PS06-02

February 12, 2018

a Intersection.
Much like union. Given two languages A and B, let M_A and M_B be two Turing machines s.t. $L(M_A) = A$ and $L(M_B) = B$.
Run both simultaneously on two tracks, and if both accept, then accept

b Concatenate.
Given two languages A and B, let M_A and M_B be two Turing machines s.t. $L(M_A) = A$ and $L(M_B) = B$.
Let w be an input string. $w = w_1 \circ w_2$ Let $|w_1| = 1$ and $|w_2| = |w| - |w_1|$
Run M_A on w_1 and M_B on w_2. If both accept, accept, otherwise add 1 to $|w_1|$ and try again. Keep trying, increasing $|w_1|$ until accept or until the length of w_1 equals the length of w_2, at which point, reject.

c *
This one is computationally intensive. Start the same way as concatenate. Essentially, run M_A on every possible set of substrings of the input string, and if at any point, all the strings in the set accept, accept.

d In order to prove closure for recursive languages, you need to prove decidability for the constituent languges. In the event of A not being a recursive language(just r.e.), intersection could potentially never reject, just accept and loop forever.