02257 Applied functional programming

Michael R. Hansen
DTU COMPUTE
May 30, 2023

Project 1: Functional Pearls: Drawing Trees

The topic of this project is aesthetically pleasant drawings of trees and it is based on the journal article [4]. You should study this article and solve the tasks given in this description.

The project has three core tasks:

1. Design of aesthetic pleasant renderings of arbitrarily branching trees.
2. Formalization, using F#, of properties expressing aesthetic rules. Furthermore, property-based testing should be used to validate that renderings satisfy the properties.
3. Design and implementation of a library for visualizing trees. This part should be based on an existing graphing tool of your own choice.

1 Design of aesthetic pleasant renderings of trees

This part is directly based on the article [4], where a polymorphic type

```
Tree<'a>
```

is defined for arbitrarily branching *trees* having *labels* of type `'a` in the node, together with a function

```
design: Tree<'a> -> Tree<'a*float>
```

that transforms a tree of type `Tree<'a>` into a *positioned tree* of type `Tree<'a*float>`. The label in a node of a positioned tree has the form (v, p), where p is a float that describes the position of the node in the tree. Thus, positioned trees provide abstract representations of renderings of trees.

The programs in the article are expressed in the functional programming language SML [5]. The programming language F# [7, 3] belongs to, what is called, the SML-family of languages.

1. In this first part you should simply make F# versions of the programs presented in [4].

2 Property-Based Testing: Validation of rendering properties

The article presents *aesthetic rules* on Page 527, and on Pages 532 and 533 correctness is discussed.

In the part you shall use `FsCheck` [2] to validate the aesthetic (and correctness) rules using *property-based testing* [1]. Doing so you need to express the properties in the article as Boolean-valued F# functions. By use of `FsCheck` it can then be tested whether a property holds on a sample of randomly generated values.

If `FsCheck` finds a counter example showing that a property does not hold, then an error appears

- in your implementation of the rendering,
- in your property, or
- in both.

The small counter examples produced by `FsCheck` are a big help when finding errors. Notice that errors also can appear in journal articles.

2. Use property-based testing to validate that your implementation satisfies the four aesthetic rules appearing on Page 527 in [1]. See also Pages 532 and 533.

3 Visualization of trees

In this part you shall produce a collection of functions that facilitate *visualization* of positioned trees. In doing so, you may use a suitable library/tool of your own choice, with the exception that you should *not* base the visualization on a translation to PostScript.

One option is to use `Plotly.Net` [6]. For further inspiration look at `https://fsharp.org/`, in particular, `https://fsharp.org/guides/data-science/`, for example. You may also consider using Latex or ....

2. This part has the following aspects:
   - Selection of a graphing tool.
   - Design and implementation of functions that can be used to visualize arbitrarily branching trees and positioned trees.
   - Presentation of representative applications of your functions for visualization. (Strengths as well as weaknesses of your solution should be illustrated.)

In this part there are many considerations, choices, design decisions, etc. you need to address. Concerning the functionality, you may, for example, consider

- parameters concerning the layout of trees,
- handling of long labels, (for example, wide labels or labels spanning several lines),
- further use of property-based testing,
- scaling of figures,
- efficiency of the functions in the visualization part,
- . . . .

4 General remark

The report should contain deliberations concerning choices in general, design choices in particular, algorithms, systematic tests and so on.

References

[1] K. Claessen and J. Hughes. *QuickCheck: A Lightweight Tool for Random Testing of Haskell Programs.* Proceedings of the International Conference on Functional Programming (ICFP), ACM 2000.

[2] `https://fscheck.github.io/FsCheck/`

[3] M.R. Hansen and H. Rischel. Functional programming using F#, Cambridge University Press, 2013.

[4] A. J. Kennedy. Functional Pearls: Drawing Trees, in *Journal of Functional Programming*, 6:527–534, 1996. Available from `www.microsoft.com/en-us/research/publication/functional-pearl-drawing-trees/`

[5] R. Milner, M. Tofte, and R. Harper. *The Definition of Standard ML.* MIT Press, Cambridge, Mass., 1989.

[6] The homepage for `Plotly.Net`: `https://plotly.net/`

[7] D. Syme, A. Granicz, A. Cisternino. *Expert F# 2.0*, Apress, New York, NY, USA, 2010.