3331 Assignment 4

Zaid Albirawi
250626065

1. For each of the following languages, prove, without using Rice's Theorem, whether it is in D, in SD but not in D or not in SD.

   a. $\{< M > \mid L(M) \text{ contains at least two strings}\}$.

   The language is in SD. We can run the dovetailing algorithm to enumerate all the strings in the language in a lexicographical order and as soon as the machine finds two strings it will halt and accept, otherwise, it will keep enumerating until it finds at least two strings.

   To prove this, we reduce H to L, where $H = \{< M, \omega >: \text{TM } M \text{ halts on } \omega\}$. Also, we assume that their exits a TM, $Oracle$, that decides the language generated by the reduction, R, where R is reduction of H to L.

   $R(< M, \omega >) =$

   1. Construct $< M\# >$, where $M\#$ operates as follows:
      1.1. Erase the tape
      1.2. Write ω on the tape
      1.3. Run M on ω
      1.4. Accept
   2. Return $< M\# >$

   Finally, take the composite of the $Oracle$ and L, C, to decide the problem.
   If $Oracle$ exists, $C = Oracle(R(< M, \omega >))$ decides H.

   - $< M, \omega > \in H$: M halts on ω, so $M\#$ halts on everything. Therefore it halts on at least two strings. $Oracle$ accepts.

   - $< M, \omega > \notin H$: M does not halt on ω, so $M\#$ halts on nothing. Therefore it does not halt on at least two strings. $Oracle$ rejects.

   However, there doesn't exist a machine that can decide H, therefore, the $Oracle$ does not exist.

b. $\{< M > \mid L(M) \text{ is infinite}\}$.

The language is not in SD. To prove this, we reduce $\neg H$ to L, where $\neg H = \{< M, \omega >: \text{TM } M \text{ does not halt on } \omega\}$. Also, we assume that their exits a TM, $Oracle$, that semi-decides the language generated by the reduction, R, where R is reduction of $\neg H$ to L.

$R(< M, \omega >) =$

1. Construct $< M\# >$, where $M\#(x)$ operates as follows:
   1.1. Copy the input x on a second tape
   1.2. Erase the tape
   1.3. Write ω on the tape
   1.4. Run M on ω for $|x|$ or until it naturally halts
   1.5. If M naturally halted, loop
   1.6. Accept
2. Return $< M\# >$

If $Oracle$ exists, $C = Oracle(R(< M, \omega >))$ semi-decides $\neg H$.

- $< M, \omega > \in \neg H$: M does not halt on ω, so $M\#$ does not halt in $|x|$ steps, regardless of x's size. Hence, $M\#$ will always reach step 1.6, which makes this language infinite. $Oracle$ accepts.

- $< M, \omega > \notin \neg H$: M halts on ω, so $M\#$ halts in $|x|$ steps. If $|\omega| \leq |x|$, $M\#$ will reach 1.6 and halt and accept. If $|\omega| > |x|$ then the machine will naturally halt at 1.4, which will lead it to enter 1.5 which will loop infinitely. Hence, this is a finite set. $Oracle$ fails to accept.

However, there doesn't exist a machine that can semi-decide $\neg H$, therefore, the $Oracle$ does not exist.

c. $\{< M > \mid L(M) \text{ is not context} - \text{free}\}$.

The language is not in SD. To prove this, we reduce $\neg H$ to L, where $\neg H = \{< M, \omega >: \text{TM } M \text{ does not halt on } \omega\}$. Also, we assume that their exits a TM, $Oracle$, that semi-decides the language generated by the reduction, R, where R is reduction of $\neg H$ to L.

$R(< M, \omega >) =$

1. Construct $< M\# >$, where $M\#(x)$ operates as follows:
    1.1. If $x \in A^n B^n C^n$ then accept. Else:
    1.2. Erase the tape
    1.3. Write ω on the tape
    1.4. Run M on ω
    1.5. Accept
2. Return $< M\# >$

If $Oracle$ exists, $C = Oracle(R(< M, \omega >))$ semi-decides $\neg H$.

- $< M, \omega > \in \neg H$: M does not halt on ω, so $M\#$ halts at 1.1. and accepts, therefore, ω is not context-free. $Oracle$ accepts.

- $< M, \omega > \notin \neg H$: M halts on ω, so $M\#$ reaches 1.5 which implies that it accepts everything. $Oracle$ does not accept.

However, there doesn't exist a machine that can semi-decide $\neg H$, therefore, the $Oracle$ does not exist.

d. $\{< M > \mid M \text{ is the only Turing machine that accpets } L(M)\}$.

This language is decidable, because it is empty. There does not exists a Turing machine that is distinguishable from all other Turing machines. Meaning, for every Turing machine there is at least one other Turing machine that is equivalent. Nevertheless, for every Turing machine there exists an infinite number of equivalent Turing machines.

2. For each of the languages in question 1 prove whether Rice's Theorem can be used or not.

   a. $\{< M > \mid L(M) \text{ contains at least two strings}\}$.

      Rice's theorem does not apply. The property, P, does not ask about the language of the machine.

   b. $\{< M > \mid L(M) \text{ is infinite}\}$.

      Rice's theorem does apply. The property, P, of the language is the finiteness of the language. The domain of P is SD. The property is non-trivial because it is true if the language is infinite, and false if the language is finite.

   c. $\{< M > \mid L(M) \text{ is not context} - \text{free}\}$.

      Rice's theorem does apply. The property, P, of the language is whether the language falls under the context-free set of languages or not. The domain of P is SD. The property is non-trivial because it is true if the language is not context free (determined by checking if the language belongs to $A^nB^nC^n$), and false if the language is context-free.

   d. $\{< M > \mid M \text{ is the only Turing machine that accpets } L(M)\}$.

      Rice's theorem does not apply. The question being asked is about the machine itself not the language that the machine accepts.