Technical Report
COMP1100 Assignment 2

Jacob Bos
ANU u7469354

April 26, 2022

Lab: Tuesday 11am

Tutor: Abhaas Goyal

Word-count beyond cover page at $\leq$ 1250 words

Contents

Introduction

The program detailed in this report is an implementation of Erik Fransson's *QR World* cellular automata with a graphical representation in Haskell. The automata is contained within a module called `Automata` with user input handling in module `App` and graphical output handled in `GridRenderer`. Testing is handled by three modules with unit tests within `AutomataTest`.

1 Documentation

1.1 Design and Technical Decisions

Task 1 consisted of 5 functions. Firstly it was chosen to define the type of `QRCell` as either `Dead` or `Alive` as these are more descriptive of the program's meaning than just boolean values. Function `toQR` uses an if then else (ITE) statement to convert values in the textual representation to useful values with `'A'` to `Alive::QRCell` and any other character as `Dead`. An ITE statement was chosen as computationally we only care about if the value is `'A'` or not. `cycleQR` swaps the value of a cell upon cursor clicks. A case statement was chosen due to having greater readability than ITE statement as there was only two cases. If `QRCell` was `Bool` then the function could just be `not`. `renderQR` used a piecewise case definition to render each cell as the specified codeWorld picture. A piecewise definition was used for improved style. `get` retrieves the value of the model at a given `GridCoord`. It is guarded to return `Nothing` for nonsensical arguments. Elsewhere it just retrieves the appropriate element of the model list. `allCoords` generates a row-major list of all grid coordinates in an $a \times b$ for $a, b > 0$ grid. It returns an empty list for nonsensical arguments of $a, b \not> 0$ for enhanced error tolerance. Otherwise it calls 3 helper functions. `nList` generates an ascending list from 0 to (a-1). `nPair` then pairs each value in the `nList` with some integer. `allPairs` then does this to create one list from (0,0) to (a-1,b-1). The problem was broken up this way out of ease of understanding.

Task 2