Purely functional palindromic trees Iteration V results

Timur Khazhiev
Innopolis University
t.khazhiev@innopolis.ru

Nikolai Kudasov*
Innopolis University
n.kudasov@innopolis.ru

ABSTRACT

This is the results of BS-3 Project and it contains a general project results so far.

CCS CONCEPTS

• **Theory of computation → Data structures design and analysis**;

KEYWORDS

palindrome, eertree, purely functional, persistent

1 ABSTRACTION FROM ALPHABET

Let's start from example of some alphabet denoted by σ_1

$$\sigma_1 = [a, b, c, d, e, f, g]$$

To represent a symbol from that alphabet we can convert it to number representing its position in that alphabet. Simply, mapping alphabet to natural number. We will get

$$\sigma_{num1} = [0, 1, 2, 3, 4, 5, 6]$$

And for alphabet σ_2

$$\sigma_2 = [A, B, C, D, E, F, G]$$

The result will be the same

$$\sigma_{num2} = [0, 1, 2, 3, 4, 5, 6]$$

In fact that list can be simply generated given the length of alphabet. Lengths of alphabets $\sigma_1 and \sigma_2$ are same, in general the tree will be same structurally. So all data types are parametrized by length of alphabet instead of alphabet. If wanted, results from tree can be easily mapped back, for example, to human-readable palindromes. In implementation data type *Symbol* represents symbol in the alphabet.

2 INFINITE TREE

Data type *Node* corresponds to a palindrome in the alphabet. From any node we can build tree of palindromes and get access to *maxPrefixes/maxSuffixes*. From empty nodes we can derive "whole" infinite tree: we can observe all possible palindromes for any alphabet.

3 STRING REPRESENTATION BASED ON INFINITE TREE

Data type *EERTREE* represents palindromic tree for some symbol sequence. In fact it is directLink version of original Rubinchik's eertree [1]. To represent some palindrome in that sequence we store node corresponding that palindrome in infinite tree. Basic operations contain addition a symbol and removing symbol from top.

4 DOUBLE ENDED TREE

Data type *Mergeable* represents palindromic tree for some symbol sequence but with information for both suffixes and prefixes to allow merge operation. In fact it is represented as two *EERTREEs*, one for prefixes and suffixes. Basic operations contain addition a symbol and removing symbol from top for any side of tree. Also merging of two trees is possible.

REFERENCES

[1] Mikhail Rubinchik and Arseny M. Shur. 2018. EERTREE: An efficient data structure for processing palindromes in strings. *European Journal of Combinatorics* 68 (2018), 249 – 265. https://doi.org/10.1016/j.ejc.2017.07.021 Combinatorial Algorithms, Dedicated to the Memory of Mirka Miller.

*Project supervisor.