

Efficient Coalgebraic Partition Refinement

Thorsten Wißmann

Joint work with:

Ulrich Dorsch, Stefan Milius, Lutz Schröder

Published in CONCUR 2017

Friedrich-Alexander-Universität Erlangen-Nürnberg

ERATO MMSD Colloquium

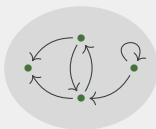
March 15, 2018

Efficient Coalgebraic Partition Refinement

Efficient Coalgebraic Partition Refinement

1. Coalgebras:

State based
systems

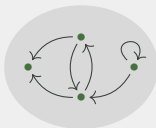


Labels, Non-Determinism,
Probabilities, Automata,
... and their combinations!

Efficient Coalgebraic Partition Refinement

1. Coalgebras:

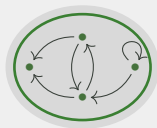
State based
systems



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

2. Partition Refinement:

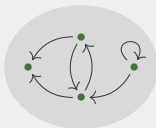
Successively distinguish
different behaviour



Efficient Coalgebraic Partition Refinement

1. Coalgebras:

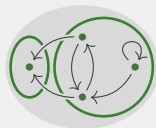
State based
systems



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

2. Partition Refinement:

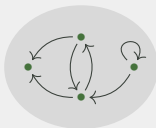
Successively distinguish
different behaviour



Efficient Coalgebraic Partition Refinement

1. Coalgebras:

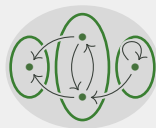
State based
systems



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

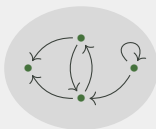
$$\mathcal{O}(m \cdot \log n)$$

Edges

States

1. Coalgebras:

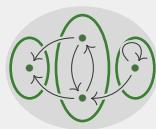
State based
systems



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

Deterministic
Finite Automata

$n \cdot \log n$ $|A| \cdot n \cdot \log n$
Hopcroft '71 Gries '73
 Knuutila '01

Similar Run-Time

Variations in Details

Share Common Structure & Ideas

Deterministic
Finite Automata

$n \cdot \log n$ $|A| \cdot n \cdot \log n$
Hopcroft '71 Gries '73
Knuutila '01

Similar Run-Time

Variations in Details

Transition Systems

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

Share Common Structure & Ideas

Deterministic
Finite Automata

$n \cdot \log n$ $|A| \cdot n \cdot \log n$
Hopcroft '71 Gries '73
Knuutila '01

Similar Run-Time

Variations in Details

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

Deterministic
Finite Automata

$n \cdot \log n$ | $|A| \cdot n \cdot \log n$
Hopcroft '71 | Gries '73
Knuutila '01

Similar Run-Time

Variations in Details

Transition Systems

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

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

Deterministic
Finite Automata

$n \cdot \log n$ | $|A| \cdot n \cdot \log n$
Hopcroft '71 | Gries '73
Knuutila '01

Similar Run-Time

Variations in Details

Transition Systems

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

Segala Systems

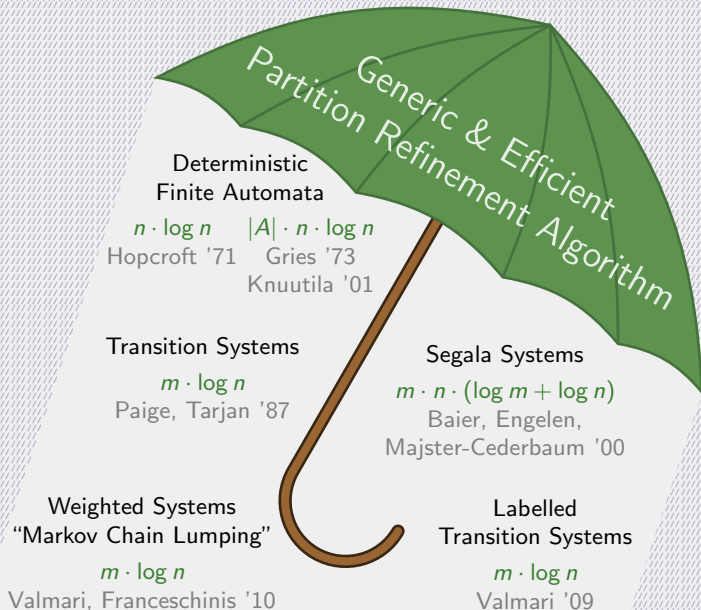
$m \cdot n \cdot (\log m + \log n)$
Baier, Engelen,
Majster-Cederbaum '00

Weighted Systems
"Markov Chain Lumping"

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

Labelled
Transition Systems

$m \cdot \log n$
Valmari '09



Ingredient 1: Factorizations

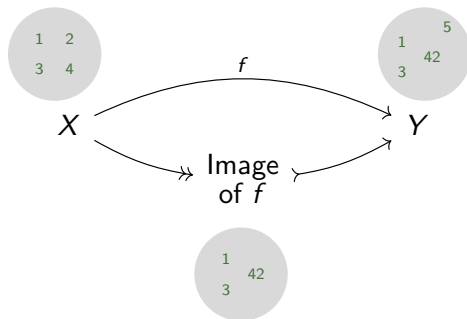
Equivalence Relations	\cong	Quotients \cong Partitions
Kernels	\cong	Regular Epimorphisms

Ingredient 1: Factorizations

Equivalence Relations \cong Quotients \cong Partitions

Kernels \cong Regular Epimorphisms

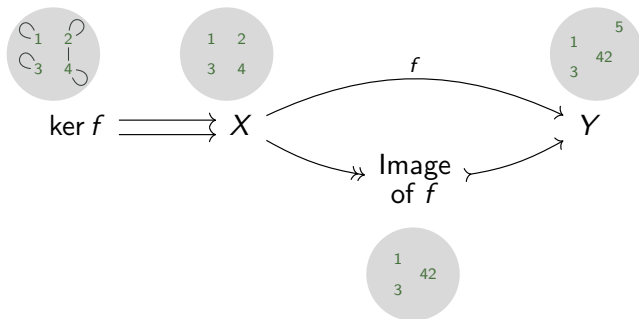
Category with (Regular Epi, Mono)-Factorizations



Ingredient 1: Factorizations

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

Category with (Regular Epi, Mono)-Factorizations

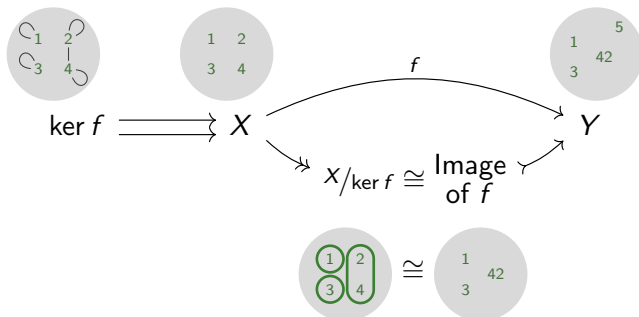


$$\ker f = \{(x_1, x_2) \in X^2 \mid f(x_1) = f(x_2)\}$$

Ingredient 1: Factorizations

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

Category with (Regular Epi, Mono)-Factorizations

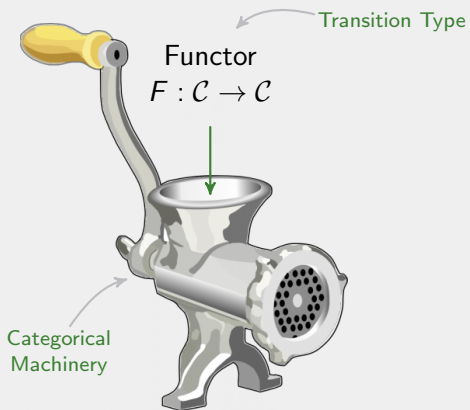


$$\ker f = \{(x_1, x_2) \in X^2 \mid f(x_1) = f(x_2)\}$$

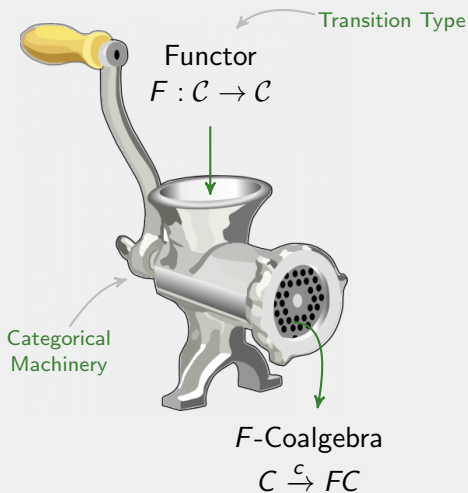
Ingredient 2: Coalgebra – Generic state based systems



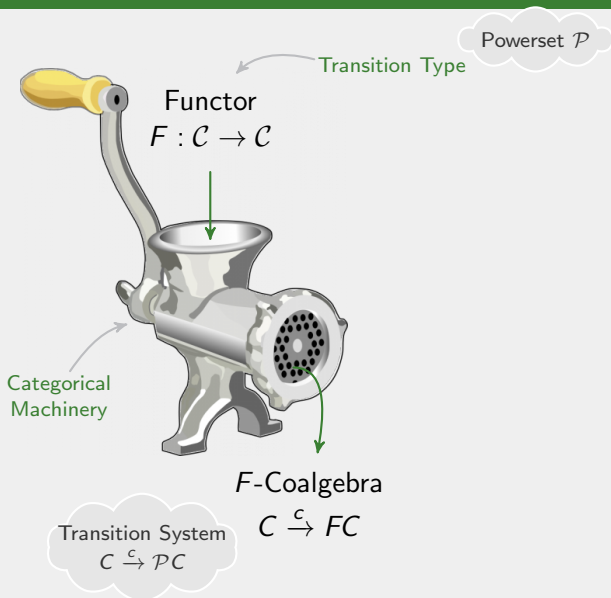
Ingredient 2: Coalgebra – Generic state based systems



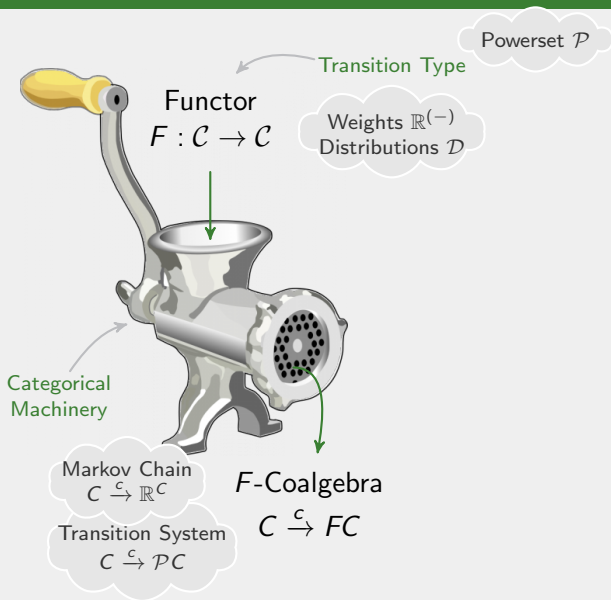
Ingredient 2: Coalgebra – Generic state based systems



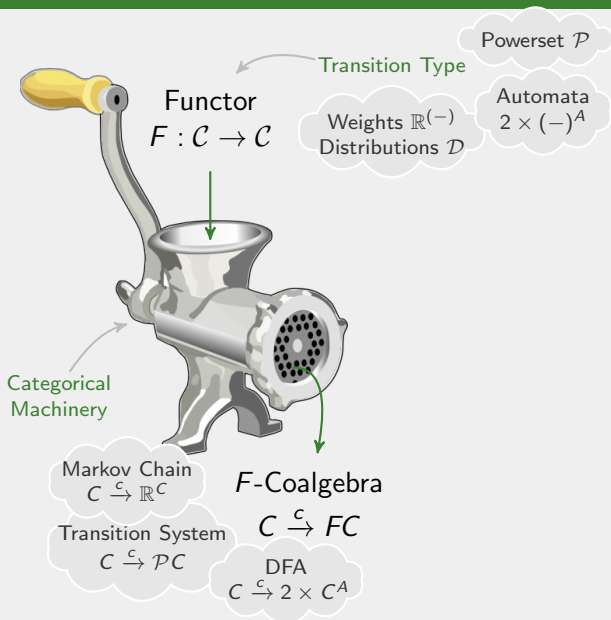
Ingredient 2: Coalgebra – Generic state based systems



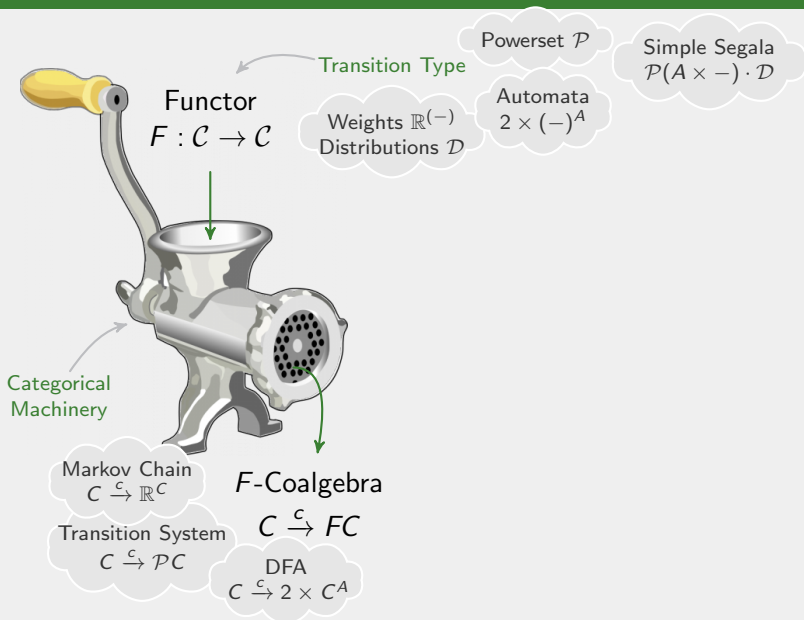
Ingredient 2: Coalgebra – Generic state based systems



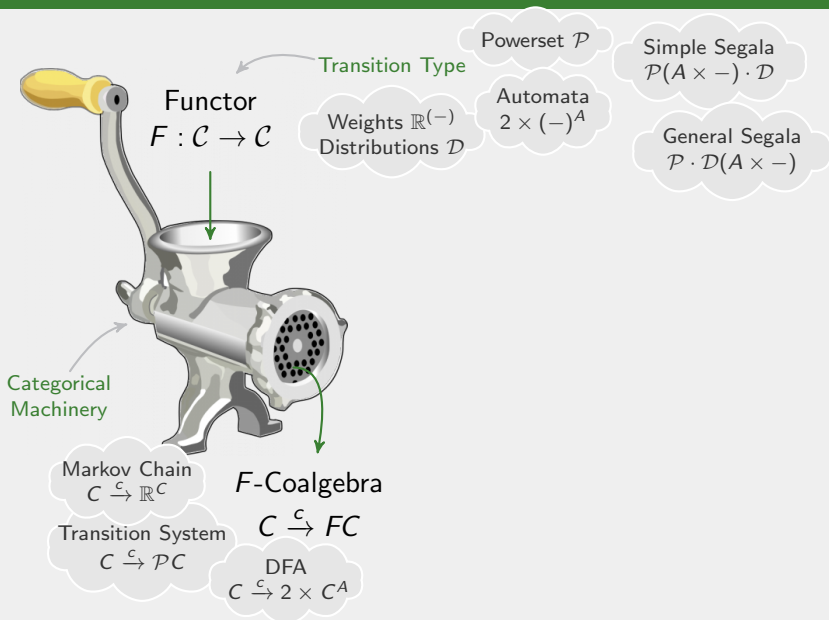
Ingredient 2: Coalgebra – Generic state based systems



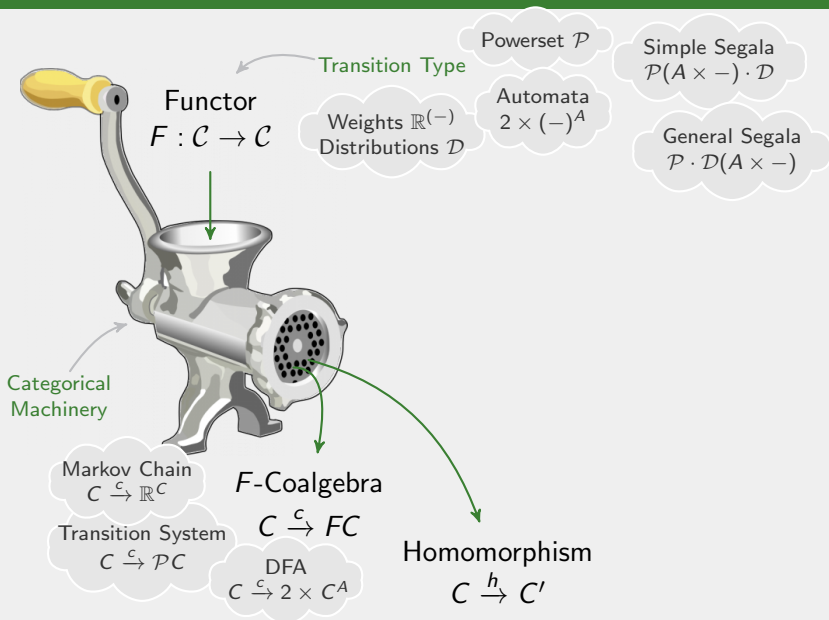
Ingredient 2: Coalgebra – Generic state based systems



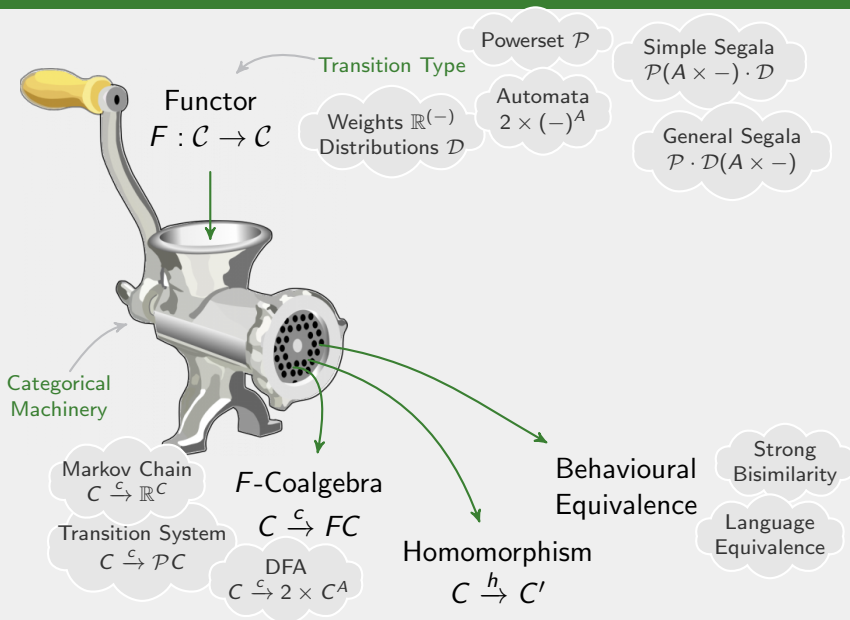
Ingredient 2: Coalgebra – Generic state based systems



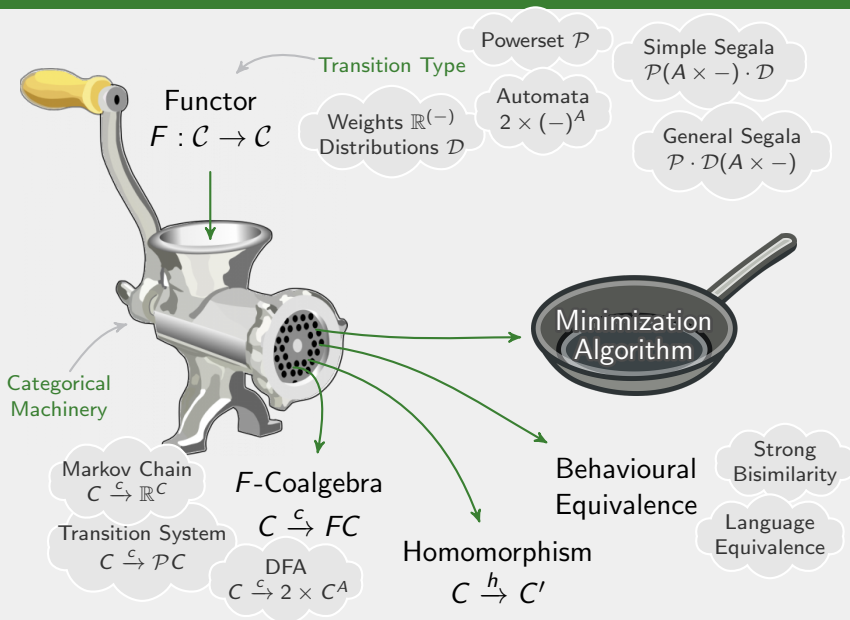
Ingredient 2: Coalgebra – Generic state based systems



Ingredient 2: Coalgebra – Generic state based systems



Ingredient 2: Coalgebra – Generic state based systems



The Coalgebraic Task

For a functor $F : \mathcal{C} \rightarrow \mathcal{C}$

Given a coalgebra

$$\begin{array}{ccc}
 C & \xrightarrow{c} & FC \\
 h \downarrow & & \downarrow Fh \\
 C' & \xrightarrow{c'} & FC'
 \end{array}$$

no proper
quotient

find the simple quotient

all equivalent
behaviours
identified

The Coalgebraic Task

For a functor $F : \mathcal{C} \rightarrow \mathcal{C}$

Given a coalgebra

$$\begin{array}{ccc} C & \xrightarrow{c} & FC \\ h \downarrow & & \downarrow Fh \\ C' & \xrightarrow{c'} & FC' \end{array}$$

no proper
quotient

find the simple quotient

all equivalent
behaviours
identified

Instance

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

Automata
minimization

The Coalgebraic Task

For a functor $F : \mathcal{C} \rightarrow \mathcal{C}$

$$\begin{array}{ccc} \text{Given a coalgebra } C & \xrightarrow{c} & FC \\ h \downarrow & & \downarrow Fh \\ C' & \xrightarrow{c'} & FC' \end{array}$$

no proper
quotient

find the simple quotient

all equivalent
behaviours
identified

Instance

Instance

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

Automata
minimization

For $\mathcal{P} : \text{Set}$

Bisimilarity
minimization

The Coalgebraic Task

For a functor $F : \mathcal{C} \rightarrow \mathcal{C}$

$$\begin{array}{ccc} \text{Given a coalgebra} & C & \xrightarrow{c} & FC \\ & \downarrow h & & \downarrow Fh \\ & C' & \xrightarrow{c'} & FC' \end{array}$$

no proper
quotient

find the simple quotient

all equivalent
behaviours
identified

Instance

Instance

Instance

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

Automata
minimization

For $\mathcal{P} : \text{Set}$

Bisimilarity
minimization

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

Markov chain
lumping

The Coalgebraic Task

For a functor $F : \mathcal{C} \rightarrow \mathcal{C}$

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

$$\begin{array}{ccc} C & \xrightarrow{c} & FC \\ h \downarrow & & \downarrow Fh \\ C' & \xrightarrow{c'} & FC' \end{array}$$

no proper
quotient

find the simple quotient

all equivalent
behaviours
identified

Instance

Instance

Instance

Instance

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

Automata
minimization

For $\mathcal{P} : \text{Set}$

Bisimilarity
minimization

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

Markov chain
lumping

...

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 $c : C \rightarrow FC$ by one step

1. Assume everything equivalent

2. Have a quotient on C

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

4. Pick some of the new information

refine further

```
graph TD; 1[1. Assume everything equivalent] --> 2[2. Have a quotient on C]; 2 --> 3[3. Unravel c : C -> FC by one step]; 3 --> 4[4. Pick some of the new information]; 4 -- refine further --> 2;
```

1. Assume everything equivalent

C
 \Downarrow
1

2. Have a quotient on C

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

4. Pick some of the new information

refine further

```
graph TD; 1[1. Assume everything equivalent] --> 2[2. Have a quotient on C]; 2 --> 3[3. Unravel c : C -> FC by one step]; 3 --> 4[4. Pick some of the new information]; 4 -- refine further --> 2;
```

1. Assume everything equivalent

C
 \Downarrow
 1

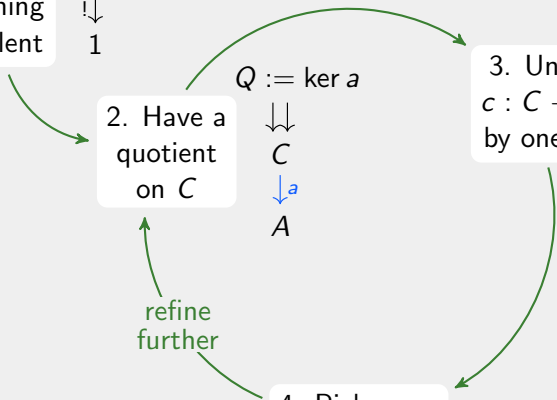
2. Have a quotient on C

$Q := \ker a$
 \Downarrow
 C
 \downarrow^a
 A

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

refine further

4. Pick some of the new information



1. Assume everything equivalent

$$C$$

$$\Downarrow$$

$$1$$

2. Have a quotient on C

$$Q := \ker a$$

$$\Downarrow$$

$$C$$

$$\downarrow^a$$

$$A$$

refine further

4. Pick some of the new information

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

$$P := \ker(Fa \cdot c)$$

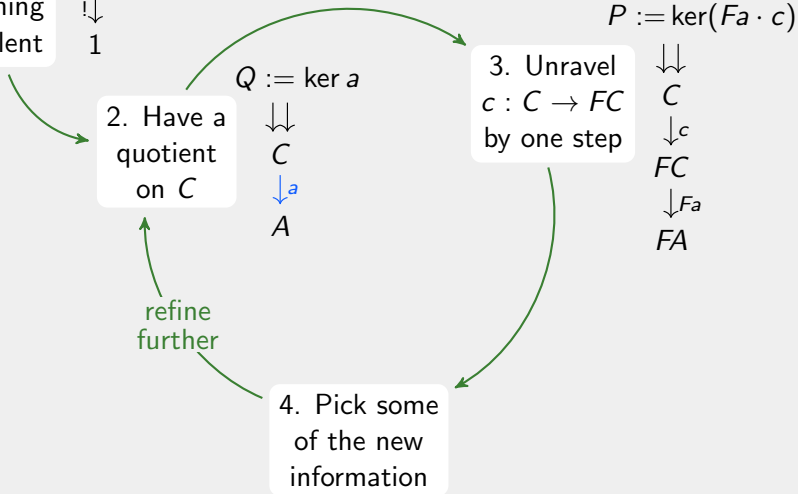
$$\Downarrow$$

$$C$$

$$\downarrow^c$$

$$FC$$

$$\downarrow^{Fa}$$

$$FA$$


1. Assume everything equivalent

$$C \twoheadrightarrow 1$$

2. Have a quotient on C

$$Q := \ker a$$

$$C \twoheadrightarrow A$$

refine further

4. Pick some of the new information

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

$$P := \ker(Fa \cdot c)$$

$$C \twoheadrightarrow FC \xrightarrow{Fa} FA$$

$$C \twoheadrightarrow C/P \twoheadrightarrow C/Q$$

1. Assume everything equivalent

$$C \begin{array}{l} \Downarrow \\ 1 \end{array}$$

2. Have a quotient on C

$$Q := \ker a \begin{array}{l} \Downarrow \\ C \\ \downarrow a \\ A \end{array}$$

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

$$P := \ker(Fa \cdot c) \begin{array}{l} \Downarrow \\ C \\ \downarrow c \\ FC \\ \downarrow Fa \\ FA \end{array}$$

refine further

4. Pick some of the new information

$$\begin{array}{ccc} C & & \\ \downarrow & \dashrightarrow b & \\ C/P & \longrightarrow & B \\ \downarrow & \nearrow \text{heuristic} & \\ C/Q & & \end{array}$$

1. Assume everything equivalent

$$C \begin{array}{l} \Downarrow \\ 1 \end{array}$$

2. Have a quotient on C

$$Q := \ker a \begin{array}{l} \Downarrow \\ C \\ \downarrow a \\ A \end{array}$$

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

$$P := \ker(Fa \cdot c) \begin{array}{l} \Downarrow \\ C \\ \downarrow c \\ FC \\ \downarrow Fa \\ FA \end{array}$$

$a' = \langle a, b \rangle$ $\begin{array}{l} C \\ \downarrow \\ A \times B \end{array}$ refine further

4. Pick some of the new information

$$\begin{array}{ccc} C & & \\ \downarrow & & \text{---} b \text{---} \\ C/P & \longrightarrow & B \\ \downarrow & \nearrow \text{heuristic} & \\ C/Q & & \end{array}$$

1. Assume everything equivalent

$$C \begin{array}{c} \Downarrow \\ \Downarrow \\ 1 \end{array}$$

2. Have a quotient on C

$$Q := \ker a \begin{array}{c} \Downarrow \\ \Downarrow \\ C \\ \downarrow a \\ A \end{array}$$

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

$$P := \ker(Fa \cdot c) \begin{array}{c} \Downarrow \\ \Downarrow \\ C \\ \downarrow c \\ FC \\ \downarrow Fa \\ FA \end{array}$$

$a' = \langle a, b \rangle$ $\begin{array}{c} C \\ \downarrow \\ A \times B \end{array}$ refine further

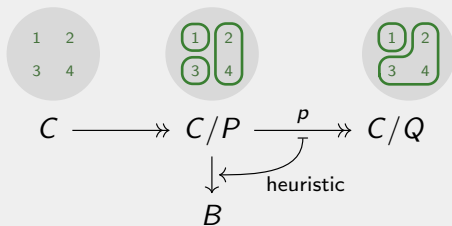
4. Pick some of the new information

$$\begin{array}{c} C \\ \downarrow \\ C/P \rightarrow B \\ \downarrow \text{heuristic} \\ C/Q \end{array}$$

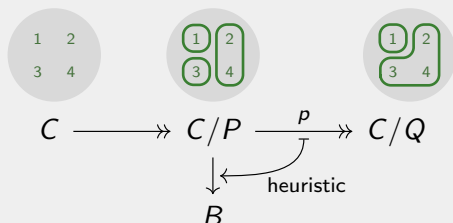
id on C/P :
use all new information

use smaller half

Heuristic



Heuristic

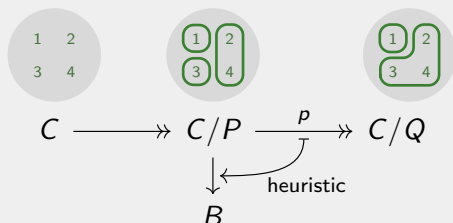


Use all new information

$B = C/P \rightsquigarrow$ Final Chain algorithm

König, Küpper '14

Heuristic



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 $2 \cdot |S| \leq |p(S)|$

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

Assume

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

Theorem (Correctness)

$$\begin{array}{ccc} C & \xrightarrow{c} & FC \\ \downarrow & & \downarrow \\ C/P_i & \longrightarrow & F(C/Q_i) \end{array}$$

Assume

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

Theorem (Correctness)

$$\begin{array}{ccc} C & \xrightarrow{c} & FC \\ \downarrow & & \downarrow \\ C/P_i & \longrightarrow & F(C/Q_i) \end{array}$$

If $P_i \cong Q_i$, then this

1. is a coalgebra
2. has no proper quotient

Efficiency: Incremental Partitions

Incremental partitions

$Q := \ker a$

\Downarrow

C

\downarrow^a

A

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc} Q := \ker a & & Q \cap \ker b \\ \Downarrow & & \Downarrow \\ C & \longrightarrow & C \\ \downarrow a & & \downarrow a' = \langle a, b \rangle \\ A & & A \times B \end{array}$$

Efficiency: Incremental Partitions

Incremental partitions

$$Q := \ker a$$

$$\Downarrow$$

$$C$$

$$\downarrow a$$

$$A$$

$$Q \cap \ker b$$

$$\Downarrow$$

$$C$$

$$\downarrow a' = \langle a, b \rangle$$

$$A \times B$$


$$P := \ker(c \cdot Fa)$$

$$\Downarrow$$

$$C$$

$$\downarrow c$$

$$FC$$

$$\downarrow Fa$$

$$FA$$

Efficiency: Incremental Partitions

Incremental partitions

$$Q := \ker a$$

$$\Downarrow$$

$$C$$

$$\downarrow a$$

$$A$$

$$Q \cap \ker b$$

$$\Downarrow$$

$$C$$

$$\downarrow a' = \langle a, b \rangle$$

$$A \times B$$


$$P := \ker(c \cdot Fa)$$

$$\Downarrow$$

$$C$$

$$\downarrow c$$

$$FC$$

$$\downarrow Fa$$

$$FA$$


$$?? \cap ??$$

$$\Downarrow$$

$$C$$

$$\downarrow c$$

$$FC$$

$$\downarrow F\langle a, b \rangle$$

$$F(A \times B)$$

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & & P := \ker(c \cdot Fa) \\
 \Downarrow & & \Downarrow \\
 C & \xrightarrow{\quad} & C \\
 \downarrow a & & \downarrow c \\
 A & & FA \\
 & & \downarrow Fa \\
 & & FA
 \end{array}
 \qquad
 \begin{array}{ccc}
 & & ?? \cap ?? \\
 & & \Downarrow \\
 & & C \\
 & & \downarrow c \\
 & & FC \\
 & & \downarrow F\langle a, b \rangle \\
 & & F(A \times B)
 \end{array}$$

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

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & & Q \cap \ker b \\
 \Downarrow & & \Downarrow \\
 C & \longrightarrow & C \\
 \downarrow a & & \downarrow a' = \langle a, b \rangle \\
 A & & A \times B
 \end{array}
 \qquad
 \begin{array}{ccc}
 P := \ker(c \cdot Fa) & & \text{??} \cap \text{??} \\
 \Downarrow & & \Downarrow \\
 C & & C \\
 \downarrow c & \longrightarrow & \downarrow c \\
 FC & & FC \\
 \downarrow Fa & & \downarrow F\langle a, b \rangle \\
 FA & & F(A \times B)
 \end{array}$$

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

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & Q \cap \ker b & P := \ker(c \cdot Fa) \\
 \Downarrow & \Downarrow & \Downarrow \\
 C & C & C \\
 \downarrow a & \downarrow a' = \langle a, b \rangle & \downarrow c \\
 A & A \times B & FC \\
 & & \downarrow Fa \\
 & & FA
 \end{array}
 \longrightarrow
 \begin{array}{ccc}
 ?? \cap ?? & & \\
 \Downarrow & & \\
 C & & \\
 \downarrow c & & \\
 FC & & \\
 \downarrow F\langle a, b \rangle & & \\
 F(A \times B) & &
 \end{array}$$

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

$$\begin{array}{c}
 F(L + R) \\
 \downarrow \text{injective and} \\
 F(L+1) \times F(1+R)
 \end{array}$$

“zippable”

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & Q \cap \ker b & P := \ker(c \cdot Fa) \\
 \Downarrow & \Downarrow & \Downarrow \\
 C & C & C \\
 \downarrow a & \downarrow a' = \langle a, b \rangle & \downarrow c \\
 A & A \times B & FC \\
 & & \downarrow Fa \\
 & & FA \\
 & & \longrightarrow \\
 & & \downarrow c \\
 & & FC \\
 & & \downarrow F\langle a, b \rangle \\
 & & F(A \times B)
 \end{array}$$

?? \cap ??

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

$$\begin{array}{c}
 F(L + R) \\
 \downarrow \text{injective and} \\
 F(L+1) \times F(1+R)
 \end{array}$$

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

“zippable”

Functors F zipplable, if

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

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

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

$$\begin{array}{l} a_1 \ a_2 \ b_1 \ a_3 \ b_2 \xrightarrow{\text{unzip}} \\ (a_1 \ a_2 \ - \ a_3 \ -, \\ \ - \ - \ b_1 \ - \ b_2) \end{array}$$

$(-)^*$ is zipplable

$$\begin{array}{l} \{a_1, a_2, b_1\} \xrightarrow{\text{unzip}} \\ (\{a_1, a_2, -\}, \\ \{-, b_1\}) \end{array}$$

\mathcal{P} is zipplable

Functors F zipplable, if

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

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

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

$$\begin{array}{c} a_1 \ a_2 \ b_1 \ a_3 \ b_2 \xrightarrow{\text{unzip}} \\ \left(\begin{array}{l} a_1 \ a_2 \ - \ a_3 \ -, \\ \ - \ - \ b_1 \ - \ b_2 \end{array} \right) \end{array}$$

$(-)^*$ is zipplable

$$\begin{array}{c} \{a_1, a_2, b_1\} \xrightarrow{\text{unzip}} \\ \left(\begin{array}{l} \{a_1, a_2, -\}, \\ \{-, b_1\} \end{array} \right) \end{array}$$

\mathcal{P} is zipplable

$$\{\{a_1, b_1\}, \{a_2, b_2\}\} \quad \{\{a_1, b_2\}, \{a_2, b_1\}\}$$

$$\begin{array}{c} \text{unzip} \left[\begin{array}{c} \{ \{ \{a_1, -\}, \{a_2, -\} \}, \\ \{ \{-, b_1\}, \{-, b_2\} \} \end{array} \right. \\ \left. \right] \text{unzip} \end{array}$$

$\mathcal{P}\mathcal{P}$ is not zipplable

~~Composition~~

~~Quotients~~

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

$\ker a \cup \ker b$ 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



Non-Example



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

$\ker a \cup \ker b$ 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



Non-Example



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

Find $x \in X$, with $S := [x]_f$, $C := [x]_{gf}$, such that $2 \cdot |S| \leq |C|$.

Return $\langle \chi_S, \chi_C \rangle : X \rightarrow 2 \times 2$

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & & Q \cap \ker b \\
 \Downarrow & \longrightarrow & \Downarrow \\
 C & & C \\
 \downarrow a & & \downarrow a' = \langle a, b \rangle \\
 A & & A \times B
 \end{array}
 \qquad
 \begin{array}{ccc}
 P := \ker(c \cdot Fa) & & ?? \cap ?? \\
 \Downarrow & & \Downarrow \\
 C & & C \\
 \downarrow c & \longrightarrow & \downarrow c \\
 FC & & FC \\
 \downarrow Fa & & \downarrow F\langle a, b \rangle \\
 FA & & F(A \times B)
 \end{array}$$

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

$$\begin{array}{c}
 F(L + R) \\
 \downarrow \text{injective and} \\
 F(L+1) \times F(1+R)
 \end{array}$$

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

“zippable”

Efficiency: Incremental Partitions

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & Q \cap \ker b & P := \ker(c \cdot Fa) \quad P \cap \ker(Fb \cdot c) \\
 \Downarrow & \Downarrow & \Downarrow \quad \Downarrow \\
 C & C & C \\
 \downarrow a & \downarrow a' = \langle a, b \rangle & \downarrow c \\
 A & A \times B & FC \\
 & & \downarrow Fa \\
 & & FA \\
 & & \downarrow F \\
 & & F(A \times B)
 \end{array}
 \xrightarrow{\quad}$$

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

$$\begin{array}{c}
 F(L+R) \\
 \downarrow \text{injective and} \\
 F(L+1) \times F(1+R)
 \end{array}$$

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

“zippable”

Efficiency: Incremental Partitions

Concrete
Algorithm?

Incremental partitions

$$\begin{array}{ccc}
 Q := \ker a & Q \cap \ker b & P := \ker(c \cdot Fa) \quad P \cap \ker(Fb \cdot c) \\
 \Downarrow & \Downarrow & \Downarrow \quad \Downarrow \\
 C & C & C \\
 \downarrow a & \downarrow a' = \langle a, b \rangle & \downarrow c \\
 A & A \times B & FC \\
 & & \downarrow Fa \\
 & & FA \\
 & & \downarrow F \\
 & & F(A \times B)
 \end{array}
 \longrightarrow$$

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

$$\begin{array}{c}
 F(L+R) \\
 \downarrow \text{injective} \\
 F(L+1) \times F(1+R)
 \end{array}
 \text{ and}$$

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

“zippable”

Setting for complexity analysis

Category:

Set

Heuristic:

smaller half

Functor:

zippable &
encoding

Setting for complexity analysis

Category:

Set

Heuristic:

smaller half

Functor:

zippable &
encoding

Assumption: Functor encoding

- Refinement interface $\{ \text{type } W, \text{type } L, \text{init}(), \text{w}(), \text{update}() \}$
- coalgebra structure as edges with labels

$$C \xrightarrow{c} FC \xrightarrow{b} \mathcal{P}(L \times C)$$

\Rightarrow compute “smaller half” intersections in linear time

Functor encoding

- type of weights W , type of edge labels L
- weight function $w : \mathcal{P}X \times FX \rightarrow W$
- $b : FX \rightarrow \mathcal{B}_f(L \times X)$
- $\text{init} : F1 \times \mathcal{B}_f(L) \rightarrow W$
- $\text{update} : \mathcal{B}_f(L) \times W \rightarrow W \times F(2 \times 2) \times W$

edges into S weight of C weight of S $F\langle \chi_S, \chi_C \rangle$ weight of $C \setminus S$

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

Setting for complexity analysis

Category:

Set

Heuristic:

smaller half

Functor:

zippable &
encoding

Assumption: Functor encoding

- Refinement interface $\{ \text{type } W, \text{type } L, \text{init}(), \text{w}(), \text{update}() \}$
- coalgebra structure as edges with labels

$$C \xrightarrow{c} FC \xrightarrow{b} \mathcal{P}(L \times C)$$

\Rightarrow compute “smaller half” intersections in linear time

Setting for complexity analysis

Category:

Set

Heuristic:

smaller half

Functor:

zippable &
encoding

Assumption: Functor encoding

- Refinement interface $\{ \text{type } W, \text{type } L, \text{init}(), \text{w}(), \text{update}() \}$
- coalgebra structure as edges with labels

$$C \xrightarrow{c} FC \xrightarrow{b} \mathcal{P}(L \times C)$$

\Rightarrow compute “smaller half” intersections in linear time

Theorem

Overall complexity: $\mathcal{O}((m+n) \cdot \log n)$ for $n = |C|$, $m = \sum_{x \in C} |bc(x)|$

System	Functor	Concrete algorithm		Our instantiation
Transition Systems	\mathcal{P}	$(m + n) \cdot \log n$ Paige, Tarjan '87	=	$(m + n) \cdot \log n$
LTS	$\mathcal{P}(A \times -)$	$(m + n) \cdot \log(m + n)$ Dovier, Piazza, Policriti '04	=	$(m + n) \cdot \log(m + n)$
		$(m + n) \cdot \log m$ Valmari '09	<	
Markov Chains	$\mathbb{R}(-)$	$(m + n) \cdot \log n$ Valmari, Franceschinis '10	=	$(m + n) \cdot \log n$
DFA	$2 \times (-)^A$	$n \cdot \log n$ for fixed A , Hopcroft '71	=	$n \cdot \log n$
	$2 \times \mathcal{P}(A \times -)$	$ A \cdot n \cdot \log n$, Gries '73, Knuutila '01	\approx	$ A \cdot n \cdot \log n$ $+ A \cdot n \cdot \log A $
Segala Systems	$\mathcal{P}(A \times -) \cdot \mathcal{D}$	$m \cdot n \cdot \log(m \cdot n)$ Baier, Engelen, Majster-Cederbaum '00	>	$(m + n) \cdot \log(m + n)$

System	Functor	Concrete algorithm	Our instantiation
Transition Systems	\mathcal{P}	$(m+n) \cdot \log n$ Paige, Tarjan '87	$(m+n) \cdot \log n$
LTS	$\mathcal{P}(A \times -)$	$(m+n) \cdot \log(m+n)$ Dovier, Piazza, Proietti '04 $(m+n) \cdot \log n$ Mimari '02	$(m+n) \cdot \log(m+n)$
Markov Chains	$\mathbb{R}(-)$	$(m+n) \cdot \log n$ Valmari, Franceschini '10	$(m+n) \cdot \log n$
DFA	$2 \times (-)$	$n \cdot \log n$ for fixed A , Hopcroft '71	$n \cdot \log n$
	$2 \times \mathcal{P}(A \times -)$	$ A \cdot n \cdot \log n$, Gries '73, Knuutila '01	$ A \cdot n \cdot \log n$ $+ A \cdot n \cdot \log A $
Segala Systems	$\mathcal{P}(A \times -) \cdot \mathcal{D}$	$m \cdot n \cdot \log(m \cdot n)$ Baier, Engelen, Majster-Cederbaum '00	$(m+n) \cdot \log(m+n)$

Generic & Efficient

System	Functor	Algorithm	Our instantiation
Transition Systems	\mathcal{P}		$(m+n) \cdot \log n$
LTS	$\mathcal{P}(A \times -)$	Dovier, Piazza, Proc. Criti '04 $(m+n) \cdot \log n$ Mimari '02	$(m+n) \cdot \log(m+n)$
Markov Chains	$\mathbb{R}(-)$	Valmari, Franceschini '10 $(m+n) \cdot \log n$	$(m+n) \cdot \log n$
DFA	$2 \times (-)$	$n \cdot \log n$ for fixed A , Hopcroft '71	$n \cdot \log n$
	$2 \times \mathcal{P}(A \times -)$	$ A \cdot n \cdot \log n$, Gries '73, Knuutila '01	$\approx A \cdot n \cdot \log n + A \cdot n \cdot \log A $
Segala Systems	$\mathcal{P}(A \times -) \cdot \mathcal{D}$	$m \cdot n \cdot \log(m \cdot n)$ Baier, Engelen, Majster-Cederbaum '00	$> (m+n) \cdot \log(m+n)$

More instances:
further system types
& categories

Generic & Efficient

System	Functor	Algorithm	Our instantiation
Transition Systems	\mathcal{P}	$(m+n) \cdot \log n$ =	$(m+n) \cdot \log n$
LTS	$\mathcal{P}(A \times -)$	$(m+n) \cdot \log n$ Dovier, Piazza, Proc. Criti '04 =	$(m+n) \cdot \log(m+n)$
Mark Chains	$\mathcal{P}(A \times -)$	$(m+n) \cdot \log n$ Valmari, Franceschini '10 =	$(m+n) \cdot \log n$
DFA	$2 \times (-)$	$n \cdot \log n$ for fixed A , Hopcroft '71 =	$n \cdot \log n$
	$2 \times \mathcal{P}(A \times -)$	$ A \cdot n \cdot \log n$, Gries '73, Knuutila '01 \approx	$ A \cdot n \cdot \log n$ $+ A \cdot n \cdot \log A $
Segala Systems	$\mathcal{P}(A \times -) \cdot \mathcal{D}$	$m \cdot n \cdot \log(m \cdot n)$ Baier, Engelen, Majster-Cederbaum '00 >	$(m+n) \cdot \log(m+n)$

More instances:
further system types
& categories

Implementation:
functor & system
as the input

Generic & Efficient

System	Functor	Algorithm	Our instantiation
Transition Systems	\mathcal{P}	Dovier, Piazza, Proc. Criti '04	$(m+n) \cdot \log n$
LTS	$\mathcal{P}(A \times -)$	Valmari '02	$(m+n) \cdot \log(m+n)$
Markov Chains	$\mathcal{P}(A \times -)$	Valmari, Franceschini '10	$(m+n) \cdot \log n$
DFA	$2 \times (-)$	Holzer '04	$n \cdot \log n$
	$2 \times \mathcal{P}(A \times -)$	Gries '04	$ A \cdot n \cdot \log n$ $+ A \cdot n \cdot \log A $
Segala Systems	$\mathcal{P}(A \times -) \cdot \mathcal{D}$	Baier, Engelen, Majster-Cederbaum '00	$(m+n) \cdot \log(m+n)$

More instances:
further system types
& categories

Implementation:
functor & system
as the input

Generic & Efficient

Compare to
existing concrete
implementations

System	Functor	Concrete algorithm		Our instantiation
Transition Systems	\mathcal{P}	$(m + n) \cdot \log n$ Paige, Tarjan '87	=	$(m + n) \cdot \log n$
LTS	$\mathcal{P}(A \times -)$	$(m + n) \cdot \log(m + n)$ Dovier, Piazza, Policriti '04	=	$(m + n) \cdot \log(m + n)$
		$(m + n) \cdot \log m$ Valmari '09	<	
Markov Chains	$\mathbb{R}(-)$	$(m + n) \cdot \log n$ Valmari, Franceschinis '10	=	$(m + n) \cdot \log n$
DFA	$2 \times (-)^A$	$n \cdot \log n$ for fixed A , Hopcroft '71	=	$n \cdot \log n$
	$2 \times \mathcal{P}(A \times -)$	$ A \cdot n \cdot \log n$, Gries '73, Knuutila '01	\approx	$ A \cdot n \cdot \log n$ $+ A \cdot n \cdot \log A $
Segala Systems	$\mathcal{P}(A \times -) \cdot \mathcal{D}$	$m \cdot n \cdot \log(m \cdot n)$ Baier, Engelen, Majster-Cederbaum '00	>	$(m + n) \cdot \log(m + n)$

Appendix ...

Genericity: Initial partition

Given

$$C \xrightarrow{c} FC$$

Usual partition refinement algorithms

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

Genericity: Initial partiton

Given

$$C \xrightarrow{c} FC$$

Usual partition refinement algorithms

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



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,

$$C \xrightarrow{c} FG C \quad \rightsquigarrow \quad D \xrightarrow{d} GC$$

Genericity: Composition

If F finitary,

$$\begin{array}{ccc} C & \xrightarrow{c} & FG C \\ & \searrow^{c'} & \uparrow Fd \\ & & FD \end{array} \quad \rightsquigarrow \quad D \xrightarrow{d} GC$$

Genericity: Composition

If F finitary,

$$\begin{array}{ccc}
 C & \xrightarrow{c} & FG C \\
 & \searrow^{c'} & \uparrow Fd \\
 & & FD
 \end{array}
 \quad \rightsquigarrow \quad
 D \xrightarrow{d} GC$$

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,

$$\begin{array}{ccc}
 C & \xrightarrow{c} & FG C \\
 & \searrow^{c'} & \uparrow Fd \\
 & & FD
 \end{array}
 \quad \rightsquigarrow \quad
 D \xrightarrow{d} GC$$

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

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

Examples

$$\begin{array}{ll}
 \mathcal{P} \cdot (A \times (-)) & (2 \times \mathcal{P}) \cdot (A \times (-)) \\
 \mathcal{P} \cdot (A \times (-)) \cdot \mathcal{D} & \mathcal{P} \cdot \mathcal{D} \cdot (A \times (-)) \quad \dots
 \end{array}$$

References

- [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.
- [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, Robert Tarjan. “Three partition refinement algorithms”. In: **SIAM J. Comput.** 16.6 (1987), pp. 973–989.
- [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.