TOC Assignment 4

Submitted By - Sayam Kumar
Roll No - S20180010158
UG2 Section - A

Part 1. Closure under Union

Answer - Both Recursive and Recursively Enumerable Languages are closed under union.

Proof for Recursive Languages -

Let's suppose we have two Turing Machines M1 and M2 for two Recursive Languages L1 and L2 respectively. Then we can simulate a Multi-tape Turing Machine M (for Union) with both tapes of M1 and M2 stacked over each other. The behaviour of M can be categorised for a string 's' as -

1. If both M1 and M2 accept string 's' **or** either of M1 or M2 accepts string 's', then M also accepts that string.
2. If both Turing Machines M1 and M2 halt i.e. both reject the string 's', then M also does not accept that string.

Considering both points above, the Recursive Languages are closed under Union.

Proof for Recursively Enumerable Languages -

Let's suppose we have two Turing Machines M1 and M2 for two Recursively Enumerable Languages L1 and L2 respectively. Then we can simulate a Multi-tape Turing Machine M (for Union) with both tapes of M1 and M2 stacked over each other. The behaviour of M can be categorised for a string 's' as -

1. If both M1 and M2 accept string 's' **or** either of M1 or M2 accepts string 's', then M also accepts that string.
2. If both Turing Machines M1 and M2 either halt or loop forever (indicating partial decidability), then M also halts or loops forever.

Considering both points above, the Recursively Enumerable Languages are closed under Union.

Part 2. Closure under Intersection

Answer - Both Recursive and Recursively Enumerable Languages are closed under Intersection.

Proof for Recursive Languages -

Let's suppose we have two Turing Machines M1 and M2 for two Recursive Languages L1 and L2 respectively. Then we can simulate a Multi-tape Turing Machine M (for Intersection) with both tapes of M1 and M2 stacked over each other. The behaviour of M can be categorised for a string 's' as -

1. If both M1 and M2 accept string 's', then only M also accepts that string.
2. If both Turing Machines M1 and M2 halt i.e. both reject the string 's' **or** either of M1 or M2 accepts string 's', then M also does not accept that string.

Considering both points above, the Recursive Languages are closed under Intersection.

Proof for Recursively Enumerable Languages -

Let's suppose we have two Turing Machines M1 and M2 for two Recursively Enumerable Languages L1 and L2 respectively. Then we can simulate a Multi-tape Turing Machine M (for Intersection) with both tapes of M1 and M2 stacked over each other. The behaviour of M can be categorised for a string 's' as -

1. If both M1 and M2 accept string 's' , then M also accepts that string.
2. If both Turing Machines M1 and M2 either halt or loop forever (indicating partial decidability) **or** either of M1 or M2 accepts string 's', then M also halts or loops forever.

Considering both points above, the Recursively Enumerable Languages are closed under Intersection.

Part 3. Closure under Complementation

Answer - Recursive Languages are closed under Complementation but Recursively Enumerable Languages are not.

Proof for Recursive Languages -

Let's suppose we have a Turing Machine M for Recursive Language L, then we can make another Turing Machine M' which accepts L' by simply changing the non-final into final states and final into non-final states. This is driven by the fact that machine M always halts for any input. In the same way, by reversing the nature of states, the newly formed machine M' will also halt for not accepting strings which are accepted by M.

Proof for Recursively Enumerable Languages -

Let's suppose we have a Turing Machine M for Recursively Enumerable Language L. So, machine M may or may not halt for strings which are not accepted by the language. Now, it is not possible to make Turing Machine M' to accept all the rejected strings of M as M is semi-decidable. First, we need to capture those rejected strings to pass to M'. But we cannot capture those rejected strings as M may loop forever. In this way, we cannot build M' for complementation of Recursively Enumerable Language L.

Part 4. Closure under Concatenation

Answer - Both Recursive and Recursively Enumerable Languages are closed under Concatenation.

Proof for Recursive Languages -

Let's suppose we have two Turing Machines M1 and M2 for two Recursive Languages L1 and L2 respectively. Then we can simulate a Turing Machine M (for Concatenation). The behaviour of M can be categorised for a string 's' as -

1. If a string 's' is accepted by M, then we can find a split of the string 's' as **' s1s2 '** such that 's1' is accepted by M1 and 's2' is accepted by M2. For finding a split of the string 's', it is also possible to iterate over it for all combinations. Thus, the overall complexity of acceptance is increased by order of length of the string 's'.
2. If we cannot find a split of the string 's', then machine M rejects the given string 's'.

Considering both points above, the Recursive Languages are closed under Concatenation.

Proof for Recursively Enumerable Languages -

The same proof logic applies to Recursively Enumerable Languages. If we are able to find a split of string 's' into parts such that one part is accepted by M1 and other by M2, then accept that string. Otherwise reject that string.

Part 5. Closure under Star Operation

Answer - Both Recursive and Recursively Enumerable Languages are closed under Star Operation.

Proof - Above we have seen that both Recursive and Recursively Enumerable Languages are closed under Concatenation. The Star Operation is concatenation of the same language with itself infinite times. So, the languages R and RE are closed under Star Operation because they are closed under Concatenation.