Problem Set 4

Due Date .... September 27, 2022
Name .... **Alex Ojemann**
Student ID **109722375**
Collaborators .... **None**

Contents

1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. (See this short intro to LaTeX plus other resources on Canvas.)
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.

- You **must** virtually sign the Honor Code (see Section 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign)

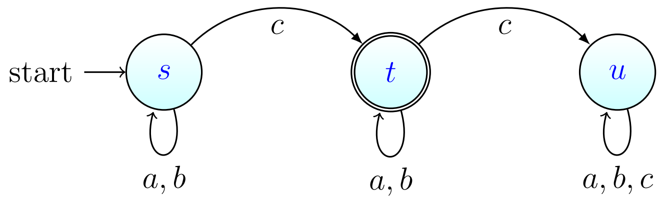
- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

(I agree to the above, Alex Ojemann). □

3 Standard 10: Classification of Regular Languages

Problem 1. For each of the following languages, decide whether the language is regular. For the justification, when showing a language is regular you can demonstrate using the usual constructions (DFA,NFA,regexp) and do not need to prove that your construction works. You could also use closure properties and known regular languages. To justify the claim that a language is non-regular, you can use either the pumping lemma or closure properties.

(a) $L = \{xcy : x, y \in \{a, b\}^*\}$.



Proof. □

(b) $L = \{a^n b^m : n - m \leq 3434\}$.

Proof. Assume L is regular

k given $(k \geq 0)$

$x = \epsilon, y = a^{k+3434}, z = b^k$

$y = uvw, |v| > 0, v = a^{|v|}$

take i = 2:

$xuv^iwz = xuvvwz = a^{k+3434+|v|}b^k$

The number of (a)s minus the number of (b)s is now greater than 3434 so this string is not in the language so this language is non-regular by the pumping lemma □

(c) $L((a^*b)^*a)$.

Proof. $(a^*b)^*a$ (already a regular expression) □

(d) $L = \{a^n b^n c^n : n \geq 0\}$.

Proof. Assume L is regular

k given $(k \geq 0)$

$x = a^k, y = b^k, z = c^k$

$y = uvw, |v| > 0, v = b^{|v|}$

take i = 2:

$xuv^iwz = xuvvwz = a^k b^{k+|v|} c^k$

The number of (b)s is now not equal to the number of (a)s and (c)s so this string is not in the language so this language is non-regular by the pumping lemma □

(e) The set of syntactically correct Python programs. [**Hint:** Think about using an absurdly simple and even nonsensical (but still syntactically correct) program, that only uses a few symbols. A language from PS3 might be relevant here as well.]

Proof. Python programs must have the same number of open and closed parentheses. In PS3, we proved that BRACKET, the language of balanced brackets, is irregular. The proof is as follows.

Assume Bracket is regular.

k given

Let $x =, y = [^k, z =]^k$

$y = uvw$ and $|v| > 0$,so v must contain at least one [

Take i $= 0$:

$xuv^iwz = xuwz = [^{k-|v|}]^k$

$|v| > 0$, so $[^{k-|v|}]^k$ isn't in BRACKET.

Therefore, BRACKET is not regular by the pumping lemma.

Since the language of balanced parentheses behaves exactly the same but with open and closed parentheses instead of brackets, this language is not regular. □

4 Standard 14: Designing CFG

Problem 2. Complete the relevant Automata Tutor problems. You need not submit anything via the PDF for these.

Problem 3. Let $\Sigma = \{0, 1\}$. Let $\overline{x}$ denote the Boolean complement of x; that is, the string obtained from x by changing all 0's to 1's and 1's to 0's. Let $\mathbf{rev}\, x$ denote the reverse of x; that is, the string x written backwards. Consider the set:

$$A = \{x : x = \mathbf{rev}\, \overline{x}\}.$$

For instance, $011001, 010101 \in A$, but $101101 \notin A$. Give a CFG for this set. You need not prove that it generates the set you claim.

Answer. S -> 0 S 1 | 1 S 0 | ϵ $\square$

5 Standard 13: Normal Forms

Problem 4. Complete the requisite Automata Tutor problems. For full credit, please submit here your Greibach grammar. Automata Tutor does not handle Greibach normal form, so it will only test whether the grammar generates the correct language, not whether it is in Greibach normal form.

Answer. Greibach Normal Form for $(da)^*(aa)(bb)^*$:
S -> d A | a B
A -> a S
B -> a C | ϵ
C -> b D
D -> b C | ϵ □

6 Standard 17: CFL Classification

Problem 5. Classify each language as regular (REG), nonregular but context-free (CF), or not context-free (NCF). Justify your answer. (Note: you can show NCF using closure properties; use of the CFL pumping lemma is optional.)

(a) $\{a^k b^\ell a^m b^n : k = m \text{ or } \ell = n\}$

Proof.

$$\begin{aligned} \{a^k b^\ell a^m b^n : k = m\} &= \{a^k b^\ell a^k b^n\} && (1)\\ &= (\{a^k b a^k\} \cup \{a b^\ell a\})(\{b^n\}) && (2)\\ & && (3) \end{aligned}$$

Since $\{a^k b a^k\}$ is non regular but context free and $\{a b^\ell a\}$ and $\{b^n\}$ are regular, $\{a^k b^\ell a^m b^n : k = m\}$ is non regular but context free.

$$\begin{aligned} \{a^k b^\ell a^m b^n : \ell = n\} &= \{a^k b^n a^m b^n\} && (4)\\ &= (\{a^k\})(\{b^n a b^n\} \cup \{b a^m b\}) && (5)\\ & && (6) \end{aligned}$$

Since $\{b^n a b^n\}$ is non regular but context free and $\{b a^m b\}$ and $\{a^k\}$ are regular, $\{a^k b^\ell a^m b^n : \ell = n\}$ is non regular but context free.

Since both of the potential cases of this language ($k = m$ and $\ell = n$) yield languages that are non regular but context free, $\{a^k b^\ell a^m b^n : k = m \text{ or } \ell = n\}$ is non regular but context free. □

(b) $\{a^k b^\ell a^m b^n : k = m \text{ and } \ell = m\}$

Proof. Assume the given language is context-free.

k given (k $\geq$ 0)

Let $z = a^k b^k a^k b^k$ such that $z = uvwxy$, $vx \neq \epsilon$, and $|vwx| < k$.

Take i = 2 :

If either v or x contains both (a)s and (b)s, then uv^2wx^2y wouldn't be of the form $a^*b^*a^*b^*$.

If v and x both only contain (a)s, then uv^2wx^2y would have one a^k larger than the other a^k and thus not be in the language.

If v and x both only contain (b)s, then uv^2wx^2y would have one b^k larger than the other b^k and thus not be in the language.

If v contains only (a)s and x contains only (b)s or vice versa, then uv^2wx^2y would have one a^k larger than the other a^k and one b^k larger than the other b^k and thus not be in the language.

Thus, in any case, $uv^2wx^2y \notin \{a^k b^\ell a^m b^n : k = m \text{ and } l = n\}$. So, $\{a^k b^\ell a^m b^n : k = m \text{ and } l = n\}$ is not context free. □

7 Standards 2/3: Proofs

Problem 6. Prove that the following grammar generates the set of all strings over $\{a, b\}$ with equally many a's and b's.

$$S \to aSb \mid bSa \mid SS \mid \epsilon.$$

Proof. Base Case: S $->$ ϵ
a(ϵ) = b(ϵ) = 0
Inductive Case: Let s_k be a string in this grammar where #a(s_k) = #b(s_k) and there is at least one non terminal state. S is the only non terminal state in this grammar, so it's the only one we need to consider. We will show that evaluating an instance of S in s_k will result in a string s_{k+1} where #a(s_{k+1}) = #b(s_{k+1}).
Case S $->$ aSb: #a(s_{k+1}) = #b(s_{k+1}) = #a(s_k) + 1 = #b(s_k) + 1
Case S $->$ bSa: #a(s_{k+1}) = #b(s_{k+1}) = #a(s_k) + 1 = #b(s_k) + 1
Case S $->$ SS: #a(s_{k+1}) = #b(s_{k+1}) = #a(s_k) = #b(s_k)
Case S $->$ ϵ: #a(s_{k+1}) = #b(s_{k+1}) = #a(s_k) = #b(s_k) □