Monash University
Faculty of Information Technology
2^{nd} Semester 2018

FIT2014
Tutorial 3
Pumping Lemma, and Context Free Languages

Although you may not need to do all the many exercises in this Tutorial Sheet, it is still important that you attempt all the main questions and a selection of the Supplementary Exercises.

ASSESSED PREPARATION: Question 1.
You must present a serious attempt at this entire question to your tutor at the start of your tutorial.

1.

Recall that the EVEN-EVEN *language* consists of all strings over alphabet $\{\mathtt{a}, \mathtt{b}\}$ that contain an even number of `a`s and an even number of `b`s.

The **EVEN-EVEN Game** is played between two players, Nona and Reg, as follows. Firstly, a number N is specified for the players by someone else (e.g., a spectator or referee), or they might agree it between themselves. Then, they take turns, with Nona moving first. The game is very short: Nona moves first, then Reg moves, then Nona moves, making only three moves in total. The rules for their moves are as follows.

- Nona first chooses a string $w \in$ EVEN-EVEN such that $|w| > N$.
- Then Reg divides w up into substrings x, y, z such that $y \neq \varepsilon$ and $|xy| \leq N$.
- Then Nona chooses a non-negative integer i.

The result of the game is that:

- if $xy^iz \in$ EVEN-EVEN, then Reg wins;
- if $xy^iz \notin$ EVEN-EVEN, then Nona wins.

For example, suppose $N = 4$. Here are two possible games between Nona and Reg.

- **First game:**
  1. **Nona:** chooses $w = \mathtt{ababbaab}$.
  2. **Reg:** chooses $x = \varepsilon$, $y = \mathtt{abab}$, $z = \mathtt{baab}$.
  3. **Nona:** chooses $i = 2$.

  Outcome: **Reg wins**, because $xy^iz = xy^2z = \mathtt{abababab baab} \in$ EVEN-EVEN.

- **Second game:**
  1. **Nona:** chooses $w = \mathtt{ababbaab}$.
  2. **Reg:** chooses $x = \mathtt{ab}$, $y = \mathtt{ab}$, $z = \mathtt{baab}$.
  3. **Nona:** chooses $i = 0$.

  Outcome: **Nona wins**, because $xy^iz = xy^0z = \mathtt{abbaab} \notin$ EVEN-EVEN.

(a) Play this game *twice* with one of your fellow FIT2014 students (or a friend or family member, if you prefer), using different moves to those shown in the example games above, and different moves in each game. For the second game, you must reverse the roles you had in the first game. So each player plays one game as Nona, and one as Reg. Record each of the games, showing for each game:

- the name of each player, and which role they each played (Nona/Reg),
- the value of N used,
- the sequence of moves by each player,
- the outcome of the game.

(b) Using quantifiers, write down the assertion that Reg has a winning strategy. (I.e., no matter what first move Nona chooses, Reg can choose a move so that, for any last move by Nona, the outcome is that Reg wins.)

(c) Using quantifiers, write down the assertion that Nona has a winning strategy.

(d) One of the players has a winning strategy, provided $N \geq 4$. Determine who this is, and describe the winning strategy.

2.

The HALF-AND-HALF *language* is defined by

$$\text{HALF-AND-HALF} \;=\; \{\mathtt{a}^n\mathtt{b}^n \;:\; n \in \mathbb{N}\}.$$

The **HALF-AND-HALF Game** is also played between Nona and Reg, exactly as for the EVEN-EVEN game except that the language HALF-AND-HALF is used instead. So, the string w must be chosen to belong to HALF-AND-HALF, and the outcome is determined by whether or not the string xy^iz belongs to HALF-AND-HALF. The rules for the moves are as follows.

- Nona first chooses a string $w \in$ HALF-AND-HALF such that $|w| > N$.
- Then Reg divides w up into substrings x, y, z such that $y \neq \varepsilon$ and $|xy| \leq N$.
- Then Nona chooses a non-negative integer i.

The result of the game is that:

- if $xy^iz \in$ HALF-AND-HALF, then Reg wins;
- if $xy^iz \notin$ HALF-AND-HALF, then Nona wins.

(a) Play this game twice, for practice.

(b) One of the players has a winning strategy, regardless of the choice of N. Determine who this is, and describe the winning strategy.

3. The games EVEN-EVEN and HALF-AND-HALF are instances of a huge class of games called **Pumping Games**. You can play a Pumping Game for *any* language. As usual, the players are Nona and Reg. First, the players are given a language L (which may or may not be regular) and a number N. Nona moves first, then Reg moves, then Nona moves. The rules for their moves are as follows.

- Nona first chooses a string $w \in L$ with $|w| > N$.
- Then Reg divides w up into substrings x, y, z such that $y \neq \varepsilon$ and $|xy| \leq N$.
- Then Nona chooses a non-negative integer i.

The result of the game is that:

- if $xy^iz \in L$, then Reg wins;
- if $xy^iz \notin L$, then Nona wins.

(a) Using quantifiers, write down the assertion that Nona has a winning strategy.

(b) Using quantifiers, write down the assertion that Reg has a winning strategy.

(c) What does the Pumping Lemma tell us about circumstances in which winning strategies exist?

4.

(a) Prove that the difference between two squares of two consecutive positive integers increases as the numbers increase.

(b) Use the Pumping Lemma to prove that the language $\{\mathbf{a}^{n^2} : n \in \mathbb{N}\}$ is not regular.

(c) Hence prove that the language of binary string representations of adjacency matrices of graphs is not regular.

Definitions.

The *adjacency matrix* $A(G)$ of a graph G on n vertices is an $n \times n$ matrix whose rows and columns are indexed by the vertices of G, with the entry for row v and column w being 1 if v and w are adjacent in G, and 0 otherwise.

The *binary string representation* of $A(G)$ is obtained from $A(G)$ by just turning each row of $A(G)$ into a string of n bits, and then concatenating all these strings in row order, to form a string of n^2 bits.

5. Let CENTRAL-ONE be the language of binary strings of odd length whose middle bit is 1.

(a) Prove or disprove: CENTRAL-ONE is regular.

(b) Prove or disprove: CENTRAL-ONE is context-free.

6. Consider the following Context Free Grammar for arithmetic expressions:

$$\begin{aligned} S &\rightarrow E \\ E &\rightarrow E + T \mid E - T \mid T \end{aligned}$$

$$
\begin{array}{rcl}
T & \to & T * F \mid T/F \mid F \\
F & \to & \textbf{int} \mid (E)
\end{array}
$$

where **int** stands for any integer. Find parse trees for each of the following arithmetic expressions.

i) 4 + 6 - 8 + 9

ii) (4 + 10)/(8 - 6)

iii) (2 - 3)*(4 - 8)/(3 - 8)*(2 - 4)

7. In this question, you will write a Context Free Grammar for a very small subset of English.

For many years, children in Victorian schools learned to read from *The Victorian Readers* (Ministry of Education, Victoria, Australia, 1928; many reprints since). Some of you may have grandparents (or even parents!) who used these books in school.

The *First Book* of this series (there were eight altogether) contains simple sentences, with illustrations. Among these sentences are:

I can hop.
I can run.
I can stop.
I am big.
I am six.
I can dig.
I can run and hop and dig.
Tom can hop and dig.
Tom is big.
Tom and I can run.

(a) Write a simple CFG in BNF which can generate these sentences and a variety of others.
(b) Using your grammar, give a derivation and a parse tree for the sentence

Tom and I can dig and hop and run.

8. Consider the following Context Free Grammar.

$$
\begin{array}{rcl}
S & \to & E \\
E & \to & E \cup T \mid T \\
T & \to & TF \mid F \\
F & \to & F^* \mid (E) \mid a \mid b \mid \epsilon
\end{array}
$$

Note: in the last line, ϵ is an actual symbol (i.e., one letter); it is not the empty string itself, but rather a symbol used in regular expressions to match the empty string.

(a) Find the **leftmost** and **rightmost** derivations of the following words generated by the above Context Free Grammar.

i) $a^*b^* \cup b^*a^*$

ii) $(aa \cup bb)^*$

iii) $(a \cup \epsilon)(b \cup \epsilon)(a \cup \epsilon)$

(b) Prove that the language generated by this grammar is not regular.

9. Prove the following theorem, by induction on derivation length.

> If a string in a context-free language has a derivation of length n, then it has a leftmost derivation of length n.

It may help to do this by proving the following stronger result.

> Let σ be a string, which can contain terminal and non-terminal symbols. If a string in a context-free language has a derivation from σ of length n, then it has a leftmost derivation from σ of length n.

10. Find Regular Grammars for the following Regular Expressions:

i) $(a \cup b)^*(aa \cup bb)$

ii) $((a \cup b)(a \cup b))^*$

iii) $(aa \cup bb)^*$

iv) $(ab)^*aa(ba)^*$

v) $(ab \cup ba)(aa \cup bb)^*(ab \cup ba)$

11.

Describe Pushdown Automata for each of the Regular Grammars you found in the previous question.

12.

(a) Find a Context-Free Grammar for the language

$$\{\mathbf{a}^n\mathbf{b}^i\mathbf{c}^{2n} : i, n \in \mathbb{N}\}.$$

(b) Prove, by induction on the string length, that every string of the form $\mathbf{a}^n\mathbf{b}^i\mathbf{c}^{2n}$ can be generated by your grammar.

(c) Find a Pushdown Automaton that recognises this language.

13.

Let k be a fixed positive integer. A *k-limited Pushdown Automaton* is a Pushdown Automaton whose stack can store at most k symbols (regardless of the length of the input string).

(a) Explain how a k-limited Pushdown Automaton can be simulated by a Nondeterministic Finite Automaton.

(b) Deduce that a language is recognised by a k-limited Pushdown Automaton if and only if it is regular.

Supplementary exercises

14. Prove or disprove:

> In a Context-Free Grammar, if the right-hand side of a production is a palindrome, then any string in the generated language is a palindrome.

15. Recall that $\overleftarrow{x}$ denotes the reverse of string x. If L is any language, define $\overleftarrow{L} = \{x : \overleftarrow{x} \in L\}$, i.e., the set of all reversals of strings in x.

Prove that, if L is a Context-Free Language, then so is $\overleftarrow{L}$.

16. Use Questions 6 and 12 to prove that, if a string in a context-free language has a derivation of length n, then it has a rightmost derivation of length n.

17. For our purposes, a *permutation* is represented as a string, over the three-symbol alphabet containing 0, 1 and "," (comma), as follows: each of the numbers $1, 2, \ldots, n$ is represented in binary, and the numbers are given in some order in a comma-separated list. Each number in $1, 2, \ldots, n$ appears exactly once, in binary, in this list. For example, the permutation 3,1,2 is represented by the string "11,1,10".

Prove that the language of permutations is not regular.

18.

For this question, you may use the fact that there exist arbitrarily long sequences of positive integers that are not prime. In other words, for any N, there is a sequence of numbers x, $x+1$, $\ldots$, $x+N$ none of which is prime.[1]

Prove that the language $\{\mathbf{a}^p : p \text{ is prime}\}$ is not regular.

19. You saw in Q5 that the language of regular expressions, over the seven-character alphabet $\{$a,b,ε,$\cup$,(,),$*\}$, is not, itself, regular(!), but it is context-free.

Now, prove that the language of context-free grammars is regular. Assume that the non-terminal symbols are S, $X_1, \ldots X_m$, where S is the start symbol, and that the terminal symbols are $x_1, \ldots, x_n$. So the alphabet for your regular expression is $\{S, X_1, \ldots X_m, x_1, \ldots, x_n, \rightarrow\}$.

But don't get confused about this! It certainly does *not* follow that every context-free language is regular! Remind yourself of the actual relationship between regular languages and context-free languages.

[1] Mathematically-inclined students are encouraged to try to prove this fact for themselves.