4 Regular Expressions and Languages

4.1 Operations on Languages

Since languages are sets, we can use set operations like $\cup$ and $\cap$ to create new languages. But there are some additional operations that we will use.

- concatenation of strings: xy is the concetanation of strings x and y (x followed by y)
- concatenation of sets: $AB = \{xy \mid x \in A \wedge y \in B\}$
- power: $A^0 = \emptyset$; $A^1 = A$; $A^{n+1} = A^n A$. (Power is iterated concatenation.)
- Kleene star: $A^* = \cup_{n=0}^{\infty} A^n$

1. What strings belong to $\{0, 01\}^2$?
2. Which of these strings belong to $\{0, 01\}^*$?
    a. 01001
    b. 10110
    c. 00010
3. Which of these strings belong to $\{101, 111, 11\}^*$?
    a. 1010111
    b. 1011011
    c. 1110111
    d. 11110111

4.2 Regular Expressions

Regular expression are a type of shorthand notation for representing certain types of sets.

4.2.1 Syntax

A regular expression over an input alphabet I is defined recursively.

- $\boldsymbol{\emptyset}$ and $\boldsymbol{\lambda}$ and regular expressions.
- $\boldsymbol{x}$ is a regular expression for each $i \in I$.
- If $\boldsymbol{A}$ and $\boldsymbol{B}$ are regular expressions, then
  - $\boldsymbol{A}^*$ is a regular expression
  - $(\boldsymbol{A} \cup \boldsymbol{B})$ is a regular expression
  - $(\boldsymbol{AB})$ is a regular expression

4.2.2 Semantics

Each regular expression represents a language as follows.

- $\boldsymbol{\emptyset}$ represents the empty langauge: $\emptyset$
- $\boldsymbol{\lambda}$ represents language that contains only the empty string: $\{\lambda\}$
- For each $x \in I$, $\boldsymbol{x}$ represents the language containing only x: $\{x\}$
- If $\boldsymbol{A}$ and $\boldsymbol{B}$ are regular expressions representing languages A and B, then
  - $(\boldsymbol{AB})$ represents AB
  - $(\boldsymbol{A} \cup \boldsymbol{B})$ represents $A \cup B$ [Note: This is often written '(A|B)' in programming languages.]
  - $\boldsymbol{A}^*$ represents A^* [Note: This is often written as 'A*' in programming languages.]

The langauges represented by regular expressions are called **regular languages** (how creative).

4. In Rosen 7, the definition of regular expression is missing boldface in a couple places. Find them.

> The *regular expressions* over a set I are defined recursively by:
>
> the symbol $\emptyset$ is a regular expression;
> the symbol λ is a regular expression;
> the symbol $\boldsymbol{x}$ is a regular expression whenever $x \in I$;
> the symbols $(\mathbf{AB})$, $(\mathbf{A} \cup \mathbf{B})$, and $\mathbf{A}^*$ are regular expressions whenever $\mathbf{A}$ and $\mathbf{B}$ are regular expressions.

5. What is the difference between $\mathbf{1}$ and 1? between $\boldsymbol{\lambda}$ and λ? between $\boldsymbol{\emptyset}$ and $\emptyset$?
6. Describe in words the languages represented by each of these regular expressions:

   a. $\mathbf{10^*}$ b. $\mathbf{(10)^*}$ c. $\mathbf{0 \cup 01}$ d. $\mathbf{0^* \cup 01}$ e. $\mathbf{0(0 \cup 1)^*}$ f. $\mathbf{(0 \cup 01)^*}$
7. Can each of the languages above be recognized by an NFA? a DFA?
8. What would we need to do to show that every regualar language can be recognized by an NFA? Give it a try and see how far you get. Which parts are easy? Which are trickier? Why?