

CS 334 – Fall 2004: Answers for Assignment 10

1. 9.1.1(a)

37 in binary is 100101. Remove the leading 1 to get the string 00101, which is thus w_{37} .

2. 9.1.3a

Suppose this language were accepted by some TM M . We need to find an i such that $M = M_{2^i}$. Fortunately, since all the codes for TM's end in a 0, that is not a problem; we just convert the specification for M to a code in the manner described in the section.

We then ask if w_i is accepted by M_{2^i} ? If so, then w_i is not accepted by M , and therefore not accepted by M_{2^i} , which is the same TM. Similarly, if w_i is not accepted by M_{2^i} , then w_i is accepted by M , and therefore by M_{2^i} . Either way, we reach a contradiction, and conclude that M does not exist.

3. 9.2.2a

$A(2,1) = A(A(1,1),0)$ [rule 4] = $A(A(A(0,1),0),0)$ [rule 4] = $A(A(1,0),0)$ [rule 1] = $A(2,0)$ [rule 2] = 4 [rule 3].

4. 9.2.3a

Let's keep i , the integer in unary, whose square we have most recently printed on the output tape, on tape 1, and keep i^2 on tape 2, also in unary. Initially, $i = 0$. Repeatedly do the following:

1. Add 1 to tape 1; now we have $i+1$ there.
2. Copy tape 1 to tape 2 twice, and remove one to change i^2 to $i^2 + 2(i+1) - 1 = (i+1)^2$.
3. Copy tape 2 to the output as a block of 0's and append a 1.

5. 9.2.6a, c

To test whether an input w is in the union of two recursive languages $L1$ and $L2$, we design a TM to copy its input w onto a second tape. It then simulates the halting TM for $L1$ on one tape and the halting TM for $L2$ on the other. If either accepts, then we accept. If both halt without accepting, we halt without accepting. Thus, the union is accepted by a TM that always halts.

In the case where $L1$ and $L2$ are RE, do the same, and accept if either accepts. The resulting TM accepts the union, although it may not halt. We conclude that both the recursive languages and the RE languages are closed under union.

6. 9.3.6a

After making m transitions (not $m+1$ as suggested by the hint), the TM will have been in $m+1$ different states. These states cannot all be different. Thus, we can find some repeating state, and the moves of the TM look like $[q_0] \mid^* q \mid^* q \mid^* \dots$, where the central \mid^* represents at least one move. Note that we assume the tape remains blank; if not then

we know the TM eventually prints a nonblank. However, if it enters a loop without printing a nonblank, then it will remain forever in that loop and never print a nonblank. Thus, we can decide whether the TM ever prints a nonblank by simulating it for M moves, and saying "yes" if and only if it prints a nonblank during that sequence of moves.

7. 9.3.8d

This language is the complement of the language of Exercise 9.3.8(a), so it is surely not recursive. But is it RE? We can show it isn't by a simple reduction from the nonhalting problem. Given (M, w) , construct M' as follows:

1. M' ignores its own input and simulates M on w .
2. If M halts, M' halts on its own input. However, if M never halts on w , then M' will never halt on its own input.

As a result, M' fails to halt on at least one input (in fact, on all inputs) if M fails to halt on w . If M halts on w , then M' halts on all inputs.

8. 9.4.1a

There is no solution. First, a solution would have to start with pair 1, because that is the only pair where one is a prefix of the other. Thus, our partial solution starts:

A: 01
B: 011

Now, we need a pair whose A-string begins with 1, and that can only be pair 3. The partial solution becomes

A: 0110
B: 01100

Now, we need a pair whose A-string begins with 0, and either pair 1 or pair 2 might serve. However, trying to extend the solution with pair 1 gives us:

A: 011001
B: 01100011

while extending by pair 2 yields:

A: 0110001
B: 0110010

9. 9.4.3

The problem is decidable by the following, fairly simple algorithm. First, if all the A-strings are strictly longer than their corresponding B-strings, then there is surely no solution. Neither is there a solution in the opposite case, where all the B-strings are strictly longer than their corresponding A-strings.

We claim that in all other cases, there is a solution. If any corresponding pair of strings are the same length, then they are identical, and so just that pair is a solution. The only possibility remains has at least one pair, say i , with the A-string longer than the B-string, say by m symbols, and another pair, say j , where the B-string is longer than the A-string, say by n symbols. Then $i^n j^m$, i.e., n uses of pair i followed by m uses of pair j is a solution.

In proof, it is easy to check that both the A - and B -strings that result have the same length. Since there is only one symbol, these strings are therefore identical.

10. 9.5.1

Given an instance (A, B) of PCP, construct the grammar G_A as in the text. Also, construct a grammar G_{BR} , that is essentially G_B , but with the bodies of all productions reversed, so its language is the reverse of the language of G_B . Assume the start symbols of these grammars are A and B , they contain no variables in common, and that c is a terminal that does not appear in these grammars.

Construct a new grammar G with all the productions of G_A and G_{BR} , plus the production $S \rightarrow AcB$. Then a solution to the PCP instance yields a string y such that y is generated by G_A and y^R is generated by G_{BR} . Thus, G generates the palindrome ycy^R . However, any palindrome generated by G must have the c in the middle and thus implies a solution to the PCP instance, that is, a string y that appears in $L(G_A)$ while y^R appears in $L(G_{BR})$ [and therefore y appears in $L(G_B)$].