Regular Expressions: Motivation

Consider the following (beautiful) Sage code:

\[
\begin{align*}
x &= 11^2 \\
galoisf121 &= \text{GF}(x)
\end{align*}
\]

If you’re the Sage interpreter, how do you recognize the variable names? How do you know what a number should look like?
For a simple example, suppose we want to recognize integers.

- An integer may begin with a - sign.
- The first digit of an integer is a 1-9.
- Subsequent digits may be 0-9.
Regular Expressions: FSM

**Integer-Recognizing State Machine**

**State 0:** If next input is a - go to State 1. If 1-9, go to State 2. Otherwise remain.

**State 1:** If next input is a 1-9, go to State 2. Otherwise, go to State 0.

**State 2:** If next input is a 0-9, remain. Otherwise report the observed integer and go to State 0.
A character literal matches against itself. E.g. $a$ matches an "a".

Character literals can be concatenated. *apotheosis* matches "apotheosis".

$a|b$ matches "a" or "b".

$a^*$ matches a sequence of 0 or more "a"s.

We can rewrite the Integer-Recognizing State Machine as $(\lnot)(1|2|3|4|5|6|7|8|9)(0|1|2|3|4|5|6|7|8|9)^*.$
Shorthands are used for large chains of |. For example, in most regular expression systems \([1 – 9]\) captures any numeral from 1 to 9.

Additional operations can be defined using the basic ones. For example + is used to indicate 1 or more, a shorthand for \(aa^*\). ? is used to indicate 0 or 1, so \(a?\) is equivalent to (\(|a\)).

With these conveniences, we can rewrite the IRSM as 
\(-?\[1 – 9\][0 – 9]^*\).
A *word* is a possibly-empty sequence of inputs from some alphabet $A$.

An *event* is a set of words.

The operation $|$ is defined as set-theoretic union $\cup$.

Concatenation is defined as $AB = \{ab \mid a \in A, b \in B\}$.

Define 0 to be the empty event and 1 to be the event containing only the empty word.

Exponentiation is $A^0 = 1$, $A^n = AA^{n-1}$.

$A^* = A^0 \cup A^1 \cup A^2 \cup \ldots$.

Any event that can be constructed using only the primitives, $|$, concatenation, and $*$ is a *regular event*.
What’s a Kleene Algebra?

- Kleene Algebras are an attempt to generalize the properties of Regular Expressions.
- A Kleene Algebra consists of a set $K$ with 3 operations.
- Binary operations: $+$, $\cdot$.
- Unary operation: $\ast$.
- Special elements: 0, 1.
Kleene Algebra Axioms: + and ·

- $a + (b + c) = (a + b) + c$
- $a + b = b + a$
- $a + a = a$
- $a + 0 = a$
- $a(bc) = (ab)c$
- $1a = a1 = a$
- $0a = a0 = 0$
- $(a + b)c = ac + bc$
- $a(b + c) = ab + ac$
Define a partial order on $K$ as $a \leq b$ if $a + b = b$.

- $1 + aa^* \leq a^*$
- $1 + a^*a \leq a^*$
- $ax \leq x \implies a^*x \leq x$
- $xa \leq x \implies xa^* \leq x$
Kleene Algebra Properties

- $1 \leq a^*$
- $a \leq a^*$
- $a \leq b \implies ac \leq bc$
- $a \leq b \implies ca \leq cb$
- $a \leq b \implies a + c \leq b + c$
- $a \leq b \implies a^* \leq b^*$
- $1 + a + a^* a^* = a^*$
- $a^{**} = a^*$
- $0^* = 1$
- $1 + aa^* = a^*$
- $1 + a^* a = a^*$
- $b + ax \leq x \implies a^* b \leq x$
- $b + xa \leq x \implies ba^* \leq x$
- $ax = xb \implies a^* x = xb^*$
- $(cd)^* c = c(d c)^*$
- $(a + b)^* = a^* (b a^*)^*$
Matrices

- The set of matrices over a Kleene algebra is a Kleene algebra.
- + and · are just matrix addition and multiplication.
- * is defined as

\[
E = \begin{bmatrix}
a & b \\
c & d
\end{bmatrix}
\]

\[
E^* = \begin{bmatrix}
(a + bd^* c)^* & (a + bd^* c)^* bd^* \\
d^* c (a + bd^* c)^* & d^* + d^* c (a + bd^* c)^* bd^*
\end{bmatrix}
\]
Fact

Any element of a Kleene algebra can be used to construct a corresponding state machine.
Kleene Algebra with Tests

- A Kleene Algebra is a Kleene Algebra with Tests if it has a subset $B$ that is a Boolean Algebra with $+$ as the meet and $\cdot$ as the join.
- This implies that a complement operator $'$ is defined for members of $B$.
- This allows encoding of conditionals. For example, if $a$ then $b$ else $c$ can be encoded as $ab + a'c$.

- Loops can also be encoded. while $a$, $b$ is encoded as $(ab)^* a'$.
- This allows the description of more complicated programs.
References I

  *Regular algebra and finite machines.*  
  Courier Corporation, 2012.

- Dexter Kozen.  
  A completeness theorem for kleene algebras and the algebra of regular events.  

- Dexter Kozen.  
  On kleene algebras and closed semirings.  

References II

Peter Höfner and Bernhard Möller.  
Dijkstra, floyd and warshall meet kleene.  

Dexter Kozen.  
Kleene algebra with tests.  
Questions?