CS281 Spring 2010: Homework 4

Due Wednesday, May 12th – SOLUTION

1. **(15 points)**
We know by Rice's theorem that none of the following problems are decidable. However, are they recursively enumerable, or non-RE?
a) Does $L(M)$ contain at least two strings?
b) Is $L(M)$ infinite?
c) Is $L(M)$ a context free language? [1]
d) Is $L(M) = (L(M))^R$?

Sol'n:

a) *This problem is r.e.* – Given $L(M)$, we may use a nondeterministic machine M' to guess two distinct strings w and w' and run M on each of these. Let M' accept iff both w and w' are accepted. Thus M' accepts iff there exist two distinct strings in $L(M)$.

b) *This problem is NOT r.e.* – Indeed we may reduce membership in the diagonalization language $L_d = \{w_i : w_i \notin L(M_i)\}$, a known non-r.e. problem, to this problem as follows: Given the string w_i let N_i be a TM which on input w_j simulates M_i on w_i for j steps and accepts iff this simulation did not accept. Then $w_i \in L_d$ implies N_i accepts all strings, and so $L(N_i)$ is infinite. While $w_i \notin L_d$ implies that within some finite number J of steps M_i will accept w_i and so N_i will not accept any w_j with $j \geq J$ and thus $L(N_i)$ is finite.

c) *This problem is NOT r.e.* – Indeed we may reduce membership in the language $L_e = \{M : L(M) = \emptyset\}$, a known non-r.e. problem, to this problem as follows: Given the string w_i let N_i be a NTM which on input w, guesses a string w' and simulates M_i on w', accepting iff this simulation accepts and $w = 01$. Note that if $M_i \in L_e$ then M_i does not accept any strings and therefore neither does N_i, so $L(N_i) = \emptyset = \emptyset^R = L(N_i)^R$. While if $M_i \notin L_e$ then M_i will accept some w' and using this guess N_i will accept 01 and no other strings, hence $L(N_i) = \{ 01 \} \neq \{ 01 \}^R = L(N_i)^R$.

[1] Optional, try if you know what a CFL is, it will not be graded

2. **(10 points)**
Let L be the language consisting of pairs of TM codes plus an integer, (M_1, M_2, k), such that $L(M_1) \cap L(M_2)$ contains atleast k strings. Show that L is RE, but not recursive.

Sol'n: If L were decidable then, fixing a TM M_2 and an integer $k \in \mathbb{N}$, the language $L' = \{M_1 : (M_1, M_2, k) \in L\}$ would also be decidable. However, suppose we take any nonempty $L(M_2)$ and let $k = 1$ then the property of belonging to L' is clearly a non-trivial property of TM's (i.e. of RE languages). Indeed if $L(M_1) = \emptyset$ we have $M_1 \notin L'$, while if $L(M_1) = L(M_2)$ we have $M_1 \in L'$. Therefore, by Rice's Theorem, L' is not decidable, a contradiction.

The language L is however r.e.. To see this construct a NTM N which, given input (M_1, M_2, k) guesses k distinct strings $w_1, \ldots, w_k$ and simulates M_1 and then M_2 on each of these in turn, accepting iff all $2k$ simulations accept. Then clearly $(M_1, M_2, k) \in L(N)$ iff there are k distinct strings in $L(M_1) \cap L(M_2)$, i.e. iff $(M_1, M_2, k) \in L$.

3. **(15 points)**
Show that the following questions are decidable:
a) The set L of codes for TM's M such that, when started with the blank tape will eventually write some nonblank symbol on its tape. *Hint*: If M has m states, consider the first m transitions that it makes.
b) The set L of codes for TM's that never make a move left on any input.
c) The set L of pairs (M, w) such that TM M, started with input w, never scans any tape cell more than once.

Sol'n: a) Construct a TM M' which given the code for a TM M, counts the number m of states of M and then simulates M on a blank tape for up to m moves, accepting if ever the simulated M writes a non-blank symbol. By construction this is a halting machine, and if $M \notin L$ then $M \notin L(M')$. On the other hand, if $M \in L$ then suppose that M wrote its first non-blank symbol after j transitions. These transitions must start in j distinct states, otherwise we contradict the definition of j. Hence $j \leq m$ and so the machine M' will accept.

b) Construct a NTM N which given the code for a TM M, counts the number t of 5-tuples that M uses to define δ, and then proceeds as follows: N nondeterministically guesses a starting string $s, |s| \leq t$

and then simulates M on s for t steps; N halts (and rejects) if ever M makes a move left during this time, and accepts at the end of the simulation otherwise.

By construction, this is a deciding NTM, since all possible branches halt[2].

As for acceptance, note that if M *does* move left on some input x for the first time after r steps then it has been moving *steadily to the right* prior to this and so there is another input x' on which M moves to the left for the first time after r' steps with $r' < t$. This is because when $r \geq t$ some transition (not counting the one that moves left) will have been used twice, and the second time M will be scanning a cell to the right of the initial bit, thus all bits to the left of the currently scanned bit may be removed and M started on this shorter input with the same subsequent behaviour. This means that if $M \notin L$ then the NTM N can guess a string on which M moves left, i.e. N halts and rejects. On the other hand if $M \in L$ then no such string exists so N will surely accept.

c) *Note, that unlike in (a) and (b), instances of this problem specify both the input string w and the TM M.*

Let such a pair (M, w) be given. Construct a TM M' which first computes m, the number of states of M, and then computes $l = |w|$. Now let M' simulate one step of M on input w and check the direction M steps.

If M stepped right (left) then let the simulated M run for $m + l$ more steps. Halt rejecting if ever it makes a move left (right), and halt accepting at the end of the simulation otherwise.

By construction this is a halting TM. Moreover, when we are waiting for a move left (right) by the simulated M, this machine is steadily moving right (left). It may make at most l such steps before meeting a blank and then, as argued in (a), may make at most m additional steps before changing direction if indeed it ever changes direction. Now, if $(M, w) \notin L$, the simulated M on w *will* eventually change direction and so will do so within $l + m$ steps and M' will reject (M, w). While if $(M, w) \in L$ then no simulation of M on w (for however many steps) will involve a change of direction and hence M' will accept.

[2](assuming that the NTW constructs a bounded length "guess" string on a working tape by - as long as it has not yet written b bits - repeatedly picking to write a 0 and move right, or a 1 and move right, or else to stop)