CS3331 - Assignment 3 - 2018

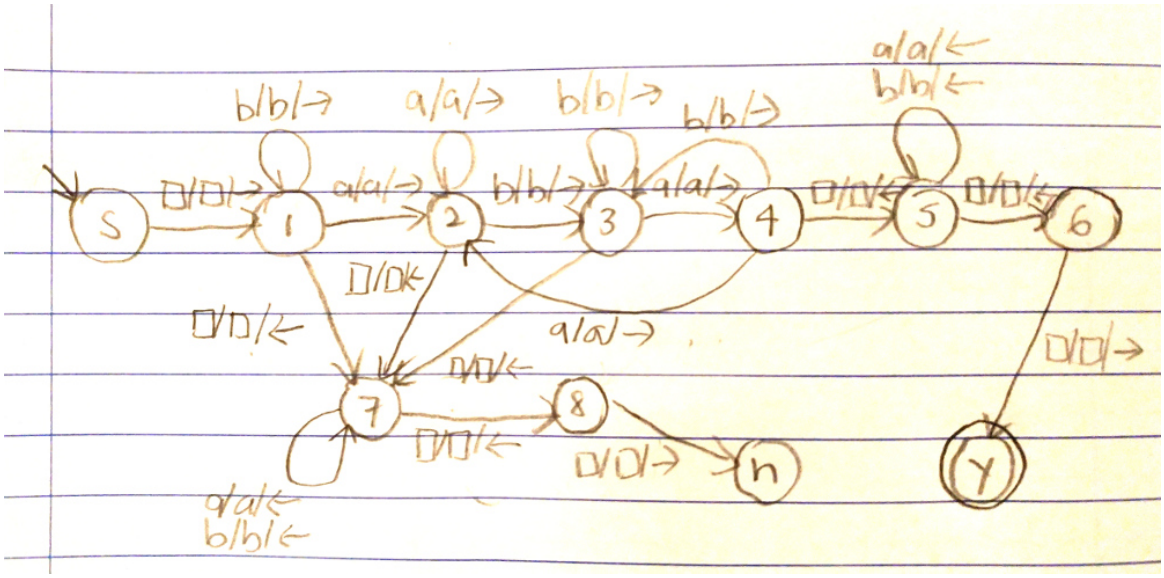
Decidable and Semi-Decidable Languages I

Due: Tuesday, Nov 27, 2018 (Latest to submit: Friday, Nov 30)

1. (20pt) Construct a deterministic Turing machine M that decides the language

$$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid w \text{ has } \mathtt{ab} \text{ as a substring and ends with } \mathtt{ba}\}.$$

M starts with the initial configuration $(s, \sqcup w)$ and halts with the configuration $(q, \sqcup w)$, for the appropriate $q \in \{y, n\}$. Describe M in detail using a directed graph whose edges are labelled by transitions (such as the one in Example 17.2, p. 368 of textbook).



2. (20pt) Construct a deterministic Turing machine M that subtracts one from its binary input if it is positive and sets it to zero if its input is zero. This machine computes the function

$$f(n) = \begin{cases} 0 & \text{if } n \text{ is } 0 \\ n-1 & \text{otherwise} \end{cases}$$

M starts with the initial configuration $(s, \sqcup w)$, where $w \in \{\mathtt{0}, \mathtt{1}\}^*$; the binary input w is interpreted as an integer number. The machine must remove all leading zeros from the input and halt in the appropriate configuration $(h, \sqcup(w-\mathtt{1})_{(2)})$ or $(h, \sqcup\mathtt{0})$, where $w_{(2)}$ is the binary representation of w. If $w = \varepsilon$, treat it as a representation of $\mathtt{0}$. Here are some examples of M s behaviour:

$(s, \sqcup) \mid\!-^* (h, \sqcup\mathtt{0})$

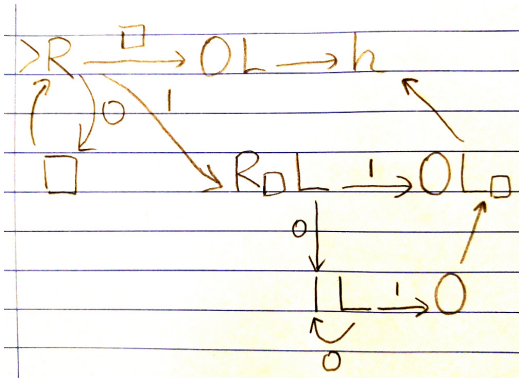
$(s, \sqcup\mathtt{000}) \mid\!-^* (h, \sqcup\mathtt{0})$

$(s, \sqcup\mathtt{01}) \mid\!-^* (h, \sqcup\mathtt{0})$

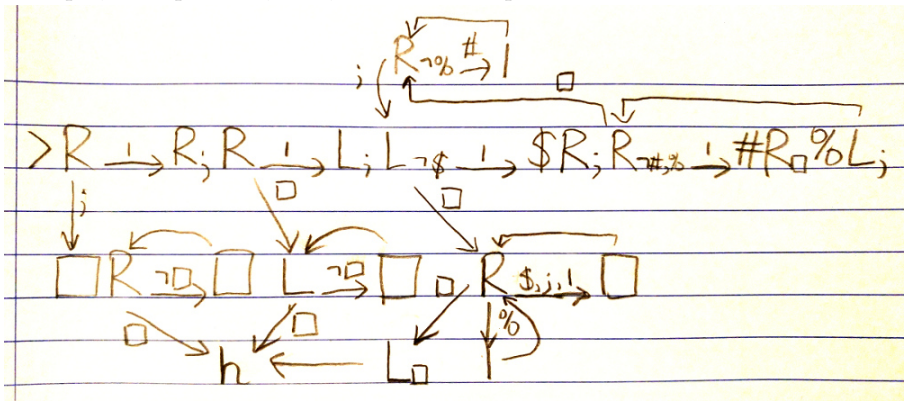
$(s, \sqcup\mathtt{111}) \mid\!-^* (h, \sqcup\mathtt{110})$

$(s, \sqcup\mathtt{001100}) \mid\!-^* (h, \sqcup\mathtt{1011})$

Describe M using the macro language (such as the one in Example 17.8, p. 377 of textbook).



3. (20pt) Construct a deterministic Turing machine M that multiplies two unary numbers. Specifically, given the input string $< x >;< y >$, where $< x >$ is the unary encoding of a natural number x and $< y >$ is the unary encoding of a natural number y, M should output $< z >$, the unary encoding of $z = xy$. For example, on input `111;1111`, M should output `111111111111`. Describe M using the macro language



4. (20pt) Consider the language $L = \{< M > \mid M \text{ accepts at least two strings}\}$.

   (a) Describe in clear English a Turing machine M that semidecides L.

   M generates the strings in Σ_M^* in lexicographic order and uses dovetailing to interleave the computation of M on those strings. As soon as two computations accept, M halts and accepts.

   (b) Suppose we changed the definition of L just a bit. We now consider:

   $$L' = \{< M > \mid M \text{ accepts } \textit{exactly } 2 \text{ strings}\}.$$

   Can you tweak the Turing machine you described in part (a) to semidecide L'?

   No. M could discover that two strings are accepted. But it will never know that there are not any more.

5. (20pt) Describe in clear English a Turing machine that semidecides the language

   $$L = \{< M > \mid M \text{ accepts the binary encodings of the first four Fibonacci numbers}\}.$$

The meaning of "the first four Fibonacci numbers" could be read in different ways. Taking this to mean 0, 1, 1, 2, or 1, 1, 2, 3, though redundant, would still be acceptable. Otherwise, given that we are considering set membership, this could be taken to mean, 0, 1, 2, 3, or 1, 2, 3, 5, depending on the starting convention for the Fibonacci sequence. Taking this last case as the interpretation, a solution is as follows:

We define a semideciding Turing machine $M'(< M >)$ that operates as follows:

1. Run M on `1`. If M rejects, loop.
2. Run M on `10`. If M rejects, loop.
3. Run M on `11`. If M rejects, loop.
4. Run M on `101`. If M accepts, accept. Otherwise loop.

This procedure will halt and accept iff M accepts the binary encodings of the first four Fibonacci numbers. If, on any of those inputs, M either fails to halt or halts and rejects, this procedure will fail to halt.