

Lexical Analysis

CMPT 379: Compilers

Instructor: Anoop Sarkar

anoopsarkar.github.io/compilers-class

Regular Expressions are Trees

Regular Expressions: Definition

- Note that operators apply recursively and these applications can be ambiguous
 - E.g. is $aa|bc$ equal to $a(a|b)c$ or $((aa)|b)c$?
- Avoid such cases of ambiguity - provide explicit arguments for each regexp operator
 - For convenience, for examples on this page, let us use the symbol ‘.’ to denote the operator for concatenation
- Remove ambiguity with an explicit regexp tree

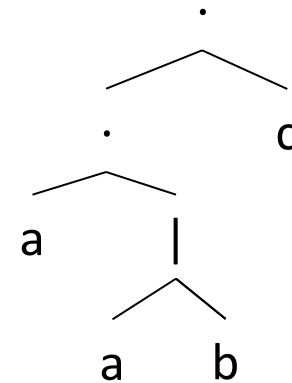
Regular Expressions: Definition

- Remove ambiguity with an explicit regexp tree

$a(a|b)c$ is written as

$(\cdot(\cdot a(|ab)))c$

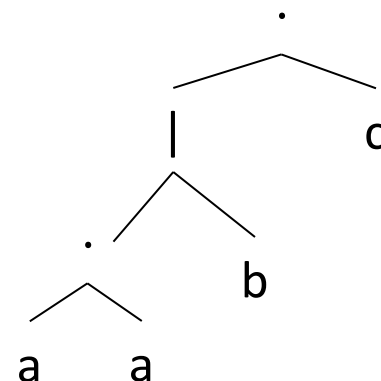
or in postfix: $aab| \cdot c \cdot$



$((aa)|b)c$ is written as

$(\cdot(|(\cdot aa)b))c$

or in postfix: $aa \cdot b|c \cdot$



- Does the order of concatenation matter?

Equivalence of Regexps

- $(R|S)|T == R|(S|T) == R|S|T$
- $(RS)T == R(ST)$
- $(R|S) == (S|R)$
- $R^*R^* == (R^*)^* == R^*$
 $== RR^* | \epsilon$
- $R^{**} == R^*$
- $(R|S)T = RT|ST$
- $R(S|T) == RS | RT$
- $(R|S)^* == (R^*S^*)^* == (R^*S)^*R^* == (R^*|S^*)^*$
- $RR^* == R^*R$
- $(RS)^*R == R(SR)^*$
- $R = R|R = R\epsilon$

$(0(10)^*1)|(01)^* ??$

Equivalence of Regexps

- $(0(10)^*1) \mid (01)^*$
- $(01(01)^*) \mid (01)^*$
- $(01(01)^*) \mid (01)^*$
- $((01)(01)^*) \mid (01)^*$
- $((01)(01)^*) \mid (01)^*$
- $(01)^+ \mid (01)^*$
- $(01)^+ \mid (01)^*$
- $(01)^*$
- $(RS)^*R == R(SR)^*$
- $RS == (RS)$
- $R^+ == RR^*$
- $R^+ \mid R^* == (RR^*) \mid R^* == R^*$