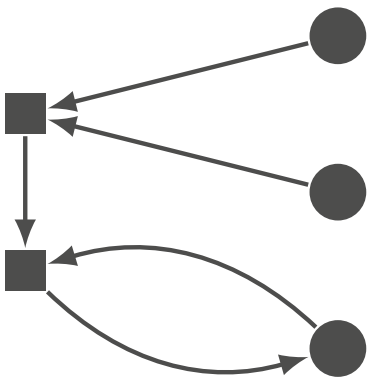
Generic Partition Refinement and Weighted Tree Automata

Hans-Peter Deifel, Stefan Milius, Lutz Schröder, Thorsten Wißmann

FM'19

10.10.2019

Partition Refinement



Paige-Tarjan $m \cdot \log n$

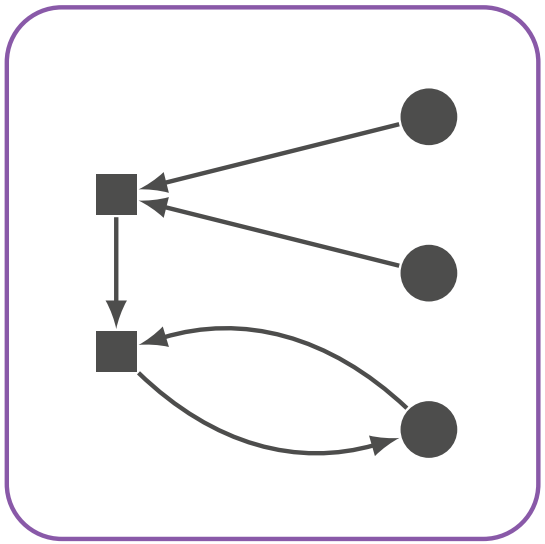
Hopcroft $n \cdot \log n$

Colour Refinement $m \cdot \log n$

Markov Chain Lumping $m \cdot \log n$

...and many more

Partition Refinement



Paige-Tarjan $m \cdot \log n$

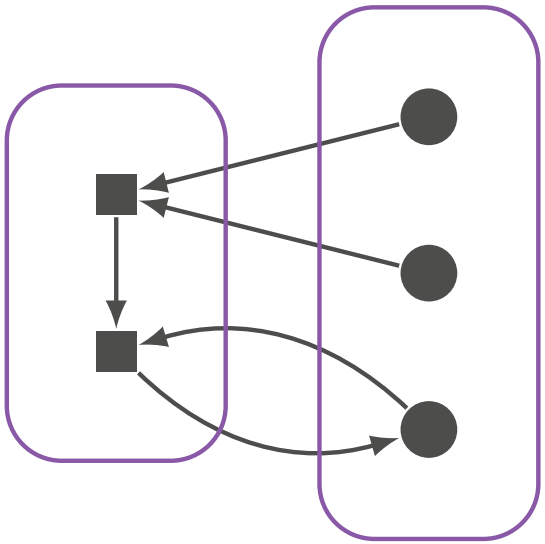
Colour Refinement $m \cdot \log n$

Hopcroft $n \cdot \log n$

Markov Chain Lumping $m \cdot \log n$

...and many more

Partition Refinement



Paige-Tarjan $m \cdot \log n$

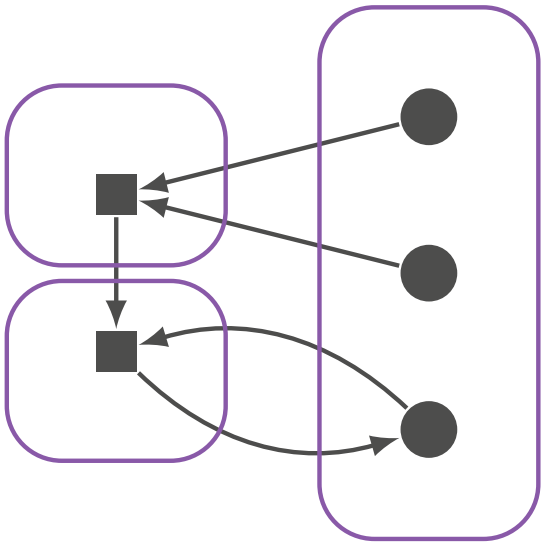
Colour Refinement $m \cdot \log n$

Hopcroft $n \cdot \log n$

Markov Chain Lumping $m \cdot \log n$

...and many more

Partition Refinement



Paige-Tarjan $m \cdot \log n$

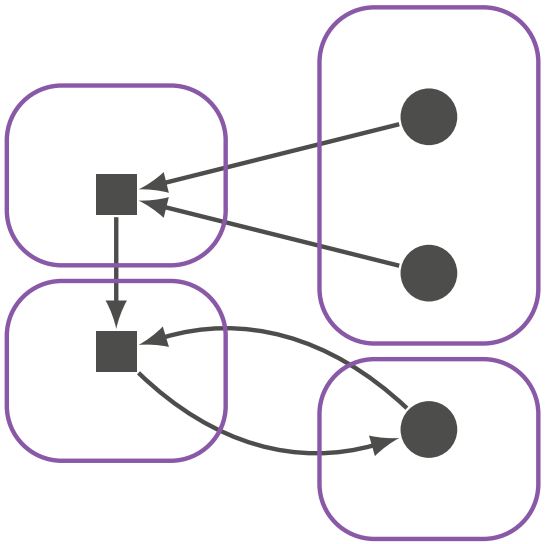
Colour Refinement $m \cdot \log n$

Hopcroft $n \cdot \log n$

Markov Chain Lumping $m \cdot \log n$

...and many more

Partition Refinement



Paige-Tarjan $m \cdot \log n$

Hopcroft $n \cdot \log n$

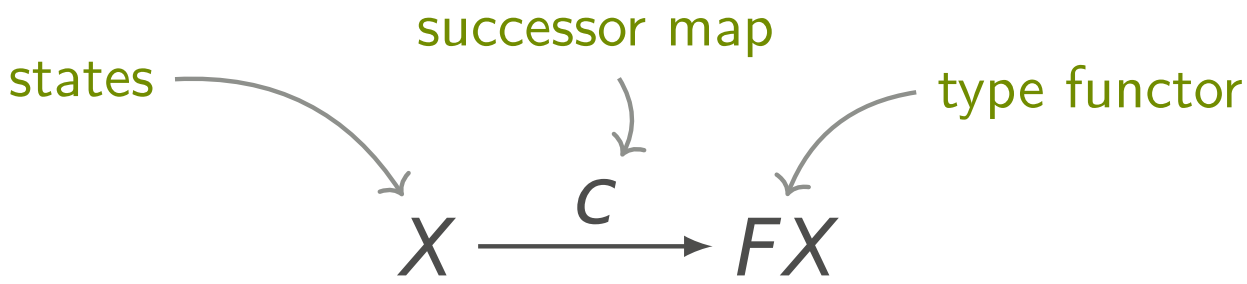
Colour Refinement $m \cdot \log n$

Markov Chain Lumping $m \cdot \log n$

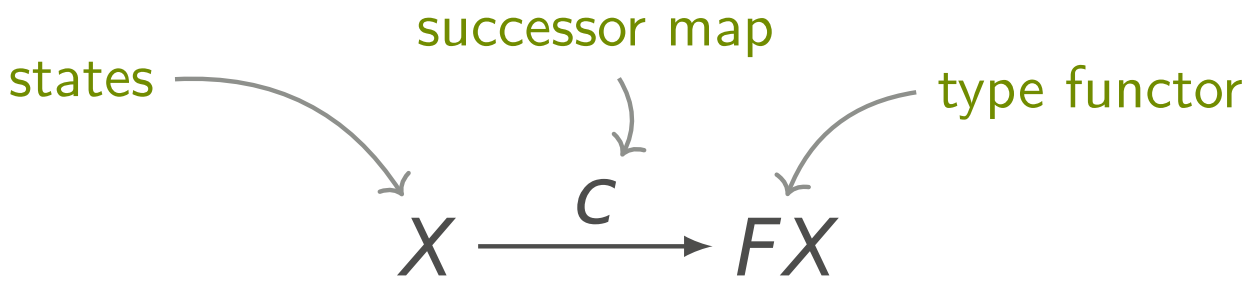
...and many more

Coalgebraic Partition Refinement

Coalgebras



Coalgebras

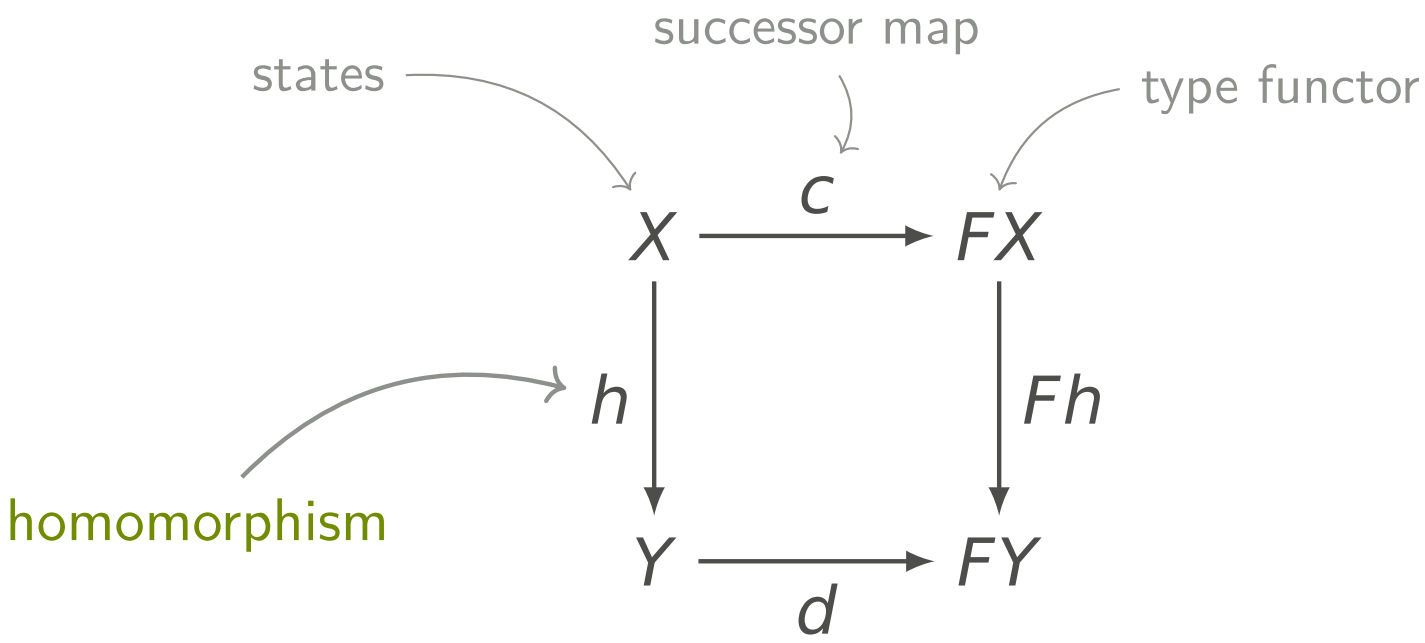


Type Functor $F : Set \to Set$

$FX =$

$\mathcal{P}(A \times X)$	$2 \times X^A$	$\mathbb{R}^{(X)}$	...
Labelled Transition Systems	Deterministic Automata	Markov Chains	

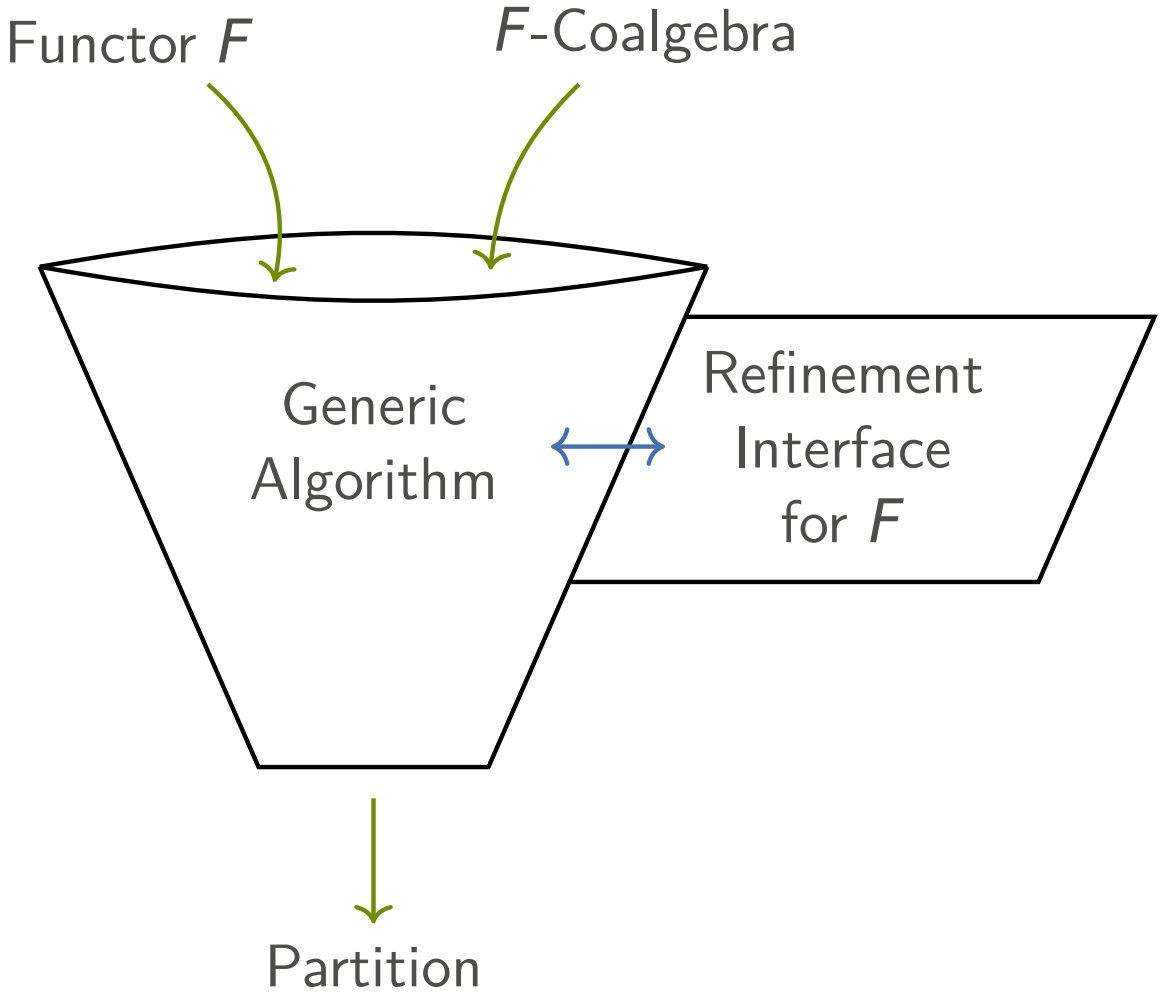
Behavioural Equivalence



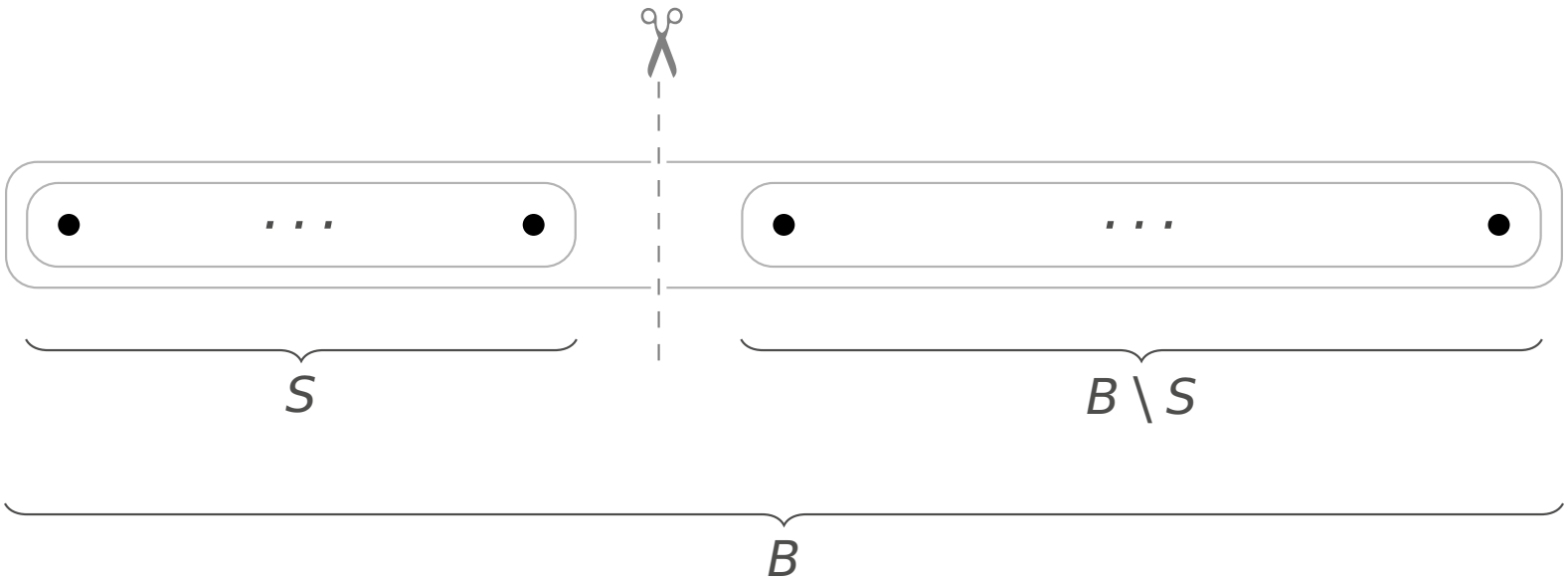
Identified by Coalgebra Homomorphism

$FX =$	$\mathcal{P}(A \times X)$	$2 \times X^A$	$\mathbb{R}^{(X)}$	...
	Bisimilarity	Language Equivalence	Weighted Bisimilarity	

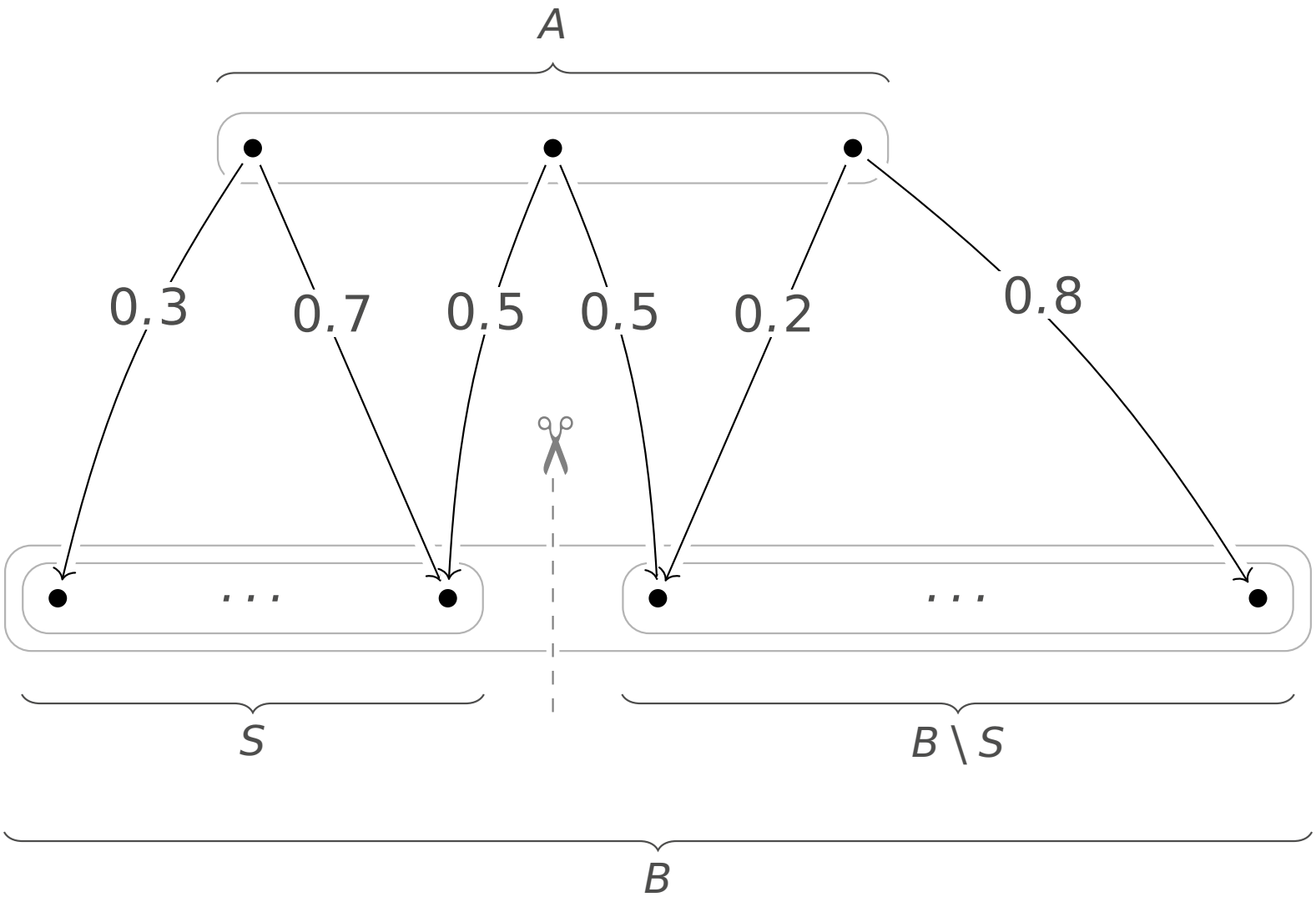
Coalgebraic Partition Refinement



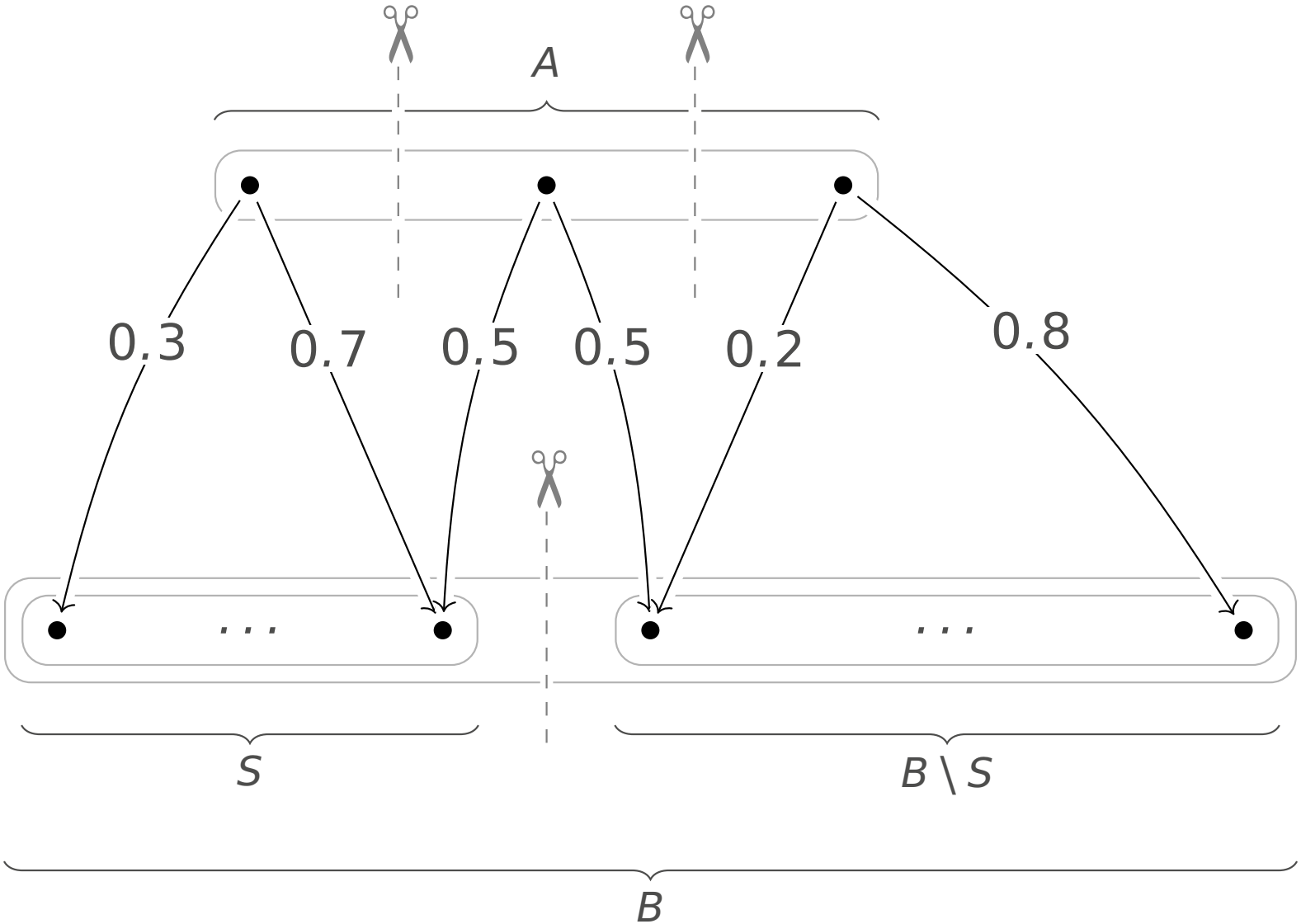
Splitting Blocks



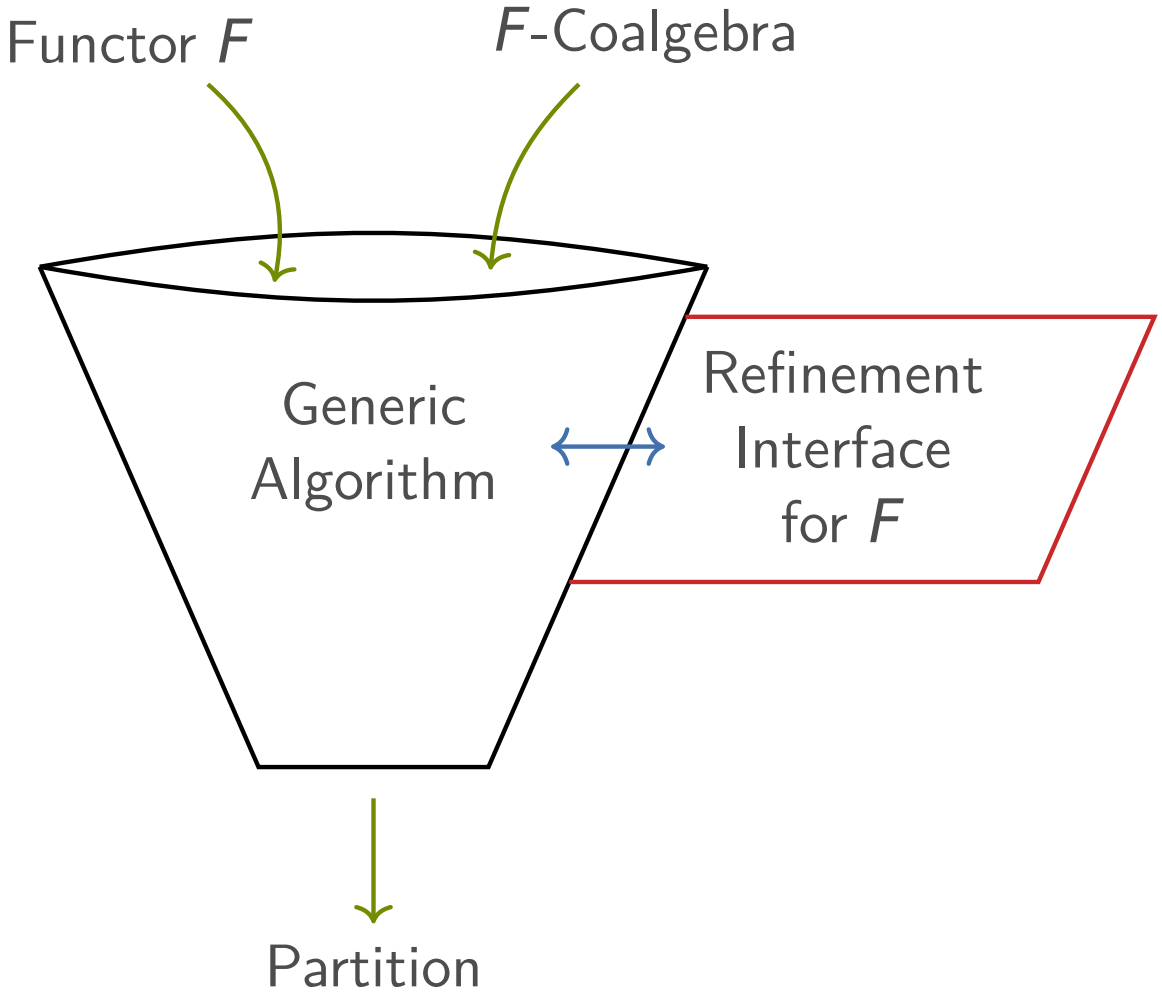
Splitting Blocks



Splitting Blocks



Coalgebraic Partition Refinement



Refinement Interface for F

Functor Encoding

Bags

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

Refinement Interface

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

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

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

Refinement Interface for F

Functor Encoding

Bags

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

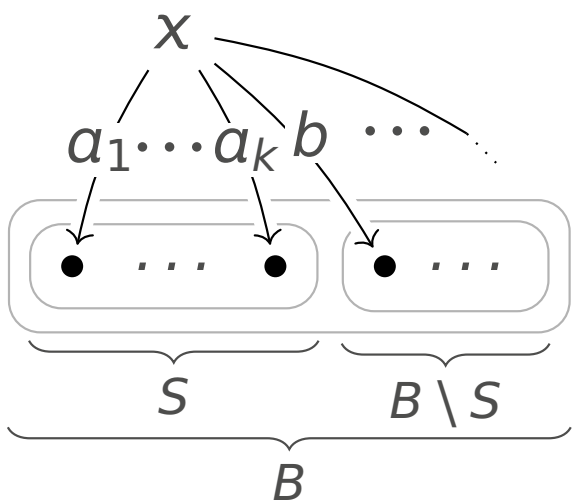
Refinement Interface

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 $\quad$ Weight of B



Refinement Interface for F

Functor Encoding

Bags

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

Refinement Interface

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

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

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

Example: $FX = \mathbb{R}^X$ (real-valued functor)

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

$$\texttt{init}(\_, l) = (0, \Sigma l)$$

$$\texttt{update}(l, (r, b)) = ((r + b - \Sigma l, \Sigma l), \ldots)$$

Refinement Interface for F

Functor Encoding

Bags

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

Refinement Interface

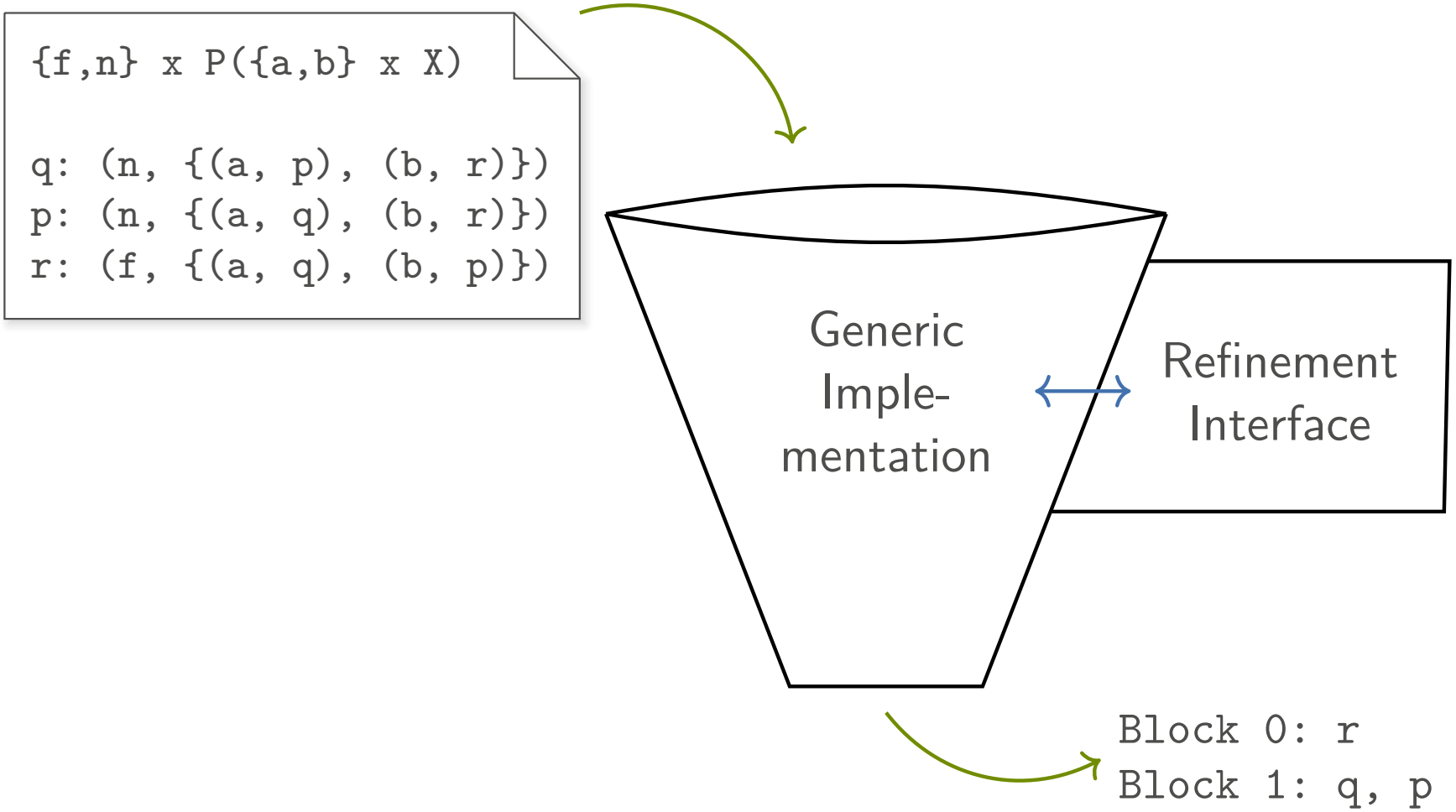
Type W (abstract, could be ints, reals, trees, ...)

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

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

Modular: Refinement Interface can automatically be derived for composite functors (e.g. $PD(-), M \times M^{(\Sigma -)}$)

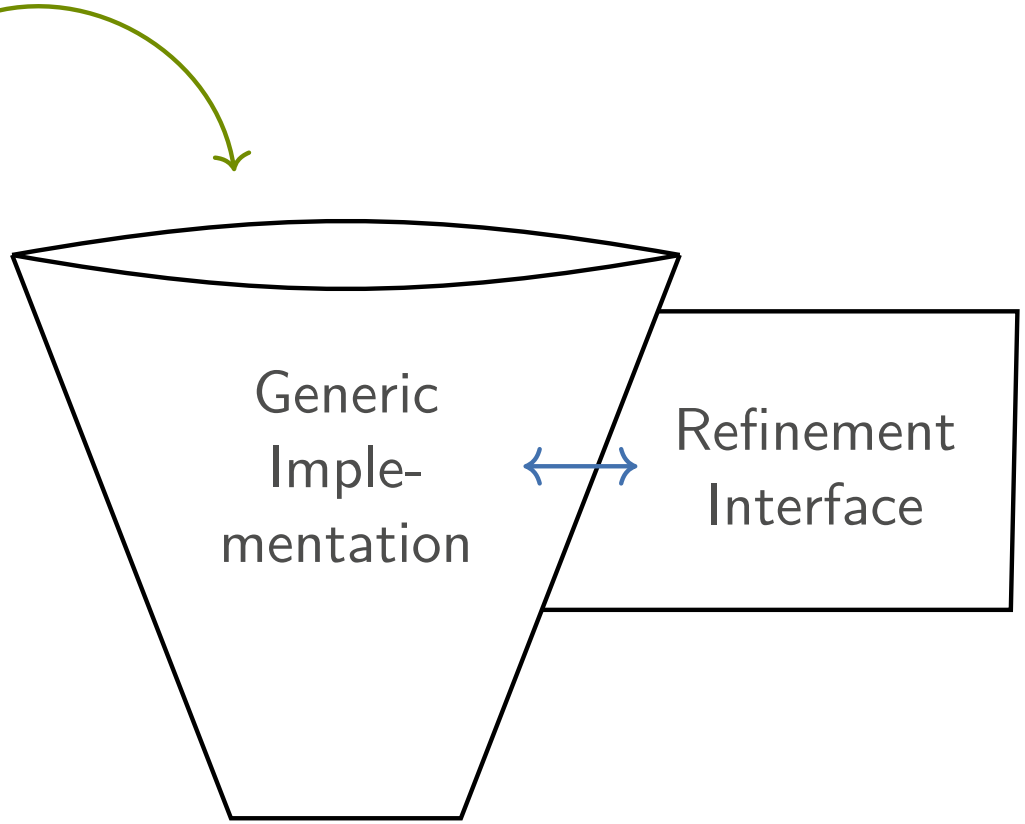
Implementation “CoPaR”



Implementation “CoPaR”

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

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



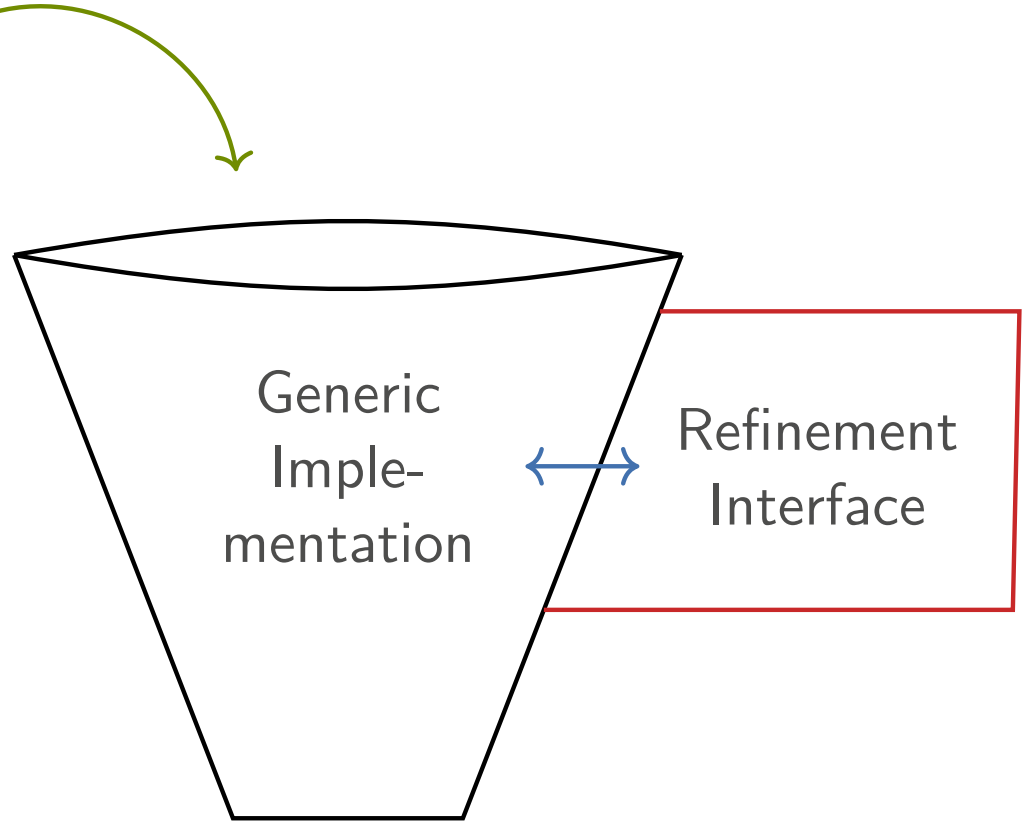
Available at:
`git8.cs.fau.de/software/copar`

```
Block 0: r
Block 1: q, p
```

Implementation “CoPaR”

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

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




```
Block 0: r
Block 1: q, p
```

Available at:
`git8.cs.fau.de/software/copar`

Implementation - Refinement Interface

Math

$\texttt{init}: F1 \times \mathcal{B}A \to W$
$\texttt{update}: \mathcal{B}A \times W \to W \times F3 \times W$

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)
```

Also Modular: Automatically derived for composite functors (e.g. $PD(-), M \times M^{(\Sigma -)}$)

Complexity

Refinement Interface

$\texttt{init}: F1 \times \mathcal{B}A \to W, \quad \texttt{update}: \mathcal{B}A \times W \to W \times F3 \times W$

- $|\ell|$ number of labels to subblock ($\mathcal{B}A$)
- n number of states in coalgebra
- m number of edges in coalgebra

Complexity

Refinement Interface

$\texttt{init}: F1 \times \mathcal{B}A \to W, \quad \texttt{update}: \mathcal{B}A \times W \to W \times F3 \times W$

$\mathcal{O}(|\ell| \cdot p(n, m))$ $\quad$ $\mathcal{O}(|\ell| \cdot p(n, m))$

Overall

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

Modularity: May add intermediate states

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

Case Study: Weighted Tree Automata

Definition

WTA: $(Q, \Sigma, \mathcal{M}, f, \mu)$

- Q: Set of states
- Σ: Ranked alphabet
- $\mathcal{M}$: Semiring $(M, \cdot, +)$
- f: Final weight distribution $f : Q \to M$
- μ: Transition function $\mu : \Sigma_k \to Q^k \to Q \to M$

Case Study: Weighted Tree Automata

Example Language: zigzag (Högberg, Maletti, May)

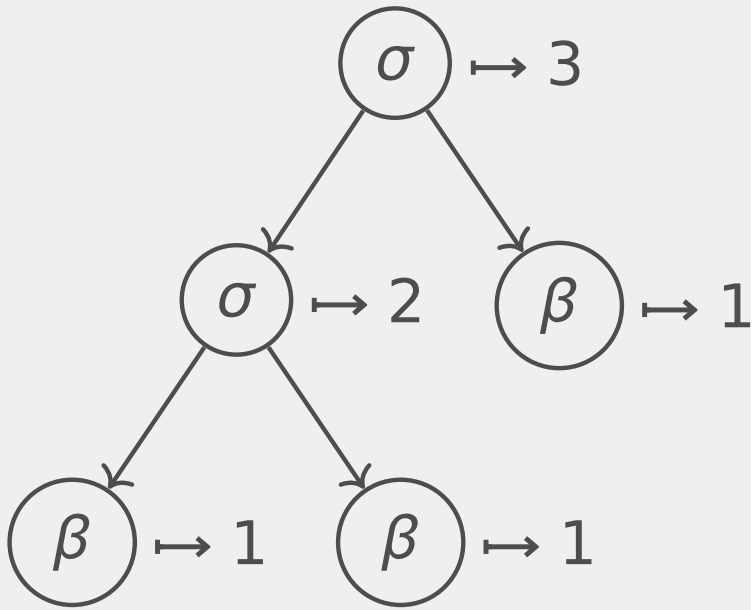
$\Sigma = \{\beta_{/0}, \sigma_{/2}\}, \quad \mathcal{M} = \mathbb{N}$

$\mathrm{zigzag} : T_\Sigma \to \mathbb{N}$

$\mathrm{zigzag}(\beta) = 1$

$\mathrm{zigzag}(\sigma(\beta, \_)) = 2$

$\mathrm{zigzag}(\sigma(\sigma(\_, t), \_)) = 2 + \mathrm{zigzag}(t)$



⇒ Recognized by automaton with 3 states

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
- Find equivalence that corresponds to coalgebraic behavioural equivalence
- Implement refinement interface

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
  $FX = M \times M^{(\Sigma X)}$
- Find equivalence that corresponds to coalgebraic behavioural equivalence
- Implement refinement interface

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
  $FX = M \times M^{(\Sigma X)}$
- Find equivalence that corresponds to coalgebraic behavioural equivalence
  "Backwards Bisimulation"
- Implement refinement interface

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
  $FX = M \times M^{(\Sigma X)}$
- Find equivalence that corresponds to coalgebraic behavioural equivalence
  "Backwards Bisimulation"
- Implement refinement interface

  Composite of:
  - $M \times -$
  - $M^{(-)}$
  - $\Sigma -$

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
  $FX = M \times M^{(\Sigma X)}$
- Find equivalence that corresponds to coalgebraic behavioural equivalence
  "Backwards Bisimulation"
- Implement refinement interface

  Composite of:
  - $M \times -$ Done
  - $M^{(-)}$
  - $\Sigma -$

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
  $FX = M \times M^{(\Sigma X)}$
- Find equivalence that corresponds to coalgebraic behavioural equivalence
  "Backwards Bisimulation"
- Implement refinement interface

  Composite of:
  - $M \times -$ Done
  - $M^{(-)}$
  - $\Sigma -$ Done

Case Study: Weighted Tree Automata

Tasks

- Describe the functor
  $FX = M \times M^{(\Sigma X)}$
- Find equivalence that corresponds to coalgebraic behavioural equivalence
  "Backwards Bisimulation"
- Implement refinement interface

  Composite of:
  - $M \times -$ Done
  - $M^{(-)}$???
  - $\Sigma -$ Done

Refinement Interface for $M^{(-)}$

$$M \text{ cancellative} \quad \Leftrightarrow \quad a + c = b + c \Rightarrow a = b$$

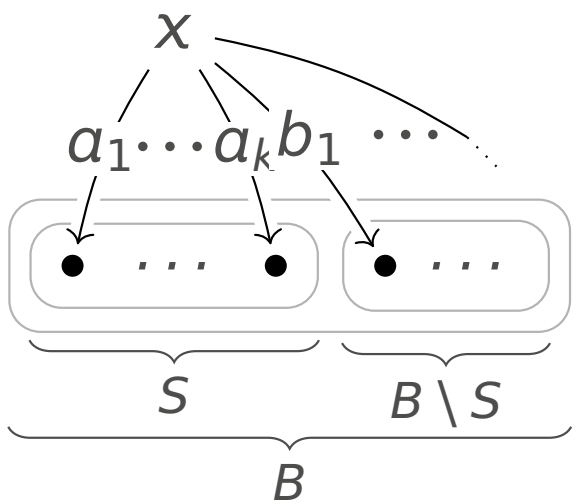
Refinement Interface

- M cancellative? Done (via Grothendieck construction)
- M non-cancellative? ???

Non-cancellative Monoids

Problem

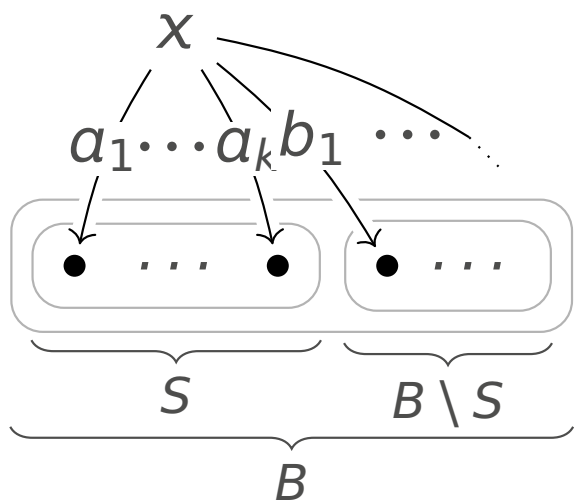
Need to subtract $w_B - w_S$



Non-cancellative Monoids

Problem

Need to subtract $w_B - w_S$



Trick

We know:

$$w_S = a_1 + \ldots + a_n, \quad w_B = a_1 + \ldots + a_n + b_1 + \ldots + b_m$$

$\Rightarrow$ Don't evaluate sums; store and manipulate symbolically

Complexity for non-cancellative monoids

Sums stored as balanced search trees $M \to \mathbb{N}$

$\Rightarrow$ Size of those trees $\min(|M|, m)$

$\Rightarrow$ Operations in $\mathcal{O}(\log \min(|M|, m))$

init	update
$\mathcal{O}(1)$	$\mathcal{O}(\lvert\ell\rvert \cdot \log \min(\lvert M\rvert, m))$

Complexity for non-cancellative monoids

Sums stored as balanced search trees $M \to \mathbb{N}$

$\Rightarrow$ Size of those trees $\min(|M|, m)$

$\Rightarrow$ Operations in $\mathcal{O}(\log \min(|M|, m))$

init	update
$\mathcal{O}(1)$	$\mathcal{O}(\lvert\ell\rvert \cdot \log \min(\lvert M\rvert, m))$

$$\Downarrow$$

$$p(n, m) = \log \min(|M|, m)$$

Complexity for non-cancellative monoids

Sums stored as balanced search trees $M \to \mathbb{N}$

$\Rightarrow$ Size of those trees $\min(|M|, m)$

$\Rightarrow$ Operations in $\mathcal{O}(\log \min(|M|, m))$

init

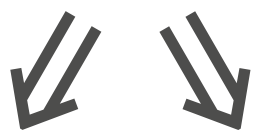
$\mathcal{O}(1)$

update

$\mathcal{O}(|\ell| \cdot \log \min(|M|, m))$

$\Downarrow$

$p(n, m) = \log \min(|M|, m)$



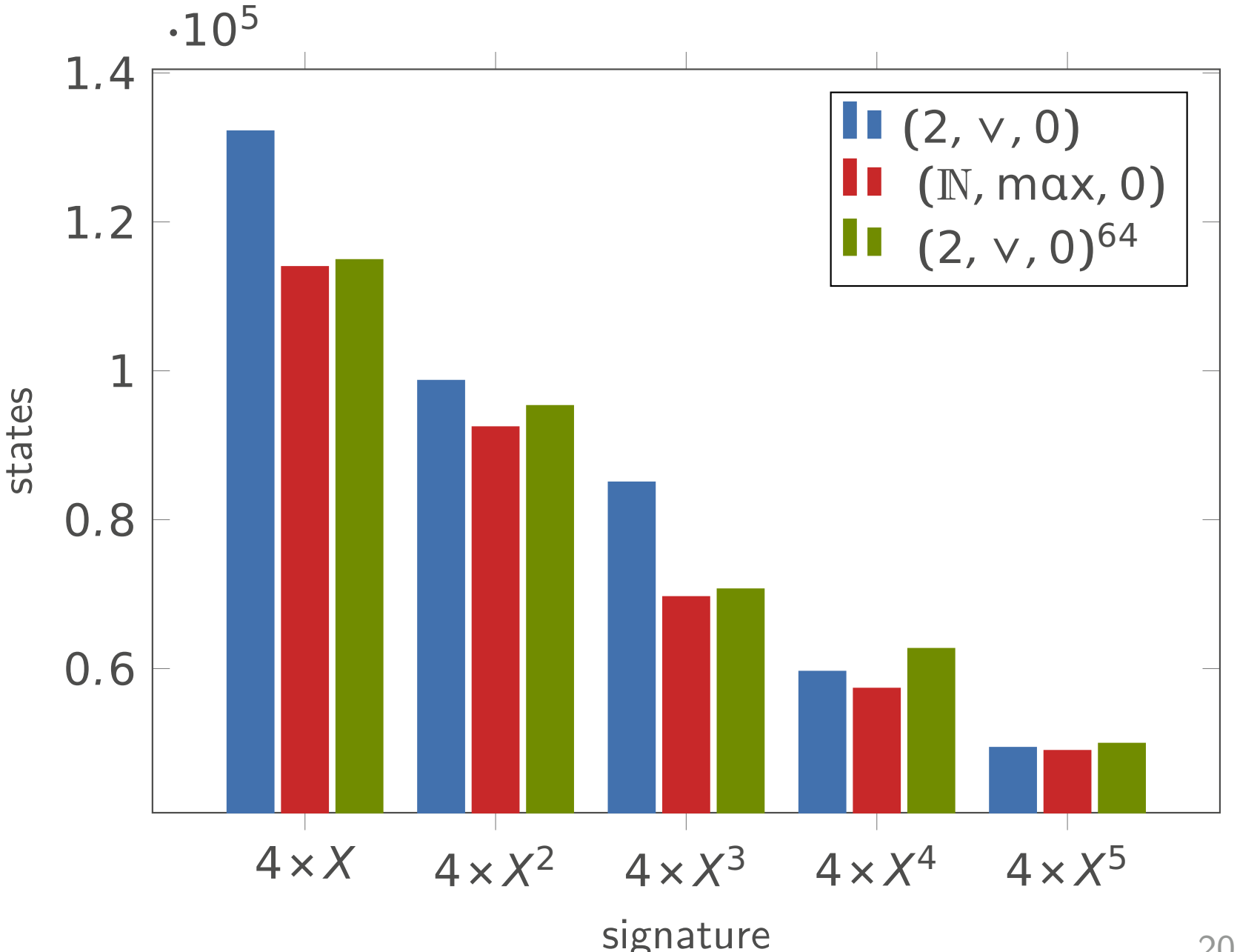
M finite

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

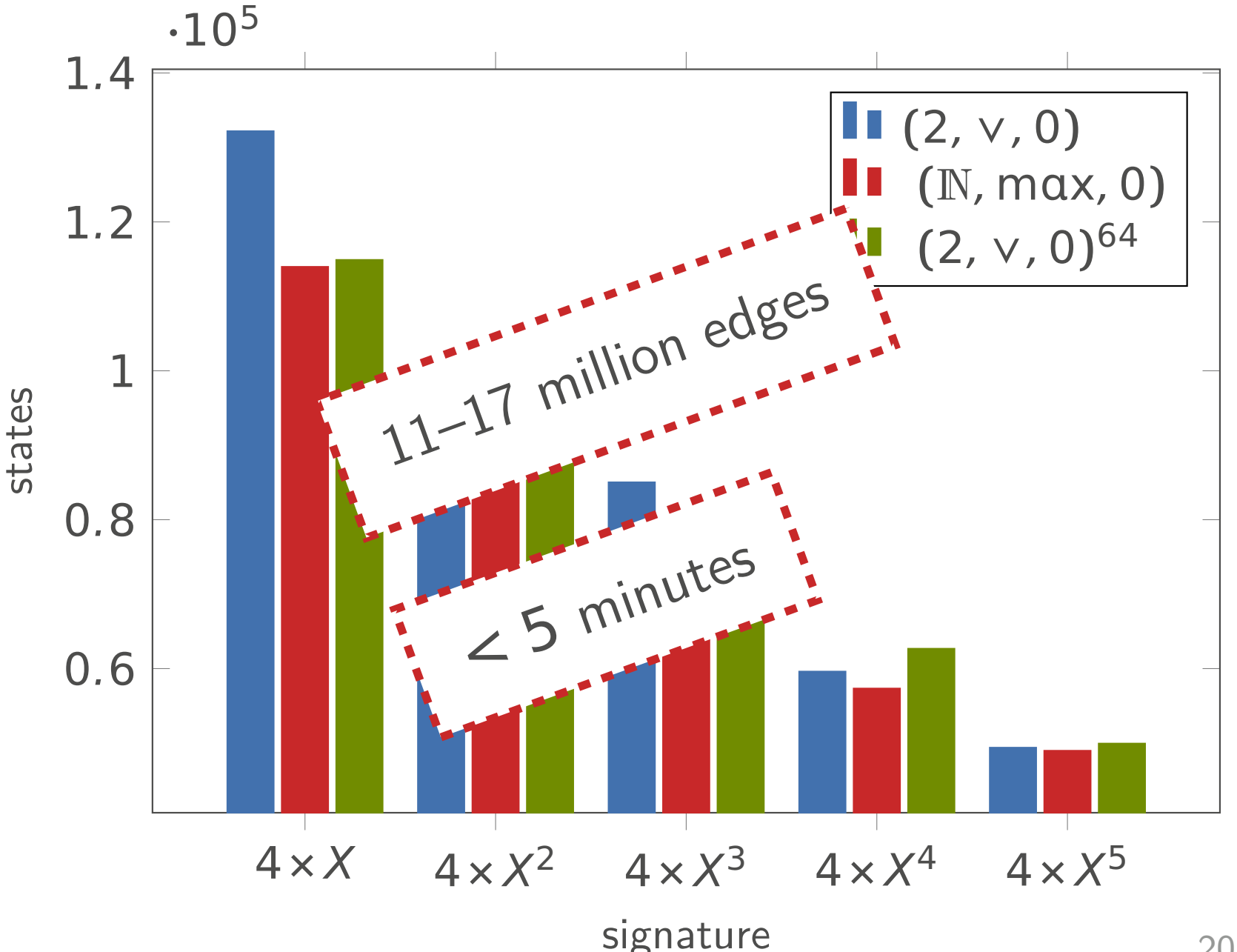
M infinite

$\mathcal{O}(m \log^2 m)$

Random WTAs in 16GB of RAM



Random WTAs in 16GB of RAM



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

Thanks!