Efficient Coalgebraic Partition Refinement

From Categorical Constructions to Tool Implementation

Stefan Milius

Joint work with:

Fabian Birkmann, Hans-Peter Deifel, Ulrich Dorsch, Lutz Schröder, Thorsten Wißmann

Friedrich-Alexander-Universität Erlangen-Nürnberg

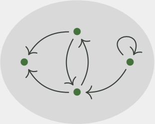
CATMI 2023, Bergen
June 28, 2023

Efficient Coalgebraic Partition Refinement

Efficient Coalgebraic Partition Refinement

1. Coalgebras:

State based systems

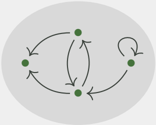


Labels, Non-Determinism,
Probabilities, Automata,
... and their combinations!

Efficient Coalgebraic Partition Refinement

1. Coalgebras:

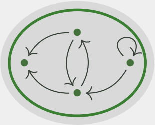
State based systems



Labels, Non-Determinism,
Probabilities, Automata,
... and their combinations!

2. Partition Refinement:

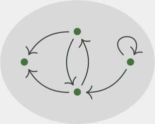
Successively distinguish different behaviour



Efficient Coalgebraic Partition Refinement

1. Coalgebras:

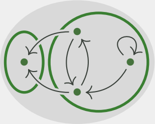
State based systems



Labels, Non-Determinism,
Probabilities, Automata,
... and their combinations!

2. Partition Refinement:

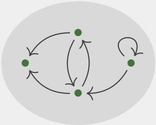
Successively distinguish different behaviour



Efficient Coalgebraic Partition Refinement

1. Coalgebras:

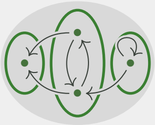
State based systems



Labels, Non-Determinism,
Probabilities, Automata,
... and their combinations!

2. Partition Refinement:

Successively distinguish different behaviour



Efficient Coalgebraic Partition Refinement

3. Efficiency:

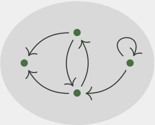
(a) Incrementally compute partitions

(b) Complexity Analysis: $\mathcal{O}(\underbrace{m}_{\text{edges}} \cdot \log \underbrace{n}_{\text{states}})$

(c) Tool

1. Coalgebras:

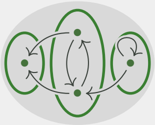
State based systems



Labels, Non-Determinism,
Probabilities, Automata,
... and their combinations!

2. Partition Refinement:

Successively distinguish different behaviour



Share Common Structure & Ideas

Similar Run-Time

Variations in Details

Share Common Structure & Ideas

Similar Run-Time

Variations in Details

Deterministic Finite Automata

$n \cdot \log n$ Hopcroft '71

$|A| \cdot n \cdot \log n$ Gries '73 Knuutila '01

Deterministic
Finite Automata

$n \cdot \log n$
Hopcroft '71

$|A| \cdot n \cdot \log n$
Gries '73
Knuutila '01

Transition Systems

$m \cdot \log n$
Paige, Tarjan '87

Share Common
Structure & Ideas

Similar
Run-Time

Variations in
Details

Share Common Structure & Ideas

Similar Run-Time

Variations in Details

Deterministic Finite Automata

$n \cdot \log n$ Hopcroft '71

$|A| \cdot n \cdot \log n$ Gries '73 Knuutila '01

Transition Systems

$m \cdot \log n$ Paige, Tarjan '87

Segala Systems

$m \cdot n \cdot (\log m + \log n)$ Baier, Engelen, Majster-Cederbaum '00

Share Common Structure & Ideas

Similar Run-Time

Variations in Details

Deterministic Finite Automata

$n \cdot \log n$
Hopcroft '71

$|A| \cdot n \cdot \log n$
Gries '73
Knuutila '01

Transition Systems

$m \cdot \log n$
Paige, Tarjan '87

Tree Automata

$m \cdot n$
Högberg, Maletti, May '07

Segala Systems

$m \cdot n \cdot (\log m + \log n)$
Baier, Engelen, Majster-Cederbaum '00

Share Common Structure & Ideas

Similar Run-Time

Variations in Details

Deterministic Finite Automata

$n \cdot \log n$ Hopcroft '71

$|A| \cdot n \cdot \log n$ Gries '73, Knuutila '01

Transition Systems

$m \cdot \log n$

Paige, Tarjan '87

Tree Automata

$m \cdot n$

Högberg, Maletti, May '07

Segala Systems

$m \cdot n \cdot (\log m + \log n)$

Baier, Engelen, Majster-Cederbaum '00

Labelled Transition Systems

$m \cdot \log n$

Valmari '09

Share Common Structure & Ideas

Similar Run-Time

Variations in Details

Deterministic Finite Automata

$n \cdot \log n$ Hopcroft '71

$|A| \cdot n \cdot \log n$ Gries '73 Knuutila '01

Transition Systems

$m \cdot \log n$

Paige, Tarjan '87

Tree Automata

$m \cdot n$

Högberg, Maletti, May '07

Weighted Systems ("Markov Chain Lumping")

$m \cdot \log n$

Valmari, Franceschinis '10

Segala Systems

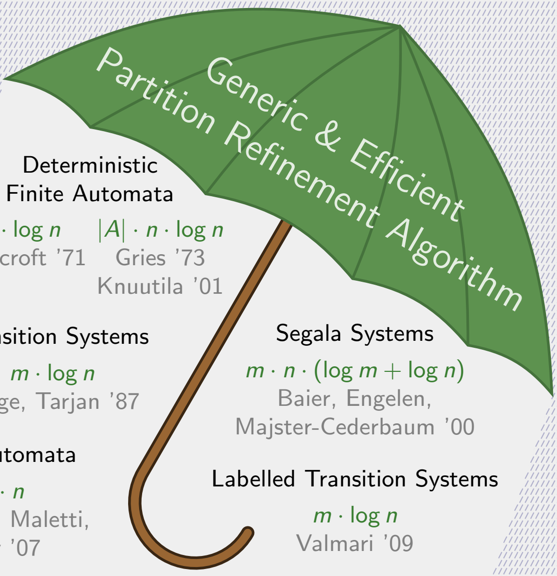
$m \cdot n \cdot (\log m + \log n)$

Baier, Engelen, Majster-Cederbaum '00

Labelled Transition Systems

$m \cdot \log n$

Valmari '09



Generic & Efficient
Partition Refinement Algorithm

Deterministic
Finite Automata

$n \cdot \log n$
Hopcroft '71

$|A| \cdot n \cdot \log n$
Gries '73
Knuutila '01

Transition Systems

$m \cdot \log n$
Paige, Tarjan '87

Segala Systems

$m \cdot n \cdot (\log m + \log n)$
Baier, Engelen,
Majster-Cederbaum '00

Tree Automata

$m \cdot n$
Högberg, Maletti,
May '07

Labelled Transition Systems

$m \cdot \log n$
Valmari '09

Weighted Systems ("Markov Chain Lumping")

$m \cdot \log n$
Valmari, Franceschinis '10

Ingredient 1: Factorizations

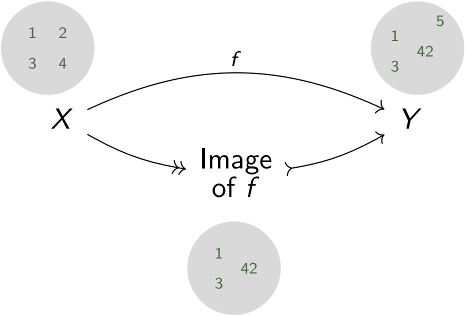
Equivalence Relations	$\cong$	Quotients $\cong$ Partitions
Kernels	$\cong$	Regular Epimorphisms

Ingredient 1: Factorizations

Equivalence Relations ≅ Quotients ≅ Partitions

Kernels ≅ Regular Epimorphisms

Category with (Regular Epi,Mono)-Factorizations

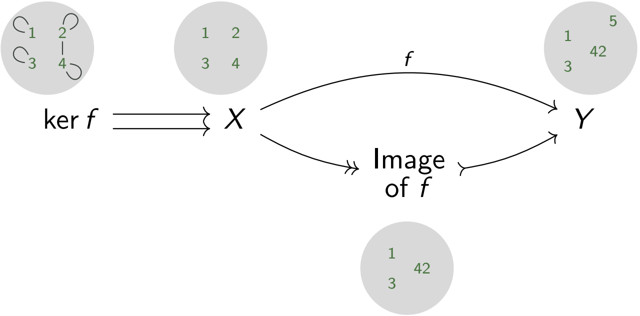


Ingredient 1: Factorizations

Equivalence Relations ≅ Quotients ≅ Partitions

Kernels ≅ Regular Epimorphisms

Category with (Regular Epi,Mono)-Factorizations



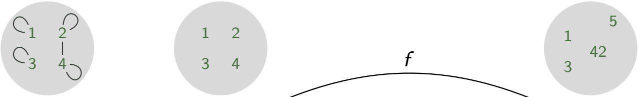
$\ker f = \{(x_1, x_2) \in X \times X \mid f(x_1) = f(x_2)\}$

Ingredient 1: Factorizations

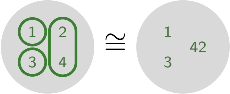
Equivalence Relations ≅ Quotients ≅ Partitions

Kernels ≅ Regular Epimorphisms

Category with (Regular Epi,Mono)-Factorizations

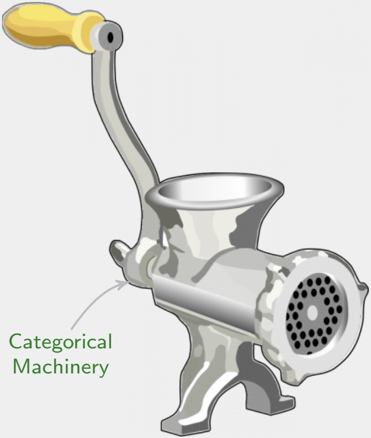


$$\begin{tikzcd}
\ker f \arrow[r, shift left] \arrow[r, shift right] & X \arrow[rr, bend left, "f"] \arrow[dr, two heads] & & Y \\
& & X/_{\ker f} \cong \text{Image of } f \arrow[ur, tail] &
\end{tikzcd}$$



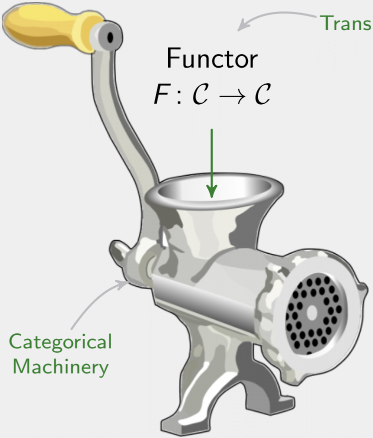
$$\ker f = \{(x_1, x_2) \in X \times X \mid f(x_1) = f(x_2)\}$$

Ingredient 2: Coalgebra – Generic state based systems



Categorical Machinery

Ingredient 2: Coalgebra – Generic state based systems



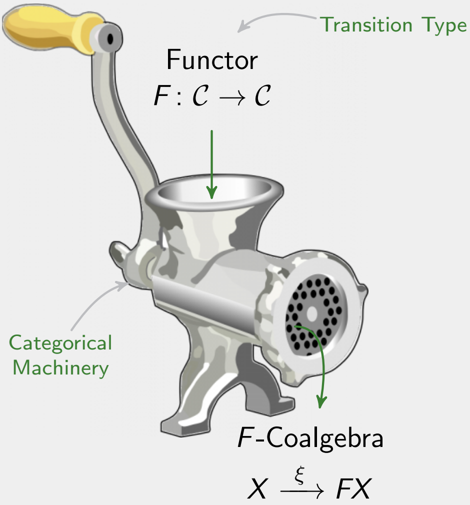
Transition Type

Functor

$F : \mathcal{C} \to \mathcal{C}$

Categorical Machinery

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor

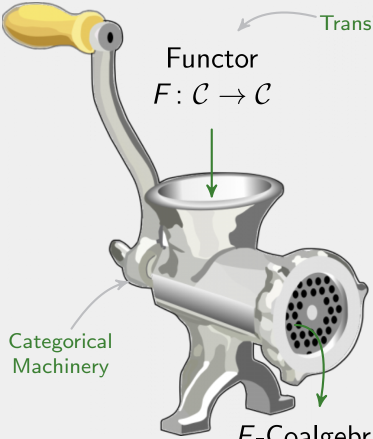
$F\colon \mathcal{C} \to \mathcal{C}$

Categorical Machinery

F-Coalgebra

$X \xrightarrow{\xi} FX$

Ingredient 2: Coalgebra – Generic state based systems



Powerset $\mathcal{P}$

Transition Type

Functor

$F\colon \mathcal{C} \to \mathcal{C}$

Categorical Machinery

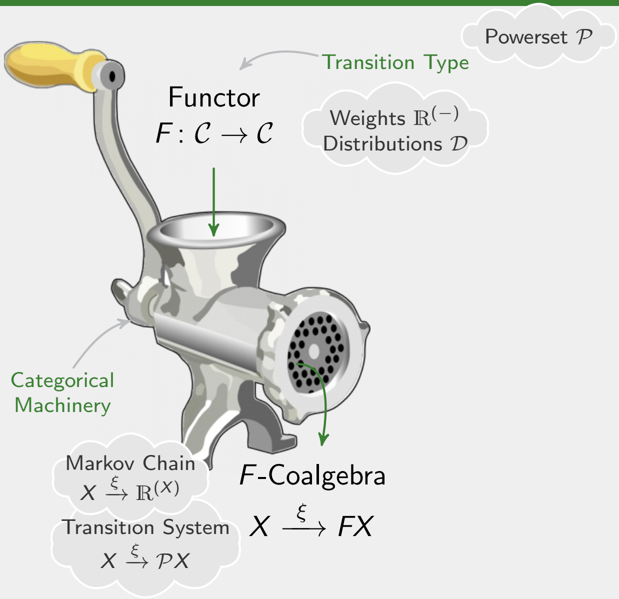
F-Coalgebra

$X \xrightarrow{\xi} FX$

Transition System

$X \xrightarrow{\xi} \mathcal{P}X$

Ingredient 2: Coalgebra – Generic state based systems



Powerset $\mathcal{P}$

Transition Type

Functor

$F\colon \mathcal{C} \to \mathcal{C}$

Weights $\mathbb{R}^{(-)}$

Distributions $\mathcal{D}$

Categorical Machinery

Markov Chain

$X \xrightarrow{\xi} \mathbb{R}^{(X)}$

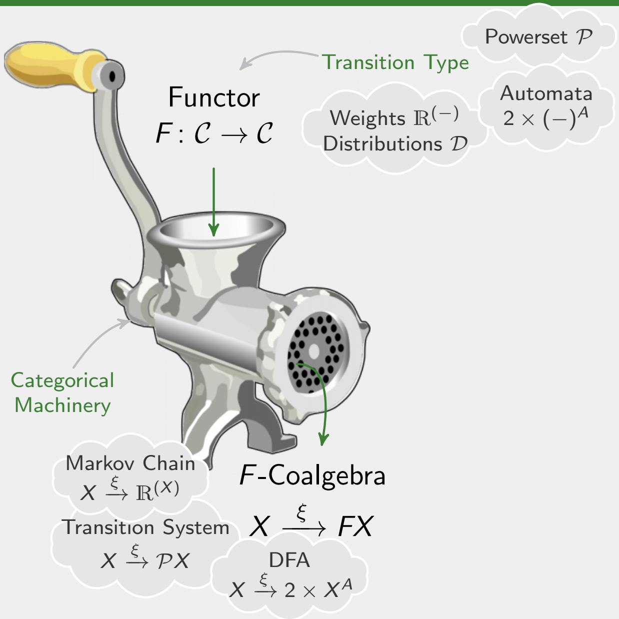
F-Coalgebra

$X \xrightarrow{\xi} FX$

Transition System

$X \xrightarrow{\xi} \mathcal{P}X$

Ingredient 2: Coalgebra – Generic state based systems



Powerset $\mathcal{P}$

Transition Type

Functor

$F : \mathcal{C} \to \mathcal{C}$

Weights $\mathbb{R}^{(-)}$

Distributions $\mathcal{D}$

Automata

$2 \times (-)^A$

Categorical Machinery

F-Coalgebra

$$X \xrightarrow{\xi} FX$$

Markov Chain

$X \xrightarrow{\xi} \mathbb{R}^{(X)}$

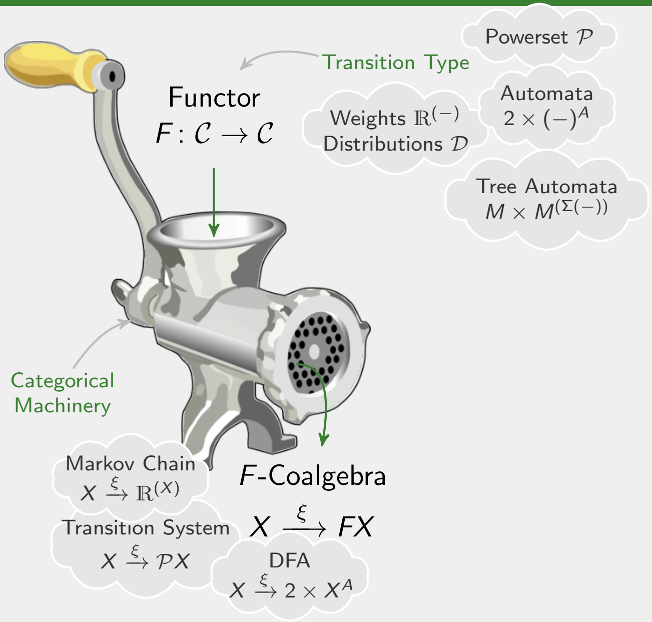
Transition System

$X \xrightarrow{\xi} \mathcal{P}X$

DFA

$X \xrightarrow{\xi} 2 \times X^A$

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor

$F\colon \mathcal{C} \to \mathcal{C}$

Powerset $\mathcal{P}$

Weights $\mathbb{R}^{(-)}$
Distributions $\mathcal{D}$

Automata
$2 \times (-)^A$

Tree Automata
$M \times M^{(\Sigma(-))}$

Categorical Machinery

F-Coalgebra

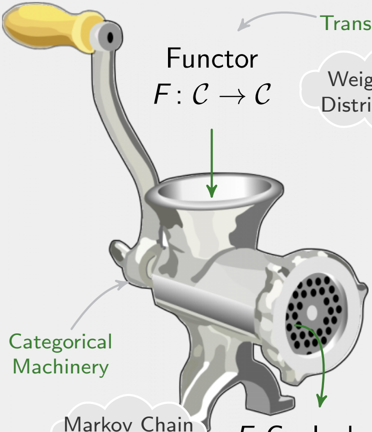
$$X \xrightarrow{\xi} FX$$

Markov Chain
$X \xrightarrow{\xi} \mathbb{R}^{(X)}$

Transition System
$X \xrightarrow{\xi} \mathcal{P}X$

DFA
$X \xrightarrow{\xi} 2 \times X^A$

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor
$F\colon \mathcal{C} \to \mathcal{C}$

Powerset $\mathcal{P}$

Simple Segala
$\mathcal{P}_{\mathrm{f}}(A \times \mathcal{D}(-))$

Weights $\mathbb{R}^{(-)}$
Distributions $\mathcal{D}$

Automata
$2 \times (-)^A$

Tree Automata
$M \times M^{(\Sigma(-))}$

Categorical Machinery

F-Coalgebra

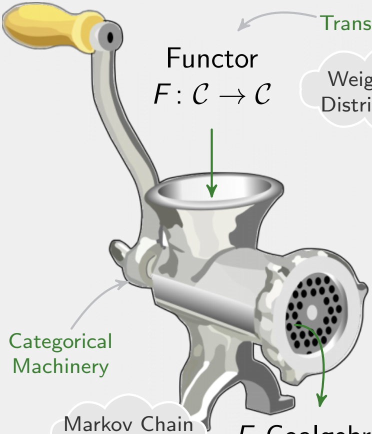
$$X \xrightarrow{\xi} FX$$

Markov Chain
$X \xrightarrow{\xi} \mathbb{R}^{(X)}$

Transition System
$X \xrightarrow{\xi} \mathcal{P}X$

DFA
$X \xrightarrow{\xi} 2 \times X^A$

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor
$F\colon \mathcal{C} \to \mathcal{C}$

Powerset $\mathcal{P}$

Weights $\mathbb{R}^{(-)}$
Distributions $\mathcal{D}$

Automata
$2 \times (-)^A$

Tree Automata
$M \times M^{(\Sigma(-))}$

Simple Segala
$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}(-))$

General Segala
$\mathcal{P}_\mathrm{f}\mathcal{D}(A \times -))$

Categorical Machinery

F-Coalgebra

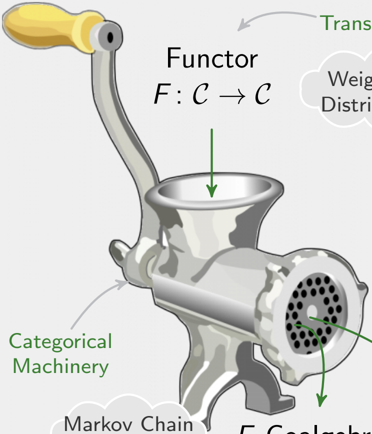
$$X \xrightarrow{\xi} FX$$

Markov Chain
$X \xrightarrow{\xi} \mathbb{R}^{(X)}$

Transition System
$X \xrightarrow{\xi} \mathcal{P}X$

DFA
$X \xrightarrow{\xi} 2 \times X^A$

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor

$F\colon \mathcal{C} \to \mathcal{C}$

Powerset $\mathcal{P}$

Simple Segala $\mathcal{P}_{\mathrm{f}}(A \times \mathcal{D}(-))$

Weights $\mathbb{R}^{(-)}$
Distributions $\mathcal{D}$

Automata $2 \times (-)^A$

General Segala $\mathcal{P}_{\mathrm{f}}\mathcal{D}(A \times -))$

Tree Automata $M \times M^{(\Sigma(-))}$

Categorical Machinery

Markov Chain $X \xrightarrow{\xi} \mathbb{R}^{(X)}$

Transition System $X \xrightarrow{\xi} \mathcal{P}X$

F-Coalgebra

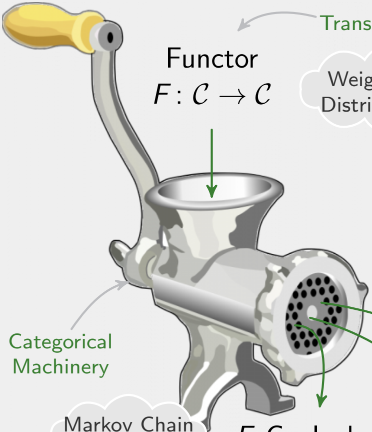
$X \xrightarrow{\xi} FX$

DFA $X \xrightarrow{\xi} 2 \times X^A$

Homomorphism

$(X, \xi) \xrightarrow{h} (X', \xi')$

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor

$F\colon \mathcal{C} \to \mathcal{C}$

Powerset $\mathcal{P}$

Simple Segala $\mathcal{P}_{\mathrm{f}}(A \times \mathcal{D}(-))$

Weights $\mathbb{R}^{(-)}$
Distributions $\mathcal{D}$

Automata $2 \times (-)^A$

General Segala $\mathcal{P}_{\mathrm{f}}\mathcal{D}(A \times -))$

Tree Automata $M \times M^{(\Sigma(-))}$

Categorical Machinery

Markov Chain $X \xrightarrow{\xi} \mathbb{R}^{(X)}$

Transition System $X \xrightarrow{\xi} \mathcal{P}X$

DFA $X \xrightarrow{\xi} 2 \times X^A$

F-Coalgebra

$X \xrightarrow{\xi} FX$

Homomorphism

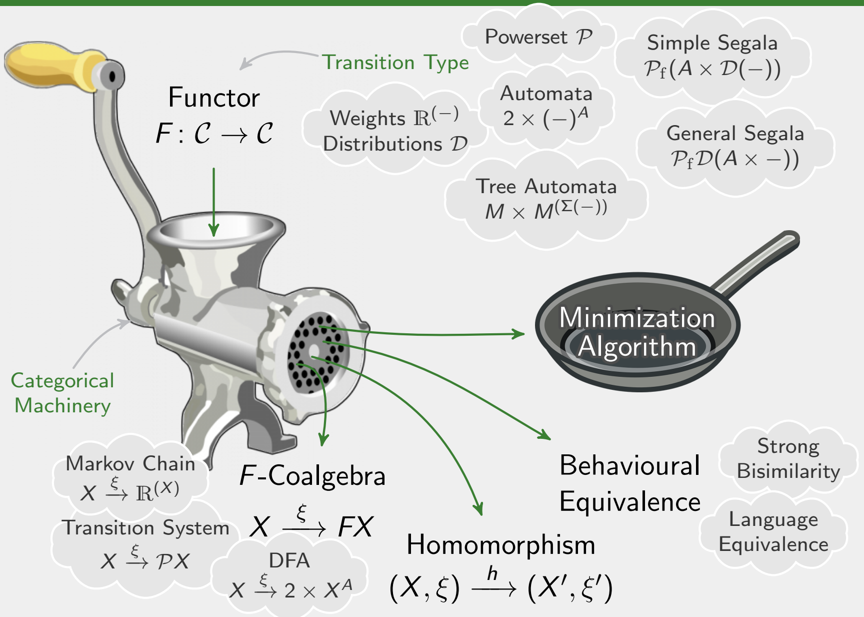
$(X,\xi) \xrightarrow{h} (X',\xi')$

Behavioural Equivalence

Strong Bisimilarity

Language Equivalence

Ingredient 2: Coalgebra – Generic state based systems



Transition Type

Functor
$F\colon \mathcal{C} \to \mathcal{C}$

Weights $\mathbb{R}^{(-)}$
Distributions $\mathcal{D}$

Powerset $\mathcal{P}$

Automata
$2 \times (-)^A$

Tree Automata
$M \times M^{(\Sigma(-))}$

Simple Segala
$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}(-))$

General Segala
$\mathcal{P}_\mathrm{f}\mathcal{D}(A \times -))$

Minimization Algorithm

Categorical Machinery

Markov Chain
$X \xrightarrow{\xi} \mathbb{R}^{(X)}$

Transition System
$X \xrightarrow{\xi} \mathcal{P}X$

DFA
$X \xrightarrow{\xi} 2 \times X^A$

F-Coalgebra

$X \xrightarrow{\xi} FX$

Homomorphism

$(X, \xi) \xrightarrow{h} (X', \xi')$

Behavioural Equivalence

Strong Bisimilarity

Language Equivalence

The Coalgebraic Task

For a functor $F\colon \mathcal{C} \to \mathcal{C}$

$$\begin{tikzcd}
\text{Given a coalgebra} & X \arrow[r, "\xi"] \arrow[d, two heads, "h"'] & FX \arrow[d, two heads, "Fh"] \\
\text{find the \textbf{simple} quotient} & X' \arrow[r, "\xi'"] & FX'
\end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

The Coalgebraic Task

For a functor $F\colon \mathcal{C} \to \mathcal{C}$

$$\begin{tikzcd}
\text{Given a coalgebra} & X \arrow[r, "\xi"] \arrow[d, two heads, "h"'] & FX \arrow[d, two heads, "Fh"] \\
\text{find the \textbf{simple} quotient} & X' \arrow[r, "\xi'"] & FX'
\end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

Instance

For $2 \times (-)^A$: $\begin{tikzcd} \mathsf{Set} \arrow[loop right] \end{tikzcd}$

Automata minimization

The Coalgebraic Task

For a functor $F\colon \mathcal{C} \to \mathcal{C}$

$$\begin{tikzcd}
\text{Given a coalgebra} & X \arrow[r, "\xi"] \arrow[d, two heads, "h"'] & FX \arrow[d, two heads, "Fh"] \\
\text{find the } \textbf{simple} \text{ quotient} & X' \arrow[r, "\xi'"] & FX'
\end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

Instance

Instance

For $2 \times (-)^A \colon \mathsf{Set} \circlearrowleft$

Automata minimization

For $\mathcal{P}_\mathrm{f} \colon \mathsf{Set} \circlearrowleft$

Bisimilarity minimization

The Coalgebraic Task

For a functor $F\colon \mathcal{C} \to \mathcal{C}$

Given a coalgebra / find the **simple** quotient

$$\begin{tikzcd}
X \arrow[r, "\xi"] \arrow[d, two heads, "h"'] & FX \arrow[d, two heads, "Fh"] \\
X' \arrow[r, "\xi'"] & FX'
\end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

Instance

Instance

Instance

For $2 \times (-)^A\colon \mathsf{Set} \circlearrowleft$

Automata minimization

For $\mathcal{P}_\mathrm{f}\colon \mathsf{Set} \circlearrowleft$

Bisimilarity minimization

For $\mathbb{R}^{(-)}\colon \mathsf{Set} \circlearrowleft$

Markov chain lumping

The Coalgebraic Task

For a functor $F\colon \mathcal{C} \to \mathcal{C}$

$$\begin{tikzcd}
\text{Given a coalgebra} & X \arrow[r, "\xi"] \arrow[d, two heads, "h"'] & FX \arrow[d, two heads, "Fh"] \\
\text{find the \textbf{simple} quotient} & X' \arrow[r, "\xi'"] & FX'
\end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

Instance

Instance

Instance

Instance

For $2 \times (-)^A \colon \mathsf{Set} \circlearrowleft$

Automata minimization

For $\mathcal{P}_{\mathrm{f}} \colon \mathsf{Set} \circlearrowleft$

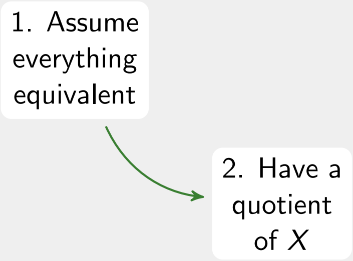
Bisimilarity minimization

For $\mathbb{R}^{(-)} \colon \mathsf{Set} \circlearrowleft$

Markov chain lumping

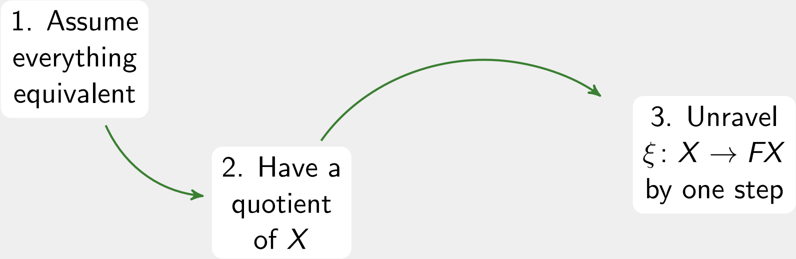
...

1. Assume everything equivalent



1. Assume everything equivalent

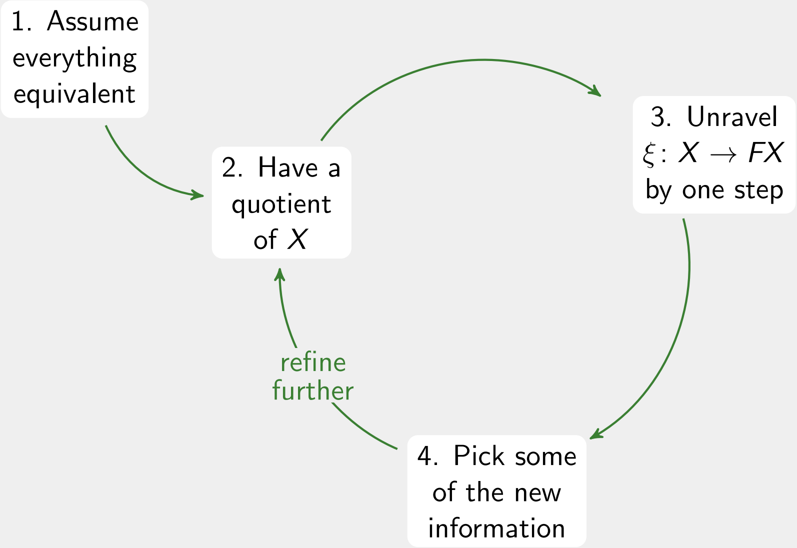
2. Have a quotient of X



1. Assume everything equivalent

2. Have a quotient of X

3. Unravel $\xi\colon X \to FX$ by one step

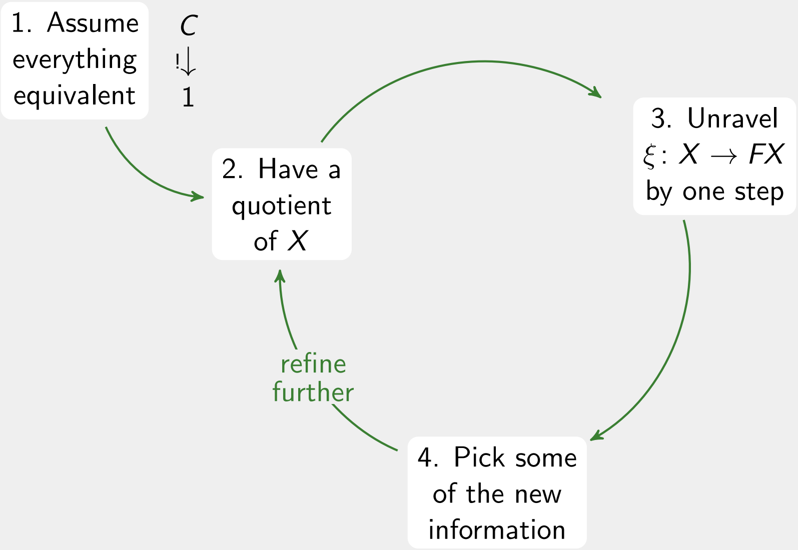


1. Assume everything equivalent

2. Have a quotient of X

3. Unravel $\xi\colon X \to FX$ by one step

4. Pick some of the new information



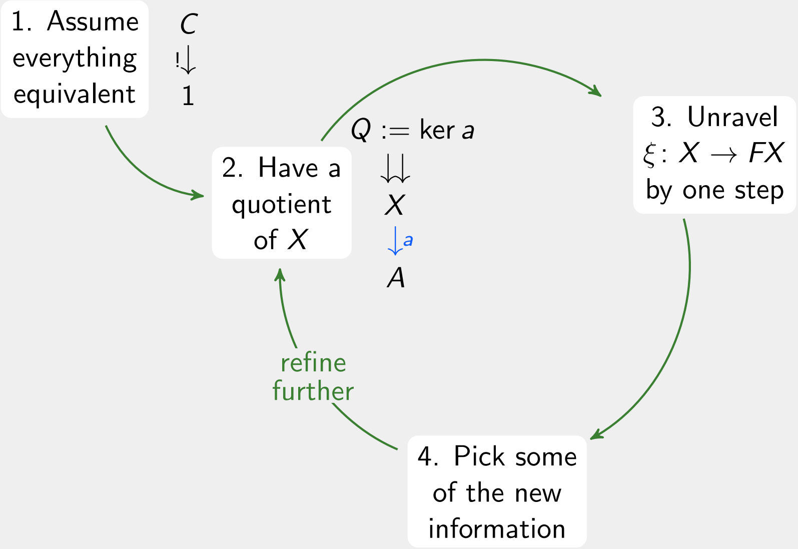
1. Assume everything equivalent

$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

3. Unravel $\xi\colon X \to FX$ by one step

4. Pick some of the new information



1. Assume everything equivalent

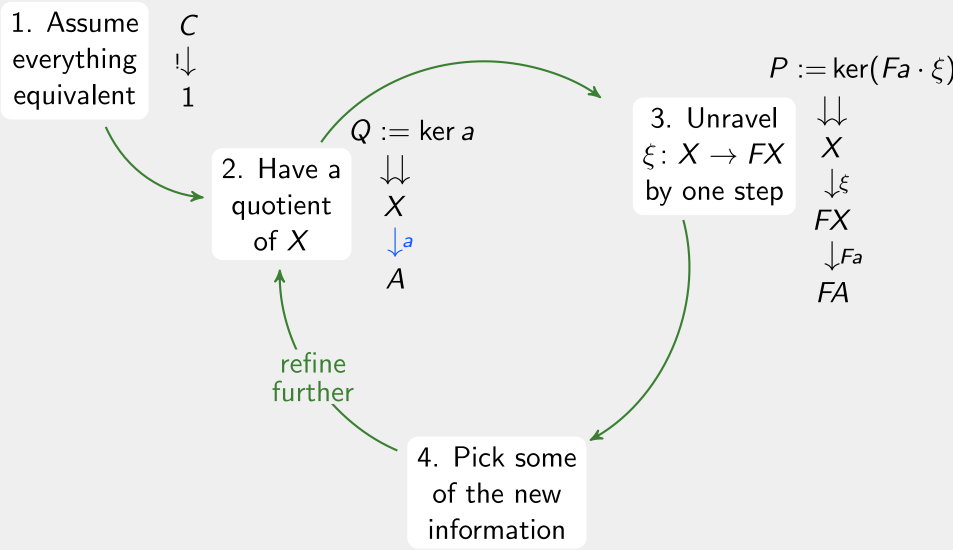
$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

$$\begin{tikzcd} Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "a"] \\ A \end{tikzcd}$$

3. Unravel $\xi\colon X \to FX$ by one step

4. Pick some of the new information



1. Assume everything equivalent

$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

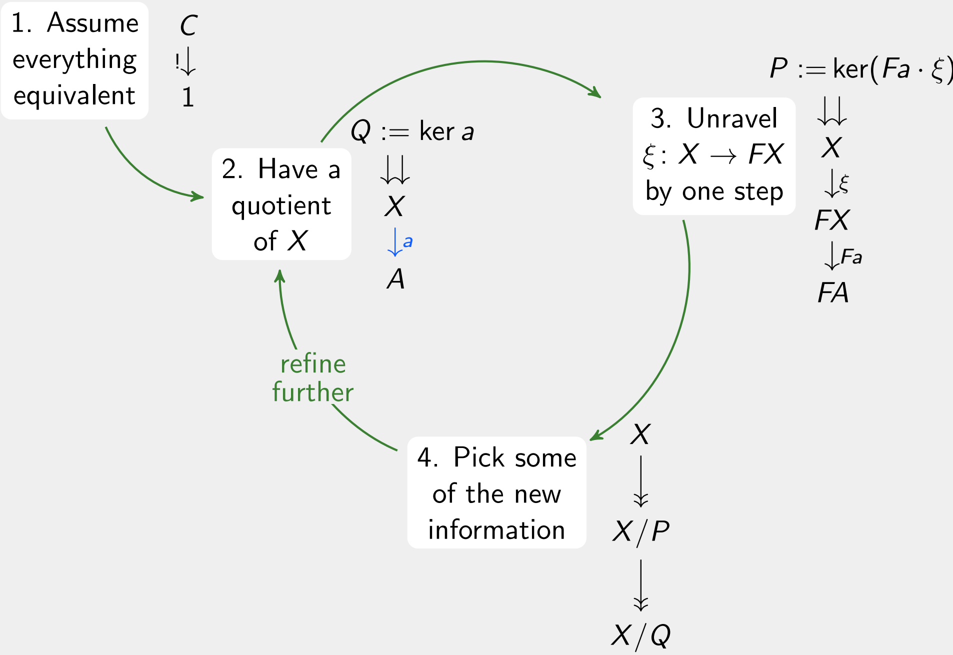
$$\begin{tikzcd} Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "a"] \\ A \end{tikzcd}$$

3. Unravel $\xi\colon X \to FX$ by one step

$$\begin{tikzcd} P := \ker(Fa \cdot \xi) \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "\xi"] \\ FX \arrow[d, "Fa"] \\ FA \end{tikzcd}$$

4. Pick some of the new information

refine further



1. Assume everything equivalent

$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

$$\begin{tikzcd} Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "a"] \\ A \end{tikzcd}$$

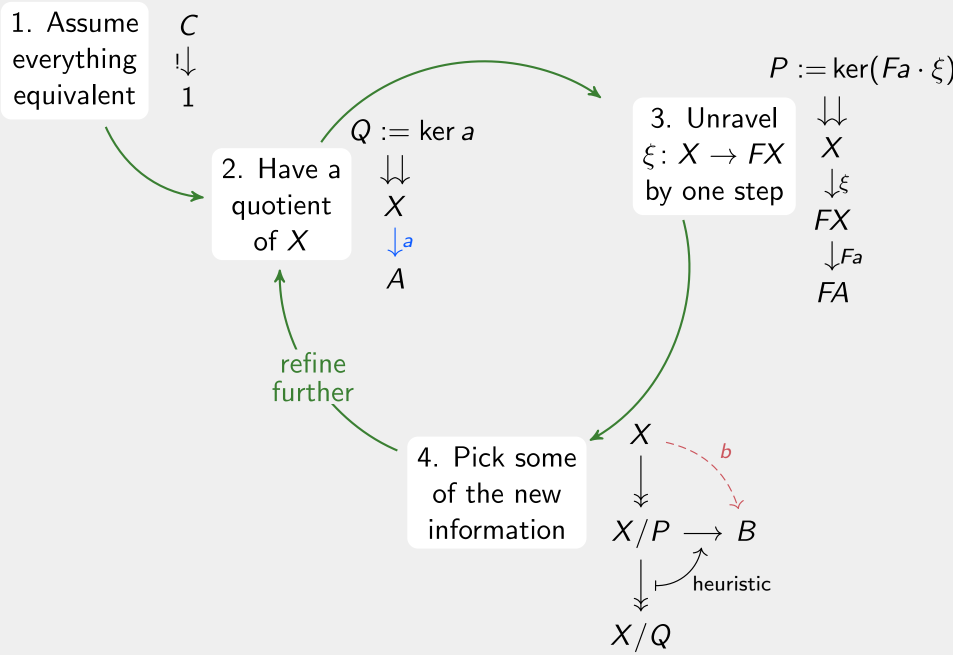
3. Unravel $\xi\colon X \to FX$ by one step

$$\begin{tikzcd} P := \ker(Fa \cdot \xi) \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "\xi"] \\ FX \arrow[d, "Fa"] \\ FA \end{tikzcd}$$

4. Pick some of the new information

$$\begin{tikzcd} X \arrow[d, two heads] \\ X/P \arrow[d, two heads] \\ X/Q \end{tikzcd}$$

refine further



1. Assume everything equivalent

$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

$$\begin{tikzcd} Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "a"] \\ A \end{tikzcd}$$

3. Unravel $\xi\colon X \to FX$ by one step

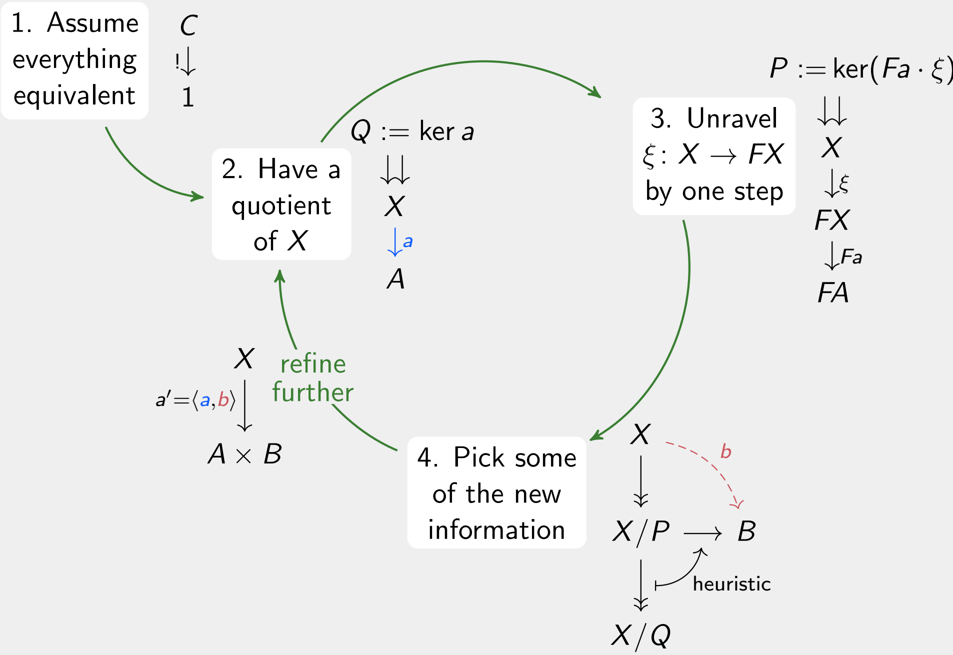
$$\begin{tikzcd} P := \ker(Fa \cdot \xi) \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "\xi"] \\ FX \arrow[d, "Fa"] \\ FA \end{tikzcd}$$

4. Pick some of the new information

$$\begin{tikzcd} X \arrow[d, two heads] \arrow[dr, dashed, bend left, "b"] & \\ X/P \arrow[r] \arrow[d, two heads] & B \\ X/Q & \end{tikzcd}$$

heuristic

refine further



1. Assume everything equivalent

$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

$$\begin{tikzcd} Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "a"] \\ A \end{tikzcd}$$

3. Unravel $\xi\colon X \to FX$ by one step

$$\begin{tikzcd} P := \ker(Fa \cdot \xi) \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "\xi"] \\ FX \arrow[d, "Fa"] \\ FA \end{tikzcd}$$

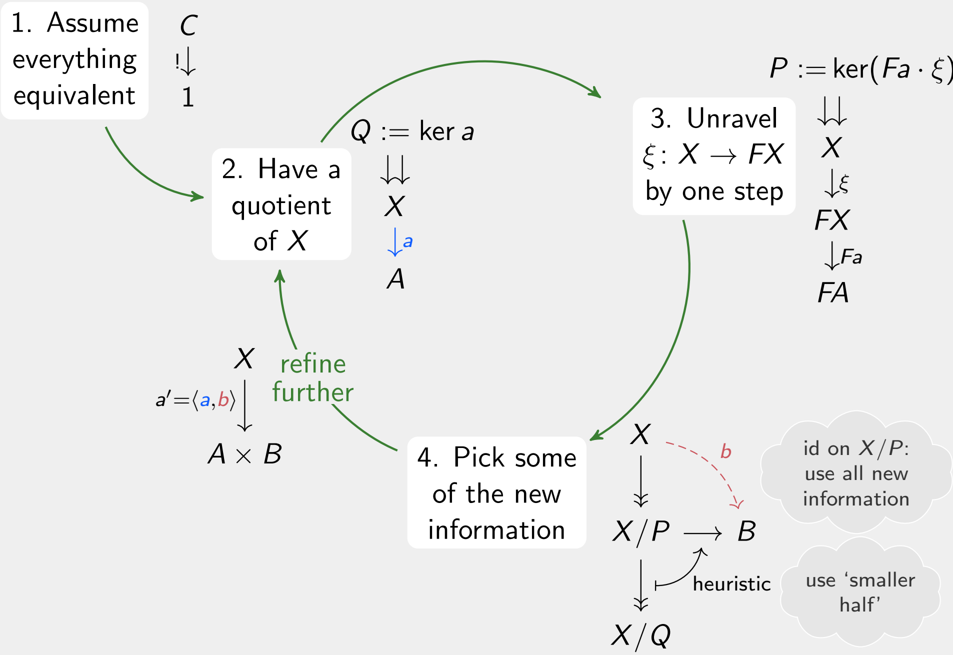
4. Pick some of the new information

$$\begin{tikzcd} X \arrow[d, two heads] \arrow[dr, dashed, bend left, "b"] & \\ X/P \arrow[r] \arrow[d, two heads] & B \\ X/Q & \end{tikzcd}$$

heuristic

refine further

$$\begin{tikzcd} X \arrow[d, "a'=\langle a,b\rangle"'] \\ A \times B \end{tikzcd}$$



1. Assume everything equivalent

$$\begin{tikzcd} C \arrow[d, "!"'] \\ 1 \end{tikzcd}$$

2. Have a quotient of X

$$\begin{tikzcd} Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "a"] \\ A \end{tikzcd}$$

3. Unravel $\xi\colon X \to FX$ by one step

$$\begin{tikzcd} P := \ker(Fa \cdot \xi) \arrow[d, shift left] \arrow[d, shift right] \\ X \arrow[d, "\xi"] \\ FX \arrow[d, "Fa"] \\ FA \end{tikzcd}$$

4. Pick some of the new information

$$\begin{tikzcd} X \arrow[d, two heads] \arrow[dr, dashed, bend left, "b"] & \\ X/P \arrow[r] \arrow[d, two heads] & B \\ X/Q & \end{tikzcd}$$

heuristic

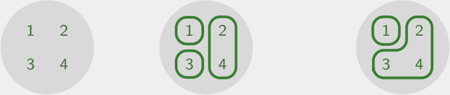
id on X/P: use all new information

use ‘smaller half’

refine further

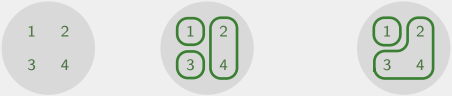
$$\begin{tikzcd} X \arrow[d, "a'=\langle a,b\rangle"'] \\ A \times B \end{tikzcd}$$

Heuristic



$$\begin{tikzcd}
X \arrow[r, two heads] & X/P \arrow[r, two heads, "p", ""{name=P, below}] \arrow[d, ""{name=D, right}] & X/Q \\
& B &
\arrow[from=P, to=D, mapsto, bend left, "\text{heuristic}"]
\end{tikzcd}$$

Heuristic

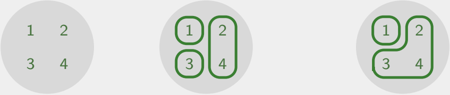


$$\begin{tikzcd}
X \arrow[r, two heads] & X/P \arrow[r, two heads, "p"{name=P}] \arrow[d, ""{name=D}] & X/Q \\
& B &
\arrow[from=P, to=D, mapsto, bend left, "heuristic"]
\end{tikzcd}$$

Use all new information

$B = X/P \rightsquigarrow$ Final Chain algorithm (König, Küpper 2014)

Heuristic



$$\begin{tikzcd}
X \arrow[r, two heads] & X/P \arrow[r, two heads, "p"{name=P}] \arrow[d, ""{name=D}] & X/Q \\
& B &
\arrow[from=P, to=D, maps to, bend left, "\text{heuristic}"]
\end{tikzcd}$$

Use all new information

$B = X/P \rightsquigarrow$ Final Chain algorithm (König, Küpper 2014)

Process the ‘smaller half’

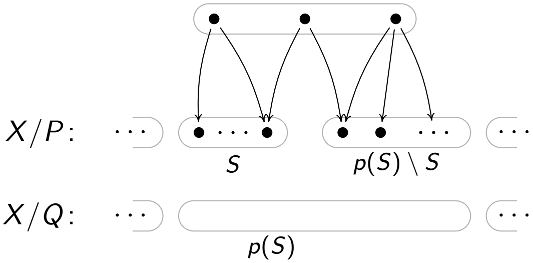
Surrounding block in X/Q

Let $S \in X/P$, such that $2 \cdot |S| \leq |p(S)|$

$$B = \{\overset{\{3\}}{\text{ChosenBlock}}, \overset{\{2,4\}}{\text{SameSurroundingBlock}}, \overset{\{1\}}{\text{RemainingBlocks}}\}$$

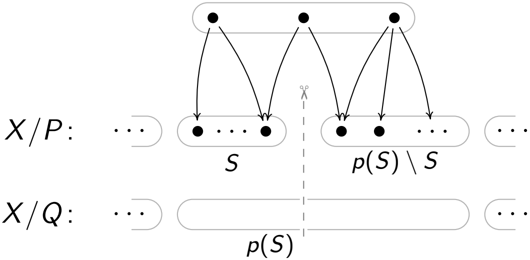
Three-way Splitting

$$\begin{tikzcd} X \arrow[r, two heads] & X/P \arrow[r, two heads, "p"] & X/Q \end{tikzcd}$$



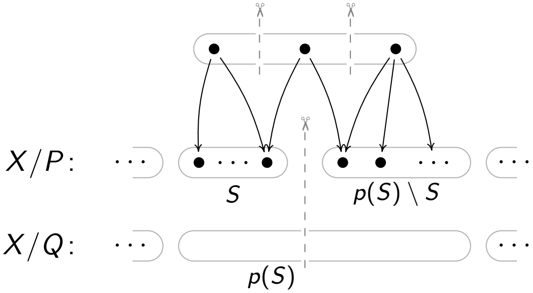
Three-way Splitting

$$\begin{tikzcd} X \arrow[r, two heads] & X/P \arrow[r, two heads, "p"] & X/Q \end{tikzcd}$$



Three-way Splitting

$$\begin{tikzcd} X \arrow[r, two heads] & X/P \arrow[r, two heads, "p"] & X/Q \end{tikzcd}$$



Assume

- Finitely complete category $\mathcal{C}$
- (RegularEpi,Mono)-factorisations
- F mono-preserving

Theorem (Correctness)

$$\begin{tikzcd}
X \arrow[r, "\xi"] \arrow[d, two heads] & FX \arrow[d] \\
X/P_i \arrow[r] & F(X/Q_i)
\end{tikzcd}$$

Assume

- Finitely complete category $\mathcal{C}$
- (RegularEpi,Mono)-factorisations
- F mono-preserving

Theorem (Correctness)

$$\begin{tikzcd}
X \arrow[r, "\xi"] \arrow[d, two heads] & FX \arrow[d] \\
X/P_i \arrow[r] & F(X/Q_i)
\end{tikzcd}$$

If $P_i \cong Q_i$, then this

1. is a coalgebra
2. has no proper quotient

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \\
A
\end{tikzcd}$$

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a' = \langle a, b \rangle"] \\
A & A \times B
\end{tikzcd}$$

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
 & & P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] \\
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] & X \arrow[d, "\xi"] \\
X \arrow[r] \arrow[d, "a"] & X \arrow[d, "a' = \langle a, b \rangle"] & FX \arrow[d, "Fa"] \\
A & A \times B & FA
\end{tikzcd}$$

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a' = \langle a, b \rangle"] \\
A & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & {\color{red}??} \cap {\color{red}??} \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] \arrow[r] & FX \arrow[d, "F\langle a, b \rangle"] \\
FA & F(A \times B)
\end{tikzcd}$$

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a' = \langle a,b \rangle"] \\
A & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & ?? \cap ?? \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] \arrow[r] & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] & FX \arrow[d, "F\langle a,b \rangle"] \\
FA & F(A \times B)
\end{tikzcd}$$

Question: When is $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$?

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a'=\langle a,b\rangle"] \\
A & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & ?? \cap ?? \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] \arrow[r] & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] & FX \arrow[d, "F\langle a,b\rangle"] \\
FA & F(A \times B)
\end{tikzcd}$$

Theorem. In Set, $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$ if

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a'=\langle a,b\rangle"] \\
A & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & {\color{red} ?? \cap ??} \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] \arrow[r] & FX \arrow[d, "F\langle a,b\rangle"] \\
FA & F(A \times B)
\end{tikzcd}$$

Theorem. In Set, $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$ if

$$\begin{tikzcd}
F(L+R) \arrow[d] \\
F(L+1) \times F(1+R)
\end{tikzcd}$$

injective and

“zippable”

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a'=\langle a,b\rangle"] \\
A & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & ?? \cap ?? \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] \arrow[r] & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] & FX \arrow[d, "F\langle a,b\rangle"] \\
FA & F(A \times B)
\end{tikzcd}$$

Theorem. In Set, $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$ if

$$\begin{tikzcd}
F(L+R) \arrow[d] \\
F(L+1) \times F(1+R)
\end{tikzcd}$$

injective and $\ker a \cup \ker b$ an equivalence

“zippable”

Ingredient 1: Zippable Functors

Definition. Functor F is **zippable**, if

$$F(L+R) \xrightarrow{\text{unzip}} F(L+1) \times F(1+R) \text{ is monic.}$$

Examples.

- $\mathsf{Id}\colon \mathcal{C} \to \mathcal{C}$ (if $\mathcal{C}$ is extensive),
- constant functors,
- products,
- coproducts (for $\mathcal{C} = \mathsf{Set}$),
- subfunctors,
- partially additive functors: $F(X+Y) \overset{\text{natural in } X, Y}{\rightarrowtail} FX \times FY$,
- $M^{(-)}$, $(-)^*$, $\mathcal{P}_\mathrm{f}$ on Set.

Examples for sets $L = \{a_1, a_2, a_3\}$, $R = \{b_1, b_2\}$, $1 = \{\_\}$

$$\begin{tikzcd} a_1\ a_2\ b_1\ a_3\ b_2 \arrow[d, mapsto, "\text{unzip}"] \\ (\,a_1\ \ a_2\ \ \_\ \ a_3\ \ \_\,,\ \ \_\ \ \_\ \ b_1\ \ \_\ \ b_2) \end{tikzcd}$$

$(-)^*$ is zippable

$$\begin{tikzcd} \{a_1, a_2, b_1\} \arrow[d, mapsto, "\text{unzip}"] \\ (\{a_1, a_2, \_\},\ \{\_, b_1\}) \end{tikzcd}$$

$\mathcal{P}_\mathrm{f}$ is zippable

Examples for sets $L = \{a_1, a_2, a_3\}$, $R = \{b_1, b_2\}$, $1 = \{\_\}$

$$\begin{tikzcd} a_1\ a_2\ b_1\ a_3\ b_2 \arrow[d, mapsto, bend left, "\text{unzip}"] \\ (a_1\ a_2\ \_\ a_3\ \_,\ \ \_\ \_\ b_1\ \_\ b_2) \end{tikzcd}$$

$(-)^*$ is zippable

$$\begin{tikzcd} \{a_1, a_2, b_1\} \arrow[d, mapsto, bend left, "\text{unzip}"] \\ (\{a_1, a_2, \_\},\ \{\_, b_1\}) \end{tikzcd}$$

$\mathcal{P}_\mathrm{f}$ is zippable

$$\begin{tikzcd} \{\{a_1, b_1\}, \{a_2, b_2\}\} \arrow[dr, mapsto, "\text{unzip}"'] & & \{\{a_1, b_2\}, \{a_2, b_1\}\} \arrow[dl, mapsto, "\text{unzip}"] \\ & (\{\{a_1, \_\}, \{a_2, \_\}\},\ \{\{\_, b_1\}, \{\_, b_2\}\}) & \end{tikzcd}$$

$\mathcal{P}_\mathrm{f}\mathcal{P}_\mathrm{f}$ is not zippable

~~Composition~~

~~Quotients~~

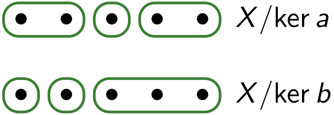
Ingredient 2: Well-behaved Heuristics

ker $a \cup$ ker b is an equivalence in Set $A \xleftarrow{a} X \xrightarrow{b} B$

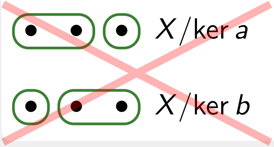
$\Leftrightarrow$ ker $a \cup$ ker b transitive

$\Leftrightarrow$ $\forall x \in C : [x]_a \subseteq [x]_b$ or $[x]_a \supseteq [x]_b$

Example



Non-Example



Ingredient 2: Well-behaved Heuristics

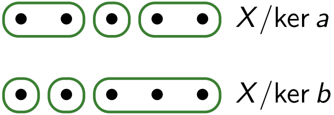
$\ker a \cup \ker b$ is an equivalence in Set

$$A \xleftarrow{a} X \xrightarrow{b} B$$

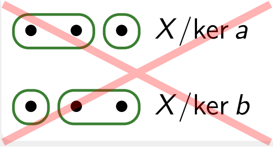
$\Leftrightarrow$ $\ker a \cup \ker b$ transitive

$\Leftrightarrow$ $\forall x \in C : [x]_a \subseteq [x]_b$ or $[x]_a \supseteq [x]_b$

Example



Non-Example



Heuristics **respecting compound blocks**:

$$\begin{tikzcd}
X \arrow[r, two heads] \arrow[rr, bend left, two heads, "a"] \arrow[dr, dashed, bend right, "b"'] & X/P \arrow[r, two heads, "p"] \arrow[d, "\text{heuristic}"] & X/Q \\
& B &
\end{tikzcd}$$

s.th. $\ker a \cup \ker b$ equivalence

E.g. **use all new information** and **process the smaller half**

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[rr] & & X \arrow[d, "a'=\langle a,b\rangle"] \\
A & & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & & ?? \cap ?? \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] \arrow[rr] & & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] & & FX \arrow[d, "F\langle a,b\rangle"] \\
FA & & F(A \times B)
\end{tikzcd}$$

Theorem. In Set, $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$ if

$$\begin{tikzcd}
F(L+R) \arrow[d] \\
F(L+1) \times F(1+R)
\end{tikzcd}$$

injective and $\ker a \cup \ker b$ an equivalence

“zippable”

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[r] & X \arrow[d, "a' = \langle a, b \rangle"] \\
A & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & P \cap \ker(Fb \cdot c) \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] \arrow[r] & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] & FX \arrow[d, "F\langle a, b \rangle"] \\
FA & F(A \times B)
\end{tikzcd}$$

Theorem. In Set, $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$ if

$$\begin{tikzcd}
F(L+R) \arrow[d] \\
F(L+1) \times F(1+R)
\end{tikzcd}$$

injective and $\ker a \cup \ker b$ an equivalence

“zippable”

Efficiency: Incremental Partitions

Concrete Algorithm?

Incremental partitions

$$\begin{tikzcd}
Q := \ker a \arrow[d, shift left] \arrow[d, shift right] & & Q \cap \ker b \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "a"] \arrow[rr] & & X \arrow[d, "a'=\langle a,b\rangle"] \\
A & & A \times B
\end{tikzcd}$$

$$\begin{tikzcd}
P := \ker(c \cdot Fa) \arrow[d, shift left] \arrow[d, shift right] & & {\color{red} P \cap \ker(Fb \cdot c)} \arrow[d, shift left] \arrow[d, shift right] \\
X \arrow[d, "\xi"] \arrow[rr] & & X \arrow[d, "\xi"] \\
FX \arrow[d, "Fa"] & & FX \arrow[d, "F\langle a,b\rangle"] \\
FA & & F(A \times B)
\end{tikzcd}$$

Theorem. In Set, $\ker F\langle a, b\rangle = \ker\langle Fa, Fb\rangle$ if

$$\begin{tikzcd}
F(L+R) \arrow[d] \\
F(L+1) \times F(1+R)
\end{tikzcd}$$

injective and $\ker a \cup \ker b$ an equivalence

“zippable”

Setting for complexity analysis

Category:	Heuristic:	Functor:
Set	process the smaller half	zippable & **refinement interface**

Setting for complexity analysis

Category:	Heuristic:	Functor:
Set	process the smaller half	zippable & **refinement interface**

Definition.

Functor encoding: set L of edge labels plus family of functions

$$\flat_X \colon FX \to \mathcal{B}_{\mathrm{f}}(L \times X) \qquad (\textbf{not} \text{ natural})$$

Encoding of coalgebra: $X \xrightarrow{\xi} FX \xrightarrow{\langle F!, \flat_X\rangle} F1 \times \mathcal{B}_{\mathrm{f}}(L \times X)$ has

$$n = |X| \textbf{ states} \text{ and } m = \textstyle\sum_{x \in X} |\flat_X(\xi(x))| \textbf{ edges}.$$

Setting for complexity analysis

Category:	Heuristic:	Functor:
Set	process the smaller half	zippable & **refinement interface**

Definition.

Functor encoding: set L of edge labels plus family of functions

$$\flat_X \colon FX \to \mathcal{B}_{\mathrm{f}}(L \times X) \qquad (\textbf{not}\ \text{natural})$$

Encoding of coalgebra: $X \xrightarrow{\xi} FX \xrightarrow{\langle F!, \flat_X \rangle} F1 \times \mathcal{B}_{\mathrm{f}}(L \times X)$ has

$n = |X|$ **states** and $m = \sum_{x \in X} |\flat_X(\xi(x))|$ **edges**.

E.g.

$L = 1$, $\flat_X \colon \mathcal{P}_{\mathrm{f}} X \to \mathcal{B}_{\mathrm{f}}(1 \times X)$

$L = \mathbb{N}$, $\flat_X \colon F_\Sigma X \to \mathcal{B}_{\mathrm{f}}(\mathbb{N} \times X)$

$\flat_X(\sigma(\vec{x}_i)) = \{(1, x_1), \ldots, (n, x_n)\}$

Definition. Refinement Interface

Set W (of weights) plus

$$\mathsf{init}\colon F1 \times \mathcal{B}_{\mathrm{f}}(L) \to W, \qquad \mathsf{update}\colon \overset{\text{edges into } S}{\mathcal{B}_{\mathrm{f}} L} \times \overset{\text{weight of } p(S)}{W} \longrightarrow \overset{\text{weight of } S}{W} \times \overset{F\langle \chi_S, \chi_{p(S)}\rangle}{F(2 \times 2)} \times \overset{\text{weight of } p(S) \setminus S}{W}$$

s.th. some axioms are satisfied.

Definition. Refinement Interface

Set W (of weights) plus

$$\mathsf{init}\colon F1 \times \mathcal{B}_\mathrm{f}(L) \to W, \qquad \mathsf{update}\colon \overset{\text{edges into } S}{\mathcal{B}_\mathrm{f} L} \times \overset{\text{weight of } p(S)}{W} \longrightarrow \overset{\text{weight of } S}{W} \times \overset{F\langle \chi_S, \chi_{p(S)}\rangle}{F(2 \times 2)} \times \overset{\text{weight of } p(S) \setminus S}{W}$$

s.th. there exists a **weight function** $w\colon \mathcal{P}_\mathrm{f} X \to (FX \to W)$ satisfying

$$\begin{tikzcd}
F1 \times \mathcal{B}_\mathrm{f} L \arrow[r, "\mathsf{init}"] & W \\
FX \arrow[u, "{\langle F!, \mathrm{fil}_X \cdot \flat_X \rangle}"] \arrow[ur, "w(X)"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
\mathcal{B}_\mathrm{f} L \times W \arrow[r, "\mathsf{update}"] & W \times F(2 \times 2) \times W \\
FX \arrow[u, "{\langle \mathrm{fil}_S \cdot \flat_X, w(T) \rangle}"] \arrow[ur, "{\langle w(S), F\langle \chi_S, \chi_T \rangle, w(T \setminus S) \rangle}"'] &
\end{tikzcd} \qquad \text{for all } S \subseteq \overset{p(S)}{T} \subseteq X$$

where $\mathrm{fil}_S\colon \mathcal{B}_\mathrm{f}(A \times X) \to \mathcal{B}_\mathrm{f}(A)$, $\qquad \mathrm{fil}_S(f)(a) = \sum_{y \in S} f(a, y)$

Examples

- For $F = \mathcal{P}_\mathrm{f}$: $\quad L = 1, \quad W = 2 \times \mathbb{N},$

  $w\colon \mathcal{P}_\mathrm{f}X \to (\mathcal{P}_\mathrm{f}X \to 2 \times \mathbb{N}) \quad w(T)(t) = (|t \setminus T| \overset{?}{>} 0, |t \cap T|)$

  $\mathrm{init}\colon \mathcal{P}_\mathrm{f}1 \times \mathbb{N} \to 2 \times \mathbb{N} \qquad \mathrm{init}(z, n) = (0, n)$

  $\mathrm{update}\colon \mathbb{N} \times (2 \times \mathbb{N}) \to (2 \times \mathbb{N}) \times \mathcal{P}_\mathrm{f}3 \times (2 \times \mathbb{N})$

  $$\begin{aligned}\mathrm{update}(n_S, (r, n_{T\setminus S})) = \big(&(r \vee (n_{T\setminus S} \overset{?}{>} 0), n_S),\\ &(r, n_{T\setminus S} \overset{?}{>} 0, n_S \overset{?}{>} 0),\\ &(r \vee (n_S \overset{?}{>} 0), n_{T\setminus S})\big),\end{aligned}$$

Examples

- For $F = \mathcal{P}_\mathrm{f}$: $\quad L = 1, \quad W = 2 \times \mathbb{N}$,

  $w\colon \mathcal{P}_\mathrm{f}X \to (\mathcal{P}_\mathrm{f}X \to 2 \times \mathbb{N}) \quad w(T)(t) = (|t \setminus T| \overset{?}{>} 0, |t \cap T|)$

  $\mathrm{init}\colon \mathcal{P}_\mathrm{f}1 \times \mathbb{N} \to 2 \times \mathbb{N} \qquad \mathrm{init}(z, n) = (0, n)$

  $\mathrm{update}\colon \mathbb{N} \times (2 \times \mathbb{N}) \to (2 \times \mathbb{N}) \times \mathcal{P}_\mathrm{f}3 \times (2 \times \mathbb{N})$

  $$\begin{aligned}\mathrm{update}(n_S, (r, n_{T\setminus S})) = \big((&r \vee (n_{T\setminus S} \overset{?}{>} 0), n_S),\\ &(r, n_{T\setminus S} \overset{?}{>} 0, n_S \overset{?}{>} 0),\\ &(r \vee (n_S \overset{?}{>} 0), n_{T\setminus S})\big),\end{aligned}$$

-

Functor:	$G^{(-)}$	$\mathcal{B}_\mathrm{f}$	$\mathcal{D}$	F_Σ
Labels L:	G	$\mathbb{N}$	$[0,1]$	$\mathbb{N}$
Weights W:	$G^{(2)}$	$\mathcal{B}_\mathrm{f}2$	$\mathcal{D}2$	$F_\Sigma 2$
$w(T)$, $T \subseteq X$:	$G\chi_T$	$\mathcal{B}_\mathrm{f}\chi_T$	$\mathcal{D}\chi_T$	$F_\Sigma\chi_T$

Initialization

```
for $e \in E$, $e = x \xrightarrow{a} y$ do
    add $e$ to toSub[$x$] and pred[$y$]
for $x \in X$ do
    $p_X$ := new cell in deref containing init(type[$x$], $\mathcal{B}_\mathfrak{f}(\pi_2 \cdot \text{graph})(\text{toSub}[x])$)
    for $e \in$ toSub[$x$] do lastW[$e$] = $p_X$
    toSub[$x$] := $\emptyset$
$X/P$ := group $X$ by type: $X \to H1$; $X/Q := \{X\}$.
```

$= p(S) \in X/Q$

Split$(X/P, S \subseteq T \subseteq X)$

```
M := $\emptyset \subseteq X/P \times H3$
for $y \in S$, $e \in$ pred[$y$] do
    $x \xrightarrow{a} y := e$
    $B$ := block with $x \in B \in X/P$
    if mark$_B$ is empty then
        $w_T^x$ := deref $\cdot$ lastW[$e$]
        $v_\emptyset := \pi_2 \cdot$ update($\emptyset$, $w_T^x$)
        add ($B$, $v_\emptyset$) to M
    if toSub[$x$] = $\emptyset$ then
        add ($x$, lastW[$e$]) to mark$_B$
    add $e$ to toSub[$x$]
```

(a) Collecting predecessor blocks

```
for ($B$, $v_\emptyset$) $\in$ M do
    $B_{\neq\emptyset} := \emptyset \subseteq X \times H3$
    for ($x$, $p_T$) in mark$_B$ do
        $\ell := \mathcal{B}_\mathfrak{f}(\pi_2 \cdot \text{graph})(\text{toSub}[x])$
        ($w_S^x$, $v^x$, $w_{T\setminus S}^x$) := update($\ell$, deref[$p_T$])
        deref[$p_T$] := $w_{T\setminus S}^x$
        $p_S$ := new cell containing $w_S^x$
        for $e \in$ toSub[$x$] do lastW[$e$] := $p_S$
        toSub[$x$] := $\emptyset$
        if $v^x \neq v_\emptyset$ then
            remove $x$ from $B$
            insert ($x$, $v^x$) into $B_{\neq\emptyset}$
    mark$_B$ := $\emptyset$
    $B_1 \times \{v_1\}, \dots, B_\ell \times \{v_\ell\}$ :=
      group $B_{\neq\emptyset}$ by $\pi_2$: $X \times H3 \to H3$
    insert $B_1, \dots, B_\ell$ := into $X/P$
```

(b) Splitting predecessor blocks

Assumptions.

- Refinement interface running in time $\mathcal{O}(\overset{\ell \in \mathcal{B}_\mathrm{f}(L)}{|\ell|} \cdot c(\overset{\text{states}}{n}, \overset{\text{edges}}{m}))$
  $\{$ `type W, type L, init(), update()` $\}$

Assumptions.

- Refinement interface running in time $\mathcal{O}(|\overset{\ell \in \mathcal{B}_{\mathrm{f}}(L)}{\ell}| \cdot c(\overset{\text{states}}{n}, \overset{\text{edges}}{m}))$

  `{ type W, type L, init(), update() }`
- coalgebra structure as edges with labels

$$C \xrightarrow{c} FC \xrightarrow{\flat_C} \mathcal{B}_{\mathrm{f}}(L \times C)$$

$\Rightarrow$ compute “smaller half” intersections in “linear” time

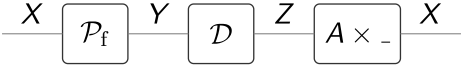
Theorem

1. Pseudocode **correctly** computes the simple quotient.
2. Overall **complexity**: $\mathcal{O}((m+n) \cdot \log n \cdot c(n, m))$

Wißmann, Dorsch, Schröder, Milius (CONCUR 2017, LMCS 2020)

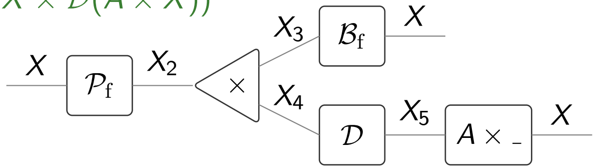
Modularity (and how to deal with non-zippable functors)

$FX = \mathcal{P}_{\mathrm{f}}(\mathcal{D}(A \times X))$



$$H\colon \mathsf{Set}^3 \to \mathsf{Set}^3$$
$$H(X, Y, Z) = (\mathcal{P}_{\mathrm{f}} Y, \mathcal{D} Z, A \times X)$$

$FX = \mathcal{P}_{\mathrm{f}}(\mathcal{B}_{\mathrm{f}} X \times \mathcal{D}(A \times X))$



$$H\colon \mathsf{Set}^5 \to \mathsf{Set}^5$$
$$H(X, X_2, X_3, X_4, X_5) = (\mathcal{P}_{\mathrm{f}} X_2, X_3 \times X_4, \mathcal{B}_{\mathrm{f}} X, \mathcal{D} X_5, A \times X)$$

Details: Wißmann, Dorsch, Milius, Schröder (LMCS 2020); based on Schröder, Pattinson 2011

$$FX = \mathcal{P}_{\mathrm{f}}(\mathcal{D}(A \times X)) \quad \rightsquigarrow \quad H(X, Y, Z) = (\mathcal{P}_{\mathrm{f}} Y, \mathcal{D}Z, A \times X)$$

$X \to FX$ in Set $\quad \rightsquigarrow \quad (X, Y, Z) \to (\mathcal{P}_{\mathrm{f}} Y, \mathcal{D}Z, A \times X)$ in Set^3

Summing up, one gets back to Set:

$$X + Y + Z \to \mathcal{P}_{\mathrm{f}} Y + \mathcal{D}Z + A \times X \xrightarrow{\mathsf{can}} (\mathcal{P}_{\mathrm{f}} + \mathcal{D} + A \times \_)(X + Y + Z)$$

$$FX = \mathcal{P}_{\mathrm{f}}(\mathcal{D}(A \times X)) \quad \rightsquigarrow \quad H(X, Y, Z) = (\mathcal{P}_{\mathrm{f}} Y, \mathcal{D} Z, A \times X)$$

$X \to FX$ in Set $\quad \rightsquigarrow \quad$ $(X, Y, Z) \to (\mathcal{P}_{\mathrm{f}} Y, \mathcal{D} Z, A \times X)$ in Set^3

Summing up, one gets back to Set:

$$X + Y + Z \to \mathcal{P}_{\mathrm{f}} Y + \mathcal{D} Z + A \times X \xrightarrow{\text{can}} (\mathcal{P}_{\mathrm{f}} + \mathcal{D} + A \times \_)(X + Y + Z)$$

Theorem.

The simple quotient of the ‘summed up’ coalgebra yields the simple quotient of the given coalgebra (X, ξ).

$FX = \mathcal{P}_\mathrm{f}(\mathcal{D}(A \times X)) \quad \rightsquigarrow \quad H(X, Y, Z) = (\mathcal{P}_\mathrm{f} Y, \mathcal{D}Z, A \times X)$

$X \to FX$ in Set $\quad \rightsquigarrow \quad (X, Y, Z) \to (\mathcal{P}_\mathrm{f} Y, \mathcal{D}Z, A \times X)$ in Set^3

Summing up, one gets back to Set:

$$X + Y + Z \to \mathcal{P}_\mathrm{f} Y + \mathcal{D}Z + A \times X \xrightarrow{\mathsf{can}} (\mathcal{P}_\mathrm{f} + \mathcal{D} + A \times \_)(X + Y + Z)$$

Theorem.

The simple quotient of the ‘summed up’ coalgebra yields the simple quotient of the given coalgebra (X, ξ).

Remark.

For zippable functors $F_1, \dots, F_n$ with their refinement interfaces one can easily construct a refinement interface for $\coprod F_i$.

The Tool CoPaR

Wißmann, Deifel, Milius, Schröder
FM 2019, Form. Asp. Comput. 2021

Generic and flexible implementation of the algorithm

- Implemented basic refinement interfaces: Σ, $\mathcal{P}_\mathrm{f}$, $\mathcal{B}_\mathrm{f}$, $\mathcal{D}$, $\underset{\uparrow \text{ monoid}}{M}^{(-)}$,
  for $M = \underbrace{\mathbb{N} \mid \mathbb{Z} \mid \mathbb{Q} \mid \mathbb{R}}_{\text{with } + \text{ or } \cdot} \mid (\mathbb{Z}, \mathsf{max}) \mid (\mathbb{R}, \mathsf{max})$

The Tool CoPaR

Wißmann, Deifel, Milius, Schröder
FM 2019, Form. Asp. Comput. 2021

Generic and flexible implementation of the algorithm

- Implemented basic refinement interfaces: Σ, $\mathcal{P}_{\mathrm{f}}$, $\mathcal{B}_{\mathrm{f}}$, $\mathcal{D}$, $\underset{\text{monoid}}{\underset{\uparrow}{M^{(-)}}}$,
  for $\quad M = \underbrace{\mathbb{N} \mid \mathbb{Z} \mid \mathbb{Q} \mid \mathbb{R}}_{\text{with } + \text{ or } \cdot} \mid (\mathbb{Z}, \max) \mid (\mathbb{R}, \max)$
- Interfaces for composed functors are automatically derived, e.g. for $\mathcal{P}_{\mathrm{f}}(A \times \mathcal{D}(-))$; in general according to the grammar

$$F ::= \overset{\text{functor variable}}{\overset{\downarrow}{\mathtt{X}}} \mid \mathcal{P}_{\mathrm{f}} F \mid \mathcal{B}_{\mathrm{f}} F \mid \mathcal{D} F \mid M^{(F)} \mid \overbrace{C \mid F + F \mid F \times F \mid F^A}^{\text{polynomial constructs}}$$

$$C ::= \mathbb{N} \mid A \qquad A ::= \{s_1, \dots, s_n\}$$

The Tool CoPaR

Wißmann, Deifel, Milius, Schröder
FM 2019, Form. Asp. Comput. 2021

Generic and flexible implementation of the algorithm

- Implemented basic refinement interfaces: Σ, $\mathcal{P}_\mathrm{f}$, $\mathcal{B}_\mathrm{f}$, $\mathcal{D}$, $M^{(-)}$,
  for $$M = \underbrace{\mathbb{N} \mid \mathbb{Z} \mid \mathbb{Q} \mid \mathbb{R}}_{\text{with } + \text{ or } \cdot} \mid (\mathbb{Z}, \max) \mid (\mathbb{R}, \max)$$
  monoid
- Interfaces for composed functors are automatically derived, e.g. for $\mathcal{P}_\mathrm{f}(A \times \mathcal{D}(-))$; in general according to the grammar

$$F ::= \overset{\text{functor variable}}{\mathtt{X}} \mid \mathcal{P}_\mathrm{f} F \mid \mathcal{B}_\mathrm{f} F \mid \mathcal{D} F \mid M^{(F)} \mid \overbrace{C \mid F + F \mid F \times F \mid F^A}^{\text{polynomial constructs}}$$

$$C ::= \mathbb{N} \mid A \qquad A ::= \{s_1, \ldots, s_n\}$$

- Users can easily implement new refinement interfaces.
- Available at `https://gitlab.cs.fau.de/i8/copar`

The Tool CoPaR

Refinement Interface Type

Math:

$$\begin{aligned} \mathsf{init}\colon & F1 \times \mathcal{B}_{\mathrm{f}} A \to W \\ \mathsf{update}\colon & \mathcal{B}_{\mathrm{f}} A \times W \to W \times F3 \times W \end{aligned}$$

Haskell:

```
class (Ord (F1 f), Ord (F3 f)) ⇒ RefinementInterface f where
  init   :: F1 f → [Label f] → Weight f
  update :: [Label f] → Weight f → (Weight f, F3 f, Weight f)
```

The Tool CoPaR

Example: Refinement Interface Implementation for $G^{(-)}$

Math:

$$\begin{aligned} \mathrm{init}(f_1, e) &= (0, \textstyle\sum e) \\ \mathrm{update}(e, (r, c)) &= ((r + c - \textstyle\sum e, \textstyle\sum e), (r, c - \textstyle\sum e, \textstyle\sum e), \\ &\qquad (\textstyle\sum e + r, c - \textstyle\sum e)) \end{aligned}$$

Haskell:

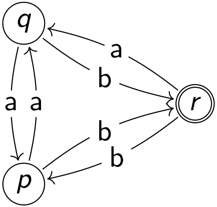
```haskell
instance RefinementInterface R where
  init  f1  e = (0, sum e)
  update e (r,c) = ((r + c - sum e, sum e),
                    (r,  c - sum e, sum e),
                    (sum e + r,  c - sum e))
```

The Tool CoPaR

Example: Input coalgebra for $\{f, n\} \times \mathcal{P}_{\mathrm{f}}(\{a, b\} \times X)$

```
{f,n} x P({a,b} x X)

p: (n, {(a,p), (b,r)})
q: (n, {(a,q), (b,r)})
r: (f, {(a,q), (b,p)})
```

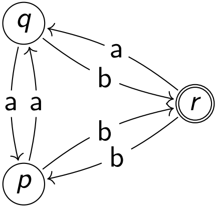


The Tool CoPaR

Example: Input coalgebra for $\{f, n\} \times \mathcal{P}_\mathrm{f}(\{a, b\} \times X)$

```
{f,n} x P({a,b} x X)

p: (n, {(a,p), (b,r)})
q: (n, {(a,q), (b,r)})
r: (f, {(a,q), (b,p)})
```



Output

```
Block 0: r
Block 2: q,p
```

Processing Times (in seconds) for weighted tree automata

$n =$ maximal states $(+\,50 \cdot n$ transitions) fitting 16 GB of RAM

$\rightsquigarrow m =$ 11–17 million

$t_p =$ parsing

$t_a =$ partition refinement

$\Sigma X =$	$4 \times X$			$4 \times X^2$			$4 \times X^3$			$4 \times X^4$			$4 \times X^5$		
M	n	t_p	t_a	n	t_p	t_a	n	t_p	t_a	n	t_p	t_a	n	t_p	t_a
2	132177	53	188	98670	46	243	85016	47	187	59596	41	146	49375	38	114
$\mathbb{N}$	113957	61	141	92434	55	175	69623	49	152	57319	47	140	48962	45	112
2^{64}	114888	58	100	95287	54	138	70660	49	107	62665	48	92	49926	44	72

PRISM Benchmarks

PRISM Model	Input		Time (s) to			Time (s) of	
	States	Edges	Parse	Init	Refine	Valmari	mCRL2
fms (n=4)	35910	237120	0.48	0.12	0.16	0.21	–
fms (n=5)	152712	1111482	2.46	0.68	1.10	1.21	–
fms (n=6)	537768	4205670	9.94	2.91	5.56	5.84	–
wlan2_collide (COL=2,TRANS_TIME_MAX=10)	65718	94452	0.51	0.29	0.59	0.14	0.42
wlan0_time_bounded (TRANS_TIME_MAX=10,DEADLINE=100)	582327	771088	5.26	3.07	5.52	0.92	3.18
wlan1_time_bounded (TRANS_TIME_MAX=10,DEADLINE=100)	1408676	1963522	13.42	6.17	16.13	2.52	8.58

System	Functor FX	Run-Time $(m \geq n)$		Specific algorithm	
Transition Systems	$\mathcal{P}_\mathrm{f} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \log m$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
			$>$	$m \cdot \log n$	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Valmari, Franceschinis 2010
DFA	$2 \times X^A$ (A fixed)	$n \cdot \log n$	$=$	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A\rvert \cdot n \cdot \log(n + \lvert A\rvert)$	$\approx$	$\lvert A\rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}X)$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	$<$	$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			$=$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	Groote, Verduzco, de Vink 2018
Color Refinement	$\mathcal{B}_\mathrm{f}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M \times M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$\lll$	$m \cdot n$	Högberg, Maletti, May 2007
	$M \times M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

System	Functor FX	Run-Time $(m \geq n)$		Specific algorithm	
Transition Systems	$\mathcal{P}_f X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_f(\mathbb{N} \times X)$	$m \cdot \log m$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
			$>$	$m \cdot \log n$	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Valmari, Franceschinis 2010
DFA	$2 \times X^A$ (A fixed)	$n \cdot \log n$	$=$	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_f(A \times X)$	$\lvert A\rvert \cdot n \cdot \log(n + \lvert A\rvert)$	$\approx$	$\lvert A\rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_f(A \times \mathcal{D}X)$	$m_{\mathcal{D}} \cdot \log m_{\mathcal{P}_f}$	$<$	[illegible] $\cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			[illegible]	$m_{\mathcal{D}} \cdot \log m_{\mathcal{P}_f}$	Groote, Verduzco, de Vink 2018
Color Refinement	$\mathcal{B}_f$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M \times M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$\ll$	$m \cdot n$	Högberg, Maletti, May 2007
	$M \times M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

Generic & Efficient

System	Functor FX	Run-Time ($m \geq n$)		Specific algorithm	
Transition Systems	$\mathcal{P}_\mathrm{f} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \mathrm{lo}$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
				n	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$			$\log n$	Valmari, nceschinis 2010
DFA	$2 \times X^A$ (A fixed)	$n \cdot \log n$	$=$	$n \cdot \log n$	pcroft 1971
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A \rvert \cdot n \cdot \log(n + \lvert A \rvert)$	$\approx$	$\lvert A \rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}X)$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	$<$	$\cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
				$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	Groote, Verduzco, de Vink 2018
Color Refinement	$\mathcal{B}_\mathrm{f}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M \times M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$\ll$	$m \cdot n$	Högberg, Maletti, May 2007
	$M \times M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

More instances:
further system types
& categories

Generic & Efficient

System	Functor FX	Run-Time $(m \geq n)$		Specific algorithm	
Transition Systems	$\mathcal{P}_\mathrm{f} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \mathrm{lo}$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
				n	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$			$\log n$	Valmari, anceschinis 2010
DFA		$n \cdot \log n$	$=$	$n \cdot \log n$	pcroft 1971
		$\lvert A\rvert \cdot n \cdot \log(n + \lvert A\rvert)$	$\approx$	$\lvert A \quad n \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(\quad \times \mathcal{D}X)$	$\mathcal{D} \cdot \log m$	$<$	$\cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
				$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	Groote, Verduzco, de Vink 2018
Color Refinement	$\mathcal{B}_\mathrm{f}$	$m \quad \mathrm{g}\ n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M \times M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$\ll$	$m \cdot n$	Högberg, Maletti, May 2007
	$M \times M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

More instances:
further system types
& categories

Nominal systems
& automata

Generic & Efficient

System	Functor FX	Run-Time $(m \geq n)$		Specific algorithm	
Transition Systems	$\mathcal{P}_\mathrm{f} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \mathrm{lo}$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
				n	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$			$\log n$	Valmari, nceschinis 2010
DFA		$n \cdot \log n$	$=$	$n \cdot \log n$	pcroft 1971
		$\lvert A \rvert \cdot n \cdot$ $\log(n + \lvert A \rvert)$	$\approx$	$\lvert A \rvert \cdot n \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(\times \mathcal{D}X)$	$\mathcal{D} \cdot \log m$	$<$	$\cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
					te, Verduzco, de Vink 2018
Color Refinement	$\mathcal{B}_\mathrm{f}$	$m \cdot \log n$			onsma, Grohe 2017
Weighted Tree Automata	$M \times M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$\ll$		erg, Maletti, May 2007
	$M \times M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

More instances:
further system types
& categories

Nominal systems
& automata

Generic & Efficient

Other types of
system equivalences

System	Functor FX	Run-Time $(m \geq n)$		Specific algorithm	
Transition Systems	$\mathcal{P}_\mathrm{f} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \mathrm{lo}$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
				n	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$			$\log n$	Valmari, nceschinis 2010
DFA		$n \cdot \log n$	$=$	$n \cdot \mathrm{lo}\ n$	pcroft 1971
		$\lvert A \rvert \cdot n \cdot$ $\log(n + \lvert A$	$\approx$	$\lvert A$ $n \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(\times \mathcal{D}X)$	$\mathcal{D} \cdot \log m$	$<$	$\cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
					te, Verduzco, de Vink 2018
Color Refi	$\mathcal{P}_\mathrm{f}$	m $g\ n$			nsma, Grohe 2017
	tive	$m \cdot \log^2 m$	$\ll$		erg, Maletti, May 2007
Au	) cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

More instances: further system types & categories

Nominal systems & automata

Generic & Efficient

Other types of system equivalences

Symbolic states & parallelization

System	Functor FX	Run-Time $(m \geq n)$		Specific algorithm	
Transition Systems	$\mathcal{P}_\mathrm{f} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Paige, Tarjan 1987
LTS	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \log m$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
			$>$	$m \cdot \log n$	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Valmari, Franceschinis 2010
DFA	$2 \times X^A$ (A fixed)	$n \cdot \log n$	$=$	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A\rvert \cdot n \cdot \log(n + \lvert A\rvert)$	$\approx$	$\lvert A\rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}X)$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	$<$	$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			$=$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	Groote, Verduzco, de Vink 2018
Color Refinement	$\mathcal{B}_\mathrm{f}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M \times M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$\lll$	$m \cdot n$	Högberg, Maletti, May 2007
	$M \times M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ poly. bound	$m \cdot \log n$	Högberg, Maletti, May 2007

Further Work

Deifel, Milius, Wißmann (FSCD 2021)

1. **Minimization:** given an (encoded) pointed coalgebra $1 \to C \to FC$, compute the minimized coalgebra:
   (a) compute the partition of the state set C as above
   (b) compute the coalgebra structure of the simple quotient $C' \to FC'$
   (c) compute the reachable part $1 \to C'' \to FC''$ of the simple quotient

Further Work

Deifel, Milius, Wißmann (FSCD 2021)

1. **Minimization:** given an (encoded) pointed coalgebra $1 \to C \to FC$, compute the minimized coalgebra:
   (a) compute the partition of the state set C as above
   (b) compute the coalgebra structure of the simple quotient $C' \to FC'$
   (c) compute the reachable part $1 \to C'' \to FC''$ of the simple quotient

   $\rightsquigarrow$ new steps (b) and (c) require non-trivial extension of the theory; minimization interface merge: $\mathcal{B}_{\mathrm{f}} L \to \mathcal{B}_{\mathrm{f}} L$

Further Work

Deifel, Milius, Wißmann (FSCD 2021)

1. **Minimization:** given an (encoded) pointed coalgebra $1 \to C \to FC$, compute the minimized coalgebra:
   (a) compute the partition of the state set C as above
   (b) compute the coalgebra structure of the simple quotient $C' \to FC'$
   (c) compute the reachable part $1 \to C'' \to FC''$ of the simple quotient

   $\rightsquigarrow$ new steps (b) and (c) require non-trivial extension of the theory; minimization interface merge: $\mathcal{B}_\mathrm{f} L \to \mathcal{B}_\mathrm{f} L$

Birkmann, Deifel, Milius (TACAS 2022)

2. **Distributed Algorithm** and Implementation:
   - memory consumption is the bottleneck in benchmarks
   - new distributed coalgebraic signature refinement algorithm (based on Blom and Orzan's signature refinement for LTSs)
   - benchmarking on high performance cluster

Further Work

Deifel, Milius, Wißmann (FSCD 2021)

1. **Minimization:** given an (encoded) pointed coalgebra $1 \to C \to FC$, compute the minimized coalgebra:
   (a) compute the partition of the state set C as above
   (b) compute the coalgebra structure of the simple quotient $C' \to FC'$
   (c) compute the reachable part $1 \to C'' \to FC''$ of the simple quotient

   $\rightsquigarrow$ new steps (b) and (c) require non-trivial extension of the theory; minimization interface merge: $\mathcal{B}_{\mathrm{f}} L \to \mathcal{B}_{\mathrm{f}} L$

Birkmann, Deifel, Milius (TACAS 2022)

2. **Distributed Algorithm** and Implementation:
   - memory consumption is the bottleneck in benchmarks
   - new distributed coalgebraic signature refinement algorithm (based on Blom and Orzan's signature refinement for LTSs)
   - benchmarking on high performance cluster

Deifel, Gebhart (2023)

3. **Symbolic representation** of states:
   - represent the state space by BDDs
   - implement a sequential version of coalgebraic signature refinement operating on BDDs

Appendix ...

Genericity: Initial partition

Given

$$C \xrightarrow{c} FC$$

Usual partition refinement algorithms

Return coarsest partition compatible with c, refining $C \overset{\kappa}{\twoheadrightarrow} \mathcal{I}$

Genericity: Initial partition

Given

$$C \xrightarrow{c} FC$$

Usual partition refinement algorithms

Return coarsest partition compatible with c, refining $C \overset{\kappa}{\twoheadrightarrow} \mathcal{I}$

$$\Updownarrow$$

Coalgebraic partition refinement for $\mathcal{I} \times F$

For the coalgebra $C \xrightarrow{\langle \kappa, c \rangle} \mathcal{I} \times FC$

Genericity: Composition

If F finitary,

$$C \xrightarrow{c} FG(C)$$

Genericity: Composition

If F finitary,

$$C \xrightarrow{c} FG(C) \qquad \rightsquigarrow \qquad D \xhookrightarrow{d} GC$$

finite subset
s.th. . . .

Genericity: Composition

If F finitary,

$$\begin{tikzcd}
C \arrow[r, "c"] \arrow[dr, dashed, "c'"'] & FG(C) & & D \arrow[r, hookrightarrow, "d"] & GC \\
& FD \arrow[u, "Fd"'] & & &
\end{tikzcd}$$

$$\rightsquigarrow$$

finite subset
s.th. . . .

Genericity: Composition

If F finitary,

$$\begin{tikzcd}
C \arrow[r, "c"] \arrow[dr, dashed, "c'"'] & FG(C) & \rightsquigarrow & D \arrow[r, hook, "d"] & GC \\
& FD \arrow[u, "Fd"'] & & \text{finite subset s.th. \dots} \arrow[u] &
\end{tikzcd}$$

A coalgebra Factor(C, c) on Set^2 for the functor $(X, Y) \mapsto (FY, GX)$:

$$(C, D) \xrightarrow{(c',d)} (FD, GC)$$

Genericity: Composition

If F finitary,

$$\begin{tikzcd}
C \arrow[r, "c"] \arrow[dr, dashed, "c'"'] & FG(C) \\
 & FD \arrow[u, "Fd"']
\end{tikzcd}
\qquad \rightsquigarrow \qquad
\begin{tikzcd}
D \arrow[r, hook, "d"] & GC
\end{tikzcd}$$

finite subset
s.th. . . .

A coalgebra Factor(C, c) on Set^2 for the functor $(X, Y) \mapsto (FY, GX)$:

$$(C, D) \xrightarrow{(c', d)} (FD, GC)$$

Examples

$$\mathcal{P}_\mathrm{f} \cdot (A \times (-)) \qquad (2 \times \mathcal{P}_\mathrm{f}) \cdot (A \times (-))$$
$$\mathcal{P}_\mathrm{f} \cdot (A \times (-)) \cdot \mathcal{D} \qquad \mathcal{P}_\mathrm{f} \cdot \mathcal{D} \cdot (A \times (-)) \qquad \dots$$

References

[BBG17] Christoph Berkholz, Paul S. Bonsma, Martin Grohe. "Tight Lower and Upper Bounds for the Complexity of Canonical Colour Refinement". In: **Theory Comput. Syst.** 60.4 (2017), pp. 581–614. DOI: `10.1007/s00224-016-9686-0`. URL: `https://doi.org/10.1007/s00224-016-9686-0`.

[BEM00] Christel Baier, Bettina Engelen, Mila Majster-Cederbaum. "Deciding Bisimilarity and Similarity for Probabilistic Processes". In: **J. Comput. Syst. Sci.** 60 (2000), pp. 187–231.

[DPP04] Agostino Dovier, Carla Piazza, Alberto Policriti. "An efficient algorithm for computing bisimulation equivalence". In: **Theor. Comput. Sci.** 311.1-3 (2004), pp. 221–256.

[Gri73] David Gries. "Describing an algorithm by Hopcroft". In: **Acta Inf.** 2 (1973), pp. 97–109. ISSN: 1432-0525.

[GVdV18] Jan Groote, Jao Verduzco, Erik de Vink. "An Efficient Algorithm to Determine Probabilistic Bisimulation". In: **Algorithms** 11.9 (2018), p. 131. DOI: 10.3390/a11090131. URL: https://doi.org/10.3390/a11090131.

[HMM07] Johanna Högberg, Andreas Maletti, Jonathan May. "Bisimulation Minimisation for Weighted Tree Automata". In: **Proceedings of the 11th International Conference on Developments in Language Theory**. DLT'07. Turku, Finland: Springer-Verlag, 2007, pp. 229–241. ISBN: 978-3-540-73207-5. URL: http://dl.acm.org/citation.cfm?id=1770310.1770335.

[Hop71] John Hopcroft. "An $n \log n$ algorithm for minimizing states in a finite automaton". In: **Theory of Machines and Computations**. Academic Press, 1971, pp. 189–196.

[Knu01] Timo Knuutila. "Re-describing an algorithm by Hopcroft". In: **Theor. Comput. Sci.** 250 (2001), pp. 333–363. ISSN: 0304-3975.

[PT87] Robert Paige, Pobert Tarjan. "Three partition refinement algorithms". In: **SIAM J. Comput.** 16.6 (1987), pp. 973–989.

[SP11] Lutz Schröder, Dirk Pattinson. "Modular Algorithms for Heterogeneous Modal Logics via Multi-Sorted Coalgebra". In: **Math. Struct. Comput. Sci.** 21.2 (2011), pp. 235–266.

[Val09] Antti Valmari. "Bisimilarity Minimization in $\mathcal{O}(m \log n)$ Time". In: **Applications and Theory of Petri Nets, PETRI NETS 2009**. Vol. 5606. LNCS. Springer, 2009, pp. 123–142. ISBN: 978-3-642-02423-8.

[VF10] Antti Valmari, Giuliana Franceschinis. "Simple $\mathcal{O}(m \log n)$ Time Markov Chain Lumping". In: **Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2010**. Vol. 6015. LNCS. Springer, 2010, pp. 38–52.