CSci 435: Formal Languages and Automata

Instructor: Dr. M. E. Kim Date: December 8th, 2020

**Due: by the end of the day, 12/15 (Tue.)**

**Final Exam: 120 points + 30 points (optional)**

Name: \_\_\_\_\_\_\_\_\_Derek Trom\_\_\_\_\_\_\_\_\_\_\_

1. Write your answer below the corresponding question.
2. Do not include the photo image of your handwriting for a text.

Abbreviation:

REG: Regular Language, REC: Recursive Language, RE: Recursive Enumerable Language,

(D)CFL: (Deterministic) Context-Free Language, FA: Finite Automata, TM: Turing Machine,

UG: Unrestricted Grammar, CSG: Context-Sensitive Grammar, etc.

Mark the followings.

Difficulty:

Very Easy: \_\_\_\_\_\_ Easy: \_\_\_\_\_\_ Moderate: \_\_\_X\_\_ Difficult: \_\_\_\_\_\_ Very Difficulty: \_\_\_\_\_\_

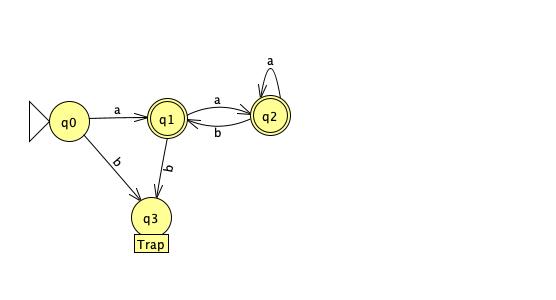
Time:

\_\_\_\_6\_\_\_ Hours and \_\_\_\_0\_\_\_\_ minutes

Q1. [10] For a language *L*(*aa*\*(*ab*+*a*)\*),

1. [5] Find a regular grammar that generates L.
   1. S -> aA | a
   2. A-> aA | aB | a
   3. B-> bA | b
2. [5] Construct the minimal DFA that accepts L.

Hint: the number of states is four.



Q2. [10] Prove that L= { *anbn ambm* | *n, m* ≥ 1 } is Context-Free Language but not Linear.

Hint: Construct its CFG and prove its non-linearity using P.L. for linear language.

S ⟶ AB

A ⟶ aAb | ab

B ⟶ aBb | ab

W = uvixyiz

uvyz<=m

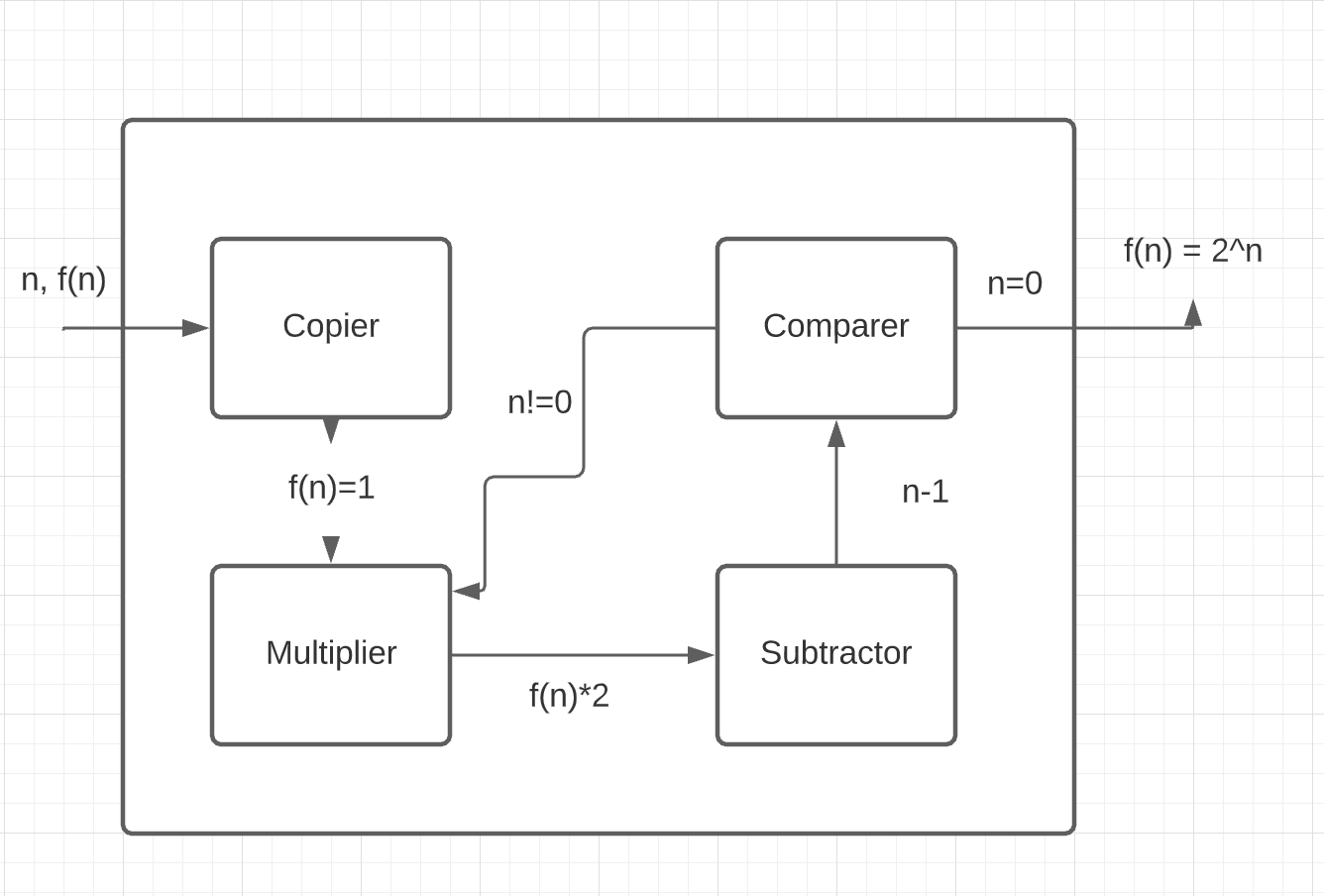
vy>0

Given ambmambm

Notice that the only choice left for v and y is v=as and y=at where at least one of s,t is greater than 0.

Assume s >0, if we pump I times we get the string am+(i-1)s+(i-1)tbmambm. When I = 2 we get w=am+s+tbmambm. Since s>0 the string is no longer in L thus a contradiction and L is not linear.

Q3. [10] Using adders, subtracters, comparers, copiers, or multipliers, construct a Turing Machine that compute the functions: *f*(*n*) = 2*n .* Draw its block diagram.



Q4. [10] Design a TM that computes the function: *f*(*x*) = 2*x* + 1, where *x* is given in unary notation with 1’s only to TM.

Use the tape symbol Γ = {1, *a*, €}. Draw its transition function.

δ (q0, 1) = (q0, a, R);

δ (q0, €) = (q1, €, L);

δ (q1, 1) = (q1, 1, L);

δ (q1, a) = (q2, 1, R);

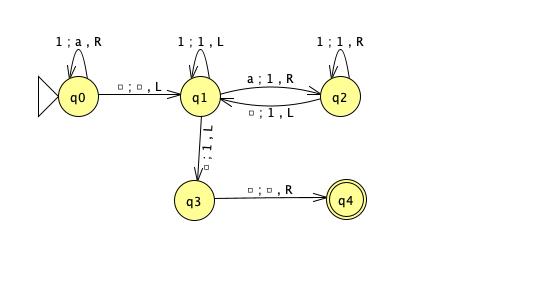
δ (q2, 1) = (q2, 1, R);

δ (q2, €) = (q1, 1, L);

δ (q1, €) = (q3, 1, L);

δ (q3, €) = (q4, €, R);

with final = {q4}.



Q5. [10] Prove that the complement of a Context-Free Language must be Recursive.

All CFLs are recursive. Recursive languages are closed under complement. Therefore, if a language 𝐿 is CFL then it is also recursive and hence, 𝐿𝐶 is also recursive.

Q6. [10] Give the language generated by the following unrestricted grammar in a formal expression.

e.g.) { *anbn* | *n* ≥ 1 }

S → AB, A → *a*Ab, bB → bbbB, *a*Ab → *aa*, B→λ

{anbm{n>=2, m>= n-2

b odd if a odd

b even if a even

}}

Q7. [10] Find a context-sensitive grammar for L = {*anbna2*n | *n* ≥ 1} and give a derivation of any string *w*∈ L by your grammar: S ⇒\* *w*.

S -> abaa | aAbaa

Ab -> bA

Aaa -> Bbaaaa

bB -> Bb

aB -> aa | aaA

A derivation of the string can be shown as:

aAbaa

abAaa

abBbaaaa

aBbbaaaa

aabbaaaa

Q8. [10] Let M1 and M2 be arbitrary Turing machines. Show that the problem “L(M1) ⊆ L(M2)" is undecidable.

Let’s try to construct a TM M when given a string in the form of (M1, M2) such that L(M1) is a subset of L(M2) is always accepted and for any other input it rejects or goes into a loop. We need to be able to check if every string recognized by M1 is also recognized by M2. M is unable to recognize whether a string is accepted right off the bat. It must run through all strings and simulate each of the machines (M1 and M2) on these strings. If it ever does run into a string that is rejected by M2 and accepted by M1 then the statement “L(M1) ⊆ L(M2)" is definitely not true. But if M1 or M2 ends up looping on a certain string M would never be able to halt and accept. Thus it is at best undecidable.

Q9. [10] Show that for A = {*wi*| 10, 00, 11, 01} and B = {*vi* | 0, 001, 1, 101}, there exists a Pot Correspondence solution.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | w1 | w2 | w3 | w4 |
| A | 10 | 00 | 11 | 01 |
| B | 0 | 001 | 1 | 101 |

w2w1w3w4 = 00101101 and v2v1v3v4 = 00101101 hence the solution is i = 2, j = 1, k =3, l = 4

Q10. [10] Determine whether the given Boolean expression is satisfiable or not.

If x1 = 0, x2 = 1, x3 = 1 is chosen then thus it is satisfiable.

Q11. [10] Show that L = {*www* | w ∈{*a, b*}+} is in DTIME(*n*). Explain how a Deterministic-TM runs on *w* ∈ L in O(*n*).

Q12. [10] Explain a Halting Problem in detail.

The basis of the halting problem was established by Alan Turing. This basis is, can we write a program or have a machine that will tell us whether any other problem will halt or run forever. So as an example let us say that we have a program let’s call it x. This program will decide whether or not a program will halt. If the program fed into x halts it will loop forever. If it loops forever it will halt. So let’s say we feed x’s outputs into itself. Here is where the contradiction happens. If the output of yes it halts is given to program x then it runs forever which is a contradiction. Let me show this if program x decides that program y will halt it loops forever so it doesn’t halt. It the output from program x is it loops forever then it halts which is contradictory as well. Thus we have the halting problem.

Q13. [10, optional]

1. Explain P-Problem, NP-Problem and NP-Complete Problem, respectively (P/NP/NP-Complete-language, equivalently).
   1. P-Problem is one that is acceptable by a DTM in polynomial time, 𝑃=⋃𝑖≥1DTIME(ni)
   2. NP-Problem is the set of all languages that is accepted by some Non-Deterministic TM in polynomial time, N𝑃=⋃𝑖≥1NTIME(ni)
   3. A problem is NP-Complete if L∈ NP, and for all L1 ∈ NP is polynomial reducible to L: L1≤p L
2. Give one example of NP-Complete problem and define/explain it.
   1. The SAT problem is NP-Complete.

Q14. [20, optional] In the given table of the closure property, mark whether each language family is closed (O) under the given operation or not closed (X) at (1) – (35). E.g.) ∀L1, L2 ∈ REG → L1 ∪ L2 ∈REG?

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **REG** | **DCFL** | **CFL** | **REC** | **RE** |
| **Union** | (1) O | (8) X | (15) O | (22) O | (29) O |
| **Complement** | (2) O | (9) O | (16) X | (23) O | (30) X |
| **Intersection** | (3) O | (10) X | (17) X | (24) O | (31) O |
| **Concatenation** | (4) O | (11) X | (18) O | (25) O | (32) O |
| **Kleene Star (L\*)** | (5) O | (12) X | (19) O | (26) O | (33) O |
| **Reversal (LR)** | (6) O | (13) O | (20) O | (27) O | (34) O |
| **Intersection**  **with REG** | (7) O | (14) O | (21) O | (28) O | (35) O |