Commutative algebras of series

Lorenzo Clemente
University of Warsaw

RAMiCS @ Będlewo, 07/04/2026

Commutative algebras of ~~series~~ number sequences

Lorenzo Clemente
University of Warsaw

RAMiCS @ Będlewo, 07/04/2026

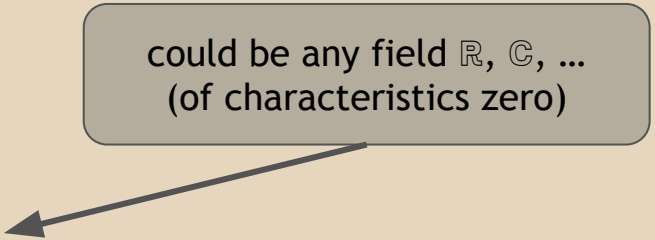
Number sequences

A *number sequence* is a mapping $f : \mathbb{N} \to \mathbb{Q}$

Examples

- 2^n, n!, 2^(2^n), …

Number sequences

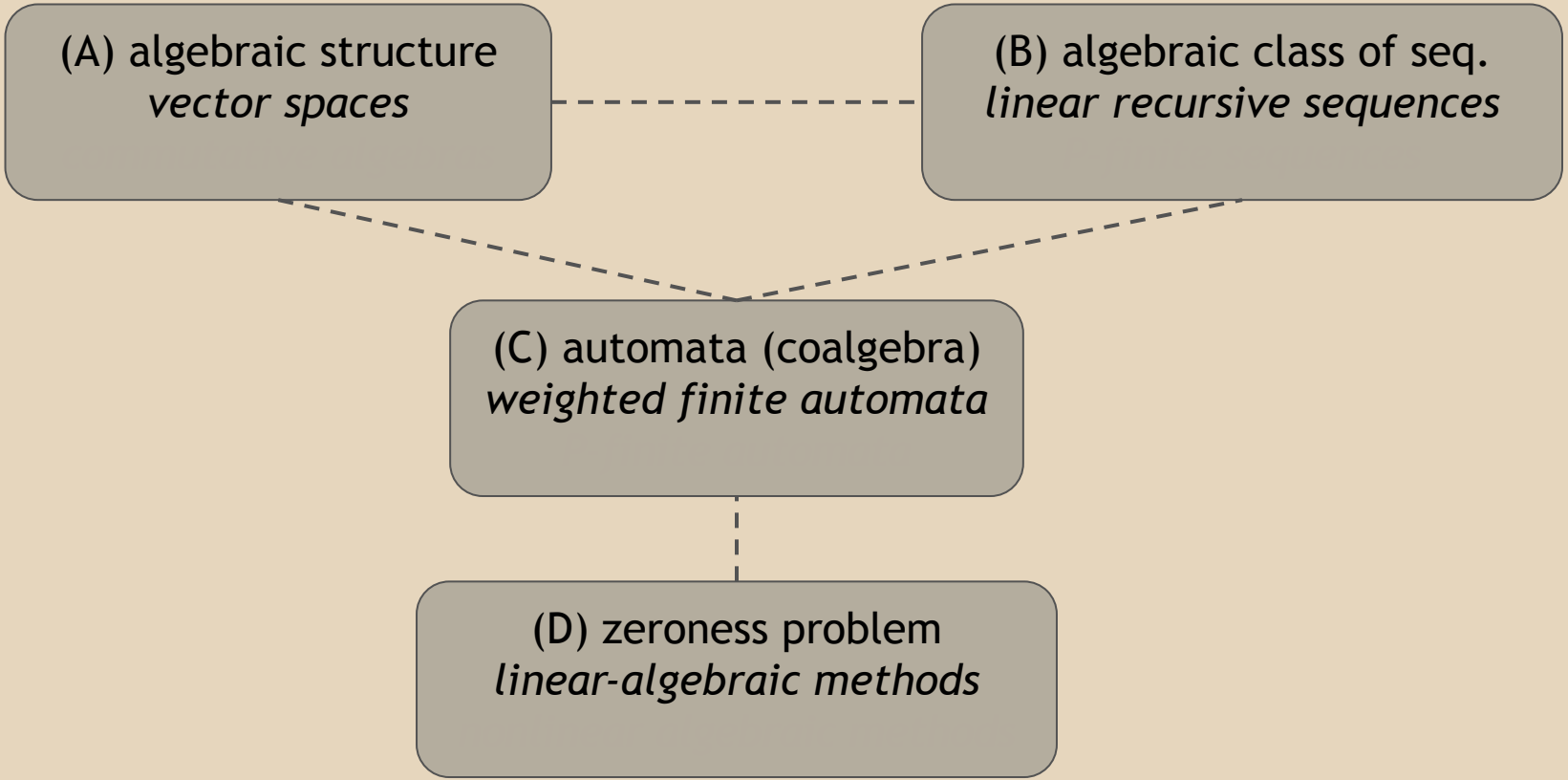


A *number sequence* is a mapping $f : \mathbb{N} \to \mathbb{Q}$

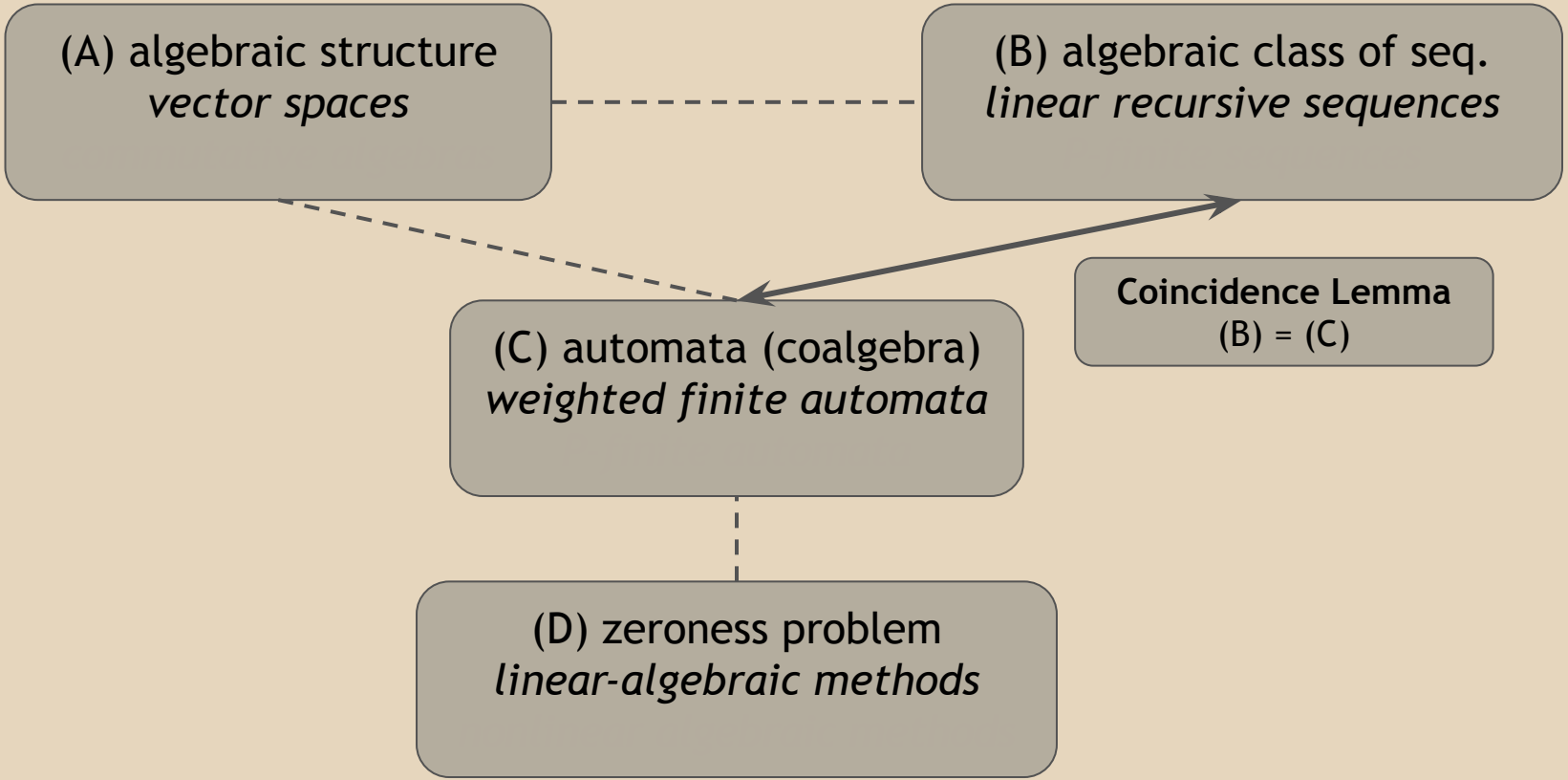
Examples

- 2^n, n!, $2^{(2^n)}$, …

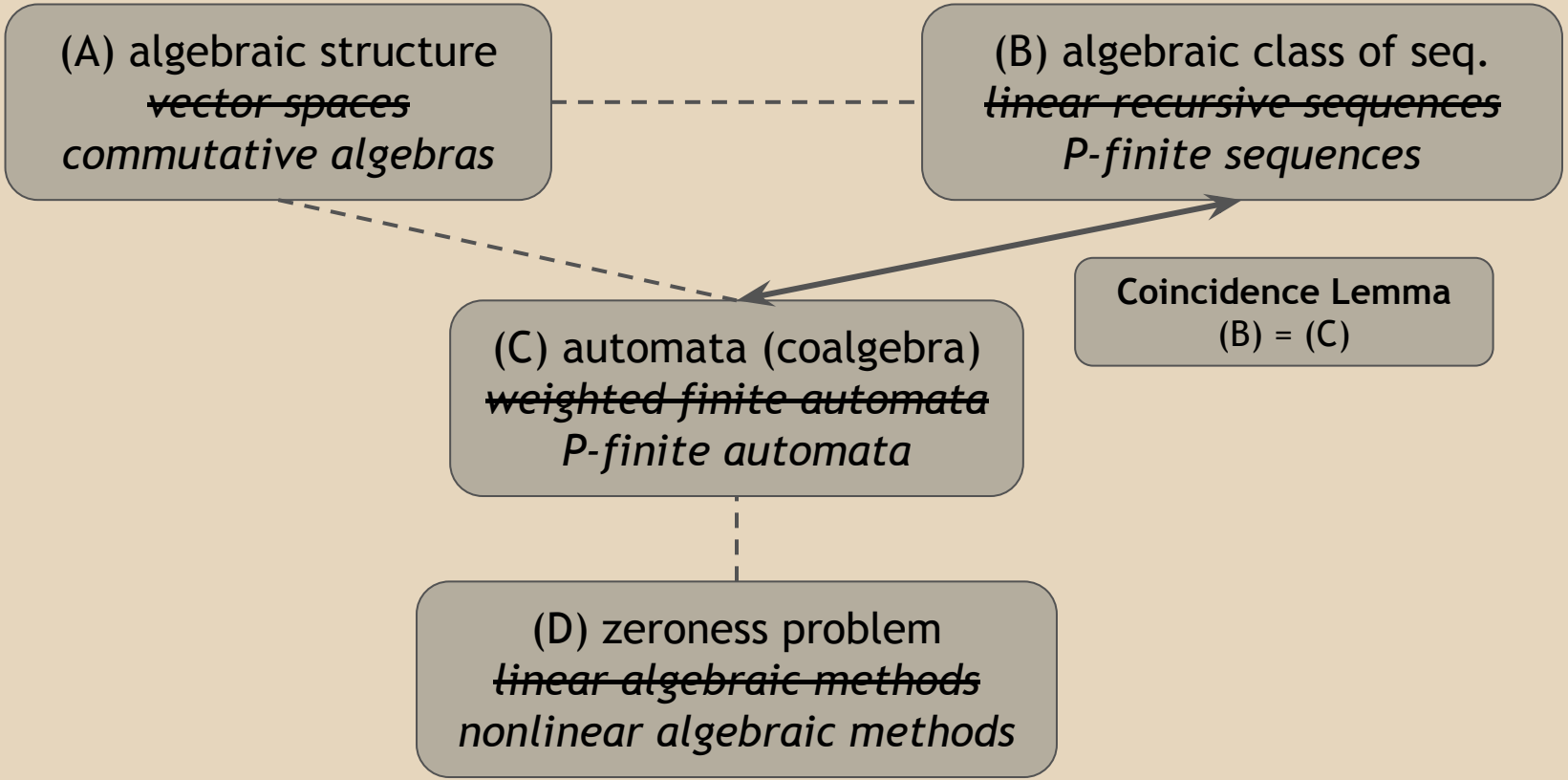
The big picture



The big picture



The big picture



Overview

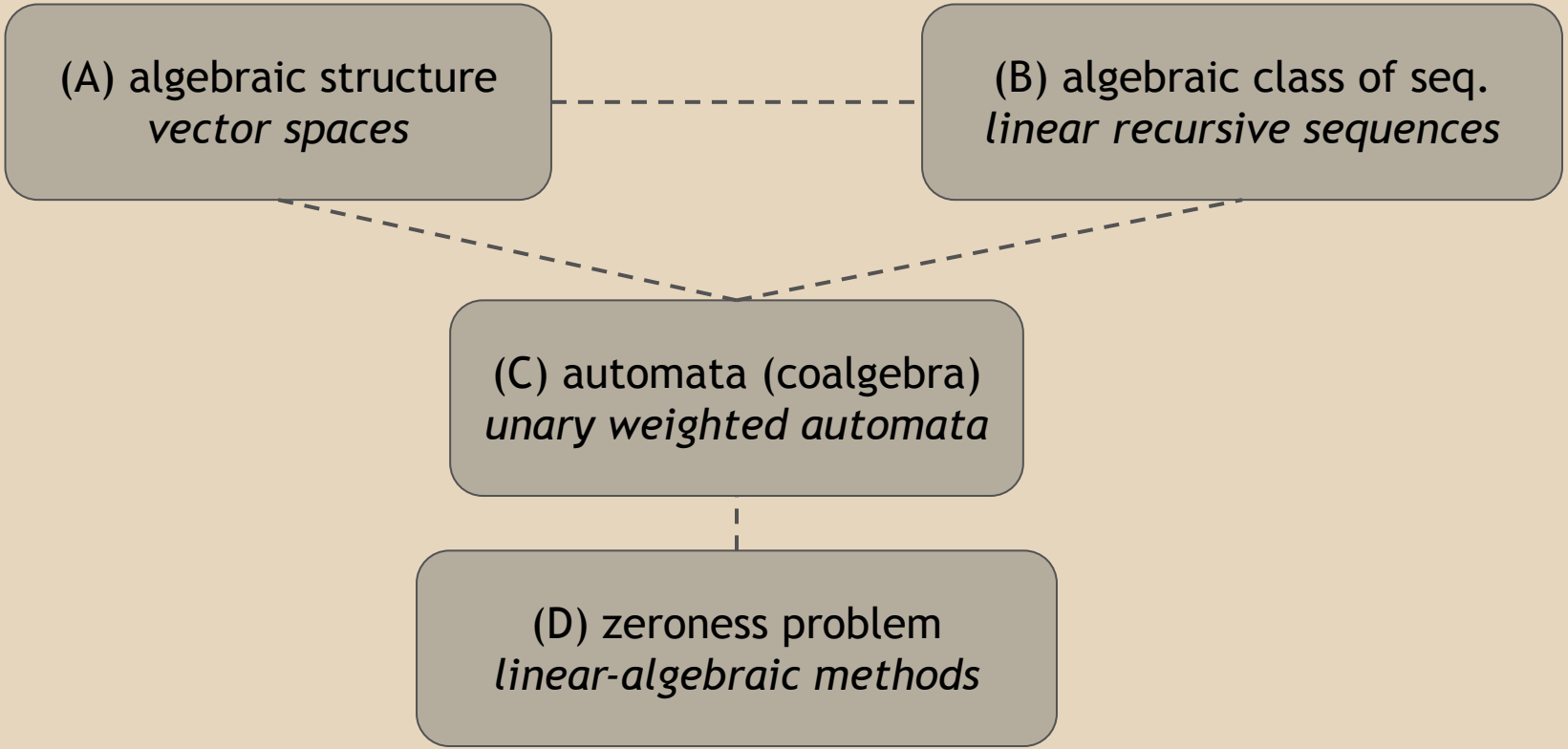
Part I
- Linear recursive sequences
- Polynomial recursive sequences and Hadamard product
- Shuffle-finite sequences and shuffle product

Part II
- Products defined by product rules
- Characterisation of commutative algebras of series

Part III
- Polynomial automata
- Decidability of the zeroness problem

Part I

The big picture



Vector space structure (A)

Number sequences $f, g : \mathbb{N} \to \mathbb{Q}$

- *Scalar multiplication*:
  $(c \cdot f)(n) := c \cdot f(n)$, $c \in \mathbb{Q}$, $n \in \mathbb{N}$

- *Addition*:
  $(f + g)(n) := f(n) + g(n)$, $n \in \mathbb{N}$

vector space
$(\mathbb{N} \to \mathbb{Q}; 0, \cdot, +)$

Transition structure (coalgebra)

Number sequence $f : \mathbb{N} \to \mathbb{Q}$

- *Left shift*

  $\sigma : (\mathbb{N} \to \mathbb{Q}) \to (\mathbb{N} \to \mathbb{Q})$
  $(\sigma\ f)(n) := f(n + 1),\ n \in \mathbb{N}$

Hello world of LRS (B)

Let f(n) = # rabbits at time n in a notorious biological model

$f(0) = f(1) = 1$
$f(n+2) = f(n+1) + f(n)$

Hello world of LRS (B)

Let f(n) = # rabbits at time n in a notorious biological model

$f(0) = f(1) = 1$
$f(n+2) = f(n+1) + f(n)$

Introduce an auxiliary sequence $g : \mathbb{N} \to \mathbb{Q}$

$f(0) = g(0) = 1$
$f(n+1) = f(n) + g(n)$
$g(n+1) = f(n)$

Hello world of LRS (B)

Let f(n) = # rabbits at time n in a notorious biological model

$f(0) = f(1) = 1$
$f(n+2) = f(n+1) + f(n)$

Introduce an auxiliary sequence $g : \mathbb{N} \to \mathbb{Q}$

$f(0) = g(0) = 1$
$f(n+1) = f(n) + g(n)$
$g(n+1) = f(n)$

$f(0) = g(0) = 1$
$\sigma f = f + g$
$\sigma g = f$

Unary weighted autom. (C)

Example of *unary weighted automaton* $A = (X, F, \Delta)$, where

- X is the set of *variables:* $X = \{x_1, x_2\}$
- $F : X \to \mathbb{Q}$ is the *output function:*
  $F\, x_1 = F\, x_2 = 1$
- $\Delta : \mathrm{Lin}(X) \to \mathrm{Lin}(X)$ is the *linear transition function*
  $\Delta\, x_1 = x_1 + x_2$, $\Delta\, x_2 = x_1$

Unary weighted autom. (C)

Example of *unary weighted automaton* $A = (X, F, \Delta)$, where

- X is the set of *variables:* $X = \{x_1, x_2\}$
- $F : X \to \mathbb{Q}$ is the *output function:*
  $F\, x_1 = F\, x_2 = 1$
- $\Delta : \mathrm{Lin}(X) \to \mathrm{Lin}(X)$ is the *linear transition function*
  $\Delta\, x_1 = x_1 + x_2$, $\Delta\, x_2 = x_1$

The *semantics* of A from x_1 is the series $A[\![x_1]\!] = n \mapsto F(\Delta^n\, x_1)$

$$
\begin{array}{lccccccc}
 & x_1 & \xrightarrow{\Delta} & x_1 + x_2 & \xrightarrow{\Delta} & 2x_1 + x_2 & \xrightarrow{\Delta} & 3x_1 + 2x_2 \xrightarrow{\Delta} \dots \\
 & \downarrow F & & \downarrow F & & \downarrow F & & \downarrow F \\
A[\![x_1]\!] = & 1 & & 2 & & 3 & & 5 \qquad \dots
\end{array}
$$

(B) = (C)

Coincidence Lemma.
The class of LRS coincides with the series recognised by unary weighted automata.

Zeroness problem (D)

INPUT: A unary weighted automaton A
OUTPUT: yes iff $A[\![x_1]\!] = 0$

Zeroness problem (D)

INPUT: A unary weighted automaton A
OUTPUT: yes iff $A[\![x_1]\!] = 0$

- Fundamental algorithmic problem
  - “Word problem” in algorithmic group theory ($g = 1$?)
  - cf. recognise the trivial knot from a complicated presentation
- Subsumes equality:

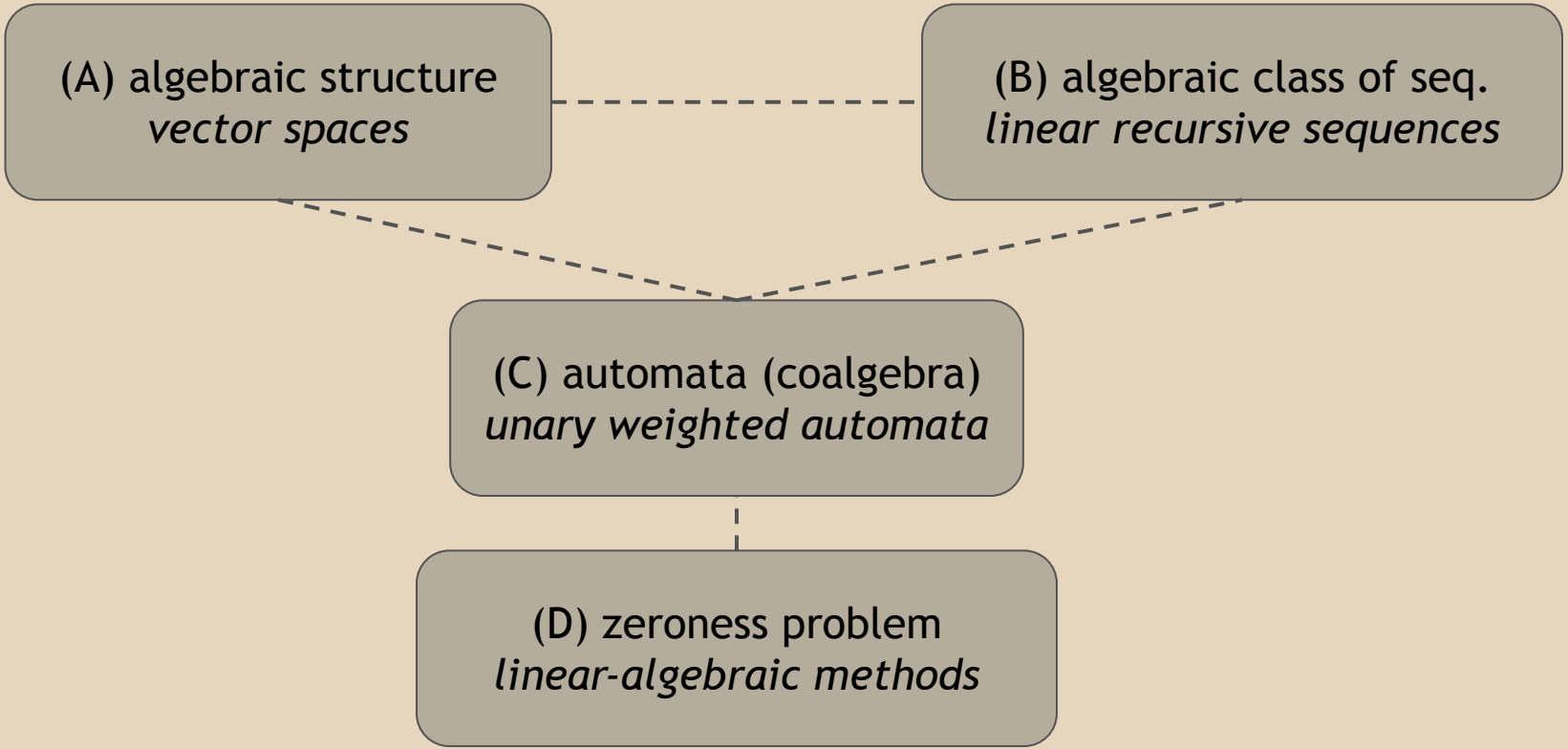
  $A[\![p]\!] = A[\![q]\!] \Leftrightarrow A[\![p]\!] - A[\![q]\!] = A[\![p - q]\!] = 0$

Zeroness problem (D)

INPUT: A unary weighted automaton A
OUTPUT: yes iff $A[\![x_1]\!] = 0$

Theorem.
The zeroness problem for unary weighted automata is decidable.
Proof. Linear algebra.

The big picture



Boolean functions

Let f(n) = # Boolean functions with n variables $\{0, 1\}^n \rightarrow \{0, 1\} = 2^{(2^n)}$

f(0) = 2
f(n+1) = f(n) · f(n)

not an LRS!

Hadamard algebra (A)

Let $\odot$ bet the element-wise product on $\mathbb{N} \to \mathbb{Q}$ (*Hadamard product*)

$(f \odot g)(n) := f(n) \cdot g(n)$

Hadamard algebra (A)

Let $\odot$ bet the element-wise product on $\mathbb{N} \to \mathbb{Q}$ (*Hadamard product*)

$(f \odot g)(n) := f(n) \cdot g(n)$

Properties (**BAC**)

- Bilinear: $(f + g) \odot h = f \odot h + g \odot h$
- Associative: $(f \odot g) \odot h = f \odot (g \odot h)$
- Commutative: $f \odot g = g \odot f$

Hadamard algebra (A)

Let $\odot$ bet the element-wise product on $\mathbb{N} \to \mathbb{Q}$ (*Hadamard product*)

$(f \odot g)(n) := f(n) \cdot g(n)$

Properties (**BAC**)

- Bilinear: $(f + g) \odot h = f \odot h + g \odot h$
- Associative: $(f \odot g) \odot h = f \odot (g \odot h)$
- Commutative: $f \odot g = g \odot f$

Hadamard algebra
$(\mathbb{N} \to \mathbb{Q}; 0, \cdot, +, \odot)$

Boolean functions (B)

Let f(n) = # Boolean functions with n variables $\{0, 1\}^n \to \{0, 1\} = 2^{(2^n)}$

f(0) = 2
$\sigma f = f \odot f$,

thus f is a *polynomial recursive sequence*

Unary polynomial aut. (C)

Example of *unary polynomial automaton* $A = (X = \{ x \}, F, \Delta)$, where

- $F : X \to \mathbb{Q}$ is the *output function*:
  $F\ x = 2$
- $\Delta : X \to \mathbb{Q}[x]$ is the *polynomial transition function*
  $\Delta\ x = x^2$

The *semantics* of A from x is the series $A[\![x]\!] = n \mapsto F(\Delta^n\ x)$

$$
\begin{array}{rccccc}
 & x & \xrightarrow{\Delta} & x^2 & \xrightarrow{\Delta} & ??? \\
 & \downarrow F & & \downarrow F & & \\
A[\![x]\!] = & 2 & & 4 & &
\end{array}
$$

Unary polynomial automata (6)

The Hadamard product satisfies the product rule

$\sigma\,(f \odot g) = (\sigma\, f) \odot (\sigma\, g)$ (homomorphism)

We extend $\Delta : X \to \mathbb{Q}[x]$ to satisfy the analogous polynomial product rule.
Let $\Delta : \mathbb{Q}[x] \to \mathbb{Q}[x]$ be unique linear extension of $\Delta : X \to \mathbb{Q}[x]$ s.t.

$\Delta\,(pq) = (\Delta p) \cdot (\Delta q)$
(homomorphic extension)

Example: $\Delta(x^n) = (\Delta\, x)^n$

$$\begin{array}{ccccc} x & \text{—}\Delta\text{—>} & x^2 & \text{—}\Delta\text{—>} & ??? \\ \downarrow F & & \downarrow F & & \\ 2 & & 4 & & \end{array}$$

$A[[x]] =$

Unary polynomial ops (6)

The Hadamard product satisfies the product rule

$$\sigma\,(f \odot g) = (\sigma\, f) \odot (\sigma\, g) \qquad \text{(homomorphism)}$$

We extend $\Delta : X \to \mathbb{Q}[x]$ to satisfy the analogous polynomial product rule.
Let $\Delta : \mathbb{Q}[x] \to \mathbb{Q}[x]$ be unique linear extension of $\Delta : X \to \mathbb{Q}[x]$ s.t.

$$\Delta\,(pq) = (\Delta p) \cdot (\Delta q)$$

(homomorphic extension)

Example: $\Delta(x^n) = (\Delta\, x)^n$

$$\begin{array}{lccccccccc} & x & \text{—}\Delta\text{—>} & x^2 & \text{—}\Delta\text{—>} & (x^2)^2 = x^4 & \text{—}\Delta\text{—>} & (x^2)^4 = x^{16} & \text{—}\Delta\text{—>} & \dots \\ & \downarrow F & & \downarrow F & & \downarrow F & & \downarrow F & & \\ A[[x]] = & 2 & & 4 & & 16 & & 256 & & \dots \end{array}$$

Zeroness problem (D)

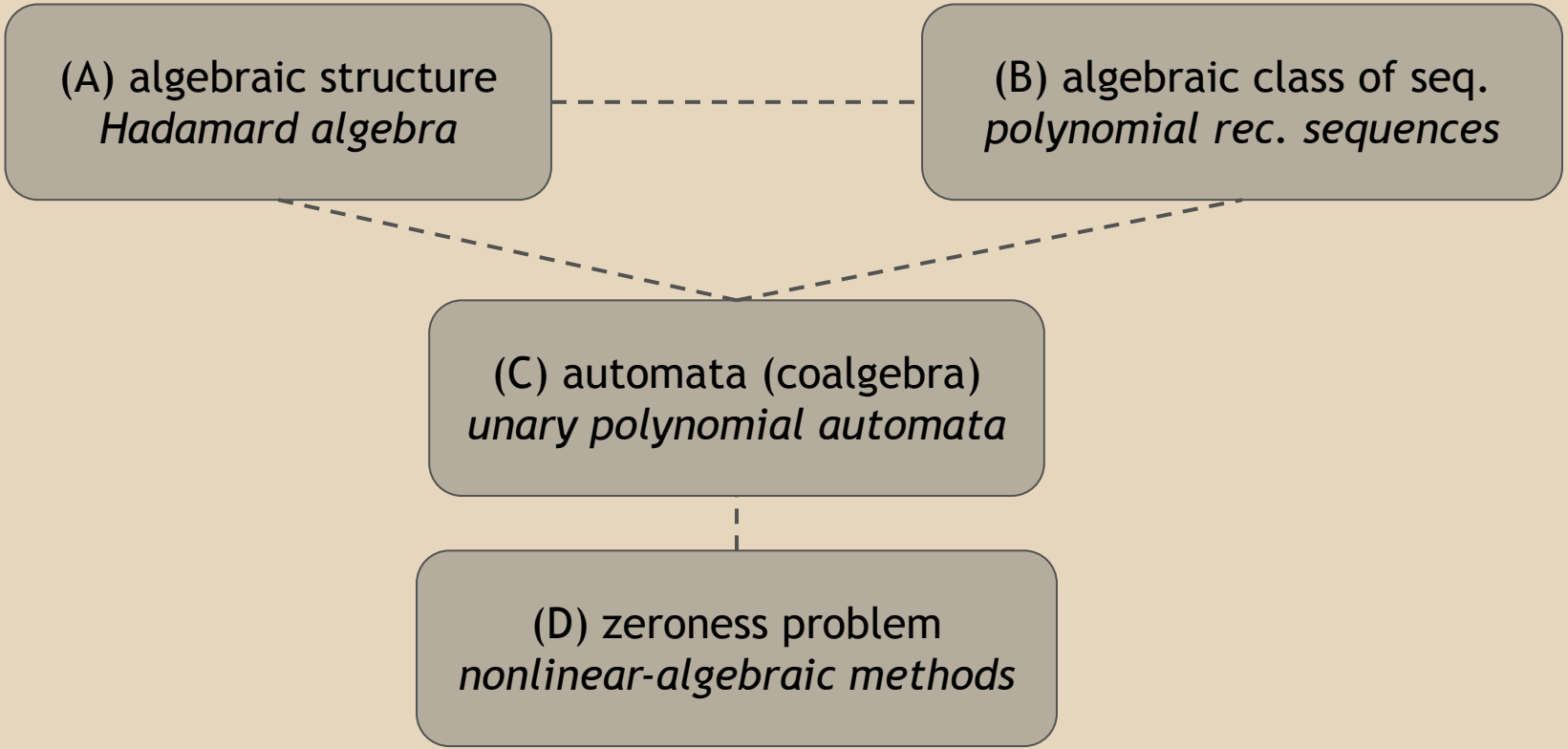
Coincidence Lemma (B) = (C).
The class of *polynomial recursive sequences* coincides with the class of sequences recognised by *unary polynomial automata*.

Zeroness problem (D)

Coincidence Lemma (B) = (C).
The class of *polynomial recursive sequences* coincides with the class of sequences recognised by *unary polynomial automata*.

Theorem.
The zeroness problem for polynomial recursive sequences is decidable.
Proof. Polynomial algebra (effective Hilbert finite basis theorem).

The big picture



permutations

Let f(n) = # permutations on n elements = n!

f(0) = 1
f(n+1) = $\sum \{ \binom{n}{m} \cdot f(m) \cdot f(n-m) \mid 0 \le m \le n \}$

Shuffle algebra (A)

Let ш be the *shuffle product* on $\mathbb{N} \to \mathbb{Q}$ (Hurwitz product / binomial conv.):

$$(f \text{ ш } g)(n) = \sum_{m} \binom{n}{m} \cdot f(m) \cdot g(n-m)$$

Shuffle algebra (A)

Let ш be the *shuffle product* on $\mathbb{N} \to \mathbb{Q}$ (Hurwitz product / binomial conv.):

$$(f \text{ ш } g)(n) = \sum_{m} \binom{n}{m} \cdot f(m) \cdot g(n-m)$$

Properties (**BAC**)

- Bilinear: $(f + g) \text{ ш } h = f \text{ ш } h + g \text{ ш } h$
- Associative: $(f \text{ ш } g) \text{ ш } h = f \text{ ш } (g \text{ ш } h)$
- Commutative: $f \text{ ш } g = g \text{ ш } f$

Shuffle algebra
$(\mathbb{N} \to \mathbb{Q}; 0, \cdot, +, \text{ш})$

permutations (B)

Let f(n) = # permutations on n elements = n!

f(0) = 1
σ f = f ш f

⇒ f is *shuffle finite*

Unary shuffle aut. (C)

Example of *unary shuffle automaton* $A = (X = \{ x \}, F, \Delta)$, where

- $F : X \to \mathbb{Q}$ is the *output function:* $F\ x = 1$
- $\Delta : X \to \mathbb{Q}[x]$ is the *polynomial transition function*: $\Delta\ x = x^2$

The *semantics* of A from x is the series $A[\![x]\!] = n \mapsto F(\Delta^n\ x)$

$$
\begin{array}{ccccc}
 & x & \text{—}\Delta\text{—>} & x^2 & \text{—}\Delta\text{—>}\ ??? \\
 & \downarrow F & & \downarrow F & \\
A[\![x]\!] = 1 & & & 1 &
\end{array}
$$

Unary shuffle automata (6)

The shuffle product of sequences satisfies the *product rule*

$\sigma\ (f \sqcup\!\sqcup g) = (\sigma f) \sqcup\!\sqcup g + f \sqcup\!\sqcup (\sigma g)$ (**Leibniz rule**)

We extend $\Delta : X \to \mathbb{Q}[x]$ to satisfy the analogous polynomial product rule.
Let $\Delta : \mathbb{Q}[x] \to \mathbb{Q}[x]$ be unique linear extension of $\Delta : X \to \mathbb{Q}[x]$ s.t.

$\Delta\ (pq) = (\Delta p) \cdot q + p \cdot (\Delta q)$
(derivation extension)

Example: $\Delta(x^n) = n \cdot x^{n-1} \cdot (\Delta\ x)$

$$\begin{array}{ccccc} x & \text{—}\Delta\text{—>} & x^2 & \text{—}\Delta\text{—>} & ??? \\ \downarrow F & & \downarrow F & & \\ A[[x]] = 1 & & 1 & & \end{array}$$

Unary shuffle exp (G)

The shuffle product of sequences satisfies the *product rule*

$\sigma\,(f \sqcup g) = (\sigma f) \sqcup g + f \sqcup (\sigma g)$ (**Leibniz rule**)

We extend $\Delta : X \to \mathbb{Q}[x]$ to satisfy the analogous polynomial product rule.
Let $\Delta : \mathbb{Q}[x] \to \mathbb{Q}[x]$ be unique linear extension of $\Delta : X \to \mathbb{Q}[x]$ s.t.

$\Delta\,(pq) = (\Delta p) \cdot q + p \cdot (\Delta q)$
(derivation extension)

Example: $\Delta(x^n) = n \cdot x^{n-1} \cdot (\Delta\, x)$

$$x \xrightarrow{\Delta} x^2 \xrightarrow{\Delta} 2x^3 \xrightarrow{\Delta} 6x^4 \xrightarrow{\Delta} \dots \xrightarrow{\Delta} n!\, x^{n+1}$$

$$\downarrow F \qquad \downarrow F \qquad \downarrow F \qquad \downarrow F \qquad\qquad\qquad \downarrow F$$

$$A[[x]] = 1 \qquad 1 \qquad\qquad 2 \qquad\quad 6 \qquad\quad \dots \qquad\quad n!$$

Zeroness problem (D)

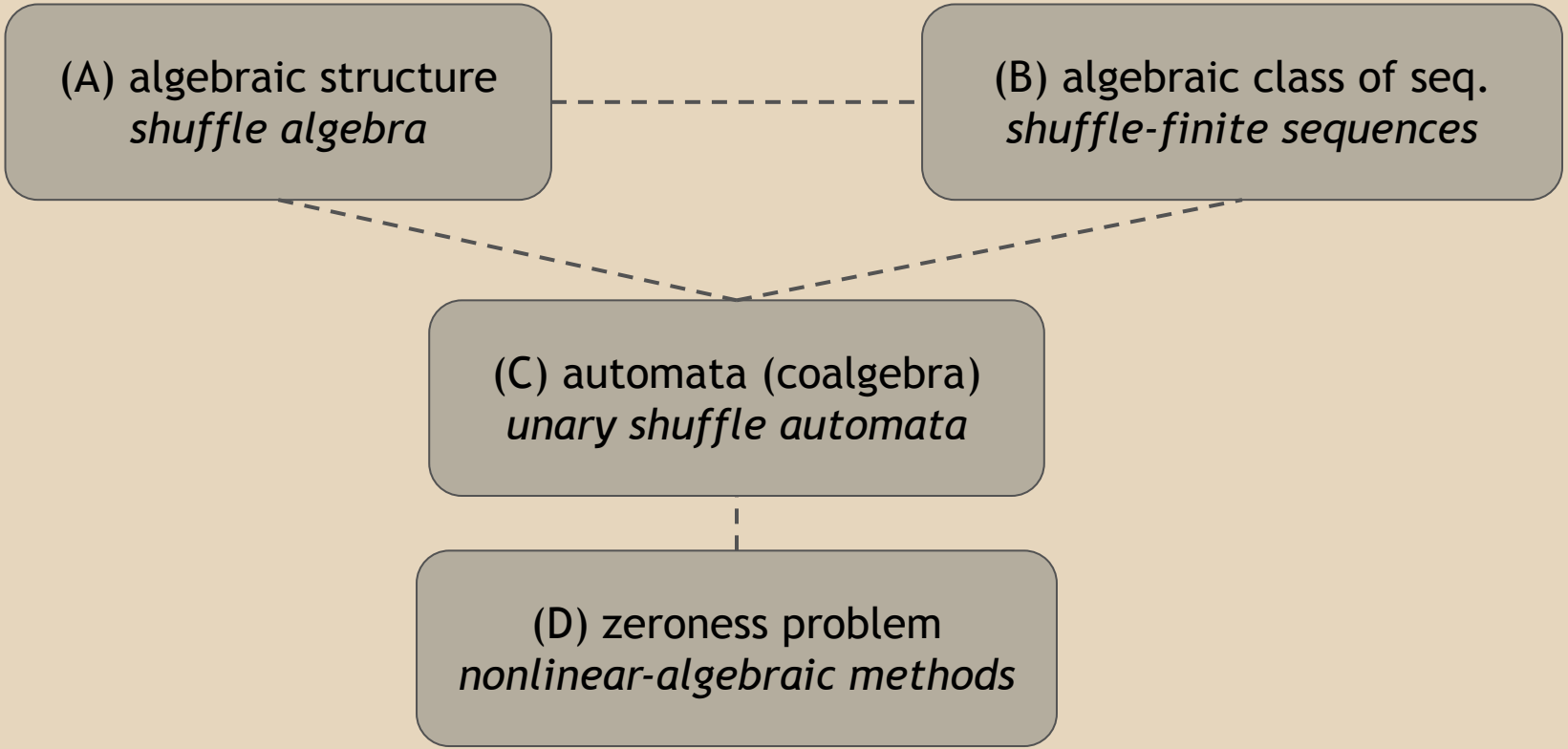
Coincidence Lemma (B) = (C).
The class of *shuffle finite sequences* coincides with the class of sequences recognised by *unary shuffle automata*.

Zeroness problem (D)

Coincidence Lemma (B) = (C).
The class of *shuffle finite sequences* coincides with the class of sequences recognised by *unary shuffle automata*.

Theorem.
The zeroness problem for shuffle finite sequences is decidable.
Proof. Polynomial algebra (effective Hilbert finite basis theorem).

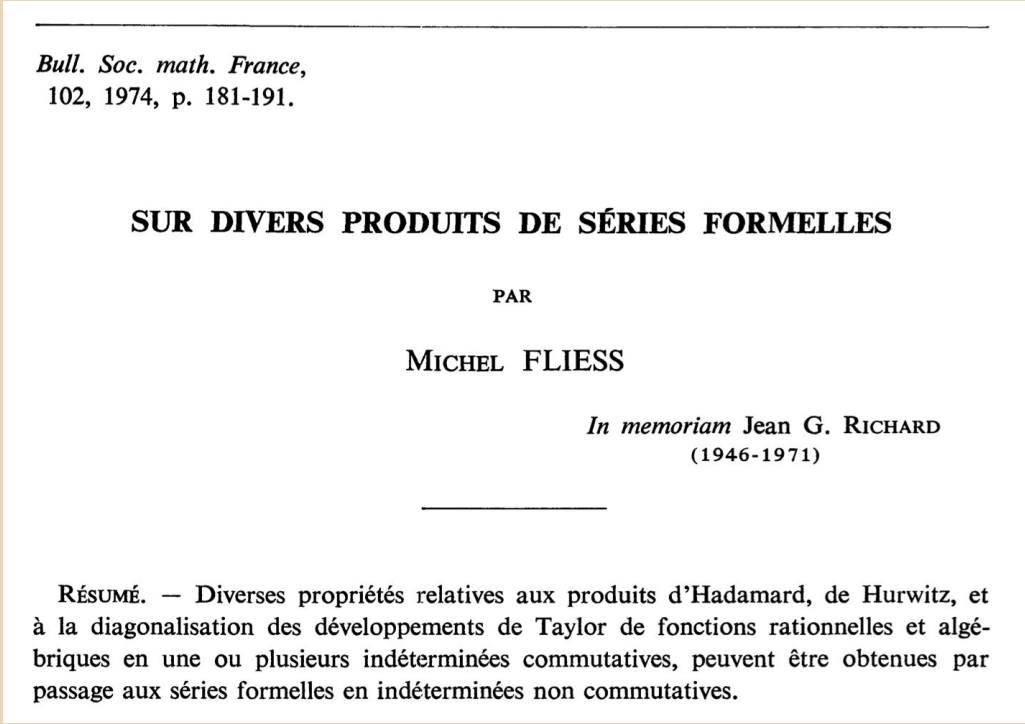
The big picture



Products of series

In computer science,
Fliess (1974) studied:

- Hadamard product
- Shuffle (Hurwitz) product



Bull. Soc. math. France,
102, 1974, p. 181-191.

SUR DIVERS PRODUITS DE SÉRIES FORMELLES

PAR

MICHEL FLIESS

In memoriam Jean G. RICHARD
(1946-1971)

RÉSUMÉ. — Diverses propriétés relatives aux produits d'Hadamard, de Hurwitz, et à la diagonalisation des développements de Taylor de fonctions rationnelles et algébriques en une ou plusieurs indéterminées commutatives, peuvent être obtenues par passage aux séries formelles en indéterminées non commutatives.

Process algebra interpretation

	product rule
Hadamard	$\sigma\ (f \odot g) = \sigma\ f \odot \sigma\ g$ both processes make a step

Process algebra interpretation

	product rule
Hadamard	$\sigma\ (f \odot g) = \sigma\ f \odot \sigma\ g$ both processes make a step
shuffle	$\sigma\ (f \shuffle g) = (\sigma\ f) \shuffle g + f \shuffle (\sigma\ g)$ exactly one process make a step

Process algebra interpretation

	product rule
Hadamard	$\sigma\ (f \odot g) = \sigma\ f \odot \sigma\ g$ both processes make a step
shuffle	$\sigma\ (f \shuffle g) = (\sigma\ f) \shuffle g + f \shuffle (\sigma\ g)$ exactly one process make a step
infiltration	$\sigma\ (f \uparrow g) = (\sigma\ f) \uparrow g + f \uparrow (\sigma\ g) + (\sigma\ f) \uparrow (\sigma\ g)$ either process makes a step

Infiltration product in TCS

Math. Struct. in Comp. Science: page 1 of 29.
doi:10.1017/S0960129517000159

Newton series, coinductively: a comparative study of composition

HENNING BASOLD[†], HELLE HVID HANSEN[‡],
JEAN-ÉRIC PIN[§] and JAN RUTTEN[¶]

[†]*Radboud University Nijmegen, P.O. Box 9010, 6500GL Nijmegen, The Netherlands, and CWI Amsterdam, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands.*
Email: h.basold@cs.ru.nl
[‡]*Delft University of Technology, P.O. Box 5015, 2600 GA Delft, The Netherlands.*
Email: h.h.hansen@tudelft.nl
[§]*Université Paris Denis Diderot and CNRS, 75205 Paris Cedex 13, France.*
Email: Jean-Eric.Pin@liafa.univ-paris-diderot.fr
[¶]*CWI Amsterdam, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, and Radboud University Nijmegen, P.O. Box 9010, 6500GL Nijmegen, The Netherlands.*
Email: jjmmrutten@gmail.com

Received 4 May 2016; revised 3 April 2017

Objectives

Product	BAC	Automata	Decidable *zeroness problem*
Hadamard	✓	polynomial automata [1]	✓ [1, Theorem 4]
shuffle	✓	shuffle automata [2]	✓ [2, Theorem 1]
infiltration	✓	infiltration automata [3]	✓ [3, Theorem 2]

[1] Benedikt, Duff, Sharad, Worrell: “Polynomial automata: Zeroness and applications” (LICS’17)
[2] C: “Weighted Basic Parallel Processes and Combinatorial Enumeration” (CONCUR’24)
[3] C: “The commutativity problem for effective varieties of formal series, and applications” (LICS’25)

Objectives

Product	BAC	Automata	Decidable *zeroness problem*
Hadamard	✓		✓ [1, Theorem 4]
shuffle	✓		✓ [2, Theorem 1]
infiltration	✓		✓ [3, Theorem 2]

Can we understand these results in a *unified framework*?

[1] Benedikt, Duff, Sharad, Worrell: “Polynomial automata: Zeroness and applications” (LICS’17)
[2] C: “Weighted Basic Parallel Processes and Combinatorial Enumeration” (CONCUR’24)
[3] C: “The commutativity problem for effective varieties of formal series, and applications” (LICS’25)

Part II

- Products defined by product rules
- Characterisation of commutative algebras of series

Enter product rules

Algebra and Coalgebra of Stream Products

Michele Boreale
University of Florence, Italy

Daniele Gorla
"Sapienza" University of Rome, Italy

(CONCUR 2021)

Abstract

We study connections among polynomials, differential equations and streams over a field $\mathbb{K}$, in terms of algebra and coalgebra. We first introduce the class of (F, G)-products on streams, those where the stream derivative of a product can be expressed as a polynomial of the streams themselves and their derivatives. Our first result is that, for every (F, G)-product, there is a canonical way to construct a transition function on polynomials such that the induced unique final coalgebra morphism from polynomials into streams is the (unique) $\mathbb{K}$-algebra homomorphism – and vice-versa. This implies one can reason algebraically on streams, via their polynomial representation. We apply this result to obtain an algebraic-geometric decision algorithm for polynomial stream equivalence, for an underlying generic (F, G)-product. As an example of reasoning on streams, we focus on specific products (convolution, shuffle, Hadamard) and show how to obtain closed forms of algebraic generating functions of combinatorial sequences, as well as solutions of nonlinear ordinary differential equations.

P-products

Consider polynomials of the form $P(x, \dot{x}, y, \dot{y}) \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-product* is a binary operation on sequences "$*$" s.t.

$$(f * g)(0) = f(0) * g(0)$$
$$\sigma\,(f * g) = P(f, \sigma\, f, g, \sigma\, g)$$

P-products

Consider polynomials of the form $P(x, \dot{x}, y, \dot{y}) \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-product* is a binary operation on sequences "$*$" s.t.

$$(f * g)(0) = f(0) * g(0)$$
$$\sigma\,(f * g) = P(f, \sigma\, f, g, \sigma\, g)$$

P-product	product rule	P
Hadamard	$\sigma\,(f \odot g) = \sigma\, f \odot \sigma\, g$	$\dot{x}\,\dot{y}$
shuffle	$\sigma\,(f \shuffle g) = (\sigma\, f) \shuffle g + f \shuffle (\sigma\, g)$	$\dot{x}\,y + x\,\dot{y}$
infiltration	$\sigma\,(f \uparrow g) = (\sigma\, f) \uparrow g + f \uparrow (\sigma\, g) + (\sigma\, f) \uparrow (\sigma\, g)$	$\dot{x}\,y + x\,\dot{y} + \dot{x}\,\dot{y}$

Boreale and Gorla (CONCUR'21)

1. Consider a certain *technical condition* (X)
2. The Hadamard, shuffle, infiltration product rules satisfy (X)
3. Product rules satisfying (X) give rise to
    a. BAC products
    b. automata with decidable zeroness

Product	BAC	Automata	Decidable *zeroness*
Hadamard	✓	Polynomial automata	✓
shuffle	✓	shuffle automata	✓
infiltration	✓	infiltration automata	✓

Limitations

Is condition (X)

- necessary for BAC?
- necessary for decidability of zeroness?
- decidable?

▶ **Definition 3.1** ((F,G)-product on streams). *Let $(\Sigma, +, \pi, 0, 1_\pi)$ be a $\mathbb{K}$-algebra, $F \in \mathbb{K}[x, y_1, y_2, y_3, y_4]$ and $G \in \mathbb{K}[y_1]$. We say that π is a (F,G)-product if, for each $\sigma, \tau \in \Sigma$, the following equations are satisfied:*

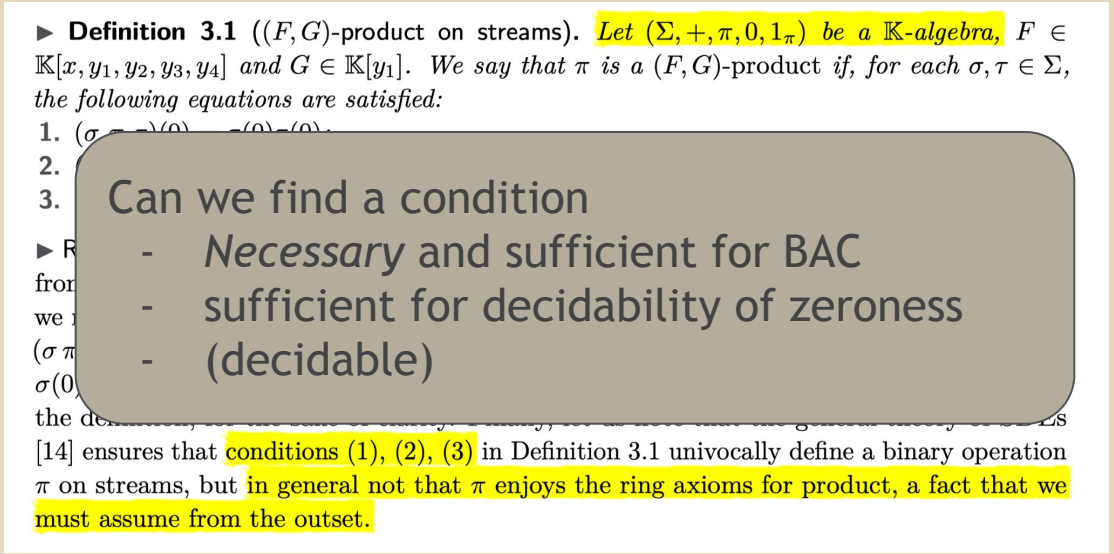
1. $(\sigma\ \pi\ \tau)(0) = \sigma(0)\tau(0)$;
2. $(\sigma\ \pi\ \tau)' = F(x, \sigma, \sigma', \tau, \tau')$;
3. $1_\pi(0) = 1$ *and* $1'_\pi = G(1_\pi)$.

▶ Remark 3.2. Notice that $1_\pi(0) = 1$ in Definition 3.1(3) is a necessary condition, that follows from Definition 3.1(1). Indeed, let $1_\pi(0) = r \in \mathbb{K}$. Since 1_π is the identity of π, for every σ we must have $\sigma\,\pi\,1_\pi = \sigma$, hence $(\sigma\,\pi\,1_\pi)(0) = \sigma(0)$. On the other hand, by Definition 3.1(1), $(\sigma\,\pi\,1_\pi)(0) = \sigma(0)\,1_\pi(0) = \sigma(0)\,r$. As σ is arbitrary, we can take $\sigma(0) \neq 0$ and multiply $\sigma(0)\,r = \sigma(0)$ by $\sigma(0)^{-1}$; this gives $r = 1$. However, we prefer to keep $1_\pi(0) = 1$ explicit in the definition, for the sake of clarity. Finally, let us note that the general theory of SDEs [14] ensures that conditions (1), (2), (3) in Definition 3.1 univocally define a binary operation π on streams, but in general not that π enjoys the ring axioms for product, a fact that we must assume from the outset.

Limitations

Is condition (X)

- necessary for BAC?
- necessary for decidability of zeroness?
- decidable?



▶ **Definition 3.1** ((F, G)-product on streams). *Let* $(\Sigma, +, \pi, 0, 1_\pi)$ *be a* $\mathbb{K}$*-algebra,* $F \in \mathbb{K}[x, y_1, y_2, y_3, y_4]$ *and* $G \in \mathbb{K}[y_1]$. *We say that* π *is a* (F, G)-product *if, for each* $\sigma, \tau \in \Sigma$, *the following equations are satisfied:*

1. $(\sigma \cdot \tau) \cdot$
2.
3.

▶ R
fro
we
$(\sigma \pi$
$\sigma(0$
the de

[14] ensures that conditions (1), (2), (3) in Definition 3.1 univocally define a binary operation π on streams, but in general not that π enjoys the ring axioms for product, a fact that we must assume from the outset.

Can we find a condition

- *Necessary* and sufficient for BAC
- sufficient for decidability of zeroness
- (decidable)

BAC (non-)examples

product rule	P	bilin	assoc	comm
$\sigma\ (f * g) = 0$	0	✓	✓	✓
$\sigma\ (f * g) = f * g$	$x\ y$	✓	✓	✓
$\sigma\ (f \odot g) = \sigma\ f \odot \sigma\ g$	$\dot{x}\ \dot{y}$	✓	✓	✓
$\sigma\ (f \amalg g) = (\sigma\ f) \amalg g + f \amalg (\sigma\ g)$	$\dot{x}\ y + x\ \dot{y}$	✓	✓	✓
$\sigma\ (f \uparrow g) = (\sigma\ f) \uparrow g + f \uparrow (\sigma\ g) + (\sigma\ f) \uparrow (\sigma\ g)$	$\dot{x}\ y + x\ \dot{y} + \dot{x}\ \dot{y}$	✓	✓	✓
$\sigma\ (f * g) = f^2 * g^2$	$x^2\ y^2$	❌		
$\sigma\ (f * g) = (\sigma\ f) * g$	$\dot{x}\ y$			❌

Necessary BAC conditions

Commutativity:

$f * g = g * h$

$\Rightarrow \sigma\,(f * g) = \sigma\,(g * h)$

$\Rightarrow P(f, \sigma f, g, \sigma g) = P(g, \sigma g, f, \sigma f)$

Necessary BAC conditions

Commutativity:

$f * g = g * h$

$\Rightarrow \sigma\,(f * g) = \sigma\,(g * h)$

$\Rightarrow P(f, \sigma f, g, \sigma g) = P(g, \sigma g, f, \sigma f)$

Now extract the constant term:

$\Rightarrow (P(f, \sigma f, g, \sigma g))\,(0) = (P(g, \sigma g, f, \sigma f))\,(0)$

$\Rightarrow P(f(0), f(1), g(0), g(1)) = P(g(0), g(1), f(0), f(1))$

$\Rightarrow P(x, \dot{x}, y, \dot{y}) = P(y, \dot{y}, x, \dot{x})$

BAC characterisation

A *P-product* is a binary operation on series "$*$" s.t.

$$(f * g)(0) = f(0) * g(0) \quad \text{and} \quad \sigma\,(f * g) = P(f, \sigma\, f, g, \sigma\, g)$$

The following conditions are *necessary* for a BAC P-product:

Bilinearity	$P(x + y, \dot{x} + \dot{y}, z, \dot{z}) = P(x, \dot{x}, z, \dot{z}) + P(y, \dot{y}, z, \dot{z})$	(1)
Associativity	$P(x, \dot{x}, yz, P(y, \dot{y}, z, \dot{z})) = P(xy, P(x, \dot{x}, y, \dot{y}), z, \dot{z})$	(2)
Commutativity	$P(x, \dot{x}, y, \dot{y}) = P(y, \dot{y}, x, \dot{x})$	(3)

BAC characterisation

A *P-product* is a binary operation on series "$*$" s.t.

$$(f * g)(0) = f(0) * g(0) \quad \text{and} \quad \sigma\,(f * g) = P(f, \sigma\, f, g, \sigma\, g)$$

The following conditions are *necessary* for a BAC P-product:

Bilinearity	$P(x + y, \dot{x} + \dot{y}, z, \dot{z}) = P(x, \dot{x}, z, \dot{z}) + P(y, \dot{y}, z, \dot{z})$	(1)
Associativity	$P(x, \dot{x}, yz, P(y, \dot{y}, z, \dot{z})) = P(xy, P(x, \dot{x}, y, \dot{y}), z, \dot{z})$	(2)
Commutativity	$P(x, \dot{x}, y, \dot{y}) = P(y, \dot{y}, x, \dot{x})$	(3)

The conditions above are also *sufficient*.

Theorem. A P-product is BAC iff it satisfies (1), (2), and (3).
Proof. By coinduction.

BAC classification

A *P-product* is a binary operation on series "$*$" s.t.

$$(f * g)(0) = f(0) * g(0) \quad \text{and} \quad \sigma\,(f * g) = P(f, \sigma f, g, \sigma g)$$

A product rule P is *simple* if there are constants $\alpha, \beta, \gamma \in \mathbb{Q}$:

$$P(x, \dot{x}, y, \dot{y}) = \alpha \cdot xy + \beta \cdot (\dot{x}y + x\dot{y}) + \gamma \cdot \dot{x}\dot{y}$$
$$\text{s.t. } \alpha\,\gamma = \beta\,(\beta - 1)$$

product	**α**	**β**	**γ**
Hadamard	0	0	1
shuffle	0	1	0
infiltration	0	1	1
shuffle-infiltration	0	1	γ

BAC classification

A *P-product* is a binary operation on series "$*$" s.t.

$$(f * g)(0) = f(0) * g(0) \quad \text{and} \quad \sigma\,(f * g) = P(f, \sigma f, g, \sigma g)$$

A product rule P is *simple* if there are constants $\alpha, \beta, \gamma \in \mathbb{Q}$:

$$P(x, \dot{x}, y, \dot{y}) = \alpha \cdot xy + \beta \cdot (\dot{x}y + x\dot{y}) + \gamma \cdot \dot{x}\dot{y}$$
$$\text{s.t. } \alpha\,\gamma = \beta\,(\beta - 1)$$

Theorem. A P-product is BAC iff P is simple.
Proof. Apply the previous characterisation.

product	**α**	**β**	**γ**
Hadamard	0	0	1
shuffle	0	1	0
infiltration	0	1	1
shuffle-infiltration	0	1	γ

Cauchy product

Let $*$ be the concatenation (Cauchy) product.
$(f * g)(n) = \sum_{\square} f(m) \cdot g(m - n)$

- bilinear ✓
- associative ✓
- commutative ✓

Brzozowski's product rule
$\sigma (f * g) = (\sigma f) * g + f(0) \cdot (\sigma g)$

NOT a P-product! (for any P)

Part III

- P-automata
- Decidability of the zeroness problem

P-automata: Syntax

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-automaton* is a tuple

$$A = (X, F, \Delta), \text{ where}$$

- $X = \{x_1, \ldots, x_n\}$ is a finite set of commuting *variables*
- $F : X \to \mathbb{Q}$ is the *output function*
- $\Delta : X \to \mathbb{Q}[X]_0$ is the *transition function*

P-automata: Semantics

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-automaton* $A = (X, F, \Delta)$ defines a *semantic function*

$$A[\![\_]\!] : \mathbb{Q}[X]_0 \to (\mathbb{N} \to \mathbb{Q})$$

mapping a *configuration* $p \in \mathbb{Q}[X]_0$ to a sequence $A[\![p]\!] : \mathbb{N} \to \mathbb{Q}$.

Technical remark:
$\mathbb{Q}[X]_0$ = polynomials without constant term

P-functions

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
$\Delta : \mathbb{Q}[X]_0 \to \mathbb{Q}[X]_0$ is a *P-function* if

- It is linear

  $\Delta (p + q) = \Delta p + \Delta q$

  $\Delta (c\ p) = c\ \Delta p,\ c \in \mathbb{Q}$

- It satisfies the product rule

  $\Delta (p\ q) = P(p, \Delta p, q, \Delta q)$

P-functions

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
$\Delta : \mathbb{Q}[X]_0 \to \mathbb{Q}[X]_0$ is a *P-function* if

- It is linear
  $\Delta (p + q) = \Delta p + \Delta q$
  $\Delta (c\ p) = c\ \Delta p,\ c \in \mathbb{Q}$

- It satisfies the product rule
  $\Delta (p\ q) = P(p, \Delta p, q, \Delta q)$

Extension Lemma. Let P be a BAC product rule.
Every function $\Delta : X \to \mathbb{Q}[X]_0$ extends (uniquely) to a P-function $\mathbb{Q}[X]_0 \to \mathbb{Q}[X]_0$.

P-automata: Semantics

Fix $A = (X, F, \Delta)$. We define the semantic function $A[\![\_]\!] : \mathbb{Q}[X]_0 \to (\mathbb{N} \to \mathbb{Q})$.

$\Delta : X \to \mathbb{Q}[X]_0$

extend to a unique *P-function*

$\Delta : \mathbb{Q}[X]_0 \to \mathbb{Q}[X]_0$

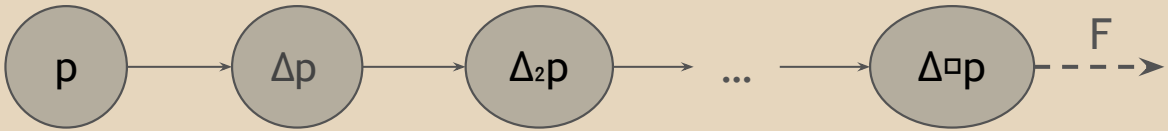
$\Delta (pq) = P(p, \Delta p, q, \Delta q)$

$\Delta : \mathbb{N} \to \mathbb{Q}[X]_0 \to \mathbb{Q}[X]_0$

extend *homomorphically*

$\Delta_0 p = p$

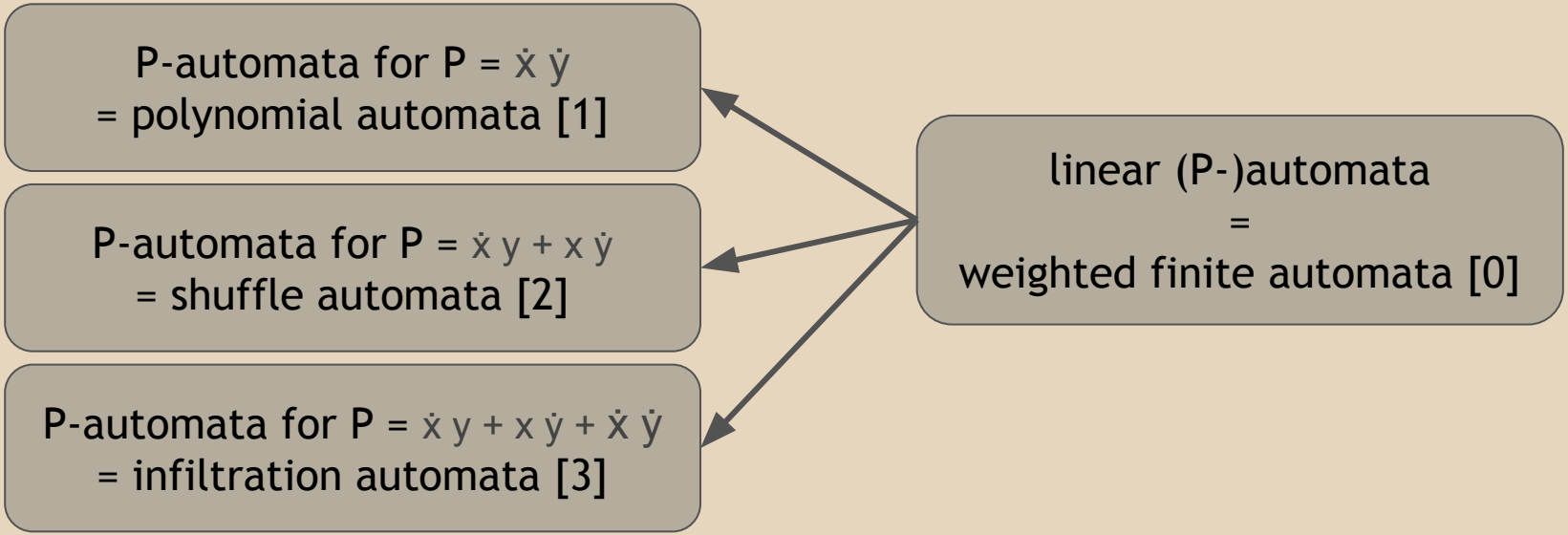
$\Delta^{\square}{}_{+1} p = \Delta^{\square} (\Delta p)$



output

$A[\![p]\!](n) := F (\Delta^{\square} p)$

P-automata



[0] Schützenberger: “On the definition of a family of automata” (IC’61)
[1] Benedikt, Duff, Sharad, Worrell: “Polynomial automata: Zeroness and applications” (LICS’17)
[2] C: “Weighted Basic Parallel Processes and Combinatorial Enumeration” (CONCUR’24)
[3] C: “The commutativity problem for effective varieties of formal series, and applications” (LICS’25)

(B) = (C)

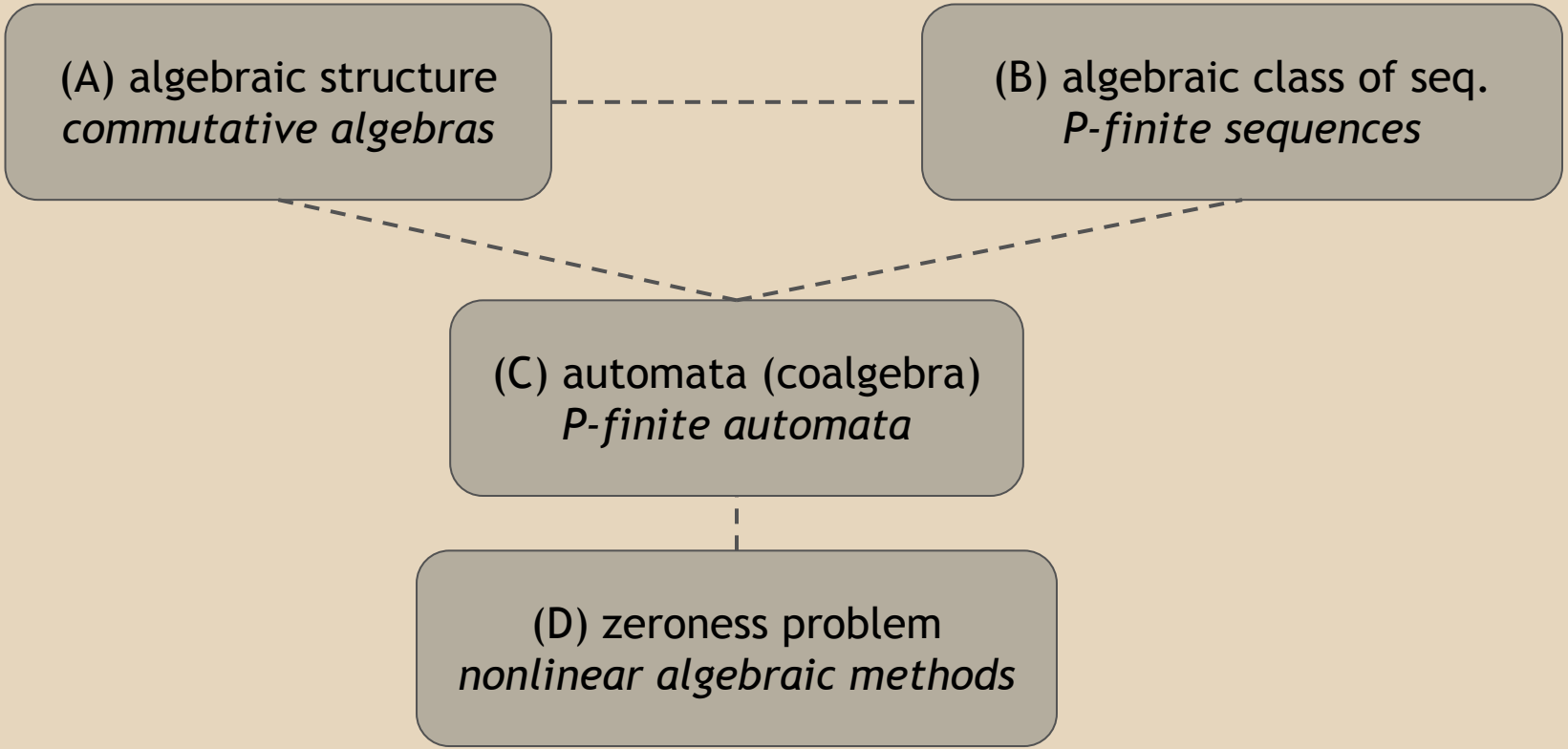
Coincidence Lemma.
A series is P-finite iff it is recognised by P-automaton.

Zeroness problem (D)

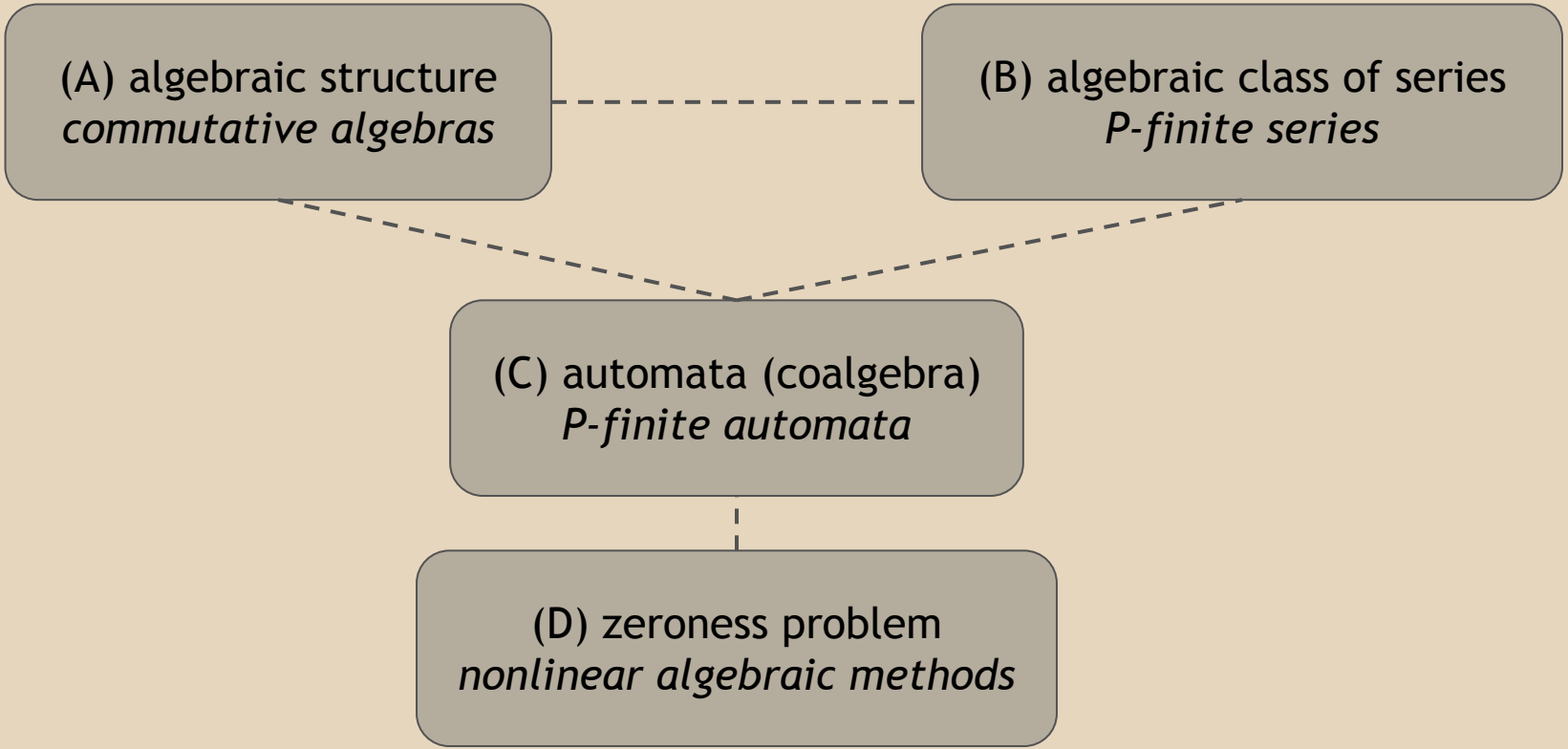
INPUT: A P-automaton A and an initial configuration p
OUTPUT: yes iff $A[\![p]\!] = 0$

Theorem. Let P be a simple product rule.
The zeroness problem is decidable for P-finite series
(= series recognised by P-automata).
Proof. Hilbert's finite basis theorem.

The big picture



… even for series $\Sigma^* \longrightarrow \mathbb{Q}$



End of the world

End of the world (2)

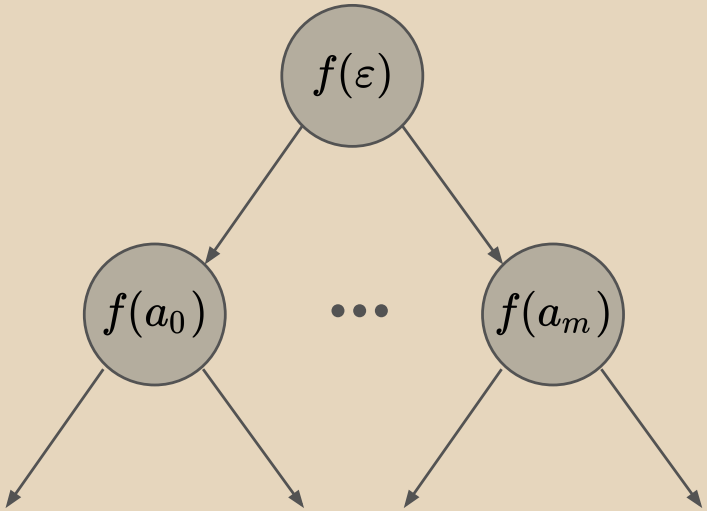
End of the world (3)

What is a series?

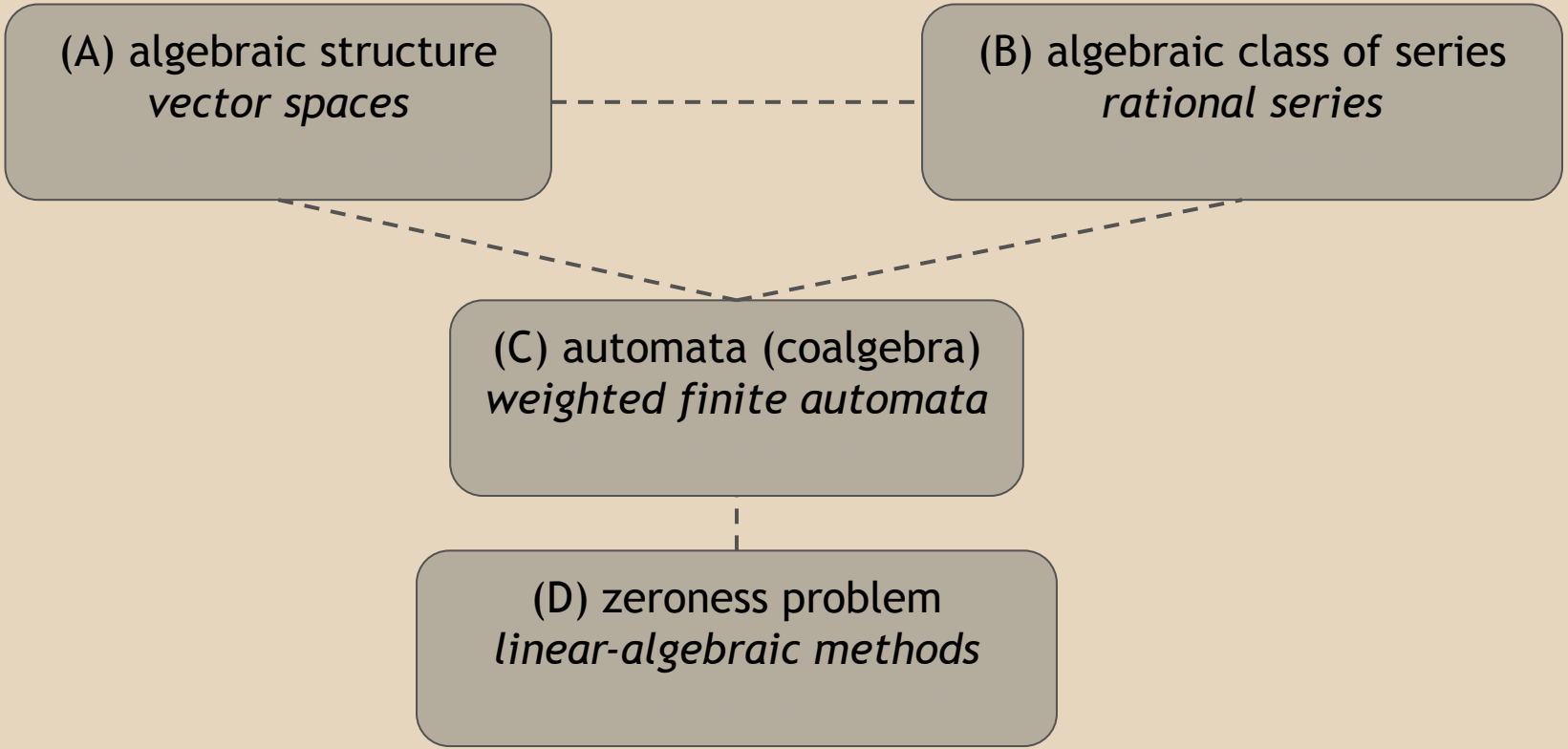
Fix a finite alphabet $\Sigma = \{a, b\}$.
A *series* is a mapping $f : \Sigma^* \to \mathbb{Q}$
(sometimes written $\mathbb{Q}\langle\langle a, b \rangle\rangle$)

Examples

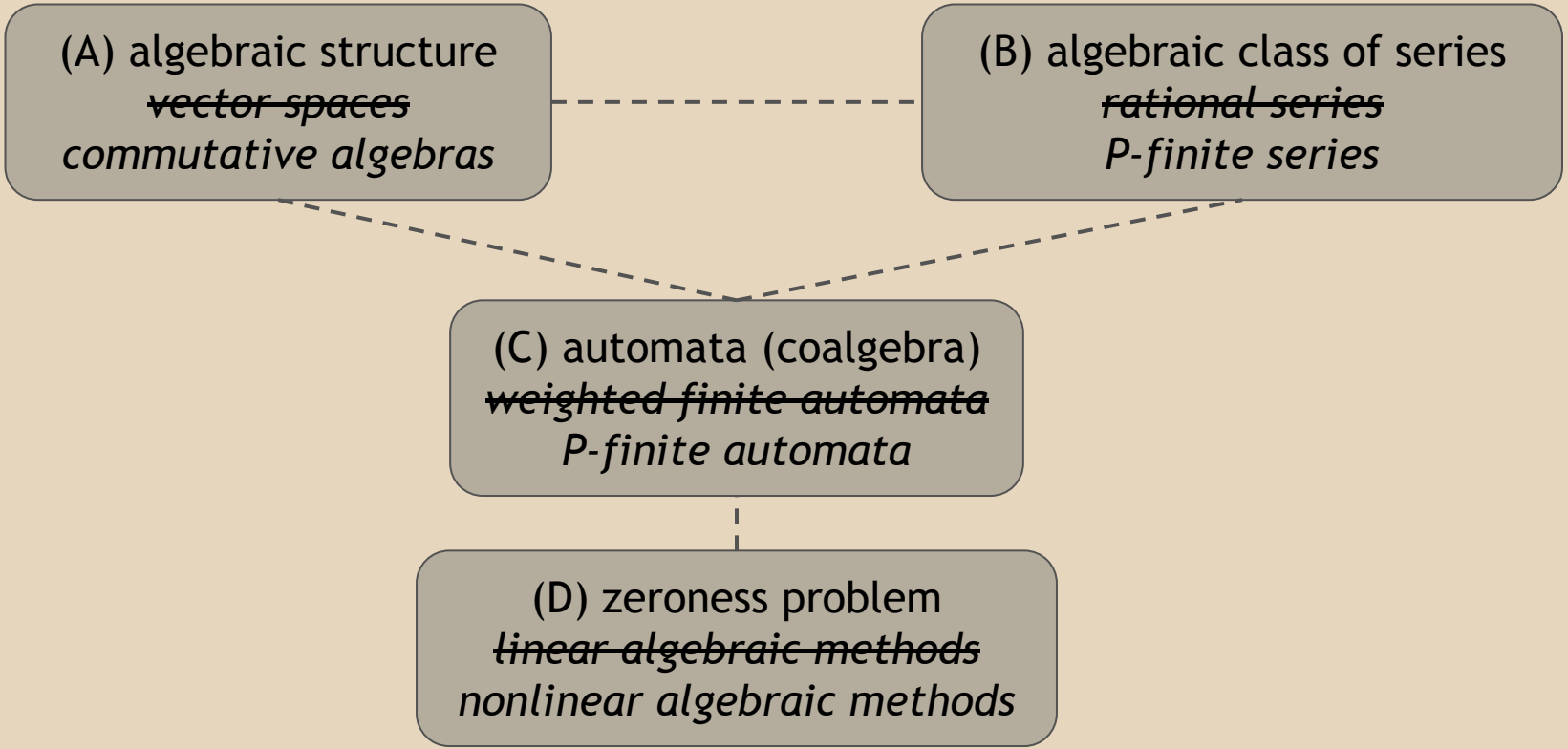
- f(ab) = 2, f(ba) = -3, otherwise f(w) = 0
  - written $f = 2 \cdot ab - 3 \cdot ba$
- $f = 1 \cdot \varepsilon + 1 \cdot a + 1 \cdot aa + \ldots$



The big picture



The big picture



Series over Σ = { a }

When Σ = {a}, a series is the same as

- a *sequence* of rationals $f : \mathbb{N} \to \mathbb{Q}$
- a univariate *power series* $f = \sum_n f_n \cdot x^n / n! : \mathbb{Q}[[x]]$

OVERARCHING GOAL
extend the theory of number sequences and power series to series

Vector space structure (A)

Number sequences $f, g : \mathbb{N} \to \mathbb{Q}$

- *Scalar multiplication*:
  $(c \cdot f)(n) := c \cdot f(n)$, $c \in \mathbb{Q}$, $n \in \mathbb{N}$

- *Addition*:
  $(f + g)(n) := f(n) + g(n)$, $n \in \mathbb{N}$

vector space
$(\mathbb{N} \to \mathbb{Q}; 0, \cdot, +)$

Power series $f = \sum_n f_n \cdot x^n / n!$, $g : \mathbb{Q}[[x]]$

- *Scalar multiplication*:
  $c \cdot f := \sum_n (c \cdot f_n) \cdot x^n / n!$, $c \in \mathbb{Q}$

- *Addition*:
  $f + g := \sum_n (f_n + g_n) \cdot x^n / n!$

vector space
$(\mathbb{Q}[[x]]; 0, \cdot, +)$

Transition structure

Number sequence $f : \mathbb{N} \to \mathbb{Q}$

- *Left shift*

$\sigma : (\mathbb{N} \to \mathbb{Q}) \to (\mathbb{N} \to \mathbb{Q})$
$(\sigma\, f)(n) := f(n + 1),\ n \in \mathbb{N}$

vector space
$(\mathbb{N} \to \mathbb{Q};\ 0,\ \cdot,\ +)$

Power series $f = \sum_{\square} f_{\square} \cdot x^n / n! : \mathbb{Q}[[\, x \,]]$

- *Derivative*

$\partial_x : \mathbb{Q}[[\, x \,]] \to \mathbb{Q}[[\, x \,]]$
$\partial_x f := \sum_{\square} f_{\square+1} \cdot x^n / n!$

vector space
$(\mathbb{Q}[[\, x \,]];\ 0,\ \cdot,\ +)$

partitions (B)

Let f(n) = # partitions on n elements = B□ (Bell numbers)
and g(n) = # sets of n elements = 1

f(0) = 0
g(0) = 1

σ f = f ш g = n ↦ ∑□ (n choose m) · f(m)
σ g = g

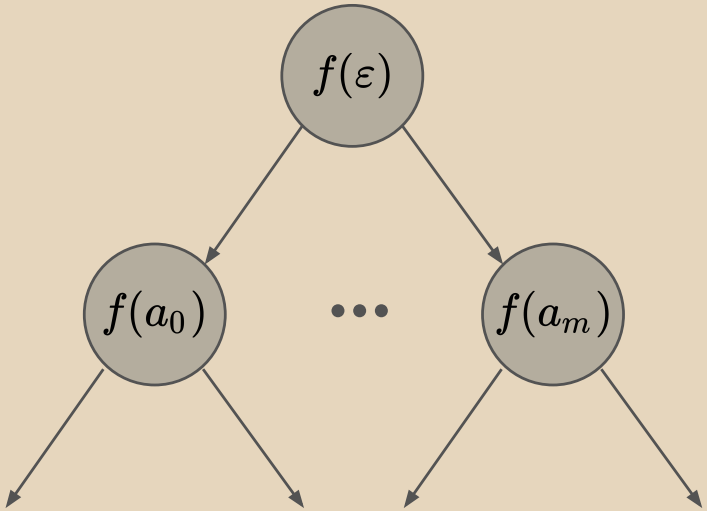
⇒ f, g are *shuffle finite*

What is a series?

Fix a finite alphabet $\Sigma = \{a, b\}$.
A *series* is a mapping $f : \Sigma^* \to \mathbb{Q}$
(sometimes written $\mathbb{Q}\langle\langle a, b \rangle\rangle$)

Examples

- f(ab) = 2, f(ba) = -3, otherwise f(w) = 0
  - written $f = 2 \cdot ab - 3 \cdot ba$
- $f = 1 \cdot \varepsilon + 1 \cdot a + 1 \cdot aa + \ldots$



Vector space structure

Series $f, g : \mathbb{Q}\langle\langle\Sigma\rangle\rangle$

- *Scalar multiplication*:

  $(c \cdot f)(w) := c \cdot f(w)$ for all $c \in \mathbb{Q}$, $w \in \Sigma^*$

- *Addition*:

  $(f + g)(w) := f(w) + g(w)$ for all $w \in \Sigma^*$

vector space
$(\mathbb{Q}\langle\langle\Sigma\rangle\rangle; 0, \cdot, +)$

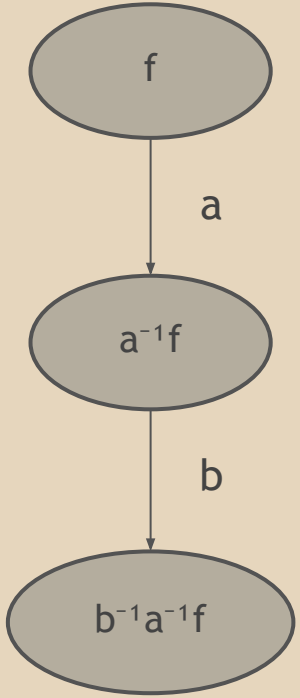
Left shift operator

For an input symbol $a \in \Sigma$ and a series $f : \mathbb{Q}\langle\langle\Sigma\rangle\rangle$
let $a^{-1}f : \mathbb{Q}\langle\langle\Sigma\rangle\rangle$ be the series s.t.

$$(a^{-1}f)(w) = f(a \cdot w), \qquad \forall\ w \in \Sigma^*$$

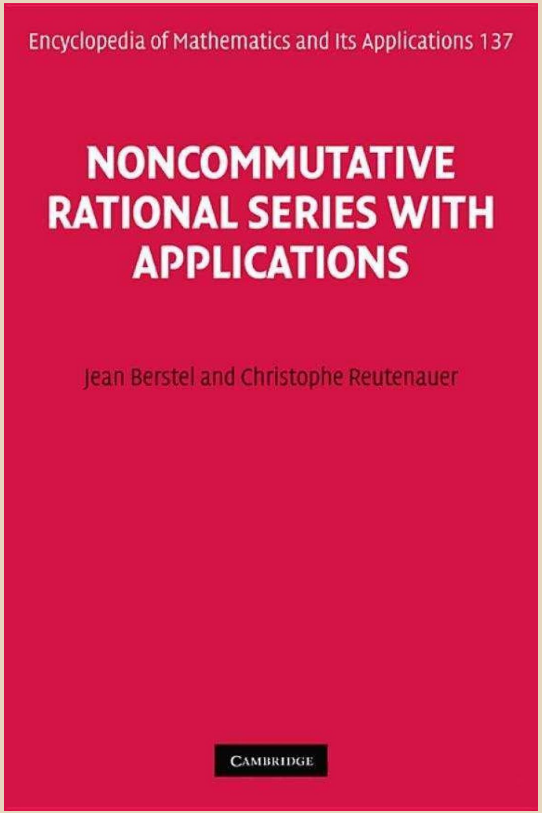
The linear operator $a^{-1} : \mathbb{Q}\langle\langle\Sigma\rangle\rangle \to \mathbb{Q}\langle\langle\Sigma\rangle\rangle$ is called *left shift*

It endows the set of series with the structure of an
(infinite) automaton (= coalgebra)



Mathematics of series

Berstel, Reutenauer:
Noncommutative rational series with applications
(2010)



Hadamard product

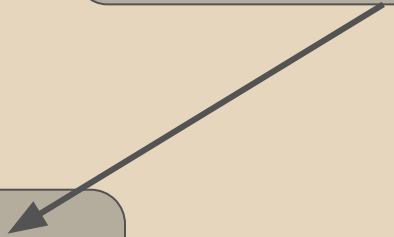
Hadamard product (elementwise):

$(f \odot g)(w) := f(w) \cdot g(w)$ for all $w \in \Sigma^*$

Properties (**BAC**)

- Bilinear: $(f + g) \odot h = f \odot h + g \odot h$
- Associative: $(f \odot g) \odot h = f \odot (g \odot h)$
- Commutative: $f \odot g = g \odot f$

associative,
commutative
(not necessarily with unit)



Hadamard algebra
$(\mathbb{Q}\langle\langle\Sigma\rangle\rangle; 0, \cdot, +, \odot)$

Inductive vs. coinductive

We can also define the Hadamard product on words by *induction* on their length:

- Base: $\varepsilon \odot \varepsilon = \varepsilon$
- Step: $(a\ u) \odot (a\ v) = u \odot v$
  $(a\ u) \odot (b\ v) = 0$, if $a \neq b$

Extend $\odot$ to all series by linearity and continuity

We can also define it by *coinduction*…

Hadamard coinductively

Coinductive ~~description~~ **definition**

$$(f \odot g)(\varepsilon) = f(\varepsilon) \cdot g(\varepsilon)$$

$$a^{-1}(f \odot g) = a^{-1}f \odot a^{-1}g$$

product rule
(homomorphism)

for all $a \in \Sigma$

$\Rightarrow a^{-1}\_$ is a *homomorphism*

Shuffle product by example

on words:

$b \shuffle c = bc + cb$

$ab \shuffle c = cab + acb + abc$

$ab \shuffle a = 2aab + aba$

on series: by linearity and continuity

Shuffle algebra

Coinductive definition

$$(f \sqcup g)(\varepsilon) = f(\varepsilon) \cdot g(\varepsilon)$$

$$a^{-1}(f \sqcup g) = (a^{-1}f) \sqcup g + f \sqcup (a^{-1}g)$$

product rule **(Leibniz rule)**

for all $a \in \Sigma$

$\Rightarrow a^{-1}\_$ is a *derivation*

Properties (**BAC**)

- Bilinear: $(f + g) \sqcup h = f \sqcup h + g \sqcup h$
- Associative: $(f \sqcup g) \sqcup h = f \sqcup (g \sqcup h)$
- Commutative: $f \sqcup g = g \sqcup f$

$\Rightarrow$ Shuffle algebra $(\mathbb{Q}\langle\langle\Sigma\rangle\rangle; 0, \cdot, +, \sqcup)$

Applications of ш-uffle

Homological algebra
- Eilenberg, Mac Lane (1952)

Control theory
- Chen's iterated integrals (1950's)
- Chen-Fliess generating series (1980's)

Combinatorics (labelled counting)
- Riffle-shuffle product

Noncommutative differential algebra
- product of multivariate power series ≤ shuffle of series

Process algebra interpretation

	product rule
Hadamard	$a^{-1}(f \odot g) = a^{-1}f \odot a^{-1}g$ both processes read a
shuffle	$a^{-1}(f \shuffle g) = (a^{-1}f) \shuffle g + f \shuffle (a^{-1}g)$ exactly one process reads a
infiltration	$a^{-1}(f \uparrow g) = (a^{-1}f) \uparrow g + f \uparrow (a^{-1}g) + (a^{-1}f) \uparrow (a^{-1}g)$ either process reads a

Infiltration product in maths

ANNALS OF MATHEMATICS
Vol. 68, No. 1, July, 1958
Printed in Japan

FREE DIFFERENTIAL CALCULUS, IV. THE QUOTIENT GROUPS OF THE LOWER CENTRAL SERIES

BY K. T. CHEN, R. H. FOX[1] and R. C. LYNDON

(Received September 3, 1957)

Infiltration product in TCS

Math. Struct. in Comp. Science: page 1 of 29.
doi:10.1017/S0960129517000159

Newton series, coinductively: a comparative study of composition

HENNING BASOLD[†], HELLE HVID HANSEN[‡],
JEAN-ÉRIC PIN[§] and JAN RUTTEN[¶]

[†]*Radboud University Nijmegen, P.O. Box 9010, 6500GL Nijmegen, The Netherlands, and CWI Amsterdam, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands.*
Email: h.basold@cs.ru.nl
[‡]*Delft University of Technology, P.O. Box 5015, 2600 GA Delft, The Netherlands.*
Email: h.h.hansen@tudelft.nl
[§]*Université Paris Denis Diderot and CNRS, 75205 Paris Cedex 13, France.*
Email: Jean-Eric.Pin@liafa.univ-paris-diderot.fr
[¶]*CWI Amsterdam, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, and Radboud University Nijmegen, P.O. Box 9010, 6500GL Nijmegen, The Netherlands.*
Email: jjmmrutten@gmail.com

Received 4 May 2016; revised 3 April 2017

Objectives

Product	BAC	Automata	Decidable *zeroness problem*
Hadamard	✓		✓ [1, Theorem 4]
shuffle	✓		✓ [2, Theorem 1]
infiltration	✓		✓ [3, Theorem 2]

Can we understand these results in a *unified framework*?

[1] Benedikt, Duff, Sharad, Worrell: “Polynomial automata: Zeroness and applications” (LICS’17)
[2] C: “Weighted Basic Parallel Processes and Combinatorial Enumeration” (CONCUR’24)
[3] C: “The commutativity problem for effective varieties of formal series, and applications” (LICS’25)

Why BAC?

An *algebra* is a vector space equipped with a *bilinear product*.
Ubiquitous in maths

- Lie algebras, Rota-Baxter algebras, …

Associative commutative algebras

- Algebraic geometry

Freely generated associative commutative algebras (multivariate polynomials)

- Powerful algorithmic techniques based on *Hilbert's finite basis theorem* (**Noetherianity**)
- computer algebra

P-products

Consider polynomials of the form $P(x, \dot{x}, y, \dot{y}) \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-product* is a binary operation on series $* : \mathbb{Q}\langle\langle\Sigma\rangle\rangle \to \mathbb{Q}\langle\langle\Sigma\rangle\rangle \to \mathbb{Q}\langle\langle\Sigma\rangle\rangle$ s.t.

$$(f * g)(\varepsilon) = f(\varepsilon) * g(\varepsilon)$$
$$a^{-1}(f * g) = P(f, a^{-1}f, g, a^{-1}g), \qquad a \in \Sigma$$

Examples

P-product	product rule	P
Hadamard	$a^{-1}(f \odot g) = a^{-1}f \odot a^{-1}g$	$\dot{x}\,\dot{y}$
shuffle	$a^{-1}(f \shuffle g) = (a^{-1}f) \shuffle g + f \shuffle (a^{-1}g)$	$\dot{x}\,y + x\,\dot{y}$
infiltration	$a^{-1}(f \uparrow g) = (a^{-1}f) \uparrow g + f \uparrow (a^{-1}g) + (a^{-1}f) \uparrow (a^{-1}g)$	$\dot{x}\,y + x\,\dot{y} + \dot{x}\,\dot{y}$

Boreale and Gorla (CONCUR'21)

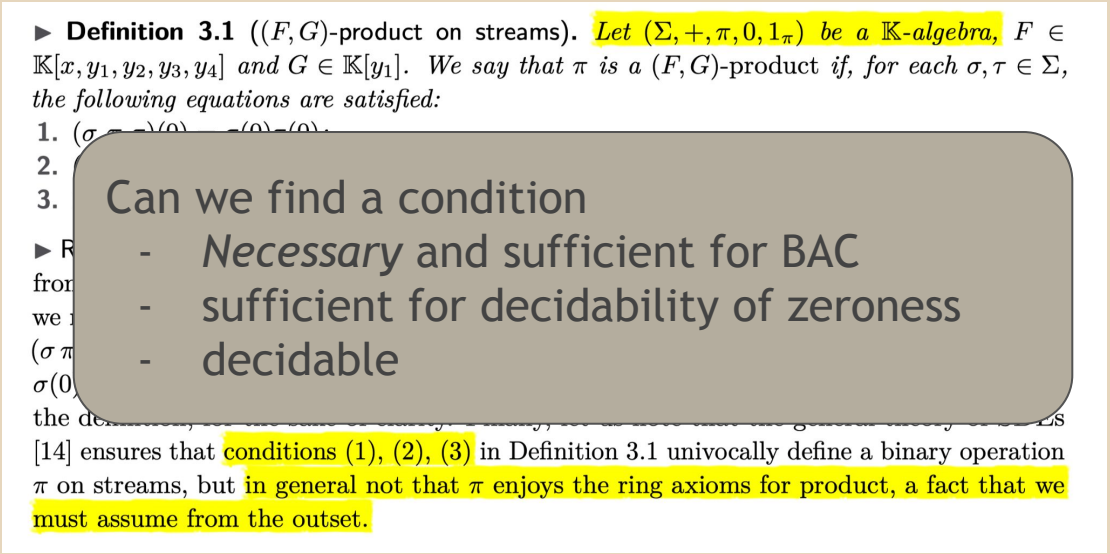
1. Unary alphabet case $\Sigma = \{ a \}$
2. Consider a certain *technical condition* (X)
3. The Hadamard, shuffle, infiltration product rules satisfy (X)
4. Product rules satisfying (X) give rise to
   a. BAC products
   b. automata with decidable zeroness

Product	BAC	Automata	Decidable *zeroness*
Hadamard	✓	Polynomial automata	✓
shuffle	✓	shuffle automata	✓
infiltration	✓	infiltration automata	✓

Limitations

- Unary alphabet Σ = { a }
  (not so limiting)

- Is condition (X)
  - necessary for BAC?
  - necessary for decidability of zeroness?
  - decidable?



▶ **Definition 3.1** ((F, G)-product on streams). *Let* $(\Sigma, +, \pi, 0, 1_\pi)$ *be a* $\mathbb{K}$*-algebra,* $F \in \mathbb{K}[x, y_1, y_2, y_3, y_4]$ *and* $G \in \mathbb{K}[y_1]$. *We say that* π *is a* (F, G)-product *if, for each* $\sigma, \tau \in \Sigma$, *the following equations are satisfied:*

1. $(\sigma$
2.
3.

▶ R
fro
we
$(\sigma\,\pi$
$\sigma(0$
the

[14] ensures that conditions (1), (2), (3) in Definition 3.1 univocally define a binary operation π on streams, but in general not that π enjoys the ring axioms for product, a fact that we must assume from the outset.

Can we find a condition
- *Necessary* and sufficient for BAC
- sufficient for decidability of zeroness
- decidable

BAC (non-)examples

product rule	P	bilin	assoc	comm
$a^{-1}(f * g) = 0$	0	✓	✓	✓
$a^{-1}(f * g) = f * g$	$x\ y$	✓	✓	✓
$a^{-1}(f \odot g) = a^{-1}f \odot a^{-1}g$	$\dot{x}\ \dot{y}$	✓	✓	✓
$a^{-1}(f \text{ Ш } g) = (a^{-1}f) \text{ Ш } g + f \text{ Ш } (a^{-1}g)$	$\dot{x}\ y + x\ \dot{y}$	✓	✓	✓
$a^{-1}(f \uparrow g) = (a^{-1}f) \uparrow g + f \uparrow (a^{-1}g) + (a^{-1}f) \uparrow (a^{-1}g)$	$\dot{x}\ y + x\ \dot{y} + \dot{x}\ \dot{y}$	✓	✓	✓
$a^{-1}(f * g) = f^2 * g^2$	$x^2\ y^2$	❌		
$a^{-1}(f * g) = (a^{-1}f) * g$	$\dot{x}\ y$			❌

Necessary BAC conditions

Commutativity:

$f * g = g * h$

$\Rightarrow a^{-1}(f * g) = a^{-1}(g * h)$

$\Rightarrow P(f, a^{-1}f, g, a^{-1}g) = P(g, a^{-1}g, f, a^{-1}f)$

Now extract the constant term:

$\Rightarrow (P(f, a^{-1}f, g, a^{-1}g))\ (\varepsilon) = (P(g, a^{-1}g, f, a^{-1}f))\ (\varepsilon)$

$\Rightarrow P(f(\varepsilon), f(a), g(\varepsilon), g(a)) = P(g(\varepsilon), g(a), f(\varepsilon), f(a))$

$\Rightarrow P(x, \dot{x}, y, \dot{y}) = P(y, \dot{y}, x, \dot{x})$

BAC characterisation

The following conditions are necessary for BAC:

Bilinearity	$P(x + y, \dot{x} + \dot{y}, z, \dot{z}) = P(x, \dot{x}, z, \dot{z}) + P(y, \dot{y}, z, \dot{z})$
Associativity	$P(x, \dot{x}, yz, P(y, \dot{y}, z, \dot{z})) = P(xy, P(x, \dot{x}, y, \dot{y}), z, \dot{z})$
Commutativity	$P(x, \dot{x}, y, \dot{y}) = P(y, \dot{y}, x, \dot{x})$

Theorem. The conditions above are also sufficient.
Proof. By coinduction.

BAC classification

A *P-product* is a binary operation on series "$*$" s.t.

$$(f * g)(\varepsilon) = f(\varepsilon) * g(\varepsilon) \quad \text{and} \quad a^{-1}(f * g) = P(f, a^{-1}f, g, a^{-1}g) \ (\forall\ a \in \Sigma)$$

A product rule P is *simple* if there are constants $\alpha, \beta, \gamma \in \mathbb{Q}$:

$$P(x, \dot{x}, y, \dot{y}) = \alpha \cdot xy + \beta \cdot (\dot{x}y + x\dot{y}) + \gamma \cdot \dot{x}\dot{y}$$
$$\text{s.t. } \alpha\,\gamma = \beta\,(\beta - 1)$$

Theorem. A P-product is BAC iff P is simple.
Proof. Apply the previous characterisation.

product	**α**	**β**	**γ**
Hadamard	0	0	1
shuffle	0	1	0
infiltration	0	1	1
shuffle-infiltration	0	1	γ

Cauchy product

Let $*$ be the concatenation (Cauchy) product of series.
E.g. $ab * c = abc$.

- bilinear ✓
- associative ✓
- *not commutative* ❌, for $|\Sigma| \geq 2$: $a * b \neq b * a$

Brzozowski's product rule
$a^{-1}(f * g) = (a^{-1}f) * g + f(\varepsilon) \cdot (a^{-1}g)$ *NOT a P-product!*

Cauchy product

Let $*$ be the concatenation (Cauchy) product of series.
E.g. $ab * c = abc$.

- bilinear ✓
- associative ✓
- *not commutative* ❌, for $|\Sigma| \geq 2$: $a * b \neq b * a$

Brzozowski's product rule

$a^{-1}(f * g) = (a^{-1} f) * g + f(\varepsilon) \cdot (a^{-1} g)$ *NOT a P-product!*

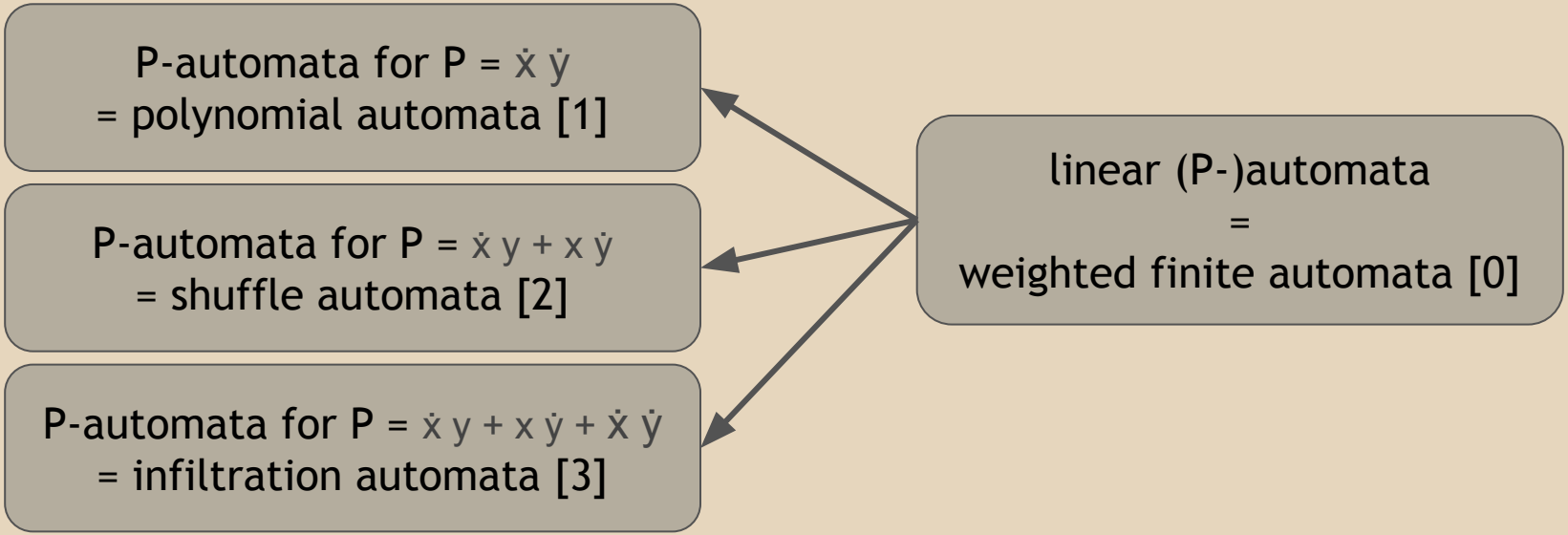
P-automata: Syntax

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-automaton* is a tuple

$$A = (\Sigma, X, F, \Delta), \text{ where}$$

- Σ is a finite *input alphabet*
- $X = \{x_1, \ldots, x_n\}$ is a finite set of commuting *variables*
- $F : X \to \mathbb{Q}$ is the *output function*
- $\Delta : \Sigma \to X \to \mathbb{Q}[X]$ is the *transition function*

P-automata



[0] Schützenberger: “On the definition of a family of automata” (IC’61)
[1] Benedikt, Duff, Sharad, Worrell: “Polynomial automata: Zeroness and applications” (LICS’17)
[2] C: “Weighted Basic Parallel Processes and Combinatorial Enumeration” (CONCUR’24)
[3] C: “The commutativity problem for effective varieties of formal series, and applications” (LICS’25)

P-automata: example

Consider the *P-automaton* A = (Σ = { a }, X = { x }, F, Δ), where

- F x = 2
- Δ_a x = x^2
- P(x, ẋ, y, ẏ) = ẋ ẏ: $\Delta_a(pq) = (\Delta_a p)\ (\Delta_a q)$

$$
\begin{array}{lccccccc}
 & x & \xrightarrow{\Delta_a} & x^2 & \xrightarrow{\Delta_a} & x^4 & \xrightarrow{\Delta_a} & x^8 \xrightarrow{\Delta_a} \dots\ x\hat{}(2^n) \\
 & \downarrow F & & \downarrow F & & \downarrow F & & \downarrow F \qquad\qquad \downarrow F \\
A[[x]] = & 2 & & 4 & & 16 & & 256 \qquad\qquad 2\hat{}(2^n)
\end{array}
$$

P-automata: example

Consider the *P-automaton* $A = (\Sigma = \{ a \}, X = \{ x \}, F, \Delta)$, where

- $F\ x = 2$
- $\Delta_a\ x = x^2$
- $P(x, \dot{x}, y, \dot{y}) = \dot{x}y + x\dot{y}$: $\Delta_a(pq) = (\Delta_a p)\ q + p\ (\Delta_a q)$

$$
\begin{array}{ccccccccccc}
x & \text{—}\Delta\text{—>} & x^2 & \text{—}\Delta\text{—>} & 2x^3 & \text{—}\Delta\text{—>} & 6x^4 & \text{—}\Delta\text{—>} & \dots & \text{—}\Delta\text{—>} & n!\ x^{n+1} \\
\downarrow F & & \downarrow F & & \downarrow F & & \downarrow F & & & & \downarrow F \\
A[[x]] = 1 & & 1 & & 2 & & 6 & & \dots & & n!
\end{array}
$$

P-automata: Semantics

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
A *P-automaton* $A = (\Sigma, X, F, \Delta)$ defines a *semantic function*

$$A[\![\_]\!] : \mathbb{Q}[X] \to \mathbb{Q}\langle\langle\Sigma\rangle\rangle$$

mapping a *configuration* $p \in \mathbb{Q}[X]$ to a series $A[\![p]\!] \in \mathbb{Q}\langle\langle\Sigma\rangle\rangle$.

P-functions

Consider a product rule $P \in \mathbb{Q}[x, \dot{x}, y, \dot{y}]$.
$\Delta : \mathbb{Q}[X] \to \mathbb{Q}[X]$ is a *P-function* if

- It is linear
  $\Delta (p + q) = \Delta p + \Delta q$
  $\Delta (c\ p) = c\ \Delta p,\ c \in \mathbb{Q}$

- It satisfies the product rule
  $\Delta (p\ q) = P(p, \Delta p, q, \Delta q)$

Lemma. Let P be a BAC product rule.
Every function $\Delta : X \to \mathbb{Q}[X]$ extends (uniquely) to a P-function $\mathbb{Q}[X] \to \mathbb{Q}[X]$.

P-automata: Semantics

Fix A = (Σ, X, F, Δ). We define the semantic function A⟦_⟧ : 𝕄[X] → 𝕄⟪Σ⟫.

$\Delta : \Sigma \to X \to \mathbb{M}[X]$

$\Delta : \Sigma \to \mathbb{M}[X] \to \mathbb{M}[X]$

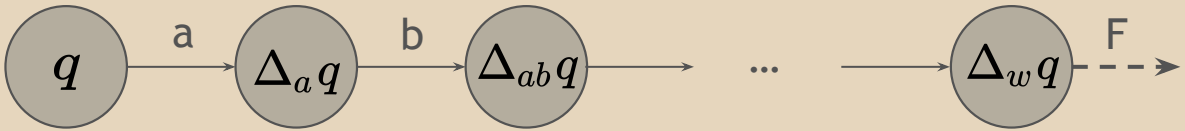
extend to a unique *P-function*

$\Delta_a(pq) = P(p, \Delta_a p, q, \Delta_a q)$

$\Delta : \Sigma^* \to \mathbb{M}[X] \to \mathbb{M}[X]$

extend *homomorphically*

$$\Delta_\varepsilon(p) := p$$
$$\Delta_{a \cdot w}(p) := \Delta_w(\Delta_a p)$$



output

$$A[\![q]\!](w) := (\Delta_w q)(F)$$

(B) = (C)

Coincidence Lemma.
A series is P-finite iff it is recognised by P-automaton.

Zeroness problem (D)

INPUT: A P-automaton A and an initial configuration p
OUTPUT: yes iff $A[\![p]\!] = 0$

- Fundamental algorithmic problem
  - “Word problem” in algorithmic group theory ($g = 1$?)
  - cf. recognise the trivial knot from a complicated presentation
- Subsumes equality:

  $A[\![p]\!] = A[\![q]\!] \Leftrightarrow A[\![p]\!] - A[\![q]\!] = A[\![p - q]\!] = 0$

Zeroness problem (D)

INPUT: A P-automaton A and an initial configuration p
OUTPUT: yes iff $A[\![p]\!] = 0$

Theorem.
Let P be a simple product rule.
The zeroness problem is decidable for P-finite series
(= series recognised by P-automata).

The big picture $\Sigma = \{ a, b \}$