Assignment 7 – Selected Model Answers

EXERCISE 1. Implement operators on the class AVL returning the elements in the tree:

(i) Define an operator that exploits a *depth-first search* through the tree and returns the elements in the order of this search.

(ii) Define an operator that exploits a *breadth-first search* through the tree and returns the elements in the order of this search.

(iii) Use one of these operators to implement a function *range* on AVL class, which for arguments x and y returns those elements in the tree that satisfy $x \leq e \leq y$.

EXERCISE 2. Take a text of your choice with a length of around one page and build a dictionary with the words in the text. Use a TRIE data structure.

Then implement a procedure, which in case that a search for a word in the dictionary fails return suggested alternatives:

(i) Return words from the dictionary that extend the given word (not found in the dictionary) by one symbol.

(ii) Return words from the dictionary that are prefixes of the given word (not found in the dictionary) with one or two symbols less.

(iii) Return words from the dictionary that differ from the given word (not found in the dictionary) by exactly one symbol.

EXERCISE 3.

(i) Prove that the total number of comparisons in a search in an (a, b)-tree with n nodes is bounded by $\lceil \log b \rceil (2 + \log_a((n-1)/2))$.

(ii) Assuming $b \leq 2a$ show that the number in (i) is in $O(\log b) + O(\log n)$.

SOLUTION.

(i) The root of an (a, b)-tree has at least two children, and all other non-leaf nodes have at least a children. Thus, in a tree of height h the number of nodes is at most

$$1 + 2\sum_{i=0}^{h-1} a^i = 1 + 2\frac{a^h - 1}{a - 1} \ .$$

This implies

$$\frac{(n-1)(a-1)}{2} + 1 \geq a^h$$

and hence

$$h \le \log_a \left(\frac{(n-1)(a-1)}{2} + 1 \right) \le 1 + \log_a \frac{n-1}{2} + \log_a (a-1) \le 2 + \log_a \frac{n-1}{2} \ .$$

The height determines the length of a search path, and in each node along the path there are at most b elements stored in an ordered way. For binary search in a node we need at most $\lceil \log b \rceil$ comparisons.

Hence the total number of comparisons is bounded by $\lceil \log b \rceil (2 + \log_a ((n-1)/2))$.

(ii) Using the result from (i) the number of comparisons is bounded by

$$\left(\log_a \frac{n-1}{2} + 2 \right) \cdot \log_2 b \ = \ \frac{\log_2 b}{\log_2 a} \log_2 \frac{n-1}{2} + 2 \log_2 b \ .$$

As we assume $b \le 2a$, we get $\log_2 b \le 1 + \log_2 a$, which gives

$$\frac{\log_2 b}{\log_2 a} \log_2 \frac{n-1}{2} + 2 \log_2 b \ = \ \left(\frac{1}{\log_2 a} + 1 \right) \log_2 \frac{n-1}{2} + 2 \log_2 b \ ,$$

which is in $O(\log n) + O(\log b)$ as claimed.