Lecture 7
Introduction to Formal Languages

CS 241: Foundations of Sequential Programs
Spring 2017

Troy Vasiga et al. (where Dan Holtby ∈ al)
University of Waterloo
Motivation

- precision of specification and recognition
- importance of this: every assignment from this point forward
- benefits of theory:
Terminology

- rooted in set theory
- alphabet: a finite set of symbols

\[ \Sigma_1 = \{a, b, c\} \]

- word (aka string, sentence): finite sequence of symbols from the alphabet

\[ W_1 = abba, \quad W_2 = c, \quad W_3 = \epsilon \]
Terminology (continued)

- **language**: set of words (finite or infinite set)

\[ L_1 = \{a, aa, aaa, \ldots\}, \quad L_2 = \{a, aa, ab, ac\}, \quad L_3 = \{\} \]

- \(|W|\): the size of \(W\) (where \(W\) is a word or language)

\[ |L_1| = |\aleph_0|, \quad |L_2| = 4 \]
Uses of Formal Languages: Specification

A statement about what’s in the language. Should be

- Precise (and not ambiguous)
- Easy to express (and understand)
- Automatable (easy to write a computer program to recognize the language)
Uses of Formal Languages: Recognition

Is a word in a language?

- In Theory: Given language $L$ and word $W$, is $W \in L$? (boolean function)
- In Practice:
  - Instead of "false" indicate location of error and type of error.
  - Instead of "true" produce a certificate of correctness ("proof" that $W \in L$)
Real example plz?

A3+A4 Pass 1 is recognizing the CS241 MIPS assembly dialect.

- If the input is not in the set of valid assembly programs, print an error message
- If it is, generate the intermediate representation.

What is Σ? The set of tokens.
Tokens

Tokens are an alphabet, but also a language!

Tokenization means recognizing words from this language

\[ L_{\text{register}} = \{0, 1, \ldots, 31\}, \quad L_{\text{label}} = \{a:, b: \ldots\}, \text{ etc.} \]

\[ L_{\text{tokens}} = L_{\text{register}} \cup L_{\text{label}} \cup \ldots \]
How could you formally specify MIPS assembly?

- **alphabet:** Tokens (Itself a language you’ll need to formally specify)

- **word:** An assembly program (a sequence of zero or more tokens that follow all of the syntax rules).

- **language:** The set of all valid assembly programs (countably infinite set!)
Let’s try to formally specify WLP4.

- **alphabet**: Tokens (see webpage)

- **word**: A valid WLP4 program

- **language**: The set of all valid WLP4 programs

Just like MIPS assembly except everything is different.
Language Classes

- a set of languages may share common characteristics
- Chomsky Hierarchy
  - Finite Languages
  - Regular Languages
  - Context-Free Languages
  - Context-Sensitive Languages
Uses of Formal Languages: Organization of Compilation

- lexical analysis (tokenization) - RL
- syntactic analysis - CFL
- context-sensitive analysis (semantic) analysis - CSL
- synthesis (code generation) - CSL
Which language level?

Let $\Sigma = \{\text{ASCII characters}\} - \{\text{CR}\}$.

- $L_1 = \{$$0, $$1, $$2, \ldots, $$31 \}$
- $L_2 = \text{valid labels in MIPS}$
- $L_3 = \text{valid load word (lw) offsets}$
- $L_4 = \text{valid line of AL for A3P2}$
- $L_5 = \text{valid line of AL for A3P4}$
Regular Languages

- Defining regular languages: two approaches
  - Rules for making a word (construction)

- Rules for recognizing a word

- We will see that these are equivalent
Specifying Regular Languages

Basic building blocks

1. finite languages
2. union
3. concatenation
4. repetition
Definition: Shouldn’t you already know this?

\[
\begin{align*}
\text{FINE: } T_1 \cup T_2 &= \{x \mid x \in T_1 \text{ or } x \in T_2\}
\end{align*}
\]

Examples:
\[
\begin{align*}
T_1 &= \{\text{dog, cat}\} \\
T_2 &= \{\text{dog, fish}\} \\
T_1 \cup T_2 &= \{\text{cat, dog, fish}\}
\end{align*}
\]
Definition:
\[ T_1 \cdot T_2 = \{xy \mid x \in T_1 \text{ and } y \in T_2\} \]

Examples:
\[ T_1 = \{\text{dog, cat}\} \]
\[ T_2 = \{\epsilon, \text{fish}\} \]
\[ T_1 \cdot T_2 = \{\text{dog, cat, dogfish, catfish}\} \]
Repetition

定义（克伦闭包）：
\[ T^* = \{ \epsilon \} \cup \{ xy \mid x \in T^* \text{ and } y \in T \} \]

交替定义：
\[ T^0 = \{ \epsilon \}, \quad T^k = T \cdot T^{k-1} \]
\[ T^* = \bigcup_{k=0}^{\infty} T^k \]

例子：
\[ T_1 = \{ \text{dog, cat} \} \]
\[ T_1^* = \{ \text{dog, dogcat, dogdog, dogdogcat, ... cat, catdog, ...} \} \]