CS 390 Solutions to Test 1

October 5, 2004

1. Find a string of minimum length in $\{0, 1\}$ that is **NOT** in the language corresponding to the regular expression $1^*(0 + 10)^*1^*$.

0110

2. Describe in English as simply as possible the language corresponding to each of the following regular expressions:

(a) $(00 + 01 + 11)^+$

The set of non-empty strings of even length which do not have 10 at odd positions.

(b) $(00 + 01 + 10 + 11)^*$

The set of strings of even length.

3. Simplify the regular expression $(10 + 1 + 0^*1^*)^*$.

(0+1). Note that 0^*1^* produces 0 which together with 1 generates all possible strings over the alphabet $\{0, 1\}$ by $*$.

4. Find a regular expression for each of the following languages over the alphabet $\{0, 1\}$:

(a) The set of strings with an even number of 0's.

$(1 + 01^*0)^*$

(b) The language L defined recursively as follows:

Basis Clause: $\Lambda \in L$
Inductive Clause: If $x \in L$ then $01x, x0, x1 \in L$
Extremal Clause: Nothing is in L unless it is obtained from the above two clauses.

$(0+1)^*$. Note that x0 and x1 generate all strings over the alphabet $\{0,1\}$. Hence 01x is redundant.

5. Express the string $((ab)^r(baa)^r(bbab)^r)^r$ over the alphabet $\{a,b\}$ without using r, where w^r denotes the reversal of w.

$bbabbaaab$. Note that $(x^r y^r z^r)^r = (z^r)^r (y^r)^r (x^r)^r = zyx$.

6. Prove that for a language L, $(L^*)^* = L^*$.

We prove this in two steps. First we prove $L^* \subseteq (L^*)^*$, then $(L^*)^* \subseteq L^*$.

(1) $L^* \subseteq (L^*)^*$

Since $(L^*)^* = (L^*)^0 \cup L^* \cup (L^*)^2 \cup ...$, $L^* \subseteq (L^*)^*$.

(2) $(L^*)^* \subseteq L^*$

Let $w \in (L^*)^*$. Then there exist strings $w_1, w_2, ..., w_k$ such that $w_i \in L^*$ for $i = 1, 2, ..., k$ and $w = w_1 w_2 ... w_k$.
However, since $w_i \in L^*$ for $i = 1, 2, ..., k$, for each w_i, there exist strings $w_{ij} \in L$ such that $w_i = w_{i1} w_{i2} ... w_{im_i}$.
Hence $w = w_{11} w_{12} ... w_{1m_1} w_{21} w_{22} ... w_{2m_2} ... w_{k1} w_{k2} ... w_{km_k}$. Hence $w \in L^*$.

$(L^*)^* \subseteq L^*$ can also be proven by structural induction on strings of $(L^*)^*$. For that we need the fact that if $x \in L^*$ and $y \in L^*$, then $xy \in L^*$.

Lemma: If $x \in L^*$ and $y \in L^*$, then $xy \in L^*$.
Proof of Lemma by induction on y:
Basis Step: If $y = \Lambda$, then $xy = x\Lambda = x$. Hence if $x \in L^*$ then $xy \in L^*$.
Inductive Step: Suppose that $xy \in L^*$ for any $y \in L^*$.
We will show that $xyz \in L^*$ for any $z \in L$.
Since $xyz = (xy)z$ and $xy \in L^*$, by the definition of L^* (see below), $(xy)z \in L^*$.

Recursive definition of L^*

Basis Clause: $\Lambda \in L^*$.
Inductive Clause: For every $x \in L^*$ and every $y \in L$, $xy \in L^*$.
Extremal Clause: Nothing is in L^* unless it is obtained from the above two clauses.

Proof by structural induction for $(L^*)^* \subseteq L^*$.

Basis Step: By the definition of L^* $\Lambda \in L^*$.
Inductive Step: Suppose that for an arbitrary string x, if $x \in (L^*)^*$, then $x \in L^*$.
Then for an arbitrary $y \in L^*$, consider the child xy of x in $(L^*)^*$. We need to show that $xy \in L^*$.
Since $x \in L^*$ and $y \in L^*$, by the lemma, $xy \in L^*$.