Alex Ojemann 109722375

Consider two operations on sets that we've discussed frequently: concatenation and asterate. These are both are closure properties of regular languages and of context-free languages. For both of these operations, determine whether it is a closure property of r.e. languages, and whether it is a closure property of recursive languages. Give a construction/proof for each of these four combinations.

Note: Avoid extremely detailed pseudocode. Your answers should be roughly the equivalent of a paragraph for each.

Hint: As always, beware of looping machines in your constructions...

Proof. Concatenation of R.E.:

Let L_1 and L_2 be r.e. languages.

Let M be the two tape Turing machine representing the concatenation of L_1 and L_2 where tapes 1 and 2 are the Turing machines that accept on strings in L_1 and L_2 respectively. We must nondeterministically guess where to break up the input string and put the first part into tape 1 and the second into tape 2.

M will accept when both of the tapes tapes accept. Since each of the tapes represent r.e. languages and M will accept when both of the tapes are accepted, the language accepted my M (L_1L_2) must also be r.e.

Concatenation of recursive:

Let L_1 and L_2 be r.e. languages.

Let M be the two tape Turing machine representing the concatenation of L_1 and L_2 where tapes 1 and 2 are the Turing machines that accept on strings in L_1 and L_2 respectively and reject on strings not in L_1 and L_2 respectively. We must nondeterministically guess where to break up the input string and put the first part into tape 1 and the second into tape 2.

M will accept when both tapes accept, and will reject when all of the possible guesses are rejected by either machine. Since each of the tapes represent recursive languages and M, in its worst case, will halt when both tapes are halted or all guesses are exhausted, the language accepted my M (L_1L_2) must also be recursive.

Asterate of R.E.:

Let L be an r.e. language and let x be in L^*.

Let M be the Turing machine accepting L. We must nondeterministically guess where to break up x and how many parts to break it into.

When M is run on each part of x, M accepts all parts of x, so x is accepted, so L^* must also be r.e.

Asterate of recursive:

Let L be an recursive language, let x be in L^*, and let y be outside L^*.

Let M be the Turing machine accepting L and rejecting anything not in L. We must nondeterministically guess where to break up the input string and how many parts to break it into.

When M is run on each part of x, M accepts all parts x, so x is accepted, meanwhile none of the possible guesses for y will accept so y is rejected. Thus, L^* must also be recursive. □