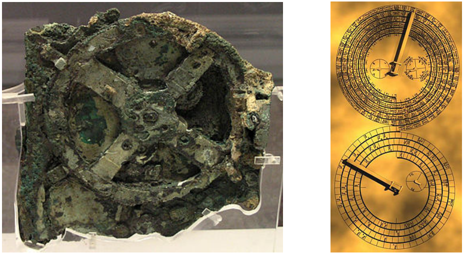
# Automata and Formal Languages (2)



Antikythera automaton, 100 BC (Archimedes?)


Email: christian.urban at kcl.ac.uk

Office: S1.27 (1st floor Strand Building)


Slides: KEATS

# Regular Expressions

Their inductive definition:

| | | | |
|---|---|---|---|
| $r$ | $::=$ | $\varnothing$ | null |
| | $\mid$ | $\epsilon$ | 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) |

# The Meaning of a Regular Expression

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

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

# 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

| | | | | |
|---|---|---|---|---|
| | $(a + b) + c$ | $\equiv^?$ | $a + (b + c)$ | yes |
| | $a + a$ | $\equiv^?$ | $a$ | |
| | $(a \cdot b) \cdot c$ | $\equiv^?$ | $a \cdot (b \cdot c)$ | |
| | $a \cdot a$ | $\equiv^?$ | $a$ | |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | |
| | $\varnothing^*$ | $\equiv^?$ | $\varnothing$ | |
| $\forall\, r.$ | $r \cdot \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \varnothing$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r \cdot \varnothing$ | $\equiv^?$ | $r$ | |
| | $c \cdot (a + b)$ | $\equiv^?$ | $(c \cdot a) + (c \cdot b)$ | |
| | $a^*$ | $\equiv^?$ | $\epsilon + (a \cdot a^*)$ | |

# Reg Exp Equivalences

| | | | | |
|---|---|---|---|---|
| | $(a + b) + c$ | $\equiv^?$ | $a + (b + c)$ | yes |
| | $a + a$ | $\equiv^?$ | $a$ | yes |
| | $(a \cdot b) \cdot c$ | $\equiv^?$ | $a \cdot (b \cdot c)$ | |
| | $a \cdot a$ | $\equiv^?$ | $a$ | |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | |
| | $\varnothing^*$ | $\equiv^?$ | $\varnothing$ | |
| $\forall\, r.$ | $r \cdot \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \varnothing$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r \cdot \varnothing$ | $\equiv^?$ | $r$ | |
| | $c \cdot (a + b)$ | $\equiv^?$ | $(c \cdot a) + (c \cdot b)$ | |
| | $a^*$ | $\equiv^?$ | $\epsilon + (a \cdot a^*)$ | |

# 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 |
| | $a \cdot a$ | $\equiv^?$ | $a$ | |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | |
| | $\varnothing^*$ | $\equiv^?$ | $\varnothing$ | |
| $\forall\, r.$ | $r \cdot \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \varnothing$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r \cdot \varnothing$ | $\equiv^?$ | $r$ | |
| | $c \cdot (a + b)$ | $\equiv^?$ | $(c \cdot a) + (c \cdot b)$ | |
| | $a^*$ | $\equiv^?$ | $\epsilon + (a \cdot a^*)$ | |

# 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | |
| | $\varnothing^*$ | $\equiv^?$ | $\varnothing$ | |
| $\forall\, r.$ | $r \cdot \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \epsilon$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r + \varnothing$ | $\equiv^?$ | $r$ | |
| $\forall\, r.$ | $r \cdot \varnothing$ | $\equiv^?$ | $r$ | |
| | $c \cdot (a + b)$ | $\equiv^?$ | $(c \cdot a) + (c \cdot b)$ | |
| | $a^*$ | $\equiv^?$ | $\epsilon + (a \cdot a^*)$ | |

# 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | 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

| | | | | |
|---|---|---|---|---|
| | $(a + b) + c$ | $\stackrel{?}{\equiv}$ | $a + (b + c)$ | yes |
| | $a + a$ | $\stackrel{?}{\equiv}$ | $a$ | yes |
| | $(a \cdot b) \cdot c$ | $\stackrel{?}{\equiv}$ | $a \cdot (b \cdot c)$ | yes |
| | $a \cdot a$ | $\stackrel{?}{\equiv}$ | $a$ | no |
| | $\epsilon^*$ | $\stackrel{?}{\equiv}$ | $\epsilon$ | yes |
| | $\varnothing^*$ | $\stackrel{?}{\equiv}$ | $\varnothing$ | no |
| $\forall\, r.$ | $r \cdot \epsilon$ | $\stackrel{?}{\equiv}$ | $r$ | yes |
| $\forall\, r.$ | $r + \epsilon$ | $\stackrel{?}{\equiv}$ | $r$ | no |
| $\forall\, r.$ | $r + \varnothing$ | $\stackrel{?}{\equiv}$ | $r$ | |
| $\forall\, r.$ | $r \cdot \varnothing$ | $\stackrel{?}{\equiv}$ | $r$ | |
| | $c \cdot (a + b)$ | $\stackrel{?}{\equiv}$ | $(c \cdot a) + (c \cdot b)$ | |
| | $a^*$ | $\stackrel{?}{\equiv}$ | $\epsilon + (a \cdot a^*)$ | |

# 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | 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 |
| | $\epsilon^*$ | $\equiv^?$ | $\epsilon$ | 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

| | | | | |
|---|---|---|---|---|
| | $(a + b) + c$ | $\stackrel{?}{\equiv}$ | $a + (b + c)$ | yes |
| | $a + a$ | $\stackrel{?}{\equiv}$ | $a$ | yes |
| | $(a \cdot b) \cdot c$ | $\stackrel{?}{\equiv}$ | $a \cdot (b \cdot c)$ | yes |
| | $a \cdot a$ | $\stackrel{?}{\equiv}$ | $a$ | no |
| | $\epsilon^*$ | $\stackrel{?}{\equiv}$ | $\epsilon$ | yes |
| | $\varnothing^*$ | $\stackrel{?}{\equiv}$ | $\varnothing$ | no |
| $\forall\, r.$ | $r \cdot \epsilon$ | $\stackrel{?}{\equiv}$ | $r$ | yes |
| $\forall\, r.$ | $r + \epsilon$ | $\stackrel{?}{\equiv}$ | $r$ | no |
| $\forall\, r.$ | $r + \varnothing$ | $\stackrel{?}{\equiv}$ | $r$ | yes |
| $\forall\, r.$ | $r \cdot \varnothing$ | $\stackrel{?}{\equiv}$ | $r$ | no |
| | $c \cdot (a + b)$ | $\stackrel{?}{\equiv}$ | $(c \cdot a) + (c \cdot b)$ | yes |
| | $a^*$ | $\stackrel{?}{\equiv}$ | $\epsilon + (a \cdot a^*)$ | 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

$$s \in L(r)$$

# A Matching Algorithm

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

$$s \in L(r)$$

- foo

# A Matching Algorithm

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

$$s \in L(r)$$

- bar

# A Matching Algorithm

whether a regular expression matches the empty string:

```
def nullable (r: Rexp) : Boolean = r match {
  case NULL => false
  case EMPTY => true
  case CHAR(_) => false
  case ALT(r1, r2) => nullable(r1) || nullable(r2)
  case SEQ(r1, r2) => nullable(r1) && nullable(r2)
  case STAR(_) => true
}
```

# 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

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

# The Derivative

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

# The Derivative

```
def deriv (r: Rexp, c: Char) : Rexp = r match {
  case NULL => NULL
  case EMPTY => NULL
  case CHAR(d) => if (c == d) EMPTY else NULL
  case ALT(r1, r2) => ALT(deriv(r1, c), deriv(r2, c))
  case SEQ(r1, r2) =>
    if (nullable(r1)) ALT(SEQ(deriv(r1, c), r2), deriv(r2, c))
    else SEQ(deriv(r1, c), r2)
  case STAR(r) => SEQ(deriv(r, c), STAR(r))
}
```

# The Rexp Matcher

```
def matches(r: Rexp, s: String) : Boolean =
  nullable(derivs(r, s.toList))


/* Examples */

println(matches(SEQ(SEQ(CHAR('c'), CHAR('a')), CHAR('b')),"cab"))
println(matches(STAR(CHAR('a')),"aaa"))

/* Convenience using implicits */
implicit def string2rexp(s : String) : Rexp = {
  s.foldRight (EMPTY: Rexp) ( (c, r) => SEQ(CHAR(c), r) )
}

println(matches("cab" ,"cab"))
println(matches(STAR("a"),"aaa"))
println(matches(STAR("a"),"aaab"))
```

# This Course

We will have a look at:

- regular expressions / regular expression matching
- automata
- the Myhill-Nerode theorem
- parsing
- grammars
- a small interpreter / web browser

# Exam

- The question "Is this relevant for the exam?" is not appreciated!

  Whatever is in the homework sheets (and is not marked optional) is relevant for the exam.
  No code needs to be written.

# Maps in Scala

- **map** takes a function, say f, and applies it to every element of the list:

```
List(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

```
List(1, 4, 9, 16, 25, 36, 49, 64, 81)
```