Lecture 9

Introduction to Formal Languages

CS 241: Foundations of Sequential Programs
Fall 2009

Troy Vasiga et al
University of Waterloo

Motivation

- precision of specification and recognition
- importance of this: A3/A4 (and later assignments)
- benefits of theory:

Terminology

- rooted in set theory
- alphabet: a finite set of symbols
- word (aka string, sentence): finite sequence of symbols
- language: set of words
- $|W|$: the size of W (where W is a word or language)

Uses of Formal Languages: Specification

Uses of Formal Languages: Recognition

Language Classes

- a set of languages may share common characteristics
- Chomsky Hierarchy

Uses of Formal Languages: Organization of Compilation

- lexical analysis
- syntactic analysis
- context-sensitive analysis (semantic) analysis
- synthesis (code generation)

Regular Languages

- Defining regular languages: two approaches
- We will see that these are equivalent

Specifying Regular Languages

Basic building blocks

1. finite languages
2. union
3. concatenation
4. repetition

Union

- Definition:
- Examples:

Concatenation

- Definition:
- Examples:

Repetition

- Definition:
- Alternate Definition:
- Examples:

Recognizers: Finite Automata

Regular languages can be recognized by *finite automata*.
We begin with *deterministic finite automata*, also called DFAs.

- states
- transitions
- start state
- final states

Finite Automata Example 1

Example: selected opcodes from MIPS assembly language, where alphabet is the ASCII characters.
Observations:

- ability to trace
- transitions out of a state are unique
- errors
- size of this language
- DFA M and language L(M)

Finite Automata Example 2

Example: MIPS labels, where the alphabet is ASCII characters.

Many More Finite Automata Examples

Let $\Sigma = \{a, b, c\}$.

- strings with one a and one b
- strings with one a, one b and one c
- strings with at least one a
- string with an even number of a's
- strings with an even number a's and odd number of b's
- strings with an even number a's or odd number of b's
- valid MIPS identifiers
- valid register designations in MIPS
- valid lines of MIPS assembly language