Exercise 2.5

\[ L(m, n, x, y) = \{ w | \text{number of a's in } w = x \mod m \text{ and number of b's in } w = y \mod n \} \]

\[ |w| = \text{number of a's in } w + \text{number of b's in } w \]
\[ = x \mod m + y \mod n \]

On this path we have \(|w| + 1\) many states, which is equal to \((x \mod m + y \mod n + 1)\) states. Also there are \(x \mod m\) many a's and \(y \mod n\) many b's.

So now the question is how many such \(w\)'s?

Let \(C(n, m)\) denote the number of all possible combinations of choosing \(m\) objects from \(n\) objects. We know that \(C(n, m) = n!/(m!(n-m)!).\)

So \(|L(m, n, x, y)| = C(x \mod m + y \mod n, x \mod m)\). Hence, we know that the number of states in an FA to accept \(L(m, n, x, y)\) has the following upper bound:

\[ C(x \mod m + y \mod n, x \mod m)(x \mod m + y \mod n + 1) \]

If \(x < m\) and \(y < n\), then \(x \mod m = x\) and \(y \mod n = y\), and so this number becomes

\[ C(x + y, x)(x+y+1) \]

Regular Expressions:

**Definition:**

1. \(\Phi, e, a\) are regular expressions. (\(a\) is a symbol in \(\Sigma\))
2. If \(r_A\) and \(r_B\) are regular expressions, then
   \(r_A + r_B\) (in our text book \(r_A \cup r_B\))
   \(r_A r_B\)
   \(r_A^*\)
   are regular expressions.
3. No other expressions are regular.
   (This is a recursive/Inductive definition)

**Theorem 1:** Let \(r\) be a regular expression then \(\exists\) an NFA \(M\) such that \(L(M)\) can be represented by \(r\) (i.e. \(M\) accepts \(r\))

**Proof:**

Proof by construction and Induction:
1. Induction Basis:

For $\Phi$:

For $e$:

For $a \in \Sigma$

Induction Hypothesis:
Assume $r_A + r_B$ are regular expressions. Let $M_1, M_2$ be NFA such that $L(M_1)$ accepts $r_A$, $L(M_2)$ accepts $r_A$

$L(M) = r_A + r_B$

For $r_A r_B$

$L = r_A r_B$
For $r_A^*$

From the proof we know that there is a simple algorithm to construct on NFA to accept given a regular expression. This algorithm is a recursive algorithm.

*Example:*
Let $r = 101^*(01+0)^*10$  (r is the input in the algorithm)

For 1:

For 0:

For $1^*$

For $101^*$
For $(01 + 0)$

For $(01 + 0)^*$
For $101^*(01 + 0)^* 10$

Theorem 2:
Let $M$ be an NFA (DFA of NFA) then there is a procedure or algorithm that generates a regular expression $r$ such that $L(M) = r$

Proof:
We are going to use a technique called “State Elimination” the idea here is this:

Each of $r$ is a regular expression
To Eliminate State $V$:
In general let M be an FA. Add two new states ‘s’ and ‘f’. Where s is the initial state and f is the final state. Both of these states do not have self-loops.

To eliminate self-loop we have:

Then we will use the state elimination procedure to eliminate the intermediate state between s and f one by one to generate

Then $r = L(M)$

Example:
We want to find regular expression $r$ such that $L(M) = r$ to do this we will add $s$ and $f$

Step 1: Let us eliminate $A$:

Eliminate $B$: 
Eliminate D:

Eliminate C:

Hence the regular expression we are looking for is:

\[(0*11*0 + 0*01*1) 1* + 0*01*\]