Tyler Holland

**Homework 5**

**7)**

**a)**

Let L2 be the set of strings over that end with aa. L2 is regular since it is expressed by a regular expression: (a U b U c)\*aa

L1 (the language in question) can be represented as L1 = L ∩ L2

Since L is regular (given) and L2 is regular (see above) then according to closure properties their intersection

∴ L1 is regular.

**b)**

Let L2 be the set of strings over that contain a. L2 is regular since it is expressed by a regular expression: (a U b U c)\*a(a U b U c)\*

L1 (the language in question) can be represented as L1 = L ∩ L2

Since L is regular (given) and L2 is regular (see above) then according to closure properties their intersection

∴ L1 is regular.

**c)**

Let L1 be the set of strings over that contain a. L1 is regular since it is expressed by a regular expression: (a U b U c)\*a(a U b U c)\* \_\_

L1 (the language in question) can be represented as L1 = \*- L1

Since L is regular (given) and L1 is regular (see above) then according to closure properties their intersection

∴ L1 is regular.

**d)**

Let L2 be the set of strings over that contain a. L2 is regular since it is expressed by a regular expression: (a U b U c)\*a(a U b U c)\* \_\_

L1 (the language in question) can be represented as L1 = \*- L1

Since L is regular (given) and L1 is regular (see above) then according to closure properties their intersection

∴ L1 is regular.

**14)**

**b)**

Proof

Suppose, to the contrary, that L is regular. Then there is a DFA that accepts L. Let *k* be the number of states in that DFA.

Let’s take the string z = akbk+1

Note that z is a string from L, and *length(z) ≥ k*. Therefore, according to PL, z contains a nonempty pumpable substring v in its prefix of length k.

Since the first *k* letters of z are a’s, the substring v may contain a’s only. This, z can be written as z = uvw, where:

u = an, n ≥ 0

v = am, m > 0

w = ak-n-mbk+1

Now let’s pump the v substring: let’s take z’ = uv3w = anamamamak-n-mbk+1 = ak+2mbk+1

z’ ~~~~ L, because #a’s is greater than #b’s since m > 0

However, according to PL, z’ should’ve belonged to L since it is a result of pumping. We got a contradiction which proves that the supposition of L being regular is false.

∴ L is not regular.

**c)**

Proof

Suppose, to the contrary, that L is regular. Then there is a DFA that accepts L. Let *k* be the number of states in that DFA.

Let’s take the string z = bkc2k

Note that z is a string from L, and *length(z) ≥ k*. Therefore, according to PL, z contains a nonempty pumpable substring v in its prefix of length k.

Since the first *k* letters of z are b’s, the substring v may contain b’s only. This, z can be written as z = uvw, where:

u = bn, n ≥ 0

v = bm, m > 0

w = bk-n-mc2k

Now let’s pump the v substring: let’s take z’ = uv3w = bnbmbmbmbk-n-mc2k = ak+2mb2k

z’ ~~~~ L, because the ratio between the #b’s and #c’s is not right since m>0 2(k+2m) ≠ 2k

However, according to PL, z’ should’ve belonged to L since it is a result of pumping. We got a contradiction which proves that the supposition of L being regular is false.

∴ L is not regular.

**d)**

Proof

Suppose, to the contrary, that L is regular. Then there is a DFA that accepts L. Let *k* be the number of states in that DFA.

Let’s take the string z = akbk

Note that z is a string from L, and *length(z) ≥ k*. Therefore, according to PL, z contains a nonempty pumpable substring v in its prefix of length k.

Since the first *k* letters of z are a’s, the substring v may contain a’s only. This, z can be written as z = uvw, where:

u = an, n ≥ 0

v = am, m > 0

w = ak-n-mbk

Now let’s pump the v substring: let’s take z’ = uv3w = anamamamak-n-mbk = ak+2mbk

z’ ~~~~ L, because the ratio between the #a’s and #b’s is not right since m>0 k+2m ≠ k

However, according to PL, z’ should’ve belonged to L since it is a result of pumping. We got a contradiction which proves that the supposition of L being regular is false.

∴ L is not regular.

**f)**

Proof

Suppose, to the contrary, that L is regular. Then there is a DFA that accepts L. Let *k* be the number of states in that DFA.

By the PL, every string z  L of length k or more can be decomposed into substrings u, v, and w such that length(uv) ≤ k, v ≠λ, and uviw  L for all i ≥ 0.

Consider the string z = ak^3 of length k3. Since z is in L and its length is greater than k, z can be written z = uvw where the u, v, and w satisfy the conditions of the pumping lemma.

In particular, 0 < length(v) ≤ k. This observation can be used to place an upper bound on the length of uv3w:

Length(uv3w) = length(uvw) + length(v) + length(v)

= k3 + length(v) + length(v)

≤ k3 + k + k

< k3 + 2k + 1

= (k + 1)3

The length of uv3w is greater than k3 and less than (k+1)3 and therefore is not a perfect cube. Thus the string uv3w obtained by pumping v once is not in L. We have shown that there is no decomposition of z that satisfies the conditions of the pumping lemma. The assumption that L is regular leads to a contradiction.

∴ L is not regular.

**15)**

Proof

Suppose, to the contrary, that L is regular. Then there is a DFA that accepts L. Let *k* be the number of states in that DFA.

Let’s take the string z = akbak

Note that z is a string from L, and *length(z) ≥ k*. Therefore, according to PL, z contains a nonempty pumpable substring v in its prefix of length k.

Since the first *k* letters of z are a’s, the substring v may contain a’s only. This, z can be written as z = uvw, where:

u = an, n ≥ 0

v = am, m > 0

w = ak-n-mbak

Now let’s pump the v substring: let’s take z’ = uv2w = anamamak-n-mbak = ak+mbak

z’ ~~~~ L, because it has more a’s in front than in back (since m>0) and thus is a palindrome.

However, according to PL, z’ should’ve belonged to L since it is a result of pumping. We got a contradiction which proves that the supposition of L being regular is false.

∴ L is not regular.

**18)**