Introduction to Formal Languages
Lecture 29

Today:

- Formal Languages
- Finite State Machines
Natural Languages

A natural language is used for the purposes of human communication

- spoken, written, or gestured
- E.g., English, French, Mandarin

There are some rules:

- valid characters (alphabet)
- valid words (spelling)
- valid sentences, punctuation (grammar)
- acceptable idioms
Formal Languages

A *formal language* is used to distinguish precisely what is allowed from what is not.

- expressed mathematically, often with recursion
- E.g., valid postfix expressions, valid C++

Similar to natural languages, there are:

- alphabets
- words
- **grammars**
- but no idioms

Noam Chomsky
(Grammar Savant)
Alphabets and Words

An *alphabet* is a finite collection of symbols

- E.g., $\Sigma = \{a, b, c, \ldots, z\}$ — letters of the alphabet
- E.g., $\Sigma = \{0, 1, 2, \ldots, 9\}$ — base ten digits
- E.g., $\Sigma = \{0, 1\}$ — binary digits

A *word* is a finite sequence of alphabet symbols

- symbols may be repeated, e.g., *baa*, *100*, *sheep*
- order matters, e.g., *stressed* vs *desserts*
- word of length 0 is special, i.e., the *empty string* ($\lambda$, $\varepsilon$)

Distinguish which words are valid vs invalid.
A [formal] language is a set of words.

- can be finite, e.g., $L = \{\text{all valid English words}\}$
- can be infinite, e.g., $L = \{\text{all valid decimal numbers}\}$
- remember that words are always of finite length

E.g., Let $L = \{\text{all valid C++ programs}\}$

- Q. What’s the alphabet?
- Q. What are the words?
- Q. Is $L$ finite or infinite?
- Q. What does it mean to have an infinite length word?
Specifying Formal Languages

Just like in natural language, use a grammar

- describes the symbols allowed and the order that they should appear
- usually specified recursively
- E.g., A valid sentence is a noun phrase followed by a verb phrase followed by a subordinate clause.

A subordinate clause may be composed of the symbol “where” followed by a valid sentence.

- E.g., A valid postfix expression is either: a single number OR two valid postfix expressions followed by an operator.

Can represent a grammar using production rules:

- E.g., Grammar for postfix: $E \rightarrow \text{number}$
  
  $E \rightarrow E E \text{ operator}$

Write algorithms to decide inclusion/exclusion in the language.
To decide a language, try a finite state machine (FSM).

Rules of the Game:

- Finite number of states.
- The FSM reads one character at a time.
- The next state is determined by examining the current state and the next input character, and nothing else.
- Each state has at most one transition on any given character.
- One state is identified as the Start state.
- One or more states are designated Final states.
- Under no circumstances may a previously read character be examined again.
- If the last state is a final state: Accept
- If not: Reject
Two Puzzles for You

Q. What languages do these FSMs represent?

$\Sigma = \{a, b\}$

Begins with $b$

Starts with 0

Starts with 1

Ends with 0

Ends with 1

$\Sigma = \{0, 1\}$

Start and end with different symbols
Using The Dead State

Default transition to the dead state if no transition present.

Q. Build a FSM that accepts all words of length 3. $\Sigma = \{a, b\}$.  
Q. Build a FSM that accepts all decimal integers. Leading zeroes are disallowed. $\Sigma = \{0, 1, 2, \ldots, 9\}$