Regular Expressions

- Fix an alphabet Σ. We define now regular expressions and how each of them specifies a language.
-
- **Definition.** The set of *regular expressions* (with respect to Σ) is defined inductively by the following rules:
  1. The symbols $\varnothing$ and Λ are regular expressions
  2. Every symbol $\alpha \in \Sigma$ is a regular expression
  - 3. If E and F are regular expressions, then (E^*), (EF) and $(E+F)$ are regular expressions.

Example Program

```
data RegExp a
  = Lambda
  | Empty
  | One a
  | Union (RegExp a) (RegExp a)
  | Cat (RegExp a) (RegExp a)
  | Star (RegExp a)
```

Regular Expressions as Languages

- **Definition.** For every regular expression E, there is an associated language *L(E)*, defined inductively as follows:

  1. $L(\emptyset)=\emptyset$ and $L(\Lambda)=\{\Lambda\}$
  2. L(a)={'a'}
  3. Inductive cases
     1. $L(E^*) = (L(E))^*$
     2. L(EF) = L(E) L(F)   recall implicit use of dot $L(E) \bullet L(F)$
     3. $L(E+F) = L(E) \cup L(F)$

- **Definition.** A language is *regular* if it is of the form L(E) for some regular expression E.

Equivalence

1. We say that regular expressions E and F are *equivalent* iff L(E)=L(F).
2. We treat equivalent expressions as equal (just as we do with arithmetic expressions; (e.g., 5+7 = 7+5).
3. Equivalences (E+F)+G = E+(F+G) and (EF)G=E(FG) allow us to omit many parentheses when writing regular expressions.
4. Even more parentheses can be omitted when we declare the precedence ordering of the three operators :
   1. star (binds tightest)
   2. concatenation
   3. union (binds least of all)

RE's over {0,1}

- Fill in the blank
- $E_1 = 0+11$ then $L(E_1)=$______
- $E_2=(00+01+10+11)^*$ then $L(E_2)=$_____
- $E_3 = 0^*+1^*$ then $L(E_3)=$______
- $E_4 = (00^*+11^*)^*$ then $L(E_4)=$______
- $E_5 = (1+\varepsilon)(01)^*(0+\varepsilon)$ then $L(E_5)=$______
-

Computing a language

- We can compute a language by using the definition of the meaning of a regular expression
- L(a+b.c*) = L(a) U L(bc*)
- L(a+b.c*) = L(a) U (L(b).L(c*))
- L(a+b.c*) = {a} U ({b} . {c}*)
- L(a+b.c*) = {a} U ({b}.{Λ,c,cc,ccc,..., c^n})
- L(a+b.c*) = {a} U ({b,bc,bcc,bccc,..., bc^n})
- L(a+b.c*) = {a,b,bc,bcc,bccc,..., bc^n})

Laws about Regular expressiosn

- The regular expressions form an algebra
- There are many laws (just as there are laws about arithmetic (5+2)=(2+5)

Laws about +

1. $R + T = T + R$
2. $R + \emptyset = \emptyset + R = R$
3. $R + R = R$
4. $R + (S + T) = (R + S) + T$

Laws about .

1. $R . \varnothing = \varnothing . R = \varnothing$
2. $R.\Lambda = \Lambda.R = R$
3. $(R.S).T = R.(S.T)$

- With Implicit .

1. $R \varnothing = \varnothing R = \varnothing$
2. $R\Lambda = \Lambda R = R$
3. $(RS)T = R(ST)$

Distributive Properties

1. $R(S + T) = RS + RT$
2. $(S + T)R = SR + TR$

Closure Properties *

1. $\emptyset^* = \Lambda^* = \Lambda$
2. $R^* = R^*R^* = (R^*)^* = R + R^*$
3. $R^* = \Lambda + R^* = (\Lambda + R)^* = (\Lambda + R)R^* = \Lambda + RR^*$
4. $R^* = (\Lambda + \ldots + R^k)^*$ *for all k >= 1*
5. $R^* = \Lambda + R + \ldots + R^{(k-1)} + R^kR^*$ *for all k >= 1*
6. $RR^* = R^*R$
7. $R(SR)^* = (RS)^*R$
8. $(R^*S)^* = \Lambda + (R + S)^*S$
9. $(RS^*)^* = \Lambda + R(R + S)^*$