# Regular Expressions

## A Language for Specifying Languages

Thursday, September 20, 2007
Reading: Stoughton 3.1

**CS235 Languages and Automata**

Department of Computer Science
Wellesley College

## Motivation for Regular Expressions

There are many languages simple to describe in English that we would like to specify via a concrete formal notation.

E.g., consider the following {a,b,c}-languages:

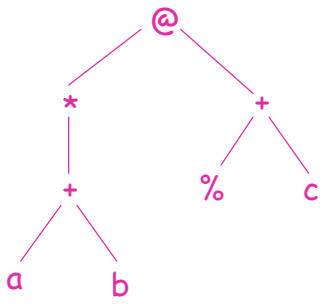
- All strings containing only a's
- All strings containing exactly one b
- All strings containing at least one b
- All strings containing the substring cab
- All strings that have even length.
- All strings with length 3.
- All strings that start and end with the same symbol
- All strings containing b and c in which the first c appears after the first b
- All strings in which every pair of consecutive b's delimits a substring containing exactly one c

**Regular expressions** are a notation for specifying languages like these.

## Regular Expression Example

A regular expression is a tree with binary @ and + nodes, unary * nodes, and leaves that are either %, $, or symbols from Sym.

*Example:* @(*(+(a,b)),+(%,c))






## Formal Syntax of Regular Expressions

Let **RegLab** = **Sym** ∪ {%, $, *, @, +}

The set **Reg** of regular expressions is the least subset of $\mathbf{Tree}_{\text{RegLab}}$ inductively defined by the following rules:

- (empty string) % ∈ **Reg**
- (empty set) $ ∈ **Reg**
- (symbol) for all a in **Sym**, a ∈ **Reg**
- (closure) for all $\alpha$ ∈ **Reg**, *($\alpha$) ∈ **Reg**
- (concatenation) for all $\alpha$, $\beta$ ∈ **Reg**, @($\alpha$,$\beta$) ∈ **Reg**
- (union) for all $\alpha$, $\beta$ ∈ **Reg**, +($\alpha$,$\beta$) ∈ **Reg**

## Formal Semantics of Regular Expressions

The meaning of a regular expression is defined by the following function L : **Reg** → **Lan**:

L(%) = {%}

L($) = ∅

L(a) = {a}

L(*(α)) = (L(α))*

L(@(α,β)) = L(α) @ L(β)

L(+(α,β)) = L(α) ∪ L(β)

A language is a **regular language** if it is the meaning of some regular expression. **RegLan** is the set of all regular languages.



## Semantics Example

L( @(*(+(a,b)), +(%,c)) )

= L( *(+(a,b)) ) @ L( +(%,c) )

= (L( +(a,b) ))* @ (L( % ) ∪ L( c ))

= (L( a ) ∪ L( b ))* @ ({%} ∪ {c})

= ({a} ∪ {b})* @ {%, c}

= {a, b}* @ {%, c}

## Abbreviations for Regular Expressions

- Abbreviate *(α) as α *
- Abbreviate @(α,β) as α@β or αβ
  right associative: a@β@γ means @(a,@(β,γ))
- Abbreviate +(α,β) as α+β
  right associative: α+β+γ means +(α,+(β,γ))
- Precedence order: * is highest, @ is middle, + is lowest
  override precedence with explicit parentheses.

Examples:

- a+b*c+d means +(a,+(@(*(b),c),d))
- a+b*(c+d) means +(a, @(*(b), +(c, d)))
- (a+b)*(c+d) means @(*(+(a,b)),+(c,d))
- (a+b)*c+d means +(@(*(+(a,b)),c),d)



## Returning to Our Initial {a,b,c} Examples

| English Specification | Regular Expression |
|---|---|
| 1. All strings containing only a's | |
| 2. All strings containing exactly one b | |
| 3. All strings containing at least one b | |
| 4. All strings containing the substring cab | |
| 5. All strings that have even length | |
| 6. All strings with length 3 | |
| 7. All strings that start and end with the same symbol | |
| 8. All strings containing b and c in which the first c appears after the first b | |
| 9. All strings in which every consecutive pair of b's delimits a substring containing exactly one c | |

## Languages and Countability

Recall that **Sym** is a fixed universe of symbols and **Str** is the set of all strings over **Sym**.

- **Str** is countably infinite. Why?
- Suppose L is a **Sym**-language (i.e., a subset of **Str**).
  L must be countable. Why?
- Let **Lan** be the set of all **Sym**-languages = *P*(**Str**).
  **Lan** is uncountable. Why?
- The set **Reg** of regular expressions is countably infinite. Why?
- The set **RegLan** of regular languages is countably infinite. Why?