Nathan Bickel, Jackson Ginn, Jack Hyatt, Chance Storey
CSCE 355: Section 001
Professor: Dr. Fenner
April 16, 2024

CSCE 355 Homework 6

Problem 1 Give a reduction from the hello-world problem to the problem below. Use the informal style of Section 8.1 of the textbook for describing for describing plausible program transformations, and do not worry about the real limits such as maximum file size or memory size that real computers impose.

Given a problem and an input, does the program eventually halt; i.e. does the program not loop forever on the input?

Given program Q and its input y, we will construct a program R and an input z such that R, with input z, halts if and and only if Q with input y prints `hello, world`:

1. Modify Q to remember the first 12 characters that it prints, storing them in a global array, storing them in a global array A. Let the resulting program be Q_1.
2. Add to Q_1 a repeat loop (or a while loop with condition true). Every second, the loop checks whether A has written 12 characters or more, and if so, whether `hello, world` are the first 12 characters. In that case, exit the program immediately (with an exit() call). The resulting program is R, and input z is the same as y.

Suppose that Q with input y prints `hello, world` as its first output. Then at most a second after, R will halt. However, if Q with input y does not print `hello, world` as its first output, then A will never contain `hello, world` as its first 12 characters, so the repeat loop will never stop and will keep checking A forever, and thus R will never halt. □

Problem 2 In this exercise we explore the equivalence between function computation and language recognition for Turing machines. For simplicity, we shall consider only functions from nonnegative integers to nonnegative integers, but the ideas of this problem apply to any computable functions. Here are the two central definitions:

- Define the graph of a function f to be the set of all strings of the form $[x, f(x)]$, where x is a nonnegative integer in binary, and $f(x)$ is the value of function f with arguument x, also written in binary.
- A Turing machine is said to compute function f if, started with any nonnegative integer x on its tape, in binary, it halts (in any state) with f, in binary, on its tape.

Answer the following, with informal, but clear constructions.

(a) Show how, given a TM that computes f, you can construct a TM that accepts the graph of f as a language.

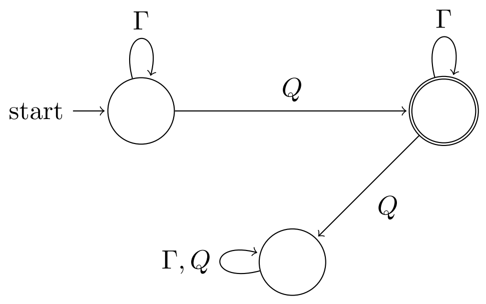
(b) Show how, given a TM that accepts the graph of f, you can construct a TM that computes f.

(a) Since we are given a TM that computes f, we can just use a second track to simulate that TM on any given input $[x, y]$, and check to see if y is the same as $f(x)$ when the TM we simulate on the second track halts. And if it does, then we accept.

(b) We want to simulate the TM, let's call M, that computes $f(x)$ given x, when all we have is the TM that accepts the graph of f.
We cannot try all values of i in $[x, i]$, since M may not halt on a certain value of i. So we must consider if M accepts $[x, i]$ in j step, where $i + j = n$, for increasing values of n. n increases until we find a pairing where $i = f(x)$ and is accepted in j steps. So then all we would need to do is write what i is on the tape that computes $f(x)$ and halt.

Problem 3 Let $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ be a Turing machine. As usual, we assume that $Q \cap \Gamma = \emptyset$. Recall that an *instantaneous description* (ID) of M is any string over the alphabet $Q \cup \Gamma$ containing exactly one symbol from Q. Give a DFA that recognizes the language of all IDs of M.

Shown is our solution, where a transition labeled Γ accepts on any symbol in Γ, and Q accepts on any symbol in Q.



Problem 4 Let M be as in the last problem. Recall that if ID_1 and ID_2 are IDs of M, then $\text{ID}_1 \vdash \text{ID}_2$ means that ID_2 results from ID_1 by a single step of M. Let \$ be some symbol not in $Q \cup \Gamma$. The languages

$$L_1 := \{w\$x^R \mid w \text{ and } x \text{ are IDs of } M \text{ and } w \vdash x\}$$
$$L_2 := \{w^R\$x \mid w \text{ and } x \text{ are IDs of } M \text{ and } w \vdash x\}$$

are both context-free. (Recall that x^R and w^R are the reversals of strings x and w, respectively.) Describe how to build CFGs for L_1 and L_2, given a complete description of M.

Our grammar can be constructed as follows:

L_1 has start symbol S. For each $a \in \Gamma$ and for each $p, q \in Q$, add:
$S \to aSa \mid qTpa\delta(q, a) \mid qaTp\delta(q, a)$
$T \to aTa \mid \$$

L_2 has start symbol S. For each $a \in \Gamma$ and for each $p, q \in Q$, add:
$S \to aSa \mid qT\delta(q, a)ap \mid aqT\delta(q, a)p$
$T \to aTa \mid \$$