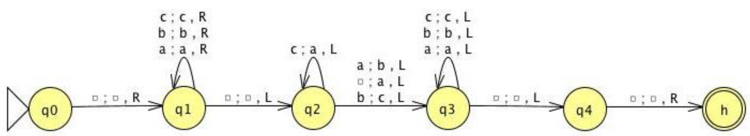
CS3331 – Assignment 3
due Nov. 26, 2019, (latest to submit: Nov. 29, 11:55pm)

1. (10pt) Consider the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ and define the function $succ : \Sigma^* \to \Sigma^*$, $succ(w)$ is the word immediately following w in lexicographic order. Construct a deterministic Turing machine M that computes the function $succ$, that is, M starts with the initial configuration $(s, \square w)$ and halts with the configuration $(h, \square succ(w))$. Describe M in details using a directed graph whose edges are labelled by transitions (such as the one in Example 17.2, p. 268 of textbook).

   Solution:

2. (10pt) Construct a deterministic Turing machine M that adds one to its binary input if it is even and subtracts one if it is odd. M starts with the initial configuration $(s, \square w)$, where $w \in \{\mathtt{0}, \mathtt{1}\}^*$; the binary input w is interpreted as an integer number. Possible leading $\mathtt{0}$'s have to be removed as well. The machine halts in the appropriate configuration $(h, \square(w \pm 1)_{(2)})$, where $w_{(2)}$ is the binary representation of w.

   Here are some examples of M's behaviour:

   $(s, \square) \vdash^* (h, \square\mathtt{1})$

   $(s, \square\mathtt{000}) \vdash^* (h, \square\mathtt{1})$

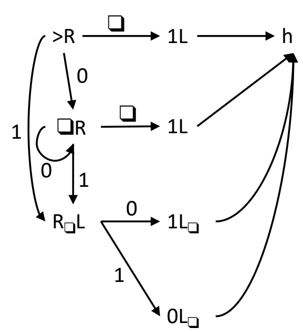
   $(s, \square\mathtt{01}) \vdash^* (h, \square\mathtt{0})$

   $(s, \square\mathtt{111}) \vdash^* (h, \square\mathtt{110})$

   $(s, \square\mathtt{001100}) \vdash^* (h, \square\mathtt{1101})$

   Describe M using the macro language (such as the one in Example 17.8, p. 275 of textbook).

   Solution:

3. (20pt) Construct a Turing Machine M that semidecides, but does *not* decide, each of the following languages over the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$:

   (a) $L_1 = \{\mathtt{a}\}$,

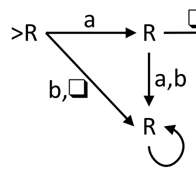
   (b) $L_2 = \Sigma^*$,

   (c) $L_3 = \emptyset$.

In each case, describe M using the macro language.

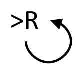
Solution:

(a) The machine accepts a and loops for any other input:



(b) This is impossible. A machine semideciding Σ^* would accept everything, thus deciding Σ^*.

(c) The machine always loops, accepting nothing:



4. (20pt) Describe in clear English a Turing machine that semidecides the language

$$L = \{< M >| \, M \text{ accepts the binary encodings of at least 3 prime numbers}\} \, .$$

Solution:

TM that semidecides L:

1. Generate all binary encodings of natural numbers in lexicographical order.
2. For each number, check if it is prime and keep the primes only.
3. On input $<M>$, run M on all primes in dovetailing mode.
4. If three computations accept, then accept.

5. (20pt) Is the set SD closed under:

(a) Intersection?

(b) Concatenation?

Prove your answers. Clear English description of any Turing machines is sufficient. (That is, you don't have to effectively build the machine, instead explain how the machine behaves.)

Solution:

(a) SD is closed under intersection. To prove this, assume $L_1, L_2 \in SD$ are arbitrary semidecidable languages and assume they are semidecided by M_1 and M_2, resp. We construct $M_\cap$ that semidecides $L_1 \cap L_2$ as follows:

On input w
Run M_1 on w
  If M_1 rejects, then reject
  If M_1 accepts, then
    Run M_2 on w
      If M_2 rejects, then reject
      If M_2 accepts, then accept

Another solution for $M_\cap$:

On input w

Run in parallel M_1 on w and M_2 on w
If both accept, then accept

(b) SD is closed under concatenation. To prove this, assume $L_1, L_2 \in SD$ are arbitrary semidecidable languages and assume they are semidecided by M_1 and M_2, resp. We construct M_c that semidecides L_1L_2 as follows:

On input w
Run in parallel, in dovetailing mode:
For each $w = w_1w_2, w_1, w_2 \in \Sigma^*$
Run M_1 on w_1 and M_2 on w_2
If both accepts, then accept

Another solution for M_c:

On input w
Nondeterministically guess a factorization $w = w_1w_2, w_1, w_2 \in \Sigma^*$
Run in parallel M_1 on w_1 and M_2 on w_2
If both accepts, then accept

6. (20pt) Let L_1 and L_2 be two languages that are not decidable.

(a) Is it possible that $L_1 - L_2$ is regular and $L_1 - L_2 \neq \emptyset$? Prove your answer.

(b) Is it possible that $L_1 \cup L_2$ is decidable but $L_1 \neq \neg L_2$? Prove your answer.

Solution:

(a) Yes. Consider TM M_L that always loops. Then $<M_L> \notin H_{\text{ANY}}$. Choose $L_1 = H_{\text{ANY}} \cup \{<M_L>\}$, $L_2 = H_{\text{ANY}}$. Since H_{ANY} is not decidable, L_1 and L_2 are also not decidable. But $L_1 - L_2 = \{<M_L>\}$ is finite, hence regular; and nonempty.

(b) Yes. Choose $L_1 = H_{\text{ANY}} \cup \{<M_L>\}$, $L_2 = \neg H_{\text{ANY}}$. Then $L_1 \cup L_2 = \{<M> \mid M \text{ is a TM}\}$ is decidable but $L_1 \neq \neg L_2$.

Note Submit your solution as a (typed) pdf file on `owl.uwo.ca`.