| | |
|---|---|
| **15-354: CDM** | **K. Sutner** |
| Assignment 6 | Due: **Oct. 31, 2024, 24:00**. |

## 1. Single Functions (30)

### Background

Automatic structures have only relations, no functions. For this problem we consider the opposite scenario: a structure with only functions, but no relations other than equality. For simplicity, we only allow one single, unary function symbol $f$.

### Task

A. Suppose we have only a single axiom of the form $\gamma_n \equiv f^n(x) = x$, $n \geq 0$. Find all the models of $\gamma_n$ up to isomorphism.

B. There is a natural way to define the product $M = M_1 \times M_2$ of two models $M_i$ of $\gamma_n$. Show that $M$ is again a model of $\gamma_n$. What do these products look like?

C. Now consider an arbitray equation $\gamma \equiv s = t$ where $s$ and $t$ are terms in our language. Again, determine all the models of $\gamma$.

D. Wurzelbrunft realizes that one could produce many more equational axioms if we had more unary functions in our vocabulary. In particular, he believes that with about a billion function symbols he can write down an exceedingly clever collection of such axioms that has only infinite models. What do you say?

**Comment** For part (C), be careful with the variables in the equation.

## 2. Reversibility of ECA (30)

**Background**

Suppose $\rho : \mathbf{2}^3 \to \mathbf{2}$ is the local map of an elementary cellular automaton (i.e., a ternary Boolean function). We have seen how to construct from $\rho$ a synchronous transducer $\mathcal{A}_{\twoheadrightarrow}$ that checks whether a finite bit sequence $x$ evolves to $y$ in one step under fixed boundary conditions. Naturally, there is a similar machine for cyclic boundary conditions, though things are a bit messier than in the fixed case.

Reversibility of a cellular automaton is expressed by the first-order formula

$$\mathsf{inj} \equiv \forall\, x, y, z\, (x \twoheadrightarrow z \wedge y \twoheadrightarrow z \Rightarrow x = y)$$

It is slightly easier to work with irreversibility, expressed by the negation $\mathsf{ninj} = \neg\mathsf{inj}$.

**Task**

A. Construct a synchronous 2-track transducer $\mathcal{B}_{\twoheadrightarrow}$ that checks whether a finite bit sequence $x$ evolves to $y$ in one step under cyclic boundary conditions.

B. Then build a synchronous 3-track transducer $\mathcal{A}$ that accepts the language defined by the matrix of $\mathsf{ninj}$.

C. What does $\mathcal{A}$ have to do with injectivity of the global map on $\mathbf{2}^n$?

D. Explain how one can directly construct a 2-track transducer $\mathcal{A}'$ that still can be used to check $\mathsf{ninj}$. This machine should be of the form $\mathcal{A}' = \mathcal{A}_0 \times \mathcal{U}$ where $\mathcal{U}$ is the un-equal transducer on 2 tracks.

**Comment**

For part (A), nondeterminism is critical (see the construction for the fixed boundary condition case). To avoid a silly edge case, let's assume that all words are non-empty.

## 3. Arithmetic Transducers (40)

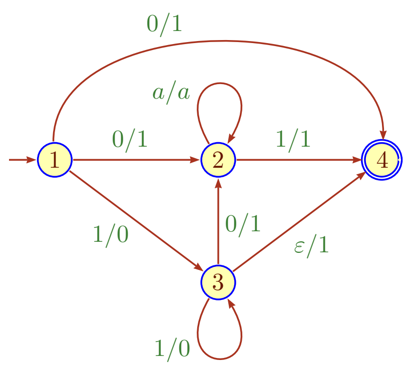
**Background**

Applying transducers to arithmetic requires a bit of care when it comes to representing numbers as strings. In the following we will use the numeration based on reverse binary, no empty string, (zero is represented by 0), no trailing zeros. Thus, numbers are represented by the regular language $\mathcal{N} = \{0\} \cup \{0,1\}^\star 1$ and the value of a string $x = x_0 x_1 \ldots x_{n-1} \in \mathcal{N}$ is $\mathsf{val}(x) = \sum_{i<n} x_i 2^i$; from our definitions, val is a bijection.

Now suppose we have some transducer $\mathcal{T}$ defining a transduction $\tau \subseteq \mathcal{N} \times \mathcal{N}$. We say that $\mathcal{T}$ with behavior $\tau$ implements an arithmetic function $f : \mathbb{N} \to \mathbb{N}$ if

$$f = \{ (\mathsf{val}(x), \mathsf{val}(y) \mid x{:}y \in \tau \}$$

For example, the following transducer implements the successor function:




**Task**

A. Construct a transducer that implements the function $n \mapsto n + 2$.

B. Construct a transducer that implements the function $n \mapsto n + 3$.

C. Construct a transducer that implements the Collatz function $n \mapsto n/2$ for $n$ even, $n \mapsto 3n + 1$ otherwise.

D. Argue that your transducers work as advertised.

**Comment**

1. Just to be clear: there is no need for synchronous transducers, anything goes.

2. "Construct" here means: build it explicitly, don't just argue that it must exist. The machines only require 5/7/7 states (assuming I did not screw up somewhere). It may help to assume initially that $\varepsilon$ for zero and trailing zeros are allowed, and then refine the machines so that they handle the additional constraints.

3. The Collatz function is defined only for positive arguments.

4. The correctness argument should be compelling, but a real formal proof is too much of a mess to consider. Say something reasonable.

**Extra Credit:** Find all the errors in the solutions.