Kleene Algebra (KA) is the algebra of regular expressions

\[ pq + qp \]

\[ p^* = \{p, pq, pqp, pqpq, \ldots\} \]

\[ (p + q)^* = (p\cdot q)^*p \cdot (q\cdot p)^* \]

\[ \{\text{all strings over } p, q\} \]

\[ p^* = \{\varepsilon\} \]

\[ 0 = \emptyset \]

Binary Relations

\( R, S \) binary relations on a set \( X \)

\[ R + S = R \cup S \]

\[ RS = R^* \cdot S = \{(u,v) \mid \exists w (u,w) \in R, (w,v) \in S\} \]

\[ R^* = \text{reflexive transitive closure of } R \]

\[ 1 = \text{identity relation} = \{(u,u) \mid u \in X\} \]

\[ 0 = \emptyset \]

Standard Interpretation

Regular sets over \( \Sigma \)

\[ A + B = A \cup B \]

\[ AB = \{xy \mid x \in A, y \in B\} \]

\[ A^* = \bigcup_{n=0}^{\infty} A^n = A^0 \cup A^1 \cup A^2 \cup \ldots \]

\[ 1 = \{\varepsilon\} \]

\[ 0 = \emptyset \]

\[ p \in \Sigma \text{ interpreted as } \{p\} \]

Applications

- Automata and formal languages
  - regular expressions
- Relational algebra
- Program logic and verification
  - Dynamic Logic
- Program analysis
- Optimization
- Design and analysis of algorithms
  - shortest paths
  - connectivity

Fundamental Questions

- Axiomatization of equational theory
  - [Salomaa 66]
- ...but no finite equational axiomatization
  - [Redko 64]
- Complexity = PSPACE complete
  - [Stock 74]

Axioms of KA [K91]

- \( K \) is an idempotent semiring under \( +, \cdot, 0, 1 \)

\[ (p + q) + r = p + (q + r) \]

\[ p\cdot q = q\cdot p \]

\[ p + p = p \]

\[ p + 0 = p \]

\[ p(q + r) = pq + pr \]

\[ (p + q) + r = pr + qr \]

- \( p^* q = \text{least } x \text{ such that } q + px \leq x \)
- \( qp^* = \text{least } x \text{ such that } q + xp \leq x \)

\[ x \leq y \text{ def } x + y = y \]
This is a universal Horn axiomatization

\[ p \cdot q = \text{least } x \text{ such that } q + px \leq x \]

\[ q + p(p \cdot q) \leq p \cdot q \]

\[ q + px \leq x \rightarrow p \cdot q \leq x \]

Every system of linear inequalities

\[ a_{11}x_1 + \ldots + a_{1n}x_n + b_1 \leq x_1 \]

\[ \ldots \]

\[ a_{m1}x_1 + \ldots + a_{mn}x_n + b_m \leq x_n \]

has a unique least solution

Alternative Characterizations of *

Complete semirings

\[ \sum_{i \in I} p_i = \text{supremum of } \{p_i \mid i \in I\} \]

with respect to \( \leq \)

\( \ast \)-continuity

\[ pq^\ast r = \text{sup } pq^r \]

infinitary

\[ \ast \text{ same equational theory } Eq(KA) = Eq(KA^\ast) \]

Some Useful Properties

1 + pp\* = 1 + p\*p = p\*

\[ p^p = p^p = p^p \]

\[ (pq\cdot p = p(q)p) \text{ sliding} \]

\[ (p^q\cdot p^r = (p + q)^\ast \text{ denesting} \]

\[ px = xq \rightarrow p^\ast x = xq^\ast \text{ bisimulation} \]

\[ qp = 0 \rightarrow (p + q)^\ast = p^q^\ast \text{ loop distribution} \]

Equational Completeness \([K91]\)

\[ \text{Reg}_\Sigma, \text{ the KA regular sets over } \Sigma, \text{ is the} \]

\[ \text{free KA on generators } \Sigma \]

\[ p = q \text{ as regular sets} \]

\[ p = q \text{ is a consequence of the KA axioms} \]

\[ \text{KA is complete over relational models} \]

\[ Eq(REL) = Eq(KA) = Eq(Reg_\Sigma) \]

Proof of the Sliding Rule

\[ (ab)^\ast a \leq a(ba)^\ast \]

\[ a + aba(ba)^\ast = a(1 + ba(ba)^\ast) \text{ distributivity} \]

\[ = a(ba)^\ast 1 + pp^\ast = p^\ast. \]

\[ a + aba(ba)^\ast \leq a(ba)^\ast \]

\[ (ab)^\ast a \leq a(ba)^\ast \]

\[ q + px \leq x \rightarrow p^\ast q \leq x \]

The reverse inequality \( \geq \) is symmetric.

Matrices over a KA

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix} +
\begin{pmatrix}
e & f \\
g & h
\end{pmatrix} =
\begin{pmatrix}
a+e & b+f \\
c+g & d+h
\end{pmatrix}
\]

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix} \cdot 
\begin{pmatrix}
e & f \\
g & h
\end{pmatrix} =
\begin{pmatrix}
ae+bg & af+bh \\
cc+dg & cf+dh
\end{pmatrix}
\]

\[0 = 
\begin{pmatrix}
0 & 0 \\
0 & 0
\end{pmatrix}
\]

\[1 = 
\begin{pmatrix}
1 & 0 \\
0 & 1
\end{pmatrix}
\]

\[\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}^\ast =
\begin{pmatrix}
(a+bd^\ast)^\ast \\
(d+ca^\ast)^\ast
\end{pmatrix}
\]

\[\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}^\ast =
\begin{pmatrix}
(a+bd^\ast)^\ast \\
(d+ca^\ast)^\ast
\end{pmatrix}
\]

\[\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}^\ast =
\begin{pmatrix}
(a+bd^\ast)^\ast \\
(d+ca^\ast)^\ast
\end{pmatrix}
\]
Matrices over a KA

\[
\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} (a + bd^*c)^* & (a + bd^*c)^*bd^* \\ (d + ca^*b)^*ca^* & (d + ca^*b)^* \end{pmatrix}
\]

Solution of Linear Inequalities

\[
a_1 x_1 + \cdots + a_n x_n + b_1 \leq x_1 \\
\vdots \\
a_{m1} x_1 + \cdots + a_{mn} x_n + b_n \leq x_n
\]

\[
\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \leq \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}
\]

Shortest Paths

The min, + algebra

\[
R_r \cup \{1\} \\
r + s = \min r, s \\
rs = r + s \\
r^* = 0 \\
0 = \infty \\
1 = 0 \\
\leq = \geq
\]

Other Models

Convex polyhedra [Iwano & Steiglitz 90]

\[AB = \{ax + by \mid x \in A, y \in B\}\]

\[A^* = \text{convex hull of } A\]

Kleene Algebra with Tests (KAT)

\[(K, B, +, \cdot, *, \cdot, 0, 1)\]

\[\text{• } (K, +, \cdot, *, 0, 1) \text{ is a Kleene algebra}\]

\[\text{• } (B, +, \cdot, *, 0, 1) \text{ is a Boolean algebra}\]

\[\text{• } \mathcal{B} \subseteq K\]

\[\text{• } p, q, r, \ldots \text{ range over } K\]

\[\text{• } a, b, c, \ldots \text{ range over } B\]
Kleene Algebra with Tests (KAT)

+, ·, 0, 1 serve double duty

- applied to programs, denote choice, composition, fail, and skip, resp.
- applied to tests, denote disjunction, conjunction, falsity, and truth, resp.
- these usages do not conflict!

\[ bc = b \land c \quad \text{and} \quad b + c = b \lor c \]

Models

- Relational models
  \( K \) = binary relations on a set \( X \)
  \( B \) = subsets of the identity relation

- Trace models
  \( K \) = sets of traces \( u_0p_0u_1p_1u_2 ... u_np_nu_n \)
  \( B \) = sets of traces of length 0

- Language-theoretic models
  \( K \) = regular sets of guarded strings over \( \Sigma \)
  \( B \) = atoms of a finite free Boolean algebra

Kripke Frames

\( K = (K, m_K) \)

\[ m_K : \{ \text{atomic programs} \} \to 2^{K \times K} \]

\[ m_K : \{ \text{atomic tests} \} \to 2^K \]

Relational Models

\( K = (K, m_K) \)

\[ m_K : \{ \text{atomic programs} \} \to 2^{K \times K} \]

\[ m_K : \{ \text{atomic tests} \} \to 2^K \]

\[ [p]_K = m_K(p) \]

\[ [b]_K = \{ (u,u) | u \in K \} \]

\[ [p*]_K = \text{reflexive transitive closure of } [p]_K \]

Trace Models

\( K = (K, m_K) \)

\[ m_K : \{ \text{atomic programs} \} \to 2^{K \times K} \]

\[ m_K : \{ \text{atomic tests} \} \to 2^K \]

A trace is a sequence

\[ x = u_0p_0u_1p_1u_2 ... u_n ... u_n, n \geq 0 \]

\( m_K(p) \)

\( \text{first}(x) = u_0, \text{last}(x) = u_n \)

Product \( xy \) exists iff \( \text{last}(x) = \text{first}(y) \)

\[ (u_0p_0u_1p_1u_2 ... u_n) \cdot (u_n ... u_n) = u_0p_0u_1p_1u_2 ... u_n ... u_n \]

Trace Models

\( K = (K, m_K) \)

\[ m_K : \{ \text{atomic programs} \} \to 2^{K \times K} \]

\[ m_K : \{ \text{atomic tests} \} \to 2^K \]

\[ [[p]]_K = \{ (u,v) | u \in m_K(p) \} \]

\[ [b]_K = \text{atoms of a finite free Boolean algebra} \]
**Guarded Strings** [Kaplan 69]

- **P**: atomic programs  
- **B**: atomic tests

\[ \alpha, \beta, \ldots \text{ atoms (minimal nonzero elements) of the free Boolean algebra on generators } B \]

*Example:* If \( B = \{b_1, \ldots, b_6\} \), then \( b_1b_2b_3b_4b_5b_6 \) is an atom.

- Guarded strings: \( \alpha_0p_0\alpha_1p_1\alpha_2p_2\alpha_3 \ldots \alpha_{n-1}p_{n-1}\alpha_n \)

- \( A + B = A \cup B \)
- \( AB = \{x \alpha y \mid x \alpha \in A, y \alpha \in B \} \)
- \( A^* = \bigcup_{n \geq 0} A^n \)
- \( 1 = \{\text{atoms}\} \)
- \( 0 = \emptyset \)

**Theorem** [Kozen & Smith 96]

The family of regular sets of guarded strings over \( P, B \) is the free KAT on generators \( P, B \).

**Corollary**

KAT is complete over relational models.

\[ \text{Eq(GS)} = \text{Eq(KAT)} = \text{Eq(KAT*)} = \text{Eq(REL)} \]

**Matrices over a KAT**

The \( n \times n \) matrices over a KAT \((K, B)\) form a KAT \((K', B')\).

\[ B' = \text{diagonal matrices over } B \]

**Modeling Programs**

Same as in PDL [Fischer & Ladner 79]

- \( p ; q = pq \)
- \( \text{if } b \text{ then } p \text{ else } q = bp + bq \)
- \( \text{while } b \text{ do } p = (bp)^*b \)

**Propositional Hoare Logic (PHL)**

Hoare Logic without the assignment rule

\( \{b[x/t]\} x := t \{b\} \)

Is a given rule

\[ \frac{\{b_1\}p_1\{c_1\}, \ldots, \{b_n\}p_n\{c_n\}}{(b) p(c)} \]

- A logical consequence of the composition, conditional, while, and weakening rules?
- Relationally valid?

**KAT subsumes PHL**

\( \{b\} p(c) \text{ modeled by } bp = bpc \text{ or } bpc = 0 \)

[Manes & Arbib 86]
Theorem
These are all theorems of KAT

Completeness Theorem [K 99]
All relationally valid rules of the form
\[
(b)p(c), (c)q(d) \\
(b)pq(d)
\]
\[
= bp = 0 \land \exists (cq) = 0 \rightarrow bpq = 0
\]

Conditional rule
\[
(bc)p(d), (bc)q(d) \\
(c)if b then p else q(d)
\]
\[
= bcp = 0 \land \exists (cq) = 0 \rightarrow c(bp + bq) = 0
\]

While rule
\[
(bc)p(c) \\
(c)while b do p(bc)
\]
\[
= bcp = 0 \rightarrow c(bp + bq)c = 0
\]

Counterexample
\[
(c)if b then p else p(c) \\
(c)p(c)
\]
is trivially unprovable in Hoare Logic, but
\[
c(bp + bp)c = 0 \rightarrow cp = 0
\]
is easily provable in KAT

Hoare formulas
\[
p_1 = 0 \land p_2 = 0 \land \ldots \land p_n = 0 \rightarrow q = r
\]

Theorem
KAT is complete for the Hoare theory of relational algebras
... not for the Horn theory!
Counterexample: \( p \leq 1 \rightarrow p^2 = p \)

Complexity of KAT and PHL
Theorem [Cohen 94]
The Hoare theory of KA (Horn formulas with premises \( p = 0 \)) is PSPACE-complete

Theorem [Cohen, Kozen & Smith 96]
The Hoare theory of KAT is PSPACE-complete

Theorem
PHL is PSPACE-complete
Typed KAT

- Extend the type discipline of KA to KAT
  \[\text{test} \Rightarrow \text{typecast or coercion operator}\]
- Hoare Logic is subsumed by the type discipline of typed KAT

Thus Hoare-style reasoning with partial correctness assertions is just typechecking.

Typed Kleene Algebra \([K\ 98]\)

\[
ax = xb \Rightarrow a^* x = xb^*
\]

Typed Kleene Algebra

\[
p: b \rightarrow c \quad qb \rightarrow c \\
p + q: b \rightarrow c \\
pq: b \rightarrow d\\n0: b \rightarrow c \\
1: b \rightarrow b \\
p: b \rightarrow b \\
p^*: b \rightarrow b \\
1: b \rightarrow b0: b \rightarrow c \\
c: b \rightarrow bc
\]

Typed KAT

\[
p: b \rightarrow c \quad q: b \rightarrow c \\
p + q: b \rightarrow c \\
pq: b \rightarrow d\\n0: b \rightarrow c \\
1: b \rightarrow b \\
p: b \rightarrow b \\
p^*: b \rightarrow b \\
c: b \rightarrow bc
\]

typecast or coercion

Typecast operator \(c: b \rightarrow bc\)

class Super {}
class Sub extends Super {}
...
void f(Super y) {
    Sub x = null;
    try {
        x = (Sub)y;
    } catch (ClassCastException e) {} 
}
... 
    f(new Sub());
\[
\{b \land c\}p\{c\} \\
\text{(c)while } b \text{ do } p\{\neg b \land c\} \quad \Rightarrow \quad p;bc \rightarrow c \\
\]

\[
b,c \rightarrow bc \quad p;bc \rightarrow c \\
p;bc \rightarrow c \\
bp;bc \rightarrow c \\
(bp)^*bc \rightarrow bc \\
(bp)^*b;bc \rightarrow bc
\]

**Special Cases**

\[
x := s; y := t \\
\quad y := t; x := s \quad (x \notin \text{Var}(t), y \notin \text{Var}(s))
\]

\[
x := t; b := b; x := t \quad (x \notin \text{Var}(b))
\]

\[
x := s; x := a; x := a \quad (x \notin \text{Var}(s))
\]

\[
x = x = s; x := a
\]

**Encoding the Hoare Assignment Axiom**

\[
x := t; b \Rightarrow b[x/t]; x := t
\]

is equivalent to

\[
(b[x/t]) x := t (b) \quad (\exists x[t]) x := t (\exists)
\]

\[
b_{p} = p c \leftrightarrow b p = p c \leftrightarrow b_{p} + b p c = 0
\]

**Scheme Equivalence**

Example of Paterson from [Manna 74]