CLEMSON UNIVERSITY, SCHOOL OF COMPUTING
CPSC 3500 FOUNDATIONS OF COMPUTER SCIENCE

Assignment 1: Languages, Induction, Recursion

Return by 11:59pm 1/24/2019

Problem 1 5%

The language *Balanced* over $\Sigma = \{(,)\}$ is defined recursively as follows

1. $\Lambda \in$ *Balanced*.
2. $\forall x, y \in$ *Balanced*, both xy and (x) are elements of *Balanced*.

A prefix of a string x is a substring of x that occurs at the beginning of x. Prove by induction that a string x belongs to this language if and only if (iff) the statement $B(x)$ is true.
$B(x)$: x contains equal numbers of left and right parentheses, and no prefix of x contains more right than left.
Reminder for this and all following assignments: if you need to prove the "iff" statement, i.e., $X \iff Y$, you need to prove both directions, namely, "given X, prove that Y follows from X ($X \implies Y$)", and "given Y, prove that X follows from Y ($X \impliedby Y$)".

Problem 2 5%

Complete proof of claim about the reverse function (see Lecture 3).

Problem 3 5%

Finite language is a language with finite number of strings in it, i.e., there exist exactly k strings in this language such that $k \in \mathbb{N}$ and $k \neq \infty$. For a finite language L, let $|L|$ denote the number of elements of L. For example, $|\{\Lambda, a, ababb\}| = 3$. (Do not mix up with the length $|x|$ of a string x.) The statement $|L_1 L_2| = |L_1||L_2|$ says that the number of strings in the concatenation $L_1 L_2$ is the same as the product of the two numbers $|L_1|$ and $|L_2|$. Is this always true? If so, prove, and if not, find two finite languages $L_1, L_2 \subseteq \{a, b\}^*$ such that $|L_1 L_2| \neq |L_1||L_2|$.

Problem 4 5%

We proved in class that if L_1, and L_2 are subsets of $\{a,b\}^*$ then $L_1^* \cup L_2^* \subseteq (L_1 \cup L_2)^*$. Show that $L_1^* \cup L_2^* \neq (L_1 \cup L_2)^*$.

Problem 5 5%

Find an example of languages L_1 and L_2 for which neither of L_1, L_2 is a subset of the other, but $L_1^* \cup L_2^* = (L_1 \cup L_2)^*$. Prove the correctness of your example.

Problem 6 10%

Given language $L = \{yy | y \in \{a,b\}^*\}$. L can be represented as a concatenation

$$L = L\{\Lambda\} = \{\Lambda\}L$$

like any language. Can you express L as $L = L_1 L_2$, where $L_1 \neq \{\Lambda\}$, and $L_2 \neq \{\Lambda\}$? Prove your answer.

Problem 7 5%

Each case below gives a recursive definition of $L \subseteq \{a,b\}^*$. Give a simple nonrecursive definiton of L in each case. Example: $a \in L$; $\forall x \in L \quad ax \in L$ can be defined as "The set of all non-empty strings that do not contain b."

1. $a \in L$; $\forall x \in L \quad xa, xb \in L$
2. $a \in L$; $\forall x \in L \quad bx, xb \in L$
3. $a \in L$; $\forall x \in L \quad ax, xb \in L$
4. $a \in L$; $\forall x \in L \quad xb, xa, bx \in L$
5. $a \in L$; $\forall x \in L \quad xb, ax, bx \in L$
6. $a \in L$; $\forall x \in L \quad xb, xba \in L$

Problem 8 5%

Suppose that Σ is an alphabet, and that $f : \Sigma^* \to \Sigma^*$ has the property that $f(\sigma) = \sigma$ for every $\sigma \in \Sigma$ and $f(xy) = f(x)f(y)$ for every $x, y \in \Sigma^*$. Prove that for every $x \in \Sigma^*$, $f(x) = x$.

Problem 9 15%

In each case below, find a recursive definition for the language L and prove that it is correct.

1. $L = \{a^i b^j | j \geq 2i\}$
2. $L = \{a^i b^j | j \leq 2i\}$

Problem 10 10%

Suppose $L \subseteq \{a,b\}^*$ is defined as follows: $\Lambda \in L$; for every x and y in L, the strings axb, bxa, and xy are in L. Show that $L = AEqB$, the language of all strings x in $\{a,b\}^*$ satisfying $n_a(x) = n_b(x)$.

Problem 11 10%

Let L_1, L_2, and L_3 be languages over some alphabet. In each case below, two languages are given. Say what the relationship is between them. (Are they always equal? If not, is one always a subset of the other?) Give reasons for your answers, including counterexamples if appropriate.

1. $L_1(L_2 \cap L_3)$, $L_1L_2 \cap L_1L_3$
2. $L_1^* \cap L_2^*$, $(L_1 \cap L_2)^*$
3. $L_1^*L_2^*$, $(L_1L_2)^*$

Problem 12 10%

For $x \in EXPR$ defined in class, $n_a(x)$ denotes the number of a's in the string, and we will use $n_{op}(x)$ to stand for the number of operators in x (the number of occurrences of $+$ or $*$). Show that for every $x \in EXPR$, $n_a(x) = 1 + n_{op}(x)$.

Problem 13 10%

Show using induction that for every $x \in \{a,b\}^*$ such that x begins with a and ends with b, x contains the substring ab.

Solutions to Assignment 1

Solution to Problem 1
See solution in the textbook.
Solution to Problem 2
See solution in the slides. Use b instead of a.
Solution to Problem 3
The statement is not true. Example: $L_1 = \{\Lambda, a, b\}$, and $L_2 = \{\Lambda, a\}$, then $L_1 L_2 = \{\Lambda, a, b, aa, ba\}$, i.e., $|L_1 L_2| = 5$, whereas $|L_1||L_2| = 6$.
Solution to Problem 4
$L_1 = \{\Lambda, a\}$, $L_1 = \{\Lambda, b\}$. String aba is in $(L_1 \cup L_2)^*$ but not in $L_1^* \cup L_2^*$.
Solution to Problem 5
Example: $L_1 = \{aa, aaaaa\}$, and $L_2 = \{aaa, aaaaa\}$.
Solution to Problem 6
See solution in the textbook.
Solution to Problem 7

1. The set of all strings beginning with a.
2. The set of all strings containing exactly one a.
3. The set of all strings containing at least one a, and no ba substring.
4. The set of all strings containing at least one a.
5. The set of all strings containing at least one a.
6. The set of all strings starting with a, with no substring aa.

Solution to Problem 8
Proof by structural induction

- basis: Show for Λ. $f(\Lambda) = f(\Lambda\Lambda) = f(\Lambda)f(\Lambda)$, i.e., we have a formula of form $z = zz$, where $z = f(\Lambda)$, which can be true for $z = \Lambda$ only.
- induction hypothesis: true for string x, $f(x) = x$
- induction step: $\forall a \in \Sigma$ $f(xa) = f(x)f(a)$ by assumption. By IH $f(x) = x$, and by assumption $f(a) = a$, then $f(xa) = xa$.

Solution to Problem 9

1. $\Lambda \in L$; $\forall x \in L$, $axbb \in L$, and $xb \in L$.
2. $\Lambda \in L$; $\forall x \in L$, $axbb \in L$, and $axb \in L$, and $ax \in L$.

   Let us prove (2) as an example. Let $L_0 = \{a^i b^j | j \leq 2i\}$.

   First we use structural induction to show that $L \subseteq L_0$. Clearly $\Lambda \in L_0$. Suppose that $a^i b^j \in L_0$ (by inductive hypothesis), which means that $j \leq 2i$. Let's verify if all possible rules that extend this string lead us to L_0. It follows that $j + 2 \leq 2(i + 1)$, which means that $a(a^i b^j)bb \in L_0$. It also follows that $j + 1 \leq 2(i + 1)$, so that $a(a^i b^j)b \in L_0$. Finally, it is also true that $j \leq 2(i + 1)$, so that $a(a^i b^j) \in L_0$. Therefore, $L \subseteq L_0$.

Now we use induction on $|x|$ to show that if $x \in L_0$, then $x \in L$. The basis step is straightforward. Suppose that $k \geq 0$ and that for any i and j in $\mathbb{N}$ satisfying $i + j \leq k$, and $j \leq 2i$, $a^i b^j \in L$. Now suppose that $i + j = k + 1$ and $j \leq 2i$. We must show that $a^i b^j \in L$. We consider three cases. First, if $j = 2i$, then $j - 2 = 2(i-1)$, which implies that $a^{i-1}b^{j-2} \in L$. By the induction hypothesis, $a^{i-1}b^{j-2} \in L$; therefore, $a^i b^j = a(a^{i-1}b^{j-2})bb \in L$, by the definition of L. Second, if $j = 2i - 1$, then $j - 1 = 2i - 2$, which means that $a^{i-1}b^{j-1} = a^{i-1}b^{2(i-1)}$ and therefore that $a^{i-1}b^{j-1} \in L_0$. Therefore, by the induction hypothesis, $a^{i-1}b^{j-1} \in L$, and by the recursive definition of L, $a^i b^j = a(a^{i-1}b^{j-1})b \in L$. Finally, if $j < 2i-1$, then $j \leq 2(i-1)$, which means that $a^{i-1}b^j \in L_0$. By the induction hypothesis, $a^{i-l}b^j \in L$; therefore, by the recursive definition of L, $a^i b^j = a(a^{i-1}b^j) \in L$.

Solution to Problem 10
See solution in the textbook.
Solution to Problem 11

1. $L_1(L_2 \cap L_3) \subseteq L_1L_2 \cap L_1L_3$. If $L_1 = \{a, aa\}, L_2 = \{\Lambda\}, L_3 = \{a\}$ then $L_1L_2 \cap L_1L_3 = \{aa\}$, and $L_1(L_2 \cap L_3) = \emptyset$.

2. $(L_1 \cap L2)^* \subseteq L_1^* \cap L_2^*$, and they are not always equal. If $L_1 = \{a\}$, and $L_2 = \{aa\}$, then $L_1^* \cap L_2^*$ contains aa, and $(L_1 \cap L_2)^* = \emptyset$.

3. $L_1^*L_2^*$, $(L_1L_2)^*$. Neither is necessarily a subset of the other. If $L_1 = \{a\}$, and $L_2 = \{b\}$, then $aabb \in L_1^*L_2^* - (L_1L_2)^*$, and $abab \in (L_1L_2)^* - L_1^*L_2^*$.

Solution to Problem 12
Use structural induction on the recursive definition of EXPR. We need to prove the statement $P(x)$ that is $n_a(x) = n_{op}(x) + 1$.

- Basis step: if $x = a$, $P(x)$ is true because $n_a(a) = n_{op}(a) + 1 = 0 + 1$.

- Induction hypothesis: we assume that $P(x)$, and $P(y)$ are true, i.e., $n_a(x) = n_{op}(x) + 1$, and $n_a(y) = n_{op}(y) + 1$.

- Induction step: We need to prove $P(z)$ for three cases $z = x + y$, $z = x * y$, and $z = (x)$. Let us prove it for $z = x + y$.

$$\begin{aligned} n_a(z) &= n_a(x) + n_a(y) && \text{(because there is no } a \text{ in +)} \\ &= n_{op}(x) + 1 + n_{op}(y) + 1 && \text{(by induction hypothesis)} \\ &= n_{op}(z) + 1 && \text{(because } n_{op}(z) = n_{op}(x) + n_{op}(y) + 1\text{)} \end{aligned}$$

Other cases are similar.
Solution to Problem 13
We prove the statement: $\forall n \in \mathbb{N}_{\geq 2}$, if $x \in \{a, b\}^*$, $|x| = n$, and x begins with a, ends with b, then x contains ab.

- BS: $n = 2$, the statement is true because if $|x| = 2$, starts with a, ends with b then $x = ab$.

- IH: the statement is true for some k

- IS: We need to prove it for $|x| = k + 1$. If the second symbol of x is b then x contains ab and the statement is correct; otherwise $x = ay$, where $|y| = k$, and y begins with a, ends with b. By IH, y contains ab, and therefore x also does.