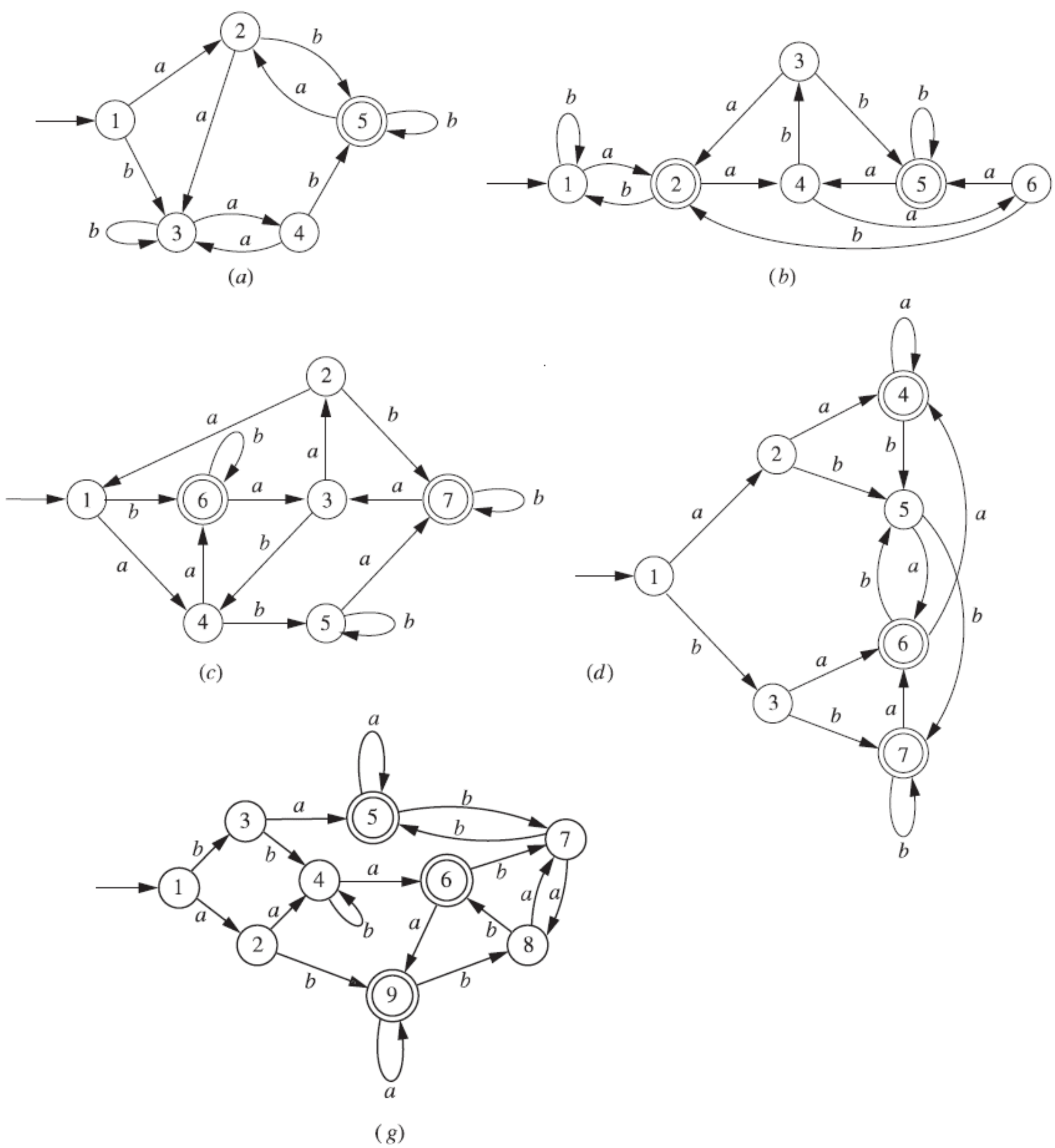
# CMPS 130
# Spring 2016
# Homework Assignment 5

Problems are from Martin 4th edition.

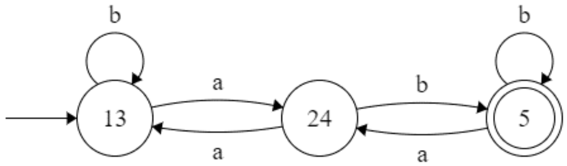
Chapter 2 (p.77): 55abcdg, 57bdgh
Chapter 3 (p.117): 1abcd, 2abcd, 3abc, 4, 7cijm, 9

1. Problem 2.55abcdg
   For each of the FAs pictured in Fig. 2.45, use the minimization algorithm described in Section 2.6 to find a minimum-state FA recognizing the same language. (It's possible that the fiven FA may already be minimal.)



**Solution:**

a.

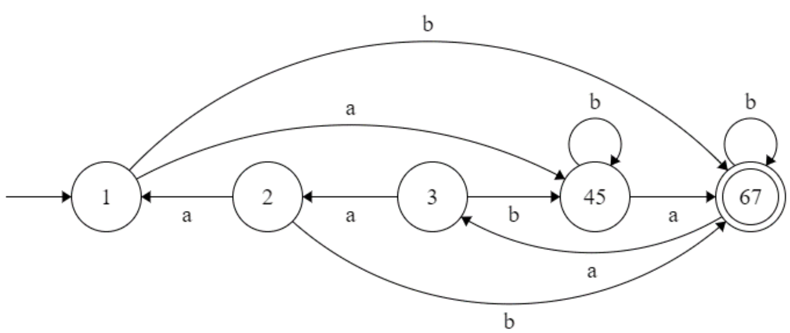



| | | | | |
|---|---|---|---|---|
| 2 | × | | | |
| 3 | | × | | |
| 4 | × | | × | |
| 5 | × | × | × | × |
| | 1 | 2 | 3 | 4 |

b. Already minimal

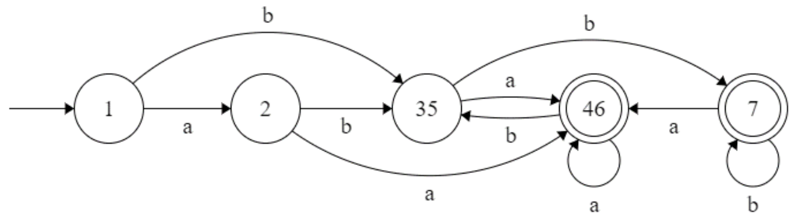
| | | | | | |
|---|---|---|---|---|---|
| 2 | × | | | | |
| 3 | × | × | | | |
| 4 | × | × | × | | |
| 5 | × | × | × | × | |
| 6 | × | × | × | × | × |
| | 1 | 2 | 3 | 4 | 5 |

c.



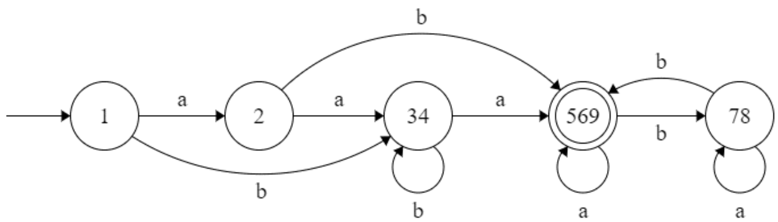

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | × | | | | | |
| 3 | × | × | | | | |
| 4 | × | × | × | | | |
| 5 | × | × | × | | | |
| 6 | × | × | × | × | × | |
| 7 | × | × | × | × | × | |
| | 1 | 2 | 3 | 4 | 5 | 6 |

d.




| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | × | | | | | |
| 3 | × | × | | | | |
| 4 | × | × | × | | | |
| 5 | × | × | | × | | |
| 6 | × | × | × | | × | |
| 7 | × | × | × | × | × | × |
| | 1 | 2 | 3 | 4 | 5 | 6 |

g.

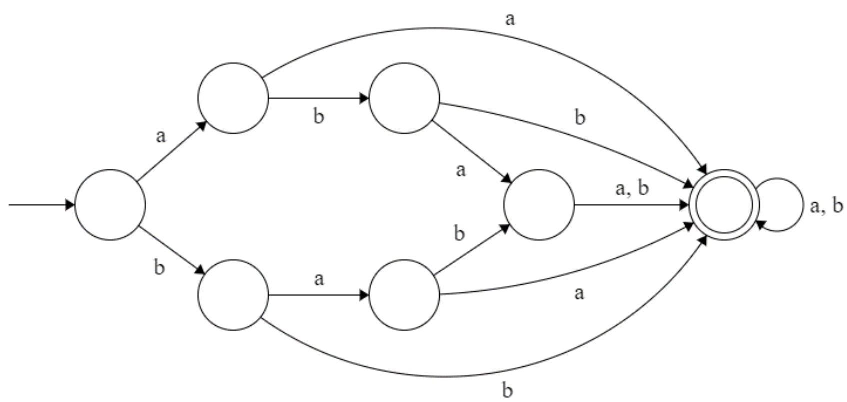



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2 | × | | | | | | | |
| 3 | × | × | | | | | | |
| 4 | × | × | | | | | | |
| 5 | × | × | × | × | | | | |
| 6 | × | × | × | × | | | | |
| 7 | × | × | × | × | × | × | | |
| 8 | × | × | × | × | × | × | | |
| 9 | × | × | × | × | | | × | × |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

2. Problem 2.57bdgh
   Each case below defines a language over $\{a, b\}$. In each case, decide whether the language can be accepted by an FA, and prove that your answer is correct.
   b. The set of all strings containing some non-null string of the form *ww*.
   d. The set of odd-length strings with middle symbol *a*.
   g. The set of non-palindromes.
   h. The set of strings in which the number of *a*'s is a perfect square.

   **Solution:**
   b. $L = \{\, x \in \{a, b\}^* \mid x = ywwz,\ w \neq \lambda \,\}$ *is* accpeted by an FA.
   **Proof:** We consider the complementary language $\overline{L} = \{a, b\}^* - L$. Notice that no string of length 4 or more can belong to $\overline{L}$ since all such strings contain either *aa*, *bb*, *abab* or *baba*. We obtain $\overline{L} = \{\lambda, a, b, ab, ba, aba, bab\}$ by examining all 15 strings over $\{a, b\}$ of length at most 3.

It's not difficult to draw an FA accepting this finite language. Upon reversing the accept/non-accept states in the FA for $\overline{L}$, we arrive at the following FA for $L$.




One can do this problem without actually drawing the diagram by arguing as follows. For each string in $\overline{L}$, there exists an FA accepting only that string. (In fact any single-string language is accepted by some FA, as has been seen in previous homework assignments.) The product construction (applied several times) yields an FA accepting the union

$$\overline{L} = \{\lambda\} \cup \{a\} \cup \{b\} \cup \{ab\} \cup \{ba\} \cup \{aba\} \cup \{bab\}$$

By simply reversing the accept/non-accept states in the FA for $\overline{L}$, we obtain an FA for $L$.

d. $L = \{\, x \in \{a, b\}^* \mid x = yaz, |y| = |z| \,\}$ *is not* accpeted by any FA.
**Proof:** Assume, to get a contradiction, that $L$ is accepted by some FA, and suppose that this FA has $n$ states. Let $x = b^n ab^n$. Then clearly $x \in L$ and $|x| \geq n$. The Pumping Lemma provides strings $u$, $v$ and $w$ such that $x = uvw$ and satisfying (1) $|uv| \leq n$, (2) $|v| > 0$ and (3) $uv^i w \in L$ for all $i \geq 0$. By the definition of $x$ and using (1), we see that $u$ and $v$ contain only $b$'s, and in particular $v = b^k$, for some $k \geq 1$. (Note (2) says $k \neq 0$.) By (3) we have $b^{n+k}ab^n = uv^2w \in L$. But clearly $b^{n+k}ab^n \notin L$ since $n + k \neq n$. This contradiction shows that no such FA can exist.

g. $L = \{\, x \in \{a, b\}^* \mid x \neq x^r \,\}$ *is not* accepted by any FA.
**Proof:** Suppose an FA exists that accepts $L$. Then, upon reversing the accept/non-accept states in this FA, we obtain an FA accepting $\overline{L} = \{a, b\}^* - L = \{\, x \in \{a, b\}^* \mid x = x^r \,\}$, which is the language of palindromes, *Pal*. But it was proved in class, and on page 62 of the text, that there are infinitely many $I_{Pal}$ equivalence classes in $\{a, b\}^*$, and hence *Pal* is *not* accepted by any FA. (This can also be proved using the Pumping Lemma.) This contradiction shows that no FA can accept $L = \overline{Pal}$.

h. $L = \{\, x \in \{a, b\}^* \mid n_a(x) \text{ is a perfect square} \,\}$ *is not* accepted by any FA.
**Solution:** We will display an infinite set of pairwise $L$-distinguishable strings in $\{a, b\}^*$. The result follows from Theorem 2.26 on page 62 of the text. We first introduce some notation and prove a lemma. Let $\mathbb{N}^2$ denote the set of perfect squares, i.e. $\mathbb{N}^2 = \{\, n^2 \mid n \in \mathbb{N} \,\}$.

**Lemma:** for any $n_1, n_2 \in \mathbb{N}$ with $n_1 \neq n_2$, there exists a number $k \geq 0$ such that $n_1 + k \in \mathbb{N}^2$ and $n_2 + k \notin \mathbb{N}^2$.
**Proof:** Assume for definitness that $n_1 < n_2$ (the other case being similar). Chose *m* sufficiently large that $n_1 < m^2$ and $n_2 - n_1 < 2m + 1$. Set $k = m^2 - n_1$. Then observe that $k \geq 0$ and $n_1 + k = m^2 \in \mathbb{N}^2$. By our choice of *m*, we also have

$$n_1 < n_2 < n_1 + 2m + 1$$

$$\Rightarrow \quad n_1 + k < n_2 + k < (n_1 + k) + 2m + 1$$

$$\Rightarrow \quad m^2 < n_2 + k < m^2 + 2m + 1 = (m+1)^2$$

Since $n_2 + k$ lies between two consecutive perfect squares, it cannot itself be a perfect square, i.e. $n_2 + k \notin \mathbb{N}^2$, as required.

**Claim:** No FA accepts $L = \{ x \in \{a,b\}^* \mid n_a(x) \in \mathbb{N}^2 \}$.
**Proof:** Let $S = \{ a^n \mid n \geq 0 \}$. Then any two distinct strings in *S* are *L*-distinguishable. Indeed, by the above lemma, if we pick $n_1, n_2 \geq 0$ with $n_1 \neq n_2$, there exists $k \geq 0$ such that $n_1 + k \in \mathbb{N}^2$ and $n_2 + k \notin \mathbb{N}^2$. Therefore $a^{n_1}a^k = a^{n_1+k} \in L$ and $a^{n_1}a^k = a^{n_2+k} \notin L$, showing that $a^k$ distinguishes $a^{n_1}$ from $a^{n_2}$ with respect to *L*. Since *S* is infinite, we are done by Theorem 2.26.

**Note 1:** In fact *x* is *L*-indistiguishable from $a^{n_a(x)}$ for any $x \in \{a,b\}^*$, and therefore the $I_L$ equivalence classes are exactly $\{ [a^n] \mid n \geq 0 \}$.
**Proof:** For any $z \in \{a,b\}^*$, $n_a(xz) = n_a(x) + n_a(z) = n_a(a^{n_a(x)}z)$ and therefore $xz \in L \leftrightarrow a^{n_a(x)}z \in L$.

**Note 2:** This result can also be proved using the Pumping Lemma.
**Proof:** Suppose *L* is accepted by some FA with *n* states. Let $x = a^{(n+1)^2}$. Then $|x| \geq n$ and $x \in L$. The Pumping Lemma provides a factorization $x = uvw$ with (1) $|uv| \leq n$, (2) $|v| \geq 1$ and (3) $uv^iw \in L$ for all $i \geq 0$. By (1) and (2) $v = a^k$ for some *k* satisfying $1 \leq k \leq n$. Letting $i = 0$ in (3) we have $uw \in L$, which implies $(n+1)^2 - k = n_a(uv) \in \mathbb{N}^2$. But observe $(n+1)^2 - k < (n+1)^2$ since $k \geq 1$. Also $n \geq k \Rightarrow n - k \geq 0 \Rightarrow 2n + 1 - k > 0$. Adding $n^2$ to both sides of this last inequality yields $(n+1)^2 - k = (n^2 + 2n + 1) - k > n^2$, and therefore $n^2 < (n+1)^2 - k < (n+1)^2$. But then $(n+1)^2 - k \notin \mathbb{N}^2$ since it lies between two consecutive perfect squares. This contradiction shows that no such *x* exists, and therefore no such FA exists.

3. Problem 3.1abcd
   In each case below, find a string of minimum length in $\{a,b\}^*$ *not* in the language corresponding to the given regular expression.
   a. $b^*(ab)^*a^*$
   b. $(a^* + b^*)(a^* + b^*)(a^* + b^*)$
   c. $a^*(baa^*)^*b^*$
   d. $b^*(a + ba)^*b^*$

   **Solution:** In what follows it will help to consider the set *S* of all 31 strings over $\{a,b\}$ of length 4 or less:

$$S = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb,$$
$$aaaa, aaab, aaba, aabb, abaa, abab, abba, abbb,$$
$$baaa, baab, baba, babb, bbaa, bbab, bbba, bbbb\}$$

a. The strings ***aab*** and ***abb*** do not belong to the language represented by $b^*(ab)^*a^*$. One checks directly that all other stings in *S* of length 3 or less match the regular expression. Therefore ***aab*** and ***abb*** are of minimum possible length.

b. The strings ***abab*** and ***baba*** do not match the regular expression $(a^* + b^*)(a^* + b^*)(a^* + b^*)$. An inspection of the strings in *S* reveals that all others match. Thus ***abab*** and ***baba*** are of minimum length.

c. The string ***bba*** does not belong to the language represented by $a^*(baa^*)^*b^*$. Direct inspection of *S* reveals that all other strings of length 3 or less match, so that ***bba*** is of minimum length.

d. The strings ***abba*** and ***abbb*** do not belong to the language represented by $b^*(a + ba)^*b^*$, while all other strings in *S* match the regular expression. Therefore ***abba*** and ***abbb*** are of minimum length.

4. Problem 3.2abcd

Consider the two regular expressions $r = a^* + b^*$ and $s = ab^* + ba^* + b^*a + (a^*b)^*$.

a. Find a string corresponding to *r* but not to *s*.
b. Find a string corresponding to *s* but not to *r*.
c. Find a string corresponding to both *r* and *s*.
d. Find a string in $\{a, b\}^*$ corresponding to neither *r* nor *s*.

**Solution:** We use the notation $L_{exp}$ to stand for the language over $\{a, b\}$ corresponding to the regular expression *exp*.

a. The string *aa* matches *r* but not *s*.
**Proof:** $aa = a^2 \in L_{a^*} \subseteq L_r$. But $aa \notin L_{ab^*}$ since strings in $L_{ab^*}$ have exactly one *a*, $aa \notin L_{ba^*}$ since strings in $L_{ba^*}$ start with *b*, $aa \notin L_{b^*a}$ since strings in $L_{b^*a}$ have exactly one *a*, and finally $aa \notin L_{(a^*b)^*}$ since $aa \neq \lambda$ and *aa* contains no *b*'s. Therefore $aa \notin L_s$.

b. The string *ab* matches *s* but not *r*.
**Proof:** Clearly $ab \in L_{ab^*} \subseteq L_s$. Also since *ab* is neither all *a*'s nor all *b*'s, $ab \notin L_{a^*}$ and $ab \notin L_{b^*}$, hence $ab \notin L_r$.

c. The string *a* matches both *r* and *s*.
**Proof:** We have $a \in L_{a^*} \subseteq L_r$ and $a \in L_{ab^*} \subseteq L_s$, so $a \in L_r \cap L_s$.

d. The string *bbaa* matches neither *r* nor *s*.
**Proof:** Since *bbaa* is neither all *a*'s nor all *b*'s, $bbaa \notin L_{a^*}$ and $bbaa \notin L_{b^*}$, hence $bbaa \notin L_r$. Also $bbaa \notin L_{ab^*}$ since it has more than one *a*, $bbaa \notin L_{ba^*}$ since it has more than one *b*, $bbaa \notin L_{b^*a}$ since it has more than one *a*. Finally any non-null string in $L_{(a^*b)^*}$ must end in *b*, hence $bbaa \notin L_{(a^*b)^*}$. Therefore $bbaa \notin L_s$.

5. Problem 3.3abc

Let *r* and *s* be arbitrary regular expressions over the alphabet Σ. In each case below, find a simpler equivalent regular expression.

a. $r(r^*r + r^*) + r^*$
b. $(r + \lambda)^*$
c. $(r + s)^*rs(r + s)^* + s^*r^*$

**Solution:** We write $\exp_1 = \exp_2$ to mean $L_{\exp_1} = L_{\exp_2}$.

a. $r(r^*r + r^*) + r^* = r^*$
**Proof:** A concatenation of one or more factors from $L_r$ is certainly a concatenation of zero or more such factors. Therefore $L_{r^*r} \subseteq L_{r^*}$ so $L_{r^*r} \cup L_{r^*} = L_{r^*}$, hence $r^*r + r^* = r^*$. Thus $r(r^*r + r^*) + r^* = rr^* + r^* = r^*$.

b. $(r + \lambda)^* = r^*$
**Proof:** Any concatenation of zero or more strings from $L_r$ and $\lambda$, is also a product of zero or more strings from $L_r$, and conversely. Hence $(r + \lambda)^* = r^*$.

c. $(r + s)^*rs(r + s)^* + s^*r^* = (r + s)^*$
**Proof:** Clearly $L_{(r+s)^*rs(r+s)^*} \subseteq L_{(r+s)^*}$ since each string in the left side is a product of strings in $L_r$ and strings in $L_s$, and the right side is the set of *all* strings of this kind. We need to show that every string in $L_{(r+s)^*}$ matches the expression on the left. Let $x \in L_{(r+s)^*}$ be chosen arbitrarily. Then *x* is a product of zero or more factors from $L_r$ and $L_s$. We have two cases.

Case 1: All factors in *x* from $L_s$ come before (i.e. to the left of) all factors from $L_r$. In this case *x* matches the regular expression $s^*r^*$.

Case 2: Some factor in *x* from $L_r$ precedes some factor from $L_s$. Those two factors match *rs* and in this case the string *x* matches the regular expression $(r + s)^*rs(r + s)^*$.

In both cases *x* matches the expression $(r + s)^*rs(r + s)^* + s^*r^*$, as required.

6. Problem 3.4

It is not difficult to show using mathematical induction that for every integer $n \geq 2$, there are nonnegative integers *i* and *j* such that $n = 2i + 3j$. With this in mind, simplify the regular expression $(aa + aaa)(aa + aaa)^*$.

**Solution:** $(aa + aaa)(aa + aaa)^* = aaa^*$
**Proof:** Since every string in $L_{(aa+aaa)(aa+aaa)^*}$ contains 2 or more *a*'s and nothing but a's, we have

$$L_{(aa+aaa)(aa+aaa)^*} \subseteq L_{aaa^*}$$

It remains to show that $L_{aaa^*} \subseteq L_{(aa+aaa)(aa+aaa)^*}$. Pick $x \in L_{aaa^*}$. Then $x = a^n$ for some $n \geq 2$. By the above fact, there exist $i, j \geq 0$ such that $n = 2i + 3j$. Note that not both *i* and *j* can be zero for otherwise *n* would be zero. If $i \geq 1$, then

$$x = a^{2i+3j} = (aa)\big((aa)^{i-1}(aaa)^j\big) \in L_{aa}L_{aa+aaa}^{i+j-1}$$

and hence $x \in L_{(aa+aaa)(aa+aaa)^*}$. If $j \geq 1$ then

$$x = a^{2i+3j} = (aaa)\big((aa)^i(aaa)^{j-1}\big) \in L_{aaa}L_{aa+aaa}^{i+j-1}$$

and again $x \in L_{(aa+aaa)(aa+aaa)^*}$. Thus $L_{(aa+aaa)(aa+aaa)^*} = L_{aaa^*}$, as required.

7. Problem 3.7cijm

Find a regular expression corresponding to each of the following subsets of $\{a, b\}^*$.

c. The language of all strings that do not end with *ab*.
i. The language of all strings containing both *bb* and *aba* as substrings.
j. The language of all strings not containing the substring *aaa*.
m. The language of all strings in which the number of *a*'s is even and the number of *b*'s is odd.

**Solution:** If *x* is any string over $\{a, b\}$, then $(a+b)^*x(a+b)^*$ is a regular expression matching any string having *x* as a substring.

c. Regular expression: $\lambda + a + b + (a+b)^*(aa + ba + bb)$
i. Regular expression: $(a+b)^*bb(a+b)^*aba(a+b)^* + (a+b)^*aba(a+b)^*bb(a+b)^*$
j. Regular expression: $(\lambda + a + aa)(b + ba + baa)^*$
m. Will talk about this in class.

8. Problem 3.9

Show that every finite language is regular.

**Solution:**

<u>Recall the recursive definition of $\mathcal{F}$:</u>

(1) (1.1) $\emptyset \in \mathcal{F}$ (1.2) $\{\lambda\} \in \mathcal{F}$ (1.3) $\{\sigma\} \in \mathcal{F}$ for all $\sigma \in \Sigma$.
(2) $L_1, L_2 \in \mathcal{F} \Rightarrow L_1 \cup L_2 \in \mathcal{F}$
(3) $L_1, L_2 \in \mathcal{F} \Rightarrow L_1L_2 \in \mathcal{F}$

**Lemma:** $\mathcal{F}$ consists of all finite languages.
**Proof:** Every language obtained by rule (1) is finite, and every language in $\mathcal{F}$ is obtained by a finite number of applications of rules (1), (2) and (3), hence every language in $\mathcal{F}$ is itself finite. It remains to show that every finite language is in $\mathcal{F}$. First observe that every single string language $\{x\}$ is in $\mathcal{F}$ by (1.2), (1.3) and (3). Indeed $\{\lambda\} \in \mathcal{F}$ by (1.2), and if $x = \sigma_1\sigma_2 \cdots \sigma_k$ then $\{x\} = \{\sigma_1\}\{\sigma_2\} \cdots \{\sigma_k\} \in \mathcal{F}$ by (1.3) and (3). Therefore if $L$ is a finite language, then (2) gives us

$$L = \{x_1, x_2, \dots, x_n\} = \{x_1\} \cup \{x_2\} \cup \cdots \cup \{x_n\} \in \mathcal{F}$$

<u>Recall the recursive definition of $\mathcal{R}$:</u>

(1) (1.1) $\emptyset \in \mathcal{R}$ (1.3) $\{\sigma\} \in \mathcal{R}$ for all $\sigma \in \Sigma$.
(2) $L_1, L_2 \in \mathcal{R} \Rightarrow L_1 \cup L_2 \in \mathcal{R}$
(3) $L_1, L_2 \in \mathcal{R} \Rightarrow L_1L_2 \in \mathcal{R}$
(4) $L \in \mathcal{R} \Rightarrow L^* \in \mathcal{R}$

**Claim:** $\mathcal{F} \subseteq \mathcal{R}$
**Proof:** Let $L \in \mathcal{F}$. We must show that $L \in \mathcal{R}$. We have two cases.
<u>Case 1:</u> $\lambda \notin L$.
In this case, $L$ is constructed by a finite number of applications of rules (1.1), (1.3), (2) and (3). But these construction steps are a subset of those defining $\mathcal{R}$. Therefore $L \in \mathcal{R}$ in this case.
<u>Case 2</u>: $\lambda \in L$.
Let $L_1 = L - \{\lambda\}$. We have $\{\lambda\} = \emptyset^* \in \mathcal{R}$ by (1.1) and (4), and $L_1 \in \mathcal{R}$ by case 1. It follows from (2) that $L = L_1 \cup \{\lambda\} \in \mathcal{R}$, in this case also.