ESERCIZIO DI ASD DEL 16 MARZO 2009

Binary Search Tree Colorabile

Algoritmo Ricorsivo. Procedendo ricorsivamente abbiamo che:

- Se l'albero è *vuoto*, allora contiene un'unica foglia fittizia NIL, quindi il colore deve essere NERO e l'altezza nera deve essere 0.
- Se l'albero non è vuoto ed il colore C è ROSSO, allora i due figli devono essere NERI ed avere altezza nera di uno inferiore.
- Se l'albero non è vuoto ed il colore C è NERO, allora ognuno dei due figli può essere o NERO con altezza nera inferiore di uno o ROSSO con la stessa altezza nera.

Algorithm 1 CheckRBTree(T, C, h)

```
return NodeCheck(root[T], C, h)
```

Algorithm 2 NodeCheck(x, C, h)

```
if x = NIL then
  if C = RED or h ≠ 0 then
    return FALSE
  else
    return TRUE
  end if
else
  if C = RED then
    return NodeCheck(left[x], BLACK, h − 1) and
           NodeCheck(right[x], BLACK, h − 1)
  else
    return (NodeCheck(left[x], BLACK, h − 1) or
            NodeCheck(left[x], RED, h)) and
           (NodeCheck(right[x], BLACK, h − 1) or
            NodeCheck(right[x], RED, h))
  end if
end if
```

Date: 16 Marzo 2009.

Correttezza.

Lemma 1. NodeCheck(x, C, h) *termina restituendo TRUE se e soltanto se il sottoalbero T_x radicato in x può essere un RB-tree con x di colore C ed altezza nera h.*

Dimostrazione. Procediamo per induzione sull'altezza di T_x.
BASE. T_x ha altezza 0.

x è NIL. Quindi x può assumere solo colore NERO e l'altezza nera di T_x è 0. Effettivamente in questo caso la procedura ritorna TRUE se e soltanto se C è NERO ed h è 0 (linee 1–6).
PASSO. T_x ha altezza n.

HpInd) Vale la tesi per tutti gli alberi di altezza minore di n.

Ts) Vale la tesi per T_x.

Se C è ROSSO, allora T_x puó essere un RB-Tree di altezza nera h con x ROSSO se e soltanto se sia $left[x]$ che $right[x]$ possono essere colorati di NERO e diventare radici di due RB-tree di altezza nera $h-1$. Nel caso C sia ROSSO le linee 8–9 effettuano questo controllo e le due chiamate ricorsive terminano correttamente per ipotesi induttiva. t Se C è NERO, allora T_x può essere un RB-Tree di altezza nera h con x NERO se e soltanto se valgono le due condizioni seguenti:

1. $left[x]$ può essere NERO e radice di un RB-Tree di altezza nera $h-1$ oppure $left[x]$ può essere ROSSO e radice di un RB-Tree di altezza nera h;
2. $right[x]$ può essere NERO e radice di un RB-Tree di altezza nera $h-1$ oppure $right[x]$ può essere ROSSO e radice di un RB-Tree di altezza nera h.

Le linee di codice 10–11 effettuano questi controlli e le chiamate ricorsive terminano correttamente per ipotesi induttiva. □

Theorem 1. CheckRBTree(T, C, h) *termina restituendo TRUE se e soltanto se l'albero T può essere un RB-tree con radice di colore C ed altezza nera h.*

Dimostrazione. La tesi segue immediatamente dal lemma. □

Complessità. Nel caso peggiore partendo da un albero di altezza k vengono effettuate 4 chiamate ricorsive su alberi di altezza $k-1$ più alcuni controlli che hanno costo $\Theta(1)$.

$$T(k) \quad \leq \quad \begin{cases} \Theta(1) & \text{if } k = 0 \\ 4 * T(k-1) + \Theta(1) & \text{if } k > 0 \end{cases}$$

Risolvendo l'equazione si ottiene $T(k) = O(4^k)$.