Week 13 - Monday

COMP 2230

Last time

- Exam 3 post mortem
- Introduction to formal languages

Questions?

Assignment 6

Logical warmup

- Imagine three four-sided dice
- Each one has a single letter on each side
- Rolling the dice yields three random letters which can sometimes be arranged into a word
- Rolling the dice has generated these words:
  - CAT, SON, POD, RIG, PEG, TAP, DIN, APE
- What are the letters on each of the dice?

Back to Formal Languages

Notation

- Let Σ be some alphabet
- For any nonnegative integer n, let
  - Σ^n be the set of all strings over Σ that have length n
  - Σ^+ be the set of all strings over Σ that have length at least 1
  - Σ^* be the set of all strings over Σ
- Σ^* is called the **Kleene closure of** Σ and the $*$ operator is often called the **Kleene star**

Examples

- Let alphabet $\Sigma = \{x, y, z\}$
- Find Σ^0, Σ^1, and Σ^2
- What is $A = \Sigma^0 \cup \Sigma^1$? What is $B = \Sigma^1 \cup \Sigma^2$? How would you describe these sets and set $A \cup B$ in words?
- Describe a systematic way of writing out Σ^+
- How would you have to change your system to write out Σ^*?

More notation

- Let Σ be a finite alphabet
- Given strings x and y over Σ, the **concatenation of x and y** is the string made by writing x with y appended afterwards
- With languages L and L' over Σ, we can define the following new languages:
  - **Concatenation of L and L'**, written LL'
    - $LL' = \{ xy \mid x \in L \text{ and } y \in L' \}$
  - **Union of L and L'**, written $L \cup L'$
    - $L \cup L' = \{ x \mid x \in L \text{ or } x \in L' \}$
  - **Kleene closure of L**, written L^*
    - $L^* = \{ x \mid x \text{ is a concatenation of any finite number of strings in } L \}$

Examples

- Let alphabet $\Sigma = \{a, b\}$
- Let L_1 be the set of all strings consisting of an even number of a's (including the empty string)
- Let $L_2 = \{b, bb, bbb\}$
- Find
  - $L_1 L_2$
  - $L_1 \cup L_2$
  - $(L_1 \cup L_2)^*$

Regular expressions

- It's getting annoying trying to describe infinite languages using ellipses
- Notation called a regular expression can allow us to express languages precisely and compactly
- Given a finite alphabet Σ, we can define regular expressions recursively:
  I. **Base**: The empty set, the empty string ε, and any individual character in Σ is a regular expression
  II. **Recursion**: If r and s are regular expressions over Σ, then the following are too:
     a) **Concatenation**: (rs)
     b) **Alternation**: $(r \mid s)$
     c) **Kleene star**: (r^*)
  III. **Restriction**: Nothing else is a regular expression

Languages defined by a regular expression

- For a finite alphabet Σ, the language $L(r)$ defined by a regular expression r is as follows
- **Base**: $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$, $L(a) = \{a\}$ for every $a \in \Sigma$
- **Recursion**: If $L(r)$ and $L(r')$ are the languages defined by the regular expressions r and r' over Σ, then
  - $L(r\, r') = L(r)L(r')$
  - $L(r \mid r') = L(r) \cup L(r')$
  - $L(r^*) = \big(L(r)\big)^*$

Examples

- Let $\Sigma = \{a, b, c\}$
- Let language $L = a \mid (b \mid c)^* \mid (ab)^*$
  - Write 5 strings in L
- Let language $M = ab^*(c \mid \varepsilon)$
  - Write 5 strings in M

Order of precedence

- For the sake of consistency, regular expressions obey a particular order of precedence
  - $*$ is the highest precedence
  - Concatenation is the next highest
  - Alternation is the lowest
- Parentheses can be omitted if there is no ambiguity
- Write $(a((bc)^*))$ with as few parentheses as possible
- Write $a \mid b^* \, c$, using parentheses to mark the precedence of each operation

Equivalences

- As before, let $\Sigma = \{a, b\}$
- Can you describe $(a \mid b)^*$ in another way?
- What about $(\varepsilon \mid a^* \mid b^*)^*$?
- Given that $L = a^*b(a \mid b)^*$, write 5 strings that belong to L
- Let $M = a^* \mid (ab)^*$
  - Which of the following belong to M?
    - a
    - b
    - $aaaa$
    - $abba$
    - $ababab$

Examples

- Let $\Sigma = \{0, 1\}$
- Find regular expressions for the following languages:
  - The language of all strings of 0's and 1's that have even length and in which the 0's and 1's alternate
  - The language consisting of all strings of 0's and 1's with an even number of 1's
  - The language consisting of all strings of 0's and 1's that do not contain two consecutive 1's
  - The language that gives all binary numbers written in normal form (that is, without leading zeroes, and the empty string is not allowed)

Upcoming

Next time…

- Finite-state automata

Reminders

- Keep working on Assignment 6
- Read 12.2 for Wednesday
  - Prepare a three-sentence summary
  - Extra credit if you get called on