MP1

Natural Language

Bruno Dias 86392
Manuel Ravasqueira 102313
Group 52

October 2021

Solution's description

For the **first** exercise (a) to h)), we used logic to create the transducers. The transducers:

mm2mmm has a first state with two transitions according to the first symbol read (1 or 0) for each option it writes on the output epsilon and then reads the second symbol that for each case(1 or 0) is the next number that together with the first number make a month number (1-12). So for each transition we use the month text symbol.

d2dd has two possible paths: a transition from a state 0 to 1 by any number, then another transition from 2 to 3 by any number except epsilon finishing in state 3 where there is a loop; a transition from a state 0 to 2 where there is an epsilon:0 transition where a 0 is added and then transition by any number to a terminal state . If a number has only one symbol it will try the first path and then since there is no epsilon transition on 2 to 3 the computation goes through the second path adding a 0. If a number has more than one symbol it follows the first.

d2dddd works in a similar way to d2dd. It has also a path where are four digits are read (one per transition) but each read without epsilon transitions so if a number does not have four digits the computation fails and follows other branch. We have then , like **d2dd**, transition eps:0 where a 0 to a state "x" is added and then the computation of three digits where, again, there is no epsilon so if the number does not have three digits the computation fails and there is a transition from "x" to a state where there are two transitions without epsilon so if a number does not have do digits the idea repeats until one symbol.

For the **second** exercise (i) to m)), we used **fst tools** to create the following transducers:

A2R : fstinvert operation performed to the R2A transducer.

birthR2A : This transducer is composed of 2 auxiliary ones. The first being responsible for the conversion from roman numbers to integers, a **fstconcat** operation using both **R2A.fst** and **copy.fst**. the latter converting those integers to the adequate arabic calendar format, a **fstconcat** operation that includes transducers **copy.fst**, **d2dd.fst** and **d2dddd.fst**

birthA2T : Recurring to the **fstconcat** operation, and the transducers **copy.fst** and **mm2mmm.fst**.

birthT2R : Composed by **birthT2A.fst** and **birthA2R.fst**. **birthT2A.fst** is created by using **fstinvert** on **birthA2T.fst**. **birthA2R.fst** is created by using the same method on **birthR2A.fst**.

birthR2L : Composed by two steps, the first being **birthR2A.fst** followed by a **date2year.fst**. Then we compose with **leap.fst** and we get the final result.

Element's contribution

The overall solution design and thought process was done in group and the coding was split with Bruno being responsible for the first module except for the d2dd's transducers and Manuel being responsible for the second module except the birthT2R transducer. Since the solution was designed together, the efforts were equally distributed between the group members,**(50/50)** ,and we helped each other solving bugs while testing.