Functional Programming
Practicals 2: Red-Black Tree

Shin-Cheng Mu

FLOLAC 2022

In this practical we aim to prove some essential properties about red-black tree insertion in order to establish the correctness of the insertion algorithm. Some notes:

- In most proofs there could be many repetitive cases. It is sufficient to show only some representative cases.
- Proof about properties of the function *balance* are mostly routine, tedious, non-inductive proofs. However, these properties are needed in other proofs.

The code are adapted from Okasaki [Oka99]. Those who interested in figuring out how to perform deletion in red-black trees may check out Germane and Might [GM14].

1. Complete the definitions in the file `RedBlackOkasaki.hs`.

2. On (black) heights.

   (a) Prove that forall t, u and z, $\mathit{bheight}\ (\mathit{balance}\ t\ z\ u) = 1 + (\mathit{bheight}\ t \uparrow \mathit{bheight}\ u)$.

   (b) Prove that for all k and t, $\mathit{bheight}\ (\mathit{ins}\ k\ t) = \mathit{bheight}\ t$.
   Note: as a corollary, we have $\mathit{bheight}\ (\mathit{insert}\ k\ t)$ equals either $\mathit{bheight}\ t$ or $1 + \mathit{bheight}\ t$, depending on the root color of $\mathit{ins}\ k\ t$.

3. On balancing.

   (a) The function *isBalanced*, when taken literally as an algorithm, has time complexity $O(n^2)$, where n is the size of the input tree. Define

   $$\begin{array}{l} \mathit{isBalHeight} :: \mathrm{RBTree}\ a \rightarrow (\mathrm{Bool}, \mathrm{Nat}) \\ \mathit{isBalHeight}\ t = (\mathit{isBalanced}\ t, \mathit{bHeight}\ t)\ \ . \end{array}$$

   Derive an implementation of *isBalHeight* that runs in time linear to the size of the input tree.

   (b) Prove that for all t and u,

   $$\begin{array}{l} \mathit{isBalanced}\ t \wedge \mathit{isBalanced}\ u\ \wedge \\ \quad \mathit{bheight}\ t = \mathit{bheight}\ u \Rightarrow \mathit{isBalanced}\ (\mathit{balance}\ t\ x\ u)\ \ . \end{array}$$

(c) Prove that for all k and t, $\mathit{isBalanced}\ t \Rightarrow \mathit{isBalanced}\ (\mathit{ins}\ k\ t)$.
Note: since $\mathit{isBalanced}\ t \Rightarrow \mathit{isBalanced}\ (\mathit{blacken}\ t)$, as a corollary we have $\mathit{isBalanced}\ t \Rightarrow \mathit{isBalanced}\ (\mathit{insert}\ k\ t)$.

4. On color invariants.

   (a) Prove that for all t and u, $\mathit{isIRB}\ t \wedge \mathit{isRB}\ u \Rightarrow \mathit{isRB}\ (\mathit{balance}\ t\ x\ u)$.

   (b) Prove that for all t:

      1. $\mathit{isRB}\ t \wedge \mathit{color}\ t = \mathrm{R} \Rightarrow \mathit{isIRB}\ (\mathit{ins}\ k\ t)$,
      2. $\mathit{isRB}\ t \wedge \mathit{color}\ t = \mathrm{B} \Rightarrow \mathit{isRB}\ (\mathit{ins}\ k\ t)$.

   Hints: 1. The two properties shall be proved simultaneously in one inductive proof. 2. Since $\mathit{isRB}\ t \Rightarrow \mathit{isIRB}\ t$, the two properties above imply that $\mathit{isRB}\ t \Rightarrow \mathit{isIRB}\ (\mathit{ins}\ k\ t)$, which you may need in the proof.

   Note: since $\mathit{isIRB}\ t \Rightarrow \mathit{isRB}\ (\mathit{blacken}\ t)$, as a corollary we have $\mathit{isRB}\ t \Rightarrow \mathit{isRB}\ (\mathit{insert}\ k\ t)$.

References

[GM14] Kimball Germane and Matthew Might. Deletion: the curse of the red-black tree. *Journal of Functional Programming*, 24(4):423–433, 2014.

[Oka99] Chris Okasaki. Red-black trees in a functional setting. *Journal of Functional Programming*, 9(4):471–477, 1999.