Kleene Algebra

Georg Struth

University of Sheffield

based on joint work with Jules Desharnais, Bernhard Möller and others
Motivation

**program/system analysis** requires formalisms that balance

- expressive interoperable *modelling languages*
- powerful *proof procedures*

**modelling languages**: e.g.

- relations used in Z or B
- functions/quantales used in refinement calculi
- modal logics/process algebras used for reactive/concurrent systems

**proof procedures** dominated by

- interactive proof checking
- model checking
Motivation

**questions:** are there formalisms that offer better balance

- unify/integrate relational, functional, modal reasoning?
- allow using off-the-shelf automated theorem provers (ATP systems)?
Motivation

**questions:** are there formalisms that offer better balance

- unify/integrate relational, functional, modal reasoning?
- allow using off-the-shelf automated theorem provers (ATP systems)?

**answer:** algebraic methods, in particular *modal Kleene algebras* (maybe)

**benefits** of algebraic approach:

- simple first-order equational calculi
- rich class of computationally meaningful models
- mechanisms for abstraction and (de)composition
- suitable for ATP systems
This Lecture Series

**goal:** introduce **modal Kleene algebras** as computational tools for modelling and analysing discrete dynamical systems

**outline:**

1. surveys foundations of (modal) Kleene algebras
2. discusses some computationally interesting models
3. sketches connection with popular computational logics
4. presents some (automation) examples

**dual rôle of ATP:** a new approach to

- computer mathematics: develop/analyse algebraic structures
- formal methods: develop/analyse programs and systems

**apology:** highly subjective and incomplete picture
Semirings, Actions and Propositions

**semiring:** \((S, +, \cdot, 0, 1)\)  “ring without minus”

\[
\begin{align*}
  x + (y + z) &= (x + y) + z &
  x + y &= y + x &
  x + 0 &= x \\
  x(yz) &= (xy)z &
  x1 &= x &
  1x &= x \\
  x(y + z) &= xy + xz &
  (x + y)z &= xz + yz \\
  x0 &= 0 &
  0x &= 0
\end{align*}
\]

**interpretation:** \(S\) represents **actions** of some discrete dynamical system

- \(+\) models nondeterministic (angelic) choice  (cf. next slide)
- \(\cdot\) models sequential composition
- \(0\) models abortive action
- \(1\) models ineffective action
Semirings, Actions and Propositions

remarks:

• swapping multiplication yields opposite semiring
• semiring is idempotent if \( x + x = x \)
• idempotent semirings are naturally ordered by \( x \leq y \iff x + y = y \) hence \((S, +, 0)\) is upper semilattice with least element 0
• idempotency turns addition into choice

questions:

• how can the state space of the system be included?
• how can the “limit behaviour” of the system be described?
Semirings, Actions and Propositions

**task:** include the state space

**test algebras:** [ManesArbib] “Boolean centre”

- Boolean subalgebra \((\text{test}(S), +, \cdot, \neg, 0, 1)\) embedded into \([0, 1]\) of \(S\)
- \(+\) coincides with Boolean join
- \(\cdot\) coincides with Boolean meet

**remarks:**
- Boolean algebra \(\text{test}(S)\) captures the main intuition behind state spaces
- elements of \(\text{test}(S)\) are sets of states
- alternative interpretations as propositions of a system or tests of a program

**notation:** \(x, y, z \ldots\) for actions; \(p, q, r, \ldots\) for tests/propositions
Kleene Algebras

**task:** describe “limit behaviour”

**Kleene algebras:** [Kozen] idempotent semiring with star satisfying

- **unfold axiom** \(1 + xx^* \leq x^*\)
- **induction axiom** \(y + xz \leq z \Rightarrow x^*y \leq z\)
- and their opposites

\[
1 + x^*x \leq x^* \quad y + zx \leq z \Rightarrow yx^* \leq z
\]
Models of Kleene Algebra

**Boolean semiring:** structure $A_2 = (\{0, 1\}, +, \cdot, *, 0, 1)$ with operations

\[
\begin{array}{c|cc}
+ & 0 & 1 \\
0 & 0 & 1 \\
1 & 1 & 1 \\
\end{array}
\quad
\begin{array}{c|cc}
\cdot & 0 & 1 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
\end{array}
\quad
0^* = 1^* = 1.

**question:** can you give the test algebra?
Models of Kleene Algebra

**binary relation:** set of ordered pairs on set $M$

$$R = \{(a, b) : a, b \in M\}$$

**operations:**

- **empty relation:** $\emptyset$ (the empty set)
- **unit relation:** $\Delta = \{(a, a) : a \in M\}$
- **union:** $R \cup S = \{(a, b) : (a, b) \in R \text{ or } (a, b) \in S\}$
- **product:** $R \circ S = \{(a, b) : (a, c) \in R \text{ and } (c, b) \in S \text{ for some } c \in M\}$
- **star:** $R^* = \bigcup_{i \geq 0} R^i$ where $R^0 = \Delta$ and $R^{i+1} = R \circ R^i$ for all $i \in \mathbb{N}$

**remark:** $R^*$ corresponds to the reflexive transitive closure of $R$
Models of Kleene Algebra

**fact:** $(2^M \times M, \cup, \circ, *, \emptyset, \Delta)$ is a Kleene algebra, the full relation Kleene algebra over $M$

**proof:** check that relations satisfy Kleene algebra axioms. . .

**fact:** every subalgebra of a full relation Kleene algebra is again a Kleene algebra; a relation Kleene algebra

**proof:**
- logically, Kleene algebras are universal Horn theories
- a general theorem of universal algebra says that universal Horn theories are closed under subalgebras
Models of Kleene Algebra

**question:** can you define the test algebra of a relation Kleene algebra?

**remarks:**
- binary relations yield a standard semantics for (imperative) programs
- they model their input/output behaviour with respect to stores
- they capture nondeterminism and are very useful for specifications (even for functional programs)
- we will consider this semantics more abstractly below
Models of Kleene Algebra

**question:** the operations of Kleene algebras are precisely the regular operations; is there any connection with language theory?

**words** are finite sequences over a (finite) alphabet $\Sigma$

**languages** are sets of words

**operations:**

- empty language: $\emptyset$ (empty set)
- unit language: $\{\epsilon\}$ with empty word $\epsilon$
- union: $L_1 \cup L_2$ as in set theory
- product: $L_1 \circ L_2 = \{w_1w_2 : w_1 \in L_1 \text{ and } w_2 \in L_2\}$
- star: $L^* = \{w_1w_2\ldots w_n : w_i \in L \text{ and } n \geq 0\}$
Models of Kleene Algebra

**fact:** \((2^{\Sigma^*}, \cup, \circ, *, \emptyset, \{\epsilon\})\) is a Kleene algebra, the **full language Kleene algebra** over \(M\)

**fact:** every subalgebra of a full language Kleene algebra is again a Kleene algebra; a **language Kleene algebra**

**regular subsets/events:** obtained from finite subsets of \(\Sigma^*\) by finite number of regular operations

**consequence:** strong link between Kleene algebras and regular languages
Models of Kleene Algebra

**slogan:** Kleene algebras are algebras of regular events

- Kozen has shown that an equation holds in Kleene algebras iff it holds about regular events/expressions
- mathematically, the algebra of regular events over $\Sigma$ is the free Kleene algebra generated by $\Sigma$

**consequence:** equations in Kleene algebras can be decided by automata

**remarks:**

- this correspondence motivates the name “Kleene algebra”
- universal Horn theory of Kleene algebras is undecidable (Post)
- there is no finite **equational** axiomatisation for the equational theory of regular events
Models of Kleene Algebra

paths: finite sequences of states from $P$; empty path $\epsilon$

path product: glue paths together

$$\sigma.p \cdot p.\sigma' = \sigma.p.\sigma' \quad \sigma.p \cdot q.\sigma' \text{ undefined}$$

operations on sets of paths:

- $P_1 \circ P_2 = \{\pi_1 \cdot \pi_2 : \pi_1 \in P_1, \pi_2 \in P_2 \text{ and } \pi_1 \cdot \pi_2 \text{ defined}\}$
- other operations as usual (what is multiplicative unit?)

consequence: sets of paths form path Kleene algebras
Models of Kleene Algebra

**trace:** alternating sequence \[ p_0a_0p_1a_1p_2 \cdots p_{n-2}a_{n-1}p_{n-1}, \quad p_i \in P, \ a_i \in A \]

**trace product:** \[ \sigma.p \cdot p.\sigma' = \sigma.p.\sigma' \quad \sigma.p \cdot q.\sigma' \text{ undefined} \]

**operations** on sets of traces:
- \[ T_1 \circ T_2 = \{ \tau_1 \cdot \tau_2 : \tau_1 \in T_1, \tau_2 \in T_2 \text{ and } \tau_1 \cdot \tau_2 \text{ defined} \} \]
- other operations as usual (what is multiplicative unit?)

**consequence:** sets of traces form trace Kleene algebras
Relationship Between Models

special cases: essentially by forgetting structure in trace MKA

- path/language Kleene algebras forget actions/propositions
- relation Kleene algebras forget sequences between endpoints

property: (equational) properties are inherited by (relations), paths, languages

remark:

- traces, paths, languages, relations are computationally interesting models
- Kleene algebras are applicable in interoperable contexts
Further Models

**matrix model:** consider \( n \times n \) matrices over Kleene algebra

- 0 and 1 are zero and unit matrix
- + and \( \cdot \) are standard matrix addition and multiplication
- star defined by partitioning a non-singleton matrix into submatrices \( a, b, c, d \), with \( a \) and \( d \) square, and setting

\[
\left( \begin{array}{cc} a & b \\ c & d \end{array} \right)^* = \left( \begin{array}{cc} f^* & f^*bd^* \\ d^*cf^* & d^* + d^*cf^*bd^* \end{array} \right)
\]

where \( f = a + bd^*c \)

**fact:** matrices over Kleene algebras form Kleene algebras
Digression: Automata, Algebraically

finite automaton: [Conway] $(u, A, v)$ with

- $u$ 0-1 vector of start states
- $v$ 0-1 vector of accepting states
- $A$ transition matrix over Kleene algebra

language accepted by automaton is element $u^T A^* v$ of Kleene algebra

simple automaton: transition matrix of form

$$A = J + \sum_{a \in \Sigma} a \cdot A_a$$

for 0-1 matrices $J$ ($\epsilon$-matrix) and $A_a$

fact: automata theory can be developed from this angle
Digression: Automata, Algebraically

**example:** consider automaton with states \( \{p, q\} \), alphabet \( \{a, b\} \), start state \( p \), accept state \( q \), and transitions

\[
p \rightarrow_a p \quad q \rightarrow_a q \quad p \rightarrow_b q \quad q \rightarrow_b q
\]

**algebraic automaton:**

\[
\left( \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} a & b \\ 0 & a+b \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right)
\]

**language accepted:**

\[
\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \left( \begin{pmatrix} a & b \\ 0 & a+b \end{pmatrix} \right)^* \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \left( \begin{pmatrix} a^* & a^*b(a+b)^* \\ 0 & (a+b)^* \end{pmatrix} \right) \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = a^*b(a+b)^*
\]
Further Models

tropical semiring: \((N_\infty, \min, +, \infty, 0, *)\) is Kleene algebra if \(n^* = 0\) for all \(n \in N_\infty\)

applications:

- combinatorial optimisation
- path problems (encoded via matrices)
- rich mathematical theory

remark: this area alone would deserve a lecture series. . .

remark: max-plus semiring on \(N_{-\infty}\) cannot be extended to a Kleene algebra
Kleene Algebras and Regular Programs

**fact:** KAs capture while-programs/guarded commands in various semantics

\[
\begin{align*}
\text{abort} &= 0 \\
\text{skip} &= 1 \\
x; y &= xy \\
\text{if } p \text{ then } x \text{ else } y &= px + \neg py \\
\text{while } p \text{ do } x &= (px)^*\neg p
\end{align*}
\]

**remarks:**

- the usual semantic mappings have been suppressed
- the assignment rule cannot be modelled in this setting
- it can be modelled in an extension formalising substitution
Calculus of Kleene Algebras

rich calculus: all regular identities hold in Kleene algebras. e.g.,

\[ 1 \leq x^* \quad x \leq x^* \quad xx^* \leq x^* \quad x^*x \leq x^* \quad 1 + xx^* = x^* \quad 1 + x^*x = x^* \]

\[ x^*x^* = x^* \quad x^{**} = x^* \quad (xy)^*x = x(yx)^* \quad (x + y)^* = x^*(yx^*)^* \]

some quasi-identities:

\[ x \leq y \Rightarrow xz \leq yz \quad x \leq y \Rightarrow zx \leq yz \]
\[ x \leq y \Rightarrow x + z \leq y + z \quad x \leq y \Rightarrow x^* \leq y^* \]
\[ x \leq 1 \Rightarrow x^* = 1 \quad x \leq y \Rightarrow x^* \leq y^* \]
\[ xz \leq zy \Rightarrow x^*z \leq zy^* \quad zx \leq yz \Rightarrow zx^* \leq y^*z \]
\[ xy \leq y \Rightarrow x^*y \leq y \quad yx \leq y \Rightarrow yx^* \leq y \]

more results:  
www.dcs.shef.ac.uk/~georg/ka
Example: Church-Rosser Theorem and Concurrency Control

**abstract reduction:** rewrite relations as binary relations

**Church-Rosser theorem:** \( y^*x^* \leq x^*y^* \Rightarrow (x + y)^* \leq x^*y^* \)

**proof:**

- \( (x + y)^* = (y^*x^*)^* \) is regular identity
- it suffices to show \( y^*x^* \leq x^*y^* \Rightarrow (y^*x^*)^* \leq x^*y^* \) (induction over number of peaks)
- by star induction it suffices to show \( 1 + y^*x^*x^*y^* \leq x^*y^* \)
- this splits into \( 1 \leq x^*y^* \) and \( y^*x^*x^*y^* \leq x^*y^* \)
- the first identity (base case) is trivial
- for the second one (induction step) we calculate

\[
y^*x^*x^*y^* = y^*x^*y^* \leq x^*y^*y^* = x^*y^*
\]
Example: Church-Rosser Theorem and Concurrency Control

discussion:
• induction on number of peaks without external induction measure
• in concurrency control \((x + y)^*\) corresponds to nondeterministic loop
• this loop can be separated if \(y^*x^*\) sequences can be rearranged
• theorem holds also in trace, path and language model

outlook:
• abstract part of Church-Rosser theorem in \(\lambda\)-calculus can be proved in a similar way
• many other rewrite theorems can be proved as well

further application: transformation of while programs
General Remarks on Kleene Algebras

**conclusion:** Kleene algebras

- focus on the essential operations for modelling programs and discrete systems
- support abstract and concise reasoning within first-order logic
- have rich class of computationally meaningful models
- are strongly linked with decision procedures
- can be integrated with ATP systems (later...)

**remark:** induction axiom $y + xz \leq z \Rightarrow x^*y \leq z$ and dual

- provide star elimination rules
- support some inductive reasoning
- we will see further examples later
General Remarks on Kleene Algebras

**variations:** (see below) by weakening some axioms

- demonic refinement algebras for reasoning about total program correctness in predicate transformer models
- probabilistic Kleene algebras for analysing probabilistic protocols via probability transformers
- game algebras that capture combined angelic and demonic behaviour of agents via gameboard semantics
- basic process algebras

**limitations:**

- terminating and diverging behaviour cannot be expressed
- “nonregular” induction is not possible
- reasoning about concrete applications is model-sensitive
Adding Modalities

motivation:

- many applications require different approach to actions/propositions
- systems dynamics is often modelled via state transitions; i.e. mappings from states to states
- various logics “use” Kleene algebras, but what is the precise connection?

idea: modal approach

- actions/propositions via Kripke frames
- system dynamics via images/preimages $\langle x|p / |x\rangle p$
- preimages via axiomatisation of domain
- images via axiomatisation of codomain
express: “terminating program $a$ starting from store $p$ creates store $q$”

in idempotent semiring: $pa \leq aq$ or equivalently $pa \neg q = 0$
proof of equivalence

\[ pa = pa(q + \neg q) = paq + pa\neg q = paq + 0 \leq aq \]

\[ pa\neg q \leq aq\neg q = a0 = 0 \]
State Transitions

alternative: “$q$ contains $a$-image of $p$”

question: how can we model images/preimages directly in idempotent semirings?
relational model: complement of image of set $p$ under relation $a$

is greatest set that does not admit an $a$-transition from $p$
Domain on Trace, Path, Language and Relation Semirings

intuition:

• relation semiring: $d(R) = \{ a : (a, b) \in R \}$
• trace semiring: $d(T) = \{ p : p = \text{first}(\tau) \text{ and } \tau \in T \}$
• path semiring: analogous
• language semiring: $d(\emptyset) = \emptyset$ and $d(L) = \{ \epsilon \}$ else
general idea:

- domain as mapping \( d : S \rightarrow S \) on semiring \( S \)
- \( d(x) \) models states at which action \( x \) is enabled
- \( d(x) \) should be
  - \( \leq 1 \)
  - least left preserver of \( x \): \( x \leq px \Leftrightarrow d(x) \leq p \)
- equational axioms would be nice
Domain Semirings

general idea:

- axiomatise domain as mapping $d : S \to S$ on semiring $S$
- $d(x)$ models states at which action $x$ is enabled
- $d(x)$ should be
  - $\leq 1$
  - least left preserver of $x$: $x \leq px \iff d(x) \leq p$
    where $px$ models restriction of action $x$ to states in $p$
- equational axioms would be nice

question: what would be the type of $p$?
Domain Semirings

**domain semiring**: semiring with mapping $d : S \rightarrow S$ that satisfies

\[
x + d(x)x = d(x)x \quad d(xy) = d(xd(y)) \quad d(x + y) = d(x) + d(y)
\]
\[
d(x) + 1 = 1 \quad d(0) = 0
\]

**some intuition:**

- axiom 1: $x \leq d(x)x$ means that domain is a left preserver
- axiom 2: $d(xy)$ is local on $y$ through its domain
- axiom 3: enabling a choice means enabling one alternative or the other
- axiom 4: domain is smaller than 1 (cf. next slide)
- axiom 5: the abortive action is never enabled
Domain Semirings

**property:** every domain semiring is automatically idempotent

**further properties:** the axioms

- are irredundant (use model generator Mace4)
- cannot be weakened to inequalities (Mace4)
- imply least left preservation
- imply many “natural properties” (cf. next slides)

**domain elements:** $d(x) = x$ says “$x$ is domain element”
Properties of Domain

**fact:** Let \( S \) be a domain semiring. Let \( x, y \in S \) and let \( p \in d(S) \). Then

- \( d(x)x = x \) (domain is a left invariant)
- \( d(p) = p \) (domain is a projection)
- \( d(xy) \leq d(x) \) (domain increases for prefixes)
- \( x \leq 1 \Rightarrow x \leq d(x) \) (domain expands subidentities)
- \( d(x) = 0 \Leftrightarrow x = 0 \) (domain is very strict)
- \( d(1) = 1 \) (domain is co-strict)
- \( x \leq y \Rightarrow d(x) \leq d(y) \) (domain is isotone)
- \( d(px) = pd(x) \) (domain elements can be exported)
- \( d(x)d(x) = d(x) \) (domain elements are multiplicatively idempotent)
- \( d(x)d(y) = d(y)d(x) \) (domain elements commute)
- \( x \leq px \Leftrightarrow d(x) \leq p \) (domain elements are least left-preservers)
- \( xy = 0 \Leftrightarrow xd(y) = 0 \) (domain is weakly local)
Domain Algebra

question: how can we relate domain elements with tests?

property: for every domain semiring $S$, the sub-structure $(d(S), +, \cdot, 0, 1)$ is a bounded distributive lattice

proof: (with ATP)
1. check closure properties, $d(1) = 1$ and $d(0) = 0$
2. this gives sub-semiring
3. $d(x) \leq 1$ is axiom and $d(x)d(x) = d(x)$
4. but semirings satisfying these two properties are distributive lattices [Birkhoff]

notation:
- $(d(S), +, \cdot, 0, 1)$ is called domain algebra of $S$
- $p, q, r \ldots$ for domain elements
Domain Algebra

question: how can we enrich the domain algebra?

answer: (examples)

1. Heyting algebra: add Galois connection (and closure condition for \( \rightarrow \))

\[
pq \leq r \iff p \leq q \rightarrow r
\]

2. Boolean algebra: add antidomain operation \( a : S \rightarrow S \) with axioms

\[
d(x) + a(x) = 1 \quad d(x)a(x) = 0
\]
Boolean Domain Algebra

**assume:** semiring that satisfies the domain/antidomain axioms

**consequence:** \( d(S) \) is the largest Boolean subalgebra of \( S \), so

\[
d(S) = \text{test}(S)
\]

**properties:** (ATP)

- \( a^2(x) = d(x) \)
- \( a(x) \) is greatest left annihilator of \( x \): \( px = 0 \iff p \leq a(x) \)

**consequence:**

- \( d \) can be replaced by \( a^2 \)
- many domain/antidomain axioms become redundant
- axiomatisation can be simplified
- this yields. . .
Boolean Domain Semirings

**Boolean domain semiring**: semiring $S$ with mapping $a : S \rightarrow S$ that satisfies

$$a(x)x = 0 \quad a(xy) \leq a(xa^2(y)) \quad a^2(x) + a(x) = 1$$

**remarks:**

- ATP/model search is very helpful in this development
- simple axioms induce rich modal calculus. . .
Modal Semirings

idea: define forward/backward diamonds as preimages/images

\[ |x⟩p = d(xp) \quad ⟨x|p = d^o(px) \]

where codomain operation \( d^o \) is dual of domain

consequence:

- we have \( |x⟩0 = 0 \) and \( |x⟩(p + q) = |x⟩p + |x⟩q \)
- this yields
  - distributive lattices with operators
  - Heyting algebras with operators
  - Boolean algebras with operators

convention: we will call KAs with Boolean domain modal KAs (MKAs)
Modalities, Symmetries, Dualities for Boolean Domain

demodalisation: $|x⟩p \leq q \Leftrightarrow \neg qxp \leq 0$  $⟨x|p \leq q \Leftrightarrow px\neg q \leq 0$

dualities:
  
  • de Morgan: $|x⟩p = \neg|x⟩\neg p$  $[x]|p = \neg⟨x|\neg p$
  • opposition: $⟨x|$, $[x| \Leftrightarrow |x⟩$, $|x]$

symmetries:
  
  • conjugation: $(|x⟩p)q = 0 \Leftrightarrow p(⟨x|q = 0$
  • Galois connection: $|x⟩p \leq q \Leftrightarrow p \leq [x|q$

benefits: rich calculus (automatically verified)
  
  • symmetries as theorem generators
  • dualities as theorem transformers
Kleene Modules

**Kleene module**: [Leiß06] structure \((K, L, : )\) with

\[
(x + y)p = xp + yp \quad x(p + q) = xp + xq \quad (xy)p = x(yp)
\]

\[
1p = p \quad x0 = 0 \quad xp + q \leq p \Rightarrow x^*q \leq p
\]

**remark**: scalar product \( : \) omitted

**fact**: modal Kleene algebras are Kleene modules with \( : = \langle x \rangle p \)

**consequence**: close relationship with computational logics
MKAs and Propositional Dynamic Logic

**fact:** MKAs are dynamic/test algebras

**proof:**
- dynamic algebras are almost Kleene modules
- main task is to show equivalence of
  - module induction law \(|x⟩p + q \leq p \Rightarrow |x^*⟩q \leq p\)
  - Segerberg axiom \(|x^*⟩p - p \leq |x^*⟩(|x⟩p - p)\)

**extensionality:** \((∀p. |x⟩p = |y⟩p) \Rightarrow x = y\)

**intuition:** extensionality forces Kripke-style models

**corollary:** extensional MKAs are essentially propositional dynamic logics
MKAs and Propositional Dynamic Logic

**benefits:** MKA offers

- simpler/more modular axioms
- richer model class (beyond Kripke frames)
- more flexible setting

**perspective:**

- simple automated reasoning about programs and systems with off-the-shelf ATP systems
- easily extendable to the automation of first-order variants, e.g.,

\[
\exists x \forall p \exists q. (|x⟩f(p) \leq |x⟩g(q) \rightarrow |x⟩h(p, q) = 0)
\]

- some temporal logics and Hoare logics subsumed
MKAs and Linear Temporal Logic

encoding:

- temporal operators (use one single action $x$)
  
  $$X p = |x\rangle p \quad F p = |x^*\rangle p \quad G p = |x^*\rangle p \quad pU q = |(px)^*\rangle q$$

- initial state $\text{init}_x = [x|0$ “there’s nothing before the beginning”

- validity of temporal implications $\sigma \models p \rightarrow q \iff \text{init}_x \cdot p = q$

- tests now model sets of traces and $x$ models the abstract tail relation
MKAs and Linear Temporal Logic

**LTL axioms:** von Karger’s variant of [Manna/Pnueli]

\[ (px)^* q = q + p|x|(px)^* q \]
\[ |(px)^* 0 \leq 0 \]
\[ |x^*](p \rightarrow q) \leq |x^*]p \rightarrow |x^*]q \]
\[ |x^*]p \leq p|x]|x^*]p \]
\[ p \leq |x]|x]p \]
\[ init_x \leq |x^*](p \rightarrow [x|q) \rightarrow |x^*](p \rightarrow [x^*|q) \]
\[ |x](p \rightarrow q) = |x]p \rightarrow |x]q \]
\[ x|p \leq [x]|p \]
\[ [x](p \rightarrow q) = [x]p \rightarrow [x]q \]
\[ x|p = |x]p \]
MKAs and Linear Temporal Logic

fact:
1. blue axioms are theorems of MKA
2. violet axioms express linearity of models (in MKA)

benefits:
- reasoning about infinite-state systems possible
- first-order temporal reasoning
- trace model available

remark:
- CTL also subsumed
- CTL* needs additional fixed points (and quantale-based setting)
MKAs and Hoare Logic

**fact:** MKA subsumes (propositional) Hoare logic

**explanation:** this is Hoare logic without the assignment rule

**convention:** Kleenean notation for syntax and semantics

**validity of Hoare triple:** \[ \models \{p\} \ x \ \{q\} \iff \left\langle x \right| p \leq q \]

“terminating program \(x\) starting from store \(p\) creates store \(q\)”

**validity of implication:** \[ \models p \rightarrow q \iff p \leq q \]

**example:** validity of while rule \[ \models_{\text{MKA}} \left\langle x \right| pq \leq q \Rightarrow \left\langle (px)^* \neg p \right| q \leq \neg pq \]
MKAs and Hoare Logic

benefits:

• weakest liberal precondition semantics for free in MKA \( \text{wlp}(x, p) = [x]p \)
• soundness and completeness of Hoare logic are easy in MKA
• formalism of Hoare logic is dissolved in modal setting
• relative completeness not an issue...
Propositional Hoare Logics

**Hoare calculus:** inference rules

- **abort:**  \( \models \{p\} \ abort \ {q} \)
- **skip:**  \( \models \{p\} \ skip \ {p} \)
- **assignment:**  \( \models \{q[e/x]\} \ x := e \ {q} \)
- **composition:**  \( \models \{p\} \ x \ {q}, \{q\} \ y \ {r} \Rightarrow \{p\} \ x ; \ y \ {r} \)
- **conditional:**  \( \models \{p \land q\} \ x \ {r}, \{\neg p \land q\} \ y \ {r} \Rightarrow \{q\} \ if \ p \ then \ x \ else \ y \ {r} \)
- **while:**  \( \models \{p \land q\} \ x \ {q} \Rightarrow \{q\} \ while \ p \ do \ x \ \{\neg p \land q\} \)
- **weakening:**  \( \models p_1 \rightarrow p, \{p\} \ x \ {q}, q \rightarrow q_1 \Rightarrow \{p_1\} \ x \ {q_1} \)
Soundness

**Hoare calculus:** coding validity in MKA

- **abort:** $\langle 0|p \leq p \rangle$
- **skip:** $\langle 1|p \leq p \rangle$
- **assignment:** expressiveness assumption
- **composition:** $\langle x|p \leq q, \langle y|q \leq r \Rightarrow \langle xy|p \leq r \rangle$
- **conditional:** $\langle x|(pq) \leq r, \langle y|\neg(pq) \leq r \Rightarrow \langle px + \neg py|q \leq r \rangle$
- **while:** $\langle x|(pq) \leq q \Rightarrow \langle (px)^*\neg p|q \leq \neg pq \rangle$
- **weakening:** $p_1 \leq p, \langle x|p \leq q, q \leq q_1 \Rightarrow \langle x|p_1 \leq q_1 \rangle$
Soundness

**Hoare calculus:** coding validity in operator Kleene algebra

- **abort:** \( 0 \leq f \)
- **skip:** \( 1 \leq 1 \)
- **assignment:** expressiveness assumption
- **composition:** \( \langle xy \rangle \leq \langle y \rangle \langle x \rangle \)
- **conditional:** \( \langle px + \neg py \rangle \leq \langle x \rangle \langle p \rangle + \langle y \rangle \langle \neg p \rangle \)
- **while:** \( \langle x \rangle \langle p \rangle f \leq f \Rightarrow \langle(px)^*\neg p \rangle f \leq \langle \neg p \rangle f \)
- **weakening:** \( f_1 \leq f, hf \leq g, g \leq g_1 \Rightarrow hf_1 \leq g_1 \)
Soundness

**Hoare calculus:** inference rules are theorems in operator Kleene algebra

- **abort:** $0 \leq f$ trivial
- **skip:** $1 \leq 1$ trivial
- **assignment:** expressiveness assumption
- **composition:** $\langle xy \rangle \leq \langle y \rangle \langle x \rangle$ contravariance
- **conditional:** $\langle px + \neg py \rangle \leq \langle x \rangle \langle p \rangle + \langle y \rangle \langle \neg p \rangle$ decomp., contravar.
- **while:** $\langle x \rangle \langle p \rangle f \leq f \Rightarrow \langle (px)^* \neg p \rangle f \leq \langle \neg p \rangle f$ next slide. . .
- **weakening:** $f_1 \leq f, hf \leq g, g \leq g_1 \Rightarrow hf_1 \leq g_1$ isotonicity
Soundness

**proof** of while-rule \( \langle x|p|f \leq f \Rightarrow \langle(px)^*\neg p|f \leq \langle \neg p|f \)

\[ \langle x|p|f \leq f \iff \langle px|f \leq f \] (contravariance)
\[ \Rightarrow \langle(px)^*|f \leq f \] (induction)
\[ \Rightarrow \langle \neg p|(px)^*|f \leq \langle \neg p|f \] (isotonicity)
\[ \iff \langle(px^*)\neg p|f \leq \langle \neg p|f \] (contravariance)

**proposition:** propositional Hoare logic is **sound** wrt algebraic semantics
Decidability

Hoare formulas: quasi-identities in modal Kleene algebra

\[ \langle x_1 | p_1 \leq q_1, \ldots, \langle x_n | p_n \leq q_n \Rightarrow \langle a_0 | p_0 \leq q_0 \]

decision procedure: (PSPACE)

1. demodalisation: rewrite as equivalent quasi-identity in Kleene algebra

\[ p_1 x_1 \neg q_1 \leq 0, \ldots, p_n x_n \neg q_n \leq 0 \Rightarrow p_0 x_0 \neg q_0 \leq 0 \]

2. hypothesis elimination: reduce to equivalent identity \[ s' \leq t' \]

3. apply PSPACE decision procedure for equational theory
MKAs and Hoare Logic

perspective:

• full automation of Hoare logic seems possible
• assignment rule requires formalising substitution
• handling numbers or data types is so far difficult for ATP systems
• approach extends to total correctness
Divergence and Termination

∇-Kleene module: Kleene module \((K, L, :)\) with divergence \(\nabla : K \rightarrow L\) satisfying

- ∇-unfold \(x\nabla \leq xx\nabla\)
- ∇-coinduction \(p \leq xp + q \Rightarrow p \leq x^\nabla + x^*q\)

remark: scalar product symbol omitted

interpretation:

1. for modal Kleene algebra, \(x\nabla\) denotes those states from which infinite behaviour may start
2. if \(K\) models finite actions and \(L\) infinite actions, then \(x\nabla\) is the infinite iteration of finite action \(x\)
Divergence and Termination

**Fact:** if $L$ is Boolean algebra, then $\nabla$-coinduction is equivalent to

$$p \leq xp \Rightarrow p \leq x^\nabla$$

**Final Part:** $\max_x(p) = p - xp$ models final part of $p$ w.r.t. $x$

**Termination:** action $x$ terminates if $x^\nabla = 0$

**Property:** if $L$ is Boolean algebra, then $x$ terminates iff

$$\max_x(p) = 0 \Rightarrow p = 0$$

**Remark:** this captures set-theoretic notion of Noethericity
Divergence and Termination

trace model:

- let $K$ be a trace Kleene algebra
- let $L$ be a set of infinite traces under union
- define, for $\tau \in K$ and $\pi \in L$ the scalar product $\tau : \pi$ like product of finite traces
- lift that product to sets of traces
- define $\pi^\nabla = \{ \pi \in L : \pi = \tau_0 \cdot \tau_1 \cdot \ldots \text{ with } \tau_i \in K \text{ for } i \geq 0 \}$

Then $(K, L, :, \nabla)$ is a (full trace) $\nabla$-Kleene module

special cases: path and language $\nabla$-Kleene modules

consequence: $\nabla$-Kleene modules useful for integrated finite/infinite behaviour
Divergence and Termination

**fact:** divergence and termination can be equationally axiomatised

- \( p \leq x^{\nabla} + x^{*}\max_{x}(p) \) is equivalent to \( \nabla \)-coinduction
- \( p \leq x^{*}\max_{x}(p) \) is equivalent to termination

**remark:** \( L \) must be Boolean algebra

**intuition:** \( p \) either leads to divergence or to final states after a finite iteration

**perspective:**
- characterisation dual to Segerberg’s axiom
- equational approach to finite and infinite behaviours of discrete dynamical systems
- very suitable for ATP systems (see below)
Domain on Sub-Semirings

near-semiring: structure \((S, +, \cdot)\) such that

- \((S, +)\) and \((S, \cdot)\) are semigroups
- right distributivity law \((x + y)z = xz + yz\) holds

pre-semiring: left pre-isotone near-semiring \(x + y = y \Rightarrow zx + zy = zy\)

units: 0, 1 or

- deadlock \(x + \delta = x\) \(\delta x = \delta\).
- silent action \(x\tau = x\)
Domain on Sub-Semirings

**basic process algebra:** idempotent near-semiring \((S, +, \cdot, *)\) or \((S, +, \cdot, *, \delta, \tau)\)

**game algebra:** idempotent pre-semiring \((S, +, \cdot, 0, 1)\)

**probabilistic Kleene algebra:** idempotent pre-semiring \((S, +, \cdot, *, 0, 1)\)

**demonic refinement algebra:** idempotent semiring \((S, +, \cdot, *, \infty, \delta, 1)\)
## Domain on Sub-Semirings

<table>
<thead>
<tr>
<th>Property</th>
<th>NS$^\tau_\delta$</th>
<th>NS$^1_\delta$</th>
<th>PS$^1_\delta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a(x)x = \delta$</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$a(xy) \leq a(xa^2(y))$</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$a^2(x) + a(x) = 1$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$a(x + y) = a(x)a(y)$</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>$x = d(x)x$</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d(xy) = d(xd(y))$</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>$d(x + y) = d(x) + d(y)$</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>$d(\delta) = \delta$</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>$d(x)d(y) = d(y)d(x)$</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>$d(a(x)) = a(x)$</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

NS: near-semiring, PS: pre-semiring
Domain on Sub-Semirings

**conclusion:**

- domain can still be defined on sub-semirings
- this models enabledness conditions for games, processes and actions in protocols
- semiring domain axioms suffice for probabilistic Kleene algebras and demonic refinement algebras
- domain does *not* induce modal operators
Automation Examples

observation: ATP systems have rather been neglected in formal methods

idea: combine MKAs with ATPs and counter example generators

results: experiments with various ATPs (Prover9, SPASS, Waldmeister, . . . )

• \(\sim 500\) theorems automatically proved
• successful case studies in program refinement, termination, . . . analysis

benefits:

• special-purpose calculi made redundant
• generic flexible library of lemmas
• new style of verification
Automating Bachmair and Dershowitz’s Termination Theorem

**Theorem**: [BachmairDershowitz86] termination of the union of two rewrite systems can be separated into termination of the individual systems if one rewrite system quasicommutes over the other.

**Formalisation**: $\nabla$-Kleene module over semilattice

**Encoding**:  
- quasicommutation $yx \leq x(x + y)^*$  
- separation of termination $(x + y)^\nabla = 0 \iff x^\nabla + y^\nabla = 0$

**Statement**: termination of $x$ and $y$ can be separated if $x$ quasicommutes over $y$

**Remark**: posed as challenge by Ernie Cohen in 2001
results: SPASS finds an extremely short proof in $< 5$ min

\[
(x + y)^\nabla = y^\nabla + y^* x (x + y)^\nabla \\
\leq y^\nabla + x (x + y)^* (x + y)^\nabla \\
= y^\nabla + x (x + y)^\nabla \\
\leq x^\nabla + x^* y^\nabla \\
= 0
\]  

(sum unfold)  
(strong quasicommutation)  
(since $z^\omega = z^* z^\omega$)  
(coinduction)  
(assumption $x^\nabla = y^\nabla = 0$)
Automating Bachmair and Dershowitz’s Termination Theorem

**surprise:** proof reveals new refinement law

\[ yx \leq x(x + y)^* \Rightarrow (x + y)^{\nabla} = x^{\nabla} + x^*y^{\nabla} \]

for separating infinite loops

**remarks:**

- reasoning essentially coinductive
- theorem holds in large class of models
- translation safe since relations form \( \nabla \)-Kleene modules
Automating the DBW-Theorem

**lazy commutation:** \[yx \leq x(x + y)^* + y\]

**theorem:** [Doornbos/Backhouse/van der Woude]
if \(x\) lazily commutes over \(y\) then termination of \(x\) and \(y\) can be separated

**comment:** this generalisation is much more difficult

**lemma:** \(x\) lazily commutes over \(y\) iff

\[
yx^* \leq x(x + y)^* + y
\]

**proof:** 44.23s by Prover9.
Automating the DBW-Theorem

**proof:** (non-trivial direction of DBW-theorem)

1. abbreviate $\nabla = (x + y)^\nabla$
2. assume that $x$ and $y$ terminate
3. for $\nabla = 0$ it suffices to show $\max_y(\max_x(\nabla)) = 0$
4. this is equivalent to $\max_x(\nabla) \leq y\max_x(\nabla)$
5. we calculate

\[
\nabla = x\nabla + y\nabla \leq x\nabla + yx^*\max_x(\nabla) \leq x\nabla + x(x + y)^*\max_x(\nabla) + y\max_x(\nabla) \\
\leq x\nabla + x(x + y)^*\nabla + y\max_x(\nabla) = x\nabla + y\max_x(\nabla)
\]

6. the claim now follows from the Galois connection for complementation and the definition of $\max_x$

**remark:** the second step uses the equational characterisation of termination
Automating the DBW-Theorem

remarks:

• proof is much more compact than previous approaches
• for the first time in first-order setting
• theorem holds again in large model class
• main calculation could again be automated
• full automation remains a challenge
Automating a Modal Correspondence Result

**modal logic:** Löb’s formula \( \Box (\Box p \rightarrow p) \rightarrow \Box p \)

**translation** to MKA/Kleene modules: \( x p \leq x (p - x p) = x \max_x (p) \)

**intuition:** all states with transitions into \( p \) are states from which no further transitions are possible

**remark:** this would correspond to Noethericity if \( x \) is transitive (\( xx \leq x \))

**reminder:** two equivalent characterisations of Noethericity

- \( p \leq x^* \max_x (p) \) (\( x \) pre-Löbian)
- \( \max_x (p) = 0 \Rightarrow p = 0 \) (\( x \) Noetherian)
Automating a Modal Correspondence Result

**property:** for every $x$ in some $\nabla$-Kleene module

(i) $x$ L"obian $\Rightarrow$ $x$ Noetherian
(ii) $x$ Noetherian $\iff$ $x$ pre-L"obian (see above)
(iii) $x$ pre-L"obian and $x = xx$ $\Rightarrow$ $x$ L"obian

**proofs:** with Prover9 in $\nabla$-Kleene algebra

(i) $\leq 4s$
(ii) $\leq 4s$ and $\leq 20s$ (hypothesis learning)
(iii) $\leq 1s$ (hypothesis learning)

**remark:** this is a modal correspondence result

- Noethericity corresponds to frame property
- proof is calculational and automated
- model theory is normally used
Automating Hoare Logic

**algorithm:** integer division \( n/m \)

```plaintext
fun DIV = k:=0; l:=n;
    while m<=l do k:=k+1; l:=l-m;
```

**precondition:** \( 0 \leq n \)

**postconditions:** \( n = km + l \quad 0 \leq l \quad l < m \)

**proof goal:** \( \langle x_1 x_2 (r y_1 y_2)^* \neg r | p \leq q_1 q_2 \neg r \)
Automating Hoare Logic

**proof:** two phases coupled by assignment rule $p[e/x] \leq \{x := e\}p$

1. **MKA:** goal follows from $p \leq |x_1||x_2|(q_1q_2)$ $q_1q_2r \leq |y_1||y_2|(q_1q_2)$ (automated with Prover9)

2. **arithmetics:** subgoals must still be manually verified, e.g.,

$$|x_1||x_2|(q_1q_2) = |\{k := 0\}|\{l := n\} (q_1q_2) \geq (\{n = km + l\}\{0 \leq l\})[k/0][l/n]$$
$$= \{n = 0m + n\}\{0 \leq n\} = \{0 \leq n\}$$
$$= p$$

**remark:**

- reasoning essentially inductive
- domain specific solvers should be integrated into ATPs
- try SPASS+T?
Newman’s Lemma: A Proof Challenge

Newman’s lemma: A term rewriting system is confluent if it is locally confluent and terminating.

generalisation and translation:

- $x$ commutes over $y$ \[ y^*x^* \leq x^*y^* \]
- $x$ locally commutes over $y$ \[ yx \leq x^*y^* \]

Theorem: In $\nabla$-Kleene algebra, if $x + y$ terminates and $x$ locally commutes over $y$, then $x$ commutes over $y$
Newman’s Lemma: A Proof Challenge

proof: (so far)

• one page of semi-calculational arguments
• main calculation

\[
\langle y^* | \langle y | p \rangle x | x^* \rangle \leq \langle y^* | \langle p_y | y | x \rangle \langle p_x | x^* \rangle \\
\leq \langle y^* | \langle p_y | x^* \rangle \langle y^* | \langle p_x | x \rangle \rangle \\
\leq \langle y^* | \langle p_y | x^* \rangle x^* \rangle \langle y^* | \rangle \\
\leq \langle y^* | \langle p_y | x^* \rangle \rangle \langle y^* | \rangle \\
\leq |x^* \rangle \langle y^* | \langle y^* | \\
\leq |x^* \rangle \langle y^* |
\]

• \( p_x = \langle x | p \) and \( p_y = \langle y | p \)
• proof lifted to level of modal operators
Newman’s Lemma: A Proof Challenge

**question:** can you automate this?

**remarks:**

- Newman’s lemma seems to require a mix of inductive and coinductive reasoning
- the main calculation mimics precisely the traditional diagrammatic proof
- more generally, Kleene algebras give an algebraic semantics to (some) rewrite diagrams
A Non-Modal Example: Back’s Atomicity Refinement Law

demonic refinement algebra: [von Wright04] Kleene algebra
  • with axiom $x0 = 0$ dropped
  • extended by strong iteration $\infty$ encompassing finite and infinite iteration

remark: abstracted from refinement calculus [BackvonWright]

atomicity refinement law for action systems
  • complex theorem first published by Back in 1989
  • long proof in set theory analysing infinite sequences
  • proof by hand in demonic refinement algebra still covers 2 pages
  • automated analysis reveals some glitches and yields generalisation

first task: build up library of verified basic refinement laws for proof
A Non-Modal Example: Back’s Atomicity Refinement Law

**Theorem:** If

(i) \( s \leq sq \)  
(ii) \( a \leq qa \)  
(iii) \( qb = 0 \)  
(iv) \( rb \leq br \)

(v) \( (a + r + b)l \leq l(a + r + b) \)  
(vi) \( rq \leq qr \)  
(vii) \( ql \leq lq \)

(viii) \( r^* = r^\infty \)  
(ix) \( q \leq 1 \)

then

\[ s(a + r + b + l)^\infty q \leq s(ab^\infty q + r + l)^\infty \]

**Two-step Proof** with “hypothesis learning”

1. Assumptions imply \( s(a + r + b + l)^\infty q \leq sl^\infty qr^\infty q(ab^\infty qr^\infty)^\infty \)
   wait 60s for 75-step proof with Prover9
2. \( q \leq 1 \) implies \( sl^\infty qr^\infty q(ab^\infty qr^\infty)^\infty \leq s(ab^\infty q + r + l)^\infty \)
   wait < 1s for 30-step proof

**Remark:** Full proof succeeds for \( l = 0 \) (1013s for 46-step proof)
A Non-Modal Example: Back’s Atomicity Refinement Law

equational proof can be reconstructed

\[ s(a + b + r + l)q = sl^\infty (a + b + r)^\infty q \]
\[ = sl^\infty (b + r)^\infty (a(b + r)^\infty)^\infty q \]
\[ = sl^\infty b^\infty r^\infty (ab^\infty r^\infty)^\infty q \]
\[ \leq sl^\infty b^\infty r^\infty (qab^\infty r^\infty)^\infty q \]
\[ = sl^\infty b^\infty r^\infty q(ab^\infty r^\infty q)^\infty \]
\[ \leq sql^\infty b^\infty r^\infty q(ab^\infty r^\infty q)^\infty \]
\[ \leq sl^\infty qb^\infty r^\infty q(ab^\infty r^\infty q)^\infty \]
\[ \leq sl^\infty qr^\infty q(ab^\infty r^\infty q)^\infty \]
\[ = sl^\infty qr^\infty q(ab^\infty r^* q)^\infty \]
\[ \leq sl^\infty qr^\infty q(ab^\infty qr^* )^\infty \]
\[ = sl^\infty qr^\infty q(ab^\infty qr^\infty )^\infty . \]
ATP Background

**Ordered resolution:** for $\phi$ maximal wrt syntactic ordering $\prec$ on terms/literals

\[
\frac{\Gamma \rightarrow \Delta, \phi \quad \Gamma', \phi \rightarrow \Delta'}{\Gamma, \Gamma' \rightarrow \Delta, \Delta'} \quad \frac{\Gamma \rightarrow \Delta, \phi, \phi}{\Gamma \rightarrow \Delta, \phi}
\]

**Redundancy:** clause is $\prec$-redundant wrt clause set $S$ if it is entailed by $\prec$-smaller instances of clauses from $S$

**Orb:** clause set closed under ordered resolution and redundancy elimination

**Refutational completeness:** orb of inconsistent clause set contains empty clause

**remark:** unification used at first-order level
ATP Background

strategy:

• transform first-order formulas into clause set
• close working set under deduction rules
• apply deduction rules lazily
• apply redundancy elimination rules eagerly
• procedure must be fair with respect to clauses

ATP systems used:

• Prover9 and Vampire: fastest provers for algebraic theories
• Waldmeister: fastest tool for unit equations
• SPASS: ATP in sorted/typed setting
Conclusion

these lectures: modal Kleene algebras offer

- simple equational calculus including some (co)induction
- rich model class (traces, paths, languages, relations, functions, . . .)
- easy automation
- interesting applications in program analysis/verification
- relevant for modelling discrete dynamical systems

related work:

- automation of relation algebras similarly successful
- code at \url{www.dcs.shef.ac.uk/~georg/ka}
- results will be integrated into TPTP library
Conclusion

**general conclusion:** ATP systems + computational algebras motivates verification challenge

- off-the-shelf ATP with domain-specific algebras
- promising alternative to conventional approaches (model checking, HOL)
- light-weight formal methods with heavy-weight automation
Seek Simplicity and Distrust It.

[Whitehead]
Some References on Modal Kleene Algebras

5. J. Desharnais and G. Struth. Modal Semirings Revisited. (Accepted for MPC 2008).


