
BOB TARJAN
EDMUND CLARKE
BUD MISHRA
© CARNEGIE MELLON UNIVERSITY (1985)

Robert Endre Tarjan

Computer science career

Tarjan has been teaching at Princeton University since 1985.
He has also held academic positions at

- Cornell University (1972–73),
- University of California, Berkeley (1973–1975),
- Stanford University (1974–1980), and
- New York University (1981–1985).

He has also been a fellow of the NEC Research Institute (1989–1997). In April 2013 he joined Microsoft Research Silicon Valley in addition to the position at Princeton. In October 2014 he rejoined Intertrust Technologies as chief scientist.

Tarjan has worked at AT&T Bell Labs (1980–1989), Intertrust Technologies (1997–2001, 2014– present), Compaq (2002) and Hewlett Packard (2006–2013).

Algorithms

Tarjan is known for his pioneering work on graph theory algorithms and data structures. Some of his well-known algorithms include Tarjan's off-line least common ancestors algorithm, and Tarjan's strongly connected components algorithm, and he was one of five co-authors of the median of medians linear time selection algorithm. The Hopcroft-Tarjan planarity testing algorithm was the first linear-time algorithm for planarity-testing.

Data structures

Tarjan has also developed important data structures such as the Fibonacci heap (a heap data structure consisting of a forest of trees), and the splay tree (a self-adjusting binary search tree; co-invented by Tarjan and Daniel Sleator). Another significant contribution was the analysis of the disjoint-set data structure; he was the first to prove the optimal runtime involving the inverse Ackermann function.

Awards

- Turing Award jointly with John Hopcroft in 1986
- ACM Fellow in 1994
- Nevanlinna Prize in Information Science (1983)
- National Academy of Sciences Award for Initiatives in Research (1984)
- Paris Kanellakis Award in Theory and Practice, ACM (1999)
- Blaise Pascal Medal in Mathematics and Computer Science, European Academy of Sciences (2004)
- Caltech Distinguished Alumni Award, California Institute of Technology (2010)

SPLAY TREES (TARJAN, SLEATOR 1985)

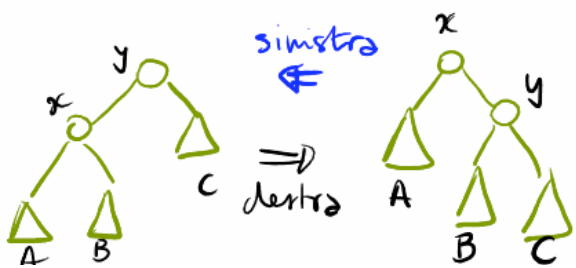
- GLI SPLAY TREES IMPLEMENTANO M OPERAZIONI CONSECUTIVE DI
  - RICERCA
  - INSERIMENTO
  - CANCELLAZIONE

  IN TEMPO $O(M \log N)$, DOVE N E' IL NUMERO DI INSERIMENTI
- IN ALTRE PAROLE CIASCUNA DELLE M OPERAZIONI VIENE ESEGUITA IN TEMPO AMMORTIZZATO $O(\log N)$

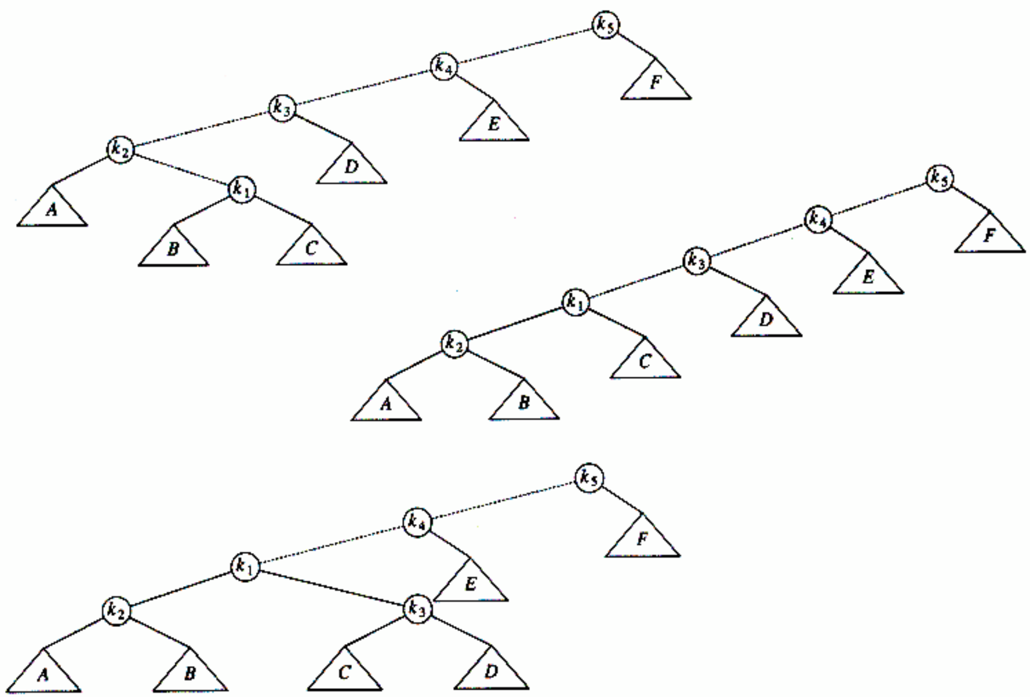
IDEA: PER QUANTO POSSIBILE AVVICINARE ALLA RADICE I NODI CHE SI INCONTRANO DURANTE UN ACCESSO

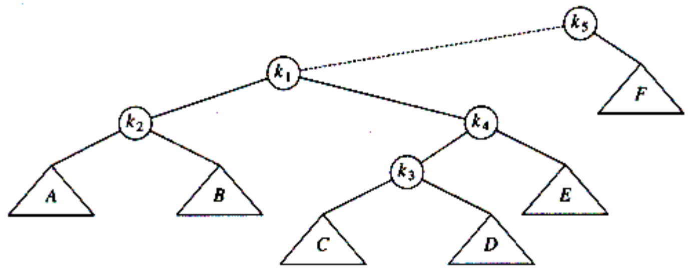
UNA SEMPLICE IDEA (CHE NON FUNZIONA)

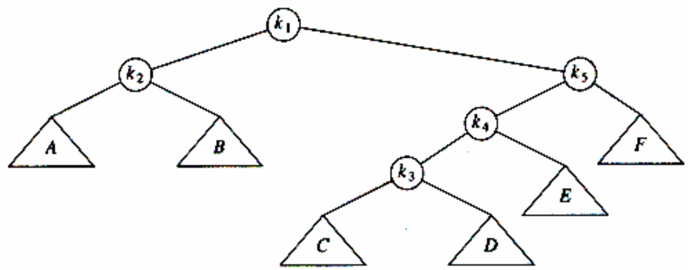
EFFETTUARE TUTTE LE POSSIBILI ROTAZIONI, DAL BASSO VERSO L'ALTO, LUNGO IL CAMMINO DI ACCESSO



ESEMPIO (ACCESSO AL NODO k_1)




k_5
k_1
k_2
k_4
k_3
A
B
C
D
E
F


k_1
k_2
k_5
k_4
k_3
A
B
C
D
E
F

ESEMPIO

SI CONSIDERI LA SEQUENZA DELLE SEGUENTI OPERAZIONI:

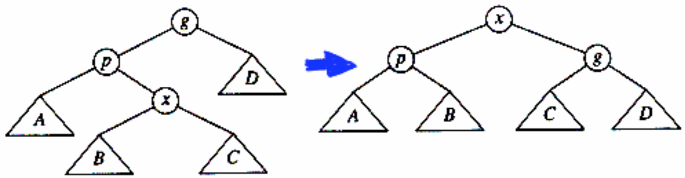
INSERT(1)
INSERT(2)
⋮
INSERT(N)

$\Theta(N)$

SEARCH(1)	N−1	ROTAZ.
SEARCH(2)	N−1	"
SEARCH(3)	N−2	"
SEARCH(4)	N−3	"
⋮		
SEARCH(N)	1	"

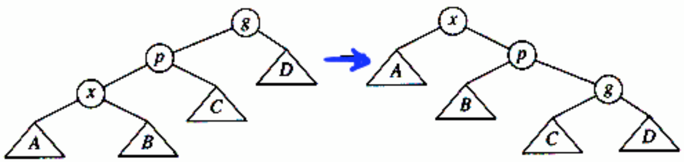
$\Theta(N^2)$

SPLAY TREES



ZIG-ZAG

+ SIMMETRICHE



ZIG-ZIG

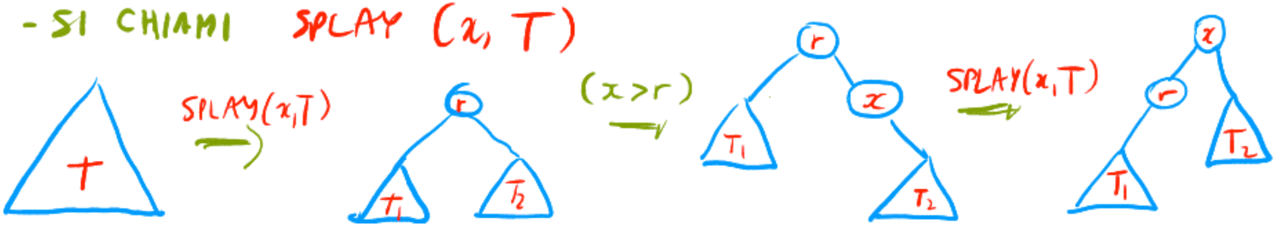
ZIG (x NON HA NONNO)

SPLAY (x, T)

- SI CERCHI x SU T MEDIANTE RICERCA BINARIA
- A PARTIRE DAL NODO OVE LA RICERCA SI E' FERMATA E PROCEDENDO VERSO LA RADICE, SI APPLICHINO LE ROTAZIONI DI TIPO ZIG, ZIG-ZAG O ZIG-ZIG NECESSARIE

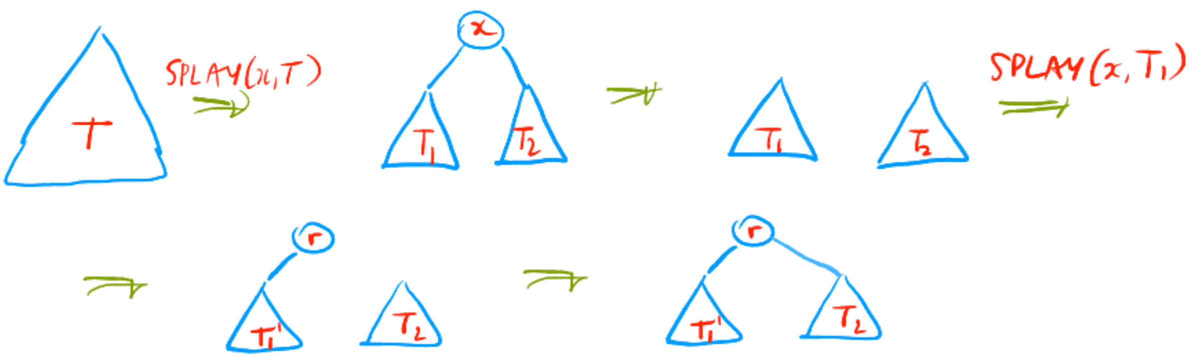
INSERT (x, T)

- SI CHIAMI SPLAY (x, T)
- SI INSERISCA IL NUOVO NODO x
- SI CHIAMI SPLAY (x, T)



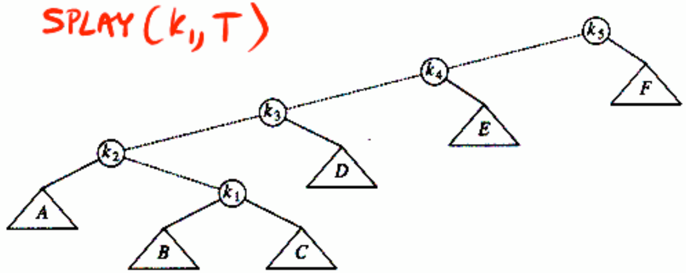
DELETE(x,T)

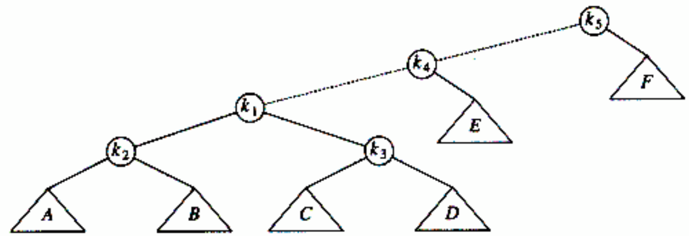
- SPLAY(x,T)
- SI CANCELLI IL NODO x OTTENENDO DUE ALBERI T_1 E T_2
- SPLAY(x,T_1); SIA r LA NUOVA RADICE
- SI PONGA LA RADICE DI T_2 COME FIGLIO DESTRO DEL NODO r

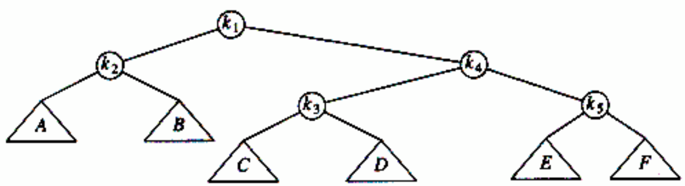


ESEMPIO

SPLAY(k_1, T)

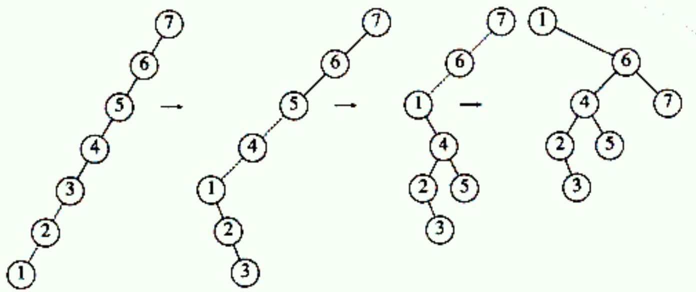




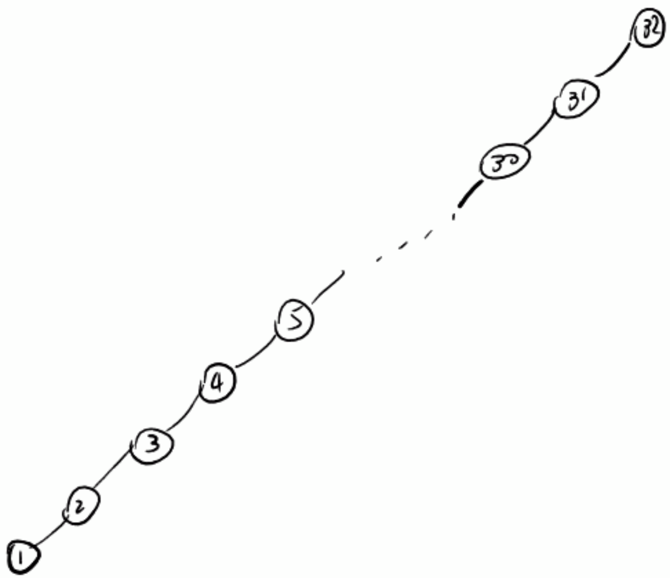


ESEMPIO

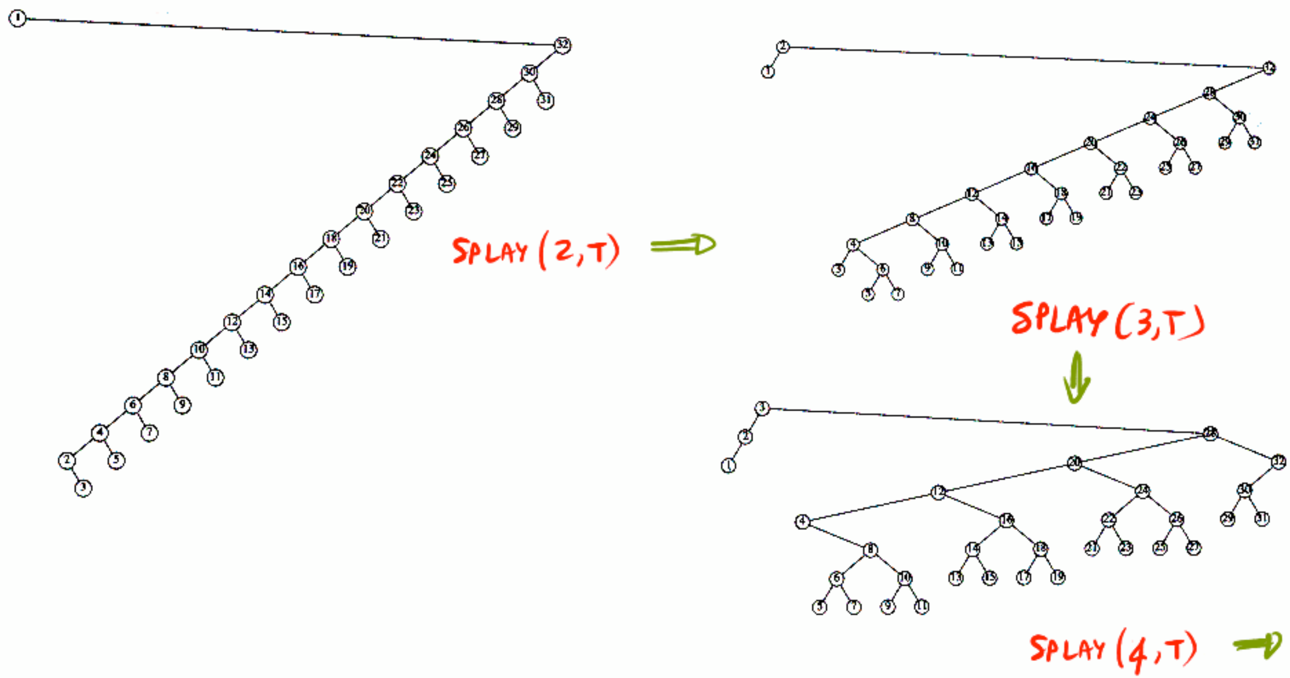
SPLAY (1, T)

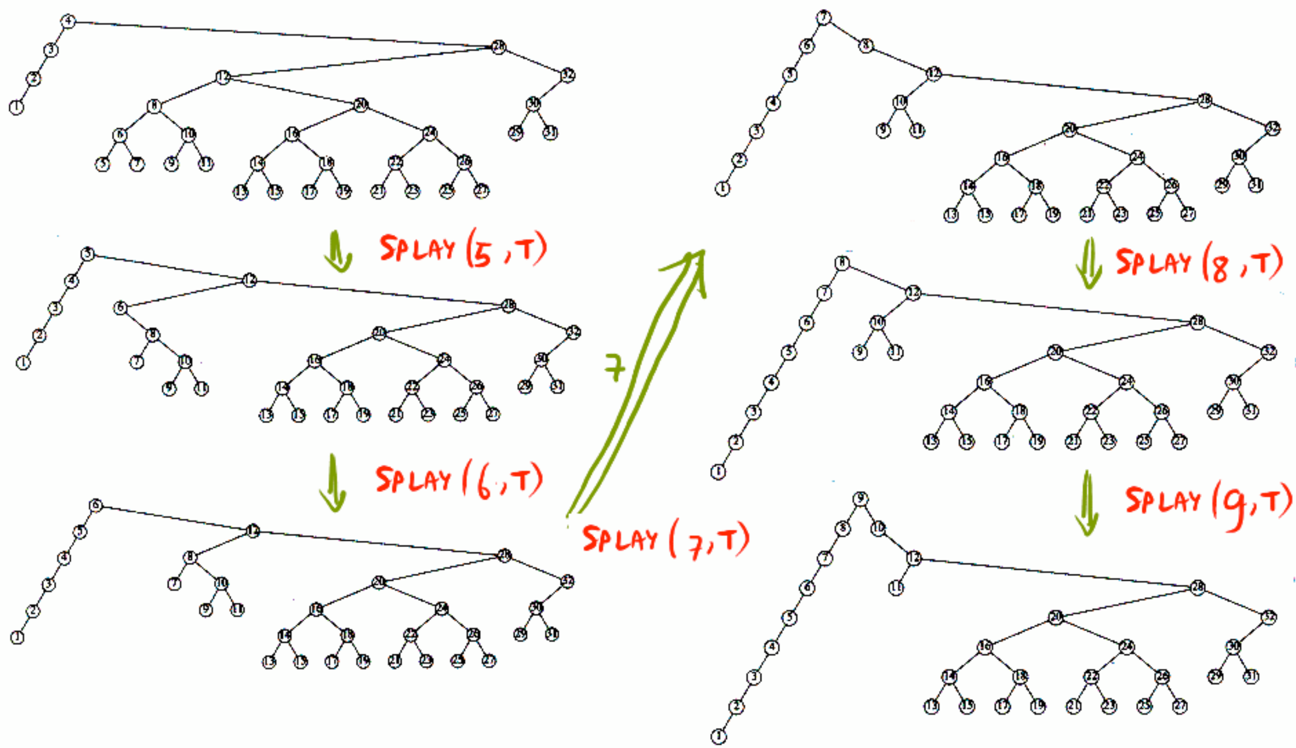


ESEMPIO



SPLAY (1, T) $\Rightarrow$


SPLAY (2,T)
SPLAY (3,T)
SPLAY (4,T)


SPLAY (5 , T)
SPLAY (6 , T)
7
SPLAY (7, T)
SPLAY (8 , T)
SPLAY (9, T)

ANALISI AMMORTIZZATA DELL'OPERAZIONE SPLAY

COSTI DELLE OPERAZIONI ELEMENTARI

ZIG	1 ROTAZIONE
ZIG-ZAG	2 ROTAZIONI
ZIG-ZIG	2 ROTAZIONI

PONIAMO:

$S(v) =_{df}$ # NODI DEL SOTTOALBERO CON RADICE v

$R(v) =_{df} \lg S(v)$ (rango di v)

$$\Phi(T) =_{df} \sum_{v \in T} R(v)$$

- SIA T_0 UN ALBERO VUOTO. SI HA

$$\Phi(T_0) = \sum_{v \in T_0} R(v) = 0$$

- SIA T UN ALBERO QUALSIASI

$$\Phi(T) = \sum_{v \in T} R(v) = \sum_{v \in T} \lg S(v) \geq 0 = \Phi(T_0)$$

- PERTANTO POSSIAMO UTILIZZARE IL POTENZIALE $\Phi(T)$ PER CALCOLARE UN UPPER BOUND AL COSTO REALE DI M OPERAZIONI (SPLAY, INSERT, DELETE) DI CUI N SONO INSERT

LEMMA

SIANO $a, b \in \mathbb{N}^+$ E SIA $c \geq a+b$.

ALLORA $\lg a + \lg b \leq 2\lg c - 2$.

DIM.

SI OSSERVI

$$(a-b)^2 \geq 0$$

$$a^2 - 2ab + b^2 \geq 0$$

$$a^2 + 2ab + b^2 \geq 4ab$$

$$(a+b)^2 \geq 4ab$$

$$a+b \geq 2\sqrt{ab}$$

$$c \geq 2\sqrt{ab} \qquad \text{(POICHE' } c \geq a+b\text{)}$$

$$\lg c \geq 1 + \frac{1}{2}(\lg a + \lg b) \qquad \text{(PRENDENDO I LOGARITMI DI AMBO I MEMBRI)}$$

$$2\lg c - 2 \geq \lg a + \lg b. \quad \blacksquare$$

TEOREMA IL COSTO AMMORTIZZATO DELL'OPERAZIONE SPLAY(x, T) E' AL PIÙ $3(R(root(T)) - R(x)) + 1$.

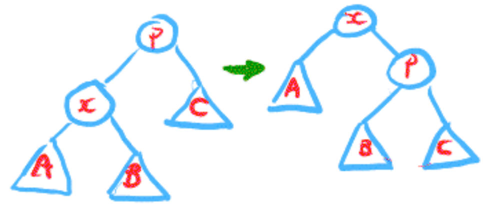
DIM. CALCOLIAMO PER INIZIARE IL COSTO AMMORTIZZATO DI UNA OPERAZIONE DI TIPO ZIG, ZIG-ZAG, ZIG-ZIG.

NOTAZIONE

INDICHIAMO CON $S_i(x)$ E $S_f(x)$ IL NUMERO DI NODI NEL SOTTOALBERO DI RADICE x PRIMA E DOPO L'OPERAZIONE IN ESAME.

ANALOGAMENTE PER $R_i(x)$ E $R_f(x)$.

CASO ZIG



COSTO REALE $= 1$

$$\Delta\Phi = R_f(x) + R_f(p) - R_i(x) - R_i(p)$$

SI OSSERVI CHE

$S_i(p) \geq S_f(p) \longrightarrow R_i(p) \geq R_f(p) \longrightarrow \Delta\Phi \leq R_f(x) - R_i(x)$

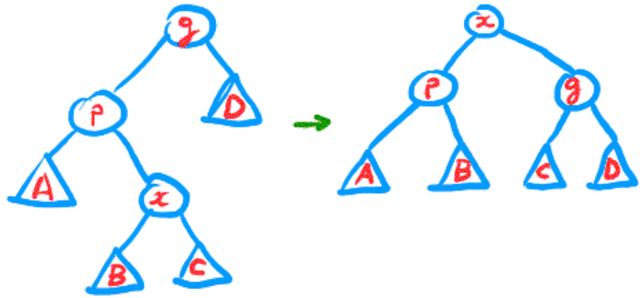
$S_f(x) \geq S_i(x) \longrightarrow R_f(x) - R_i(x) \geq 0 \longrightarrow \Delta\Phi \leq 3(R_f(x) - R_i(x))$

$$\hat{c}_{ZIG} = 1 + \Delta\Phi \leq 1 + 3(R_f(x) - R_i(x))$$

CASO ZIG-ZAG

COSTO REALE = 2

$$\Delta\Phi = R_f(x,p,g) - R_i(x,p,g)$$



SI OSSERVI QUANTO SEGUE:

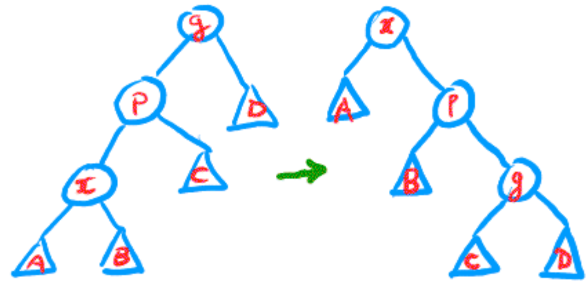
- $S_f(x) = S_i(g) \rightarrow R_f(x) = R_i(g) \rightarrow \Delta\Phi = R_f(p,g) - R_i(x,p)$
- $S_i(p) \geq S_i(x) \rightarrow R_i(p) \geq R_i(x) \rightarrow \Delta\Phi \leq R_f(p,g) - 2R_i(x)$
- $S_f(p) + S_f(g) \leq S_f(x) \rightarrow R_f(p) + R_f(g) \leq 2R_f(x) - 2$
  $\rightarrow \Delta\Phi \leq 2(R_f(x) - R_i(x)) - 2$
- $S_f(x) \geq S_i(x) \rightarrow R_f(x) - R_i(x) \geq 0 \rightarrow \Delta\Phi \leq 3(R_f(x) - R_i(x)) - 2$

$$\hat{C}_{ZIG-ZAG} = 2 + \Delta\Phi \leq 2 + 3(R_f(x) - R_i(x)) - 2 = 3(R_f(x) - R_i(x)).$$

CASO ZIG-ZIG

COSTO REALE = 2

$\Delta\Phi = R_f(x,p,g) - R_i(x,p,g)$



SI OSSERVI CHE:

- $S_f(x) = S_i(g) \rightarrow R_f(x) = R_i(g) \rightarrow \Delta\Phi = R_f(p,g) - R_i(x,p)$
- $S_f(x) \geq S_f(p) \rightarrow R_f(x) \geq R_f(p) \rightarrow \Delta\Phi \leq R_f(x,g) - R_i(x,p)$
- $S_i(x) + S_f(g) \leq S_f(x) \longrightarrow R_i(x) + R_f(g) \leq 2R_f(x) - 2$

  $\rightarrow \Delta\Phi \leq R_f(x) + R_f(g) - 2R_i(x) - R_i(p) + R_i(x) \leq 3R_f(x) - 2 - 2R_i(x) - R_i(p)$
- $S_i(x) \leq S_i(p) \longrightarrow R_i(x) \leq R_i(p) \rightarrow \Delta\Phi \leq 3\left(R_f(x) - R_i(x)\right) - 2$

$\hat{C}_{ZIG-ZIG} = 2 + \Delta\Phi \leq 3\left(R_f(x) - R_i(x)\right)$

RIASSUMENDO

$$\hat{c}_{ZIG} \le 3(R_f(x) - R_i(x)) + 1$$

$$\hat{c}_{ZIG-ZAG},\ \hat{c}_{ZIG-ZIG} \le 3(R_f(x) - R_i(x))$$

$$\hat{c}_{SPLAY} = \sum_{j=1}^{k} \hat{c}_j \le \sum_{j=1}^{k} 3(R_f^{(j)}(x) - R_i^{(j)}(x)) + 1$$

MA $R_f^{(j)}(x) = R_i^{(j+1)}(x)$ PER $j<k$, QUINDI

$$\hat{c}_{SPLAY} \le 3(R_f^{(k)}(x) - R_i^{(k)}(x)) + \sum_{j=1}^{k-1} 3(R_i^{(j+1)}(x) - R_i^{(j)}(x)) + 1$$

$$= 3(R_f^{(k)}(x) - R_i^{(k)}(x)) + 3(R_i^{(k)}(x) - R_i^{(1)}(x)) + 1$$

$$= 3(R_f^{(k)}(x) - R_i^{(1)}(x)) + 1$$

$$= 3(R(root(T)) - R(x)) + 1$$

PERTANTO $\hat{c}_{SPLAY} \leq 3 \, R(root(T)) + 1$

$$= 3 \log(|T|) + 1$$

$$= O(\log |T|) = O(\log N)$$

QUANTE SPLAY CI SONO? $O(M)$

DUNQUE

$$\sum_{i=1}^{M} c_{op_i} \leq \sum_{i=1}^{M} \hat{c}_{op_i} = \sum_j \hat{c}_{SPLAY_j} = O(M \log N)$$

TOP-DOWN SPLAY TREES

- NEI LUCIDI PRECEDENTI ABBIAMO ILLUSTRATO I COSIDDETTI BOTTOM-UP SPLAY TREES
- I BOTTOM-UP SPLAY TREES SONO PIÙ FACILI DA ANALIZZARE, MA DAL PUNTO DI VISTA PRATICO PRESENTANO ALCUNI PROBLEMI IMPLEMENTATIVI (E QUINDI DI EFFICIENZA)
- INFATTI LE ROTAZIONI ZIG, ZIG-ZIG E ZIG-ZAG HANNO LUOGO BOTTOM-UP E QUINDI OCCORRE
  - MEMORIZZARE IL CAMMINO DALLA RADICE AL NODO CERCATO, OPPURE
  - MANTENERE PER CIASCUN NODO UN PUNTATORE AL PADRE

- LA VARIANTE TOP-DOWN CONSENTE DI RISOLVERE EGREGIAMENTE IL PRECEDENTE PROBLEMA, PUR MANTENENDO UN COSTO AMMORTIZZATO LOGARITMICO PER SPLAY

- L'IDEA DI BASE E' LA SEGUENTE:
  - MENTRE SI PROCEDE NELLA RICERCA DI UN DATO NODO, SI TRATTANO OPPORTUNAMENTE I NODI INCONTRATI NONCHE' I LORO SOTTOALBERI.
  - NEL CORSO DELL'OPERAZIONE DI SPLAY, L'ALBERO RISULTERA' SPEZZATO IN TRE PARTI:
    - UN ALBERO L
    - UN ALBERO T' DI RADICE UN CERTO NODO x
    - UN ALBERO R

    TALI CHE $L \leq T' \leq R$ (NEL SENSO DELLE CHIAVI CONTENUTE IN ESSI)
  - INIZIALMENTE x E' LA RADICE DEL NOSTRO ALBERO T ED L E R SONO VUOTI

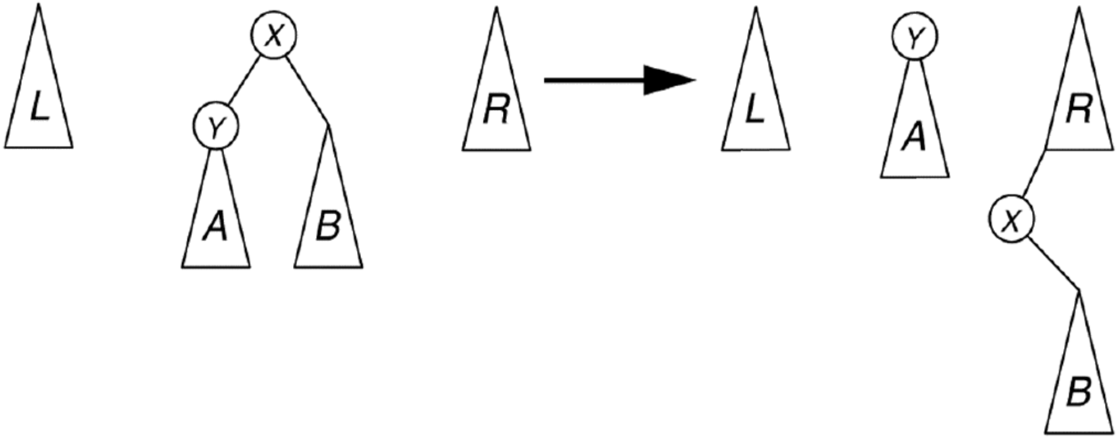
- SUCCESSIVAMENTE, SCENDIAMO DI DUE LIVELLI PER VOLTA, SE POSSIBILE, ALLA RICERCA DEL NOSTRO NODO ED APPLICHIAMO L'OPERAZIONE DI ZIG-ZAG O ZIG-ZIG (DESCRITTE SOTTO) A SECONDA DELLA SITUAZIONE.

  NEL CASO SI POSSA SCENDERE DI UN SOLO LIVELLO, SI APPLICHERÀ L'OPERAZIONE ZIG

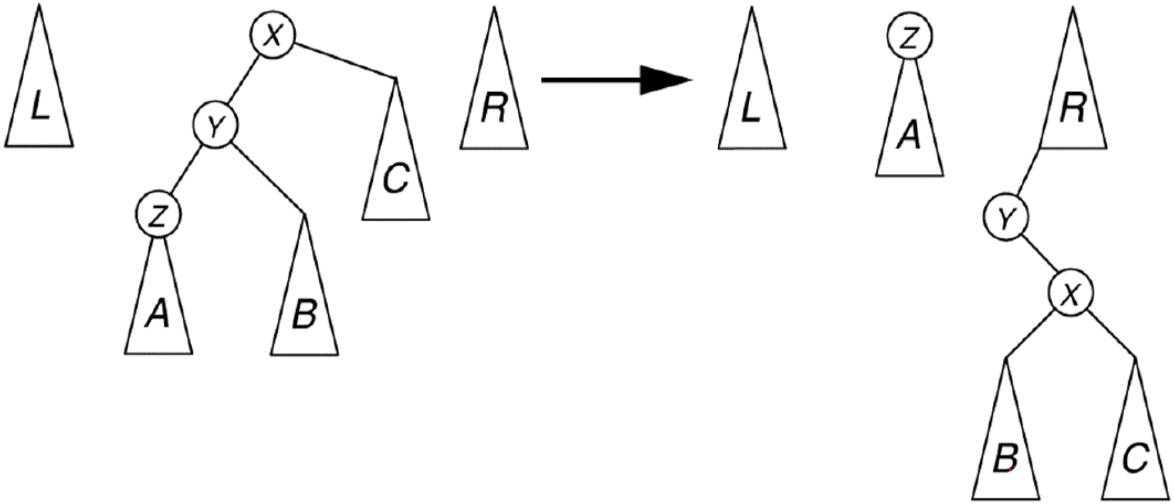
- LE OPERAZIONI ZIG, ZIG-ZIG E ZIG-ZAG HANNO L'EFFETTO DI AGGIORNARE I TRE ALBERI L, T' E R MANTENENDO L'INVARIANTE $L \leq T' \leq R$

- ALLA FINE, I TRE ALBERI VENGONI RIASSEMBLATI IN UN UNICO ALBERO

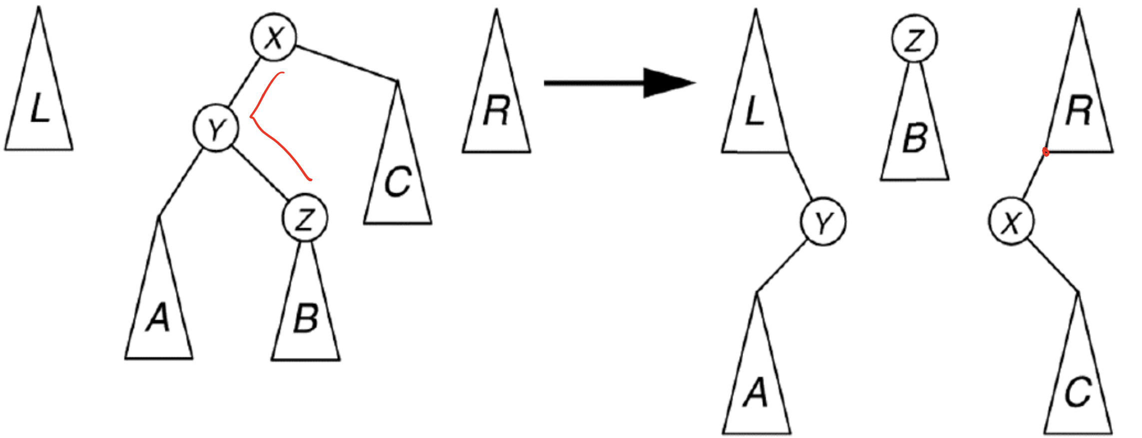
OPERAZIONE ZIG



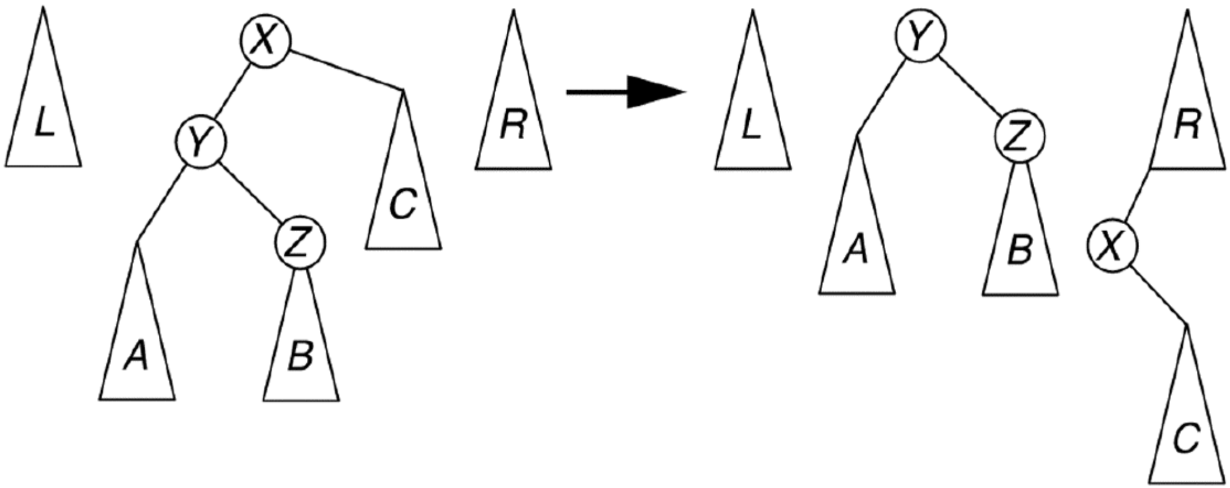
OPERAZIONE ZIG-ZIG



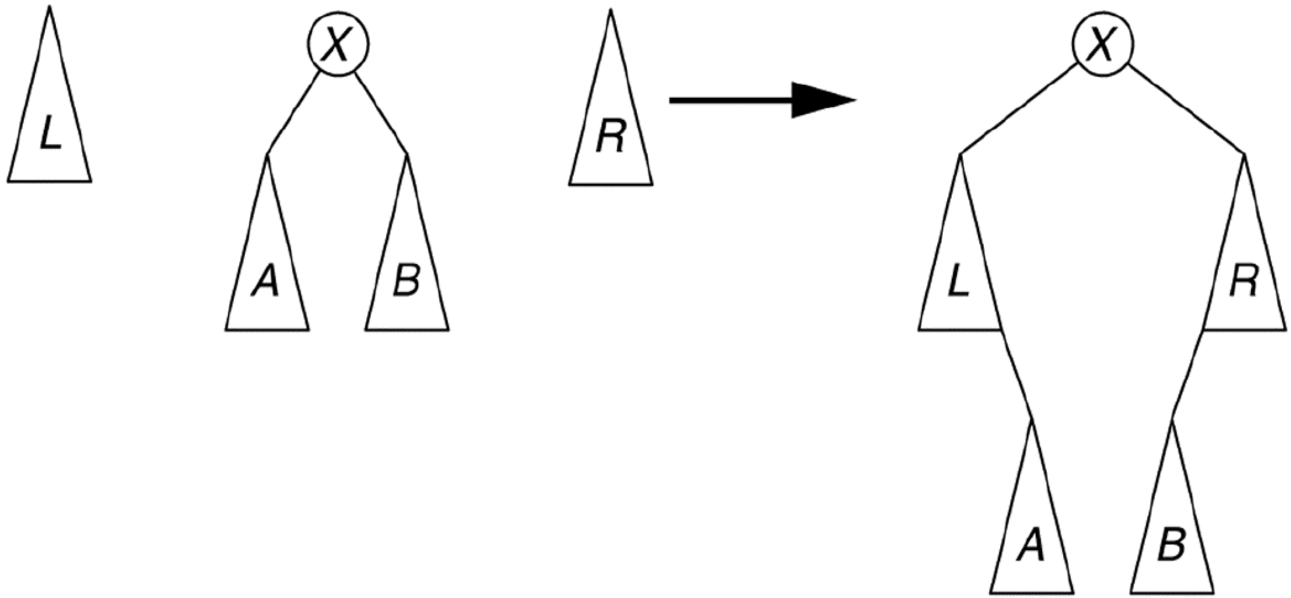
OPERAZIONE ZIG-ZAG



OPERAZIONE ZIG-ZAG (SEMPLIFICATA)



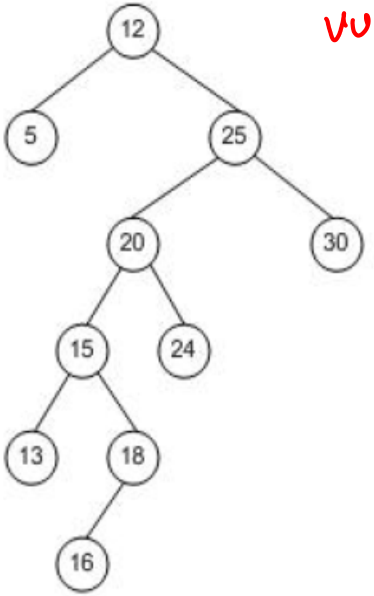
RIASSEMBLAGGIO FINALE



ESEMPIO

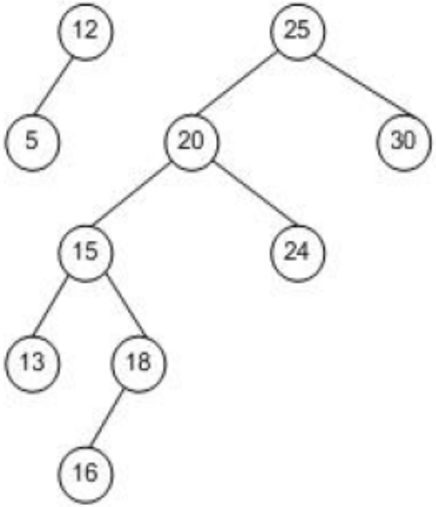
SPLAY (19)

VUOTO



VUOTO

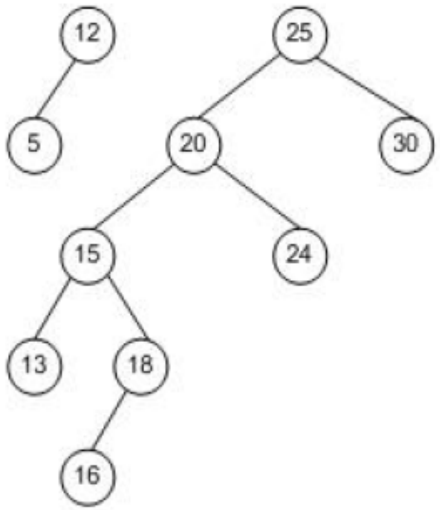
ZIG-ZAG
(SEMPLIFICATO)



VUOTO

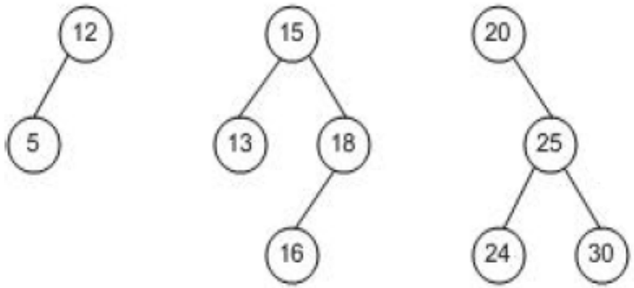
ESEMPIO

SPLAY (19)



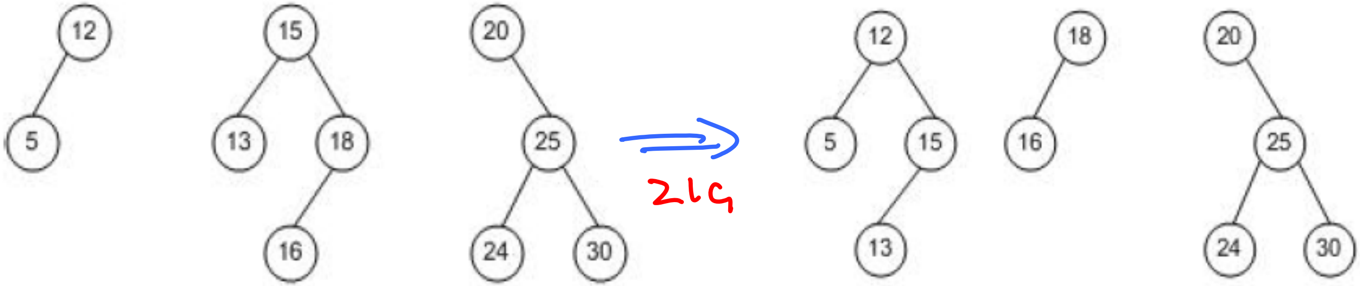
VUOTO

ZIG-ZIG

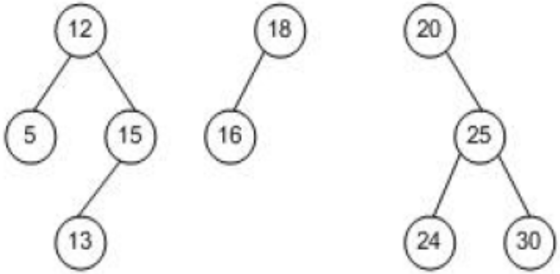


ESEMPIO

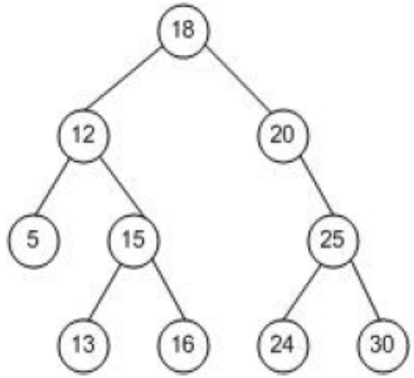
SPLAY (19)



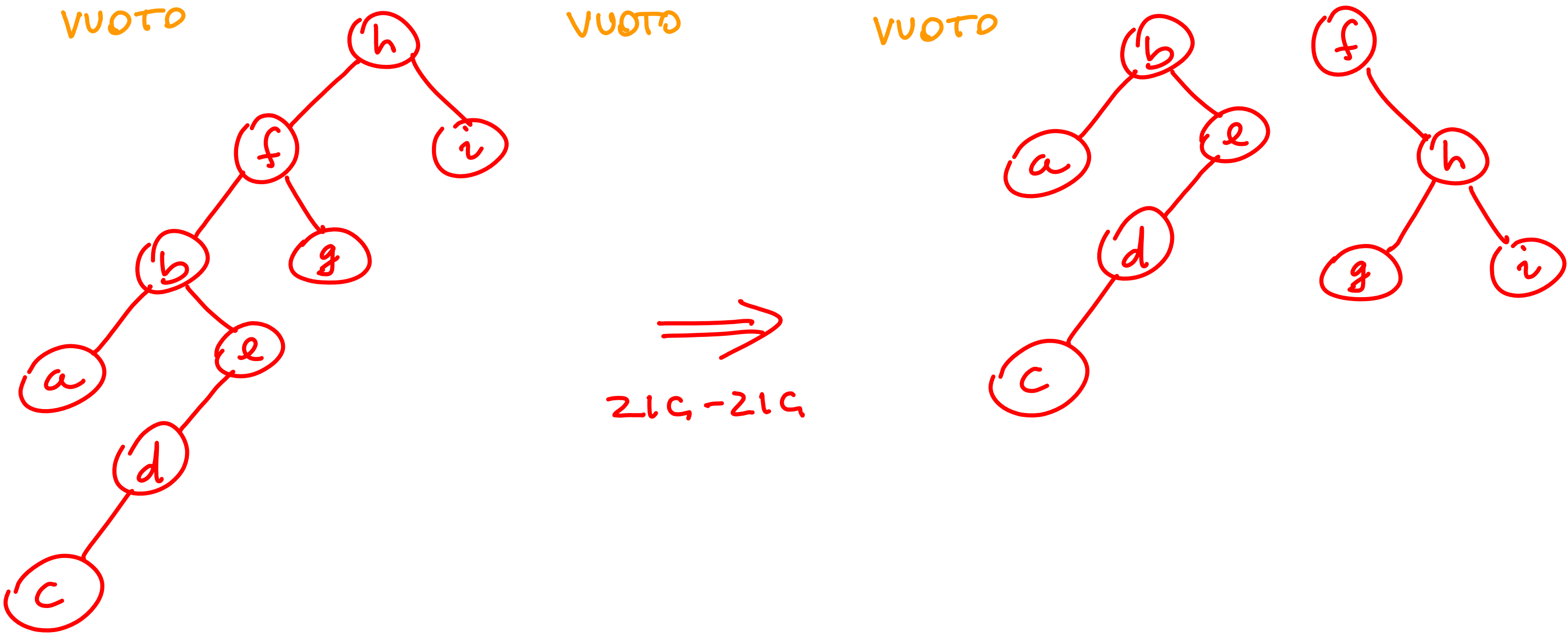
ESEMPIO SPLAY (19)



RIASSEMBLAGGIO

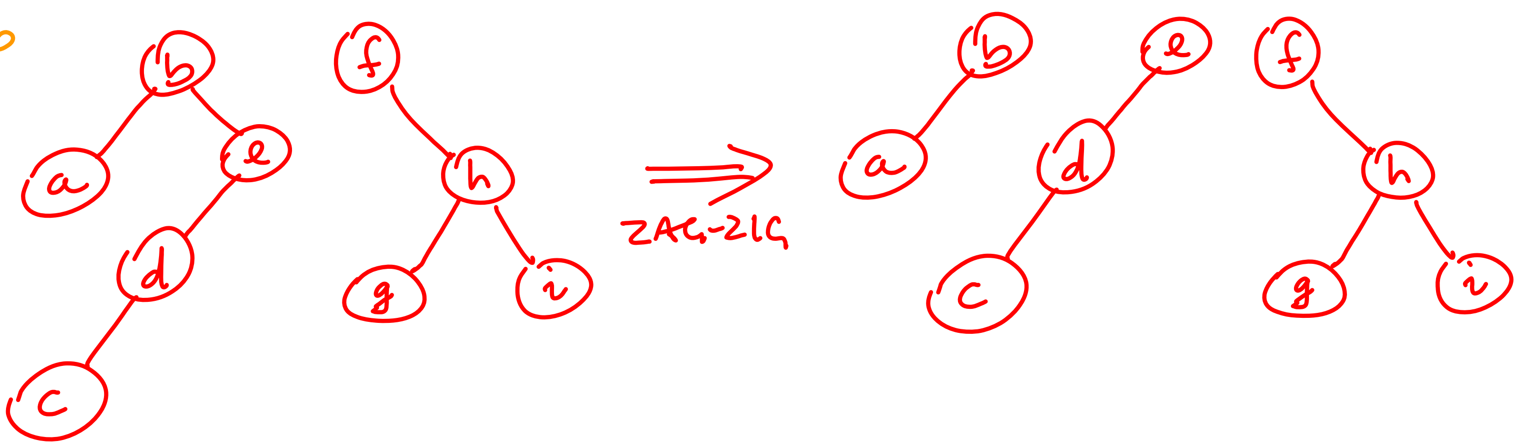


ESEMPIO 2: SPLAY(c)

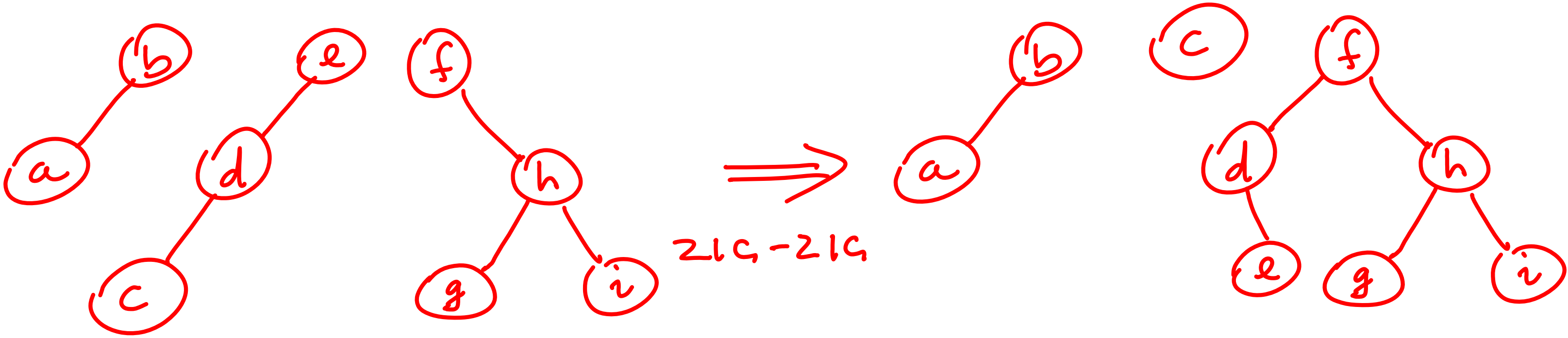


ESEMPIO 2: SPLAY (c)

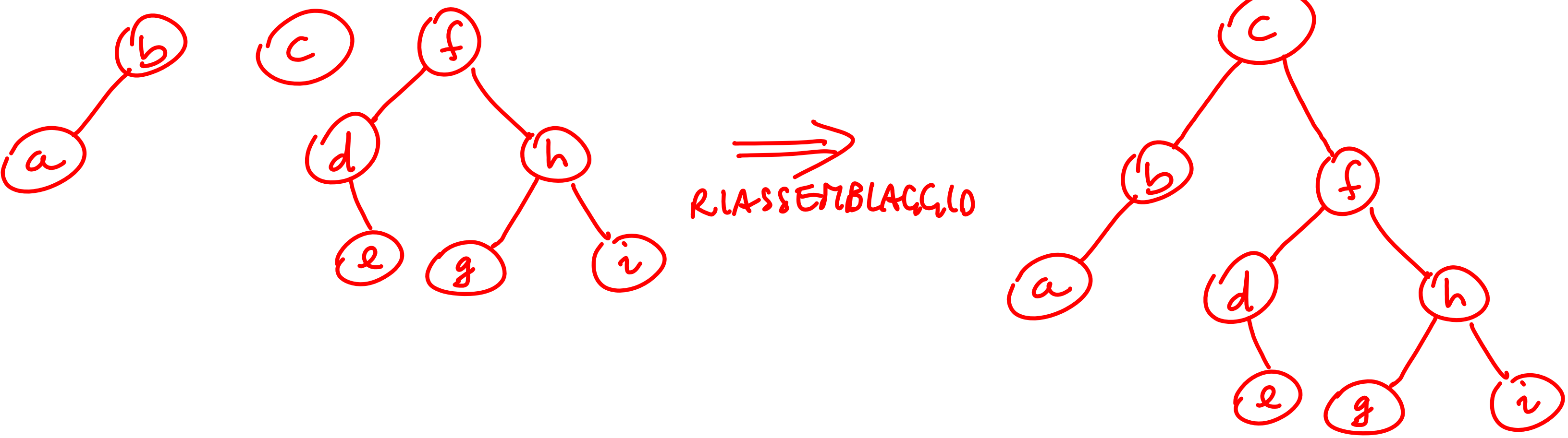
VUOTO



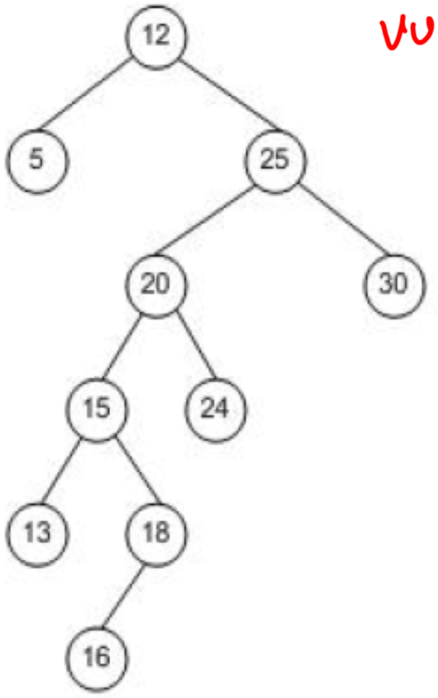
ESEMPIO 2: SPLAY (c)



ESEMPIO 2: SPLAY (c)

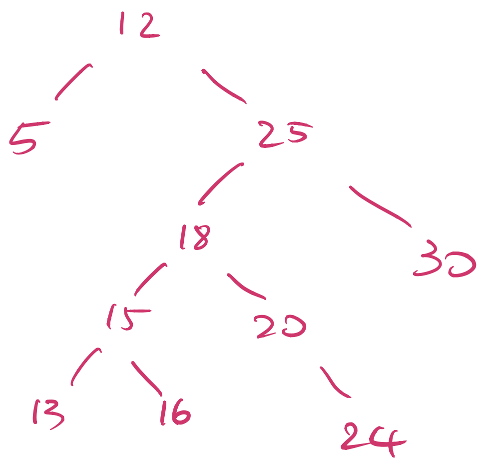


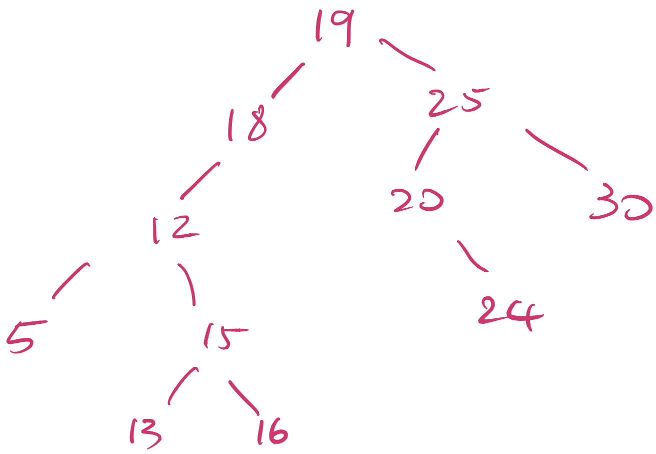
D



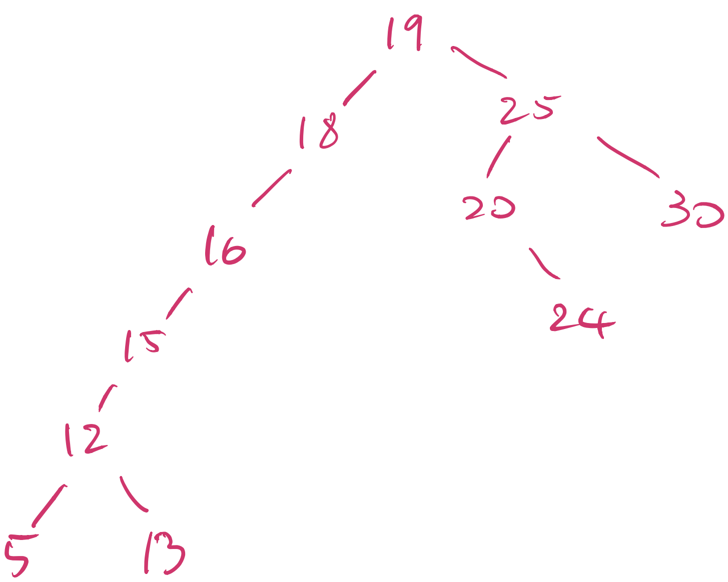
VUC

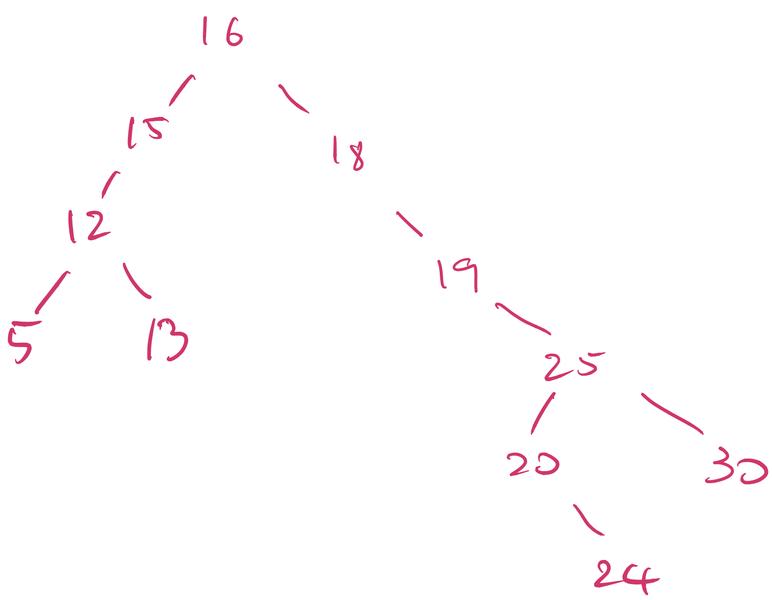
INSERT (19)

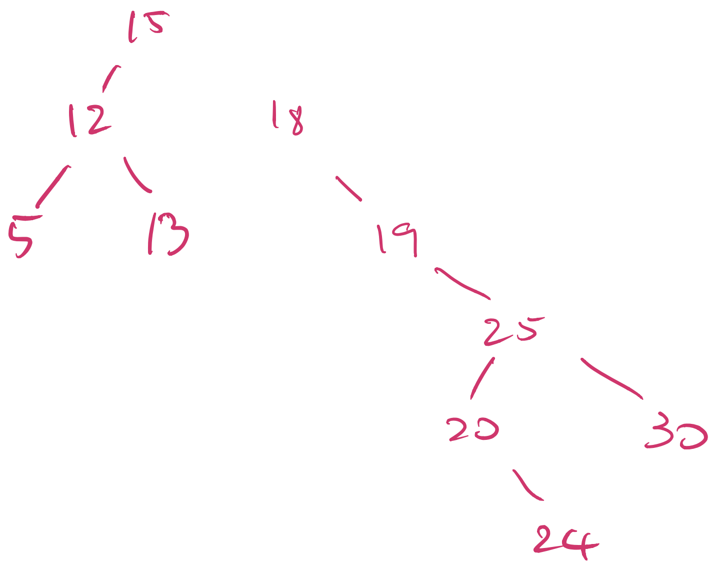




DELETE (16)






15
12
18
5
13
19
25
20
30
24