Corso di Laurea in INFORMATICA

a.a. 2011-2012

Algoritmi e Strutture Dati

MODULO 9

CODE CON PRIORITA'

Il tipo astratto coda con priorità: specifiche sintattiche e semantiche. Realizzazioni.

CODE CON PRIORITA'

IL NOME DERIVA DALL'IDEA CHE SI DISPONE DI UNA CODA A DI ELEMENTI CHE DEVONO ESSERE SERVITI RISPETTANDONE LA PRIORITA'.

PER SEMPLICITA' SI FA COINCIDERE LE PRIORITA' CON GLI ELEMENTI STESSI E SI ASSUMONO PRIORITA' DISTINTE, IN ACCORDO ALLA SPECIFICA DEL TIPO DI DATO INSIEME.

IN GENERALE, LE PRIORITA' POSSONO INTENDERSI COME PROPRIETA' ASSOCIATE AGLI ELEMENTI. E' POSSIBILE ANCHE CONSIDERARE LA PRIORITA' DI x COME UNA CHIAVE ASSOCIATA ALL'ELEMENTO key(x).

IL TIPO ASTRATTO CODA CON PRIORITA'

E' una collezione di elementi ognuno dei quali possiede una certa priorità (rappresentata ad es. da un numero naturale).

Esempio: un insieme di richieste di lavori, di cui alcuni più urgenti di altri.

- Una coda con priorità, a differenza di una coda ordinaria, non obbedisce ad una disciplina FIFO, poiché da essa si estrae ogni volta quello (o uno di quelli) con priorità più alta.
- È consuetudine che la **priorità più alta** possibile sia indicata dal numero **1**, e le priorità **inferiori** da **numeri** interi **più grandi**.
- L'operazione di estrazione dalla coda è allora un'operazione di estrazione del minimo

IL TIPO ASTRATTO CODA CON PRIORITA'

Le operazioni fondamentali sono allora:

- inserimento di un elemento con una data priorità;
- estrazione dell'elemento con il valore minimo di priorità;
- restituzione di tale minimo senza estrarlo dalla coda;
- test se la coda è vuota o no.

IL TIPO ASTRATTO CODA CON PRIORITA'

Si possono dare definizioni diverse a seconda che la coda si consideri costituita di coppie ⟨elemento, priorità⟩ oppure che si consideri ogni elemento come "contenente al proprio interno" la priorità.

Si possono poi definire altre operazioni, come:

- variazione della priorità di un elemento della coda;
- cancellazione di un elemento della coda;
- fusione di due code con priorità; ...

IL TIPO ASTRATTO CODA CON PRIORITA'

PUO' ESSERE CONSIDERATA UN CASO PARTICOLARE DI *INSIEME* CHE PERMETTE DI MANTENERE IL MINIMO IN UN INSIEME DI ELEMENTI (O CHIAVI) SUI QUALI E' DEFINITA UNA RELAZIONE "<" DI ORDINAMENTO TOTALE.

SPECIFICA SINTATTICA

TIPI: PRIORICODA,TIPOELEM, BOOLEAN

OPERATORI:

CREAPRIORICODA: ( ) → **PRIORICODA**

PRIORICODAVUOTA: (PRIORICODA) → **BOOLEAN**

INSERISCI: (TIPOELEM, PRIORICODA) → **PRIORICODA**

MIN: (PRIORICODA) → **TIPOELEM**

CANCELLAMIN: (PRIORICODA) → **PRIORICODA**

SPECIFICA SEMANTICA

TIPI: PRIORICODA: INSIEME DI CODE CON PRIORITA' CON ELEMENTI DI TIPO TIPOELEM

OPERATORI:

CREAPRIORICODA = A

POST: A = ∅

PRIORICODAVUOTA(A) = b

POST: True SE A = ∅, False ALTRIMENTI

INSERISCI (x,A) = A'

POST: A'= A∪ {x} (se x ∈ A allora A=A')

MIN(A) = x

PRE: A ≠ ∅

POST: x ∈ A e x ≤ y per ogni y ∈ A

CANCELLAMIN (A) = A'

PRE: A ≠ ∅

POST: A' = A – { x } con x = MIN(A)

RAPPRESENTAZIONE CON STRUTTURE SEQUENZIALI

SI PUO' RAPPRESENTARE UNA CODA CON PRIORITA' DI n ELEMENTI UTILIZZANDO STRUTTURE SEQUENZIALI.

Liste ordinate (in ordine crescente): l'elemento di priorità più alta, cioè il minimo, è il primo, ma l'inserimento è un inserimento al posto giusto in lista ordinata; quindi:

- estrazione del minimo: complessità O(1);
- inserimento (casi medio e peggiore): complessità O(n);

Liste non ordinate: l'inserimento può avvenire sempre in testa, ma l'estrazione è una ricerca del minimo in sequenza non ordinata, quindi:

- estrazione del minimo: complessità O(n);
- inserimento: complessità O(1);

Array: non migliora la situazione (se è ordinato, l'inserimento è O(n); se non è ordinato, è O(n) l'estrazione)

RAPPRESENTAZIONE CON ALBERI BINARI

GLI ELEMENTI DI UNA CODA CON PRIORITÀ C POSSONO ESSERE MEMORIZZATI NEI NODI DI UN ALBERO BINARIO B CHE DEVE AVERE LE PROPRIETÀ:

1. **L'ALBERO B È *QUASI PERFETTAMENTE BILANCIATO***
   - **SE k È IL LIVELLO MASSIMO DELLE FOGLIE, ALLORA B HA ESATTAMENTE 2^K-1 NODI DI LIVELLO MINORE DI k ;**
   - **TUTTE LE SUE FOGLIE DI LIVELLO k SONO ADDOSSATE A SINISTRA;**
2. **OGNI NODO CONTIENE UN ELEMENTO DI C CHE E' MAGGIORE DI QUELLO DEL PADRE**

rappresentazione

tutte le rappresentazione usate per gli alberi binari sono ammissibili

- rappresentazione collegata, eventualmente con puntatori figli-genitore
- rappresentazione tramite array (heap)

  particolarmente efficiente

ESEMPIO (CODA CON PRIORITA'): LA FIGURA MOSTRA UN ALBERO B, CHE VERIFICA LE PROPRIETÀ SUDDETTE E CONTIENE GLI ELEMENTI DELLA CODA CON PRIORITÀ C = { 5, 10, 8, 11, 13, 12, 9, 18, 3, 6 }. IL LIVELLO MASSIMO DELLE FOGLIE DI B E' 3 E CI SONO $2^3 - 1 = 7$ NODI DI LIVELLO ≤ 2. LE TRE FOGLIE DI LIVELLO 3 SONO ADDOSSATE A SINISTRA.

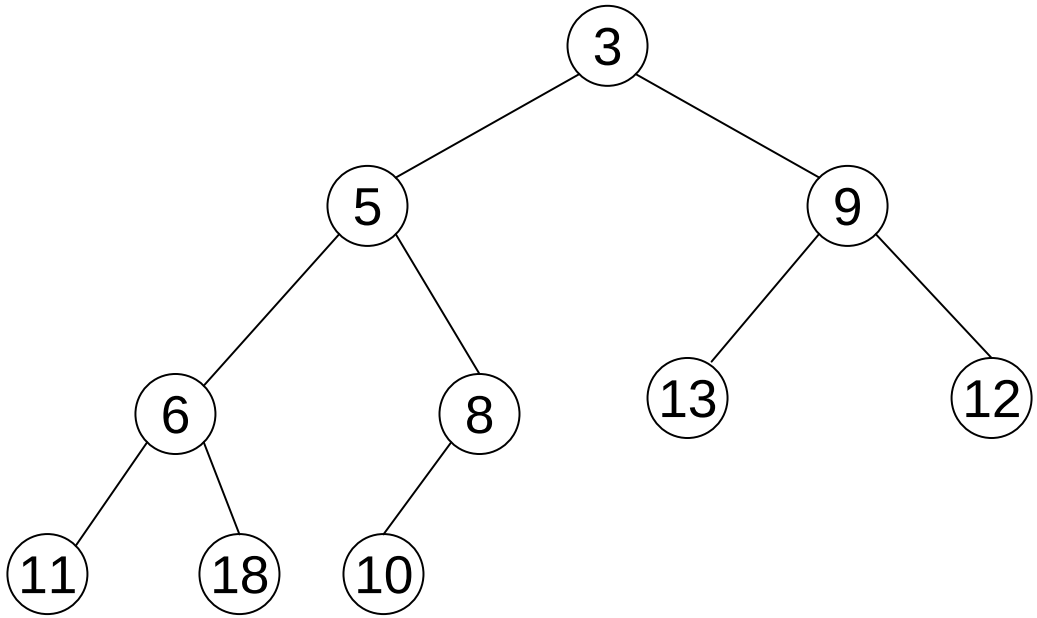


FIG. 1 ALBERO QUASI PERFETTAMENTE BILANCIATO PARZIALMENTE ORDINATO

PER REALIZZARE GLI OPERATORI SI OSSERVI CHE:

- **MIN RESTITUISCE IL CONTENUTO DELLA RADICE DELL'ALBERO B;**

- **INSERISCI DEVE INSERIRE UNA NUOVA FOGLIA IN MODO DA MANTENERE VERIFICATA LA PROPRIETÀ (1) E QUINDI FAR “SALIRE” L'ELEMENTO INTRODOTTO FINO A VERIFICARE LA PROPRIETÀ (2);**

- **CANCELLAMIN PREVEDE LA CANCELLAZIONE DELLA FOGLIA DI LIVELLO MASSIMO PIÙ A DESTRA, IN MODO DA MANTENERE VERIFICATA LA PROPRIETÀ (1) ED IL REINSERIMENTO DEL CONTENUTO DELLA FOGLIA CANCELLATA NELL'ALBERO PARTENDO DALLA RADICE E FACENDOLO "SCENDERE" IN MODO CHE L'ALBERO COSÌ MODIFICATO VERIFICHI ANCHE LA PROPRIETÀ (2).**

ESEMPIO (INSERISCI): VOLENDO INSERIRE L'ELEMENTO 4 NELL'ALBERO B DI FIG. 1, SI AGGIUNGE UN FIGLIO, CON 4, AL NODO 8, SI SCAMBIA 4 CON 8 E SUCCESSIVAMENTE 4 CON 5.

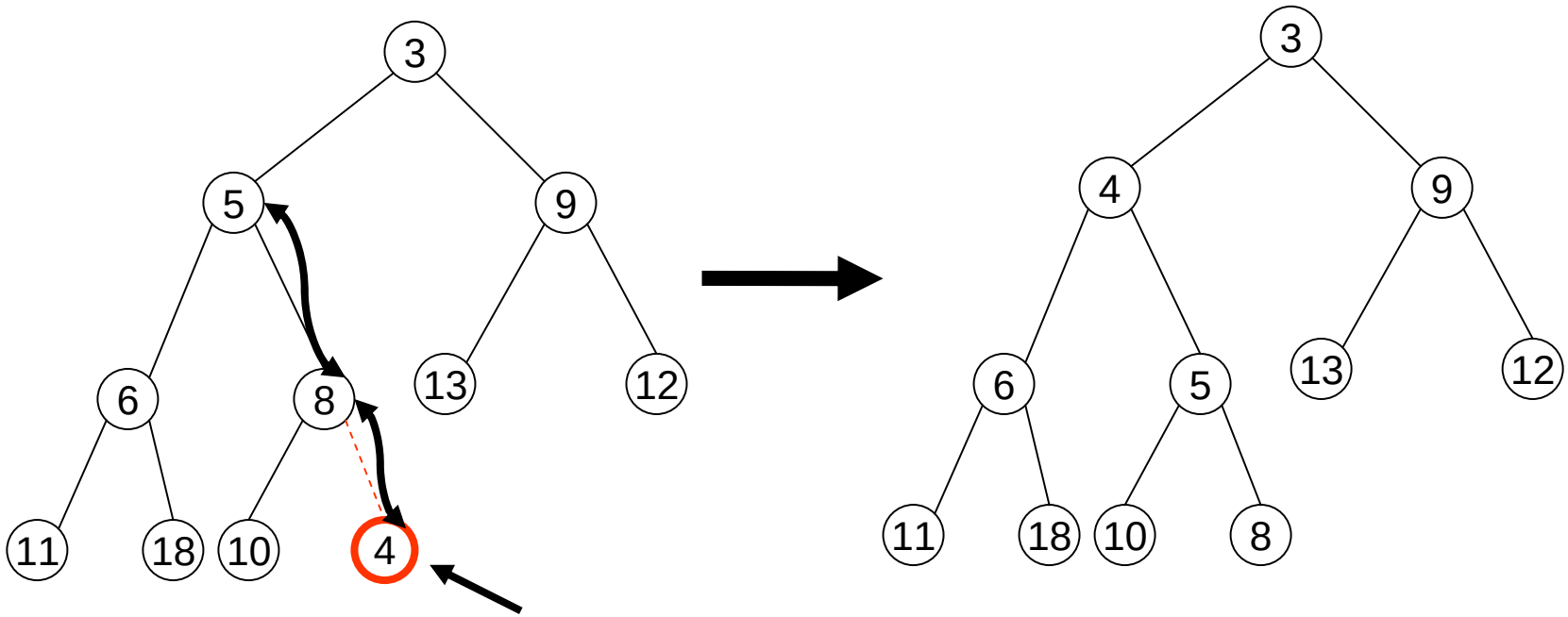


FIG. 2 INSERIMENTO DI 4

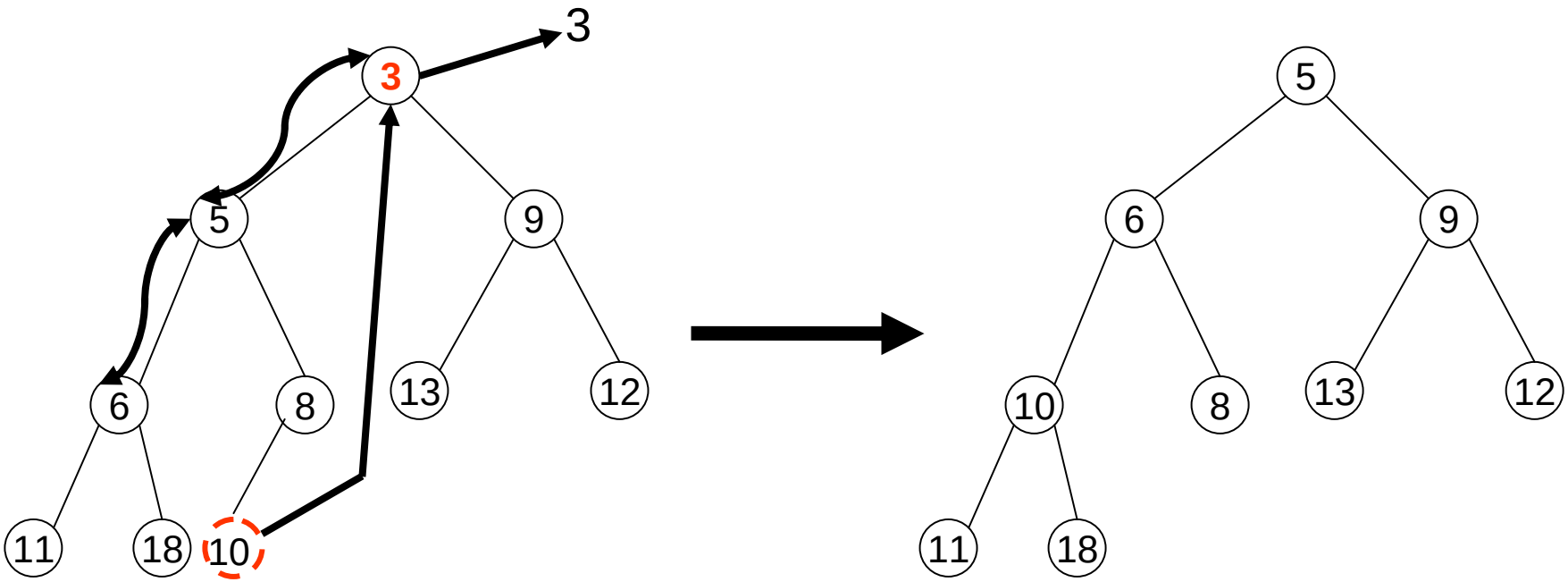


FIG. 3 CANCELLAZIONE DEL MINIMO

VOLENDO CANCELLARE 3 DALL'ALBERO DI FIG. 1 SI CANCELLA LA FOGLIA CONTENENTE 10, SI RICOPIA 10 NELLA RADICE, SI SCAMBIA 10 CON 5 E SUCCESSIVAMENTE 10 CON 6.

OSSERVAZIONE:

INSERISCI E CANCELLAMIN PREVEDONO DUE FASI, LA PRIMA DI *MODIFICA* DELLA STRUTTURA DELL'ALBERO (CONDIZIONATA DALLA PROPRIETÀ (1)), LA SECONDA DI *AGGIUSTAMENTO* DEGLI ELEMENTI IN BASE ALLE PRIORITÀ (PROPRIETÀ (2)).

NELLA PRIMA FASE SI DEVE NECESSARIAMENTE FARE RIFERIMENTO ALLA FOGLIA ALL'ESTREMA DESTRA DELL'ULTIMO LIVELLO: QUESTO EVIDENZIA L'UTILITÀ DI MANTENERE TRACCIA DI TALE FOGLIA E DI CONSEGUENZA

UNA DEFINIZIONE DEL TIPO PRIORICODA PUÒ ESSERE LA SEGUENTE:

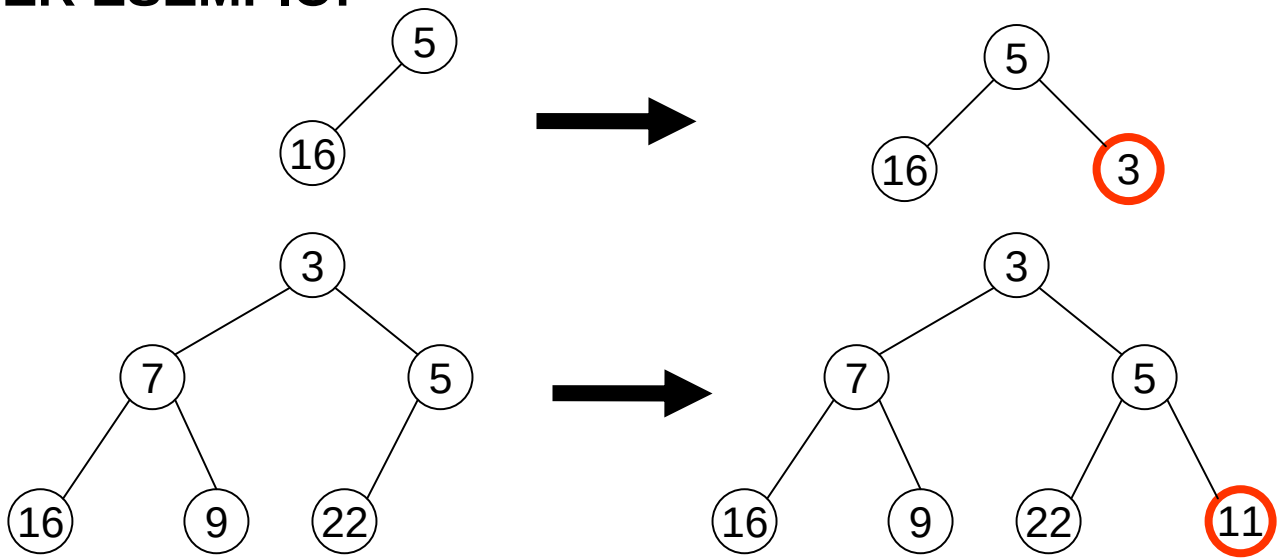
```
PrioriCoda=RECORD
albero:BinAlbero;
ultimo:nodo
END;
```

ANALIZZIAMO PIÙ IN DETTAGLIO INSERISCI.

LA PRIMA FASE, QUELLA DI MODIFICA DELLA STRUTTURA DELL'ALBERO, PREVEDE L'INSERIMENTO DI UNA FOGLIA "DOPO" L'ULTIMA. CASI PARTICOLARMENTE SEMPLICI SONO:

- **L'ALBERO È VUOTO: BASTA AGGIUNGERE L'ELEMENTO COME RADICE;**
- **L'ALBERO È COSTITUITO DALLA SOLA RADICE: L'ELEMENTO VIENE AGGIUNTO COME FIGLIO SINISTRO DELLA RADICE;**
- **L'ULTIMA FOGLIA È UN NODO FIGLIO SINISTRO: L'ELEMENTO VIENE AGGIUNTO COME FRATELLO DESTRO.**

PER ESEMPIO:



I FASE: MODIFICA DELLA STRUTTURA

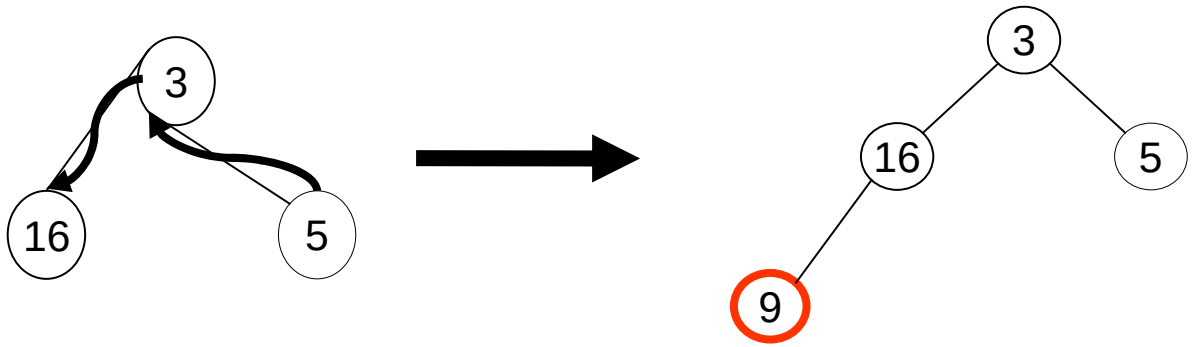
NEL CASO GENERALE LA FASE DI MODIFICA DELLA STRUTTURA PREVEDE UN PRIMO PASSO DI "*SALITA DA DESTRA*" ED UNO SUCCESSIVO DI "*DISCESA VERSO SINISTRA*" ALLO SCOPO DI INDIVIDUARE DOVE ATTACCARE IL NUOVO NODO.

IL PROCESSO DI SALITA (*SI PASSA DA FIGLIO A PADRE*) PROSEGUE FINTANTOCHÉ IL NODO CONSIDERATO È UN FIGLIO DESTRO (*LIVELLO COMPLETO*) OPPURE NON È STATA RAGGIUNTA LA RADICE

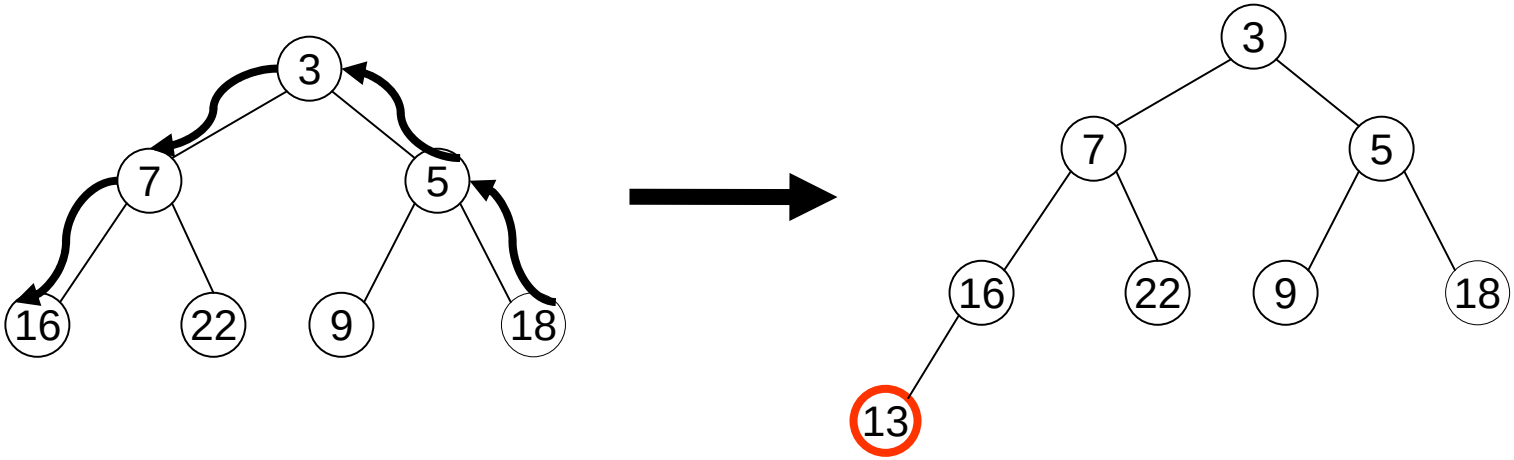
IL PROCESSO DI DISCESA (*SI PASSA DAL PADRE AL FIGLIO SINISTRO*) VIENE ITERATO FINO A QUANDO NON SI TROVA UNA FOGLIA. SOLO A QUESTO PUNTO SI PUÒ AGGIUNGERE IL NUOVO NODO COME FOGLIA.

NEL CASO IL LIVELLO NON SIA COMPLETO IL NUOVO NODO VERRA' AGGIUNTO A COMPLETARLO.

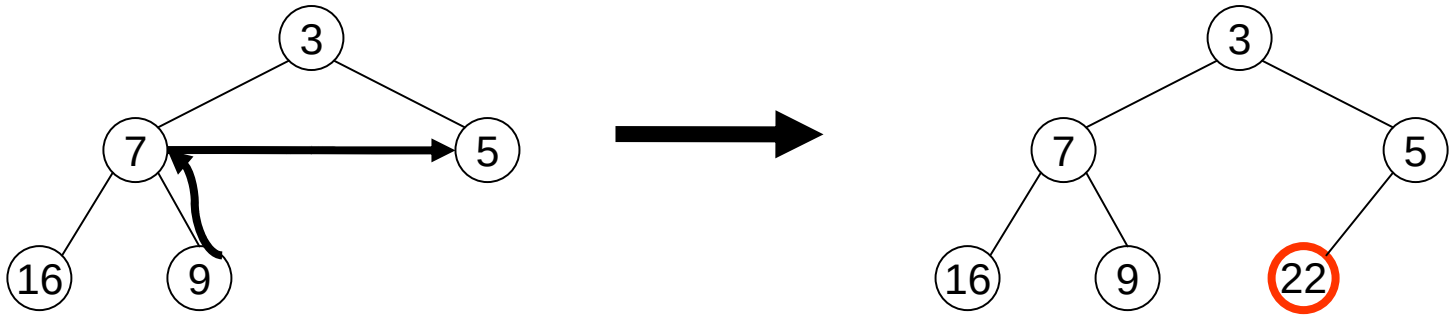
1° CASO



1° CASO

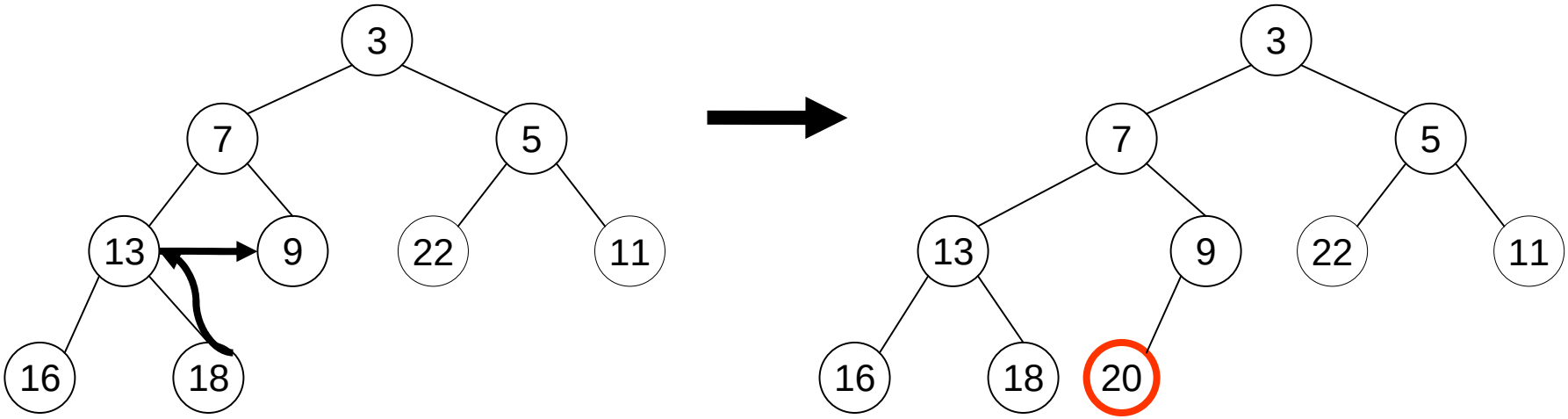


NEL SECONDO CASO, O NON SI GIUNGE ALLA RADICE, OPPURE SI GIUNGE ALLA RADICE DAL FIGLIO SINISTRO: SI SCENDE PARTENDO DAL FRATELLO DESTRO DELL'ULTIMO NODO CHE È STATO VISITATO IN SALITA COME FIGLIO DESTRO.



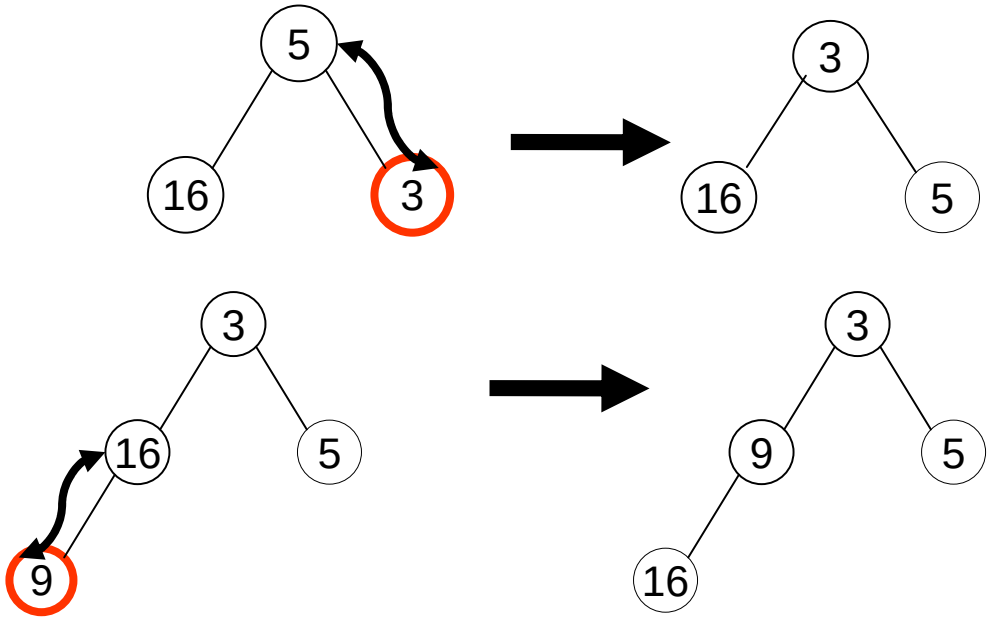
2° CASO

2° CASO



II FASE: AGGIUSTAMENTO

NELLA SECONDA FASE DI INSERISCI, CIOÈ L'AGGIUSTAMENTO DEGLI ELEMENTI IN BASE ALLE PRIORITÀ, SI PARTE DALLA FOGLIA INSERITA E SI CONFRONTA IL VALORE DI PRIORITÀ DEL NODO FIGLIO CON QUELLO DEL NODO PADRE: I VALORI VENGONO SCAMBIATI SE NON SODDISFANO LA PROPRIETÀ (2) E QUINDI SI RISALE L'ALBERO VERSO LA RADICE. QUESTO PROCESSO DI CONFRONTO-SCAMBIO SI RIPETE FINO A QUANDO NON SI TROVANO DUE ELEMENTI CHE SODDISFANO GIÀ LA PROPRIETÀ (2) OPPURE NON SI ARRIVA ALLA RADICE.



L'ALGORITMO DI INSERISCI:

```
if l'albero è vuoto then
   inserisci l'elemento come radice
else
  if l'albero ha solo la radice(che coincide con "ultimo")
then
    inserisci l'elemento come figlio sinistro della radice
  else
    if "ultimo" è un figlio sinistro then
      inserisci l'elemento come suo fratello destro
    else
      while il nodo è un figlio destro
        risali
      if il nodo attuale non è la radice then
        passa al fratello destro
        scendi verso sinistra fino ad arrivare ad una
        foglia
        inserisci l'elemento come figlio sinistro
        "ultimo" diventa la foglia inserita
   while il nodo corrente non è la radice e la sua
            priorità è minore di quella del padre
            scambia il contenuto di padre e figlio
```

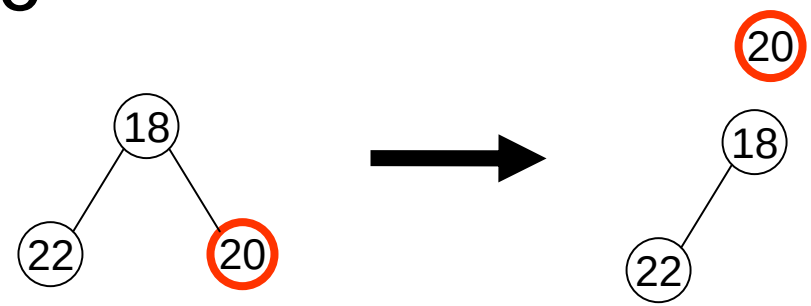
ANALIZZIAMO ORA CANCELLAMIN.

LA PRIMA FASE, QUELLA DI MODIFICA DELLA STRUTTURA DELL'ALBERO, PREVEDE LA CANCELLAZIONE DELLA FOGLIA DI LIVELLO MASSIMO PIÙ A DESTRA, MENTRE NELLA SECONDA FASE VA RISISTEMATO IL CONTENUTO.

CASI PARTICOLARMENTE SEMPLICI SONO:

- **L'ALBERO È COSTITUITO DALLA SOLA RADICE: SI CANCELLA L'INTERO ALBERO E NON È NECESSARIA LA SECONDA FASE;**
- **SI TRATTA DELL'ULTIMA FOGLIA O DI UN NODO FIGLIO DESTRO: DOPO LA CANCELLAZIONE L'ULTIMA FOGLIA DIVENTA IL NODO FRATELLO SINISTRO;**

PER ESEMPIO



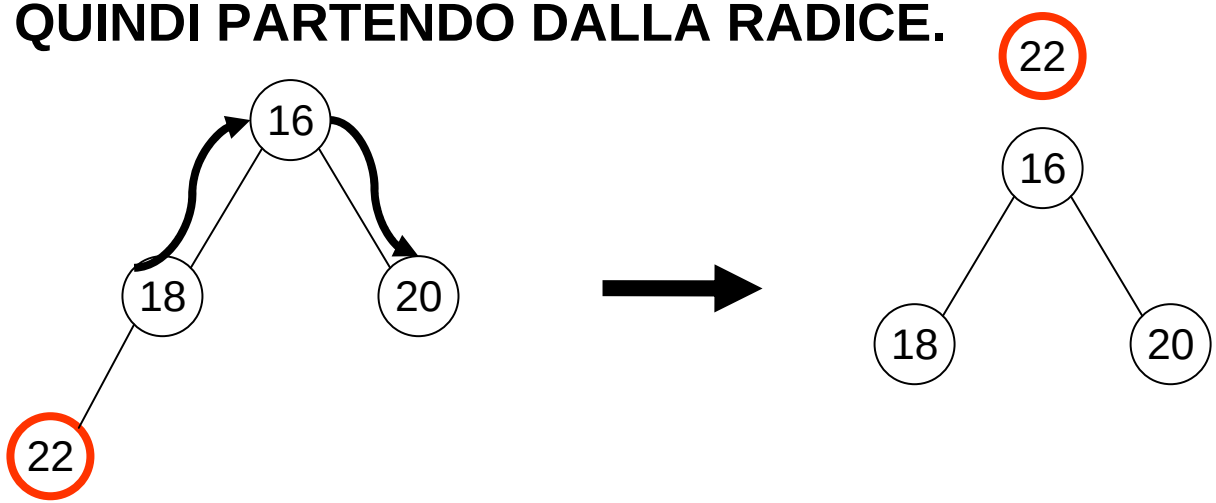
IL CASO PIU' GENERALE PREVEDE SEMPRE UNA PRIMA FASE DI MODIFICA ED UNA DI AGGIUSTAMENTO.

I FASE: MODIFICA

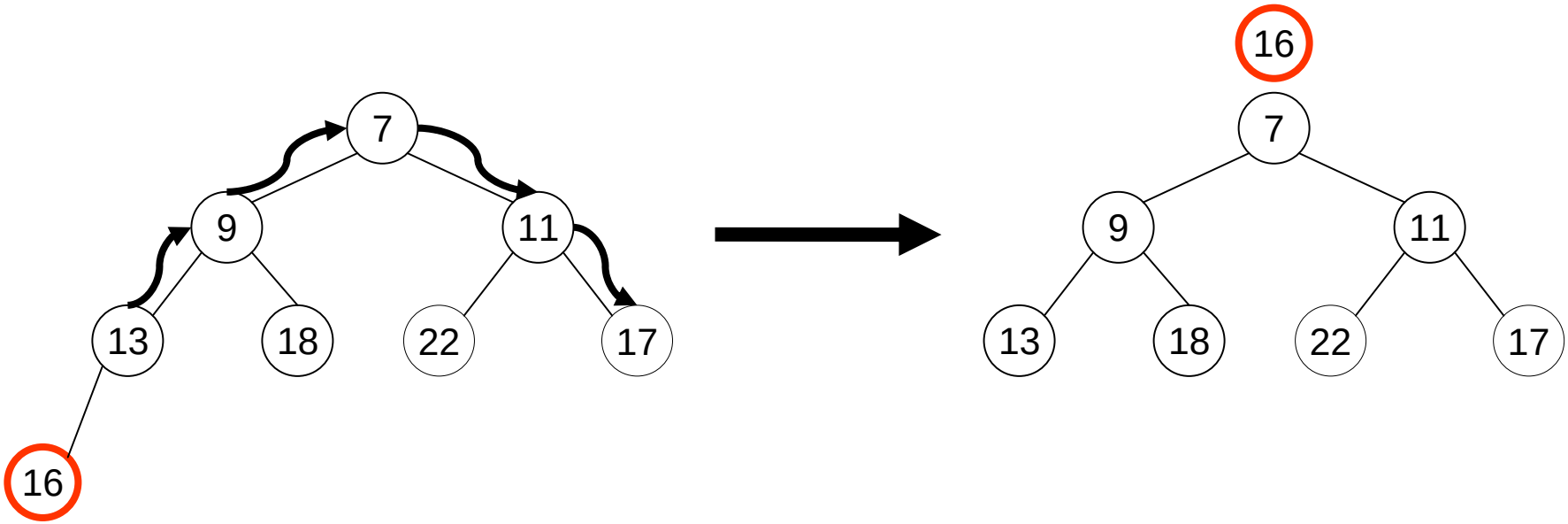
SI PROCEDE ATTRAVERSO PASSI DI "SALITA DA SINISTRA" E SUCCESSIVI PASSI DI "DISCESA VERSO DESTRA" ALLA RICERCA DELLA *NUOVA ULTIMA FOGLIA*. IL PROCESSO DI SALITA (SI PASSA DA FIGLIO A PADRE) PROSEGUE FINTANTOCHÉ IL NODO CONSIDERATO È UN FIGLIO SINISTRO OPPURE NON È STATA RAGGIUNTA LA RADICE, MENTRE IL PROCESSO DI DISCESA (SI PASSA DAL PADRE AL FIGLIO DESTRO) VIENE ITERATO FINO A QUANDO NON SI TROVA UNA FOGLIA. QUELLO CHE DIFFERENZIA I DUE CASI È SE IL LIVELLO MASSIMO VIENE O MENO SVUOTATO.

NEL PRIMO CASO SI GIUNGE ALLA RADICE DAL FIGLIO SINISTRO: SI SCENDE QUINDI PARTENDO DALLA RADICE.

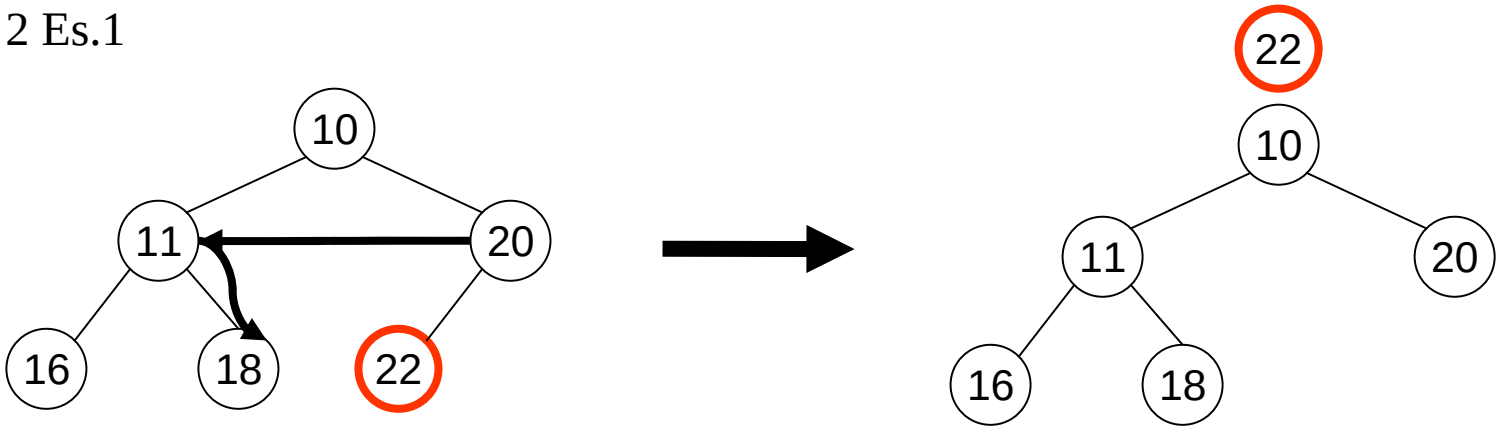
Caso 1 Es.1



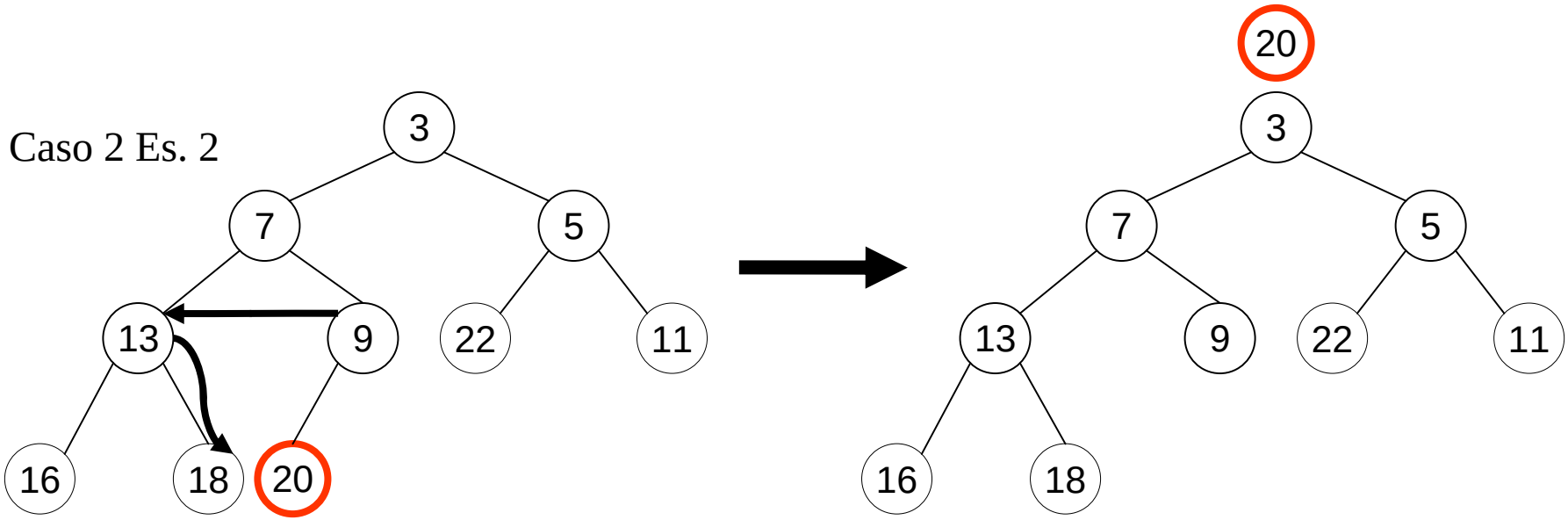
Caso 1 Es. 2



Caso 2 Es.1



Caso 2 Es. 2



NEL SECONDO CASO O NON SI GIUNGE ALLA RADICE, OPPURE SI GIUNGE ALLA RADICE DAL FIGLIO DESTRO: SI SCENDE PARTENDO DAL FRATELLO SINISTRO DELL'ULTIMO NODO CHE È STATO VISITATO IN SALITA COME FIGLIO DESTRO.

II FASE: AGGIUSTAMENTO/1

L'*AGGIUSTAMENTO* DEGLI ELEMENTI AVVIENE IN BASE ALLE PRIORITÀ, SI PARTE DALLA RADICE E SI RICERCA LA POSIZIONE NELLA QUALE INSERIRE IL CONTENUTO NELLA FOGLIA CANCELLATA IN MODO CHE SIA SODDISFATTA LA PROPRIETÀ (2).

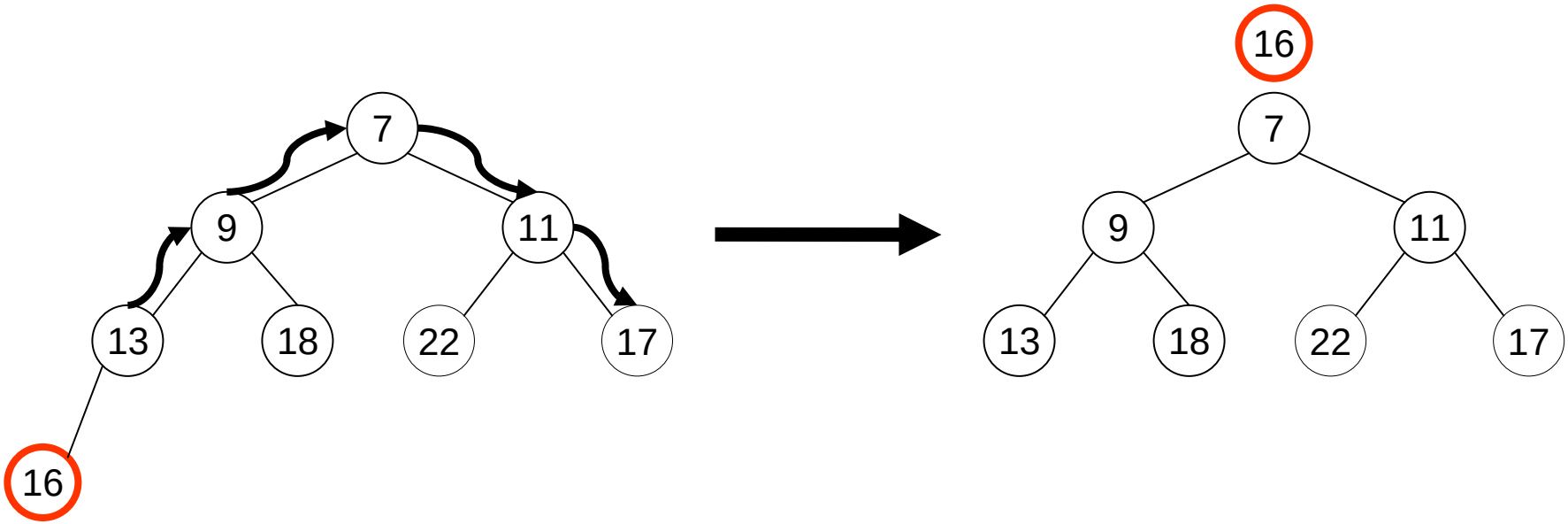
SI CONSIDERANO I FIGLI DI OGNI NODO ESAMINATO E, SE IL CONTENUTO DEL NODO CANCELLATO HA PRIORITÀ MINORE, SI SCRIVE NEL NODO ATTUALE IL CONTENUTO DEL FIGLIO CON PRIORITÀ MAGGIORE, VALUTANDO SE LA POSIZIONE DI TALE FIGLIO PUÒ ACCOGLIERE IL CONTENUTO DEL NODO CANCELLATO.

II FASE: AGGIUSTAMENTO/2

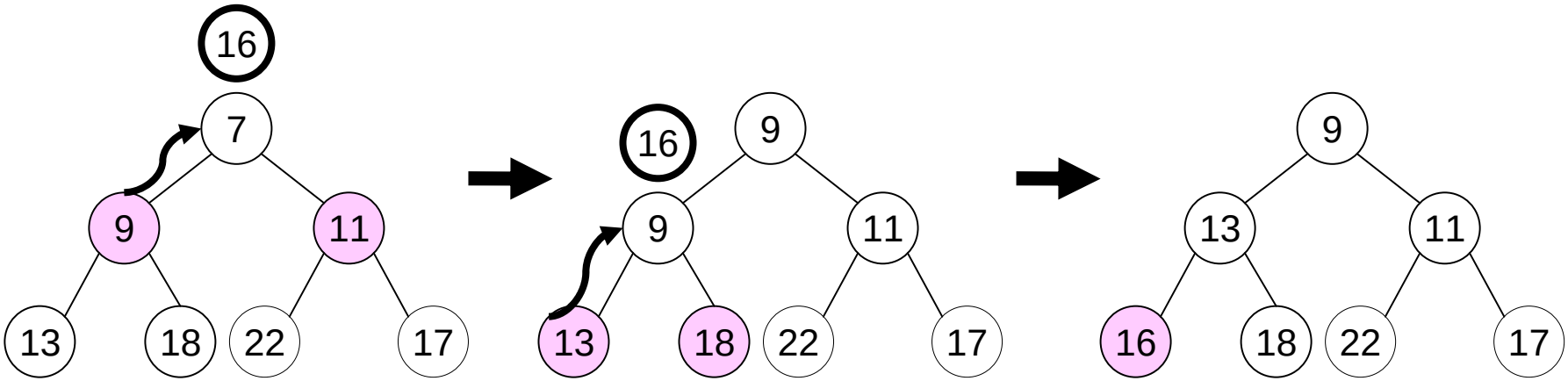
IL PROCESSO SI RIPETE FINO A QUANDO

- **NON SI TROVA UNA CONFIGURAZIONE CHE SODDISFA GIÀ LA PROPRIETÀ (2), E QUINDI IL CONTENUTO DEL NODO CANCELLATO HA PRIORITÀ MAGGIORE DI ENTRAMBI I FIGLI DEL NODO ATTUALE,**
- **OPPURE NON SI ARRIVA AL LIVELLO FOGLIARE.**

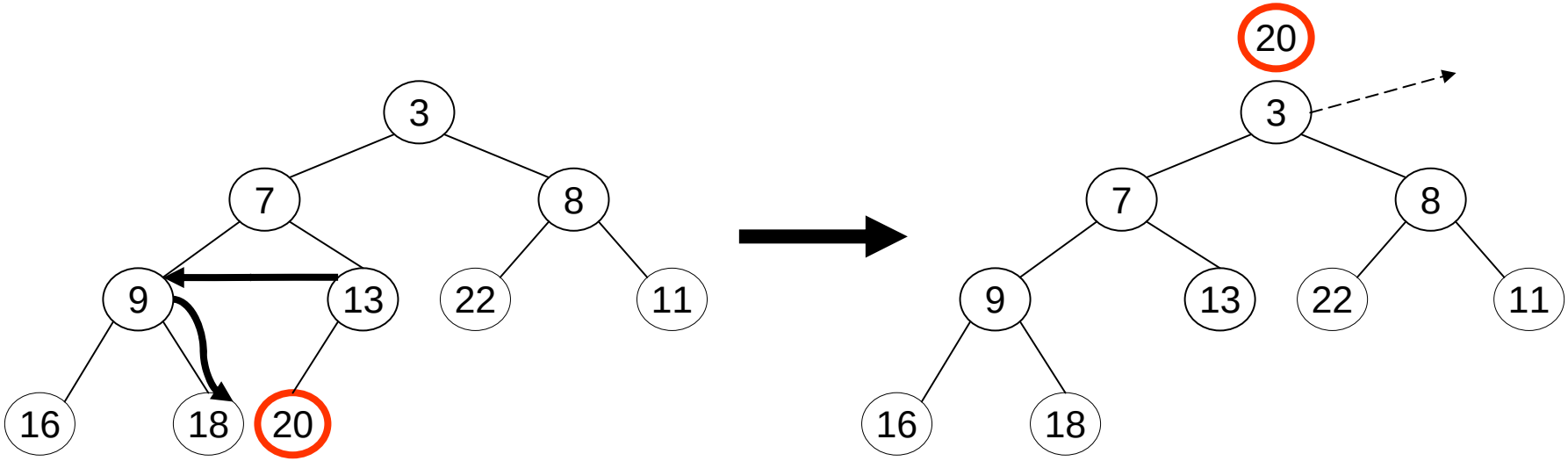
SI OSSERVI CHE, COME CONSEGUENZA DELLA PROPRIETÀ (1), IL CONFRONTO COINVOLGE ENTRAMBI I FIGLI, TRANNE IL CASO IN CUI SI È RAGGIUNTO IL PENULTIMO LIVELLO, DOVE PUÒ ESSERCI UN NODO CON IL SOLO FIGLIO SINISTRO.



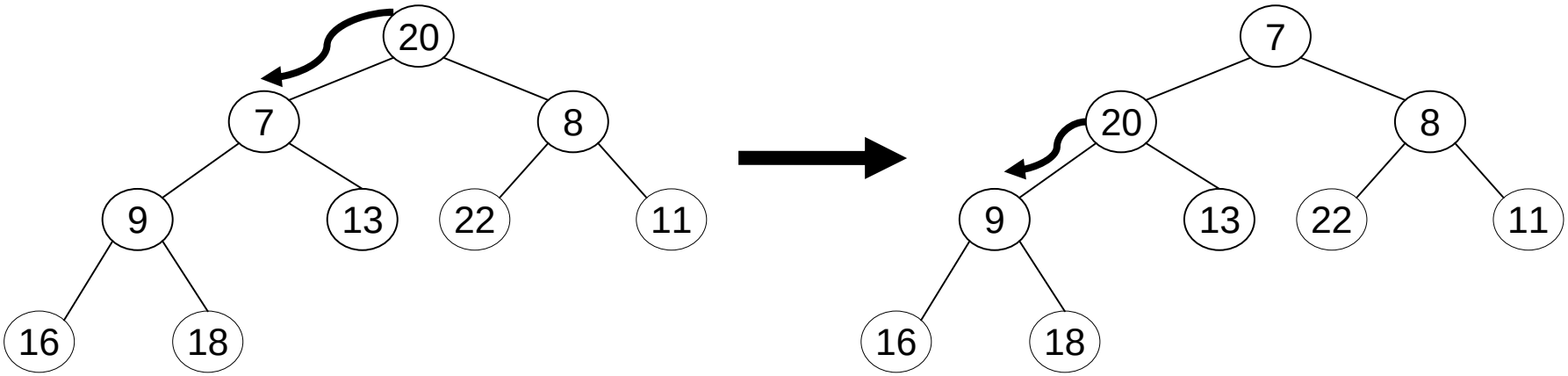
PRIMO PASSO: cancellazione dell’ultima foglia



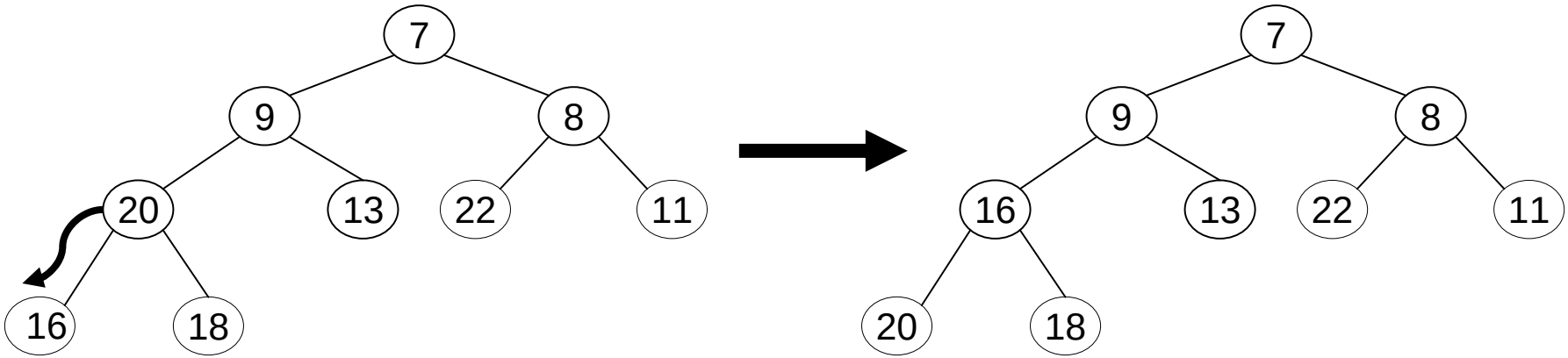
SECONDO PASSO: aggiustamento dei valori



PRIMO PASSO: cancellazione dell'ultima foglia



SECONDO PASSO: aggiustamento dei valori



Ulteriore aggiustamento dei valori

L'ALGORITMO DI CANCELLAMIN:

```
if l'albero non è vuoto then
  if l'albero ha solo la radice then
    cancella l'intero albero
  else
    copia il contenuto dell'ultima foglia nella radice,
    cancellala

/* inizio della fase di modifica della struttura */
  if ultimo è un figlio destro then
    il nuovo ultimo sarà il fratello sinistro
  else
    while il nodo è un figlio sinistro
      risali

    if non si è raggiunta la radice then
      passa al fratello sinistro
    scendi verso destra fino ad arrivare ad una foglia
    impostala come nuovo ultimo
```

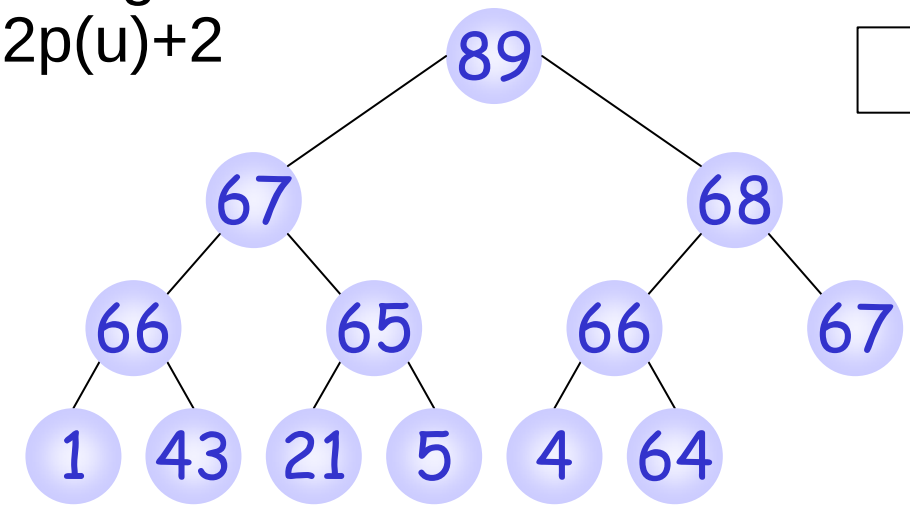
```
/* inizio della fase di aggiornamento delle priorità
partendo dalla radice */

while il nodo attuale non è una foglia e la sua priorità
          è minore di quella dei suoi figli do
  if  sono presenti entrambi i figli then
      scegli il figlio con priorità maggiore
  else
    seleziona l'unico figlio (che è il sinistro)
    scambia il contenuto del nodo attuale con quello del
      figlio  selezionato
    spostati sul figlio selezionato
```

rappresentazione tramite vettore

Quando l'albero binario, per realizzare la coda con priorità, è rappresentato con un vettore, ogni nodo v è memorizzato in posizione p(v)

- se v è la radice allora p(v)=0
- se v è il figlio sinistro di u allora p(v)=2p(u)+1
- se v è il figlio destro di u allora p(v)=2p(u)+2



89	0
67	1
68	2
66	3
65	4
66	5
67	6
1	7
43	8
21	9
5	10
4	11
64	12

rappresentazione tramite vettore

- vantaggi
  - grande efficienza in termini di spazio
    - l’occupazione può essere minima
  - facilità di navigazione
    - genitore i -> figli j
      - j = 2i + 1, 2i + 2
    - figlio i -> genitore j
      - j = (i – 1) / 2
- svantaggio
  - implementazione statica
    - possono essere necessari progressivi raddoppiamenti/dimezzamenti dell’array di supporto

REALIZZAZIONE CON HEAP

GLI ELEMENTI SONO DISPOSTI NEL VETTORE H (*HEAP*) NELL'ORDINE IN CUI SI INCONTRANO VISITANDO L'ALBERO PER LIVELLI CRESCENTI ED ESAMINANDO DA SINISTRA VERSO DESTRA I NODI ALLO STESSO LIVELLO.

IN TAL CASO, SI HA CHE:

- **H[1] E' L'ELEMENTO CONTENUTO NELLA RADICE DI B;**
- **H[2i] E H[2i+1] SONO GLI ELEMENTI CORRISPONDENTI AL FIGLIO SINISTRO E AL FIGLIO DESTRO DI H[i]**

PER LE PROPRIETA' DELL'ALBERO (QUASI PERFETTAMENTE BILANCIATO E PARZIALMENTE ORDINATO) IL VETTORE E' COMPLETAMENTE RIEMPITO.