CSci 435: Formal Languages and Automata

Instructor: Dr. M. E. Kim Name: \_Pedro Schmidt\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Due 11/21

**Home Assignment 7: 120 points + 25 points (optional)**

Q1. [20] For a given language below, construct a TM with a *single final state* that accepts it.

1. [6] L = {w ||*w*|is a multiple of 4} where Σ = {*a*, *b*}.

A close up of a map

Description automatically generated

1. [7] L = {w | *na*(*w*) ≠ *nb*(*w*)} where Σ = {*a*, *b*}.

Diagram

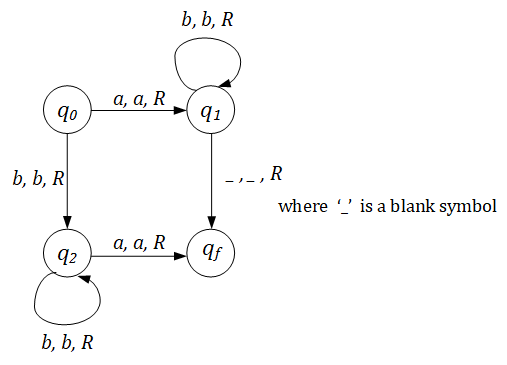
Description automatically generated

1. [7] L = {w | *anbn anbn* | *n* ≠ 0} where Σ = {*a, b*}.

A close up of a map

Description automatically generated

Q2. [10] What language is accepted by the Turning machine whose transition graph is in the figure?



The turning machine accepts strings of the form a(b)n or bn(a).

The q0->q1 ->q3 transitions accepts a(b)n strings, while q0->q1 ->q3 accepts (b)n(a).

So, if T is the turning machine.

L(t) = (strings either start or end with a and rest elements are all b)

**L = ab\* + bb\*a**

Q3. [10] Construct a TM that accepts L = {ww | w ∈ {*a, b*}+ }.

Hint: This is a standard deterministic TM.

So, TM has to **find** the middle of the string first; then, compare two halves.

Chart

Description automatically generated

Q4. [20] Construct a TM that computes the following function

1. [10] .

The input *w* is in the unary representation.

A close up of a map

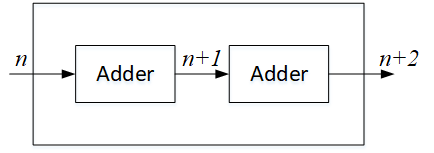
Description automatically generated

1. [10]  *f*(*x, y*) = *x* + 2*y.* A picture containing clock, object

   Description automatically generated

Q5. [20] Using adders, subtracters, comparers, copiers or multipliers, draw block diagram for TM that compute the functions:

e.g.) *f*(*n*) = *n* + 2

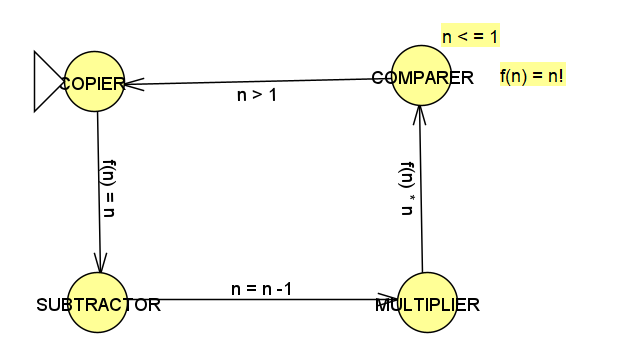


1. [10]*f*(*n*) = *2n.*

Diagram

Description automatically generated

1. [10] *f(n) = n!*

**

Q6. [15] For a two-tape Turing Machine,

1. [5] Give a *formal definition* of a transition function δ in two-tape TM.

A transition function δ is: Q x Γk → Q x Γk x { L , R , S }k where k = 2 in the case of a two-tape TM. Q is the set of states and Γ is the tape alphabet.

1. [10] Construct a two-tape TM that accepts L = { *anbn cn* | *n* ≥ 1}

A close up of a map

Description automatically generated

Q7. [15, optional] Construct a **Nondeterministic** TM (NTM) that accepts L ={ *wwRw* | *w* ∈ {*a, b*}+ }.

1. Draw its transition graph,

Mark a as X and enter qa. (X is for first segment w1, qa will move forward to mark the end of the second segment non-deterministically.)

A picture containing clock

Description automatically generated

1. explain how your transitions work out and

qa moves forward, mark an a as Y and enters qa1. (non-deterministically deter-mine the end of the second segmentw2, that is,

δ(qa,a) = {(qa,a,R),(qa1,Y,R)})

Since we must process a string, process the reverse, and then process the original again, we need at least 3 states before the blank transition since the minimal accepted full-length string would be of length 3. Since it’s nondeterministic, we don’t need to ensure the middle of wwR splits properly, since we can have multiple transitions for a single tape input.

(C) how the nondeterministic simplifies the case.

δ(qa1,a) = (Z,L) (Zis for the third segment w3).

Note that the middle of the string in wwR can be guessed in NTM.

Since it is nondeterministic, you don’t have to computer or decide the middle part of wwR

Q8. [10] Give the encoding, using the suggested method in the slide of Chap.9-#25-#27, for

δ(*q1, a1*) = (*q1, a1*, R); δ(*q1, a2*) = (*q3, a1*, L); δ(*q3, a1*) = (*q2, a2*, L)

1010101011 First transition

00 101101110101 Second transition

00 1110101101101 Third transition

Q9. [5] If *a* is encoded as 1, *b* as 11, R as 1, L as 11, decode the string 011010111011010.

If *a* is encoded as 1, *b* as 11, R as 1, L as 11

= 0E(q2)0E(a1)0E(q3)0E(a2)0E(R)0

->  (q2,a,q3,b,R)

Q10. [10, optional] Describe an algorithm that examines a string in {0, 1}+ to determine whether or not it represents an encoded Turing Machine.

Every turing machine has a finite description any specific machine will eventually be generated by this process.

1. Initial is at the left of the input, then replace it with an a.
2. Move the read and write head to the right, stopping( nondeterministically at the center of the input.
3. Compare the symbol there with the remembered one, if they match write b in the cell, if they don’t match reject the input
4. Center of the input marked with b we can move left and right comparing symbols.

Q11. [10] Describe how Linear Bounded Automata could be constructed to accept

L = { *an* | *n* is a prime number}.

L = an divides n by all natural number m where 1≤ m ≤ n−1. This only accepts when no m divides n, or else it rejects after trying all m < n.