Purely functional palindromic trees
Iteration VI 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 DOCUMENTATION AND COMMENTS

Most of the functions have a simple comment description, not heavily technical. Everything can be collected and formed into nice-looking documentation using *Haddoc*.

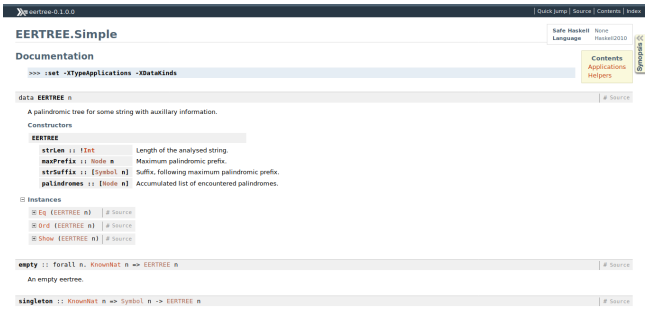

Figure 1: Example of some entry generated by haddoc

2 TESTS AND EXAMPLES

Some functions have an example which also can be used as a *doctest*.

```
updatePalindromes :: (b -> Int) -> (b -> a) -> Int -> [a] -> [b] -> [a]
```

Update list with given function to determine position at which it should be replaced

```
>>> updatePalindromes id id 10 [0,0,0,0,0,0,0,0,0,0] [5,3]
[0,0,0,0,0,5,0,3,0,0]
```

Figure 2: Example of some entry with doctest

More complex functions are covered with *QuickCheck's properties*.

```
merge :: KnownNat n => M n -> M n -> M n
```

Merge two formed eertrees into one

```
merge @2 (eertreeMod xs) (eertreeMod xy) == eertreeMod (xs ++ xy)
```

Figure 3: Example of some entry with property test

In this example it can be seen as:

$$merge(eertree(S_1), eertree(S_2)) \equiv eertree(S_1 + S_2)$$

*Project supervisor.