Languages and Regular Expressions

Robert M. Keller
Harvey Mudd College
March 2010

Definition of the Concept “Language”

- A **language** over an alphabet $\Sigma$ is any **subset** of $\Sigma^*$.
- The empty set $\emptyset$ and $\Sigma^*$ itself are both languages.
- Give some other precise examples of languages.

Operations on Languages

- Since languages are sets, we can define their **union**, **intersection**, etc. just as with any sets, e.g.
- Let $L$ and $M$ be two languages.
  
  $L \cup M = \{x \mid x \in L \text{ or } x \in M\}$
  
  $L \cap M = \{x \mid x \in L \text{ and } x \in M\}$
  
  $L - M = \{x \mid x \in L \text{ and } x \notin M\}$

Product of Languages

- Let $L$ and $M$ be two languages. Define
  
  $LM = \{xy \mid x \in L, y \in M\}$
  
  called the "product" or (loosely) the "concatenation" of languages.
- Give examples.
- What if either is $\emptyset$?

Power Operator for Languages

- Let $L$ be a language. Define the "$n$th power" of $L$ inductively:
  
  $L^0 = \{\varepsilon\}$
  
  $L^{n+1} = L \cdot L^n$
- Examples?

Plus and Star Operators for Languages

- Let $L$ be a language. Define
  
  $L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$
- Define
  
  $L^+ = L^1 \cup L^2 \cup L^3 \cup \ldots$
- Thus
  
  $L^* = \{\varepsilon\} \cup L^+$
- Give examples.
Language Identities: Devise RHS’s

• \( L \emptyset = \emptyset \)
• \( L \{ \epsilon \} = \{ \epsilon \} \)
• \( (LM)N = L(MN) \)
• \( L^* = L \emptyset^* \)
• \( \{ \epsilon \}^* = \emptyset \)
• \( \epsilon^* = \emptyset \)
• \( (L \cup M)N = (LM)N \)
• \( (L \cup M)^* = (LM)^* \)

Solving a Language Equation: Arden’s Rule

• This will be seen to be a useful device shortly:
• The equation \( L = AL \cup B \) with \( A \) and \( B \) being languages and \( L \) an unknown has as a solution for \( L \):

\[
L = A^*B
\]
• Justify by substitution for \( L \) in the equation.
• This is the smallest solution.
• When is the solution unique?

Uniqueness in Arden’s Rule

- Uniqueness holds if \( A \) does not contain \( \epsilon \).
- If \( A \) contains \( \epsilon \), then \( A^*C \) is a solution for any \( C \supseteq B \).

Regular Operators and Languages

- Union, Star, and Product (Concatenation) are called the Regular Operators on Languages.
- Definition: A language is regular if it can be formed from languages that are finite, using a finite number of regular operators.
- Note: \( ^* \) counts as only one operator, despite it being defined as an infinite union.
- Examples of Regular Languages?

True or False?

- Any language of exactly one element is regular.
- Any finite language is regular.
- \( \Sigma^* - L \), where \( L \) is finite, is regular.
- Every language is regular. To see this, let \( L = \{ x_1, x_2, x_3, \ldots \} \).
  Then \( L = \{ x_1 \} \cup \{ x_2 \} \cup \{ x_3 \} \cup \ldots \), which is clearly regular.

Regular Expressions

(cf. Sipser, section 1.3)

- A regular expression is a shorthand way of representing regular languages using regular operator symbols in conjunction with the following symbols.
- Each letter \( \sigma \) in \( \Sigma \) stands for the language with just one string of one letter, that letter.
- \( \epsilon \) stands for the language \( \{ \epsilon \} \).
- \( \emptyset \) stands for the empty language \( \emptyset \).
- Example: If \( \Sigma = \{ 0, 1 \} \), then \( 0 \) stands for the language with just one string, that string having one letter, \( 0 \).
Examples of Regular Expressions

- \(0 \cup 1\)
- \((0 \cup 1)\)*
- \((0 \cup 1)0^*1^*\)
- \(((0 \cup 1)0^*1^*)^*\)
- \(((\varepsilon \cup 1)0^*1^*)^*\)
- \(0^*110^* \cup 1^*001^*\)

Regular Expression Notation Notes

- Instead of \(\cup\), some sources use infix + or | in regular expressions.
- \(^*\) binds the tightest, then concatenation, then \(\cup\).
- \(\cap\) is not a regular operator, nor is -. However, we can show that these operators still preserve regularity.

Regular Expressions as Patterns

- Any language can be equated to a "pattern", namely the pattern that matches all strings in the language.
- Examples:
  - \(0^*\) is the pattern that matches strings containing only 0's
  - \(0^*10^*\) is the pattern that matches strings in \((0, 1)^*\) containing exactly one 1.
  - \(0^*10^*\) is the pattern that ...
  - \(0(0 \cup 1)^*1\)
  - \(((0 \cup 1)(0 \cup 1))^*\)
- Note: To qualify as a pattern, the language of the expression must be that of \textbf{exactly} the set of strings matching the pattern, not a subset or superset.

Regular Expressions as Patterns

- Give regular expressions for the following patterns over \(\{0, 1\}\):
  - Strings in which each 1 is followed by a 0.
  - Strings in which no 1 is followed by a 0.
  - Strings in which every 1 is preceded by and followed by a 0.
  - Strings in which the number of 1's is divisible by 3.
  - Strings in which there is no run of 3 consecutive 1's.

Application: Searchers

- Do `man egrep` on a UNIX system.
- How do such search algorithms work?