Azzolini Riccardo 2019-05-06

Alberi 2-3

1 Inserimento

Procedura INSERT(v, x)
  begin
    $p \coloneqq$ FIND(v, x);
    if not ISFIGLIO(p, x) **then**
      begin
        aggiungi in ordine il nodo x ai figli di p;
        RIDUCI(p);
      end
  end

1. Si usa FIND per determinare il padre p del nodo da aggiungere.
2. Se è già presente un nodo contenente x, la procedura termina (perché in quest'implementazione non vengono gestiti valori duplicati). Altrimenti, si aggiunge una nuova foglia come figlio di p, nella posizione corretta per mantenere ordinati i figli.
3. Dopo l'aggiunta della foglia, si possono verificare delle situazioni illegali:
   - p può avere più di 3 figli;
   - il valore inserito può essere il nuovo massimo del sottoalbero, rendendo quindi necessario un aggiornamento delle etichette.

   Si chiama allora la procedura RIDUCI, che ha la funzione di risolvere tali situazioni, ripristinando così un albero 2-3 valido.

Procedura RIDUCI(v)
  if v ha 4 figli (f_1, f_2, f_3, f_4) **then**
    begin
      crea un nodo v';
      trasferisci a v' i primi due figli di v (f_1, f_2),
      aggiornando opportunamente i vari riferimenti;
      if v è radice **then**
        begin
          crea una nuova radice r';
          $F_1(r') \coloneqq v'$;
          $F_2(r') \coloneqq v$;
          padre$(v') \coloneqq r'$;

```
            padre(v) := r';
        end
    else
        begin
            u := padre(v);
            poni v' figlio di u, immediatamente a sinistra di v;
            RIDUCI(u);
        end;
end
else
    aggiorna solo le etichette
```

RIDUCI adotta una strategia ricorsiva: risolve il problema localmente e lo riporta eventualmente al livello superiore.

- Se il nodo da sistemare, v, ha al massimo 3 figli, RIDUCI deve solo aggiornare le etichette, propagando l'eventuale nuovo massimo del sottoalbero fino alla radice, se necessario.[1]
- Altrimenti, se v ha 4 figli:
  1. v viene diviso in 2, creando un nuovo fratello sinistro (o equivalentemente destro), v', al quale si trasferiscono i primi (ultimi) due figli di v;
  2. Se v era la radice, si crea una nuova radice con figli v' e v, e di conseguenza aumenta l'altezza dell'albero.[2] Altrimenti, si aggiunge v' ai figli di $u = \text{padre}(v)$, immediatamente a sinistra (destra) di v, e si riduce ricorsivamente u, che adesso potrebbe avere a sua volta 4 figli. La ricorsione termina quando u ha 2 figli prima dell'aggiunta di v' o quando si crea una nuova radice.

1.1 Complessità

Il tempo di calcolo impiegato da un inserimento è $\Theta(h) = \Theta(\log n)$ in ogni caso, dove h è l'altezza dell'albero (con $h = \Theta(\log n)$ perché l'albero è bilanciato). Infatti, l'inserimento prevede sempre una discesa dalla radice fino alle foglie, seguita eventualmente da una risalita fino (al massimo) alla radice per la correzione di situazioni illegali.

In particolare, l'operazione RIDUCI ha costo $O(h) = O(\log n)$:

- $O(1)$ nel caso migliore, quando non si creano situazioni illegali in seguito all'inserimento;

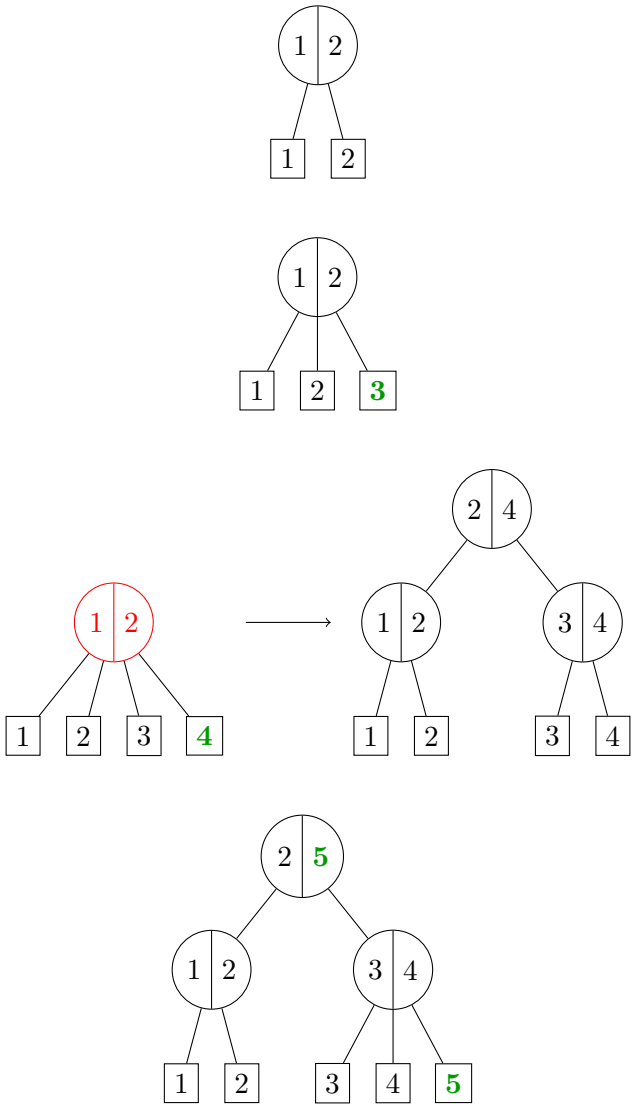
[1] Il nuovo massimo si propaga al livello superiore quando proviene dall'ultimo sottoalbero, altrimenti è sufficiente aggiornare le etichette di v.

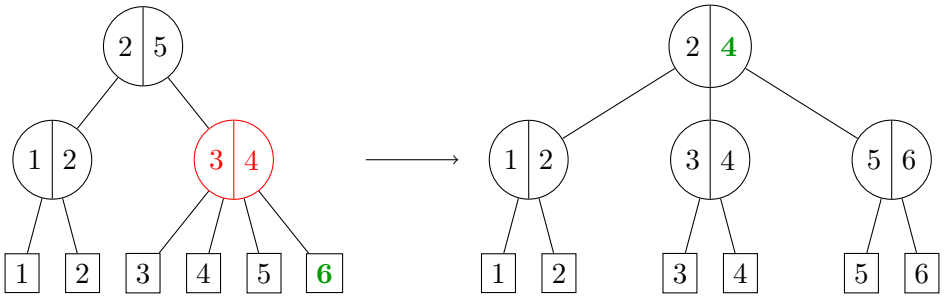
[2] Aggiungendo la radice, tutte le foglie sprofondano contemporaneamente di un livello, quindi l'altezza aumenta ma l'albero rimane bilanciato.

- $\Theta(h)$ nel caso peggiore, quando aumenta l'altezza dell'albero e/o è necessario aggiornare le etichette fino alla radice.

1.2 Esempi

Si costruisce un albero 2-3 a partire dalla sequenza $1, 2, 3, 4, 5, 6$:





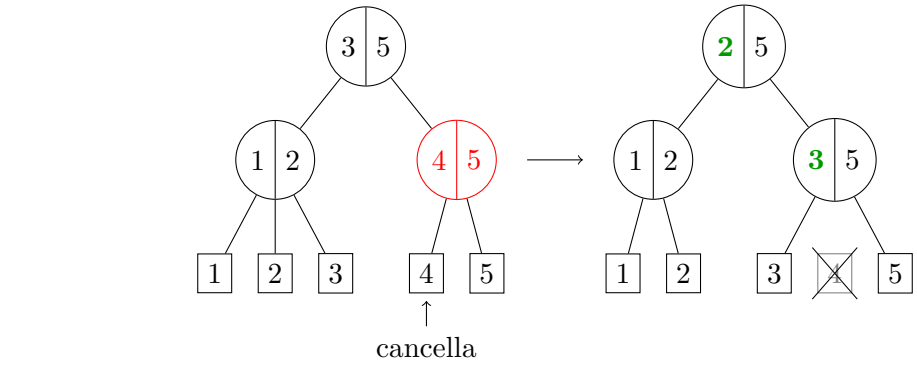
2 Cancellazione

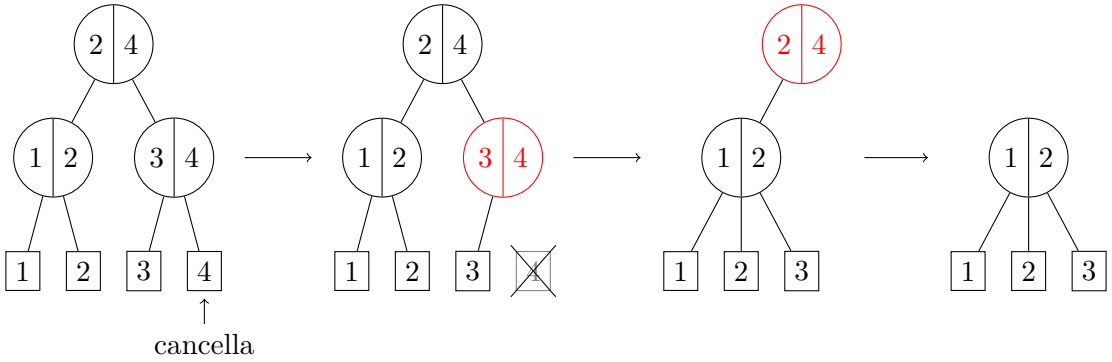
1. Si cerca il padre della foglia contenente il valore da cancellare (FIND).
2. Si elimina tale foglia, e poi si risolvono eventuali situazioni illegali.
   - Se il padre aveva inizialmente 3 figli, adesso ne ha ancora 2, perciò è sufficiente aggiornare le etichette.
   - Se il padre aveva 2 figli, dopo la cancellazione rimane con un figlio unico.
     - Se possibile, il padre "adotta" un nipote: l'ultimo figlio del fratello a sinistra o il primo del fratello a destra, purché tale fratello abbia 3 figli.
     - Altrimenti, il padre si elimina e "affida" l'unico figlio a uno dei fratelli, che hanno entrambi 2 figli e quindi spazio per un terzo (altrimenti sarebbe stata possibile un'adozione).

       Eliminando il padre, però, il nodo al livello superiore può rimanere a sua volta con un solo figlio, e in tal caso si ripete ricorsivamente questa procedura, fino eventualmente all'eliminazione della radice, con la quale l'altezza dell'albero diminuisce di 1.

Come per l'inserimento, la complessità in tempo della cancellazione è sempre $\Theta(\log n)$.

2.1 Esempi





3 Sintesi delle caratteristiche

Le operazioni sugli alberi 2-3:

- 2-3 hanno prestazioni garantite, $\Theta(\log n)$, dato che tali alberi sono bilanciati;
- sono concettualmente semplici, ma in pratica l'implementazione risulta complicata (soprattutto per la gestione delle etichette).

Al contrario, gli alberi binari di ricerca hanno una definizione particolarmente semplice, e sono quindi più facili da implementare, ma non sono sempre bilanciati, e di conseguenza le prestazioni non sono garantite.