Automata and Formal Languages (2)

Email: christian.urban at kcl.ac.uk
Office: S1.27 (1st floor Strand Building)
Slides: KEATS

Languages

A language is a set of strings.

A regular expression specifies a set of strings or language.

Regular Expressions

Their inductive definition:

r	$::=$	$\varnothing$	null
	$\mid$	ϵ	empty string / "" / []
	$\mid$	c	character
	$\mid$	$r_1 \cdot r_2$	sequence
	$\mid$	$r_1 + r_2$	alternative / choice
	$\mid$	r^*	star (zero or more)

Regular Expressions

Their implementation in Scala:

```
1  abstract class Rexp
2
3  case object NULL extends Rexp
4  case object EMPTY extends Rexp
5  case class CHAR(c: Char) extends Rexp
6  case class ALT(r1: Rexp, r2: Rexp) extends Rexp
7  case class SEQ(r1: Rexp, r2: Rexp) extends Rexp
8  case class STAR(r: Rexp) extends Rexp
```

The Meaning of a Regular Expression

$$
\begin{array}{rcl}
L(\varnothing) & \stackrel{\text{def}}{=} & \varnothing \\
L(\epsilon) & \stackrel{\text{def}}{=} & \{""\} \\
L(c) & \stackrel{\text{def}}{=} & \{"c"\} \\
L(r_1 + r_2) & \stackrel{\text{def}}{=} & L(r_1) \cup L(r_2) \\
L(r_1 \cdot r_2) & \stackrel{\text{def}}{=} & L(r_1) \mathbin{@} L(r_2) \\
L(r^*) & \stackrel{\text{def}}{=} & \bigcup_{n \geq 0} L(r)^n
\end{array}
$$

$$
\begin{array}{rcl}
L(r)^0 & \stackrel{\text{def}}{=} & \{""\} \\
L(r)^{n+1} & \stackrel{\text{def}}{=} & L(r) \mathbin{@} L(r)^n
\end{array}
$$

The Meaning of a Regular Expression

$$
\begin{array}{rcl}
L(\varnothing) & \stackrel{\text{def}}{=} & \varnothing \\
L(\epsilon) & \stackrel{\text{def}}{=} & \{""\} \\
L(c) & \stackrel{\text{def}}{=} & \{"c"\} \\
L(r_1 + r_2) & \stackrel{\text{def}}{=} & L(r_1) \cup L(r_2) \\
L(r_1 \cdot r_2) & \stackrel{\text{def}}{=} & L(r_1) \mathbin{@} L(r_2) \\
L(r^*) & \stackrel{\text{def}}{=} & \bigcup_{n \geq 0} L(r)^n
\end{array}
$$

$$
\begin{array}{rcl}
L(r)^0 & \stackrel{\text{def}}{=} & \{""\} \\
L(r)^{n+1} & \stackrel{\text{def}}{=} & L(r) \mathbin{@} L(r)^n
\end{array}
$$

A @ B

. . . you take out every string from A and concatenate it with every string in B

The Meaning of a Regular Expression

$$
\begin{array}{rcl}
L(\varnothing) & \stackrel{\text{def}}{=} & \varnothing \\
L(\epsilon) & \stackrel{\text{def}}{=} & \{""\} \\
L(c) & \stackrel{\text{def}}{=} & \{"c"\} \\
L(r_1 + r_2) & \stackrel{\text{def}}{=} & L(r_1) \cup L(r_2) \\
L(r_1 \cdot r_2) & \stackrel{\text{def}}{=} & L(r_1) \mathbin{@} L(r_2) \\
L(r^*) & \stackrel{\text{def}}{=} & \bigcup_{n \geq 0} L(r)^n
\end{array}
$$

A @ B

. . . you take out every string from A and concatenate it with every string in B

$$
\begin{array}{rcl}
L(r)^0 & \stackrel{\text{def}}{=} & \{""\} \\
L(r)^{n+1} & \stackrel{\text{def}}{=} & L(r) \mathbin{@} L(r)^n
\end{array}
$$

L is a function from regular expressions to sets of strings

L : Rexp $\Rightarrow$ Set[String]

What is $L(a^*)$?

Reg Exp Equivalences

$$
\begin{array}{lrcl}
 & (a + b) + c & \equiv^{?} & a + (b + c) \\
 & a + a & \equiv^{?} & a \\
 & (a \cdot b) \cdot c & \equiv^{?} & a \cdot (b \cdot c) \\
 & a \cdot a & \equiv^{?} & a \\
 & \epsilon^{*} & \equiv^{?} & \epsilon \\
 & \varnothing^{*} & \equiv^{?} & \varnothing \\
\forall\, r. & r \cdot \epsilon & \equiv^{?} & r \\
\forall\, r. & r + \epsilon & \equiv^{?} & r \\
\forall\, r. & r + \varnothing & \equiv^{?} & r \\
\forall\, r. & r \cdot \varnothing & \equiv^{?} & r \\
 & c \cdot (a + b) & \equiv^{?} & (c \cdot a) + (c \cdot b) \\
 & a^{*} & \equiv^{?} & \epsilon + (a \cdot a^{*})
\end{array}
$$

Reg Exp Equivalences

$$(a + b) + c \equiv^{?} a + (b + c) \qquad \text{yes}$$

$$a + a \equiv^{?} a$$

$$(a \cdot b) \cdot c \equiv^{?} a \cdot (b \cdot c)$$

$$a \cdot a \equiv^{?} a$$

$$\epsilon^* \equiv^{?} \epsilon$$

$$\varnothing^* \equiv^{?} \varnothing$$

$$\forall\, r. \quad r \cdot \epsilon \equiv^{?} r$$

$$\forall\, r. \quad r + \epsilon \equiv^{?} r$$

$$\forall\, r. \quad r + \varnothing \equiv^{?} r$$

$$\forall\, r. \quad r \cdot \varnothing \equiv^{?} r$$

$$c \cdot (a + b) \equiv^{?} (c \cdot a) + (c \cdot b)$$

$$a^* \equiv^{?} \epsilon + (a \cdot a^*)$$

Reg Exp Equivalences

$$
\begin{array}{lrcll}
 & (a + b) + c & \equiv^{?} & a + (b + c) & \text{yes} \\
 & a + a & \equiv^{?} & a & \text{yes} \\
 & (a \cdot b) \cdot c & \equiv^{?} & a \cdot (b \cdot c) & \\
 & a \cdot a & \equiv^{?} & a & \\
 & \epsilon^{*} & \equiv^{?} & \epsilon & \\
 & \varnothing^{*} & \equiv^{?} & \varnothing & \\
\forall\, r. & r \cdot \epsilon & \equiv^{?} & r & \\
\forall\, r. & r + \epsilon & \equiv^{?} & r & \\
\forall\, r. & r + \varnothing & \equiv^{?} & r & \\
\forall\, r. & r \cdot \varnothing & \equiv^{?} & r & \\
 & c \cdot (a + b) & \equiv^{?} & (c \cdot a) + (c \cdot b) & \\
 & a^{*} & \equiv^{?} & \epsilon + (a \cdot a^{*}) &
\end{array}
$$

Reg Exp Equivalences

$$
\begin{array}{lrcll}
 & (a + b) + c & \equiv^{?} & a + (b + c) & \text{yes} \\
 & a + a & \equiv^{?} & a & \text{yes} \\
 & (a \cdot b) \cdot c & \equiv^{?} & a \cdot (b \cdot c) & \text{yes} \\
 & a \cdot a & \equiv^{?} & a & \\
 & \epsilon^{*} & \equiv^{?} & \epsilon & \\
 & \varnothing^{*} & \equiv^{?} & \varnothing & \\
\forall\, r. & r \cdot \epsilon & \equiv^{?} & r & \\
\forall\, r. & r + \epsilon & \equiv^{?} & r & \\
\forall\, r. & r + \varnothing & \equiv^{?} & r & \\
\forall\, r. & r \cdot \varnothing & \equiv^{?} & r & \\
 & c \cdot (a + b) & \equiv^{?} & (c \cdot a) + (c \cdot b) & \\
 & a^{*} & \equiv^{?} & \epsilon + (a \cdot a^{*}) &
\end{array}
$$

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	
	$\varnothing^*$	$\equiv^?$	$\varnothing$	
$\forall\, r.$	$r \cdot \epsilon$	$\equiv^?$	r	
$\forall\, r.$	$r + \epsilon$	$\equiv^?$	r	
$\forall\, r.$	$r + \varnothing$	$\equiv^?$	r	
$\forall\, r.$	$r \cdot \varnothing$	$\equiv^?$	r	
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	
$\forall\, r.$	$r \cdot \epsilon$	$\equiv^?$	r	
$\forall\, r.$	$r + \epsilon$	$\equiv^?$	r	
$\forall\, r.$	$r + \varnothing$	$\equiv^?$	r	
$\forall\, r.$	$r \cdot \varnothing$	$\equiv^?$	r	
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall r.$	$r \cdot \epsilon$	$\equiv^?$	r	
$\forall r.$	$r + \epsilon$	$\equiv^?$	r	
$\forall r.$	$r + \varnothing$	$\equiv^?$	r	
$\forall r.$	$r \cdot \varnothing$	$\equiv^?$	r	
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall r.$	$r \cdot \epsilon$	$\equiv^?$	r	yes
$\forall r.$	$r + \epsilon$	$\equiv^?$	r	
$\forall r.$	$r + \varnothing$	$\equiv^?$	r	
$\forall r.$	$r \cdot \varnothing$	$\equiv^?$	r	
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall\, r.$	$r \cdot \epsilon$	$\equiv^?$	r	yes
$\forall\, r.$	$r + \epsilon$	$\equiv^?$	r	no
$\forall\, r.$	$r + \varnothing$	$\equiv^?$	r	
$\forall\, r.$	$r \cdot \varnothing$	$\equiv^?$	r	
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall r.$	$r \cdot \epsilon$	$\equiv^?$	r	yes
$\forall r.$	$r + \epsilon$	$\equiv^?$	r	no
$\forall r.$	$r + \varnothing$	$\equiv^?$	r	yes
$\forall r.$	$r \cdot \varnothing$	$\equiv^?$	r	
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall\, r.$	$r \cdot \epsilon$	$\equiv^?$	r	yes
$\forall\, r.$	$r + \epsilon$	$\equiv^?$	r	no
$\forall\, r.$	$r + \varnothing$	$\equiv^?$	r	yes
$\forall\, r.$	$r \cdot \varnothing$	$\equiv^?$	r	no
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall r.$	$r \cdot \epsilon$	$\equiv^?$	r	yes
$\forall r.$	$r + \epsilon$	$\equiv^?$	r	no
$\forall r.$	$r + \varnothing$	$\equiv^?$	r	yes
$\forall r.$	$r \cdot \varnothing$	$\equiv^?$	r	no
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	yes
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	

Reg Exp Equivalences

	$(a + b) + c$	$\equiv^?$	$a + (b + c)$	yes
	$a + a$	$\equiv^?$	a	yes
	$(a \cdot b) \cdot c$	$\equiv^?$	$a \cdot (b \cdot c)$	yes
	$a \cdot a$	$\equiv^?$	a	no
	ϵ^*	$\equiv^?$	ϵ	yes
	$\varnothing^*$	$\equiv^?$	$\varnothing$	no
$\forall r.$	$r \cdot \epsilon$	$\equiv^?$	r	yes
$\forall r.$	$r + \epsilon$	$\equiv^?$	r	no
$\forall r.$	$r + \varnothing$	$\equiv^?$	r	yes
$\forall r.$	$r \cdot \varnothing$	$\equiv^?$	r	no
	$c \cdot (a + b)$	$\equiv^?$	$(c \cdot a) + (c \cdot b)$	yes
	a^*	$\equiv^?$	$\epsilon + (a \cdot a^*)$	yes

The Meaning of Matching

a regular expression r matches a string s is defined as

$$s \in L(r)$$

The Meaning of Matching

a regular expression r matches a string s is defined as

$$s \in L(r)$$

if $r_1 \equiv r_2$, then $s \in L(r_1)$ iff $s \in L(r_2)$

A Matching Algorithm

- given a regular expression r and a string s, say yes or no for whether

$$s \in L(r)$$

or not.

A Matching Algorithm

- given a regular expression r and a string s, say yes or no for whether

$$s \in L(r)$$

or not.

A Matching Algorithm

- given a regular expression r and a string s, say yes or no for whether

$$s \in L(r)$$

  or not.

- Identifiers (strings of letters or digits, starting with a letter)
- Integers (a non-empty sequence of digits)
- Keywords (else, if, while, ...)
- White space (a non-empty sequence of blanks, newlines and tabs)

A Matching Algorithm

whether a regular expression matches the empty string:

```
1  def nullable (r: Rexp) : Boolean = r match {
2    case NULL => false
3    case EMPTY => true
4    case CHAR(_) => false
5    case ALT(r1, r2) => nullable(r1) || nullable(r2)
6    case SEQ(r1, r2) => nullable(r1) && nullable(r2)
7    case STAR(_) => true
8  }
```

The Derivative of a Rexp

If r matches the string $c::s$, what is a regular expression that matches s?

der c r gives the answer

The Derivative of a Rexp (2)

$$
\begin{array}{lcl}
\text{der c } (\varnothing) & \stackrel{\text{def}}{=} & \varnothing \\
\text{der c } (\epsilon) & \stackrel{\text{def}}{=} & \varnothing \\
\text{der c (d)} & \stackrel{\text{def}}{=} & \text{if c} = \text{d then [] else } \varnothing \\
\text{der c } (r_1 + r_2) & \stackrel{\text{def}}{=} & (\text{der c } r_1) + (\text{der c } r_2) \\
\text{der c } (r_1 \cdot r_2) & \stackrel{\text{def}}{=} & \text{if nullable } r_1 \\
 & & \text{then } ((\text{der c } r_1) \cdot r_2) + (\text{der c } r_2) \\
 & & \text{else } (\text{der c } r_1) \cdot r_2 \\
\text{der c } (r^*) & \stackrel{\text{def}}{=} & (\text{der c } r) \cdot (r^*)
\end{array}
$$

The Derivative of a Rexp (2)

$$
\begin{array}{lcl}
\text{der c } (\varnothing) & \stackrel{\text{def}}{=} & \varnothing \\
\text{der c } (\epsilon) & \stackrel{\text{def}}{=} & \varnothing \\
\text{der c (d)} & \stackrel{\text{def}}{=} & \text{if c} = \text{d then [] else } \varnothing \\
\text{der c } (r_1 + r_2) & \stackrel{\text{def}}{=} & (\text{der c } r_1) + (\text{der c } r_2) \\
\text{der c } (r_1 \cdot r_2) & \stackrel{\text{def}}{=} & \text{if nullable } r_1 \\
 & & \text{then } ((\text{der c } r_1) \cdot r_2) + (\text{der c } r_2) \\
 & & \text{else } (\text{der c } r_1) \cdot r_2 \\
\text{der c } (r^*) & \stackrel{\text{def}}{=} & (\text{der c } r) \cdot (r^*) \\
\text{ders [] r} & \stackrel{\text{def}}{=} & r \\
\text{ders (c::s) r} & \stackrel{\text{def}}{=} & \text{ders s (der c r)}
\end{array}
$$

The Derivative

```
1  def deriv (r: Rexp, c: Char) : Rexp = r match {
2    case NULL => NULL
3    case EMPTY => NULL
4    case CHAR(d) => if (c == d) EMPTY else NULL
5    case ALT(r1, r2) => ALT(deriv(r1, c), deriv(r2, c))
6    case SEQ(r1, r2) =>
7      if (nullable(r1)) ALT(SEQ(deriv(r1, c), r2), deriv(r2, c))
8      else SEQ(deriv(r1, c), r2)
9    case STAR(r) => SEQ(deriv(r, c), STAR(r))
10 }
```

The Rexp Matcher

```
1  def matches(r: Rexp, s: String) : Boolean =
2    nullable(derivs(r, s.toList))
3
4
5  /* Examples */
6
7  println(matches(SEQ(SEQ(CHAR('c'), CHAR('a')), CHAR('b')),"cab"))
8  println(matches(STAR(CHAR('a')),"aaa"))
9
10 /* Convenience using implicits */
11 implicit def string2rexp(s : String) : Rexp = {
12   s.foldRight (EMPTY: Rexp) ( (c, r) => SEQ(CHAR(c), r) )
13 }
14
15 println(matches("cab" ,"cab"))
16 println(matches(STAR("a"),"aaa"))
17 println(matches(STAR("a"),"aaab"))
```

Proofs about Rexp

Remember their inductive definition:

$$
\begin{array}{rcl}
r & ::= & \varnothing \\
 & | & \epsilon \\
 & | & c \\
 & | & r_1 \cdot r_2 \\
 & | & r_1 + r_2 \\
 & | & r^*
\end{array}
$$

If we want to prove something, say a property $P(r)$, for all regular expressions r then ...

Proofs about Rexp (2)

- P holds for $\varnothing$, ϵ and c
- P holds for $r_1 + r_2$ under the assumption that P already holds for r_1 and r_2.
- P holds for $r_1 \cdot r_2$ under the assumption that P already holds for r_1 and r_2.
- P holds for r^* under the assumption that P already holds for r.

Proofs about Rexp (3)

Assume $P(r)$ is the property:

nullable(r) if and only if "" $\in L$(r)

Proofs about Strings

If we want to prove something, say a property $P(s)$, for all strings s then ...

- P holds for the empty string, and
- P holds for the string $c::s$ under the assumption that P already holds for s

Proofs about Strings (2)

Let Der c A be the set defined as

$$\text{Der c A} \stackrel{\text{def}}{=} \{ \text{s} \mid \text{c::s} \in \text{A} \}$$

Assume that L(der c r) = Der c (L(r)). Prove that

matcher(r, s) if and only if s $\in$ L(r)

Regular Languages

A language (set of strings) is regular iff there exists a regular expression that recognises all its strings.

Automata

A deterministic finite automaton consists of:

- a set of states
- one of these states is the start state
- some states are accepting states, and
- there is transition function

  which takes a state as argument and a character and produces a new state

  this function might not always be defined