Name____________________

CS503
Homework #7
Solutions

#1. Page 321, #8 a,b

Design machines that compute the following relations. You may use the macros and machines constructed in Sections 9.2 through 9.4 and the machines constructed in Exercise 5.

a) $gt(n, m) = \{$ 1 if n > m, 0 otherwise

$gt(n, m) = \{$ 1 if n > m, 0 otherwise = 1 in unary

$gt(n, m)$

1. Compute monus(m,n)
2. if result = 0, halt
3. otherwise
    i. erase *m*
    ii. erase *n*
    iii. write a 1

b) $persq(n) = \{$ 1 if n is a perfect square, 0 otherwise

$persq(n)$

1. create a new variable on the tape, $k = 0$
2. run $mult(k, k)$
3. if the result of the $mult < n$
    i. run successor function on *k*
    ii. go back to #2
4. if result of $mult = n$
    i. erase n, k, $mult(k, k)$
    ii. write a 1
5. if result of $mult > n$
    i. erase n, k, $mult(k, k)$
    ii. write a 0

#2. Page 321, #9 a, b, c

Trace the actions of the machine MULT for computations with input

a) n = 0, m = 4

MULT 0, 4 q_0B1B11111B

- read the 1^{st} "1", and move right
- erase *m*
- move left 1 unary number
- halt

b) n = 1, m = 0

MULT 1, 0 q_0B11B1B

- read the 1^{st} "1", and move right
- replace the next "1" with an "X", and move right
- move right
- add 0, 0
- replace "X" from before with a "B"
- erase all the 0's after the 1^{st} one
- go back to the beginning
- halt

c) n = 2, m = 2.

MULT 2, 2 q_0B111B111B

- read the 1^{st} "1" – mark 1 iteration
- copy unary "2" to the end
- mark 1 iteration
- add 2, 2
- erase all trailing unary 2's
- go back to the beginning

#3. Page 321, #10 a-d

Describe the mapping defines by each of the following composite functions:

a) $add \circ (mult \circ (id, id), add \circ (id, id))$

- $add \circ (mult \circ (id, id), add \circ (id, id))$

$= add \circ (id \cdot id, add \circ (id, id))$

$= add \circ (id \cdot id, id + id)$

$= id \cdot id + id + id$

b) $p_1^{(2)} \circ (s \circ p_1^{(2)}, e \circ p_2^{(2)})$ **Strictly speaking, this is undefined because *e* is undefined – see definition of composition. But I accepted:**

- $p_1^{(2)} \circ (s \circ p_1^{(2)}, e \circ p_2^{(2)}) (n_1, n_2)$

$= p_1^{(2)} \circ (n_2 + 1, e \circ p_2^{(2)})$

$= n_2 + 1$

c) $mult \circ (c_2^{(3)}, add \circ (p_1^{(3)}, s \circ p_2^{(3)}))$

- $mult \circ (c_2^{(3)}, add \circ (p_1^{(3)}, s \circ p_2^{(3)})) (n_1, n_2)$

$= mult \circ (2, add \circ (n_1, n_2 + 1))$

$= mult \circ (2, n_1 + n_2 + 1)$

$= 2n_1 + 2n_2 + 2$

d) $mult \circ (mult \circ (p_1^{(1)}, p_1^{(1)}), p_1^{(1)})$.

$= mult \circ (mult \circ (p_1^{(1)}, p_1^{(1)}), p_1^{(1)}) (n_1, n_2)$

$= mult \circ (n_1 \cdot n_1, n_1)$

$= n_1 \cdot n_1 \cdot n_1$

#4. Page 322, #11 a,b

Give examples of total unary number-theoretic functions that satisfy the following conditions:

a) *g* is not *id* and *h* is not *id* but $g \circ h = id$. Using domain = $\mathcal{N}$:

- $g = pred$
- $h = successor$
- $g \circ h = id$

b) *g* is not a constant function and *h* is not a constant function but $g \circ h$ is a constant function. Using domain = $\mathcal{N}$

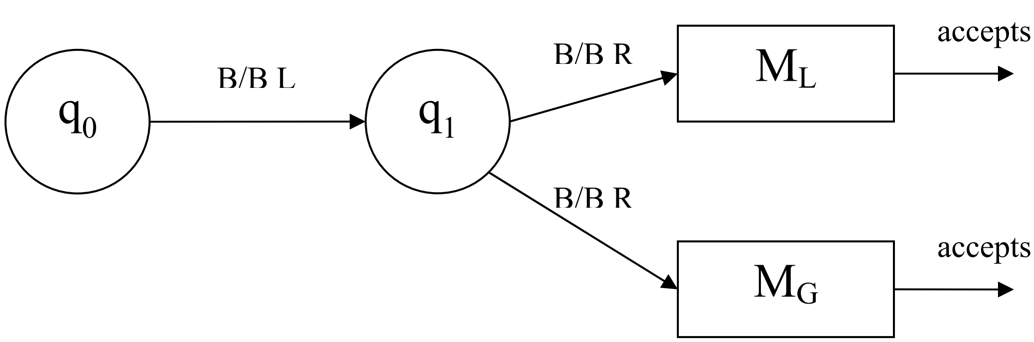
- $g = s$
- $h = z$
- $g \circ h = c_1$

#5. a) Page 339, #4 a-d

Prove that the recursively enumerable languages are closed under the following operations:

a) union

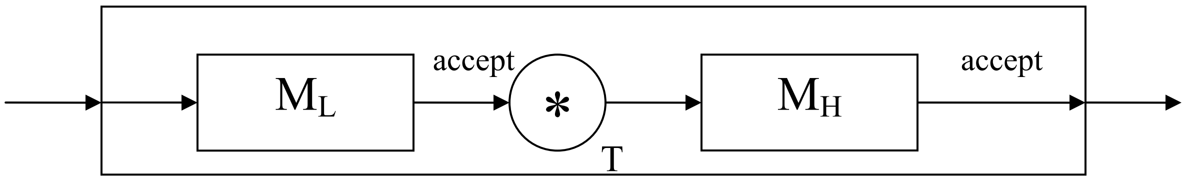
Suppose we have L, G are re languages.
L and G have TMs M_L and M_G, respectively.
Create a new TM, T.



$L(T) = L \cup G$

b) intersection

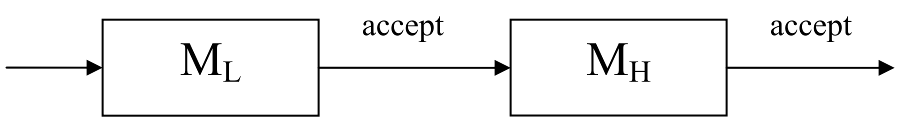
If L, H are re languages, they have TMs M_L and M_H respectively.
Construct a new TM, T such that



$L(T) = L \cap H$, and $L(T)$ is a re language because it has a TM that accepts

c) concatenation

If L, H are re languages, they have TMs M_L and M_H respectively.
Construct a new TM, T:

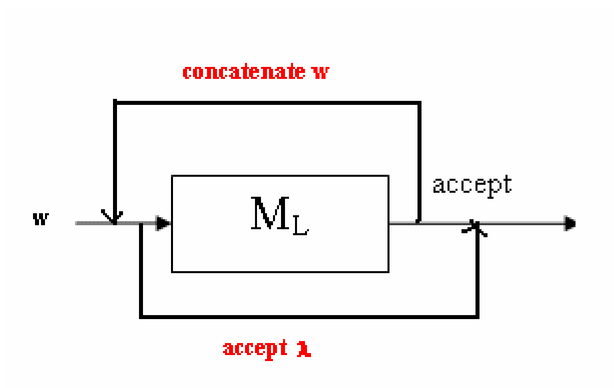


$L(T) = L \cdot H$, and $L(T)$ is a re language because it has a TM

c) Kleene star

L is a re language

L^* is $\{\lambda \cup L^n \mid n > 0\}$

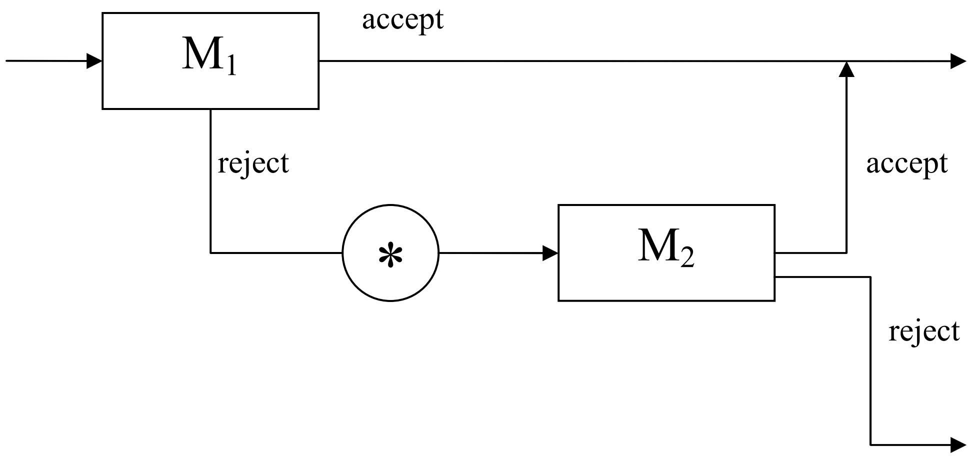


b) Do the same for recursive languages

d) union

Given the recursive languages L_1, L_2 with the always-halt TMs M_1, M_2

$L_3 = L_1 \cup L_2$ can be represented with the following TM:

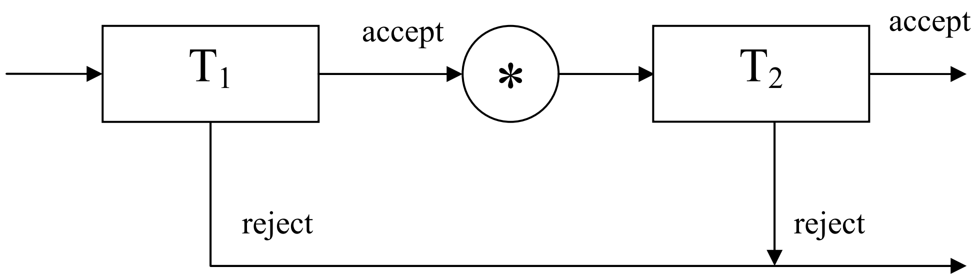


L_2 is a recursive language because it has an always-halt TM.

e) intersection

If L_1, L_2 are recursive languages, they have always-halt TMs T_1, T_2

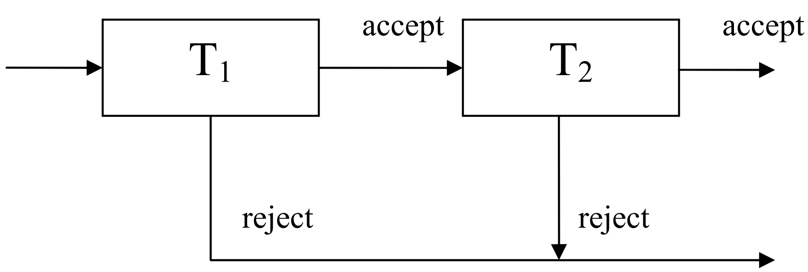
$L_3 = L_1 \cap L_2$ is a recursive language because it has the following always-halt TM:



c) concatenation

If L_1, L_2 are recursive languages, they have always-halt TMs T_1, T_2

$L_3 = L_1 \cdot L_2$ is a recursive language because it has the following always-halt TM:



f) Kleene star

Similar to above.

c) Show re languages are not closed under complement.

Let L be a re language.
Assume ~L is a re language.
Then L is recursive by Theorem
But, because not all recursive languages are re languages,
(we've shown there are uncountably many languages and countably many Turing Machines), we have a contradiction.
So ~L is necessarily a   re language.
Therefore, re languages are not closed under compliment.