CPSC 351 — Tutorial Exercise #6
Additional Practice Problems

About These Problems

These problems will not be discussed during the tutorial, and solutions for these problems will not be made available. They can be used as "practice" problems that can help you practice skills considered in the lecture presentation for Lecture #6, or in Tutorial Exercise #6.

Note: Information in the supplemental document, "Turing Machines That Compute Functions", for Lecture #6, will be useful as you try to solve the following problems.

Practice Problems

Suppose that the alphabet Σ_{binary}, and unpadded binary representations of non-negative integers, are defined as at the beginning of Section 4 of the supplemental document on "Turing Machines That Compute Functions" (see, in particular, the equations at lines (2) and (3) of that document.

Let

$$f_{\text{b\_dec}} : \Sigma^{\star}_{\text{binary}} \rightarrow \Sigma^{\star}_{\text{binary}}$$

which is defined as follows:

- If ω is the unpadded binary representation of a *positive* integer n then $f_{\text{b\_dec}}(\omega)$ is the unpadded binary representation of $n - 1$.
- If either $\omega = 0$ or ω is not the unpadded binary representation of any non-negative integer, at all, then $f_{\text{b\_dec}}(\omega) = \lambda$.

Suppose we wish to design a Turing machine that computes functions

$$M = (Q, \Sigma_{\text{binary}}, \Sigma_{\text{binary}}, \Gamma, \delta, q_0, q_{\text{halt}})$$

that computes the above function $f_{\text{b\_dec}}$.

1. To continue one should note that — for this computation — the function's value is λ whenever ω *does not* begin with "1".

   On the other hand, if ω begins with "1" then ω is the unpadded binary representation of some positive integer n.

   - If $n = 1$ then $\omega = 1$ and $f_{\text{b\_dec}}(\omega) = 0$, the unpadded binary representation of zero.
   - If $n = 2^h$, for some positive integer h, then $\omega = 10^h$ and $f_{\text{b\_dec}}(\omega) = 1^h$, the unpadded binary representation of $2^h - 1$.
   - Otherwise ω also includes at least two copies of "1", so that

     $$\omega = 1\mu 10^h$$

     for some string $\mu \in \Sigma^\star_{\text{binary}}$ and some non-negative integer h. In this case

     $$f_{\text{b\_dec}}(\omega) = 1\mu 01^h.$$

   Using this information, please write an algorithm — that might resemble the one shown in Figure 1 in the document "Turing Machines That Compute Functions" — that computes the function $f_{\text{b\_dec}}$

2. Prove that your algorithm is correct: That is, prove that its execution on any input string $\omega \in \Sigma^\star$ halts, returning the string $f_{\text{b\_dec}}(\omega)$ as output.

3. Give an ***implementation-level description*** of a Turing machine that is based on the algorithm that gave when solving the first problem, along with a description of *how* the implementation-level description is based on your algorithm.

4. Give a ***formal description*** of a Turing machine that is based on the implementation-level description that you gave when solving the previous problem, along with a description of *how* the formal description is based on the implementation-level description.

5. Sketch a proof that your Turing machine computes the function $f_{\text{b\_dec}}$ — by giving a sequence of lemmas (concerning various computations of L), describing how these can be proved, and then giving a proof, that M computes $f_{\text{b\_dec}}$, using them.