Generic Partition Refinement and Weighted Tree Automata

Thorsten Wißmann `thorsten-wissmann.de`

FM'19 Best Theory Paper, coauthored by:
Hans-Peter Deifel, Stefan Milius, Lutz Schröder

Friedrich-Alexander-Universität
Erlangen-Nürnberg, Germany

ICSE Showcase 2023
May 19, 2023, Melbourne, Australia

Generic Partition Refinement and Weighted Tree Automata

Thorsten Wißmann thorsten-wissmann.de

FM'19 Best Theory Paper, coauthored by:
Hans-Peter Deifel, Stefan Milius, Lutz Schröder

Friedrich-Alexander-Universität
Erlangen-Nürnberg, Germany

ICSE Showcase 2023
May 19, 2023, Melbourne, Australia

Generic Partition Refinement and Weighted Tree Automata

Generic Partition Refinement
and

and

Weighted Tree Automata

Generic Partition Refinement

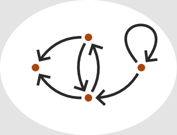
and

Weighted Tree Automata

Generic Partition Refinement

and

Algorithm for
state equivalence:

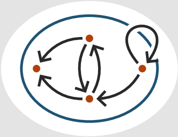


Weighted Tree Automata

Generic Partition Refinement

and

Algorithm for
state equivalence:



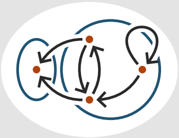
and

Weighted Tree Automata

Generic Partition Refinement

and

Algorithm for
state equivalence:

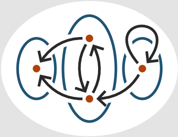


Weighted Tree Automata

Generic Partition Refinement

and

Algorithm for
state equivalence:



Weighted Tree Automata

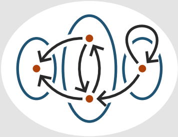
Generic Partition Refinement

and

- Deterministic Automata
- LTS/Bisimilarity
- Markov Chains

⋮

Algorithm for state equivalence:



and

Weighted Tree Automata

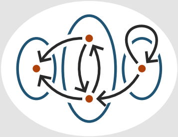
Generic Partition Refinement

and

- Deterministic Automata
- LTS/Bisimilarity
- Markov Chains

⋮

Algorithm for state equivalence:



and

Weighted Tree Automata

Application: Equivalence Checking

Application in Model Checking

Similar Ideas

Similar Run-Time

System type hard-wired

Similar Ideas

Similar Run-Time

Application: Equivalence Checking

Deterministic Finite Automata

Hopcroft '71 Gries '73 Knuutila '01

System type hard-wired

Application in Model Checking

Similar Ideas

Similar Run-Time

System type hard-wired

Application: Equivalence Checking

Deterministic Finite Automata

Hopcroft '71 Gries '73 Knuutila '01

Application in Model Checking

(Labelled) Transition Systems

Paige, Tarjan '87 Valmari '09

Similar Ideas

Similar Run-Time

System type hard-wired

Application: Equivalence Checking

Deterministic Finite Automata

Hopcroft '71 Gries '73 Knuutila '01

Application in Model Checking

(Labelled) Transition Systems

Paige, Tarjan '87

Valmari '09

Weighted Systems

"Markov Chain Lumping"

Valmari, Franceschinis '10

Similar Ideas

Similar Run-Time

System type hard-wired

Application: Equivalence Checking

Deterministic Finite Automata
Hopcroft '71 Gries '73 Knuutila '01

Application in Model Checking

(Labelled) Transition Systems
Paige, Tarjan '87 Valmari '09

Markov Decision Processes
Groote, Verduzco, de Vink '18

Weighted Systems
"Markov Chain Lumping"
Valmari, Franceschinis '10

Similar Ideas

Similar Run-Time

System type hard-wired

Application: Equivalence Checking

Deterministic Finite Automata
Hopcroft '71 Gries '73 Knuutila '01

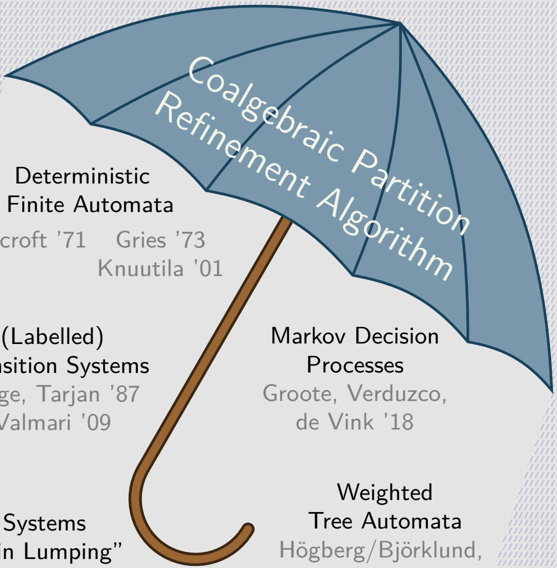
Application in Model Checking

(Labelled) Transition Systems
Paige, Tarjan '87 Valmari '09

Markov Decision Processes
Groote, Verduzco, de Vink '18

Weighted Systems "Markov Chain Lumping"
Valmari, Franceschinis '10

Weighted Tree Automata
Högberg/Björklund, Maletti, May '07



Application: Equivalence Checking

Application in Model Checking

Deterministic Finite Automata
Hopcroft '71 Gries '73
Knuutila '01

(Labelled) Transition Systems
Paige, Tarjan '87
Valmari '09

Markov Decision Processes
Groote, Verduzco, de Vink '18

Weighted Systems
"Markov Chain Lumping"
Valmari, Franceschinis '10

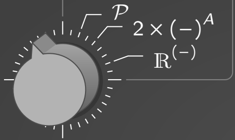
Weighted Tree Automata
Högberg/Björklund, Maletti, May '07

a category

state space

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

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



F-Coalgebra

$$c\colon C \to FC$$

Example

F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

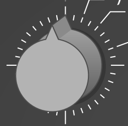
Induced Equivalence Notion:

a category

state space

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

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



F-Coalgebra

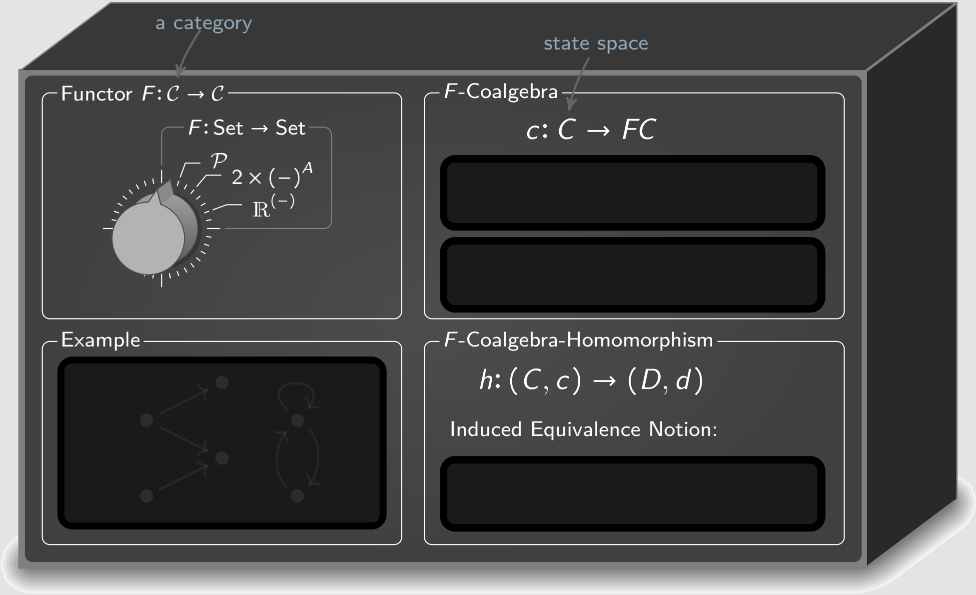
$$c\colon C \to FC$$

Example

F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

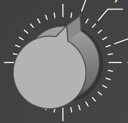
Induced Equivalence Notion:


a category
state space
Functor $F\colon \mathcal{C} \to \mathcal{C}$
$F\colon \mathsf{Set} \to \mathsf{Set}$
$\mathcal{P}$
$2 \times (-)^A$
$\mathbb{R}^{(-)}$
F-Coalgebra
$c\colon C \to FC$
Example
F-Coalgebra-Homomorphism
$h\colon (C,c) \to (D,d)$
Induced Equivalence Notion:

a category

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

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



- $\mathcal{P}$
- $2 \times (-)^A$
- $\mathbb{R}^{(-)}$

state space

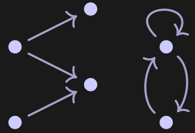
F-Coalgebra

$$c\colon C \to FC$$

$$c\colon C \to \mathcal{P}C$$

"Transition System"

Example



F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

Induced Equivalence Notion:

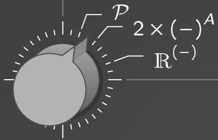
"Strong Bisimilarity"

a category

state space

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

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



F-Coalgebra

$$c\colon C \to FC$$

Example

F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

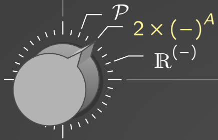
Induced Equivalence Notion:

a category

state space

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

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



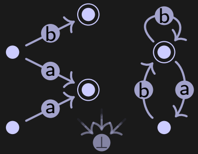
F-Coalgebra

$$c\colon C \to FC$$

$$c\colon C \to 2 \times C^A$$

"Deterministic automaton"

Example



F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

Induced Equivalence Notion:

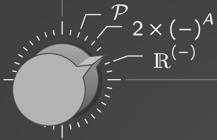
"Same Language"

a category

state space

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

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



F-Coalgebra

$$c\colon C \to FC$$

Example

F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

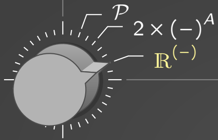
Induced Equivalence Notion:

a category

state space

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

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



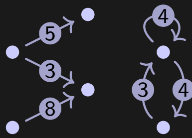
F-Coalgebra

$$c\colon C \to FC$$

$$c\colon C \to \mathbb{R}^{(C)}$$

"Markov Chain"

Example



F-Coalgebra-Homomorphism

$$h\colon (C, c) \to (D, d)$$

Induced Equivalence Notion:

"Weighted Bisimilarity"

Instances of
F-Coalgebras

Transition Systems

Deterministic Automata

Markov Chains

Coalgebraic
Results

Coinduction Principle

Logics

Constructions

Instances of F-Coalgebras

Coalgebraic Results

Transition Systems

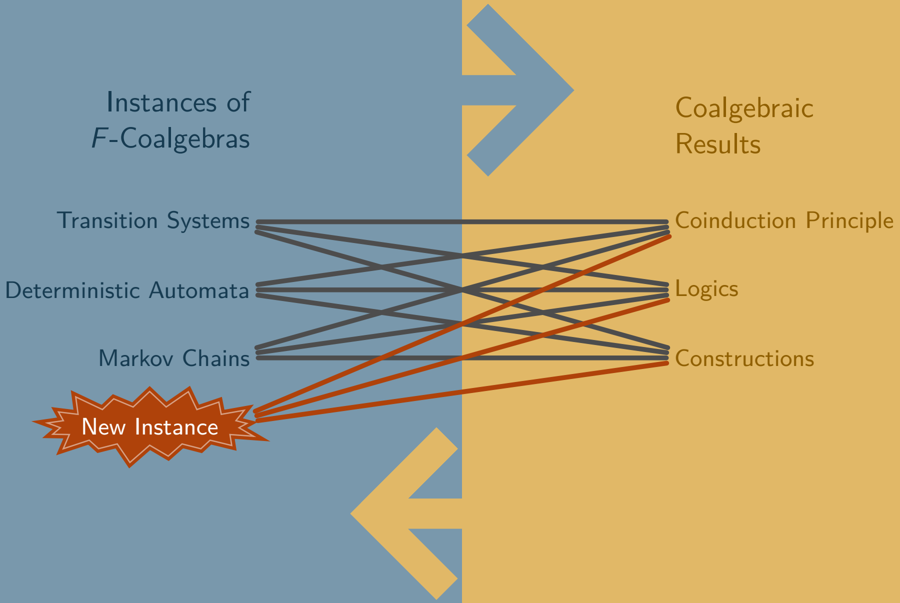
Deterministic Automata

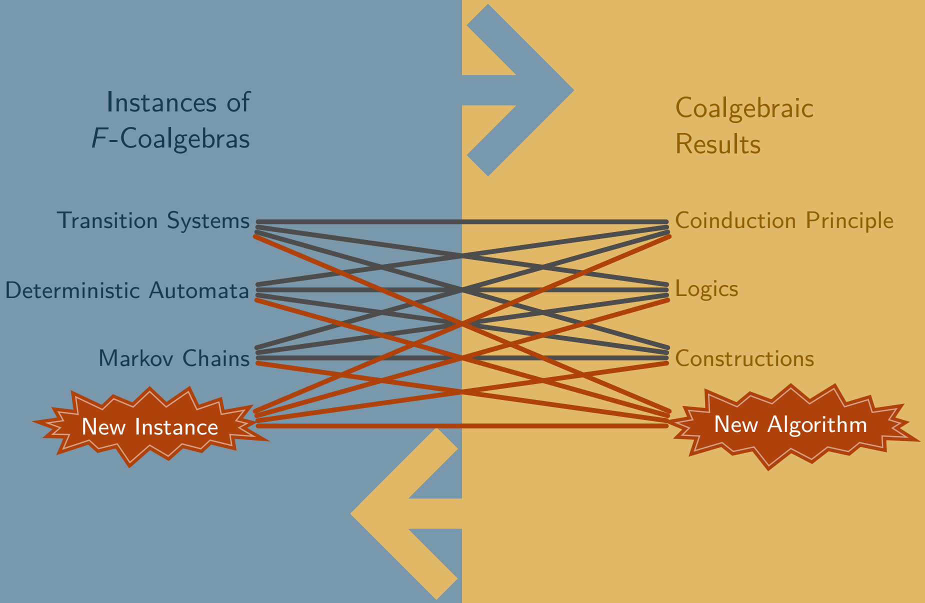
Markov Chains

Coinduction Principle

Logics

Constructions


Instances of
F-Coalgebras
Coalgebraic
Results
Transition Systems
Deterministic Automata
Markov Chains
New Instance
Coinduction Principle
Logics
Constructions


Instances of
F-Coalgebras
Coalgebraic
Results
Transition Systems
Deterministic Automata
Markov Chains
New Instance
Coinduction Principle
Logics
Constructions
New Algorithm


Functor F
e.g.
Automaton
F-coalgebra
Coalgebraic
Partition
Refinement
Behaviourally
Equivalent
States
e.g.
minimized
Automaton

New Algorithm

Theorem: Coalgebraic Partition Refinement

If F is "zippable" & is implements our interface, our algorithm correctly minimizes an F-coalgebra in time:

Transitions (m), States (n)

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

New Algorithm

Theorem: Coalgebraic Partition Refinement

If F is "zippable" & is implements our interface, our algorithm correctly minimizes an F-coalgebra in time:

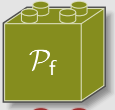
Transitions → m, States → n

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

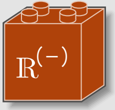
Implementation

- CoPaR
- in Haskell
- can handle large input files (200MB)

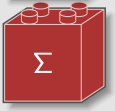
System Type	Functor FX	Run time $(m \geq n)$		Dedicated Algorithm	
Transition Systems	$\mathcal{P}_\mathsf{f}X$	$m \cdot \log n$	=	$m \cdot \log n$	Paige, Tarjan 1987
Markov Chains	$\mathbb{R}^{(X)}$	$m \cdot \log n$	=	$m \cdot \log n$	Valmari, Franceschinis 2010
Deterministic Automata	$2 \times X^A$ (A fixed)	$n \cdot \log n$	=	$n \cdot \log n$	Hopcroft 1971
Colour Refinement	$\mathcal{B}X = \mathbb{N}^{(X)}$	$m \cdot \log n$	=	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017



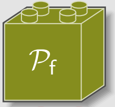
Powerset Functor
Non-Determinism & Bisimilarity



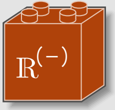
Distribution Functor
Probabilistic Behaviour



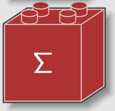
Signature Functor
Automata & Infinite Trees



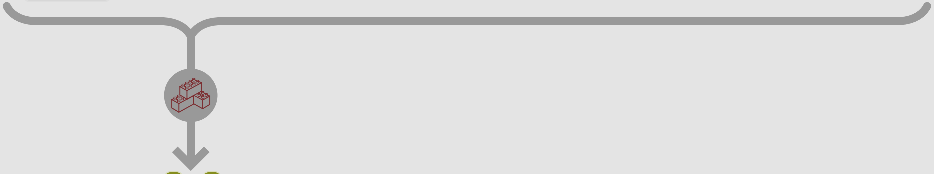
Powerset Functor
Non-Determinism & Bisimilarity

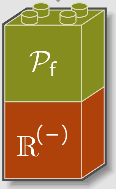


Distribution Functor
Probabilistic Behaviour

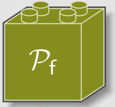


Signature Functor
Automata & Infinite Trees

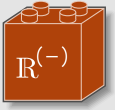




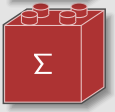
Coalgebras = Markov Decision Processes / Segala Systems



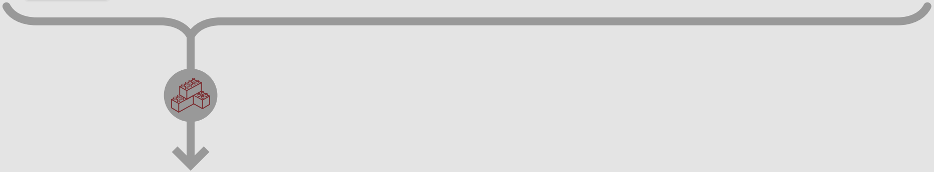
Powerset Functor
Non-Determinism & Bisimilarity

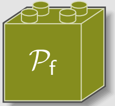


Distribution Functor
Probabilistic Behaviour

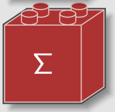


Signature Functor
Automata & Infinite Trees

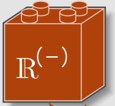




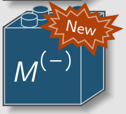
Powerset Functor
Non-Determinism & Bisimilarity



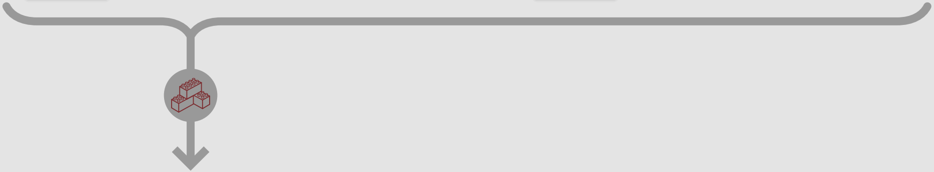
Signature Functor
Automata & Infinite Trees

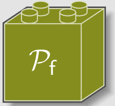


Distribution Functor
Probabilistic Behaviour

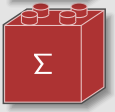


Monoid-valued Functor
Weighted Automata for General Monoids

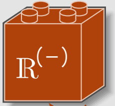




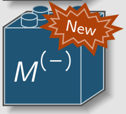
Powerset Functor
Non-Determinism & Bisimilarity



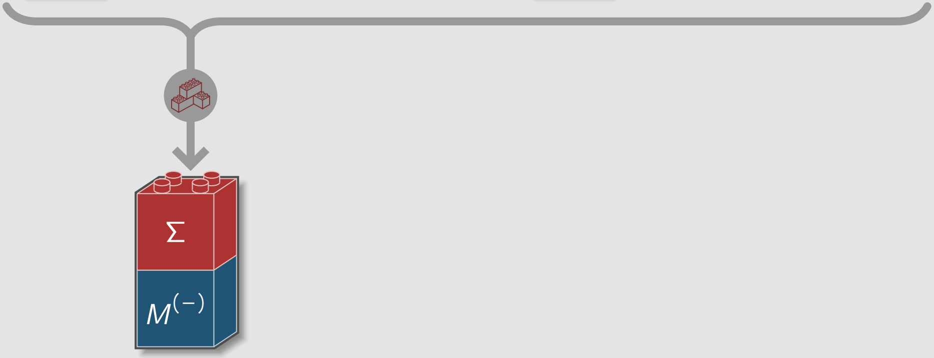
Signature Functor
Automata & Infinite Trees



Distribution Functor
Probabilistic Behaviour



Monoid-valued Functor
Weighted Automata for General Monoids



$\mathcal{P}_f$

Powerset Functor
Non-Determinism & Bisimilarity

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

Distribution Functor
Probabilistic Behaviour

Σ

Signature Functor
Automata & Infinite Trees

New

$M^{(-)}$

Monoid-valued Functor
Weighted Automata for General Monoids

Σ

$M^{(-)}$

Theorem

New Instance

Coalgebras for $M^{\Sigma(-)}$

$=$

Weighted Tree Automata with backwards bisimilarity (as considered by Högberg/Björklund, Maletti, May '07)

System Type	Functor FX	Run time $(m \geq n)$		Dedicated Algorithm	
Transition Systems	$\mathcal{P}_{\mathsf{f}}X$	$m \cdot \log n$	=	$m \cdot \log n$	Paige, Tarjan 1987
Markov Chains	$\mathbb{R}^{(X)}$	$m \cdot \log n$	=	$m \cdot \log n$	Valmari, Franceschinis 2010
Deterministic Automata	$2 \times X^A$ $(A \text{ fixed})$	$n \cdot \log n$	=	$n \cdot \log n$	Hopcroft 1971
Colour Refinement	$\mathcal{B}X = \mathbb{N}^{(X)}$	$m \cdot \log n$	=	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017

System Type	Functor FX	Run time $(m \geq n)$		Dedicated 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
Deterministic Automata	$2 \times X^A$ $(A \text{ fixed})$	$n \cdot \log n$	$=$	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_f(A \times X)$	$\lvert A\rvert \cdot n \cdot \log n$	$=$	$\lvert A\rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Markov Decision Processes (Segala Systems)	$\mathcal{P}_f(A \times \mathcal{D}X)$	$m_{\mathcal{D}} \cdot \log m_{\mathcal{P}_f}$	$<$	$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			$=$	$m_{\mathcal{D}} \cdot \log m_{\mathcal{P}_f}$	Groote, Verduzco, de Vink 2018
Colour Refinement	$\mathcal{B}X = \mathbb{N}^{(X)}$	$m \cdot \log n$	$=$	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata (Backwards Bisimulation)	$M^{(\Sigma X)}$ $(M \text{ non-cancellable})$	$m \cdot \log^2 m$	$<$	$m \cdot n$	Högberg/Björklund, Maletti, May 2007
	$M^{(\Sigma X)}$ $(M \text{ cancellable})$	$m \cdot \log m$	$=$ Σ fixed	$m \cdot \log n$	Högberg/Björklund, Maletti, May 2007

System Type	Functor FX	Run time $(m \geq n)$		Dedicated 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
Deterministic Automata	$2 \times X^A$ (A fixed)	$n \cdot \log n$	=	$n \cdot \log n$	Hopcroft 1971
	$2 \times \mathcal{P}_f(A \times X)$	$\lvert A \rvert \cdot n \cdot \log n$	=	$\lvert A \rvert \cdot n \cdot \log n$	Gries 1973/Knuutila 2001
Markov Decision Processes (Segala Systems)	$\mathcal{P}_f(A \times \mathcal{D}X)$	$m_{\mathcal{D}} \cdot \log m_{\mathcal{P}_f}$	<	$m \cdot \log n$	Baier, Engelen, Majster-Cederbaum 2000
			=	$m_{\mathcal{D}} \cdot \log m_{\mathcal{P}_f}$	Groote, Verduzco, de Vink 2018
Colour Refinement	$\mathcal{B}X = \mathbb{N}^{(X)}$	$m \cdot \log n$	=	$m \cdot \log n$	Berkholz, Bonsma, Grohe 2017
Weighted Tree Automata (Backwards Bisimulation)	$M^{(\Sigma X)}$ (M non-cancellable)	$m \cdot \log^2 m$	<	$m \cdot n$	Högberg/Björklund, Maletti, May 2007
	$M^{(\Sigma X)}$ (M cancellable)	$m \cdot \log m$	= Σ fixed	$m \cdot \log n$	Högberg/Björklund, Maletti, May 2007

Let category theory be your guide to
Generic **Efficient**
algorithm and software design

[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 Informatica* 2 (1973), pp. 97–109. ISSN: 1432-0525.

[GVdV18] Jan Friso Groote, Jao Rivera Verduzco, Erik P. 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/Björklund, Andreas Maletti, Jonathan May. "Bisimulation Minimisation for Weighted Tree Automata". In: *Developments in Language Theory, DLT 2007*. Vol. 4588. LNCS. Springer, 2007, pp. 229–241. ISBN: 978-3-540-73207-5. DOI: 10.1007/978-3-540-73208-2. URL: https://doi.org/10.1007/978-3-540-73208-2.

[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, Robert E. 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.