Homework #6

Will Holcomb CSC445 - Homework #6

December 4, 2002

1 Number 1: Exercise 8.1.1.b

Give a reduction from the Hello World problem to the problem of "given a program and an input, does the program ever produce any output?"

The Hello World problem is defined as "determine whether a given program with a given input prints "hello, world" as the first 12 characters it prints." The Hello World problem has been proven unsolvable by contradiction.

The Output problem can be mapped to the Hello World problem by writing a program which takes a program as input and if it prints "hello, world" as the first 12 characters then "hello, world" is printed to the output, otherwise nothing is printed.

If a program could tell if this program produced output, then it would be also able to tell if the original program output "hello, world" as the first 12 characters and it is known that such a program cannot exist.

2 Number 2: Exercise 9.1.1.b

What strings are w_{100}?

$$\begin{aligned} w_0 &= \epsilon \\ w_i &= (i-2)_2 \\ w_{100} &= 98_2 = 1100010 \end{aligned}$$

w_{100} has no start state, so $L(w_{100}) = \emptyset$

State	0	1	X	Y	B
$> q_0$	(q_1,X,R)	-	-	(q_3,Y,R)	-
q_1	(q_1,0,R)	(q_2,Y,L)	-	(q_1,Y,R)	-
q_2	(q_2,0,L)	-	(q_0,X,R)	(q_2,Y,L)	-
q_3	-	-	-	(q_3,Y,R)	(q_4,B,R)
q_4*	-	-	-	-	-

Table 1: Turing machine to accept $\{0^n1^n|n \geq 1\}$

Start	Current Character	Next	Write	Move	Encoding
q_0	0	q_1	X	R	010000100100100
q_0	Y	q_3	Y	R	01000100001000100
q_1	0	q_1	0	R	001000010010000100
q_1	1	q_2	Y	L	001000001000100010
q_1	Y	q_1	Y	R	0010001001000100
q_2	0	q_2	0	L	0001000010001000010
q_2	X	q_0	X	R	00010010100100
q_2	Y	q_2	Y	L	00010001000100010
q_3	Y	q_3	Y	R	000010001000010000100
q_3	Y	q_4	B	R	00001000100000010100

Table 2: Encodings of Turing machine in Table 1

3 Number 3: Exercise 9.1.2

Write one of the possible codes for the Turing machine in Table 1:

$$\begin{aligned} E(q_i) &= 0^{i+1} \\ E(B) &= 0 \\ E(X) &= 00 \\ E(Y) &= 000 \\ E(0) &= 0000 \\ E(1) &= 00000 \\ E(L) &= 0 \\ E(R) &= 00 \end{aligned}$$

The encodings of the transitions are in Table 2 and the resultant encoding from appending the transitions is:

$$\begin{aligned}\rho(M) = {} & 0100001001001001101000100001000100011 \\ & 0010000100100001001100100000100010001011 \\ & 0010001001000100110001000010001000010011 \\ & 0001001010010011000100010001000110011 \\ & 0000100010000100010011000010001000010000010100\end{aligned}$$

4 Number 4: Exercise 9.1.4

Show how all possible Turing machines could be encoded given that they draw their tape alphabet from the $\Sigma = \{a_1, a_2, ...\}$

The encoding is fairly straightforward. First define an encoding for the possible states, Q, as a unary number derived from a count of 0's:

$$E(q_i) = 0^{i+1} \tag{1}$$

Next define a similar encoding for the input alphabet:

$$\begin{aligned}\Sigma &= \{a_1, a_2, ...\} && (2)\\ E(a_i) &= 0^i && (3)\end{aligned}$$

And finally define an encoding for each possible movement on the tape:

$$\begin{aligned}D &= \{L, R, S\} && (4)\\ E(L) &= 0 && (5)\\ E(R) &= 00 && (6)\\ E(S) &= 000 && (7)\end{aligned}$$

Each transition in a given Turing machine M can be encoded as:

$$E(q_i)1E(a_c)1E(q_f)1E(a_w)1E(d) \ni \tag{8}$$

$$\begin{aligned}q_i \in Q &\text{ is the initial state} && (9)\\ a_c \in \Sigma &\text{ is the current character on the tape} && (10)\\ q_f \in Q &\text{ is the next state} && (11)\\ a_f \in \Sigma &\text{ is the character to be written to the tape} && (12)\\ d \in D &\text{ is the direction for the machine to move} && (13)\end{aligned}$$

All of the transitions may then be concatenated by separating them with "11" to get an encoding for the entire machine.

5 Number 5: Exercise 9.2.3.b

Informally describe a multi-tape Turing machine that enumerates all of the prime numbers.

a. Tape 1 will hold the first factor.

b. Tape 2 will hold the second factor.

c. Tape 3 will hold the number being tested.

d. Tape 4 will hold a scratch space.

e. Tape 5 will hold the largest number checked.

f. Tape 6 will hold a list of identified primes.

a. All tapes start blank.

b. One 0 is added to tape 5

c. Copy tape 5 to tapes 1, 2 and 3

d. while tape 2 is not blank and an answer has not been found

   (a) copy tape 1 to the holding tape for each 0 on tape 2

   (b) compare the holding tape with the test tape if they are the same (tape 1 * tape 2 = test tape) then clear tapes 1, 2 and 3 and go back to the start

   (c) if tape 1 has characters on it then remove one character and loop

   (d) if tape 1 is blank then if tape 2 has characters on then copy the count tape to tape 1, remove a 0 from tape 2 and loop

   (e) if tape 2 is blank then no factors were found and the testing tape is prime, so copy the testing tape to the output tape, clear tapes 1, 2 and 3 and go to the start

6 Number 6: Exercise 9.2.6

Informally prove or disprove the following closure properties for recursively enumerable languages:

a. Intersection

A recursively enumerable language is one whose membership can be determined by a Turing machine which halts if the string is a member of the language.

The intersection of k languages can be simulated with a Turing machine that simulates each of the test machines and if the string is a member of all of them then it outputs "y" otherwise it outputs "n". If the string is not a member of any of the test languages then the machine may never halt.

b. Concatenation

An empty transition is a transitions which writes the character that was already on the tape and does not move the head. It allows for a state transition without changing the state of the tape.

The concatenation of two Turing machines in general can be accomplished by adding empty transitions from each of the final states in the M_i to the start state of M_{i+1}.

c. Kleene Closure

Adding empty transitions from the final states back to the start state will enable a Turing machine to accept it's own Kleene Star.

7 Number 7: Exercise 9.3.2

The Big Computer Corp. has decided to bolster its sagging market share by manufacturing a high-tech version of the Turing machine, called BWTM, which is a Turing machine with bells and whistles that for each state either blows a whistle of rings a bell. Prove that it is undecidable whether a given BWTM M, on a given input w, ever blows a whistle.

Assume that the output problem has a solution, there is a Turing machine, WC, that takes as input $\rho(M), \rho(w)$, always halts, and outputs "Y" if M whistles on w and "N" if it didn't.

Now write a BWTM, M', that takes a Turing machine, R, and string, x, as input and whistles only if that Turing machine stops. The combination of

these two Turing machines could be used to determine if R halts on x, which is undecidable, so WC cannot exist.