Cpt S 516 Homework 3

Tengyang Zhang

March 21th, 2021

1 Problem .1

Assuming L is recursive, all Turing Machines in L will eventually stop. In other words, these TMs either accept a string and answer ”Yes,” or they say ”No” to a string that does not belong to it. But there is a Turing Machine M1 in the set ¡M¿, and M1 will always run to check whether it contains the string ”abba,” which means that M1 will never stop so that M1 will become a non-halting Turing Machine. So our hypothesis does not hold, and L is not recursive.

2 Problem .2

If we want to prove that the problem of emptiness in Jump-LBA is undecidable. We just need to prove that ”write and jump to the beginning” can simulate ”normal write”. For example, suppose we have an LBA as shown below.

a	b	b	a	b	a

Table 1: LBA

We numbered this LBA from one. Suppose the current pointer is under the fourth cell. At state p, a ”normal write” operation will change the ”a” in the cell to ”b” and then move one unit to the right.

For Jump-LBA, at state p, change ”a” to ”B” (where B is capitalized, to distinguish it from the other b in LBA), and then the pointer will jump to the first cell, change the pointer moves to the right until find ”B,” change ”B” to ”b,” and move one unit to the right.

As mentioned above, we can use ”write and jump to the beginning” to imitate ”normal write,” so the emptiness problem for jump-LBAs is undecidable.

3 Problem .3

Assuming that the language L’ is r.e., then L’ must be accepted by the Turing machine M’. And for all x, there are two cases: If xL, then M will say ”Yes” on x. If xL, then M will not say ”Yes” on x. Since yxyL, and L is a recursive language. So Turing machine M’ will never say ”No” for any x. So the hypothesis holds. L is r.e.

4 Problem .4

Assuming that both L and Lr are recursive, Turing machines M and Mr accepting L and Lr separately. M will produce w1, w2,... in L. Mr will produce wr1, wr2,... in Lr. For any w, we can compare w and w1 generated by M. If they are not the same, we can compare w and wr1 generated by Mr. With this loop, M or Mr will generate any w, so we will definitely get a match. If M generates the matched string, then w belongs to L. Otherwise, w belongs to Lr. So if L is recursive, then Lr must also be recursive.

5 Problem .5

The emptiness problem of 2TM is whether I have a halting run on 2TM. Assume that 2TM is M(x,y). Suppose the state of 2TM in a certain second is (p, x, y). Suppose x=2 and y=3 at this time. We can use a string to represent this state: 11p000. Among them, "1" represents the number of x, and "0" represents the number of y. Suppose you execute p: y–, q in this state. Then the string of the next state will become 11q00. So for 2TM, we can get a sequence of configurations. We only need to check whether these sequences are halting the run of 2TM. For each string, the state transition is determined by matching the number of "1" and "0".