CS 241 — Lecture 9

Bartosz Antczak *Instructor: Kevin Lanctot* *January 31, 2017*

9.1 Regular Languages

This topic focuses on building a *compiler*, which takes high-level code and translates it into assembly language (rather than what we were doing up to this point, which was translating assembly language into machine code).
The steps in compiling a program from a high level language to an assembly language program are:

1. **Scanning:** create a token sequence
2. **Syntax analysis:** create a *parse tree* (rather than a list of tokens, this is new)
3. **Semantic analysis:** create a symbol table and *type checking* (which is new)
4. **Code generation:** similar, but more complicated for a compiler

The goal of each of these steps is to find increasingly more sophisticated errors in a program. If there are no errors, then a successful compilation occurs; otherwise, print an error message.
The **Chomsky Hierarchy** can be used to find such errors.
We'll be compiling our own defined langauge, called WLP4 (CS 241's Waterloo Language Plus Pointers Plus Procedures).

9.1.1 Approach to WLP4

Use **regular expressions** to describe our language. A regular expression is a precise way of describing a set of strings (think of programs as a sequence of characters, which is actually what they are). We also define a **string** as a finite sequence of characters over some alphabet (our alphabet in this course will be some subset of the ASCII characters).

Three Operations for Building up Languages

1. **Union:**
   $R \cup S$ is the union of set R and S. If R and S are regular languages, then so is $R \cup S$. For example, if $R =$ {cow, pig} and $S =$ {dog, cat}, then $R \cup S =$ {dog, cat, cow, pig}.
2. **Concatenation:**
   Defined as $RS = \{\alpha\beta : \alpha \in R \wedge \beta \in S\}$ (i.e., take a word from R and combine it with a word from S). If $R =$ {grey, blue} and $S =$ {whale}, then $RS =$ {greywhale, bluewhale}.
   Concatenation with the empty string, ϵ, does nothing (i.e., $\epsilon\alpha = \alpha$).
3. **Repetition:**
   If R is a language, we can talk about R^0, R^1, R^2, R^3, etc. To define every possible sequence of characters, we denote R^* (a.k.a. a *Kleene star*). For example, if $R = \{0, 1\}$, then
   - $R^0 = \{\varepsilon\}$

- $R^1 = \{0, 1\}$ (all single elements)
- $R^2 = \{00, 01, 10, 11\}$ (all pairs of elements)

9.1.2 Two Types of Languages

A **language** is defined as a set of finite sequences of characters from a defined alphabet. We can define a **finite** or **infinite** language. Infinite languages are denoted with a Kleene star, which means (as mentioned) that language contains every possible sequence of characters. An example of an infinite language is:

$$a* = \{\varepsilon, a, aa, aaa, ...\}$$