Exercise 2.1 (Continuity)

The Zip function is not continuous:
\[ \text{Zip}^{-1}((a,b)^*) \cap a^* \# b^* = a^n \# b^n \]
not regular \( \leq \) regular + most regular

Exercise 2.2 (Flip-Flop)

Let \( M = (Q, \delta, \Sigma, q_0) \) be a flip-flop machine. Assume without loss of generality that \( Q = \{0,1\}^k \) define the machine \( M_i := (Q_i, \delta_i, \Sigma, q_i) \) as follows for \( 1 \leq i \leq k \):
- \( Q_i := \{0,1\} \)
- \( A_i(q, a) := \delta_i \)
- \( \delta_i(q, a) := q_0, \text{ if } \delta_a \text{ is the identity map} \)
- \( q_i = \pi_i(q_0) \)

Now the parallel composition
\[ M_1 \parallel M_2 \parallel \cdots \parallel M_k : \Sigma^* \to (\Sigma \times \{0,1\}^k)^* \]
can be obtained by composing the machines sequentially with suitable projections.

The homomorphism:
\[ \Lambda : \Sigma^* \times Q \to \Sigma \]
allows to obtain
\[ \forall w \in \Sigma^*, \quad M(w) = \Lambda_0(M_1, \ldots, M_k)(w) \]
Let us prove that this bound is tight.

Let \( R = (\Sigma, \delta, a, q_0) \) where \( a \in \Sigma \)

\[ \delta_a(q) = a \quad \text{and} \quad q_0 = a \]

\[ \lambda_a(q) = q \]

1. \( R \) has \( |\Sigma| \) states.
2. \( R \) cannot be realised by less than \( |\Sigma| \) states.
3. The composition of \( k \) binary flip flops is computable by a \( 2^k \) states flip flop.

4. Is obvious.

Assume by contradiction that there exists \( M' = (Q', \delta', a', q_{0'}) \) realising \( R \) with \( |Q'| < |\Sigma| \).

Then: \( \lambda_a: Q' \to \Sigma \) is not surjective.

Let \( b \in \Sigma \setminus \lambda_a(Q') \)

But \( M'(ba) = \lambda'_b(q_0) \cdot \lambda'(\delta'(q_0, b), a) \)

\[ R(ba) = ab \]

implies \( b \in \lambda_a'(Q') \) which is absurd.
Let $M_1, \ldots, M_k$ be binary flip flop machines with states $Q_i$, transitions $\delta_i$, productions $A_i$ and initial state $q_0$.

Note that

- $M_1: \Sigma^* \rightarrow \Sigma_i$
- $M_2: \Sigma_i^* \rightarrow \Sigma_2$
- $M_k: (\Sigma^{k-1})^* \rightarrow \Gamma^*$

Let us define $M = (Q, A, \delta, q_0)$ via

- $Q = Q_1 \times \cdots \times Q_k$ of size $2^k$
- $A: \Sigma \times Q \rightarrow \Gamma$
- $(a_i, (q_1, \ldots, q_k)) \mapsto A_k(A_{k-1}(\ldots, A_1(a_i, q_1), q_k), q_k)$
- $\delta: \Sigma \times Q \rightarrow Q$
- $(x, (q_1, \ldots, q_k)) \mapsto (\delta_1(x, q_1), \delta_2(A_1(x, q_1), q_2), \ldots)$

We claim that

$\forall \omega \in \Sigma^* \quad M(\omega) = M_k \circ \ldots \circ M_1(\omega)$

and leave the proof as an exercise.

With 1 + 2 + 3 we conclude that $\mathcal{R}$ cannot be obtained by less than $\log_2 k$ binary flip flop machines.

CQFD
Exercise 2.3 (Decide injectivity).

Let $f$ be a rational function from $\Sigma^*$ to $T^*$.
We construct the language $\mathcal{L}_f$ over the alphabet

$$\Delta := \Gamma \cup \Sigma \times \{(1, 1), (2, 2)\}$$

Words in $\mathcal{L}_f$ are of the form

$$\omega \ (\omega \omega \omega \ (\omega \ \omega) \ \omega) \ \omega$$

such that

1) the word $\omega$ is well-bracketed for $(1, 1)_1$ and (independently) well-bracketed for $(2, 2)_2$ without meeting $\omega$.

2) the $(1, 1)_1$ parentheses enclose factors of $\omega$ that are produced by $f$ on the new term on top of the $(1, 1)_1$ parentheses.

3) similar to 2) but for $(2, 2)_2$ parentheses.

Conditions 1), 2), and 3) are regular.

1) because there is no nesting.

2) because we transform the NFA with output for $f$ into an NFA that ignores $(2, 2)_2$ letters and replaces

$$a/u \ \text{by} \ \emptyset$$

3) similar to 2).

\[ q \xrightarrow{a} q' \]

\[ q \xrightarrow{a} \text{read } i \xrightarrow{a} q' \]
Now, if \( \{ \text{the } i\text{ word and the } j\text{ word} \} \neq \emptyset \) are different

it is injective.

Checking if the two words are equal is not doable by a CFG!

But we can update our definition so that one position

of each top word is selected/distinguished.

and use a pushdown automaton to validate that

the two selected letters are distinct and appear at the

same moment in both words.

The construction of the automata is done in polynomial time

and deciding equivalence is done in polynomial time too.

Hence deciding injectivity of rational functions is in \( \text{PTIME} \).

**Exercise 2.4 (Windowed Transducers)**

1. Let \( M \) be a Mealy machine. There exists a computable \( K \) such

   that \( M \) is \( K \)-windowed \( \iff \) \( M \) is windowed.

2. Let \( M \) be a Mealy machine and \( K \in \mathbb{N} \). One can build

   a Mealy machine \( W(M,K) \) that is \( K \)-windowed

   and such that

   \[ W(M,K) \equiv M \quad \text{if} \quad M \text{ is } K \text{-windowed} \]

Using (1) and (2) we can decide if a Mealy machine is windowed

by computing \( K \) (1) and checking the equivalence between

\( M \) and \( W(M,K) \).

Let us now prove (1) and (2).
Proof of 1. \( \mathcal{N} = (Q, \delta, \alpha, q_0) \). We want to prove

(M, \ast) for the monoid generated by \((\delta_a : Q \to Q)_{a \in \Sigma}\), where \( \Sigma \) is finite.

- We write \( \delta_w \) for the function \( q \mapsto \delta^*(q, w) \).
- Recall that \( \delta_w \) is idempotent if \( \delta_w \circ \delta_w = \delta_w \).

**Claim**: there exists a computable \( n_0 \) such that for all words \( w \), \( \ell(w) > n_0 \)

we can write \( w = w_1 \cdot w_2 \cdot w_3 \) with \( \delta_{w_2} \) idempotent and \( \ell(w_2) < n_0 \).

Proof sketch: use Ramsey \( \diamond \).

Let us assume that \( \mathcal{N} \) is \( K \)-windowed for some \( K \in \mathbb{N} \).

By definition \( K = \mathbb{N} \).

For all \( a \in \Sigma \)

(\*) \( K \) is windowed.

Now let us prove that \( \mathcal{N} \) is \( n_0 \)-windowed.

Let \( a \in \Sigma \)

and \( \ell(a) > n_0 \).

We have \( w_1 \cdot w_2 \cdot w_3 \) with \( \ell(w_1) < n_0 \) and \( \delta_{w_2} \) idempotent.

In particular, \( \forall m \in \mathbb{N} \),

\[
\delta^*(q_0, w_1 \cdot w_2 \cdot w_3) = \delta_3 \delta_2 \delta_1 \delta_0 (q_0) \quad \text{[and similarly for } w_2] \).

But using (\*) with a large enough \( n \in \mathbb{N} \) so that \( n \ell(w_2) > K \),

\[

\begin{align*}
A(\delta^*(q_0, w_1 \cdot w_2 \cdot w_3), a) &= A(\delta^*(q_0, w_1 \cdot w_2 \cdot w_3), a) \\
A(\delta^*(q_0, w_1 \cdot w_2 \cdot w_3), a) &= A(\delta^*(q_0, w_1 \cdot w_2 \cdot w_3), a)
\end{align*}

\]

\( \Box \)
Proof:

Let $M = (Q, \Sigma, \delta, q_0)$ and $K \in \mathbb{N}_2$. We write $W(M, K) = (Q', \Sigma', \delta', q_0')$ with

$$Q' := \Sigma^* \cup \{ \epsilon \} \quad q_0' := \epsilon$$

$$\delta'(q, a) := \text{suffix of } \delta(q, a) \text{ of size } \leq K$$

$$\Delta'(q, a) := \Delta(\delta^*(q_0, a), a)$$

We claim that if $M$ is $K$-windowed then $\Delta \equiv W(M, K)$.

We prove this claim by showing

$$\forall \omega \in \Sigma^+, \Delta(\delta^*(q_0, \omega), a) = \Delta'(\delta^*(q_0', \omega), a)$$

$$\forall a \in \Sigma$$

which is done by noticing that if $\omega = \omega_1 \omega_2$, $|\omega_2| = K$

$$\delta^*(q_0', \omega) = \omega_2$$

and

$$\Delta'(\delta^*(q_0', \omega), a) = \Delta(\delta(q_0, \omega_2), a)$$

for words $\omega$ of size less than $K$. A similar argument holds. \(\square\)
Exercise 3.1 (Semantically Functional)

Let \( A = (Q, \delta, F, q_0, A) \) be an NFA with outputs that is functional.
We endow \( Q \) with an arbitrary total ordering \( \leq \).

1. One can build a rational function \( f : \Sigma^* \to (\Sigma \cup Q)^* \) that outputs the lexicographically smallest run of \( A \) on the input word.

2. There is a rational function \( g : (\Sigma \cup Q)^* \to T^* \) taking as input a run
\[ q_0, q_1, b, q_2, \ldots \]
and producing
\[ A(q_0, q_1) A(q_1, b, q_2) \ldots \]
or inputs \( e \) and produces \( A(e) \).

Remark that \( g \circ f = A \) and is a rational function as a composition.

It is clear how to build \( g \) using a DFA with outputs.
Let us focus on \( f \).
We start by using a function \( \text{padd} : aaba \mapsto \#a#a\#b\#a\# \)
that adds hashes around letters and is clearly a rational function.
Then we build an NFA relabeling from \((\Sigma \cup \#)^*\) to \((\Sigma \cup Q)^*\) as follows:

1. \( q_a(x) = a(x) \) for all \( a \in \Sigma \).

2. \( q_q(x) = \exists (x) \in Q : E \)
   1. \( (x) \) codes a valid run of \( A \) on the hashes \#.
   2. \( x \in X_q \)
   3. for all \( (x_p) \in Q : \) valid run of \( A : (x_p) \) is
      lexicographically smaller than \((x_q)\).
Now: 1) is \( \mathbf{MSO} \) definable as in class

2) is \( \mathbf{MSO} \) definable too

3) Can be checked by asserting that either
   - \( \forall i. \bigwedge_{q \in Q} [i \in \chi_q \iff i \in \chi_q'] \) (equality)
   - \( \exists i. \left( \forall j < i, \bigvee_{q \leq q'} j \in \chi_q \land j \in \chi_{q'} \right) \land \bigvee_{q \leq q'} i \in \chi_q \land j \in \chi_{q'} \) (strict inequality)

Please note that turning an NFA into a DFA may require an exponential blowup in the number of states. Hence any "polynomial time solution" to the above problem was not plausible.