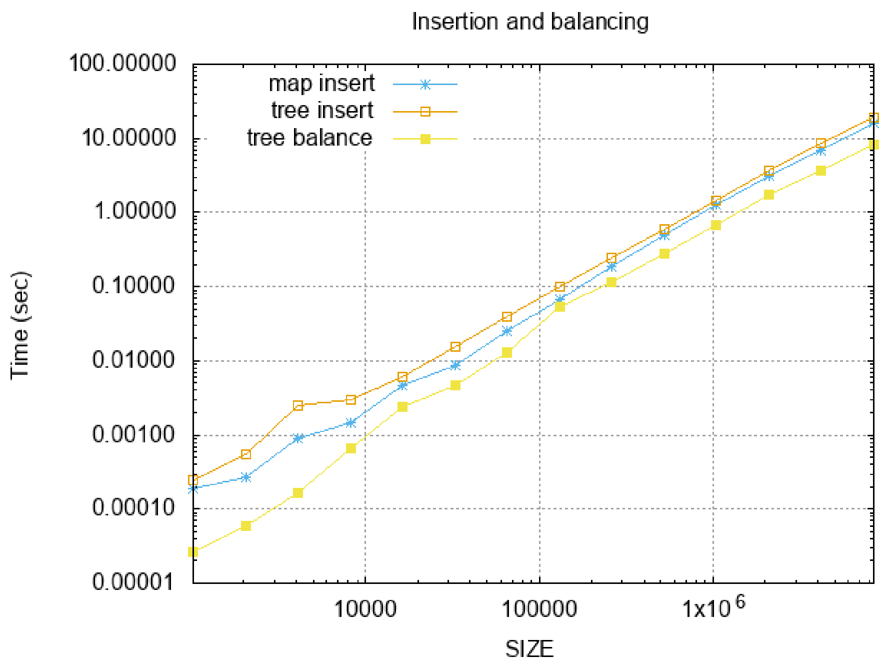
Title: Binary Search Tree report
Author: Luca Ciuffreda, Piero Coronica

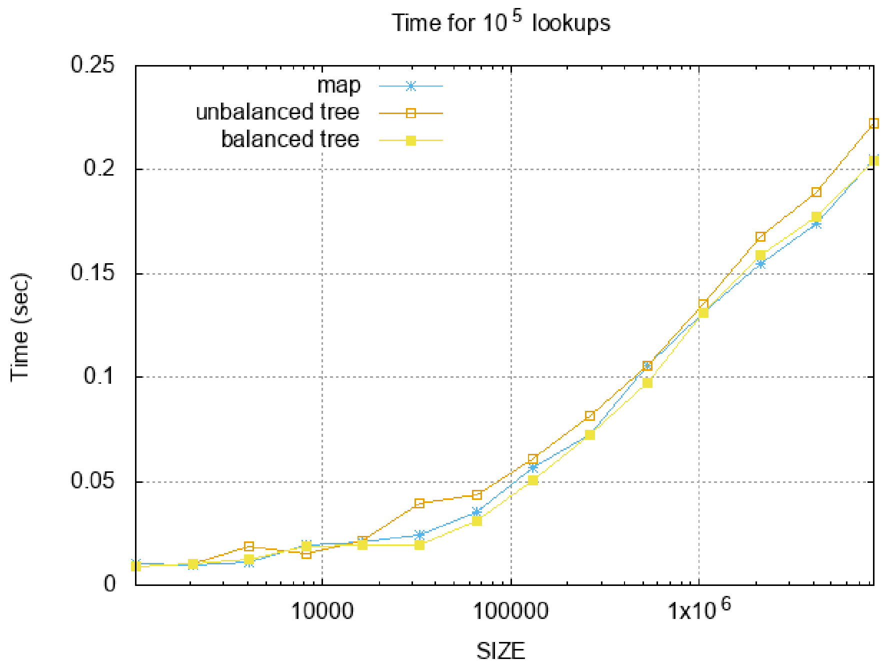
In these report we collect a few remarks about the BST exercise and show some numerical results.

The *node* class is defined in such a way that for each node pointers to previous and next nodes are stored; this choice could be memory inefficient but gave us in turn more freedom in defining a simple version for the *balance* function. This latter function *bisects* the tree recursively, once all the left/right unique pointers of a given initial configuration of the tree are released: having explicit knowledge of next/previous pointers is useful in linearizing the tree.

The balance operation implemented this way is expected to be $O(N)$: we measured execution times and confirmed these expectations. In the following picture we show the time to balance and the insertion time (and the std :: map insertion time, as a comparison) as a function of the number of nodes:



In a similar fashion we compared (10^5 lookups) the performance of the *find* function on a randomly filled tree, a balnced one ad the std :: map. The results show that, as expected, the find algorithm is $log(N)$.



The balanced tree performance match the std :: map, as expected being the latter a n implemetation of a redblack tree.