DTU Compute

Lyngby 27-09-2023
Michael R. Hansen

Validating correctness of a Map-coloring program

This exercise is based on the map-coloring example from Pages 85-88 in the textbook. You should use the version presented during the fourth lecture. This version is available as a script file `MapColoring.fsx` in the `Tips,Tricks,Programs` folder on Learn. It contains the type abbreviations:

```
type Map<'c>      = ('c * 'c) list
type Color<'c>    = 'c list
type Coloring<'c> = Color<'c> list
```

and declarations of the following functions:

```
areNb: Map<'c> -> 'c -> 'c -> bool when 'c: equality
canBeExtBy: Map<'c> -> Color<'c> -> 'c -> bool when 'c: equality
extColoring: Map<'c> -> Coloring<'c> -> 'c -> Coloring<'c> when 'c: equality
countries: Map<'c> -> 'c list
colCntrs: Map<'c> -> 'c list -> Coloring<'c> when 'c: equality
colMap: Map<'c> -> Coloring<'c> when 'c: equality
```

The script file also contains declarations of types `Country` and `SmallMap` (= `Map<Country>`). Furthermore, it is shown how `FsCheck` can be used to check whether every country in a small map m is in `countries` m. This property is called `prop1` m.

Your first task is to express the following properties in F# and validate that they hold for small maps m using FsCheck:

- `prop2` m: Every country in `countries` m occurs in m.
- `prop3` m: The countries in `countries` m are all different.
- `prop4` m: Every country that has a color in `colMap` m occurs in m.
- `prop5` m: The coloring `colMap` m does not contain the trivial color, where the trivial color is the empty list [ ].
- `prop6` m: A color *col* in `colMap` m contains no neighbouring countries.
- `prop7` m: Every country occurring in m has a color in `colMap` m.
- `prop8` m: The colors in `colMap` m are mutual disjoint, that is, a country occurring in m has at most one color in `colMap` m.

Notice that

- `prop1`, `prop2` and `prop3` capture that `countries` m is indeed a list containing all countries occurring in m and only those countries. Furthermore this list does not contain repeated elements.
- `prop4` and `prop5` express that a coloring contains no junk, that is, the trivial color or a color of a country that is not in m.
- `prop6`, `prop7` and `prop8` capture fundament properties of problem, that is, neighbouring countries are colored differently and every country in the map is assigned exactly one color.

Inject errors in the program and check (using FsCheck) whether property-based testing will spot them. Examples could be

- Allow that `contries` m may contain repeated elements, by implementing `addElem x ys` by `x::ys`.
- Introduce a typo in the `then`-branch of `extColoring` replacing `cols'` by `cols`.
- Change the base case of `colCntrs` to `[[]]`, using a coloring containing the empty color instead of the empty coloring.
- .....