Progetto di Linguaggi e Compilatori 1 – Parte 1
A.A. 2015/16

Gruppo 14

Marco Bucchiarone
Emanuele Tonetti
Francesco Zilli

1 Esercizio 1

La funzione Haskell `boundedMaximum(n [BST t])` data una lista di BST `t` trova, se esiste

$$max_n t = max(\forall x \in t \mid x < n).$$

La soluzione all'esercizio è implementata nel modulo BoundMax (`BoundMax.hs`) che viene importato nel `Main`. Nel `Main` la funzione viene chiamata fornendole gli argomenti inseriti a tastiera a runtime (usando le funzioni importate tramite `System.Environment`.

La `boundedMaximum` viene costruita sulla `foldl1`: `MaxNT` individua il massimo tra i minoranti nella lista costruita dalla funzione di visita del BST `getmin` che sfrutta la struttura dati considerata. Il modulo `BoundMax` testabile nell'interprete di GHC tramite i test case forniti nel file `queryEs1.txt`, il quale comprende anche una rappresentazione grafica di alcuni degli alberi usati per una maggiore leggibilit.

Il primo esercizio è stato risolto tramite la funzione boundedmaximum , la quale accetta un intero n e una lista di BST x e ritorna una lista di interi, che sono l'elenco di tutti i maxnt degli elementi di ciascun BST.

Maxnt è la funzione che passato un BST ritorna l'elemento dello stesso che è il maggiore dei minoranti, che è ricavato applicanto la funzione max, attraverso un foldl1 aumentato di funzionalità,

alla lista degli elementi strettamente minori di n ricavata con una funzione di visita del BST in modo da sfruttare la natura della struttura dati considerata.

La funzione è testabile attraverso ghci e si forniscono opportuni test case dentro il file es1queryghci.txt, che comprende una rappresentazione pi leggibile dei BST proposti in esso.

il metodo e stato impla

2 es2