Generic and Efficient Partition Refinement

Thorsten Wißmann

University Erlangen-Nürnberg

Joint work with:

Hans-Peter Deifel, Ulrich Dorsch, Stefan Milius, Lutz Schröder

- Published in Concur 2017
- Extended version in LMCS 2020
- Implementation & more functors in FM2019

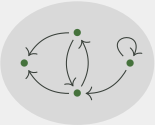
CS Theory Seminar (TSEM), Feb 04, 2021

Generic and Efficient Partition Refinement

Generic and Efficient Partition Refinement

Coalgebras:

State based systems

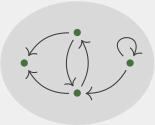


Labels, Non-Determinism, Probabilities, Automata, ...

Generic and Efficient Partition Refinement

Coalgebras:

State based systems



Labels, Non-Determinism, Probabilities, Automata, ...

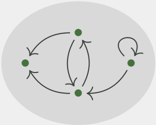
Modularity:

Combine system types by $\circ$, $\times$, $+$

Generic and Efficient Partition Refinement

Coalgebras:

State based systems



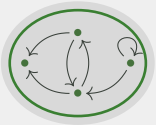
Labels, Non-Determinism, Probabilities, Automata, ...

Modularity:

Combine system types by $\circ$, $\times$, $+$

Partition Refinement:

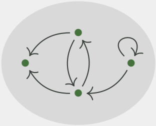
Successively distinguish different behaviour



Generic and Efficient Partition Refinement

Coalgebras:

State based systems



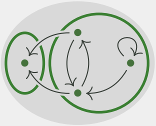
Labels, Non-Determinism, Probabilities, Automata, ...

Modularity:

Combine system types by $\circ$, $\times$, $+$

Partition Refinement:

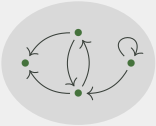
Successively distinguish different behaviour



Generic and Efficient Partition Refinement

Coalgebras:

State based systems



Labels, Non-Determinism, Probabilities, Automata, ...

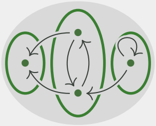
Modularity:

Combine system types by

$\circ$, $\times$, $+$

Partition Refinement:

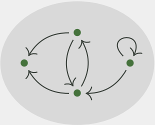
Successively distinguish different behaviour



Generic and Efficient Partition Refinement

Coalgebras:

State based systems



Labels, Non-Determinism, Probabilities, Automata, …

Modularity:

Combine system types by $\circ$, $\times$, $+$

Efficiency:

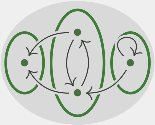
Complexity Analysis

$\mathcal{O}(m \cdot \log n)$

Edges States

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

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

(Labelled) Transition Systems

$m \cdot \log n$

Paige, Tarjan '87

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

(Labelled) Transition Systems

$m \cdot \log n$

Paige, Tarjan '87

Valmari '09

Weighted Systems "Markov Chain Lumping"

$m \cdot \log n$

Valmari, Franceschinis '10

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

(Labelled) Transition Systems

$m \cdot \log n$

Paige, Tarjan '87

Valmari '09

Segala Systems

$m_{\text{dist}} \cdot \log m_{\text{acts}}$

Groote, Verduzco, de Vink '18

Weighted Systems

"Markov Chain Lumping"

$m \cdot \log n$

Valmari, Franceschinis '10

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

(Labelled) Transition Systems

$m \cdot \log n$

Paige, Tarjan '87
Valmari '09

Weighted Systems "Markov Chain Lumping"

$m \cdot \log n$

Valmari, Franceschinis '10

Segala Systems

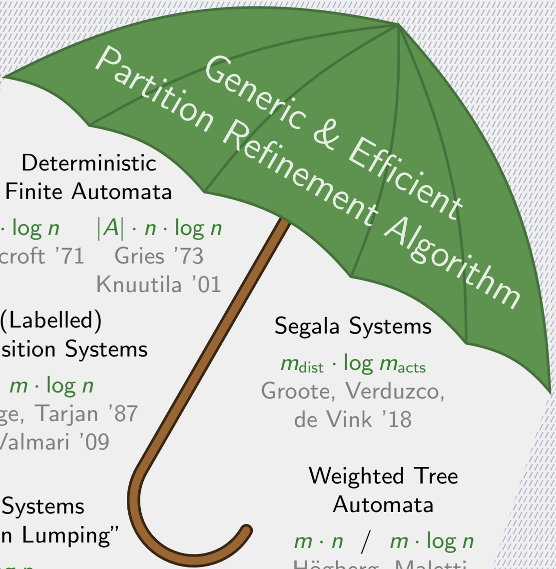
$m_{\text{dist}} \cdot \log m_{\text{acts}}$

Groote, Verduzco, de Vink '18

Weighted Tree Automata

$m \cdot n$ / $m \cdot \log n$

Högberg, Maletti, May '07



Deterministic
Finite Automata

$n \cdot \log n$ — Hopcroft '71

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

(Labelled)
Transition Systems

$m \cdot \log n$

Paige, Tarjan '87
Valmari '09

Weighted Systems
"Markov Chain Lumping"

$m \cdot \log n$

Valmari, Franceschinis '10

Segala Systems

$m_{\text{dist}} \cdot \log m_{\text{acts}}$

Groote, Verduzco,
de Vink '18

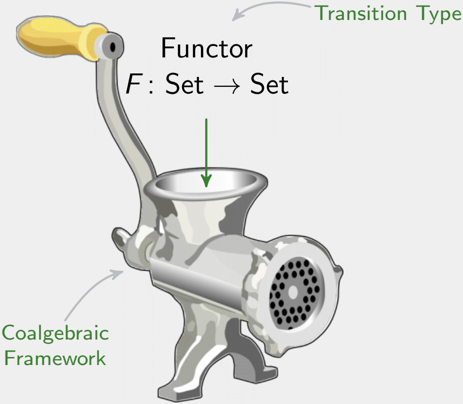
Weighted Tree
Automata

$m \cdot n$ / $m \cdot \log n$

Högberg, Maletti,
May '07

Coalgebra – Generic state based systems

Coalgebra – Generic state based systems

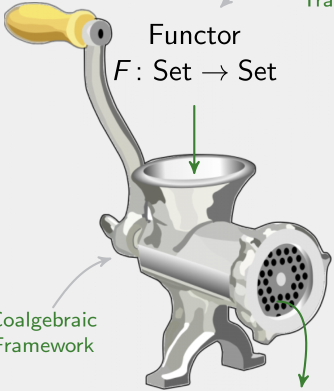


Coalgebra – Generic state based systems

Transition Type

Functor

$F\colon \mathsf{Set} \to \mathsf{Set}$



Coalgebraic Framework

F-Coalgebra

$C \xrightarrow{g} FC$

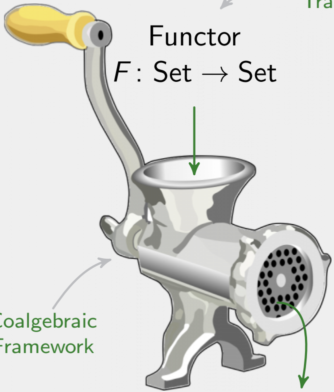
Coalgebra – Generic state based systems

Powerset $\mathcal{P}$

Transition Type

Functor

$F\colon \mathsf{Set} \to \mathsf{Set}$



Coalgebraic Framework

F-Coalgebra

$C \xrightarrow{g} FC$

Transition System

$C \xrightarrow{g} \mathcal{P}C$

Coalgebra – Generic state based systems

Powerset $\mathcal{P}$

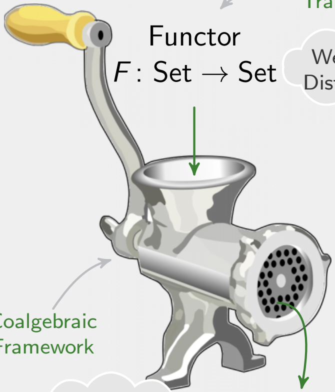
Transition Type

Functor

$F : \mathsf{Set} \to \mathsf{Set}$

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

Distributions $\mathcal{D}$



Coalgebraic Framework

Markov Chain

$C \xrightarrow{g} \mathbb{R}^C$

Transition System

$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra

$C \xrightarrow{g} FC$

Coalgebra – Generic state based systems

Powerset $\mathcal{P}$

Transition Type

Functor

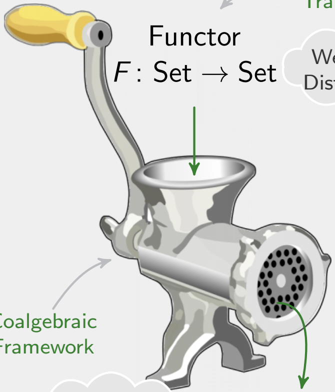
$F\colon \mathsf{Set} \to \mathsf{Set}$

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

Distributions $\mathcal{D}$

Automata

$2 \times (-)^A$



Coalgebraic Framework

Markov Chain

$C \xrightarrow{g} \mathbb{R}^C$

Transition System

$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra

$C \xrightarrow{g} FC$

Coalgebra – Generic state based systems

Powerset $\mathcal{P}$

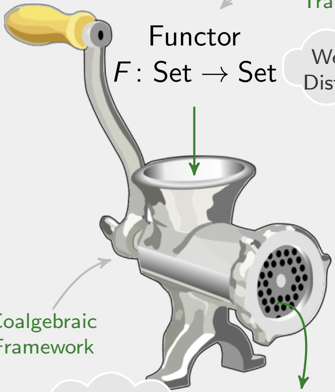
Simple Segala $\mathcal{P}_\mathrm{f}(A \times \mathcal{D}(-))$

Transition Type

Functor
$F\colon \mathsf{Set} \to \mathsf{Set}$

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

Automata
$2 \times (-)^A$



Coalgebraic Framework

Markov Chain
$C \xrightarrow{g} \mathbb{R}^C$

Transition System
$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra
$C \xrightarrow{g} FC$

Coalgebra – Generic state based systems

Transition Type

Functor

$F\colon \mathsf{Set} \to \mathsf{Set}$

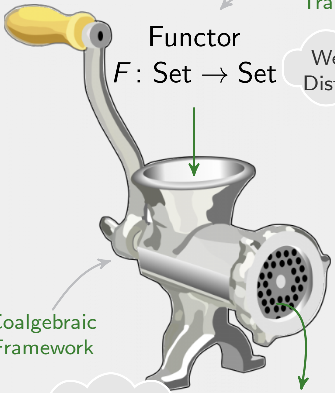
Powerset $\mathcal{P}$

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

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

Automata
$2 \times (-)^A$

Weighted Tree
Automata
$(M, +, 0)^{(\Sigma(-))}$



Coalgebraic Framework

Markov Chain
$C \xrightarrow{g} \mathbb{R}^C$

Transition System
$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra

$C \xrightarrow{g} FC$

Coalgebra – Generic state based systems

Powerset $\mathcal{P}$

Simple Segala $\mathcal{P}_f(A \times \mathcal{D}(-))$

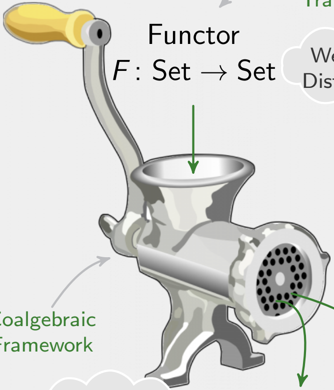
Transition Type

Functor
$F\colon \mathsf{Set} \to \mathsf{Set}$

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

Automata
$2 \times (-)^A$

Weighted Tree Automata
$(M, +, 0)^{(\Sigma(-))}$



Coalgebraic Framework

Markov Chain
$C \xrightarrow{g} \mathbb{R}^C$

Transition System
$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra
$C \xrightarrow{g} FC$

Homomorphism
$(C, g) \xrightarrow{h} (C', g')$

Coalgebra – Generic state based systems

Transition Type

Functor
$F\colon \mathsf{Set} \to \mathsf{Set}$

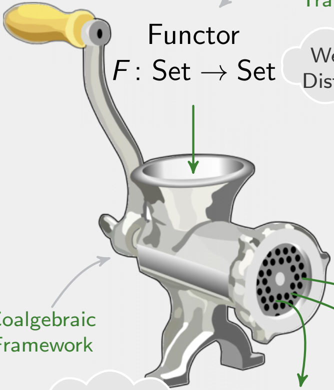
Powerset $\mathcal{P}$

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

Automata
$2 \times (-)^A$

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

Weighted Tree Automata
$(M, +, 0)^{(\Sigma(-))}$



Coalgebraic Framework

Markov Chain
$C \xrightarrow{g} \mathbb{R}^C$

Transition System
$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra
$C \xrightarrow{g} FC$

Homomorphism
$(C, g) \xrightarrow{h} (C', g')$

Behavioural Equivalence

Strong Bisimilarity

Language Equivalence

Coalgebra – Generic state based systems

Transition Type

Functor
$F\colon \mathsf{Set} \to \mathsf{Set}$

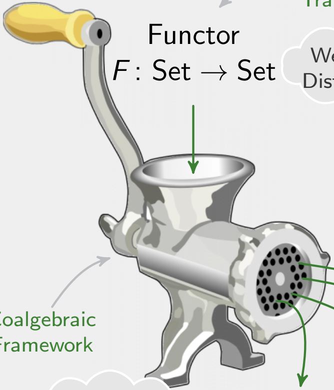
Powerset $\mathcal{P}$

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

Automata
$2 \times (-)^A$

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

Weighted Tree Automata
$(M, +, 0)^{(\Sigma(-))}$



Coalgebraic Framework

Minimization Algorithm

Behavioural Equivalence

Strong Bisimilarity

Language Equivalence

Markov Chain
$C \xrightarrow{g} \mathbb{R}^C$

Transition System
$C \xrightarrow{g} \mathcal{P}C$

F-Coalgebra
$C \xrightarrow{g} FC$

Homomorphism
$(C, g) \xrightarrow{h} (C', g')$

The Coalgebraic Task

For a functor $F : \mathsf{Set} \to \mathsf{Set}$

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

no proper quotient

all equivalent behaviours identified

The Coalgebraic Task

For a functor $F : \mathsf{Set} \to \mathsf{Set}$

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

no proper quotient

all equivalent behaviours identified

Instance

For $2 \times (-)^A$: Set

Automata minimization

The Coalgebraic Task

For a functor $F : \mathsf{Set} \to \mathsf{Set}$

Given a coalgebra

find the simple quotient

$$\begin{tikzcd} C \arrow[r, "g"] \arrow[d, two heads, "h"'] & FC \arrow[d, two heads, "Fh"] \\ C' \arrow[r, "g'"] & FC' \end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

Instance

Instance

For $2 \times (-)^A$:Set

Automata minimization

For $\mathcal{P}_\mathrm{f}$:Set

Bisimilarity minimization

The Coalgebraic Task

For a functor $F : \mathsf{Set} \to \mathsf{Set}$

$$\begin{tikzcd} C \arrow[r, "g"] \arrow[d, two heads, "h"'] & FC \arrow[d, two heads, "Fh"] \\ C' \arrow[r, "g'"] & FC' \end{tikzcd}$$

Given a coalgebra $C \xrightarrow{g} FC$

find the simple quotient $C' \xrightarrow{g'} FC'$

no proper quotient

all equivalent behaviours identified

Instance

Instance

Instance

For $2 \times (-)^A : \mathsf{Set}$
Automata minimization

For $\mathcal{P}_\mathrm{f} : \mathsf{Set}$
Bisimilarity minimization

For $\mathbb{R}^{(-)} : \mathsf{Set}$
Markov chain lumping

The Coalgebraic Task

For a functor $F : \mathsf{Set} \to \mathsf{Set}$

Given a coalgebra

find the simple quotient

$$\begin{tikzcd} C \arrow[r, "g"] \arrow[d, two heads, "h"'] & FC \arrow[d, two heads, "Fh"] \\ C' \arrow[r, "g'"] & FC' \end{tikzcd}$$

no proper quotient

all equivalent behaviours identified

Instance

Instance

Instance

Instance

For $2 \times (-)^A$: Set

Automata minimization

For $\mathcal{P}_\mathrm{f}$: Set

Bisimilarity minimization

For $\mathbb{R}^{(-)}$: Set

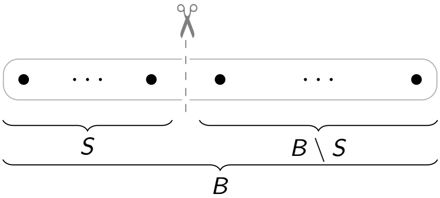
Markov chain lumping

. . .

Initially

All states of $g\colon C \to FC$ are grouped w.r.t. $C \xrightarrow{g} FC \xrightarrow{F!} F1$
(e.g. final vs. non-final states)

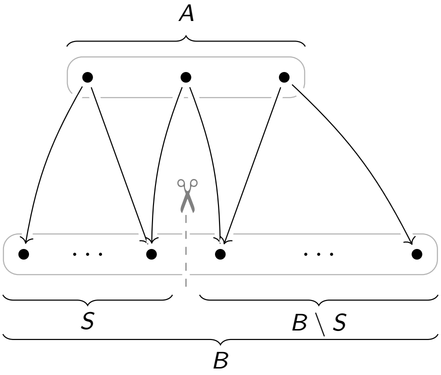
Refinement Step for $\mathcal{P}$



Initially

All states of $g\colon C \to FC$ are grouped w.r.t. $C \xrightarrow{g} FC \xrightarrow{F!} F1$ (e.g. final vs. non-final states)

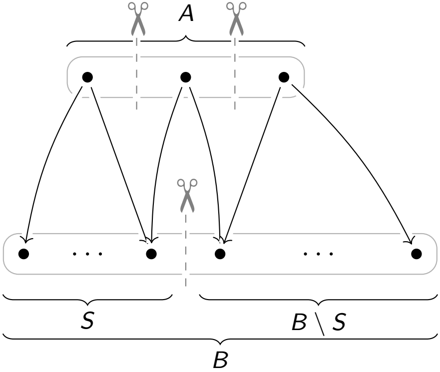
Refinement Step for $\mathcal{P}$



Initially

All states of $g\colon C \to FC$ are grouped w.r.t. $C \xrightarrow{g} FC \xrightarrow{F!} F1$
(e.g. final vs. non-final states)

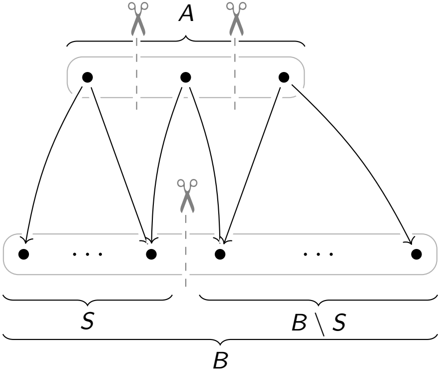
Refinement Step for $\mathcal{P}$



Initially

All states of $g\colon C \to FC$ are grouped w.r.t. $C \xrightarrow{g} FC \xrightarrow{F!} F1$
(e.g. final vs. non-final states)

Refinement Step for $\mathcal{P}$



States $x_1, x_2 \in A$ stay together iff

$$\mathcal{P}\chi_S^B(g(x_1)) = \mathcal{P}\chi_S^B(g(x_2)).$$

$$\chi_S^B\colon C \to 3$$

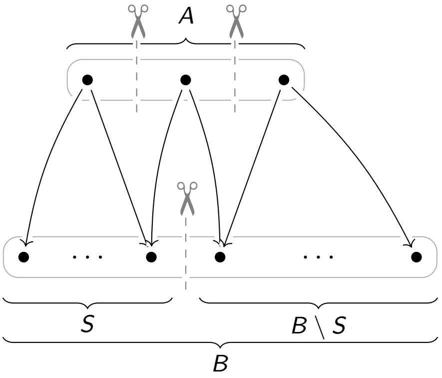
$$\chi_S^B(x) = \begin{cases} 2 & \text{if } x \in S \\ 1 & \text{if } x \in B \setminus S \\ 0 & \text{if } x \notin B \end{cases}$$

$$C \xrightarrow{g} \mathcal{P}C \xrightarrow{\mathcal{P}\chi_S^B} \mathcal{P}3$$

Initially

All states of $g\colon C \to FC$ are grouped w.r.t. $C \xrightarrow{g} FC \xrightarrow{F!} F1$ (e.g. final vs. non-final states)

Refinement Step for F



States $x_1, x_2 \in A$ stay together iff

$$F\chi_S^B(g(x_1)) = F\chi_S^B(g(x_2)).$$

$$\chi_S^B\colon C \to 3$$

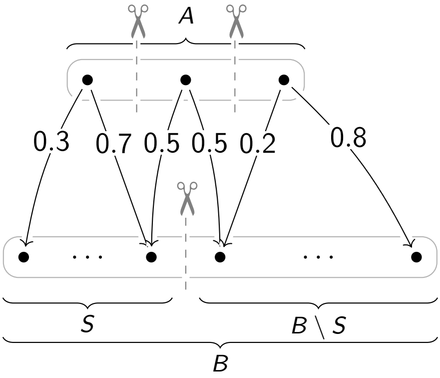
$$\chi_S^B(x) = \begin{cases} 2 & \text{if } x \in S \\ 1 & \text{if } x \in B \setminus S \\ 0 & \text{if } x \notin B \end{cases}$$

$$C \xrightarrow{g} FC \xrightarrow{F\chi_S^B} F3$$

Initially

All states of $g\colon C \to FC$ are grouped w.r.t. $C \xrightarrow{g} FC \xrightarrow{F!} F1$
(e.g. final vs. non-final states)

Refinement Step for F



States $x_1, x_2 \in A$ stay together iff

$$F\chi_S^B(g(x_1)) = F\chi_S^B(g(x_2)).$$

$$\chi_S^B\colon C \to 3$$

$$\chi_S^B(x) = \begin{cases} 2 & \text{if } x \in S \\ 1 & \text{if } x \in B \setminus S \\ 0 & \text{if } x \notin B \end{cases}$$

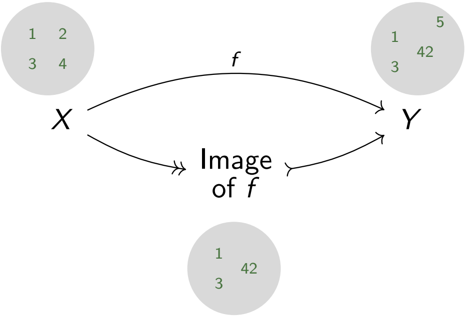
$$C \xrightarrow{g} FC \xrightarrow{F\chi_S^B} F3$$

Factorizations

$$x_1, x_2 \text{ in the same block} \quad :\Longleftrightarrow \quad f(x_1) = f(x_2)$$

Kernel pairs

$$\ker f = \{(x_1, x_2) \in X^2 \mid f(x_1) = f(x_2)\}$$

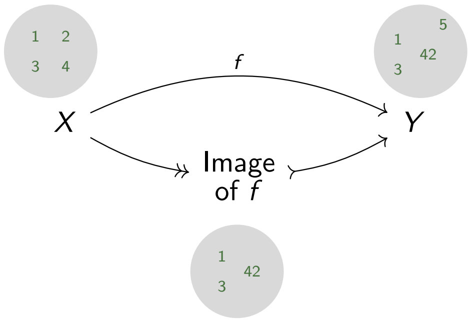


Factorizations

x_1, x_2 in the same block $\quad :\Longleftrightarrow \quad f(x_1) = f(x_2)$

Kernel pairs

$$\ker f = \{(x_1, x_2) \in X^2 \mid f(x_1) = f(x_2)\}$$

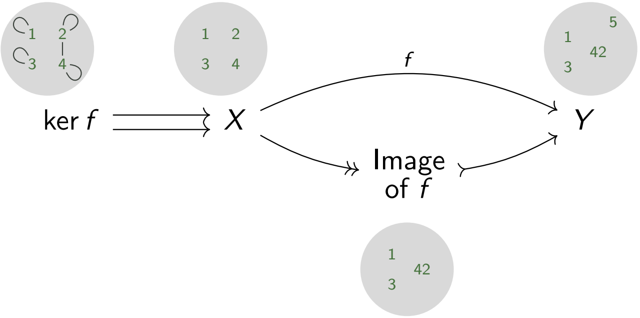


Factorizations

$$x_1, x_2 \text{ in the same block} \quad :\Longleftrightarrow \quad f(x_1) = f(x_2)$$

Kernel pairs

$$\ker f = \{(x_1, x_2) \in X^2 \mid f(x_1) = f(x_2)\}$$

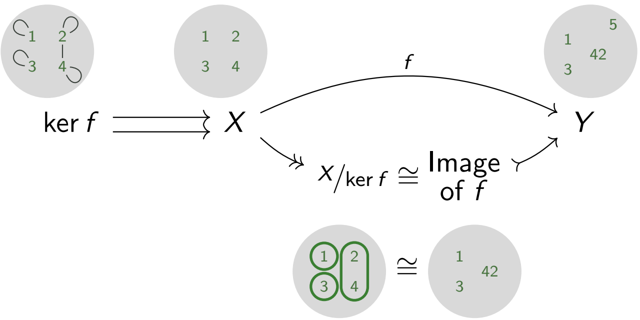


Factorizations

$$x_1, x_2 \text{ in the same block} \quad :\Longleftrightarrow \quad f(x_1) = f(x_2)$$

Kernel pairs

$$\ker f = \{(x_1, x_2) \in X^2 \mid f(x_1) = f(x_2)\}$$



Algorithm for a finite $c\colon C \to FC$

- $C/Q := \{C\}$
- $P := \ker(C \xrightarrow{c} FC \xrightarrow{F!} F1)$
- While P properly finer than Q:
  - Pick $S \subsetneq B$, $\quad S \in C/P$, $\quad B \in C/Q$, $\quad |S| \leq \frac{1}{2} \cdot |B|$
  - $C/Q := C/Q - \{B\} \;\cup\; \{S, B \setminus S\}$
  - $P := P \cap \ker(C \xrightarrow{c} FC \xrightarrow{F\chi_S^B} F3)$
- Return C/P

Correctness

If F is zippable, then the above algorithm computes the simple quotient of $c\colon C \to FC$.

Functor F zippable, if the canonical map

$$F(L+R) \longrightarrow F(L+1) \times F(1+R) \text{ is injective.}$$

E.g. Id, Constants, $\times$, $+$, $\hookrightarrow$, $M^{(-)}$, part. additive

$F(X+Y) \rightarrowtail FX \times FY$

Functor F zippable, if the canonical map

$$F(L+R) \longrightarrow F(L+1) \times F(1+R) \text{ is injective.}$$

E.g. Id, Constants, $\times$, $+$, $\hookrightarrow$, $M^{(-)}$, part. additive $F(X+Y) \rightarrowtail FX \times FY$

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

$$a_1\, a_2\, b_1\, a_3\, b_2 \overset{\text{unzip}}{\longmapsto} (a_1 a_2 \_\, a_3 \_,\ \_\,\_\, b_1\, \_\, b_2)$$

$(-)^5$ is zippable

$$\{a_1, a_2, b_1\} \overset{\text{unzip}}{\longmapsto} (\{a_1, a_2, \_\}, \{\_, b_1\})$$

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

Functor F zippable, if the canonical map

$$F(L+R) \longrightarrow F(L+1) \times F(1+R) \text{ is injective.}$$

E.g. Id, Constants, $\times$, $+$, $\hookrightarrow$, $M^{(-)}$, part. additive $F(X+Y) \rightarrowtail FX \times FY$

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

$$a_1\, a_2\, b_1\, a_3\, b_2 \overset{\text{unzip}}{\longmapsto} (a_1 a_2\, \_\, a_3\, \_,\ \_\,\_\, b_1\, \_\, b_2)$$

$(-)^5$ is zippable

$$\{a_1, a_2, b_1\} \overset{\text{unzip}}{\longmapsto} (\{a_1, a_2, \_\}, \{\_, b_1\})$$

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

$$\{\{a_1, b_1\}, \{a_2, b_2\}\} \overset{\text{unzip}}{\longmapsto} (\{\{a_1, \_\}, \{a_2, \_\}\}, \{\{\_, b_1\}, \{\_, b_2\}\}) \overset{\text{unzip}}{\longmapsfrom} \{\{a_1, b_2\}, \{a_2, b_1\}\}$$

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

~~Composition~~

~~Quotients~~

Algorithm for a finite $c\colon C \to FC$

- $C/Q := \{C\}$
- $P := \ker(C \xrightarrow{c} FC \xrightarrow{F!} F1)$
- While P properly finer than Q:
  - Pick $S \subsetneq B$, $\quad S \in C/P$, $\quad B \in C/Q$, $\quad |S| \leq \frac{1}{2} \cdot |B|$
  - $C/Q := C/Q - \{B\} \;\cup\; \{S, B \setminus S\}$
  - $P := P \cap \ker(C \xrightarrow{c} FC \xrightarrow{F\chi_S^B} F3)$
- Return C/P

Correctness

If F is zippable, then the above algorithm computes the simple quotient of $c\colon C \to FC$.

How to compute $C \xrightarrow{c} FC \xrightarrow{F\chi_S^B} F3$ efficiently?

Functor Encoding

Bags

Labels A $\qquad \flat : FX \to \mathcal{B}(A \times X)$

How to compute $C \xrightarrow{c} FC \xrightarrow{F\chi_S^B} F3$ efficiently?

Functor Encoding

Bags

Labels A $\qquad \flat : FX \to \mathcal{B}(A \times X)$

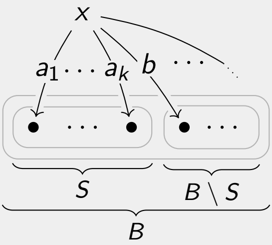
Refinement Interface

Type W (abstract, could be ints, reals, trees, ...)

init: $F1 \times \mathcal{B}A \to W$

update: $\mathcal{B}A \times W \to W \times F3 \times W$

Labels to S $\qquad$ Weight of B



How to compute $C \xrightarrow{c} FC \xrightarrow{F\chi_S^B} F3$ efficiently?

Functor Encoding

Bags

Labels A $\qquad \flat : FX \to \mathcal{B}(A \times X)$

Refinement Interface

Type W (abstract, could be ints, reals, trees, ...)

init: $F1 \times \mathcal{B}A \to W$

update: $\mathcal{B}A \times W \to W \times F3 \times W$

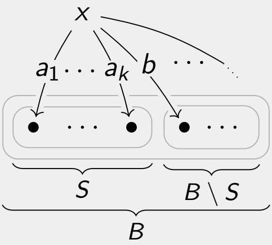
Labels to S Weight of B

Example: $FX = \mathbb{R}^{(X)}$

$$A := \mathbb{R} \qquad W := \mathbb{R} \times \mathbb{R}$$

$$\mathrm{init}(\_, \ell) = (0, \Sigma\ell)$$

$$\mathrm{update}(\ell, (r, b)) = ((r + b - \Sigma\ell, \Sigma\ell), \ldots)$$



Initialization: Partitioning w.r.t. $C \xrightarrow{c} FC \xrightarrow{F!} F1$

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], $\mathcal{B}(\pi_2 \cdot \text{graph})(\text{toSub}[x])$)
  for $e \in$ toSub[x] **do** lastW[e] = p_X
  toSub[x] := $\emptyset$
X/P := group X by type: $X \to F1$.

Refinement Step: Refine by $C \xrightarrow{c} F3 \xrightarrow{F\chi_S^T} F3$

Split(X/P, S)

M := $\emptyset \subseteq X/P \times F3$
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 · 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]

for $(B, v_\emptyset) \in$ M **do**
  $B_{\neq\emptyset} := \emptyset \subseteq X \times F3$
  for (x, p_C) in mark$_B$ **do**
    $\ell := \mathcal{B}(\pi_2 \cdot \text{graph})(\text{toSub}[x])$
    $(w_S^x, v^x, w_{C\setminus S}^x)$:= update(ℓ, 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$_B$:= $\emptyset$
  $B_1 \times \{v_1\}, \ldots, B_\ell \times \{v_\ell\}$:=
   group $B_{\neq\emptyset}$ by $\pi_2\colon X \times F3 \to F3$
  insert $B_1, \ldots, B_\ell$:= into X/P

(a) Collecting predecessor blocks

(b) Splitting predecessor blocks

Efficiency

$F\colon \mathsf{Set} \to \mathsf{Set}$ is zippable
&
F has a refinement interface
(with linear run-time)

$\Longrightarrow$

Minimization runs
in $\mathcal{O}((m+n)\cdot \log n)$

Edges
States

Efficiency

$F\colon \mathsf{Set} \to \mathsf{Set}$ is zippable
&
F has a refinement interface
(with linear run-time)

$\Longrightarrow$

Minimization runs
in $\mathcal{O}((\underbrace{m}_{\text{Edges}} + n) \cdot \log \underbrace{n}_{\text{States}})$

Refinement Interfaces for

- Polynomial Functors Σ
- $G^{(-)}$, G abelian group, e.g. $\mathbb{R}^{(-)}$, finite multisets $\mathcal{B} = \mathbb{N}^{(-)}$
- $\mathcal{P}_\mathrm{f}$ finite powerset
- $M^{(-)}$, M commutative monoid (additional factor $\log \min(|M|, m)$)

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
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
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017

The Tool CoPaR

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

The Tool CoPaR

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

Refinement Interface Type

Math:

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

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)
```

Example: Refinement Interface Implementation for $\mathbb{R}^{(-)}$

Math:

$$\mathrm{init}(f_1, e) = (0, \textstyle\sum e)$$

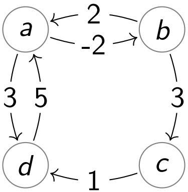
$$\mathrm{update}(e, (r, c)) = ((r + c - \textstyle\sum e, \sum e), (r, c - \sum e, \sum e), (\sum e + r, c - \sum e))$$

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))
```

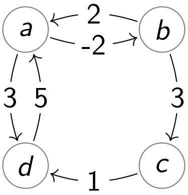
Example: Input coalgebra for $FX = \mathbb{R}^{(X)}$

```
R^(X)

a: { d: 3, b: -2}
b: { a: 2, c: 3}
c: { d: 1 }
d: { a: 5 }
```



Example: Input coalgebra for $FX = \mathbb{R}^{(X)}$

```
R^(X)

a: { d: 3, b: -2}
b: { a: 2, c: 3}
c: { d: 1 }
d: { a: 5 }
```

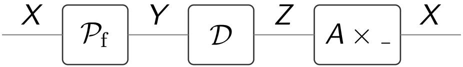


Output

```
Block 0: d, b
Block 1: a, c
```

Modularity: Composed System Types

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



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

$\Longrightarrow \quad H'\colon \mathsf{Set} \to \mathsf{Set} \quad H'X = \mathcal{P}_{\mathrm{f}}X + \mathcal{D}X + A \times X$

Wißmann, Dorsch, Milius, Schröder '20

Modularity: Composed System Types

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

$$X \;—\; \boxed{\mathcal{P}_{\mathrm{f}}} \;—\; Y \;—\; \boxed{\mathcal{D}} \;—\; Z \;—\; \boxed{A \times \_} \;—\; X$$

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

$\Longrightarrow \quad H'\colon \mathsf{Set} \to \mathsf{Set} \quad H'X = \mathcal{P}_{\mathrm{f}} X + \mathcal{D} X + A \times X$

Theorem (for every such F)

Every F-coalgebra can be transformed into a H'-coalgebra, and they have the same simple quotient.

Wißmann, Dorsch, Milius, Schröder '20

Modularity: Composed System Types

$FX = \mathcal{P}_f(\mathcal{D}(A \times X))$

$$X \;—\; \boxed{\mathcal{P}_f} \;—\; Y \;—\; \boxed{\mathcal{D}} \;—\; Z \;—\; \boxed{A \times \_} \;—\; X$$

$$\Longrightarrow \quad H\colon \mathsf{Set}^3 \to \mathsf{Set}^3 \quad H(X, Y, Z) = (\mathcal{P}_f Y, \mathcal{D}Z, A \times X)$$

$$\Longrightarrow \quad H'\colon \mathsf{Set} \to \mathsf{Set} \quad H'X = \mathcal{P}_f X + \mathcal{D}X + A \times X$$

Theorem (for every such F)

Every F-coalgebra can be transformed into a H'-coalgebra, and they have the same simple quotient.

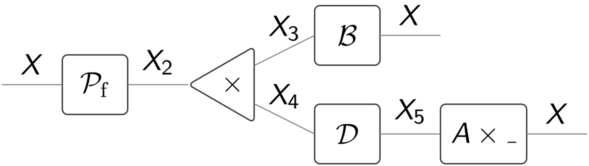
Efficiency

For zippable functors $F_1, \ldots, F_n$ with refinement interfaces one can construct a refinement interface for $F_1 + \ldots + F_n$.

Wißmann, Dorsch, Milius, Schröder '20

Modularity – for more complicated compositions

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



$\Longrightarrow$ $H\colon \mathsf{Set}^5 \to \mathsf{Set}^5$

$$H(X, X_2, X_3, X_4, X_5) = (\mathcal{P}_\mathrm{f}X_2, X_3 \times X_4, \mathcal{B}X, \mathcal{D}X_5, A \times X)$$

$\Longrightarrow$ $H'\colon \mathsf{Set} \to \mathsf{Set}$

$$H'X = \mathcal{P}_\mathrm{f}X + X \times X + \mathcal{B}X, \mathcal{D}X + A \times X$$

Schröder, Pattinson 2011

$$\begin{tikzcd} \mathrm{Coalg}(T) \arrow[d, bend left, "\mathrm{Pad}"] \arrow[d, phantom, "\dashv"] \\ \mathrm{Coalg}(F) \arrow[u, bend left, "\mathrm{Comp}"] \end{tikzcd}$$

$$T\colon \mathsf{Set} \to \mathsf{Set}$$
$$TX = \mathcal{P}(\mathcal{D}(A \times X))$$
$$T = \mathcal{P} \cdot \mathcal{D} \cdot (A \times \_)$$

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

Schröder, Pattinson 2011

Wißmann, Dorsch, Milius, Schröder 2020

$$
\begin{tikzcd}
\mathrm{Coalg}(T) \arrow[d, bend left, "\mathrm{Pad}"] \\
\mathrm{Coalg}(F) \arrow[u, bend left, "\mathrm{Comp}"] \arrow[u, phantom, "\dashv"] \arrow[d, hook, "F\eta\cdot\_"', "\text{preserves behavioural equivalence}"] \\
\mathrm{Coalg}(F\Delta\coprod) \arrow[d, "\bar{\coprod}"', "\cong"] \\
\mathrm{Coalg}(\coprod F\Delta)
\end{tikzcd}
$$

$$T\colon \mathsf{Set} \to \mathsf{Set}$$
$$TX = \mathcal{P}(\mathcal{D}(A\times X))$$
$$T = \mathcal{P}\cdot\mathcal{D}\cdot(A\times \_)$$

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

$$F\Delta\coprod\colon \mathsf{Set}^3 \to \mathsf{Set}^3$$
$$F(X,Y,Z) = (\mathcal{P}(X{+}Y{+}Z), \mathcal{D}(X{+}Y{+}Z), A\times(X{+}Y{+}Z))$$

$$F\Delta\coprod\colon \mathsf{Set}\to\mathsf{Set}$$
$$FX = \mathcal{P}X + \mathcal{D}X + A\times X$$

Assumptions

F mono-preserving, $\mathcal{C}$ extensive, (RegEpi,Mono)-factorization

$\coprod\colon \mathcal{C}^n \to \mathcal{C} \quad\dashv\quad \Delta\colon \mathcal{C}\to\mathcal{C}^n \qquad \eta\colon \mathrm{Id}_{\mathcal{C}^n} \hookrightarrow \Delta\coprod$

In CoPaR

Modularity reduction during preprocessing

- Implemented basic functors: Σ, $\mathcal{P}_\mathrm{f}$, $\mathcal{B}$, $\mathcal{D}$, $\overset{\text{monoid}}{M}{}^{(-)}$,
  for $M = \underbrace{\mathbb{N} \mid \mathbb{Q} \mid \mathbb{Z} \mid \mathbb{R}}_{\text{with } +} \mid (\mathbb{Z}, \max) \mid (\mathbb{R}, \max) \mid (\mathcal{P}_\mathrm{f}(64), \cup)$

Deifel, Milius, Schröder, Wißmann '19

In CoPaR

Modularity reduction during preprocessing

monoid

- Implemented basic functors: Σ, $\mathcal{P}_\mathrm{f}$, $\mathcal{B}$, $\mathcal{D}$, $M^{(-)}$,
  for $M = \underbrace{\mathbb{N} \mid \mathbb{Q} \mid \mathbb{Z} \mid \mathbb{R}}_{\text{with } +} \mid (\mathbb{Z}, \max) \mid (\mathbb{R}, \max) \mid (\mathcal{P}_\mathrm{f}(64), \cup)$
- Interfaces for composed functors are automatically derived:

functor variable

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

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

Deifel, Milius, Schröder, Wißmann '19

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
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
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017

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
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A\rvert \cdot n \cdot \log n$	$=$	$\lvert A\rvert \cdot n \cdot \log n$	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
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$<$	$m \cdot n$	Högberg, Maletti, May 2007
	$M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	Högberg, Maletti, May 2007

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
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A \rvert \cdot n \cdot \log n$	$=$	$\lvert A \rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(A \times \mathcal{D} X)$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	[illegible]	$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			$=$	$m_\mathcal{D} \cdot \log m_{\mathcal{P}_\mathrm{f}}$	Groote, Verduzco, de Vink 2018
Colour Refinement	$\mathcal{B} X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$<$	$m \cdot n$	Högberg, Maletti, May 2007
	$M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	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	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times X)$	$m \cdot \log m$	$=$	$m \cdot \log m$	Dovier, Piazza, Policriti 2004
			$>$	$m \cdot \log n$	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$	$m \cdot \log n$	$=$	[illegible] $\log n$	[illegible]almari, Franceschinis 2010
DFA	$2 \times X^A$ (A fixed)	$n \cdot \log n$	$=$	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A\rvert \cdot n \cdot \log n$	$=$	$\lvert A\rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}X)$	$m_\mathcal{D} \cdot \log n_{\mathcal{P}_\mathrm{f}}$	[illegible]	$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			$=$	$m_\mathcal{D} \cdot$ [illegible]	Groote, Verduzco, de Vink 2018
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$	[illegible]	[illegible]	[illegible]sma, Grohe 2017
Weighted Tree Automata	$M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$<$	[illegible]	[illegible]erg, Maletti, May 2007
	$M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	Högberg, Maletti, May 2007

Generic & Efficient

Articles & Tool
tinyurl.com/coalgebra

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	$\mathcal{P}_{\mathrm{f}}(\mathbb{N} \times$			$m \cdot \log m$	Dovier, Piazza, Policriti 2004
				$\cdot \log n$	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$			$\log n$	almari, Franceschinis 2010
DFA	$2 \times X^A$ (A fixed)	$n \cdot \log n$	$=$	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_{\mathrm{f}}(A \times X)$	$\lvert A \rvert \cdot n \cdot \log n$	$=$	$\lvert A \rvert \cdot n \cdot \log n$	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
			$=$	$m_{\mathcal{D}} \cdot \log$	Groote, Verduzco, de Vink 2018
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$			sma, Grohe 2017
Weighted Tree Automata	$M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$<$		erg, Maletti, May 2007
	$M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	Högberg, Maletti, May 2007

More instances:
further system types
& equivalences

Generic & Efficient

Articles & Tool
tinyurl.com/coalgebra

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	$\mathcal{P}_\mathrm{f}(\mathbb{N} \times$			$m \cdot \log m$	Dovier, Piazza, Policriti 2004
				$\cdot \log n$	Valmari 2009
Markov Chains	$\mathbb{R}^{(X)}$			$\log n$	almari, Franceschinis 2010
DFA	A fixed)	$n \cdot \log n$	$=$		Hopcroft 1971
	$X)$	$\lvert A \rvert \cdot$	$=$	$\lvert A \rvert \cdot$	Gries 1973/Knuutila 2001
Segala Systems	$\mathcal{P}_\mathrm{f}(A \times \mathcal{D}$			$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			$=$	$m_\mathcal{D} \cdot$	Groote, Verduzco, de Vink 2018
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$			sma, Grohe 2017
Weighted Tree Automata	$M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$<$		erg, Maletti, May 2007
	$M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	Högberg, Maletti, May 2007

More instances:
further system types
& equivalences

E.g. Nominal
Automata

Generic & Efficient

Articles & Tool
tinyurl.com/coalgebra

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
	$2 \times \mathcal{P}_\mathrm{f}(A \times X)$	$\lvert A\rvert \cdot n \cdot \log n$	$=$	$\lvert A\rvert \cdot n \cdot \log n$	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
Colour Refinement	$\mathcal{B}X$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata	$M^{(\Sigma X)}$ M non-cancellative	$m \cdot \log^2 m$	$<$	$m \cdot n$	Högberg, Maletti, May 2007
	$M^{(\Sigma X)}$ M cancellative	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	Högberg, Maletti, May 2007

Appendix ...

Functor encoding

- internal weights W, $w : FX \to \mathcal{P}_\mathrm{f} X \to W$
- edge labels L
- $\flat : FX \to \mathcal{B}(L \times X)$
- update : $\mathcal{B}(L) \times W \longrightarrow W \times F(2 \times 2) \times W$

weight of C $\quad$ $F\langle \chi_S, \chi_C \rangle$

edges into S $\quad$ weight of S $\quad$ weight of $C \setminus S$

Functor:	$G^{(-)}$	$\mathcal{B}$	$\mathcal{D}$	$\mathcal{P}_\mathrm{f}$	F_Σ
Labels L:	G	$\mathbb{N}$	$[0,1]$	1	$\mathbb{N}$
Weights W:	$G^{(2)}$	$\mathcal{B}2$	$\mathcal{D}2$	$\mathbb{N}$	$F_\Sigma 2$
$w(C)$, $C \subseteq Y$:	$G\chi_C$	$\mathcal{B}\chi_C$	$\mathcal{D}\chi_C$	$\lvert C \cap (-) \rvert$	$F_\Sigma \chi_C$

1. Assume everything equivalent

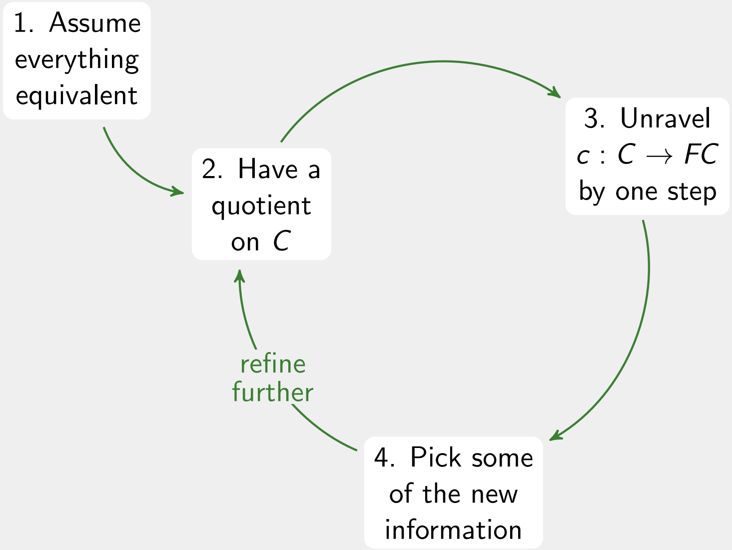
1. Assume everything equivalent

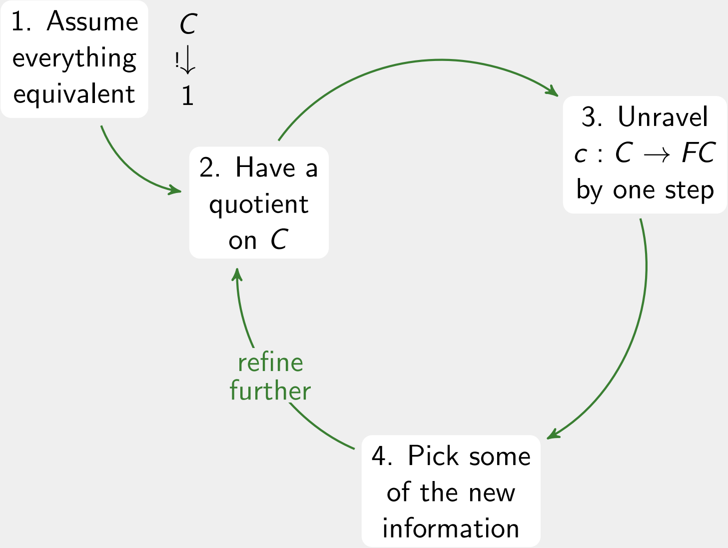
2. Have a quotient on C

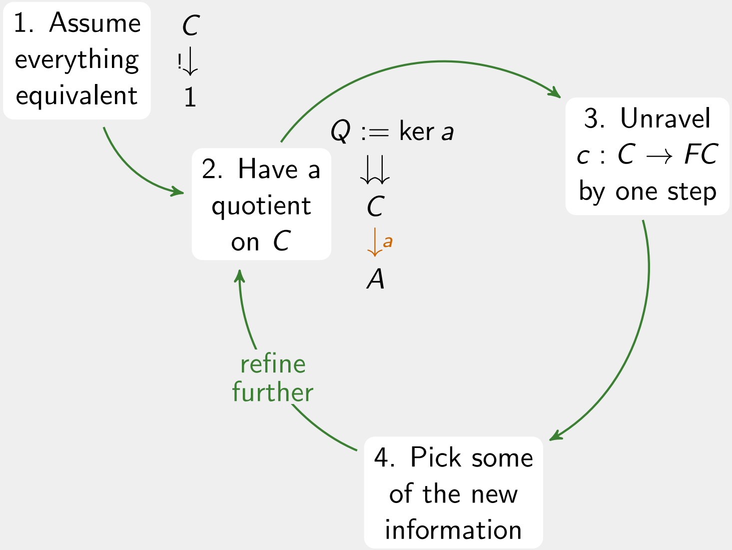
1. Assume everything equivalent

2. Have a quotient on C

3. Unravel $c : C \to FC$ by one step


1. Assume everything equivalent
2. Have a quotient on C
3. Unravel $c : C \to FC$ by one step
4. Pick some of the new information
refine further


1. Assume everything equivalent
$C \xrightarrow{!} 1$
2. Have a quotient on C
3. Unravel $c : C \to FC$ by one step
4. Pick some of the new information
refine further


1. Assume everything equivalent
$C \xrightarrow{!} 1$
2. Have a quotient on C
$Q := \ker a \rightrightarrows C \xrightarrow{a} A$
3. Unravel $c : C \to FC$ by one step
4. Pick some of the new information
refine further

1. Assume everything equivalent

$$C \xrightarrow{!} 1$$

2. Have a quotient on C

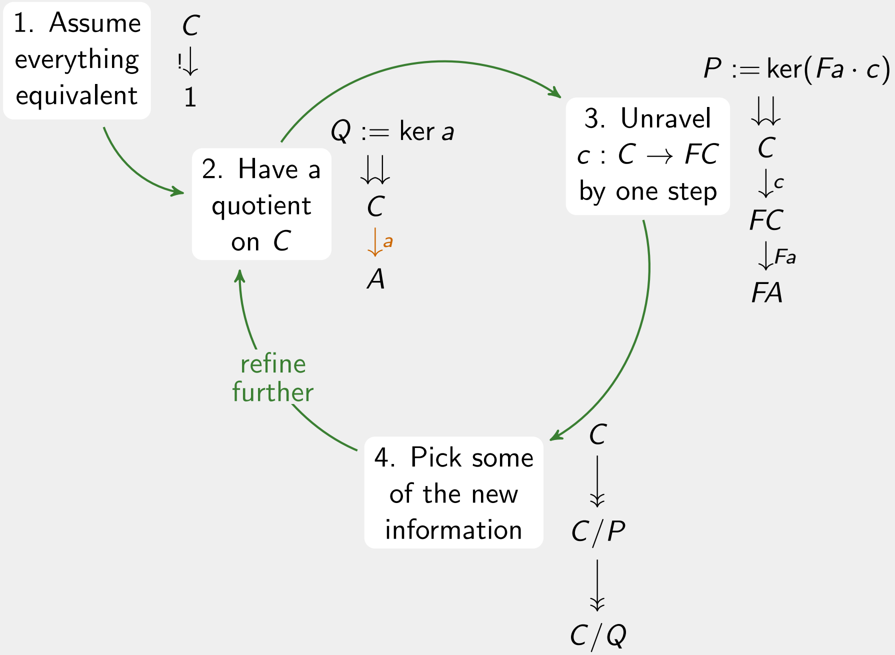
$$Q := \ker a \twoheadrightarrow C \xrightarrow{a} A$$

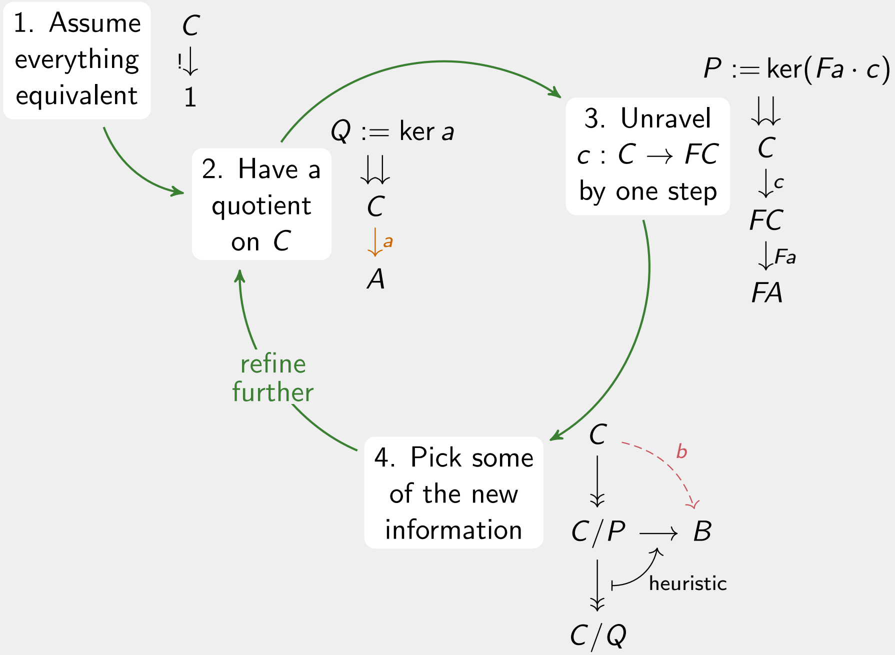
3. Unravel $c : C \to FC$ by one step

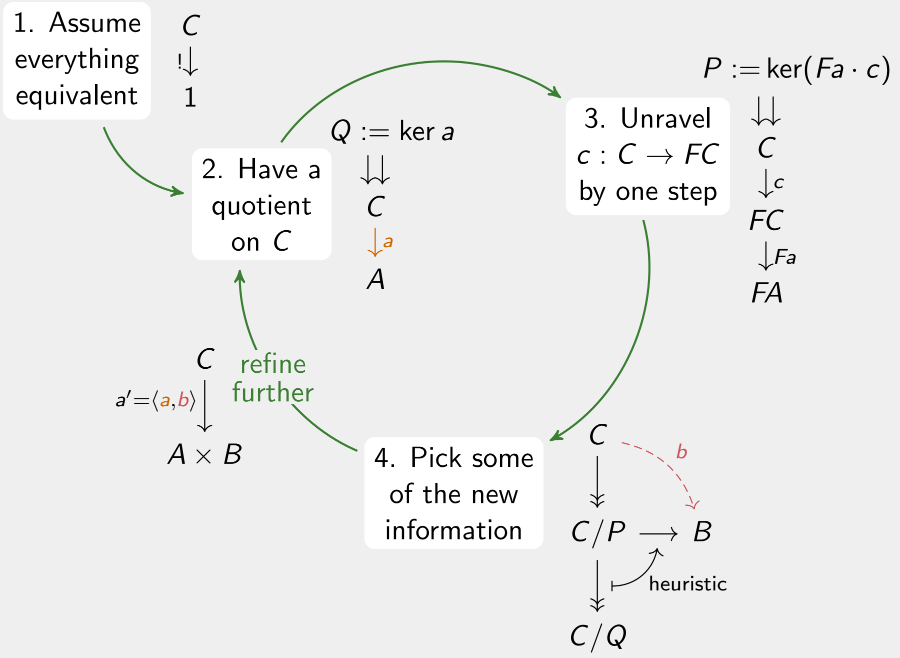
$$P := \ker(Fa \cdot c) \twoheadrightarrow C \xrightarrow{c} FC \xrightarrow{Fa} FA$$

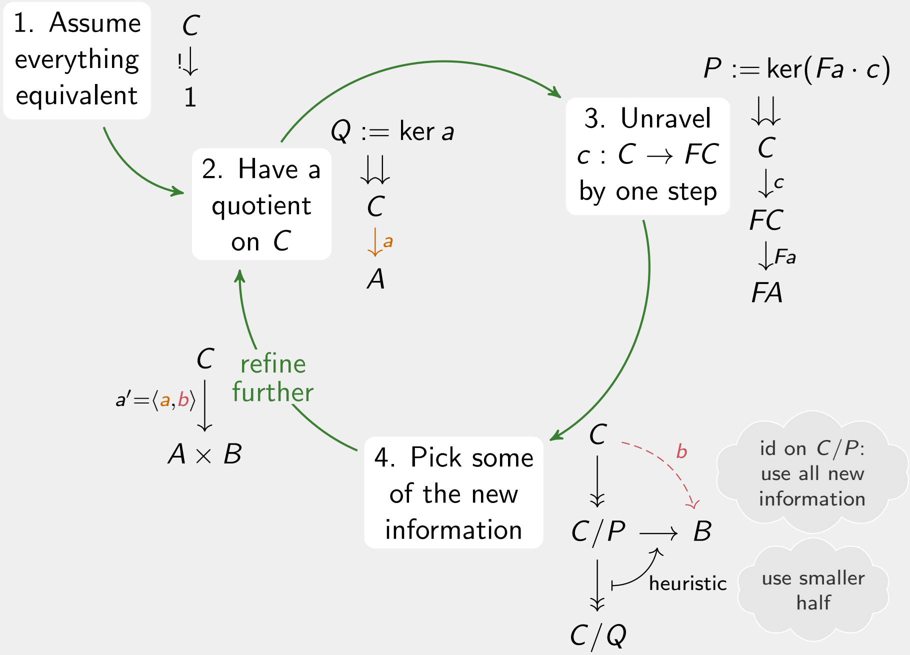
4. Pick some of the new information

refine further


1. Assume everything equivalent
$C \xrightarrow{!} 1$
2. Have a quotient on C
$Q := \ker a \rightrightarrows C \xrightarrow{a} A$
3. Unravel $c : C \to FC$ by one step
$P := \ker(Fa \cdot c) \rightrightarrows C \xrightarrow{c} FC \xrightarrow{Fa} FA$
4. Pick some of the new information
$C \twoheadrightarrow C/P \twoheadrightarrow C/Q$
refine further


1. Assume everything equivalent
$C \xrightarrow{!} 1$
2. Have a quotient on C
$Q := \ker a$
$C \xrightarrow{a} A$
3. Unravel $c : C \to FC$ by one step
$P := \ker(Fa \cdot c)$
$C \xrightarrow{c} FC \xrightarrow{Fa} FA$
4. Pick some of the new information
$C \twoheadrightarrow C/P \twoheadrightarrow C/Q$
$C/P \longrightarrow B$
b
heuristic
refine further


1. Assume everything equivalent
$C \xrightarrow{!} 1$
2. Have a quotient on C
$Q := \ker a \rightrightarrows C \xrightarrow{a} A$
3. Unravel $c : C \to FC$ by one step
$P := \ker(Fa \cdot c) \rightrightarrows C \xrightarrow{c} FC \xrightarrow{Fa} FA$
4. Pick some of the new information
$C \twoheadrightarrow C/P \to B$, $b : C \to B$
$C/P \twoheadrightarrow C/Q$
heuristic
refine further
$C \xrightarrow{a' = \langle a, b \rangle} A \times B$


1. Assume everything equivalent
$C \xrightarrow{!} 1$
2. Have a quotient on C
$Q := \ker a$
$C \xrightarrow{a} A$
3. Unravel $c : C \to FC$ by one step
$P := \ker(Fa \cdot c)$
$C \xrightarrow{c} FC \xrightarrow{Fa} FA$
4. Pick some of the new information
C
b
$C/P \longrightarrow B$
C/Q
heuristic
id on C/P: use all new information
use smaller half
refine further
$a' = \langle a, b \rangle$
$C \to A \times B$

Genericity: Initial partiton

Given

$$C \xrightarrow{c} FC$$

Usual partition refinement algorithms

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

Genericity: Initial partiton

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,

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

Genericity: Composition

If F finitary,

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

Genericity: Composition

If F finitary,

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

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

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

Genericity: Composition

If F finitary,

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

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

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

Examples

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

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

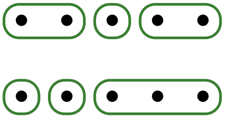
$$A \xleftarrow{a} X \xrightarrow{b} B$$

$\ker a \cup \ker b$ a kernel in Set

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

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

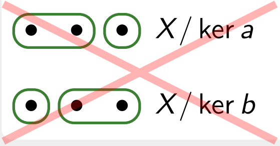
Example



$X/\ker a$

$X/\ker b$

Non-Example



$X/\ker a$

$X/\ker b$

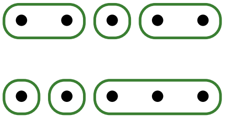
$$A \xleftarrow{a} X \xrightarrow{b} B$$

$\ker a \cup \ker b$ a kernel in Set

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

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

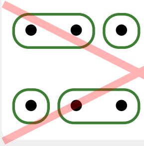
Example



$X/\ker a$

$X/\ker b$

Non-Example



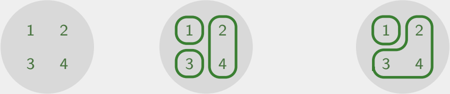
$X/\ker a$

$X/\ker b$

Process smaller half for $X \xrightarrow{f}\!\!\!\!\rightarrow F \xrightarrow{g}\!\!\!\!\rightarrow G$

Find $x \in X$, with $S := [x]_f$, $C := [x]_{gf}$, such that $2 \cdot |S| \leq |C|$.
Return $\langle \chi_S, \chi_C \rangle : X \to 2 \times 2$

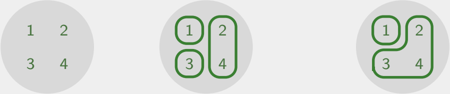
Heuristic



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

heuristic

Heuristic



$$\begin{tikzcd} C \arrow[r, two heads] & C/P \arrow[r, two heads, "p"] \arrow[d] & C/Q \arrow[dl, phantom, "\text{heuristic}"] \\ & B & \end{tikzcd}$$

Use all new information

$B = C/P \rightsquigarrow$ Final Chain algorithm König, Küpper ’14

Heuristic

$$\begin{tikzcd} C \arrow[r, two heads] & C/P \arrow[r, two heads, "p"] \arrow[d, "\text{heuristic}"] & C/Q \\ & B & \end{tikzcd}$$

Use all new information

$B = C/P \rightsquigarrow$ Final Chain algorithm König, Küpper '14

Process the smaller half

Surrounding block in C/Q

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

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

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.

[DMSW19] Hans-Peter Deifel, Stefan Milius, Lutz Schröder, Thorsten Wißmann. "Generic Partition Refinement and Weighted Tree Automata". In: **Formal Methods – The Next 30 Years**. Ed. by Maurice H. ter Beek, Annabelle McIver, José N. Oliveira. Cham: Springer International Publishing, Oct. 2019, pp. 280–297. ISBN: 978-3-030-30942-8. DOI: 10.1007/978-3-030-30942-8_18. URL: https://arxiv.org/abs/1811.08850.

[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.

[KK14] Barbara König, Sebastian Küpper. "Generic Partition Refinement Algorithms for Coalgebras and an Instantiation to Weighted Automata". In: **Theoretical Computer Science, IFIP TCS 2014**. Vol. 8705. LNCS. Springer, 2014, pp. 311–325. ISBN: 978-3-662-44601-0.

[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.

[WDMS20] Thorsten Wißmann, Ulrich Dorsch, Stefan Milius, Lutz Schröder. "Efficient and Modular Coalgebraic Partition Refinement". In: **Logical Methods in Computer Science** Volume 16, Issue 1 (Jan. 2020). DOI: 10.23638/LMCS-16(1:8)2020. URL: https://lmcs.episciences.org/6064.