Towards an Algebraic Theory of Pushdown Automata and Turing Machines

Peter Höfner (joint work with Yi Yao and Liam O'Connor)

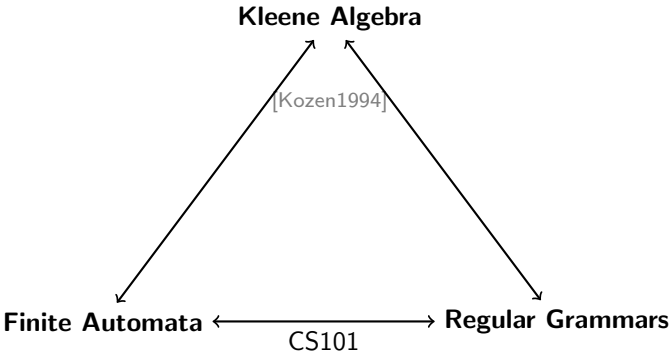
5 June 2026

Australian
National
University

Program Algebras over the Years

- Laws of Programming, 1987 [Hoare, Hayes, Morgan, ...]
- Guarded Strings and Applications, 2003 [Kozen]
- Algebra of Hybrid Systems, 2009 [Höfner]
- Separation algebra, 2009 [Dang]
- Concurrent Kleene Algebra, 2011 [Hoare, Möller, Struth]
- Synchronous Program Algebra, 2019 [Hayes, Meinicke]
- ...

Kozen's Algebraic Bridge



Kozen's Key Findings

- **Finitary axiomatisation**
  a finite system of equations and equational implications

- **Matrix closure**
  $n \times n$ matrices over any Kleene algebra form a Kleene algebra, enabling an algebraic treatment of finite automata.

Completeness

two regular expressions α, β denote the same regular set if and only if $\alpha = \beta$ is derivable from the axioms.

Kleene Algebra

A **Kleene algebra** is $(K, +, \cdot, {}^*, 0, 1)$:

Idempotent semiring

$a + (b + c) = (a + b) + c$

$a + b = b + a$

$a + 0 = a$

$a + a = a$

$a(bc) = (ab)c$

$1a = a = a1$

$a(b + c) = ab + ac$

$(a + b)c = ac + bc$

$0a = 0 = a0$

Star axioms

$1 + aa^* = a^*$

$1 + a^*a = a^*$

$b + ax \leq x \;\Rightarrow\; a^*b \leq x$

$b + xa \leq x \;\Rightarrow\; ba^* \leq x$

where $a \leq b \;\Leftrightarrow\; a + b = b$

Advantages of the Algebraic Approach

- **Uniform across models**
  same axioms govern regular languages, binary relations, program traces, shortest-path semirings, hybrid systems, ...
  one proof works in all interpretations simultaneously

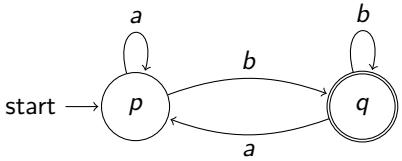
- **First-order reasoning**
  standard first-order logic applies: no higher-order machinery,
  no semantic side-conditions as in other approaches

Automatic / tool-supported reasoning

equality of two Kleene algebra terms is **decidable**
(PSPACE-complete, reducible to regular-expression equivalence)
allows use of off-the-shelf theorem provers (Isabelle/HOL, Rocq, Prover9)

Matrix Construction: Example

DFA accepting $(a + b)^* b$



Encoding as automaton (u, A, v):

$$A = \begin{matrix} \\ p \\ q \end{matrix} \begin{matrix} \begin{matrix} p & q \end{matrix} \\ \begin{pmatrix} a & b \\ a & b \end{pmatrix} \end{matrix}, \quad u = \begin{pmatrix} \varepsilon \\ 0 \end{pmatrix}, \quad v = \begin{pmatrix} 0 \\ \varepsilon \end{pmatrix}$$

Convention: A_{ij} is the sum of all letters σ with $\delta(i, \sigma) = j$
Accepted language: $u^T \cdot A^* \cdot v$

Computing A^*

The star of a matrix can be defined recursively by block decomposition.

For $A = \begin{pmatrix} a & b \\ a & b \end{pmatrix}$,

expression in Kleene algebra

$$A^* = \begin{pmatrix} (b^*a)^* & (b^*a)^* bb^* \\ b^*a\,(b^*a)^* & (a^*b)^* \end{pmatrix}$$

$$u^T A^* v \;=\; (b^*a)^* bb^* \;=\; (a+b)^* b$$

Adapting to other Models of Computation

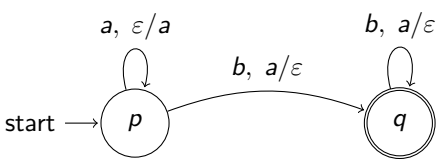
- $n \times n$ matrices over any Kleene algebra form a Kleene algebra
- regular expressions could be seen as sets of strings
  - $+$ union
  - $\cdot$ concatenation
  - * finite iteration
- could we use different underlying Kleene algebras

What about the other 'core' models of computations

- beautiful algebras for all kinds of applications
- none for Pushdown Automata
- none for Turing machines
  (coming from an other angle)

PDA Construction: Example

PDA accepting $\{a^n b^n \mid n \geq 1\}$



Notation: (*input*, pop/push)

$\{a^n b^n \mid n \geq 1\}$ is **not regular**

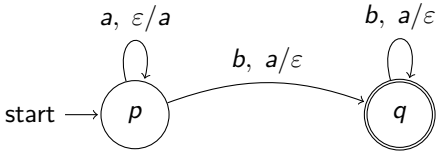
Stack enables unbounded counting:

- push a for each a read
- pop a for each b read
- accept in q when stack empty

PDA Construction: Example

PDA accepting $\{a^n b^n \mid n \geq 1\}$

Encoding as automaton (u, A, v):



$$A = \begin{array}{c} \\ p \\ q \end{array} \begin{array}{c} \begin{array}{cc} p & q \end{array} \\ \begin{pmatrix} \{(\mathfrak{s}, a, a\mathfrak{s})\} & \{(a\mathfrak{s}, b, \mathfrak{s})\} \\ \emptyset & \{(a\mathfrak{s}, b, \mathfrak{s})\} \end{pmatrix} \end{array}$$

$$u = \begin{pmatrix} \bot \\ \emptyset \end{pmatrix}, \quad v = \begin{pmatrix} \emptyset \\ \bot \end{pmatrix}$$

Notation: $\{(x\mathfrak{s}, a, y\mathfrak{s})\} =_{\text{abbv}} \{(x\mathfrak{s}, a, y\mathfrak{s}) \mid \mathfrak{s} \in \Gamma^\star\}$,
where Γ is the alphabet of the stack with $\bot$ being the symbol for empty stack.

Going Algebraic

- From triples to *guarded strings*

$$(\mathfrak{s}, a, \mathfrak{t}) \quad \mapsto \quad \mathfrak{s}.a.\mathfrak{t} \quad \text{of type } (\Gamma^\star \times \Sigma)^* \times \Gamma^\star$$

- Composition of guarded strings

$$\rho_0 \circ \rho_1 = \begin{cases} w_0.\mathfrak{s}.w_1 & \rho_0 = w_0.\mathfrak{s} \text{ and } \rho_1 = \mathfrak{s}.w_1 \\ \text{undefined} & \text{otherwise .} \end{cases}$$

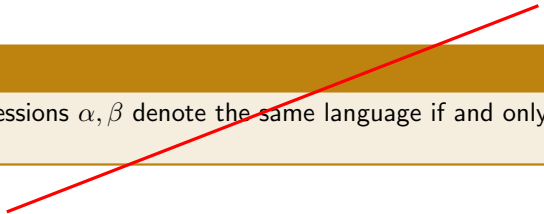
- $(\mathcal{P}((\Gamma^\star \times \Sigma)^* \times \Gamma^\star), \cup, \bullet, \emptyset, \{\})$ is a Kleene algebra,
  with $P \bullet Q = \{\rho_0 \circ \rho_1 \mid \rho_0 \in P, \rho_1 \in Q, \rho_0 \circ \rho_1 \text{ is defined}\}$

Soundness and Completeness

- **Soundness:**
  yes (quite confident)
- **Completeness**

Completeness

two context-free expressions α, β denote the same language if and only if $\alpha = \beta$ is derivable from the axioms.



Completeness – Changing the Underlying Algebra

- $\mathfrak{s}.a.\mathfrak{t} \quad \sim \quad \mathfrak{s}.a.\mathfrak{t}.\varepsilon.(\mathfrak{x}\mathfrak{t}).\varepsilon.\mathfrak{t}$
- ignore the stacks in-between

$$(\Gamma^\star \times \Sigma)^* \times \Gamma^\star \quad \mapsto \quad \Gamma^\star \times \Sigma^* \times \Gamma^\star$$

(alternatively use projections)

- soundness is still okay
- completeness is a challenge

Completeness

- but wait: equational theory of **Kleene algebra is decidable**
- and we have learnt in CS101: **equivalence of PDAs is undecidable**

that's a contradiction, is it not?

- no: While KA is decidable, KA+x is not necessarily

Current Ideas and Questions

- DFA/NFA $\sim$ Kleene algebra [Kozen1994]
- PDA $\sim$ Kleene algebra $+$ additional equations
  - can we prove completeness

Beyond Pushdown Automata

- Turing Machines = (weakened) Kleene algebra + infinite traces and infinite loops
  a.k.a. Lazy Omega Algebra
- Reactive Turing Machines = ?
- ...

Aim

a systematic algebraic theory of standard computational models
new bridges between machine models and program algebras

Answers?

Appendix

Axiomatisations of Regular Events

System	Finite?	Sound everywhere?
Salomaa F1/F2 (1966)	Yes	No
Kozen (1981)	No	Yes ($*$-continuous KAs only)
Krob / Bloom	No (schematic)	Yes
Kozen (1994)	**Yes**	**Yes**