



# Università degli Studi dell'Aquila



Dipartimento di Ingegneria e Scienze  
dell'Informazione e Matematica

Università degli Studi dell'Aquila

Corso di Algoritmi e Strutture Dati con Laboratorio  
**Alberi di ricerca bilanciati**  
(rif. Algoritmi in Java, di R. Sedgewick)

# Prestazioni degli Alberi Binari di Ricerca (BST)

- ▶ Prestazioni variabili da:
  - **logaritmiche**: caso migliore, albero bilanciato
  - **lineari**: caso peggiore, albero degenerare.
- ▶ **Soluzioni**:
  - applicazione periodica di procedure di ribilanciamento
  - imposizione di vincoli sull'albero per limitare lo sbilanciamento.

# Alberi di ricerca 2-3-4

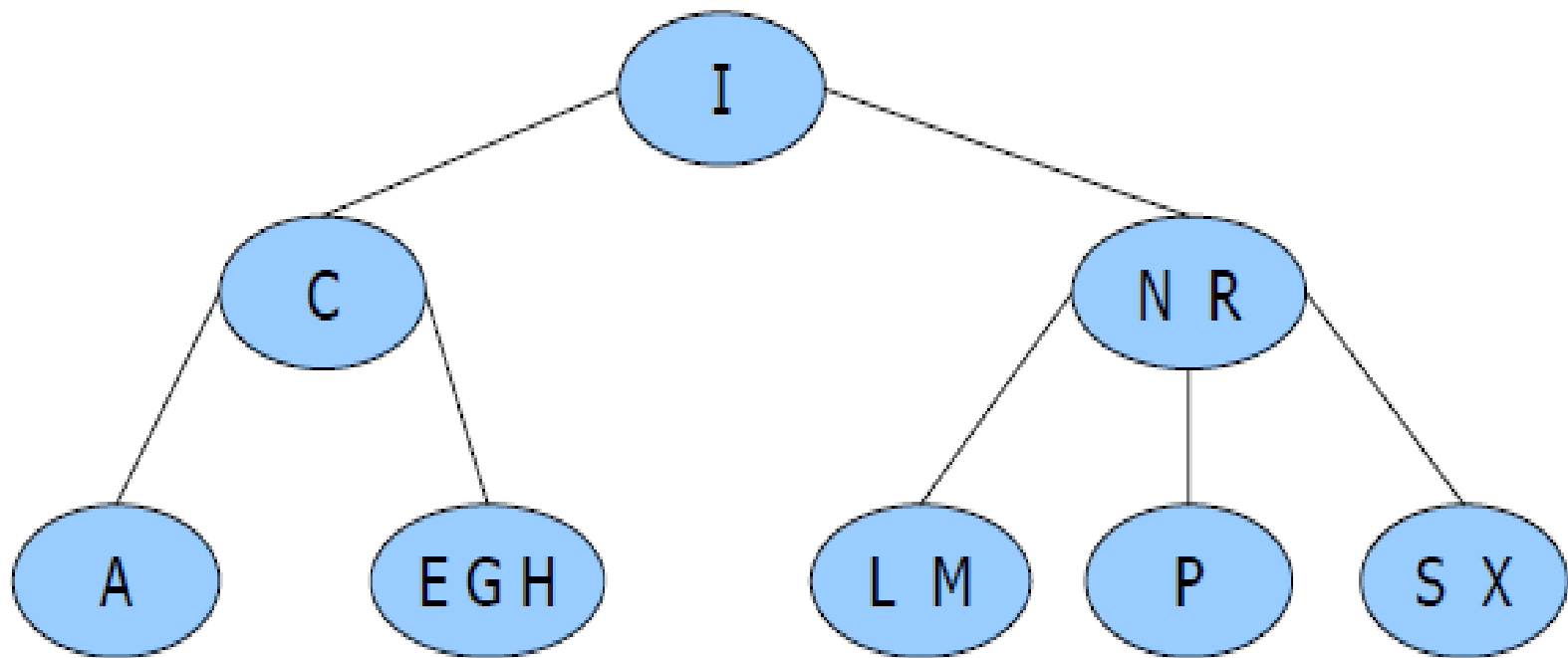
Albero vuoto o con 3 tipi di nodi:

- ▶ **2-nodi**: 1 chiave, sottoalbero sinistro delle chiavi minori, sottoalbero destro delle chiavi maggiori
- ▶ **3-nodi**: 2 chiavi ordinate, sottoalbero sinistro delle chiavi minori di entrambe le chiavi, sottoalbero centrale delle chiavi comprese tra le due, sottoalbero destro delle chiavi maggiori di entrambe le chiavi
- ▶ **4-nodi**: 3 chiavi ordinate, 4 sottoalberi con chiavi con valori che stanno negli intervalli di valori definiti dalle 3 chiavi.

# Alberi di ricerca 2-3-4 bilanciati

- ▶ Si considerano esclusivamente alberi di ricerca 2-3-4 bilanciati:
- ▶ **Bilanciamento**: tutte le foglie hanno uguale distanza dalla radice.
- ▶ BST: alberi 2-3-4 non bilanciati formati da soli 2-nodi.

# Esempio

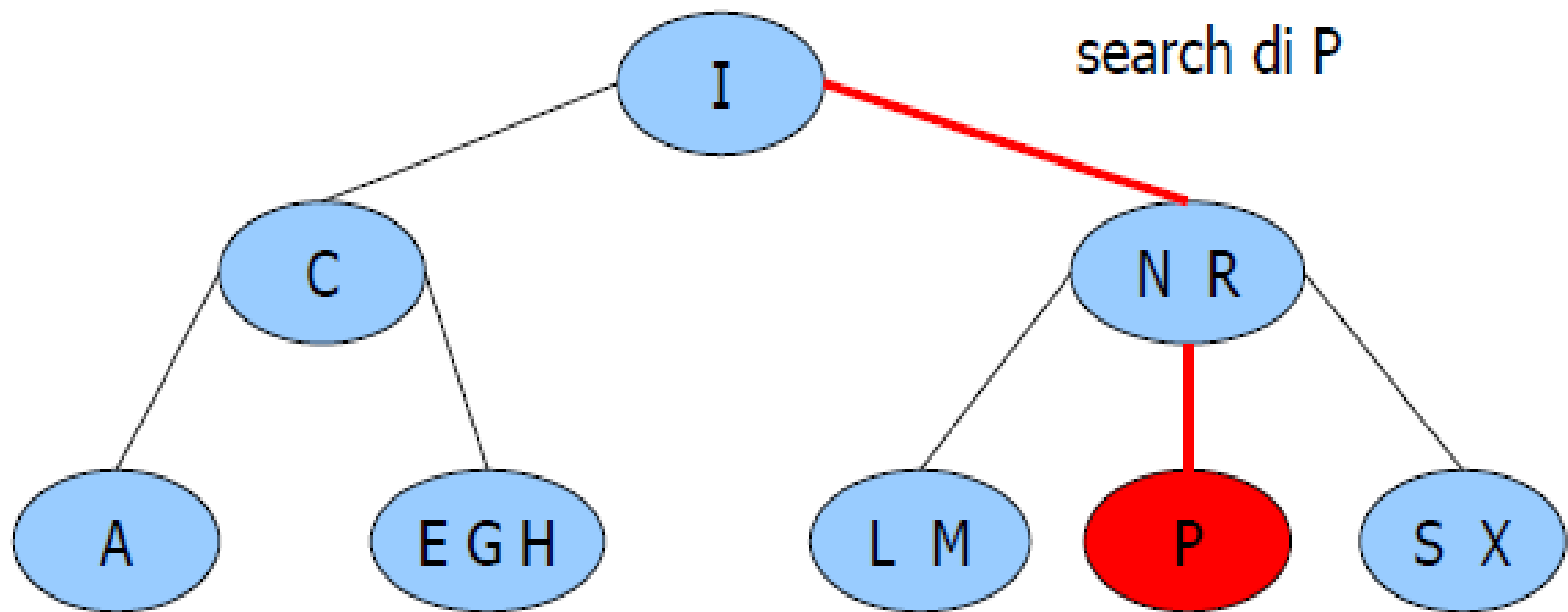


# Ricerca di una chiave

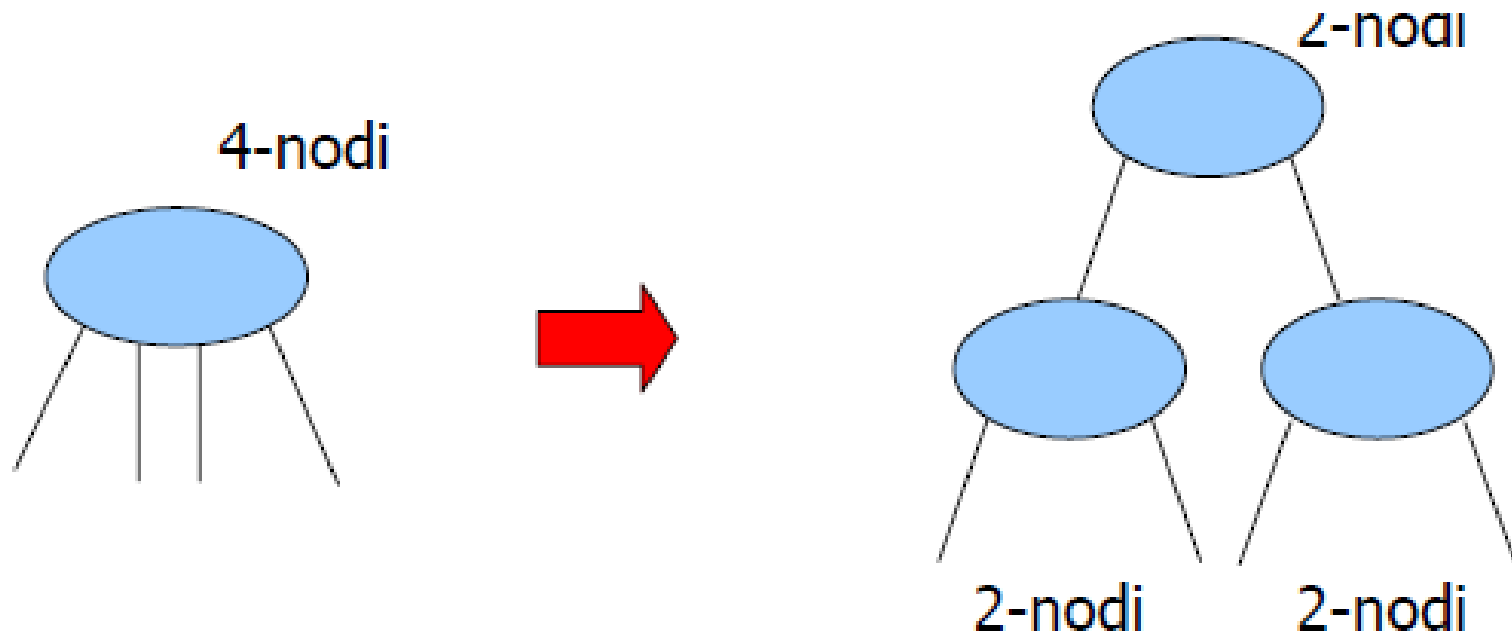
Generalizzazione della ricerca nei BST:

- ▶ Confronto sequenziale della chiave cercata con le chiavi contenute della radice
- ▶ search hit se trovata
- ▶ se non trovata, si scende nel sottoalbero che corrisponde all'intervallo di valori che comprende la chiave
- ▶ Si ripete (ricorsivamente) la ricerca nel sottoalbero.

# Esempio



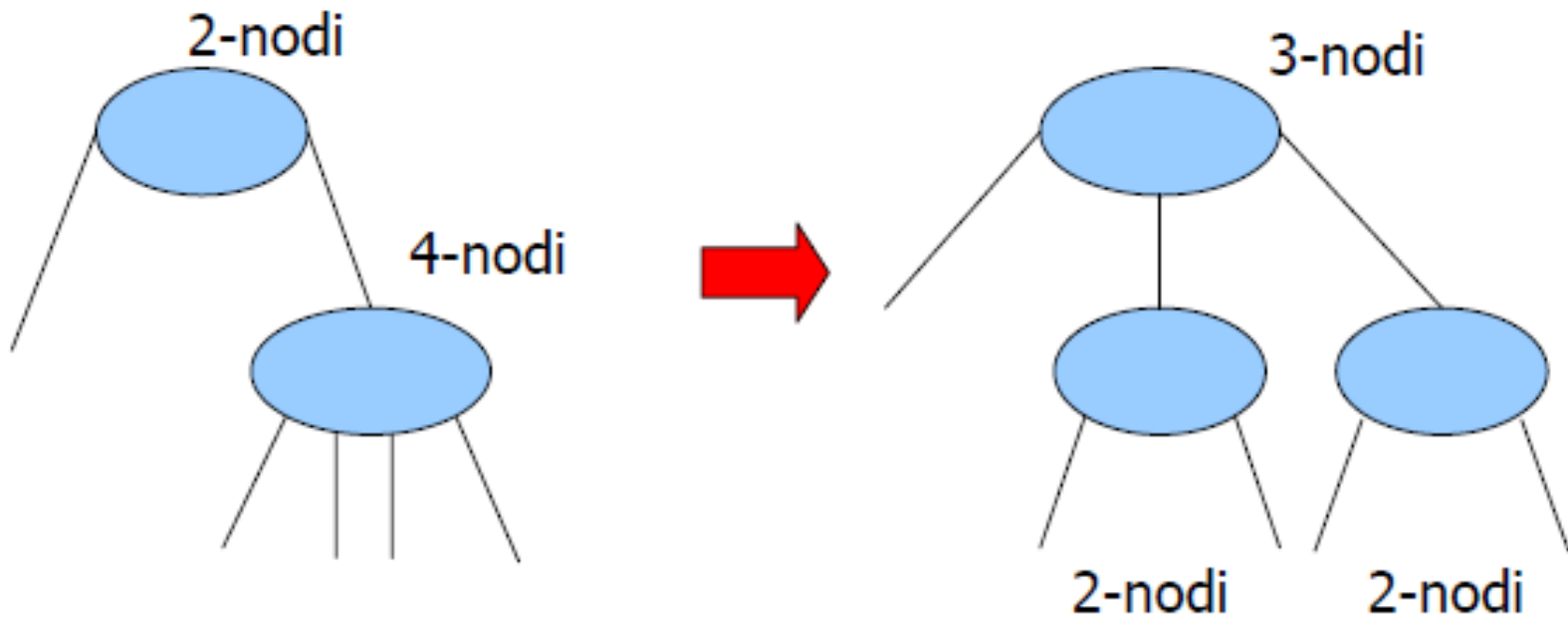
# Split di un 4-nodo radice



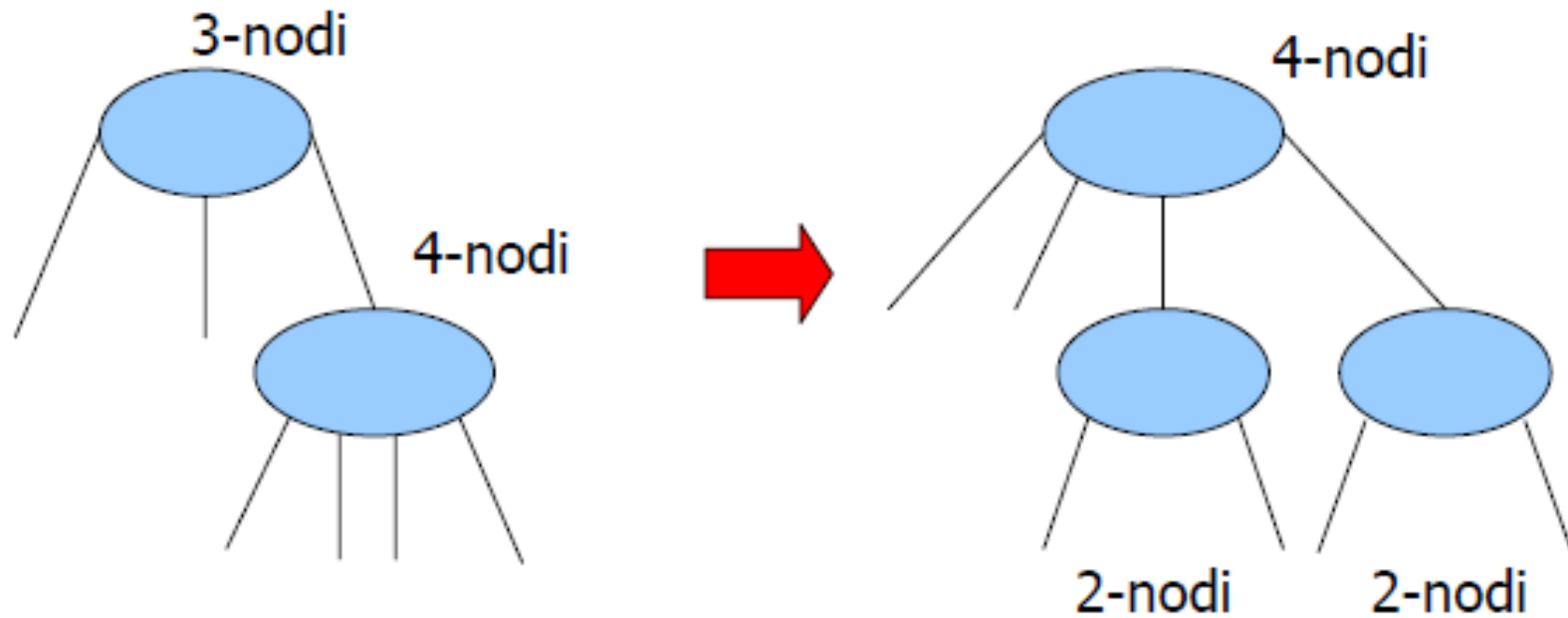
- ▶ L'altezza dell'albero cresce di 1.
- ▶ L'albero cresce sempre dalla radice verso l'alto.



# Split di un 4-nodo figlio di un 2-nodo



# Split di un 4-nodo figlio di un 3-nodo



# Inserimento bottom-up

- ▶ Si ricerca la nuova chiave a partire dalla radice
- ▶ Se la chiave non è contenuta nell'albero, la ricerca termina su una foglia (altrimenti la ricerca fallisce):
  - se la foglia è un 2-nodo, si inserisce ordinatamente la chiave trasformando la foglia in 3-nodo
  - se la foglia è un 3-nodo, si inserisce ordinatamente la chiave trasformando la foglia in 4-nodo
  - Se la foglia è un 4-nodo figlio di un 2-nodo o 3-nodo si effettua uno split del nodo foglia.
  - Se la foglia è un 4-nodo figlio di un 4-nodo, si effettua prima uno split del padre e poi uno split della foglia.

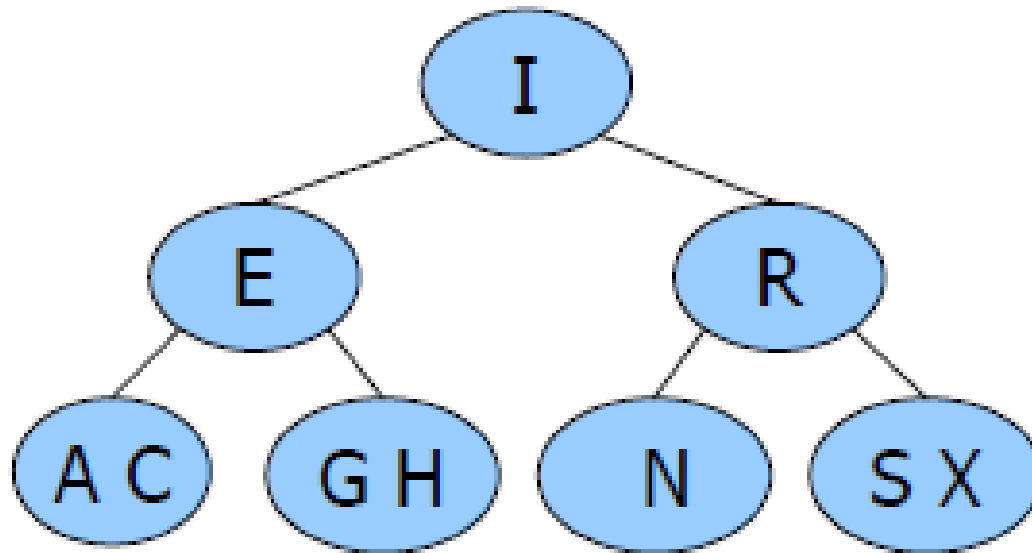
Ma anche il padre del padre potrebbe essere un 4-nodo...

# Inserimento top-down

- ▶ Durante la ricerca a partire dalla radice per identificare il nodo in cui inserire la nuova chiave:
  - se il nodo corrente (radice compresa) sul cammino di ricerca è un 4-nodo, lo si decompone in due 2-nodi (**split**) e si inserisce ordinatamente la chiave di mezzo nel nodo padre
  - **Remark:** il nodo padre del 4-nodo può essere solo un 2-nodo o un 3-nodo, per cui si procede con eventuali operazioni di split come per l'inserimento bottom-up

# Esempio

- Inserimento in sequenza di ASERCHINGX



# Analisi della complessità

- ▶ Una ricerca in un albero 2-3-4 bilanciato di  $N$  nodi non visita mai più di  $\lg N + 1$  nodi
- ▶ Un inserimento in un albero 2-3-4 bilanciato di  $N$  nodi richiede nel caso peggiore meno di  $\lg N + 1$  split.

# Alberi red-black (RBtree)

## Definizione 1

BST in cui:

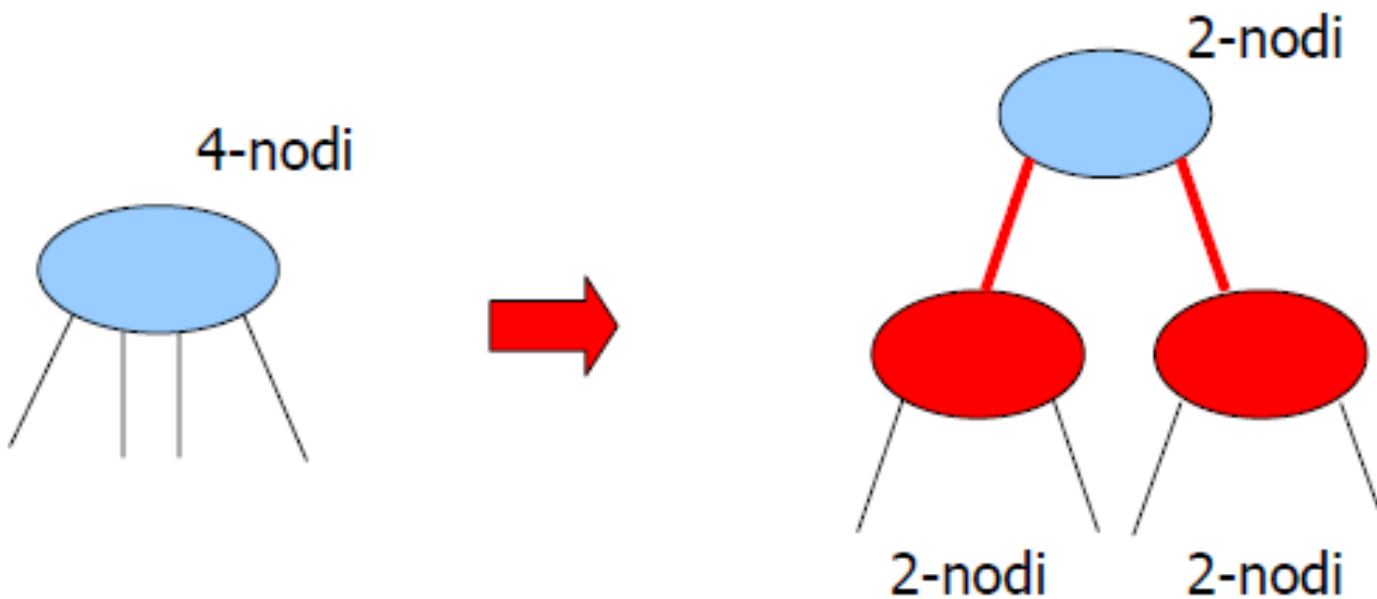
- ▶ ogni nodo è o **rosso** o **nero**
- ▶ se un nodo è rosso, non può avere figli rossi
- ▶ ogni cammino semplice dalla radice a una foglia contiene lo stesso numero di nodi neri

## Definizione 2

- ▶ Rappresentazione degli alberi 2-3-4 come BST con ulteriore bit di informazione per codificare 2-nodi e 3-nodi:
- ▶ link rossi che connettono piccoli alberi binari che formano 3-nodi e 4-nodi
- ▶ link neri che connettono l'intero albero 2-3-4
- ▶ Ogni nodo è raggiunto tramite 1 solo link, quindi colorare i link equivale a colorare i nodi.

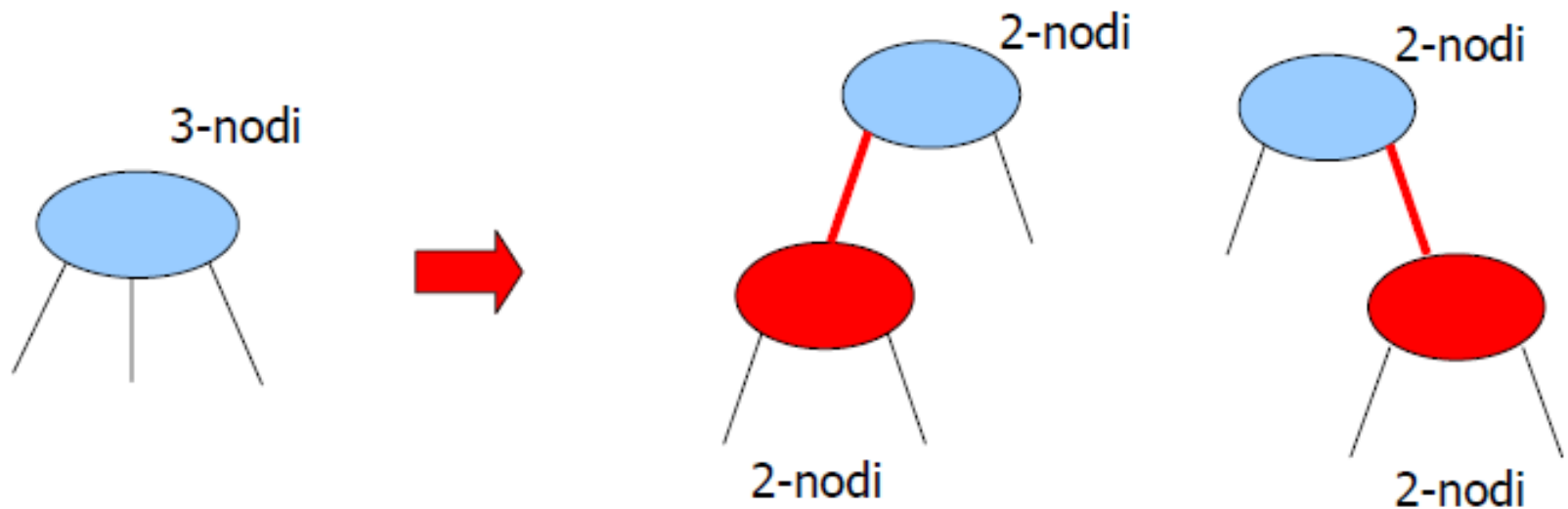


# Rappresentazione di un 4-nodo

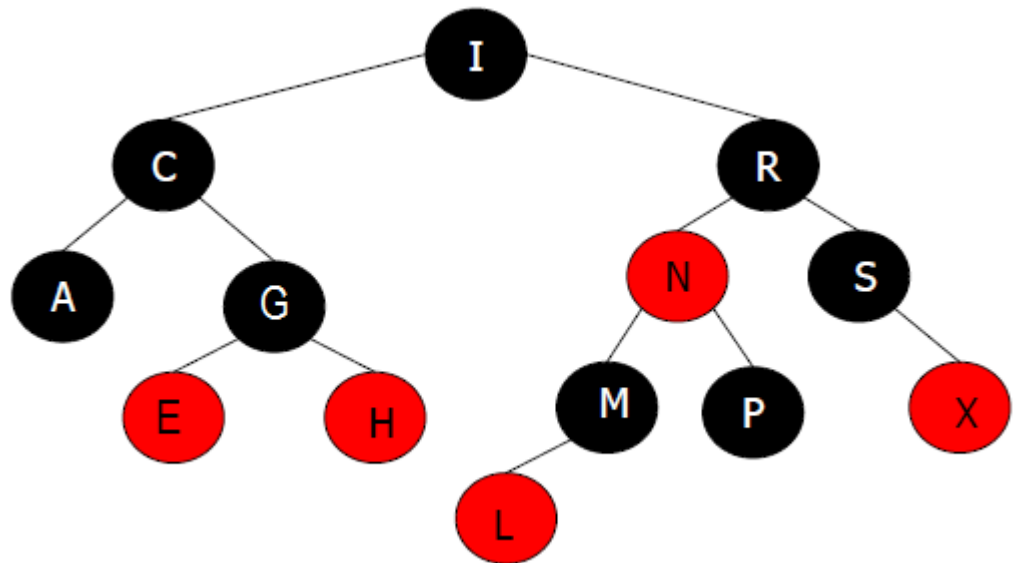
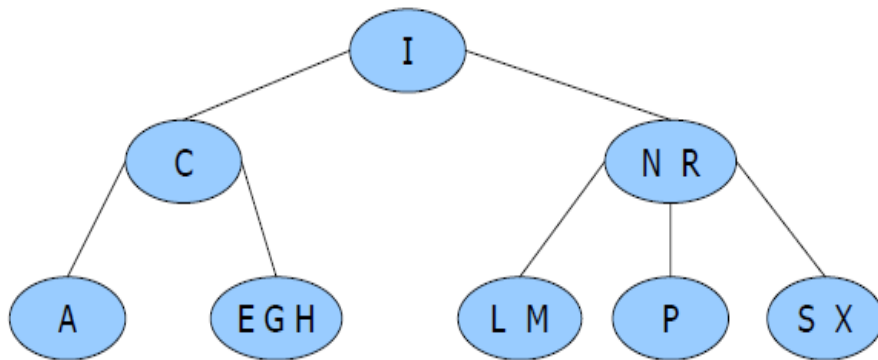


# Rappresentazione di un 3-nodo

Due alternative:



# Dall'albero 2-3-4 al RBtree



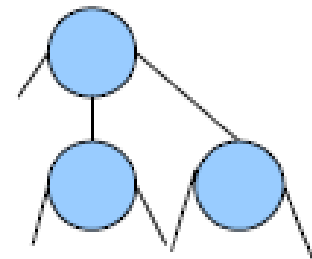
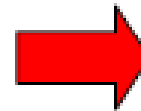
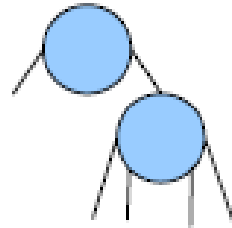
# Operazioni su RBtree

- ▶ La ricerca è identica a quella nel BST (il colore non influisce)
- ▶ L'inserimento deve garantire le proprietà dell'RBtree
- ▶ Inserimento top-down: immaginiamo di operare su un albero 2-3-4 implementato tramite un RBtree.

# Inserimento top-down

4-nodi figlio di 2-nodi (link DX)

Albero 2-3-4

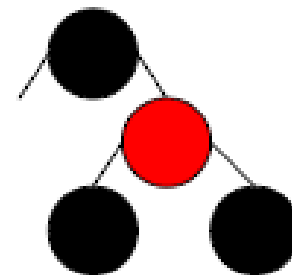
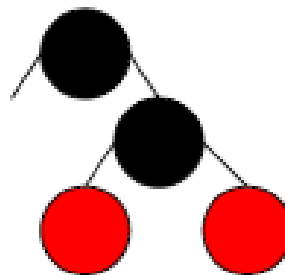


trasformazione



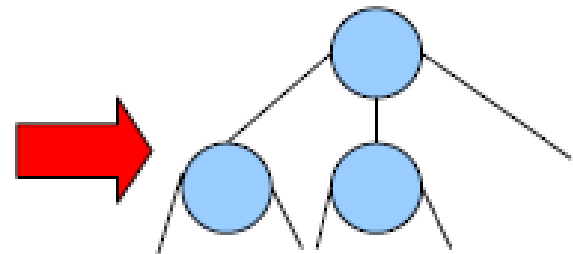
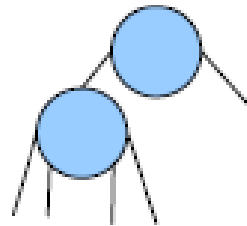
cambio di colore

RB-tree



## 4-nodi figlio di 2-nodi (link SX)

Albero 2-3-4

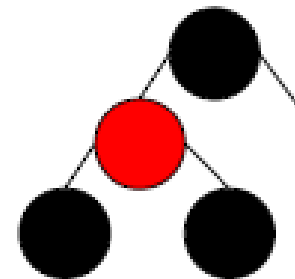
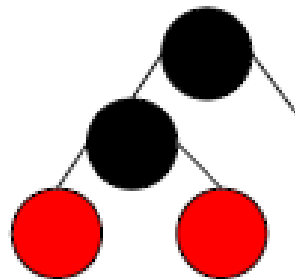


trasformazione



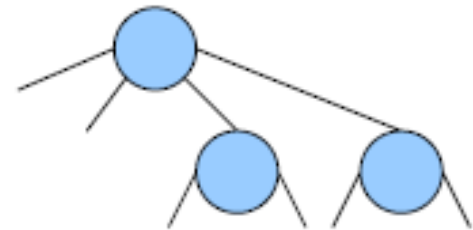
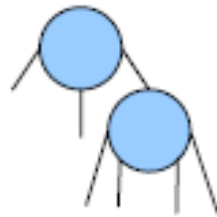
cambio di colore

RB-tree



## 4-nodi figlio di 3-nodi (link DX)

Albero 2-3-4

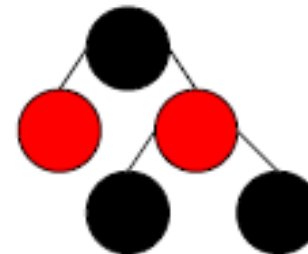
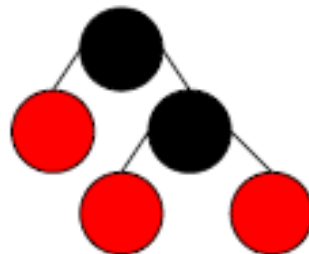


RB-tree

trasformazione

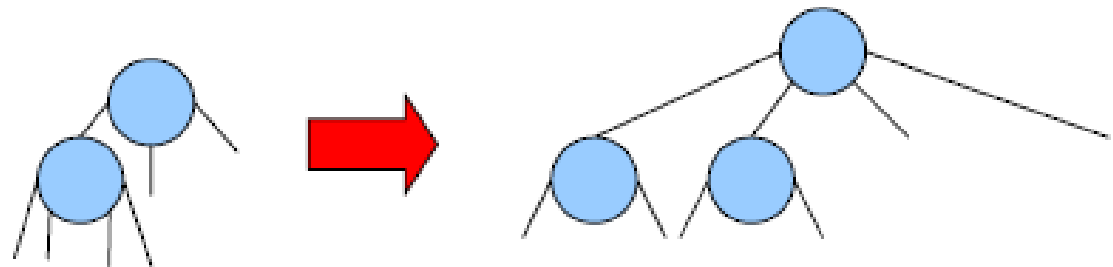


cambio di colore



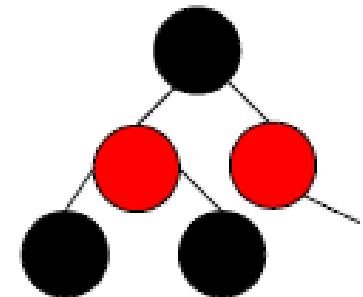
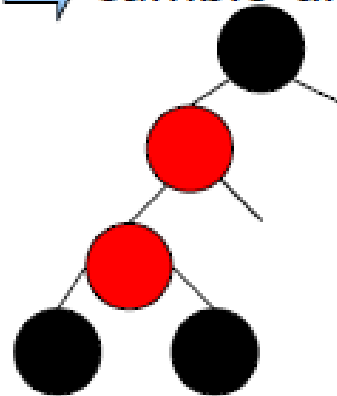
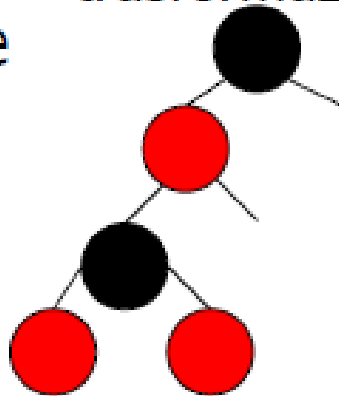
## 4-nodi figlio di 3-nodi (link SX)

Albero 2-3-4



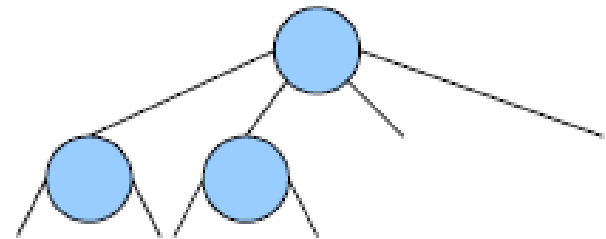
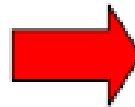
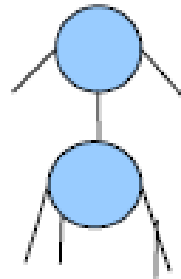
RB-tree

trasformazione ➡ cambio di colore ➡ rotazione DX



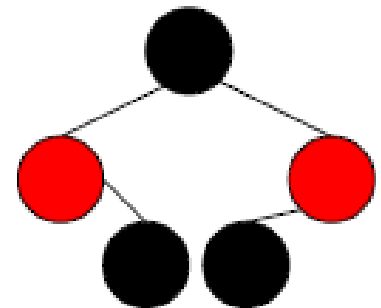
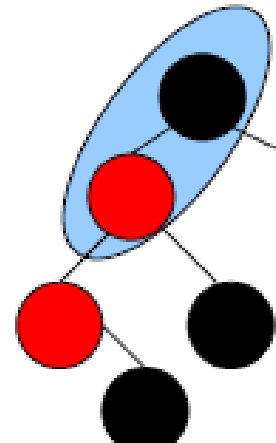
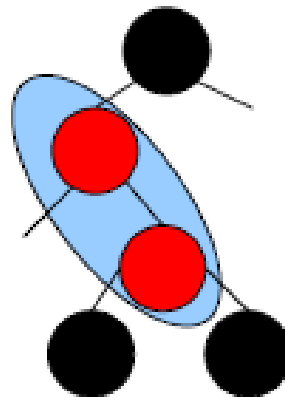
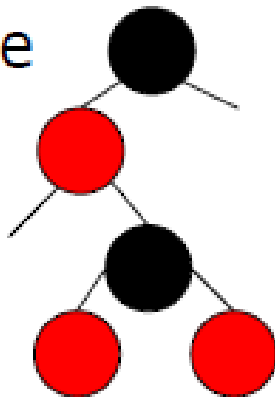


Albero 2-3-4



trasformazione ➡ cambio colore ➡ rot. SX ➡ rot. DX

RB-tree

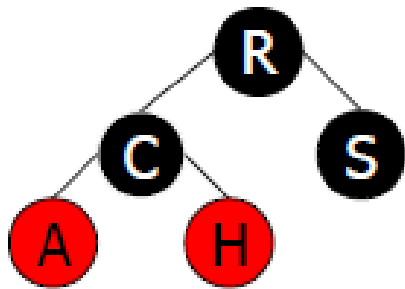


# L'inserimento

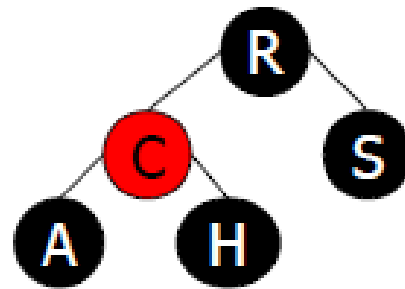
- ▶ L'inserimento richiede operazioni locali di “ristrutturazione” dell'albero con:
  - cambi di colore
  - ribilanciamento
- ▶ scendendo nell'albero si scompongono i 4-nodi, invertendo il colore dei 3 nodi risultanti
- ▶ risalendo si eseguono le rotazioni se necessario

# Esempio

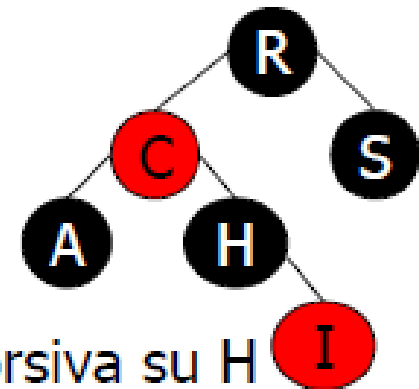
Inserimento di I



discesa ricorsiva su C  
e cambio colore

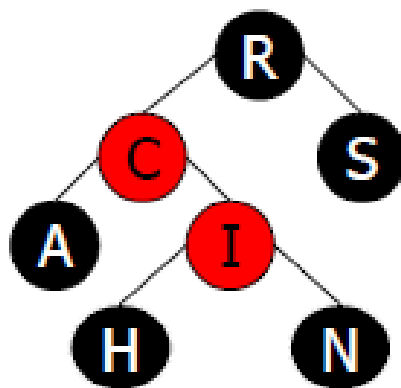
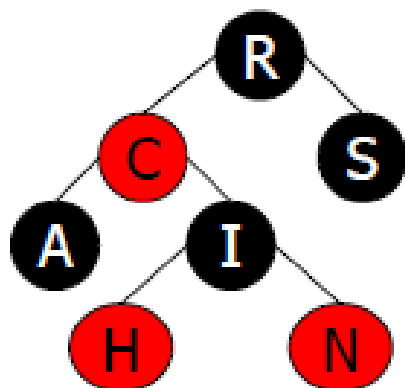


discesa ricorsiva su H  
e creazione di nuovo  
nodo rosso

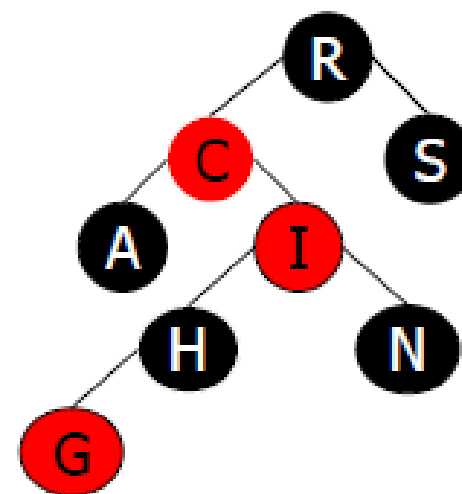


risalita senza rotazioni

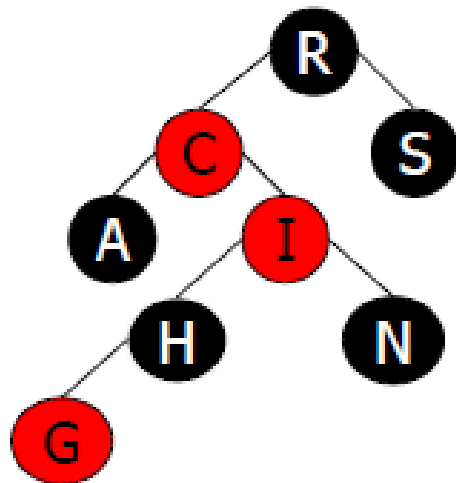
## Inserimento di G



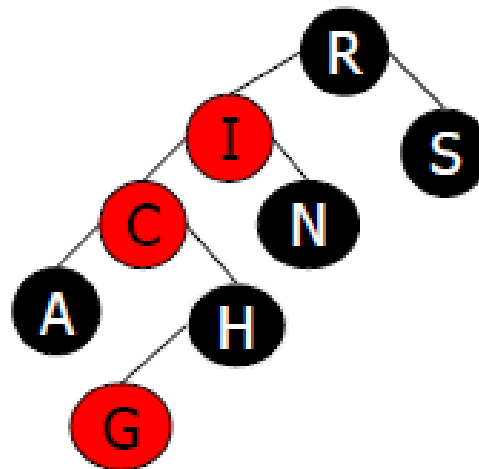
discesa ricorsiva su I  
e cambio colore



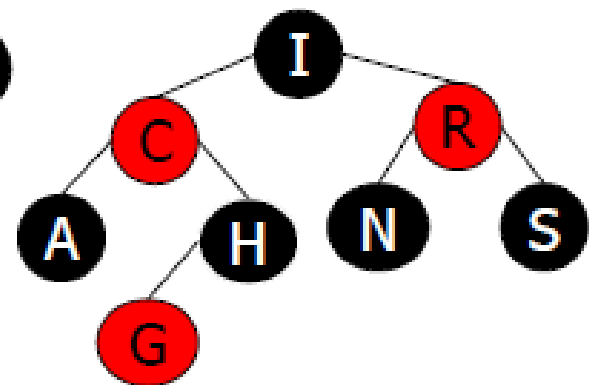
discesa ricorsiva su H  
e creazione di nuovo  
nodo rosso



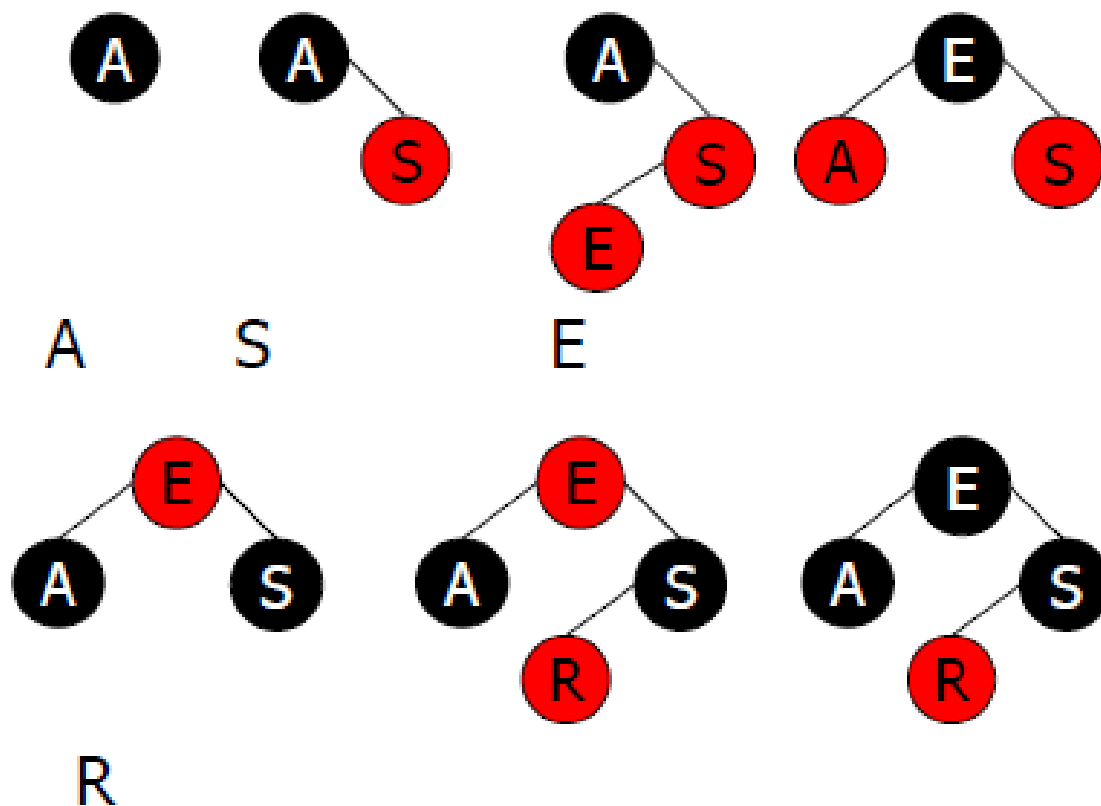
risalita e rotazione  
a SX rispetto a C

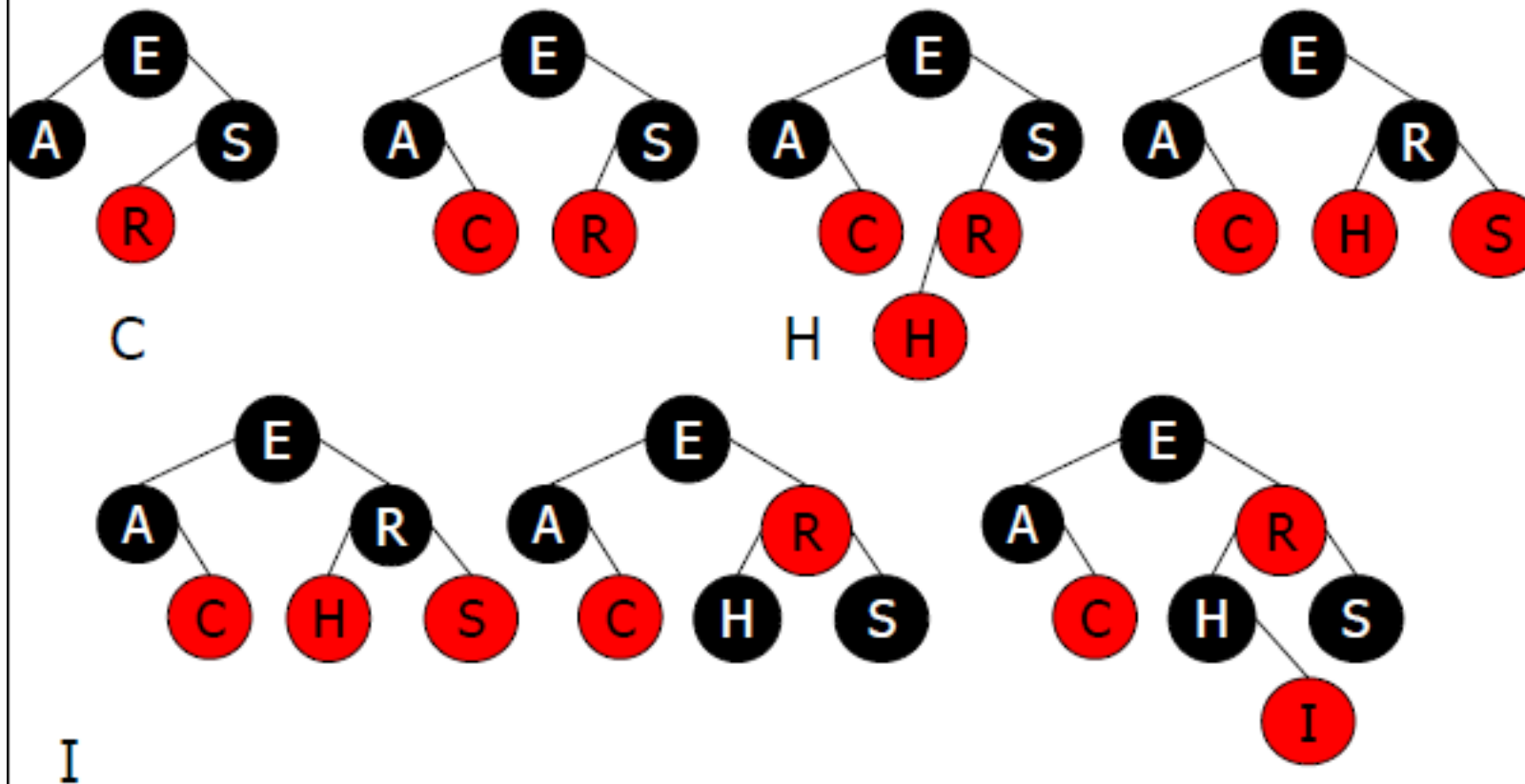


risalita e rotazione  
a DX rispetto a R  
cambio colore della radice



Inserimento di A S E R C H I in RB-tree vuoto





- ▶ Gli Rbtree costituiscono l'elemento centrale per la realizzazione di due classi del JCF molto usate:
  - TreeSet
  - TreeMap