CPSC 313 Fall 2020
Assignment 6
Ali Akbari
30010402

1. Prove that the following decision problem is undecidable by reducing it to a known unsolvable problem. Be sure to clearly define your reduction. It should be written like a pseudocode program that is straightfoward to implement. Two different TAs reading it should have the same understanding of how your reduction works.

(a) Given a TM T and a word w, does T accept w in an even number of moves?

D = Does T accept w in an even number of moves?
By contradiction we first assume that D is solvable. Therefore there exist an always halting turing machine T_D that decides this problem. We will use it to solve the problem of:
E = “given T and w, does T halt on w.”

Our algorithm is the following for T_E:

```
bool does_T_halt_on_w(TM T, WORD w) {

T ’ = T with ha and hr swapped.
Ti = T with a state q in front of the machine that makes no difference other than
increasing the number of transitions by one, i.e even number moves become odd.
Ti ’ = T with ha and hr swapped.

if (T_accepts_w_in_even_number_of_moves(T, w) == True){
return true
}
else if (T_accepts_w_in_even_number_of_moves(T ‘ , w) == True){
return true
}

if (T_accepts_w_in_even_number_of_moves(Ti, w) == True){
return true
}
else if (T_accepts_w_in_even_number_of_moves(Ti ‘ , w) == True){
return true
}

return false
}
```

The subroutines represents our T_D.
The first if statement returns true if and only if T accepts w in an even number of moves.
The second statement returns true if and only if T rejects w in an even number of moves.
The third if statement returns true if and only if T accepts w in an odd number of moves.
The second statement returns true if and only if T rejects w in an odd number of moves. The inner if statements suggest that T or T ‘ halts on w, the return false statement suggests that T does not halt on w. However the algorithm halts if we assume that T_D halts. In the lecture notes we are shown that if T accepts w it is also undecidble therefore it cannot halt. Therefore such a Turing machine T_D cannot exist.

(b) Given a TM T, a word w, and a state q such that $q \neq ha$ and $q \neq hr$, does T ever enter q when processing w.

D = Does T ever enter q when processing w?
By contradiction we first assume that D is solvable. Therefore there exist an always halting turing machine T_D that decides this problem. We will use it to solve the problem of
E =“given T and w, does T halt on w.”

Our algorithm is the following for T_E:

```
bool does_T_halt_on_w(TM T, WORD w) {

T ' = T but with a new state q, where all states in T that pointed to ha, now point
the new q.

T ” = T but with a new state q, where all states in T that pointed to hr, now point
the new q.

if (does_enter_q_while_processing_w(T ‘, w) == True){
return true
}
if (does_enter_q_while_processing_w(T “ , w) == True){
return true
}

return false
}
```

The subroutines represents our T_D .The first if statement returns true if and only if T ‘ accepts w if it enters q(ha of T) while processing w. The second statement returns true if and only if T ’’ accepts w if it enters q(hr of T) while processing w. The inner if statements suggest that T ‘ or T “ halts on w, the return false statement suggests that T does not halt on w. However the algorithm halts if we assume that T_D halts. In the lecture notes we are shown that if T accepts w it is also undecidble therefore it cannot halt. Therefore such a Turing machine T_D cannot exist.

(c) Given a TM T, and two words w and x, does T accept either wx or xw?

D = Does T accept w either wx or xw?
By contradiction we first assume that D is solvable. Therefore there exist an always halting turing machine T_D that decides this problem. We will use it to solve the problem of E = "given T, does T accept ε." Observe that if w, x = ε then wx, and xw = $\varepsilon * \varepsilon = \varepsilon$.

Our algorithm is the following for T_E:

```
bool T_accepts_epsilon(T) {
return T_accepts_wx_or_xw(T, epsilon, epsilon)
}
```

The subroutines represents our T_D.
The return statement returns true if and only if T_D accepts w and x.The return statement suggests that T_D does halt on w, and x when equal to epsilon. However the algorithm T_E halts if we assume that T_D halts. In the lecture notes we are shown that if T accepts ε it is also undecidble therefore it cannot halt. Therefore such a Turing machine T_D cannot exist.