   **Foreach** nodo  $v$  adiacente ad  $u$
7.     **if** ( $v$  non appartiene ad  $L_0, \dots, L_{i+1}$ )
8.        $L_{i+1} = L_{i+1} \cup \{v\}$
9.     **EndIf**
10.   **Endforeach**
11. **Endforeach**
12. **Endfor**

} il for alle linee 6-10 è  
eseguito al più  $\sum_{u \in V} \text{deg}(u)$   
volte

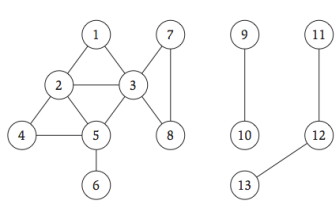
Occorre un modo per capire se un nodo è già stato visitato in precedenza. Il tempo di esecuzione dipende dal modo scelto, da come è implementato il grafo e da come sono rappresentati gli insiemi  $L_i$  che rappresentano i livelli

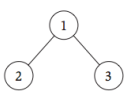
Progettazione di Algoritmi a.a. 2019-20  
A. De Bonis

29

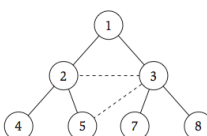
### Esempio di esecuzione di BFS

$G$

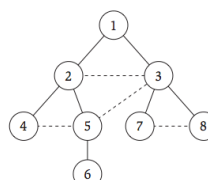




a



b



c

$L_0 = \{1\}$   
 a.  $L_1 = \{2, 3\}$   
 b.  $L_2 = \{4, 5, 7, 8\}$   
 c.  $L_3 = \{6\}$

30

30