CMPS 130 - Spring 2016

Homework 2

Seongwoo Choi

SID: 1368039

Patrick Tantalo

**Problem 31:**

Show that for every language L, LL\* = L\* if and only if Λ ∈L.

In order to show that for every language L, we need to show this first. Λ ∈L\*, and if A = xy, then x = y = Λ. Therefore, this means that if LL\* = L, Λ ∈L. On the other hand, if Λ ∈L, then every element x of L\* is the concatenation of a string in L and x ∈L\*.

**Problem 33:**

Let L1 and L2 be subsets of {a, b}\*.

a) Show that if L1 ⊆ L2, then L1\* ⊆ L2\*

Consider that L1⊆ L2, which means that every member formed using L­1 is present a member of L2. We are assuming that W1 be a string that belongs to L1. Here, we can see that L1\* be the language that satisfies over the alphabet set containing any number of W1’s. Thus, W1\* is the string that is present in L1\*. Assume W1 be a string that also belongs to L2. This is because that L1⊆L2. Now, L2\* language that satisfies over the alphabet set containing any number of W1’s. So, W1\* is the string present in L2\*. So, the string W1\*, is present in both L1\* and also in L2\*. This means that L1\* and also in L2\*. This means that L1\* is a subset of L2\*. Thus, L1\* ⊆ L2\* only if L1⊆L2.

b) Show that L1\* ∪ L2\* ⊆ (L1 ∪ L2)\*.

Proof:

We are going to prove this by using these:

Let x be the string that satisfies language L1 and let y be the string that satisfies language L2.

Now, there is L1\*. It is the language that satisfies the alphabet set containing any number of x’s. Similar to that, L2\* is the language that satisfies the alphabet set containing any number of y’s.

L1\* ∪ L2\* defines the language that consists of the strings starting and ending with any number of x’s or y’s. It is either L1\* or L2\*.

L1\*∪L2\* defines the language that consists of the strings starting and ending with x or y. If we apply a star operation, then it becomes (L1∪L2)\*, which means that the strings starting and ending with any number of x’s or y’s or a combination of both.

Therefore, we can say that every element of L1\*∪L2\* is present in (L1∪L2)\*. However, L1\* ∪ L2\* does not contain all elements of (L1∪L2)\*. So, L1\*∪L2\* is subset or equal to (L1∪L2)\*.

Thus, L1\*∪L2\* ⊆ (L1∪L2)\*.

c) Give an example of two languages L1 and L2 such that L1\* ∪ L2 ≠ (L1 ∪ L2)\*.

Let the languages be represented as follows:

L1 = {a}

L2 = {b}

Now, when we apply star operations, then they become:

L1\* = {Λ, a, aa, aaa, aaaa, ……}

L2\* = {Λ, b, bb, bbb, bbbb,……}

The union of L1\* and L2\*, that is L1\*∪L2\* either L1\* or L2\*.

The union of L1 and L2 is as follows:

L1∪L2 = {a∪b}

(L1∪L2)\* = {a∪b}\*

= {a, b}\*

= {Λ, a, b, aa, ab, ba, bb, aaa, aab, ……}

So, L1\*∪L2\* does not contain all elements in (L1∪L2)\*.

Thus, L1\*∪L2\* ≠ (L1∪L2)\*

**Problem 36:**

1. Consider the language L of all strings of a’s and b’s that do not end with b and do not contain the substring *bb*. Find a finite language S such that L = S\*.

L = {a, ba}

1. Show that there is no language S such that S\* is the language of all strings of a’s and b’s that do not contain the substring *bb*.

If there were such an S, then b would be an element of S\* because it does not contain the substring bb and therefore bb would be because the concatenation of two elements of S\* is an element of S\*. However, this is impossible to do so.

**Problem 44:**

Each case below gives, a recursive definition of a subset L of {a, b}\*. Give a simple non-recursive definition of L in each case.

1. a ∈ L; for any x ∈L, *xa* and *xb* are in L.

The set of all strings beginning with a.

1. a ∈ L; for any x ∈L, *bx* and *xb* are in L.

The set of all strings containing exactly one a.

1. a ∈ L; for any x ∈L, *ax* and *xb* are in L.

It can also be described as the set of all strings containing at least one a in which all the a’s precede all the b’s.

D) a ∈ L; for any x ∈L, *xb, xa*, and *bx* are in L.

The set of all strings containing at least one a.

**Problem 52:**

Prove that for every language L ⊆ {a, b}\*, if L2⊆ L, then LL\* ⊆ L.

The language L\* can be defined recursively by saying that Λ∈L\*, and that for every x∈L\* and every y ∈L, xy∈L\*. We can formulate the statement in the exercise as follows: For every x ∈L\*, if z is any string in L, then zx ∈L. Now we can use structural induction to show that this statement is true.

The basis step:

This is to show that if z is any string in L, then zΛ∈L, and this is true.

The induction hypothesis:

This is that for some string x∈L\*, if z ∈L, then zx ∈L. In the induction step we need to show that for every y ∈L, z(xy) ∈L. This is true because z(xy) = (zx)y, zx is assumed to be in L, and the assumption is that LL⊆L.

**Problem 63:**

For a string x in the language Expr defined in Example 1.19, na(x) denotes the number of a’s in the string, and we will use nop(x) to stand for the number of operators in x (the number of occurrences of + or \*). Show that for every x ∈Expr,na(x) = 1 + nop(x).

We can prove this by using structural induction based in the recursive definition of Expr.

**Problem 65:**

Suppose L ⊆{a, b}\* is defined as follows:

Λ∈L; for every x ∈L, both

xa and xba are in L.

Show that for every x ∈L, both of the following statements are true.

1. na(x) ≥ nb(x).

This is a simple proof by using structural induction.

b) x does not contain the substring *bb*.

It is easiest to prove that for every x∈L, x does not end with b and does not contain the substring bb. The proof is by structural induction. Λ neither ends with b nor contains the substring bb. Suppose x∈L and x does not end with b or contain the substring bb. Then clearly xa doesn’t end with b or contain bb. The string xba certainly does not end with b; because x does not contain bb, the only way xba could contain this substring would be for x to end with b, and the assumption is that it does not.

**Problem 66:**

Suppose L {a, b}\* is defined as follows: Λ ∈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 na(x) = nb(x).

The proof that L⊆ AEqB is a trivial structural induction argument. If we prove this in the other direction by strong induction on the length of the string. Every string of length 0 in AEqB is in L. Suppose that every string in AEqB of length k or less is in L, and now assume x∈AEqB and |x| = k + 1. The length must be even and must therefore be 2 or more. If the first symbol and the last symbol of x are different, then either x = ayb or x = bya for some string y. In either case, because x is an element of AEqB, the string y must also be in this language. Therefore, by the induction hypothesis, y∈L; therefore, by the recursive definition of L, x∈L. In the other case, the first and last symbols of x are the same. Suppose for example that x =aya for some string y. We consider, for each prefix z of x, the number d(z) = na(z) – nb(z). The shortest non null prefix of x is a, and d(a) = 1. The longest, prefix of x other than x is ay, and d(a.y) = - 1, since d(x) = 0. If z1 is a prefix obtained from the prefix z0 by adding a single symbol, d(z1 differs from d(z0) by 1. In the other words, if we start with a and consider longer and longer prefixes of x, the d value starts at 1, changes by 1 at each step, and eventually has the value -1. Therefore, there must be some prefix z, longer than a and shorter than ay, for which d(z) = 0. This means that z ∈A EqB, and if x =zw, that w ∈AEqB as well. Therefore, by the induction hypothesis, both z and w are in L. Therefore, x =zw∈L, according to the recursive definition of L. The proof in the case when x begins and ends with b is almost identical.