STRUTTURE RELAZIONALI, GRAFI E ORDINAMENTI

(parte 4)

Stefania Bandini

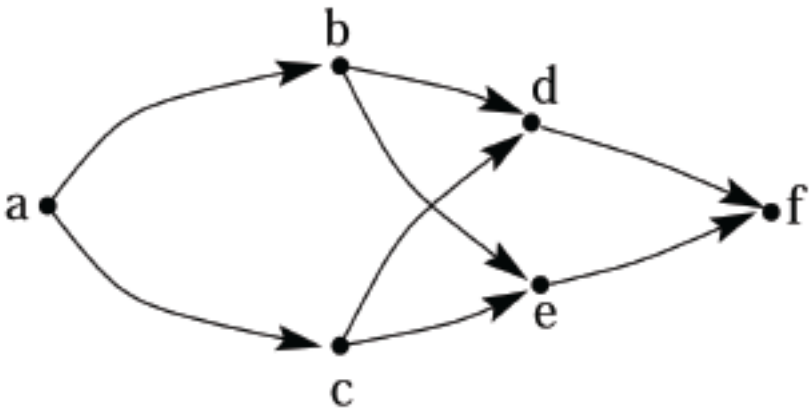
ALBERI

GRAFO DIRETTO ACICLICO (DAG)

Un *grafo diretto aciclico* (detto anche DAG, dall'inglese "Directed Acyclic Graph") è un grafo diretto senza cicli.

Sia $R \subseteq S \times S$ dove $S = \{a, b, c, d, e, f\}$;

$$R = \{\langle a, b\rangle, \langle a, c\rangle, \langle b, d\rangle, \langle c, d\rangle, \langle b, e\rangle, \langle c, e\rangle, \langle d, f\rangle, \langle e, f\rangle, \}.$$



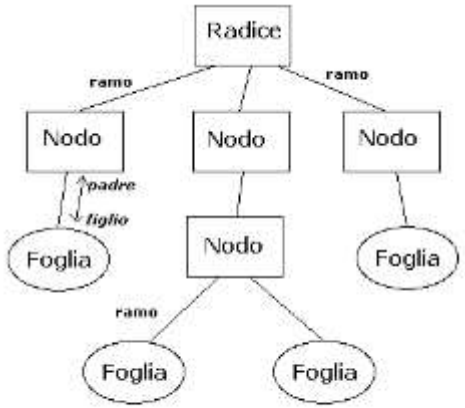
ALBERI

Un *albero* è un DAG connesso con un solo nodo sorgente (detto *radice* dell'albero) in cui ogni nodo diverso dalla radice ha un solo arco entrante.

I nodi privi di archi uscenti sono detti *foglie* dell'albero.

Per analogia con gli alberi genealogici i nodi intermedi si chiamano *padre*, *figli*, *fratelli*, *discendenti*, *avi*, con l'ovvio significato.

Di solito gli alberi vengono disegnati ponendo la radice in alto e le foglie in basso, in analogia con gli alberi genealogici. Quindi non è necessario disegnare le "punte" alle frecce.



ALBERI

Un **albero** o **struttura ad albero** (*tree* in inglese) è la struttura dati che si riconduce al concetto di albero con radice presente nella teoria dei grafi.

Un albero si compone di due tipi di sottostrutture fondamentali: il **nodo**, che in genere contiene informazioni, e l'**arco** che stabilisce un collegamento gerarchico fra due nodi: si parla allora di un **nodo padre** dal quale esce un arco orientato che lo collega ad un **nodo figlio**.

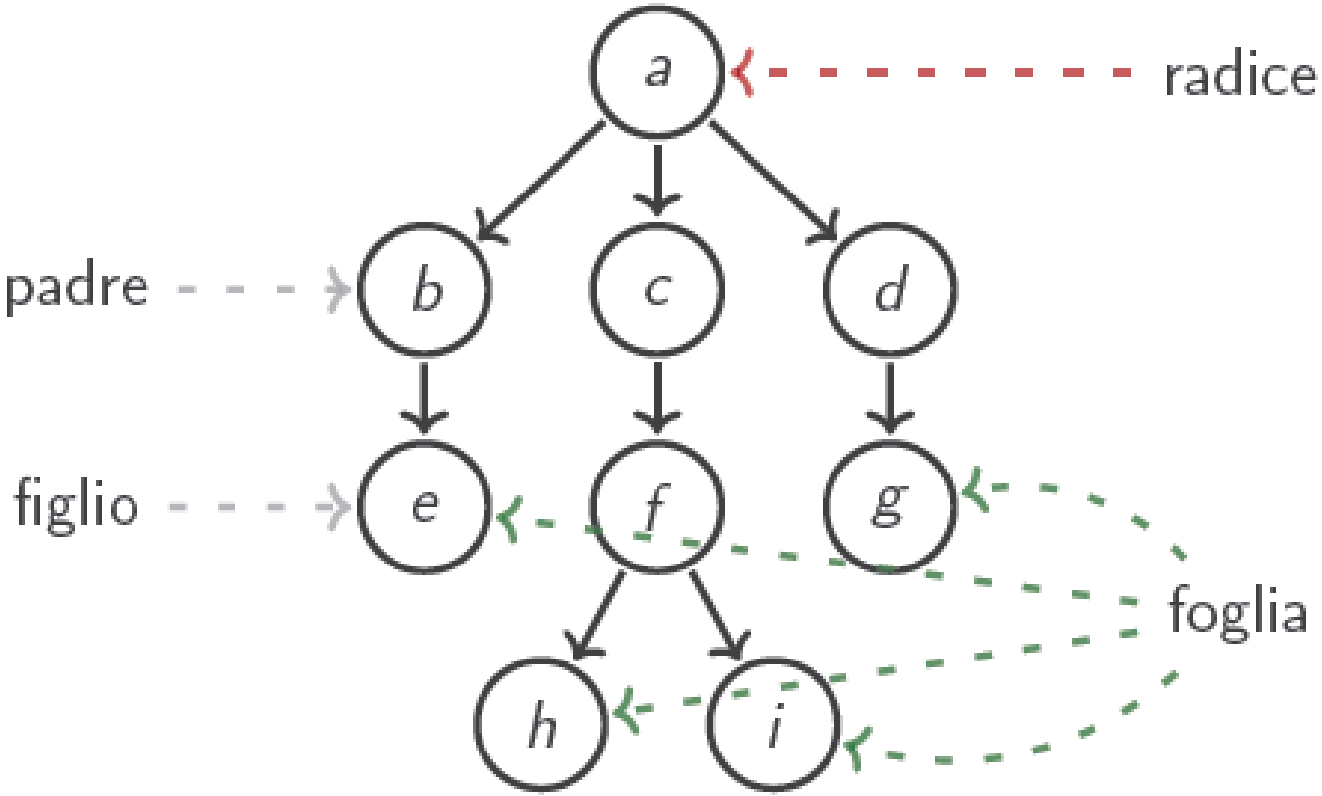
Ogni nodo può avere al massimo un unico arco entrante, mentre dai diversi nodi possono uscire diversi numeri di archi uscenti.

Un albero possiede un unico nodo privo di arco entrante: questo nodo viene detto **radice** (*root*) dell'albero.

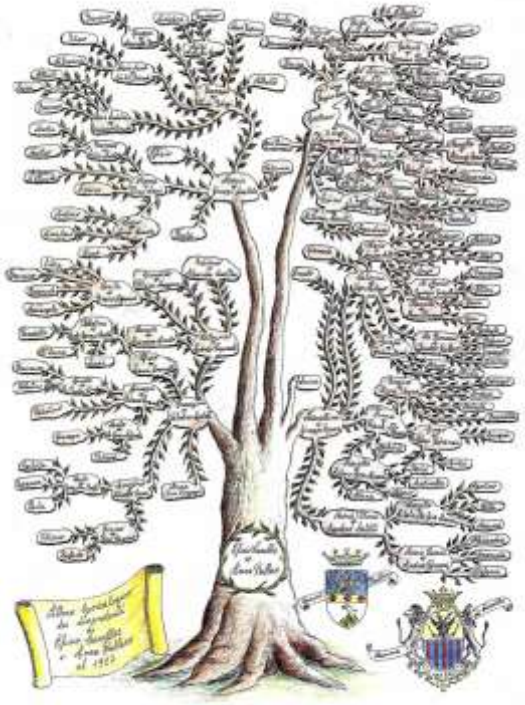
Ogni nodo che non presenta archi uscenti, è detto **foglia** (*leaf node*); e in ogni albero finito, cioè con un numero finito di nodi, si trova almeno un nodo foglia.

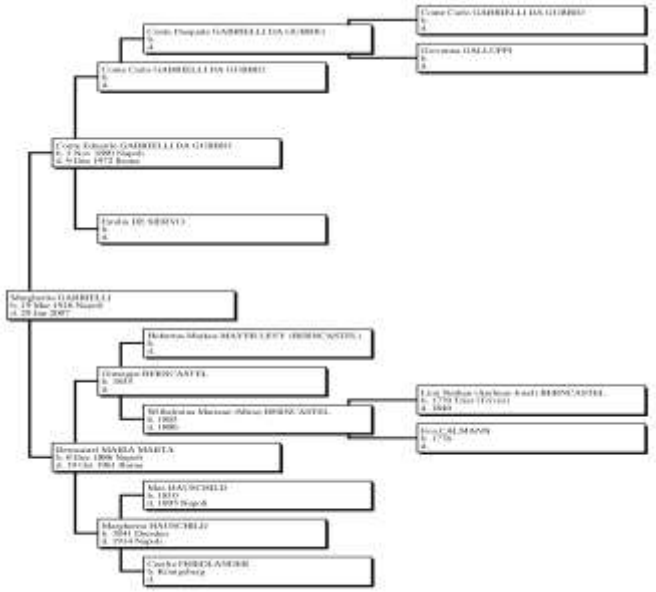
Un nodo può essere contemporaneamente padre (se ha archi uscenti) e figlio (se ha un arco entrante, ovvero se è diverso dalla radice).

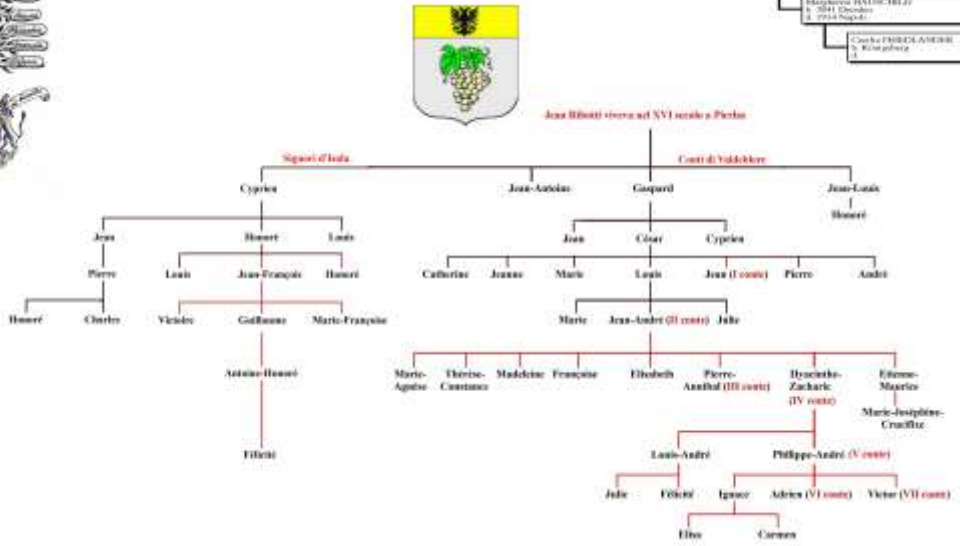
ALBERI



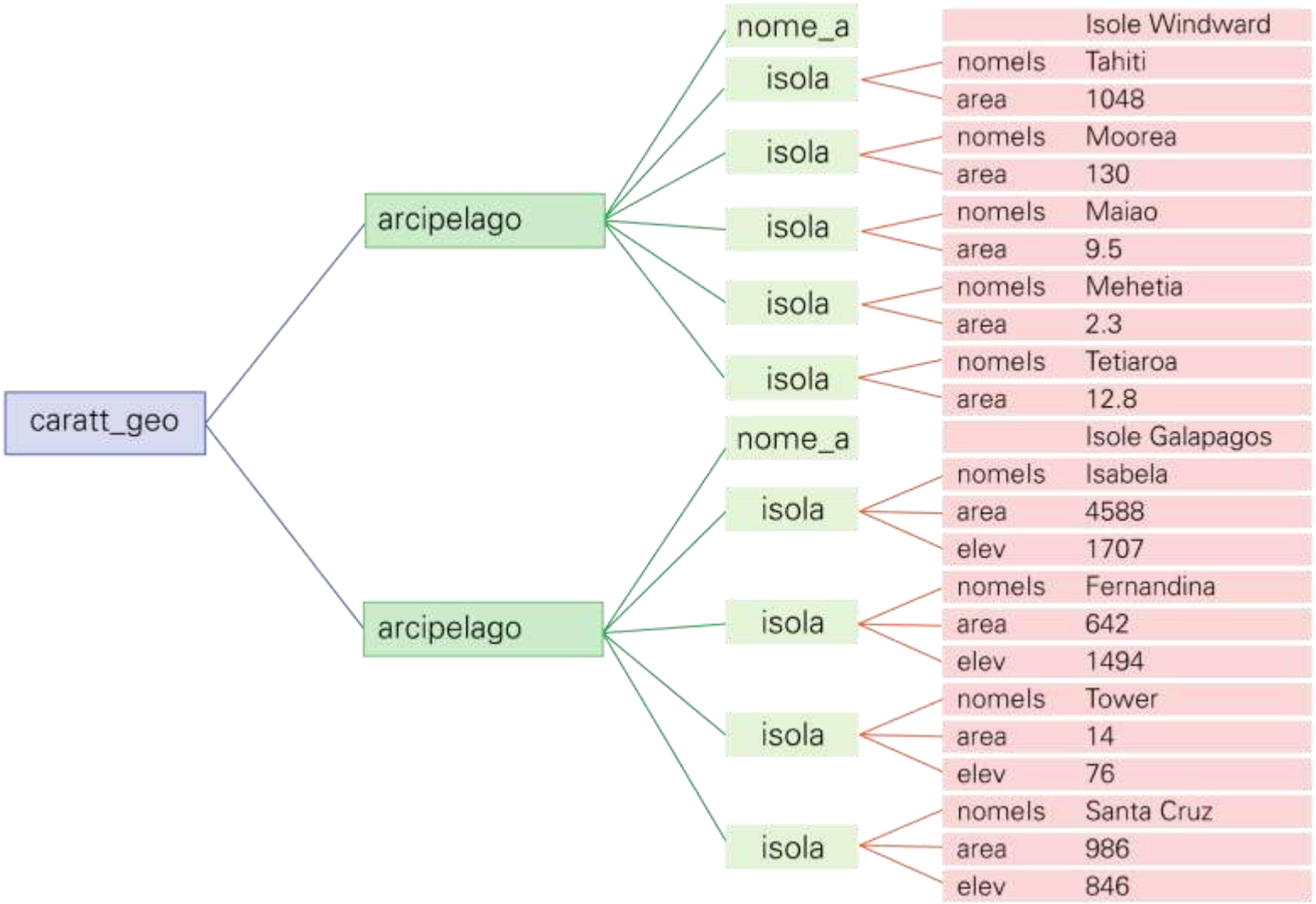
ALBERI - ESEMPI



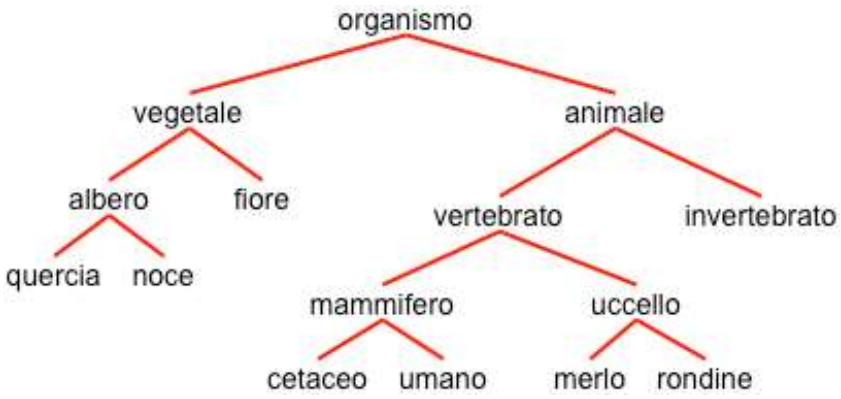


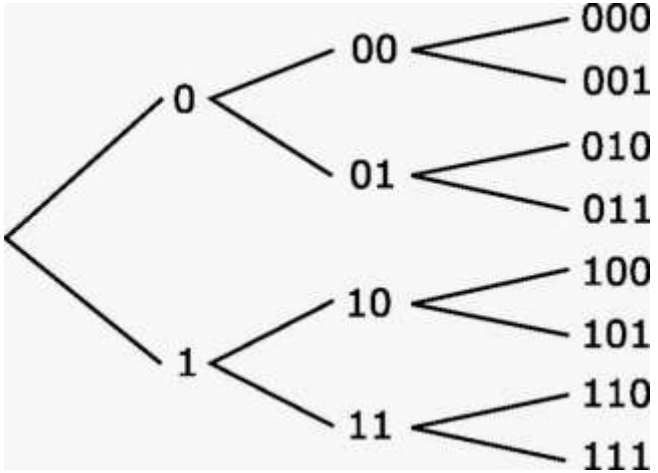


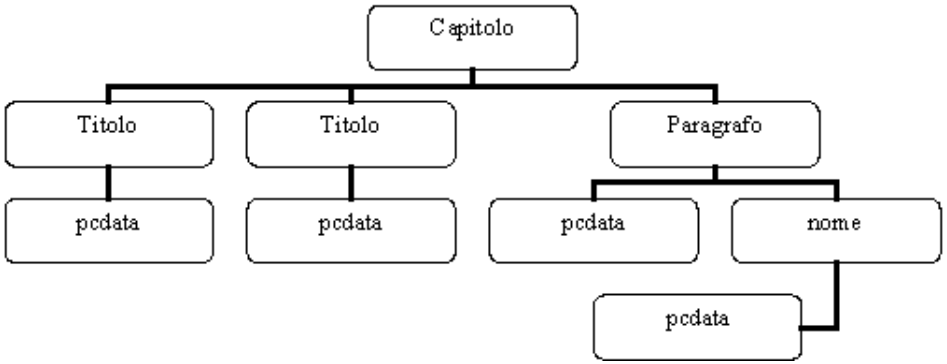
ALBERI - ESEMPI



ALBERI - ESEMPI



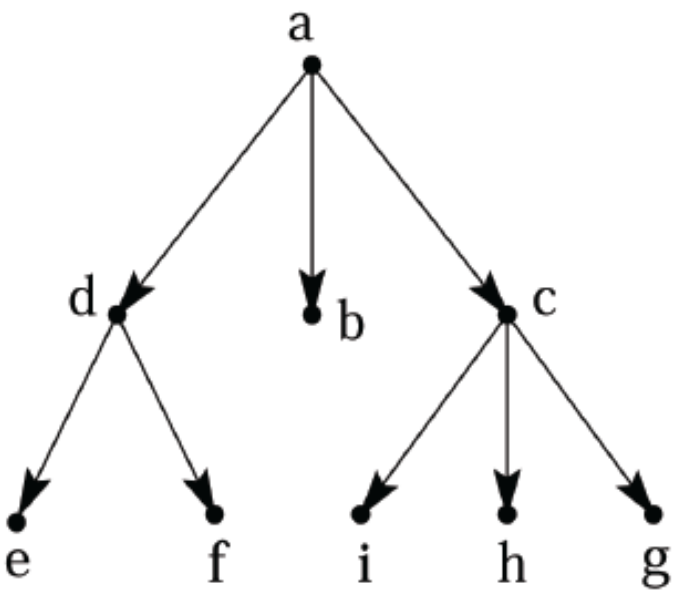




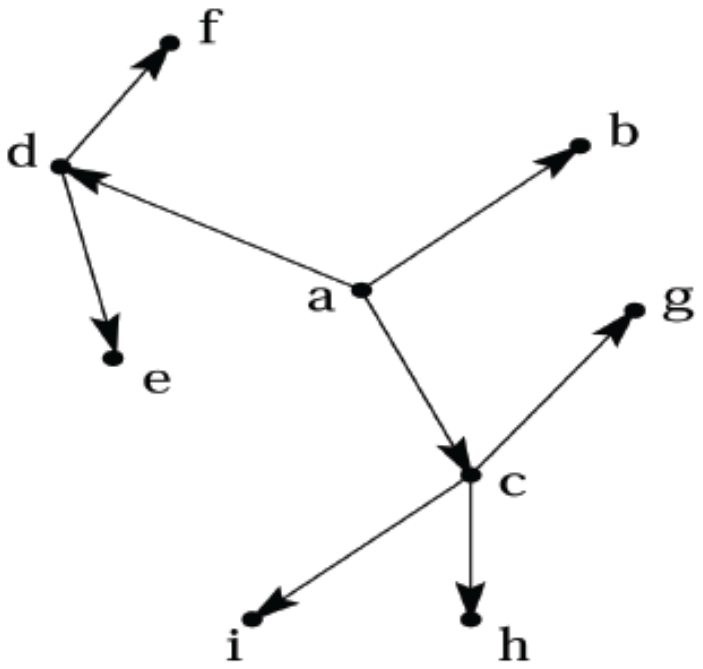
ALBERO

Sia $R \subseteq S \times S$ dove $S = \{a, b, c, d, e, f, g, h, i\}$

$$R = \{\langle a, b\rangle, \langle a, c\rangle, \langle a, d\rangle, \langle c, g\rangle, \langle c, h\rangle, \langle c, i\rangle, \langle d, e\rangle, \langle d, f\rangle, \}.$$



ALBERI



Esempio di grafo che ha struttura di albero

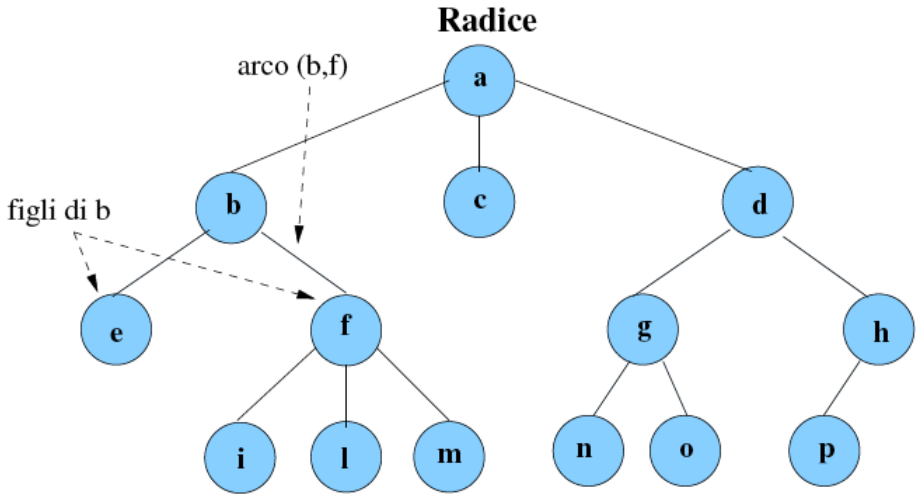
CAMMINO SU UN ALBERO

Un **cammino dal nodo** i **al nodo** j è la sequenza di archi che devono essere attraversati per raggiungere il nodo j partendo dal nodo i

Ogni nodo y che si trova sul cammino tra r e x è un **ascendente** di x; viceversa, x è un **discendente** di y

- r è l'unico nodo che non ha ascendenti

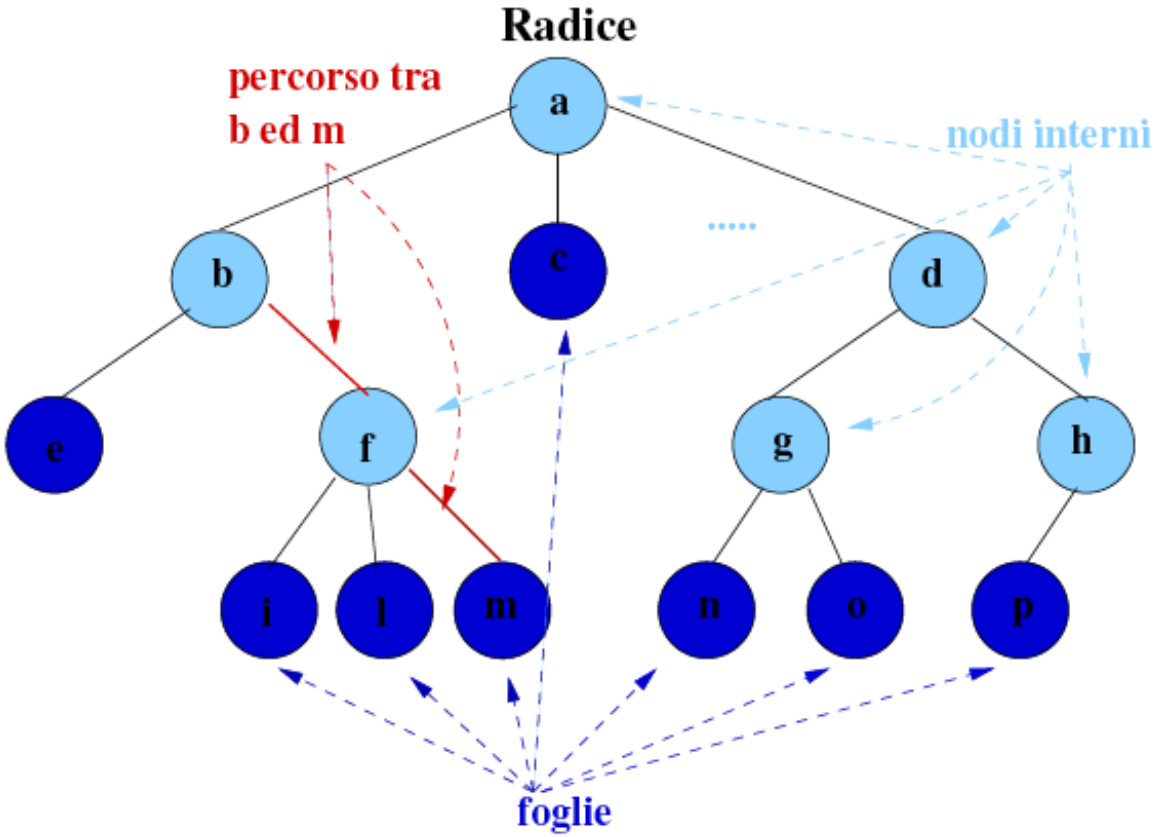
Se l'ultimo arco del cammino da r a x è (y, x), allora y è il **padre** di x, e x è **figlio** di y



FOGLIE, NODI INTERNI, PERCORSI

Un nodo che non ha figli si dice **nodo foglia**

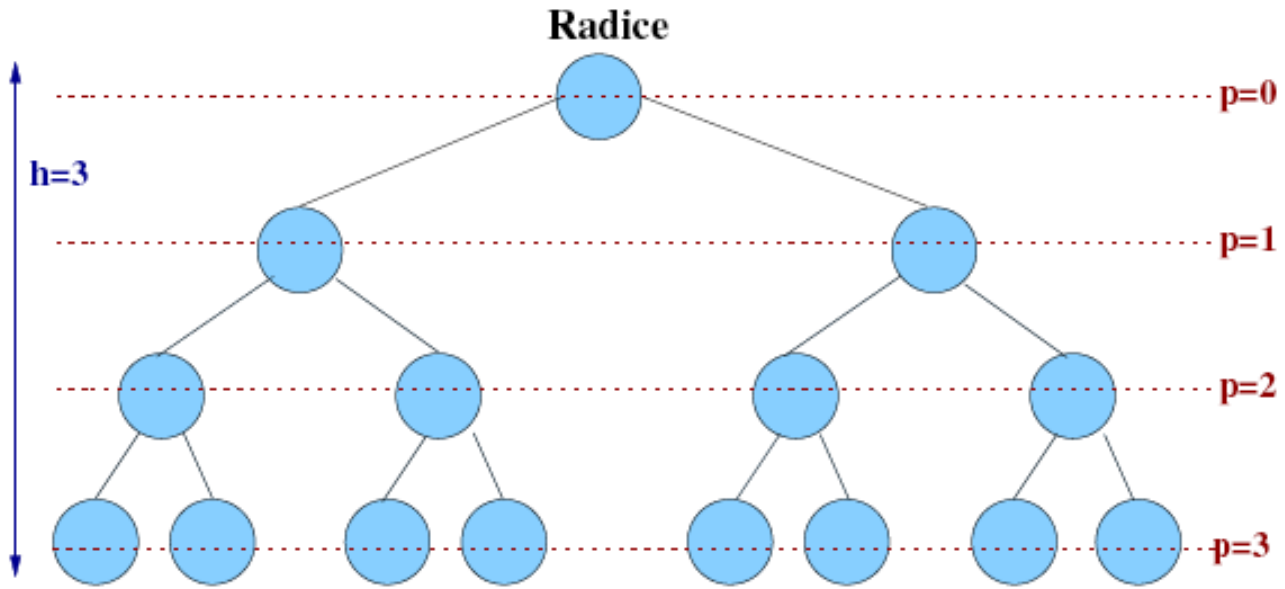
Un nodo si dice **nodo interno** se ha almeno un figlio.



PROFONDITA' E ALTEZZA

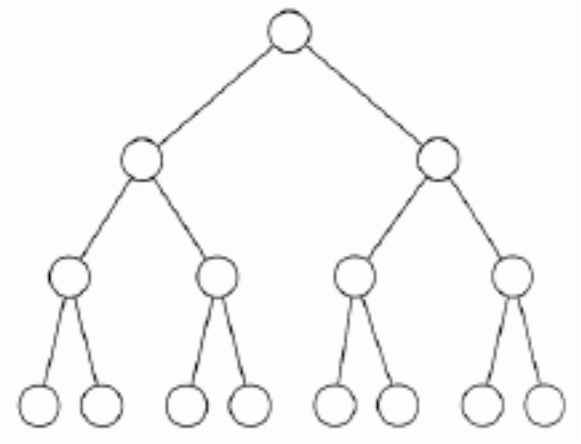
La **profondità di un nodo** x è la lunghezza del percorso per andare da r a x.

L'**altezza dell'albero** è la profondità massima che può avere un nodo dell'albero.



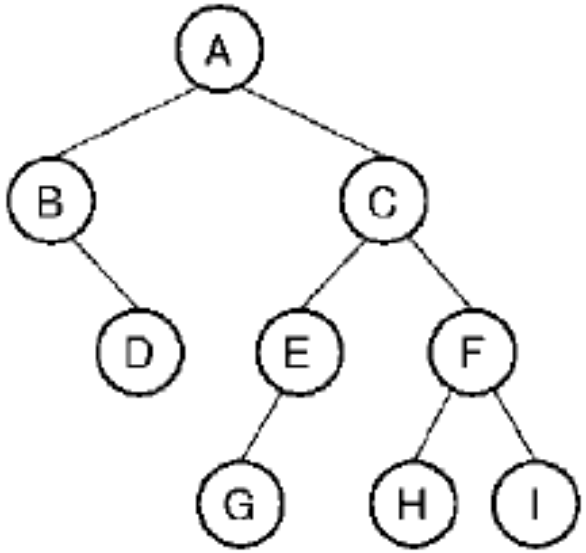
ALBERO BINARIO

Un *albero binario* è un albero in cui ogni nodo diverso dalle foglie ha al massimo due figli ordinati, detti *figlio sinistro* e *figlio destro*.



ALBERO BINARIO

E' una struttura costituita da un insieme finito di elementi detti nodi. Quest'insieme è vuoto oppure è costituito da una radice e da due alberi binari (sottoalbero sinistro e destro) disgiunti tra loro e dalla radice.



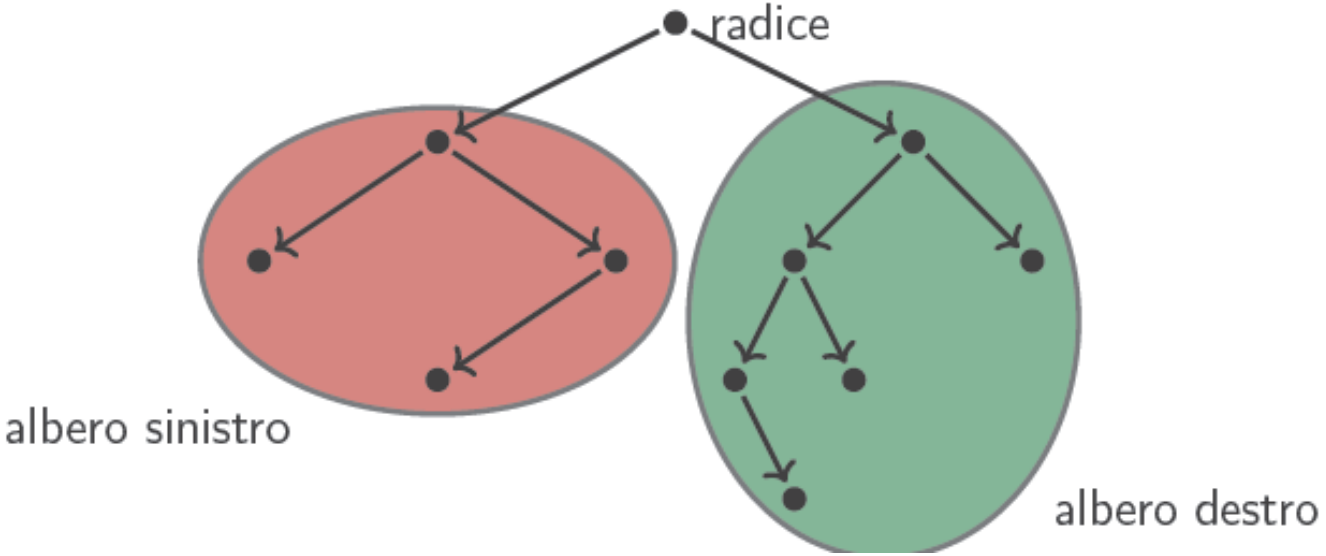
ALBERi BINARI: ESEMPIO

ALBERO BINARIO

Struttura

Possiamo vedere un albero binario come una **struttura ricorsiva** composta per:

- un nodo (radice)
- un albero binario sinistro (eventualmente vuoto) e
- un albero binario destro (eventualmente vuoto)

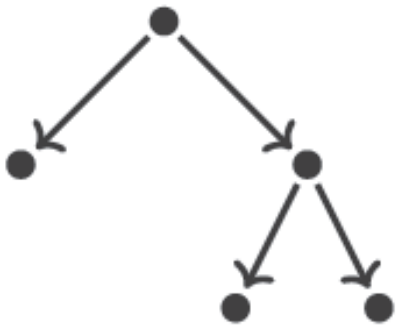


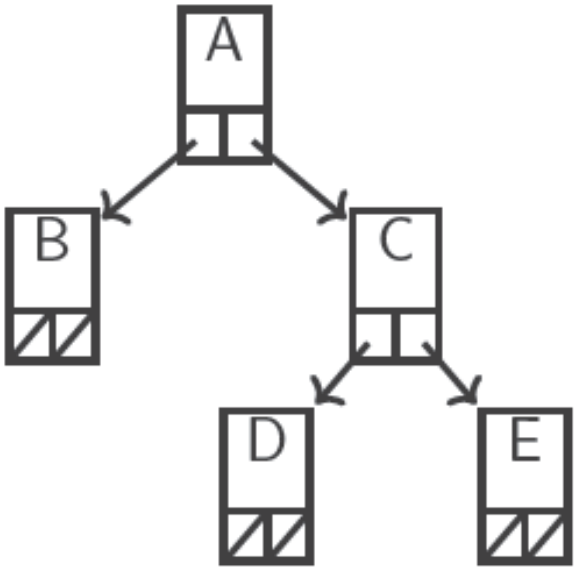
ALBERO BINARIO

Rappresentazione

Possiamo rappresentare un albero binario sia:

- come una collezione di nodi, dove la radice è segnalata, e ogni nodo ha due puntatori (alle radici degli alberi sinistro e destro), oppure
- come una tabella con $2^{n+1} - 1$ righe, dove n è la altezza dell'albero





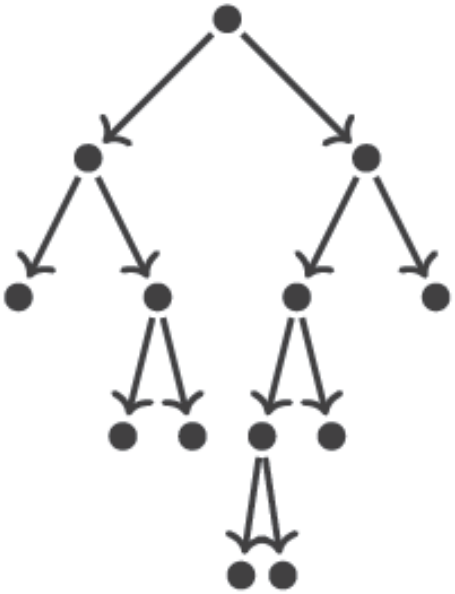
1	A	1
2	B	1
3	C	1
4	0	0
5	0	0
6	D	1
7	E	1

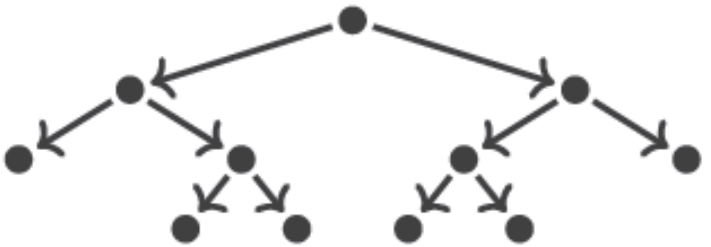
ALBERO BINARIO

Tipi di alberi binari

Un albero binario è

- **pieno** se ogni nodo interno ha due figli
- **completo** se ha altezza n, ad ogni profondità $i, 0 \leq i < n$ ci sono 2^i nodi, e l'ultimo livello è riempito da sinistra a destra
  (in rappresentazione tabulare, i nodi vuoti sono soltanto sulle ultime righe)



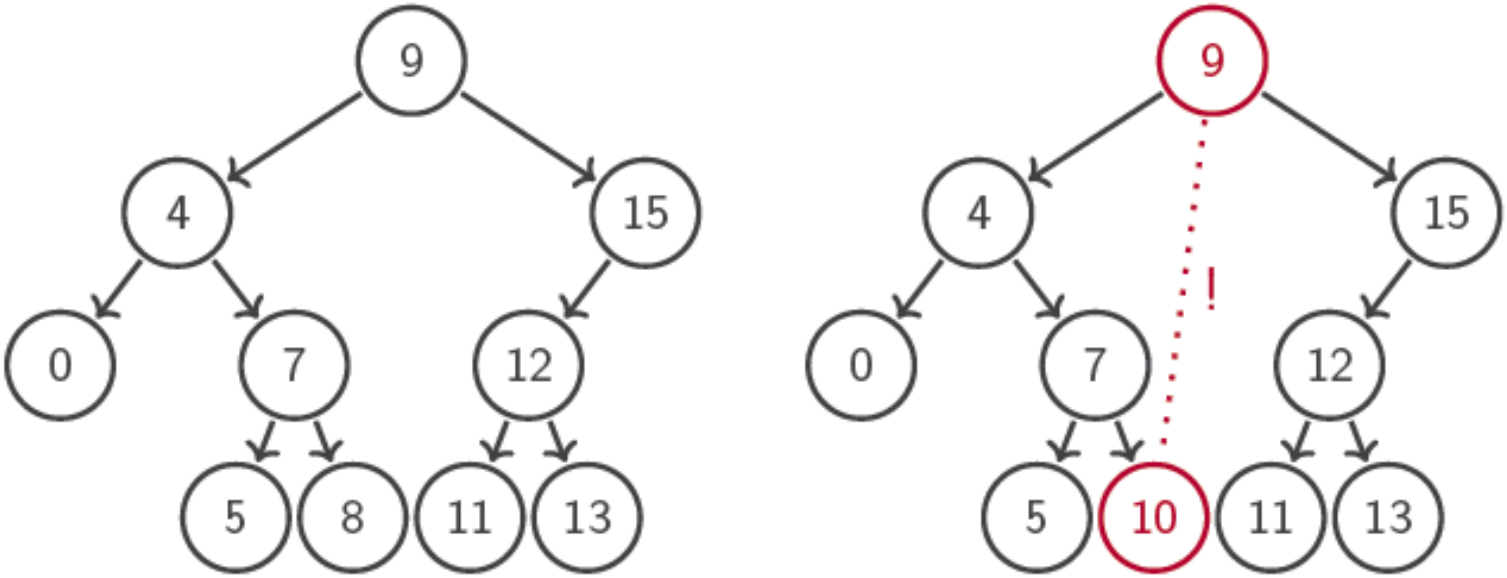


ALBERO BINARIO

Albero binario di ricerca

Un **albero di ricerca** è un albero binario $G = (V, E)$ con $V \subseteq \mathbb{Z}$ $(*)$ tale che per ogni nodo z:

- ogni nodo dell'albero sinistro di z è minore a z e
- ogni nodo dell'albero destro di z è maggiore a z

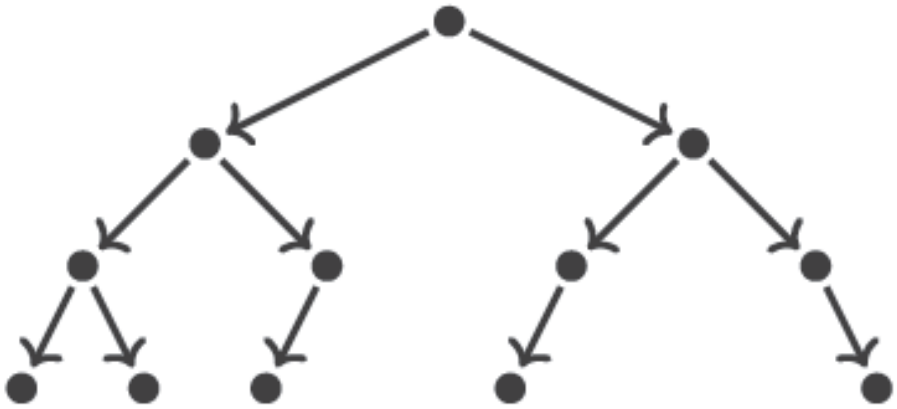


Utili per rappresentare liste ordinate dinamiche

ALBERO BINARIO

Albero bilanciato

Un albero binario (di ricerca) è **bilanciato** se per ogni nodo v, la differenza fra il numeri di nodi nell'albero sinistro di v e nell'albero destro di v è al massimo 1

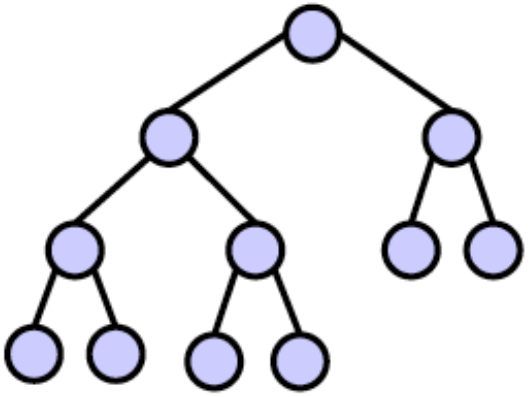


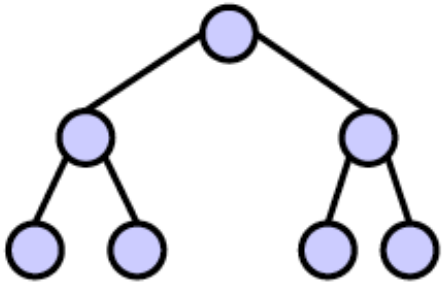
ALBERO BINARIO

Definizione 1.2 *(Alberi binari completi)* Il **grado** di un nodo è il numero di figli di quel nodo. Un albero binario si dice **completo** se:

1. tutte le foglie hanno la stessa altezza h;
2. tutti i nodi interni hanno grado 2 (hanno esattamente due figli).

Definizione 1.3 *(Alberi binari quasi completi)* Un albero binario si dice **quasi completo** se tutti i livelli, tranne al più l'ultimo, sono completi; nell'ultimo livello possono mancare alcune foglie consecutive a partire dall'ultima foglia a destra.





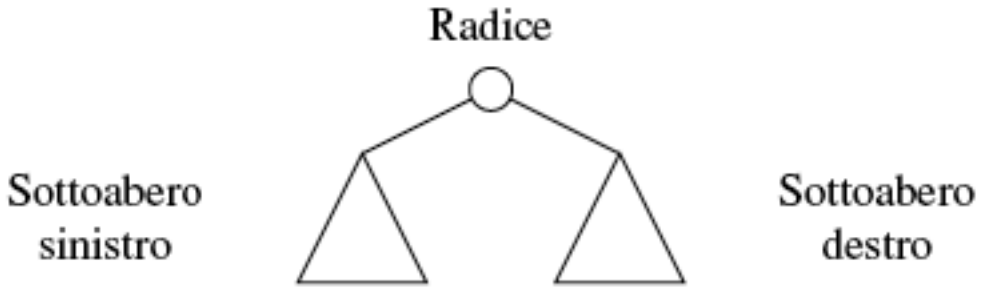
ALBERO BINARIO

Definizione ricorsiva di albero binario

Un albero binario è:

- un albero vuoto

- costituito da un nodo, detto radice e da due sottoalberi binari disgiunti, chiamati sottoalbero sinistro e sottoalbero destro



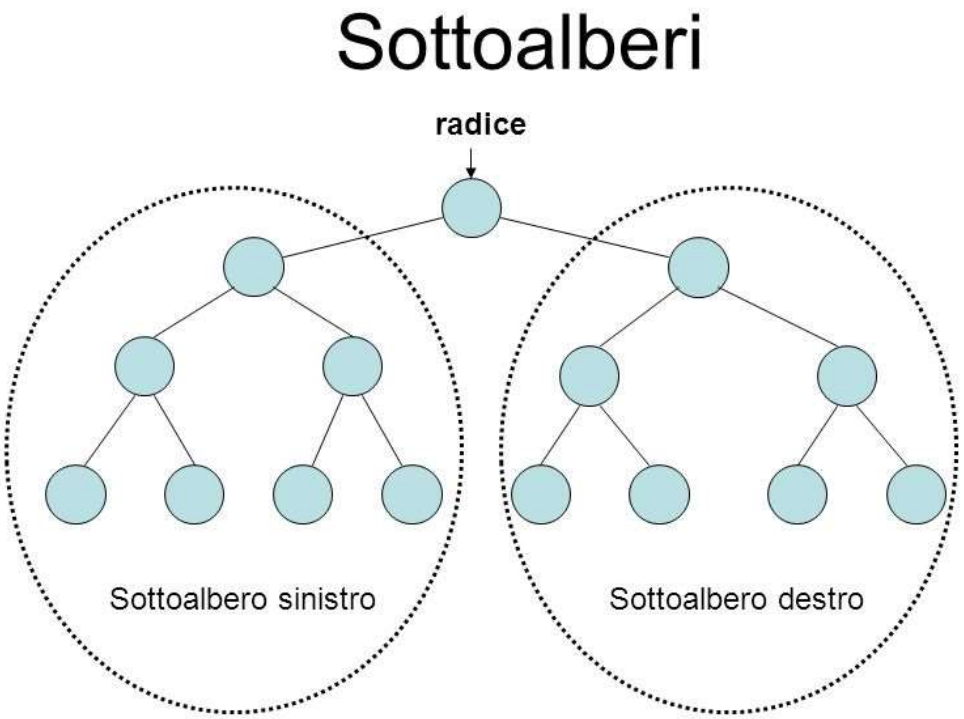
ALBERO BINARIO: PROPRIETA'

Un *albero binario* con N nodi interni ha N+1 nodi esterni.

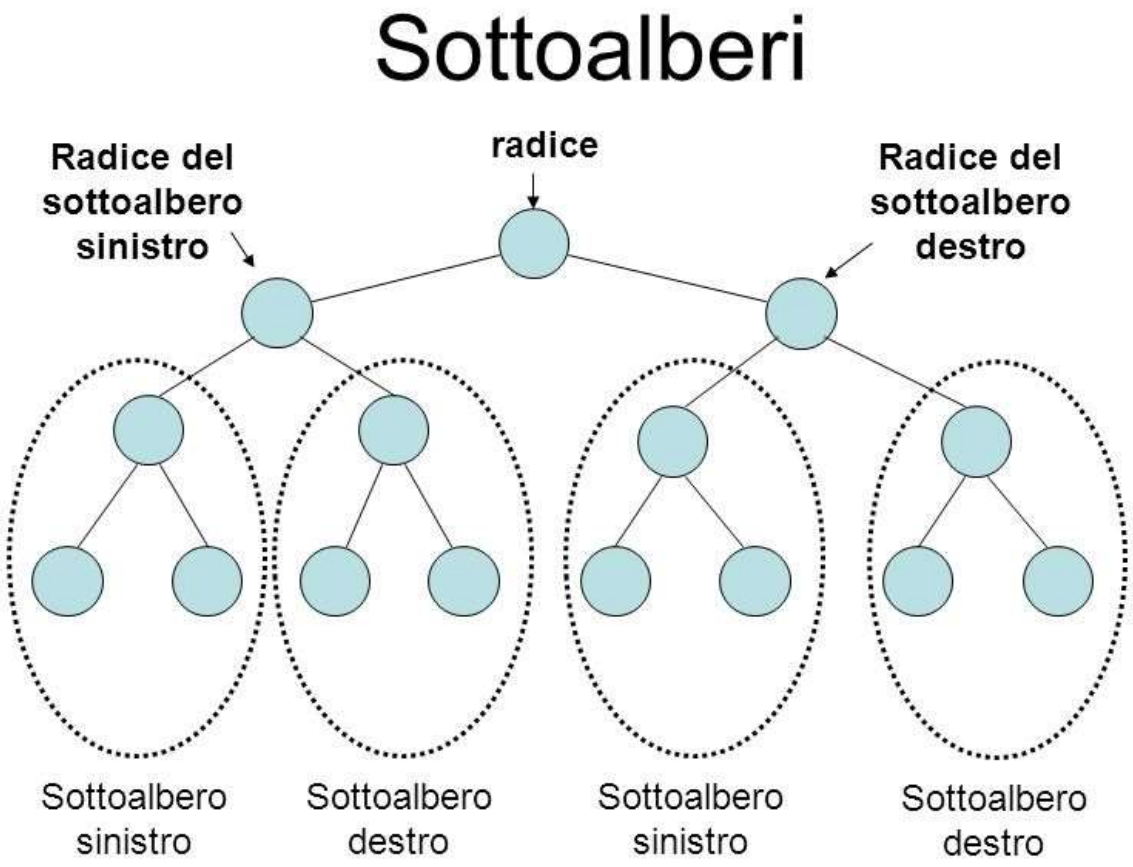
Un *albero binario completo* di altezza h ha $2^{h+1} - 1$ nodi.

Sia T un albero binario quasi completo di altezza h.
Allora: $2^h \leq \#_{nodi}(T) \leq 2^{h+1} - 1$.

ALBERO BINARIO

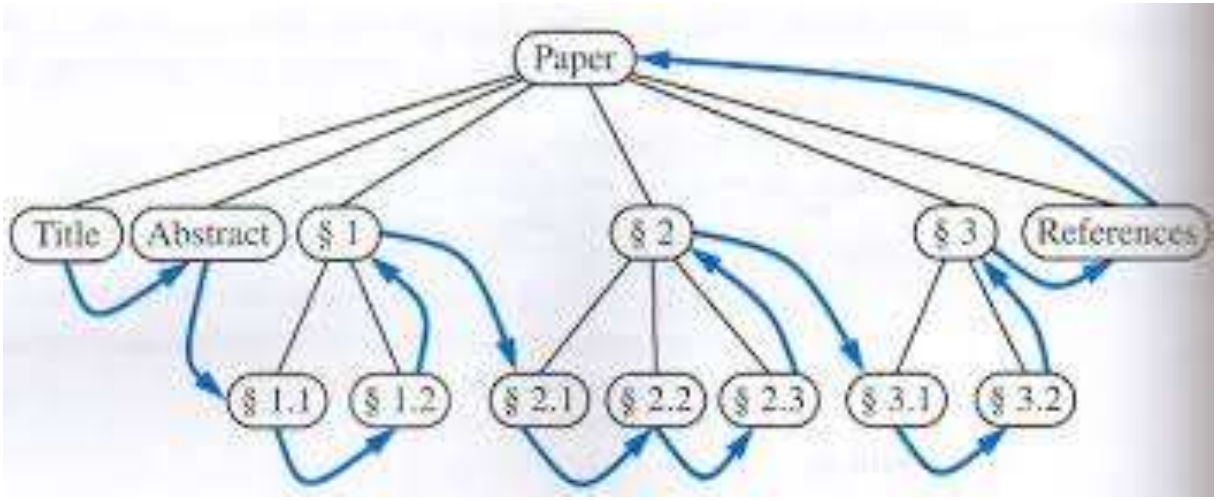


ALBERO BINARIO



ATTRAVERSAMENTO DI UN ALBERO BINARIO

- Attraversamento di un albero: visita di tutti i suoi nodi secondo un particolare ordine
  - un attraversamento che elenca tutti i nodi esattamente una volta si chiama **enumerazione** dei nodi dell'albero



ATTRAVERSAMENTO DI UN ALBERO BINARIO

Tipi di Attraversamento

Distinguiamo fra due tipi di attraversamento:

- in **profondità** esplora ogni ramo dell'albero fino in fondo (figli prima di fratelli)

- in **larghezza** esplora prima i nodi più vicini alla radice (fratelli prima di figli)

Enumerazione in profondità (I)

Ci sono tre tipi diversi di ordine in profondità, basati su **quando** enumeriamo un elemento

Si usa la notazione:

- **L** per sinistra (left)
- **R** per destra (right)
- **V** per enumerazione (visit)

Enumerazione in profondità (II)

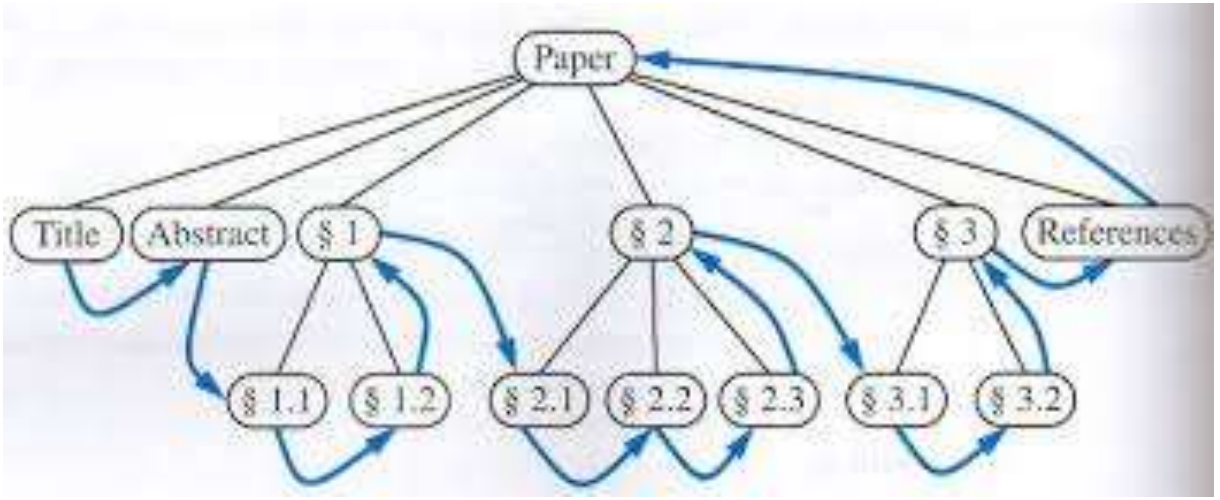
I tre ordini di enumerazione in profondità sono:

- in **preordine**: si visita un nodo prima di visitare i figli (VLR)
- in **ordine**: si visita l'albero sinistro, poi il nodo, por l'albero destro (LVR)
- in **postordine**: si visitano prima i figli e poi il nodo (LRV)

Implementazione: una **pila** contiene gli elementi da esplorare (LIFO)

ATTRAVERSAMENTO DI UN ALBERO BINARIO

- Attraversamento di un albero: visita di tutti i suoi nodi secondo un particolare ordine
  - un attraversamento che elenca tutti i nodi esattamente una volta si chiama **enumerazione** dei nodi dell'albero



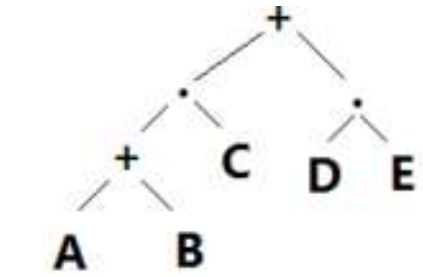
ATTRAVERSAMENTO DI UN ALBERO BINARIO

Gli alberi binari possono essere visitati (o attraversati) in diversi modi:

- Visita in **Preordine**: prima si visita il nodo e poi i suoi sottoalberi

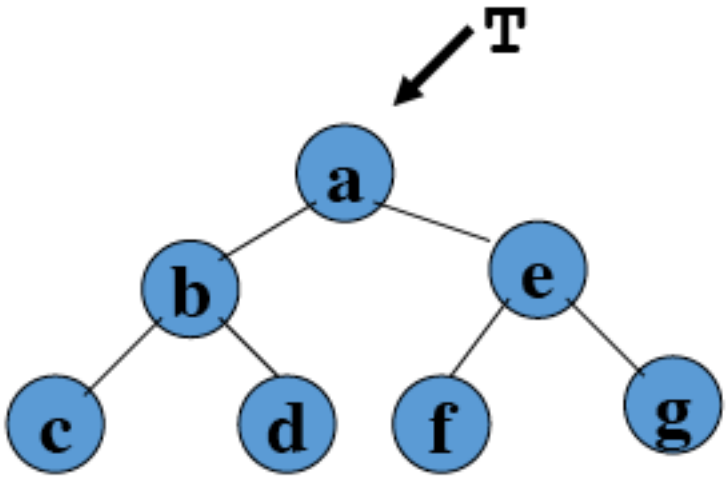
- Visita **Inordine**: prima si visita il sottoalbero sinistro, poi il nodo e infine il sottoalbero destro

- Visita in **Postordine** : prima si visitano i sottoalberi, poi il nodo.



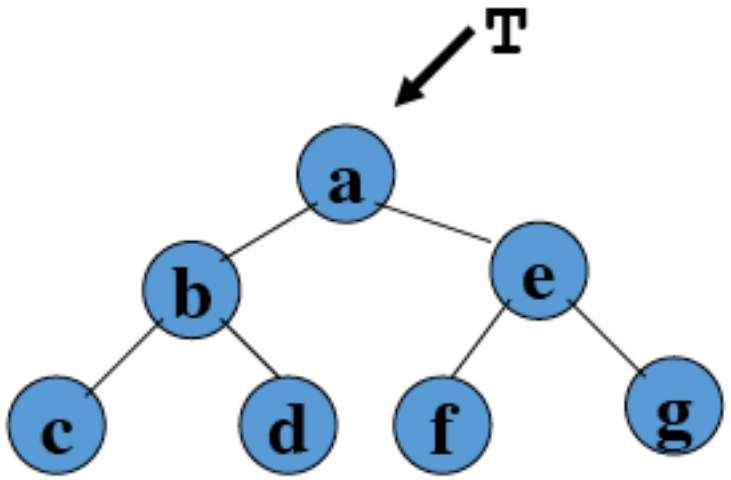
+·+ABC·DE (pre order)
A+B·C+D·E (in order)
AB+C·DE·+ (post order)

ATTRAVERSAMENTO DI UN ALBERO BINARIO
PREORDER



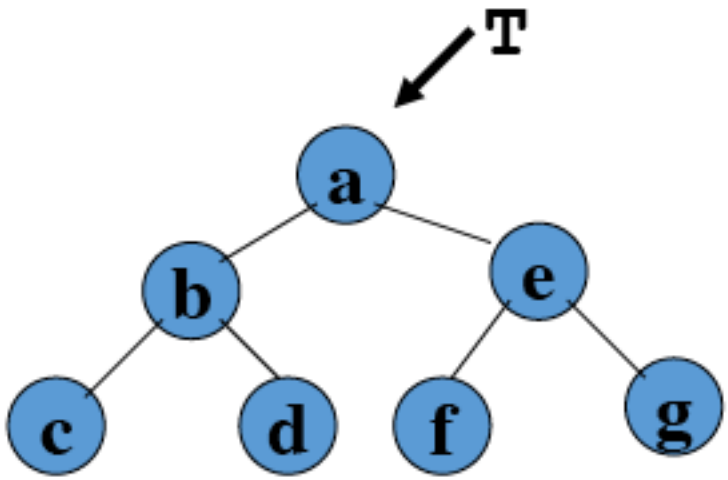
Sequenza: a b c d e f g

ATTRAVERSAMENTO DI UN ALBERO BINARIO INORDER



Sequenza: c b d a f e g

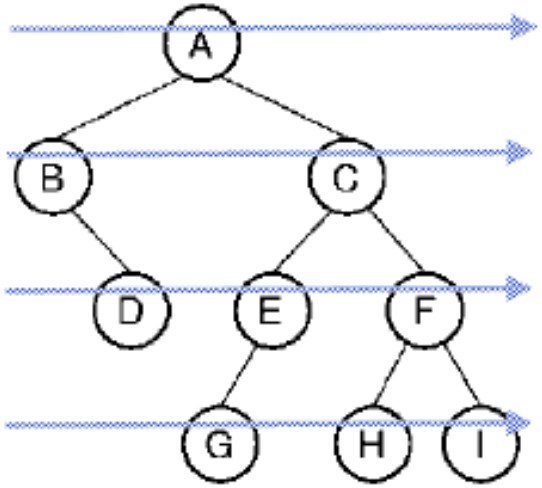
ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER



Sequenza: c d b f g e a

ATTRAVERSAMENTO IN AMPIEZZA

- Attraversamento in ampiezza (breadth-first): visita di ciascun nodo per livelli



ABCDEFGHI

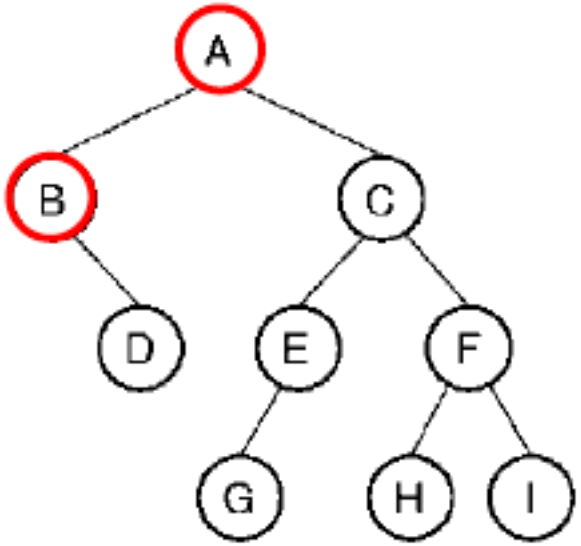
STRUTTURE RELAZIONALI, GRAFI E ORDINAMENTI

(parte 4)

END

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

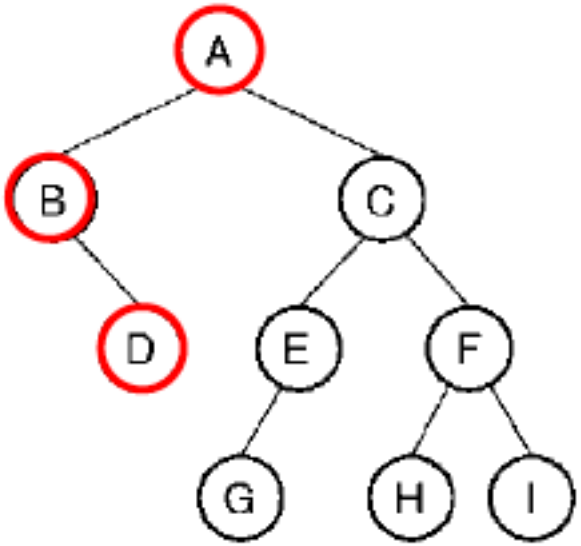
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

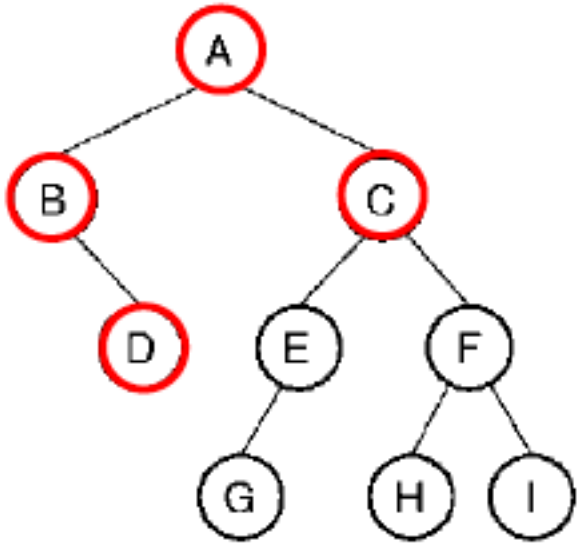
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

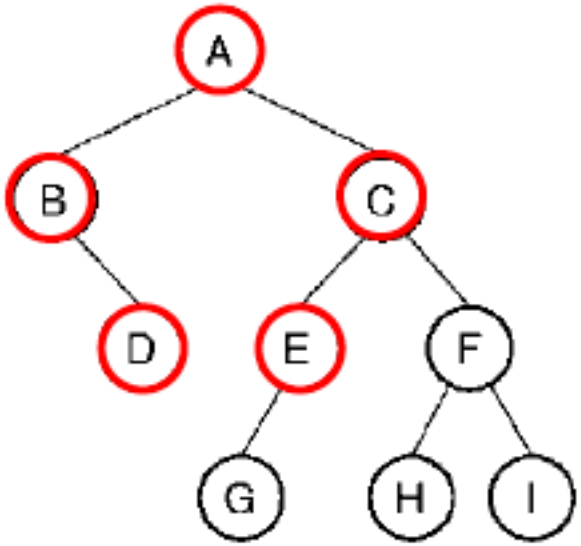
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

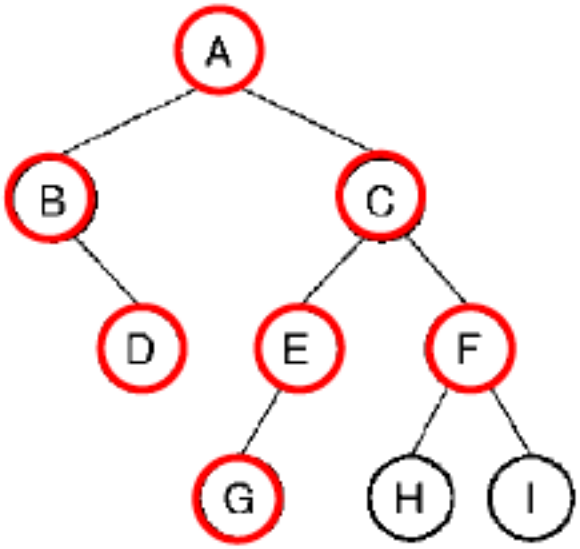
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

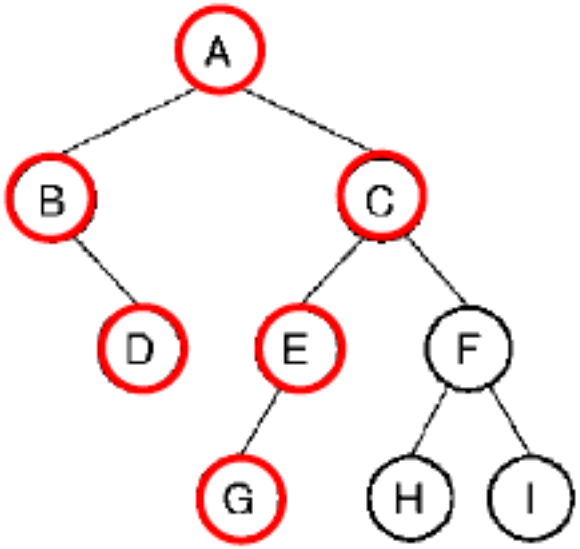
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

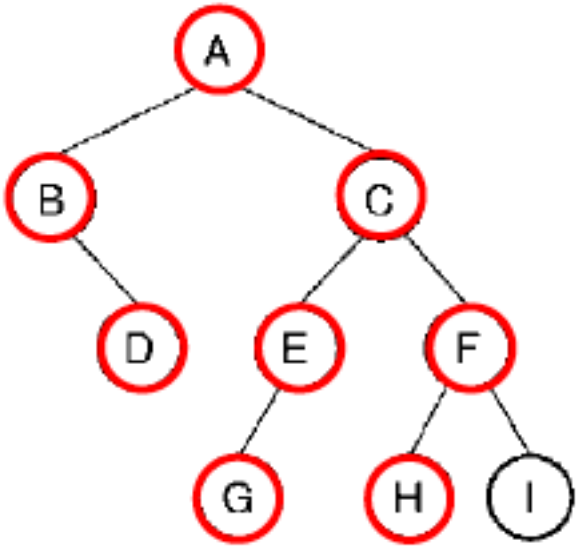
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

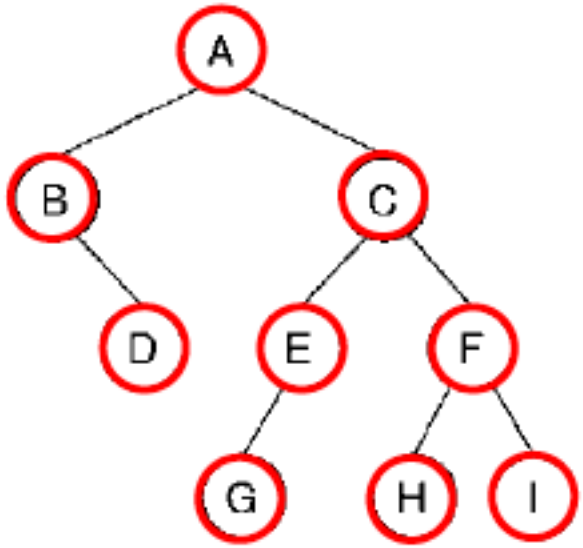
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO PREORDER

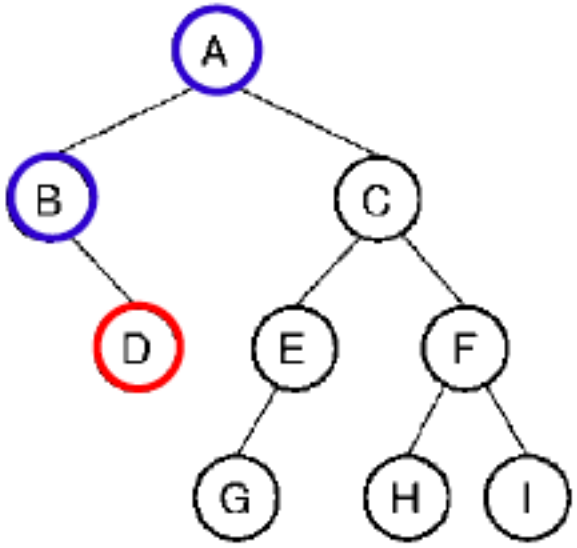
- VLR



ABDCEGFHI

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

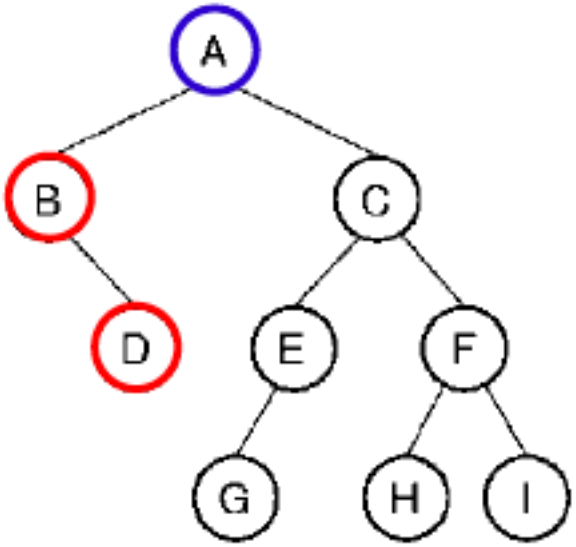
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

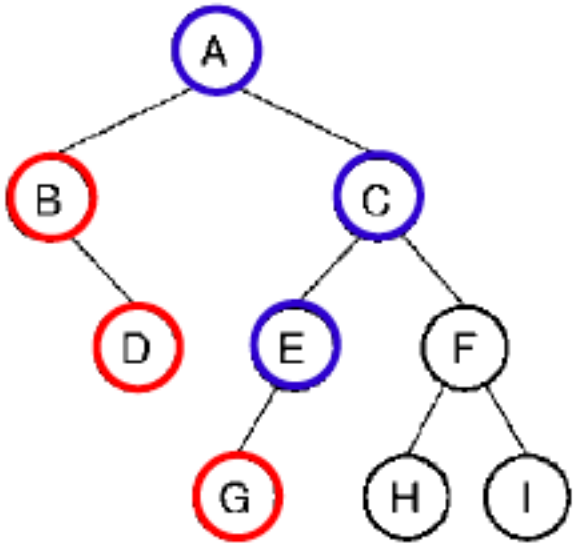
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

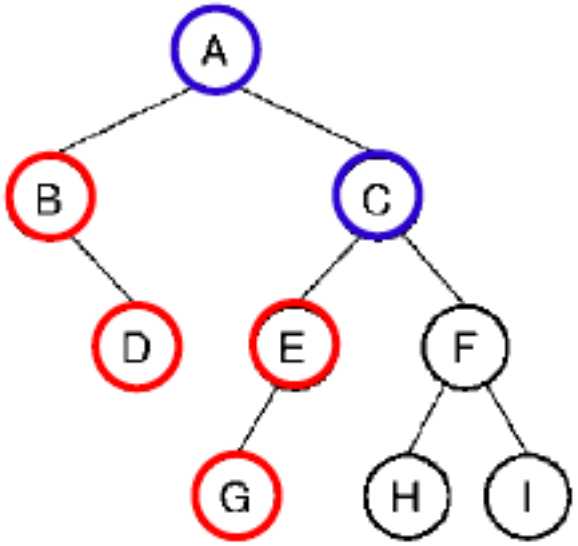
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

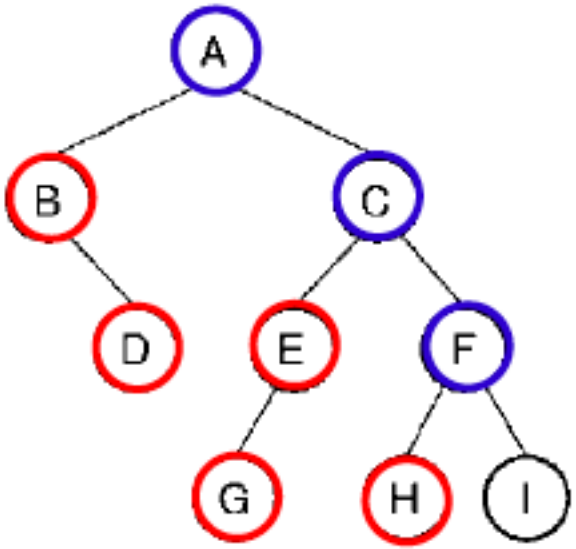
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

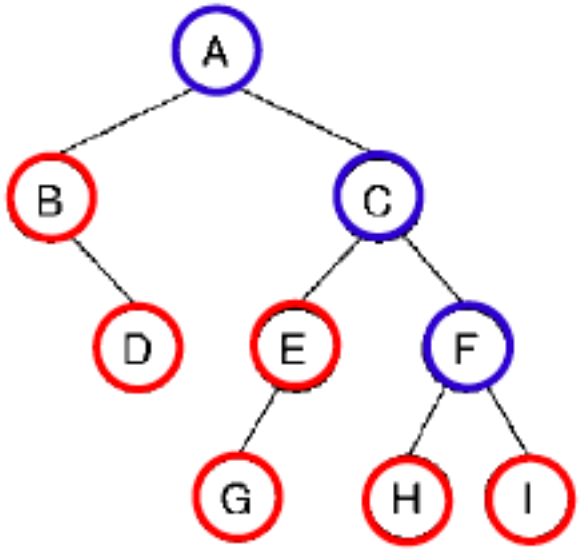
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

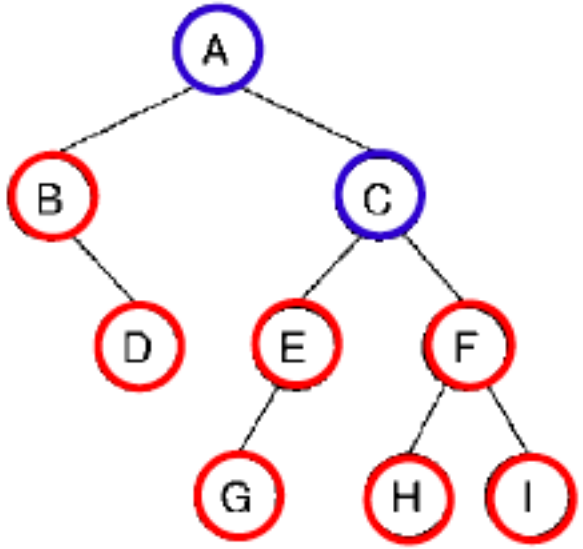
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

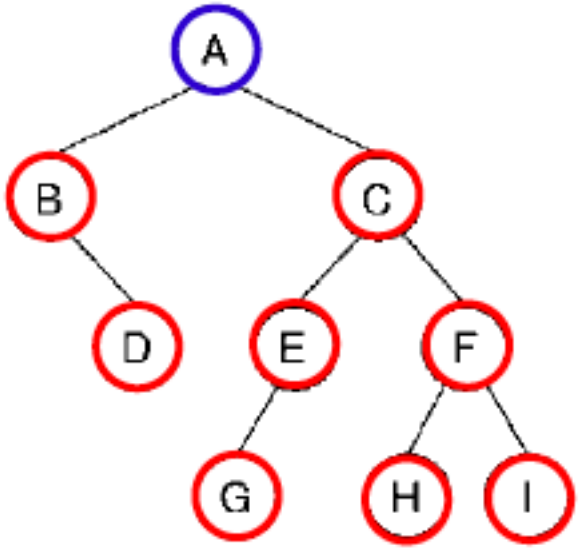
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

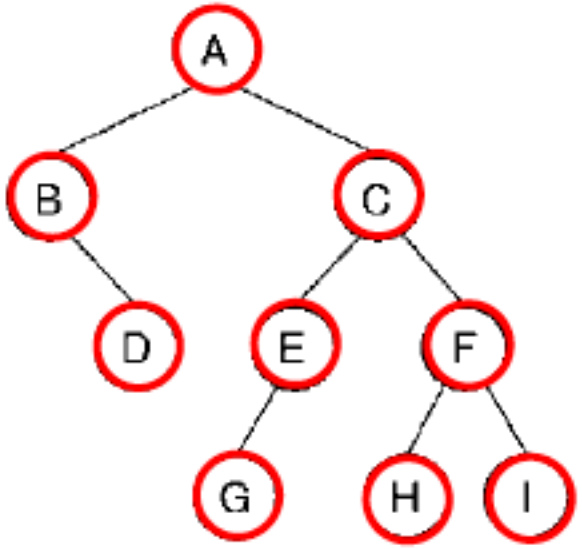
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO POSTORDER

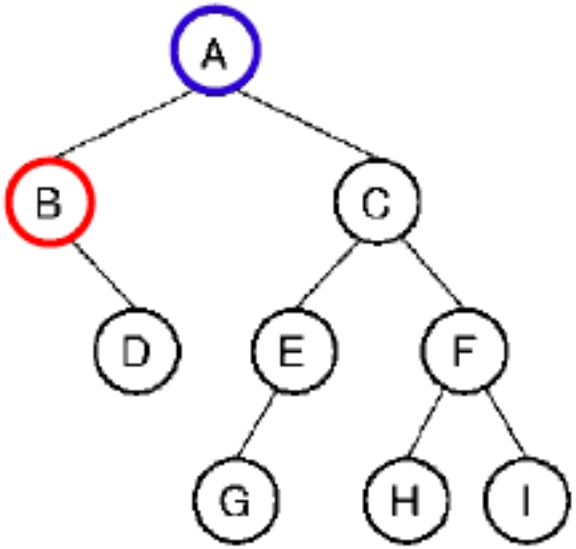
- LRV



DBGEHIFCA

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

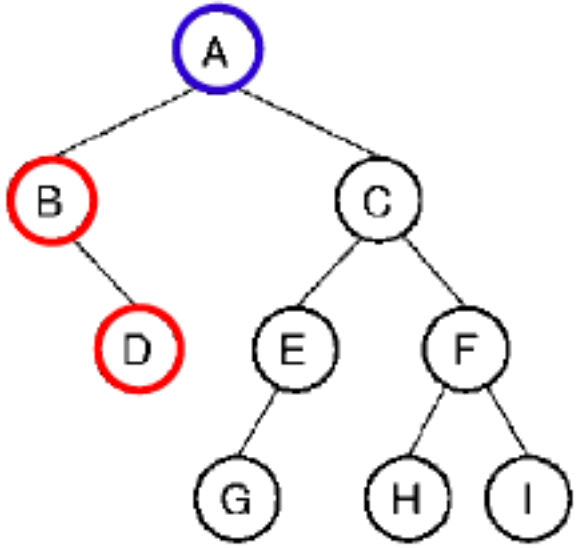
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

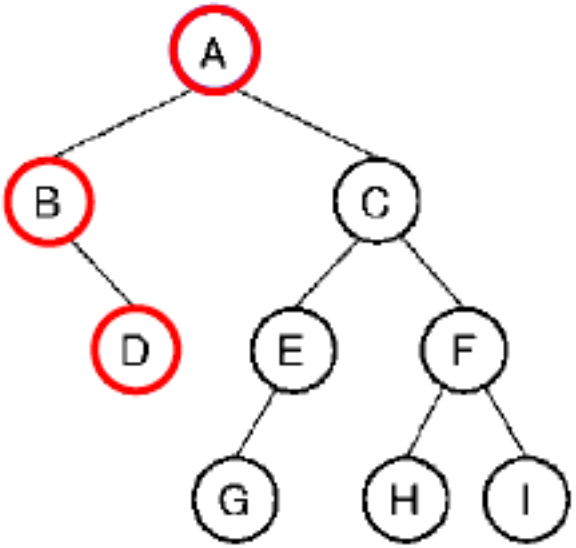
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

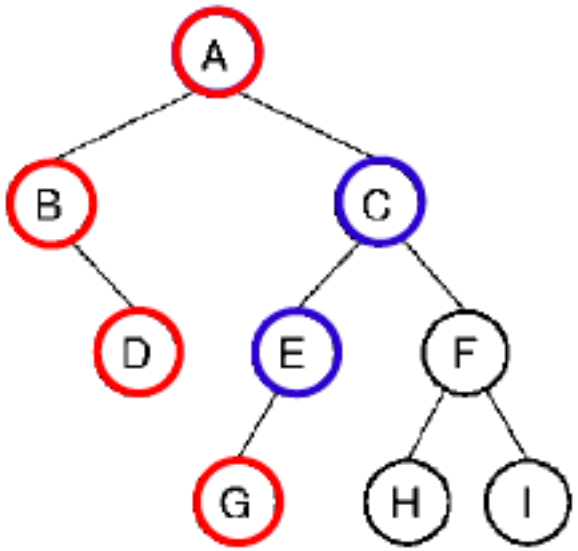
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO ORDER

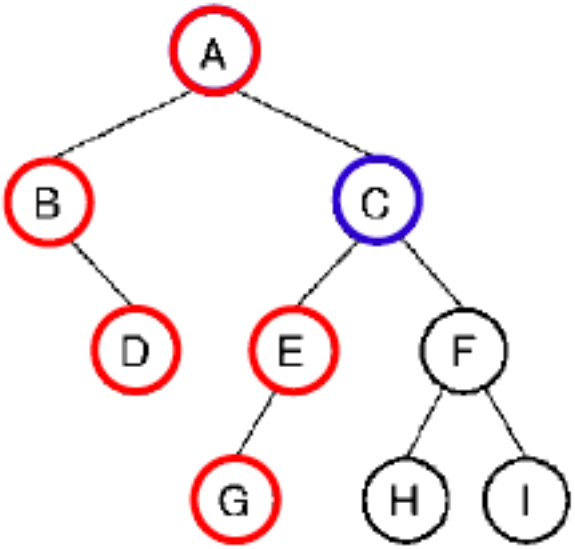
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

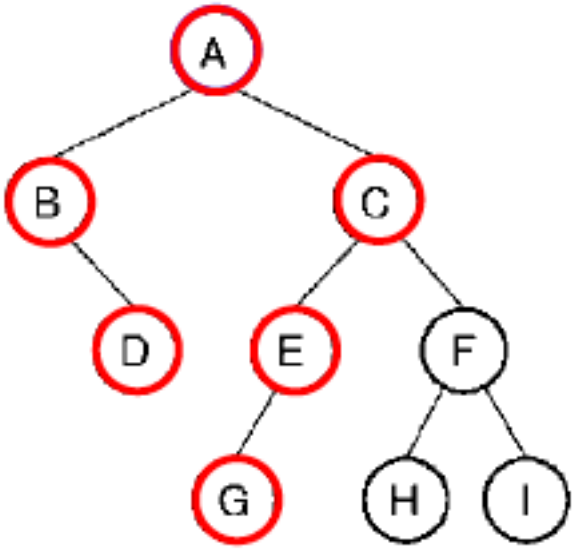
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

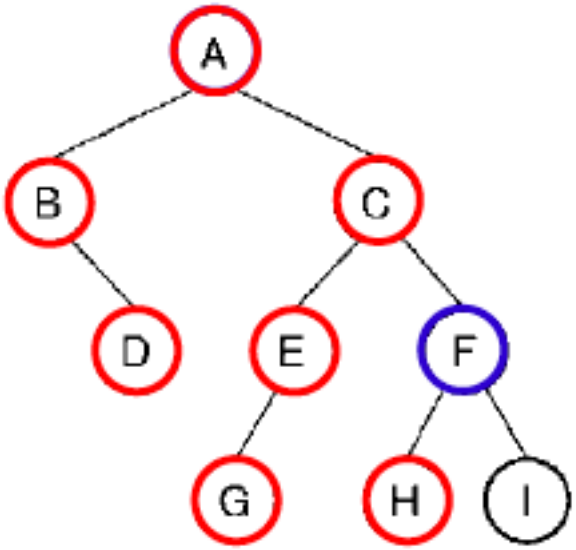
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

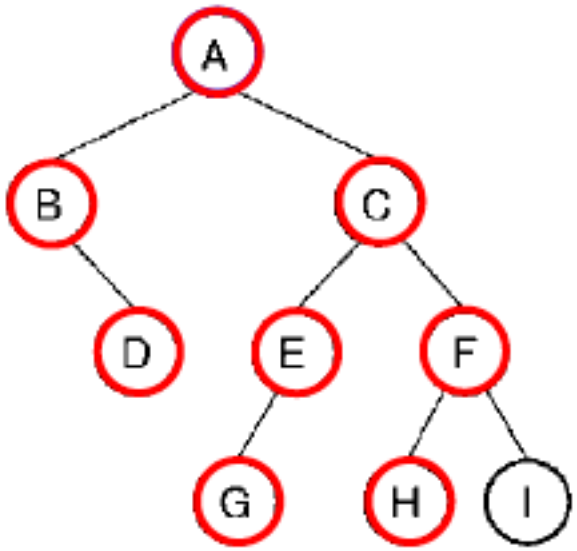
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

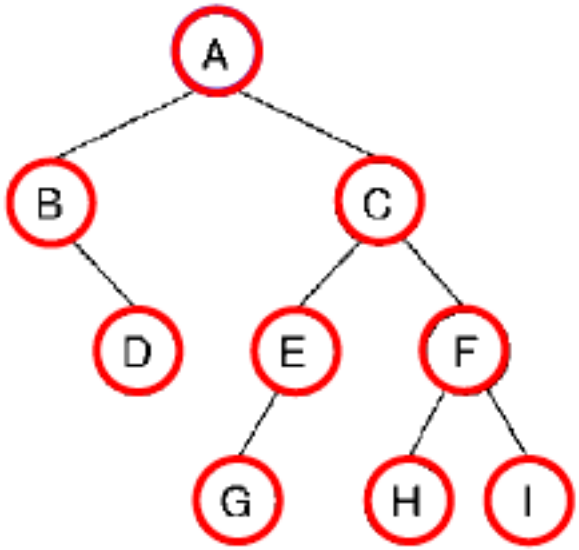
- LVR



BDAGECHFI

ATTRAVERSAMENTO DI UN ALBERO BINARIO
ORDER

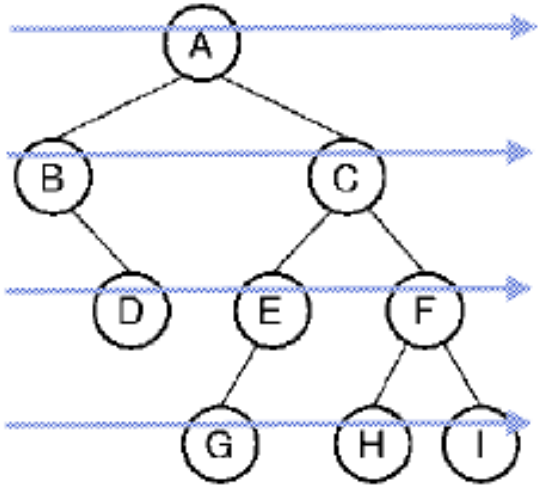
- LVR



BDAGECHFI

ATTRAVERSAMENTO IN AMPIEZZA

- Attraversamento in ampiezza (breadth-first): visita di ciascun nodo per livelli



ABCDEFGHI

- La realizzazione è semplice se si utilizza una coda

STRUTTURE RELAZIONALI, GRAFI E ORDINAMENTI

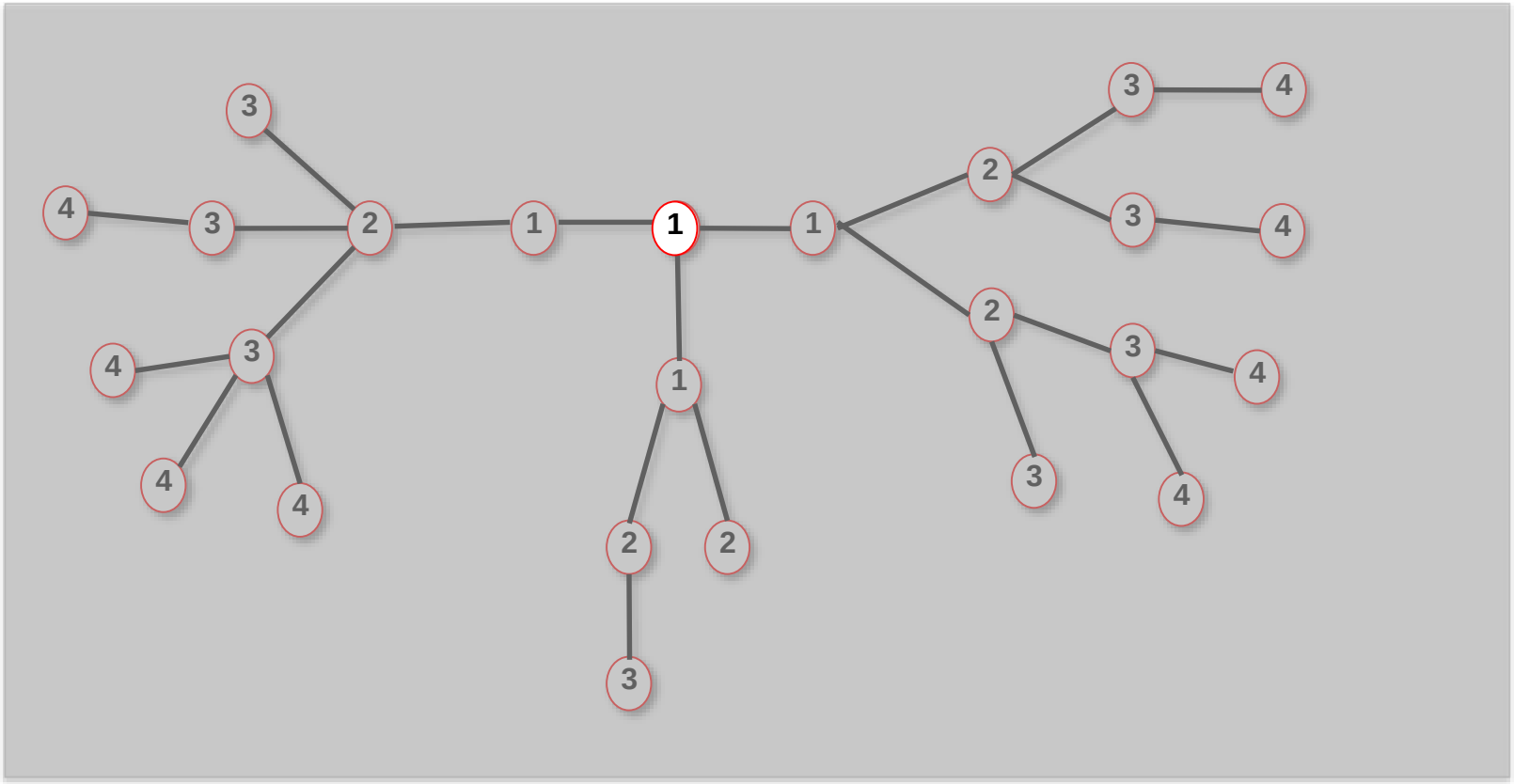
(parte 4)

END

FINDING DISTANCES: BREADTH FIRST SEATCH

Distance between node 1 and node 4:

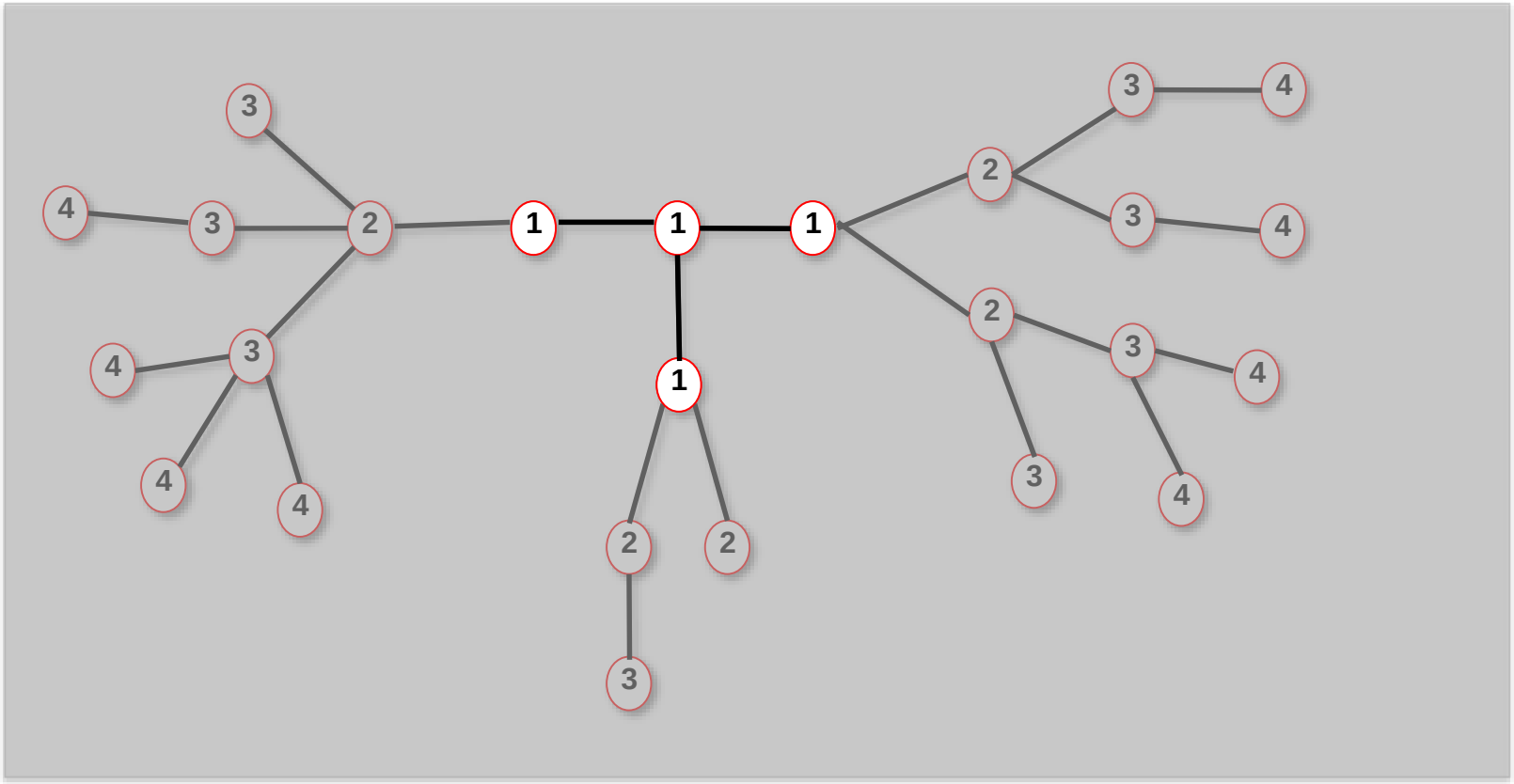
1.Start at 1.



FINDING DISTANCES: BREADTH FIRST SEATCH

Distance between node 1 and node 4:

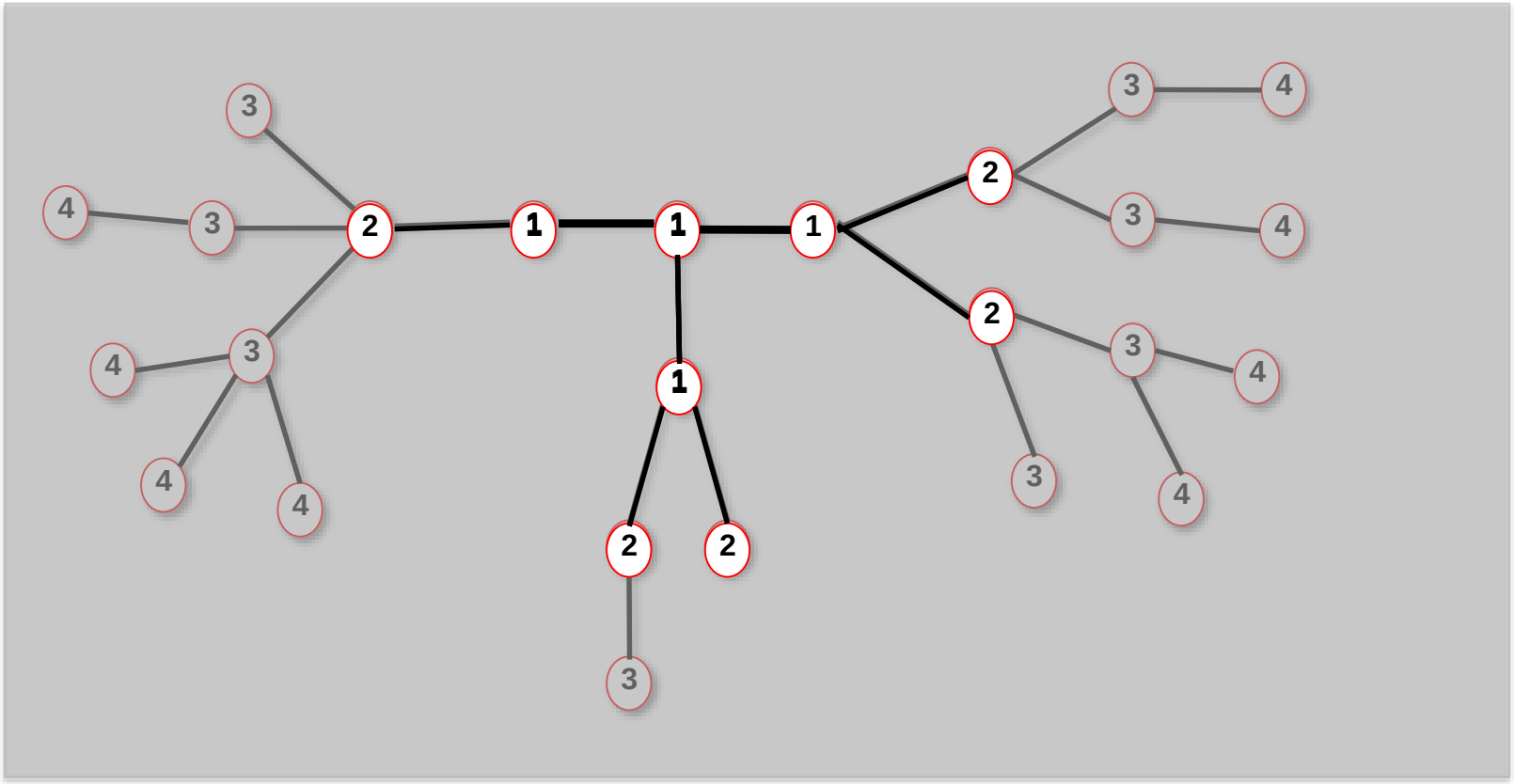
1.Start at 1.
2.Find the nodes adjacent to 1. Mark them as at distance 1. Put them in a queue.



FINDING DISTANCES: BREADTH FIRST SEATCH

Distance between node 1 and node 4:

1.Start at 1.
2.Find the nodes adjacent to 1. Mark them as at distance 1. Put them in a queue.
3.Take the first node out of the queue. Find the unmarked nodes adjacent to it in the graph. Mark them with the label of 2. Put them in the queue.



FINDING DISTANCES: BREADTH FIRST SEATCH

Distance between node 1 and node 4:

1.Repeat until you find node 4 or there are no more nodes in the queue.
2.The distance between 1 and 4 is the label of 4 or, if 4 does not have a label, infinity.