Kleisli Simulation for Real-Weighted Automata and Its Algorithm

卜部 夏木
(東京大学理学部情報科学科４年)

Overview

- Simulation
- Kleisli Simulation
  1. Definition
  2. Searching Kleisli Simulation for Probabilistic System
- Contribution
  1. Implementation
  2. Increasing the Chance of Simulation
- Experimental Results and Comparison

卜部夏木（東大理学部情報科学科）

Overview

- Simulation
- Kleisli Simulation
  1. Definition
  2. Searching Kleisli Simulation for Probabilistic System
- Contribution
  1. Implementation
  2. Increasing the Chance of Simulation
- Experimental Results and Comparison

卜部夏木（東大理学部情報科学科）

目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

卜部夏木（東大理学部情報科学科）

目標

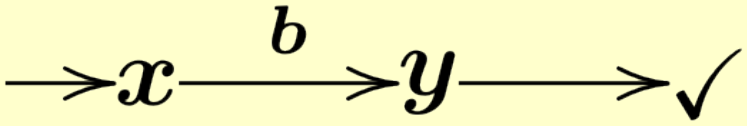
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係
- 例:非決定性オートマトンの場合

ト部夏木（東大理学部情報科学科）

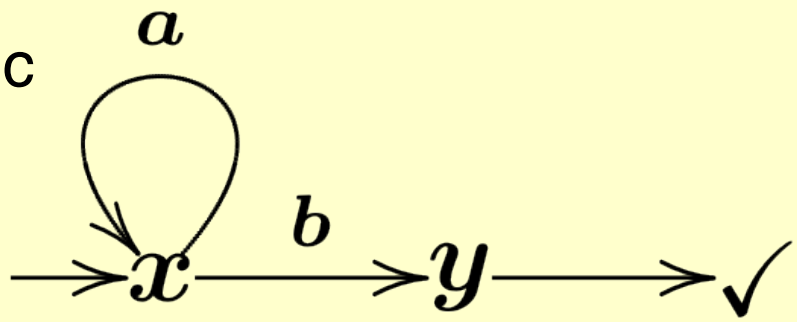
目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係
- 例:非決定性オートマトンの場合

Impl



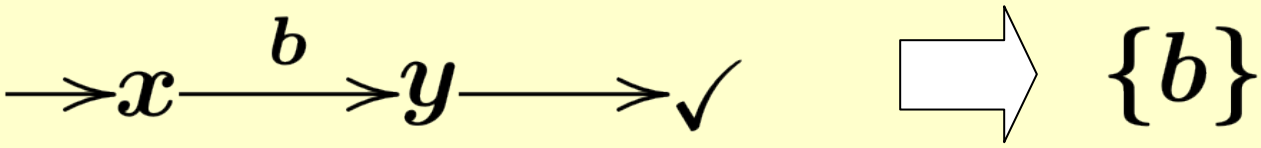
Spec



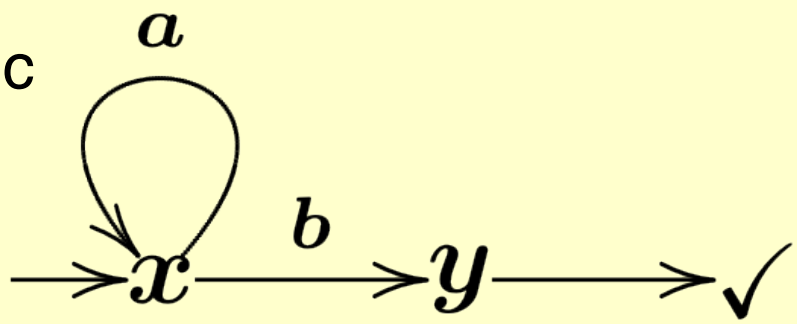
目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係
- 例:非決定性オートマトンの場合

Impl



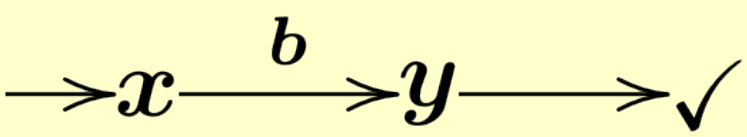
Spec



目標

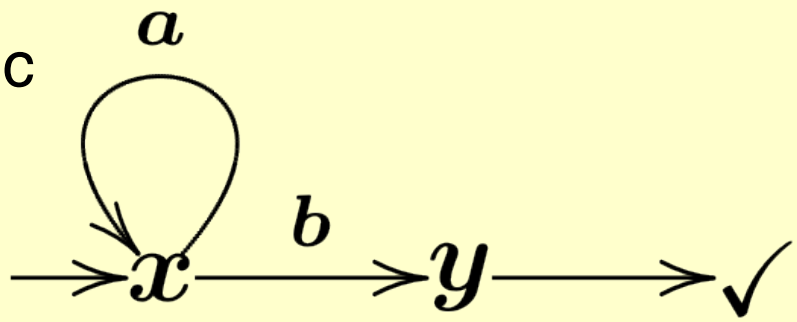
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係
- 例:非決定性オートマトンの場合

Impl



$\{b\}$

Spec

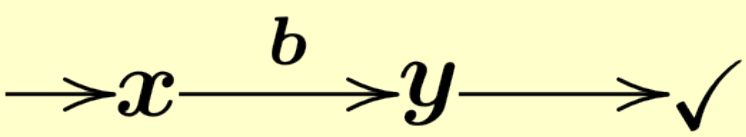


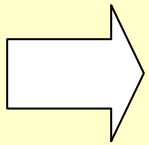
$\{b, ab, aab, \ldots\}$

目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係
- 例:非決定性オートマトンの場合

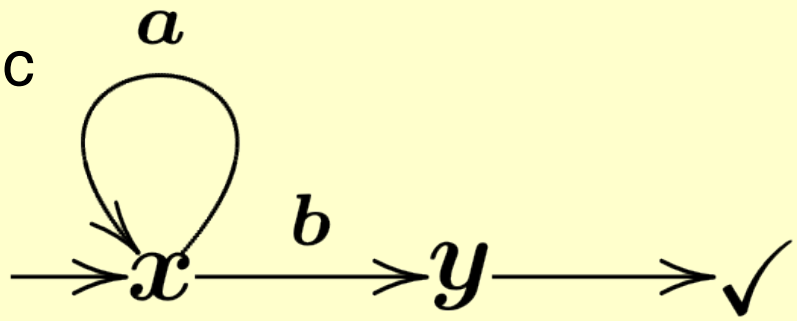
Impl

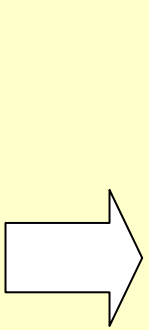




$\{b\}$

Spec





$\{b, ab, aab, \ldots\}$

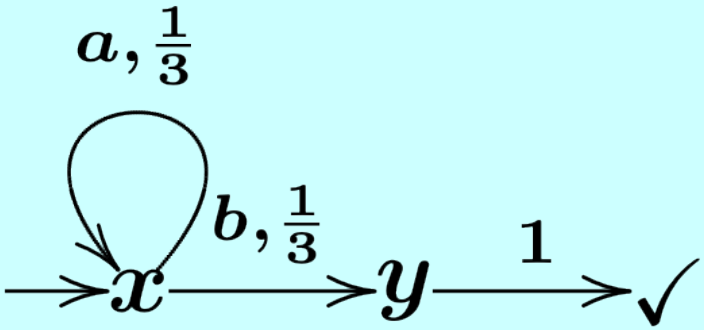
目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

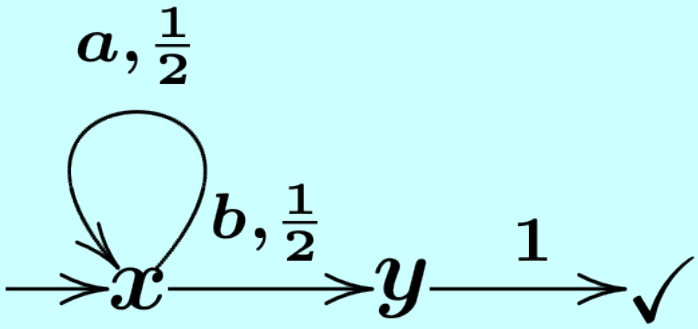
目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

Imple



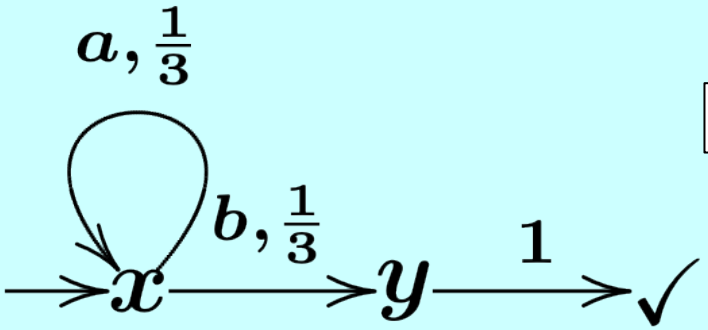
Spec



目標

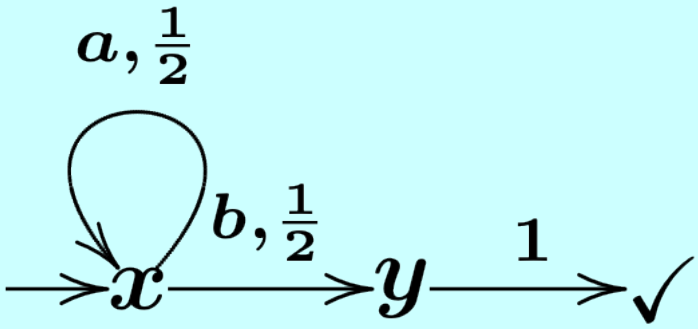
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

Imple



$$[b \mapsto \frac{1}{3}, ab \mapsto \frac{1}{9}, aab \mapsto \frac{1}{27}, \ldots]$$

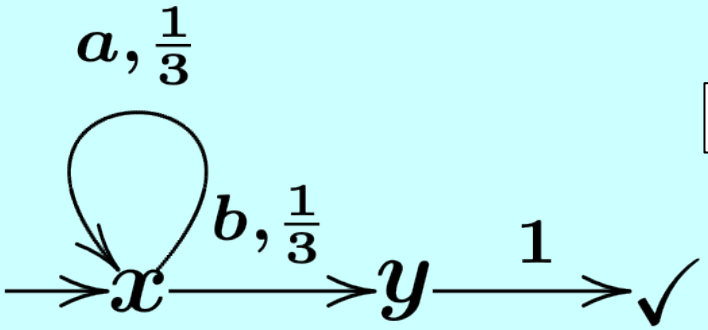
Spec



目標

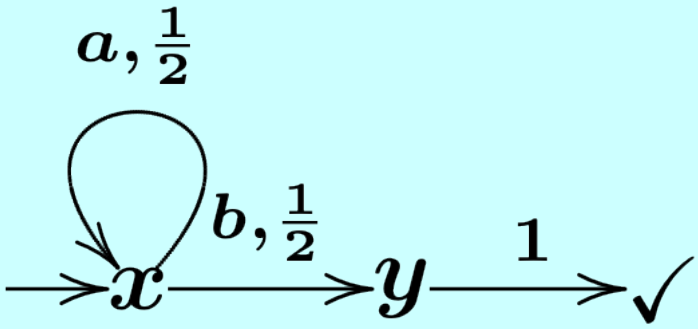
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

Imple



$$[b \mapsto \frac{1}{3}, ab \mapsto \frac{1}{9}, aab \mapsto \frac{1}{27}, \ldots]$$

Spec

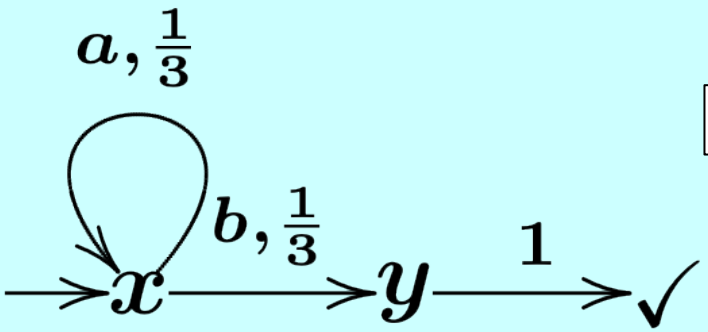


$$[b \mapsto \frac{1}{2}, ab \mapsto \frac{1}{4}, aab \mapsto \frac{1}{8}, \ldots]$$

目標

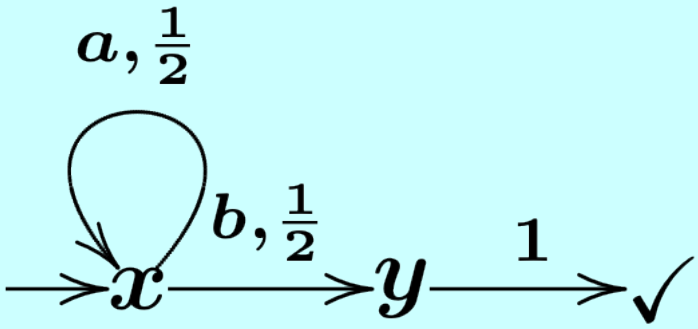
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

Imple



$$[b \mapsto \frac{1}{3}, ab \mapsto \frac{1}{9}, aab \mapsto \frac{1}{27}, \ldots]$$

Spec

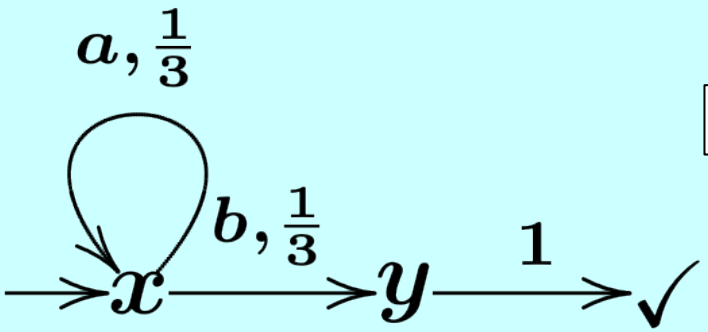


$$[b \mapsto \frac{1}{2}, ab \mapsto \frac{1}{4}, aab \mapsto \frac{1}{8}, \ldots]$$

目標

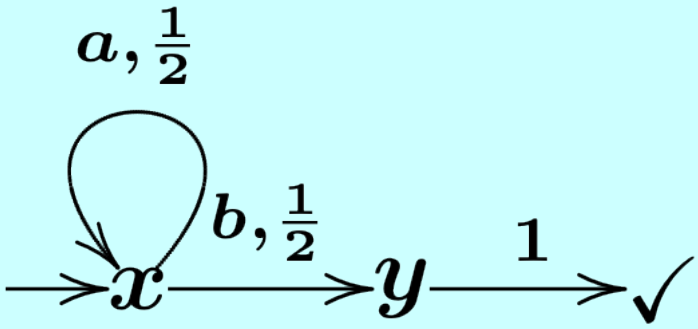
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

Imple



$$[b \mapsto \frac{1}{3}, ab \mapsto \frac{1}{9}, aab \mapsto \frac{1}{27}, \ldots]$$

Spec

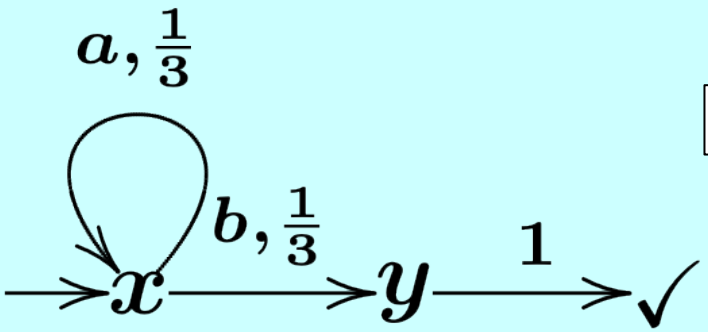


$$[b \mapsto \frac{1}{2}, ab \mapsto \frac{1}{4}, aab \mapsto \frac{1}{8}, \ldots]$$

目標

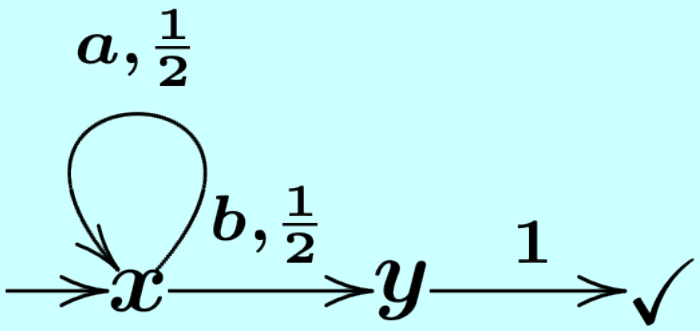
- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係

Imple



$$[b \mapsto \frac{1}{3}, ab \mapsto \frac{1}{9}, aab \mapsto \frac{1}{27}, \ldots]$$

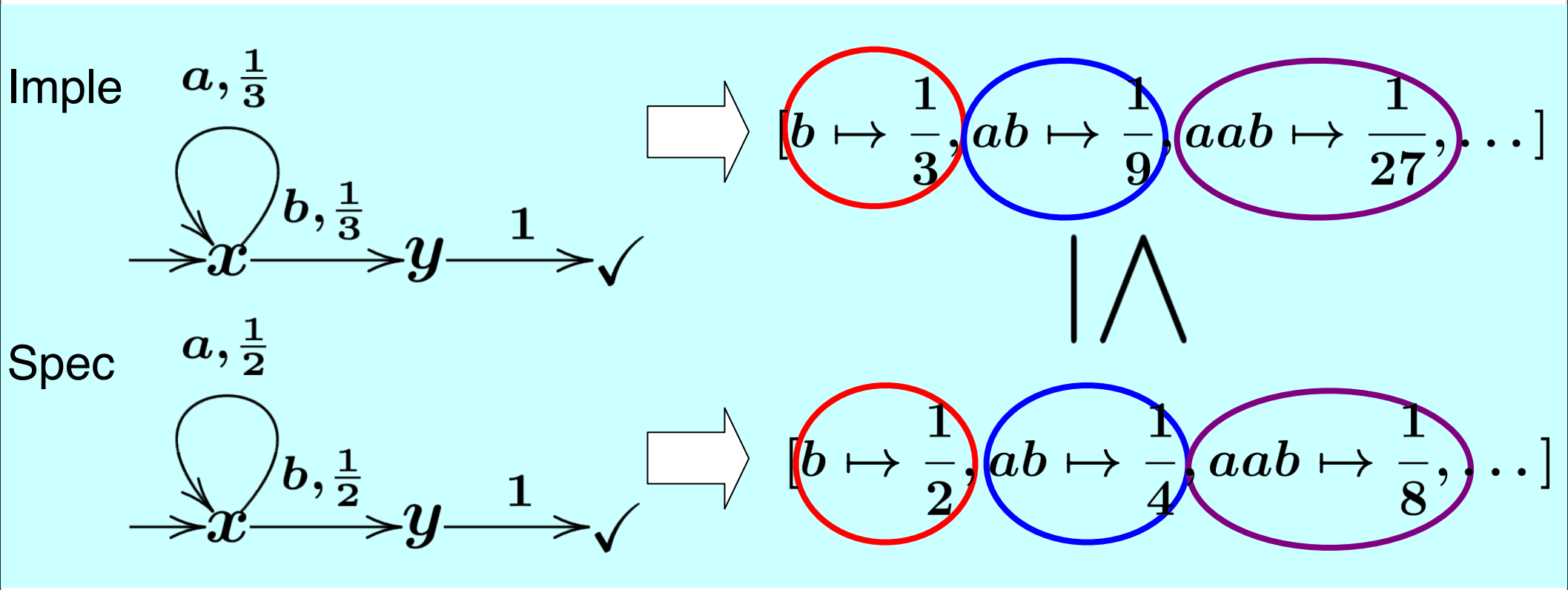
Spec



$$[b \mapsto \frac{1}{2}, ab \mapsto \frac{1}{4}, aab \mapsto \frac{1}{8}, \ldots]$$

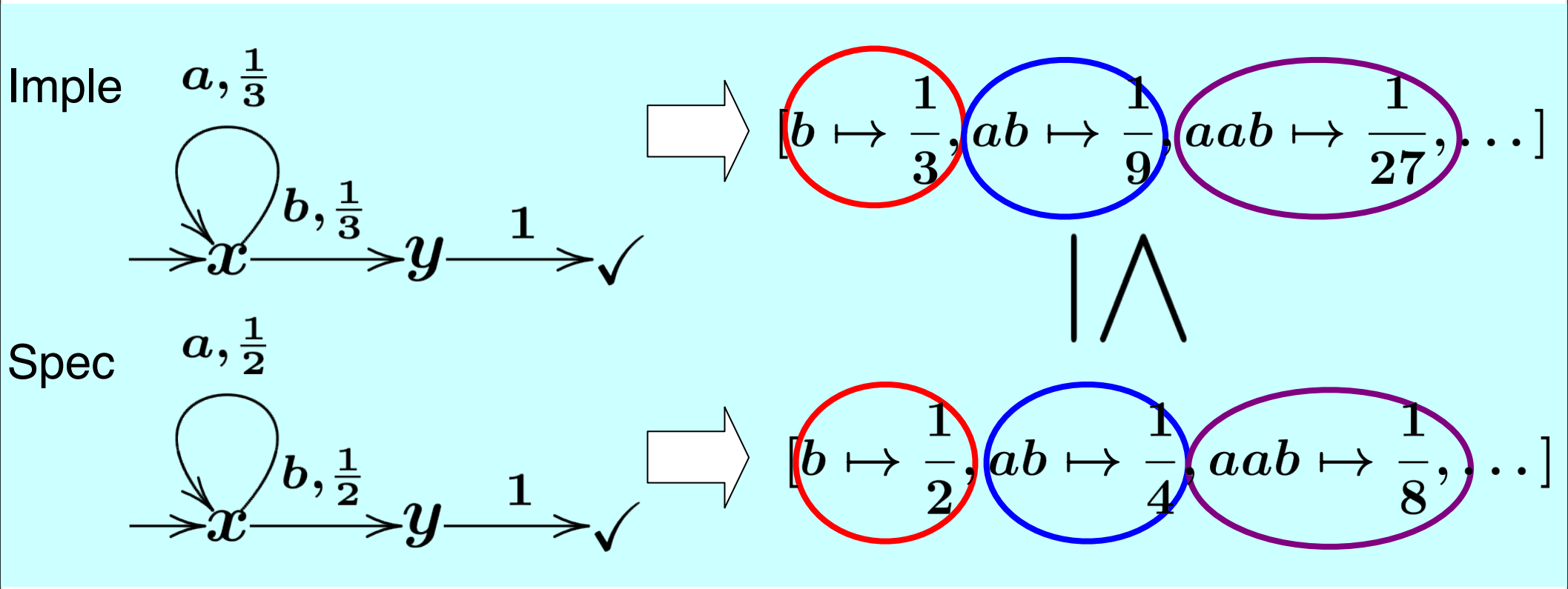
目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係



目標

- State-based system の trace inclusion を調べる
- Trace semantics = Linear-time behavior
- Trace inclusion = Trace Semantics の間の包含関係
- 例:非決定性オートマトンの場合



目標

- State-based system の trace inclusion を調べる
- Trace semantics は普通は無限

$$\{b, ab, aab, \dots\} \quad [b \mapsto \frac{1}{2}, ab \mapsto \frac{1}{4}, aab \mapsto \frac{1}{8}, \dots]$$

卜部夏木（東大理学部情報科学科）

目標

- State-based system の trace inclusion を調べる
- Trace semantics は普通は無限

$$\{b, ab, aab, \dots\} \quad [b \mapsto \frac{1}{2}, ab \mapsto \frac{1}{4}, aab \mapsto \frac{1}{8}, \dots]$$

⇨ 自動で調べることは難しい

Simulation = Stepwise Trace Inclusion

- 全体の trace inclusion を調べる代わりに

stepwise な trace inclusion を調べる

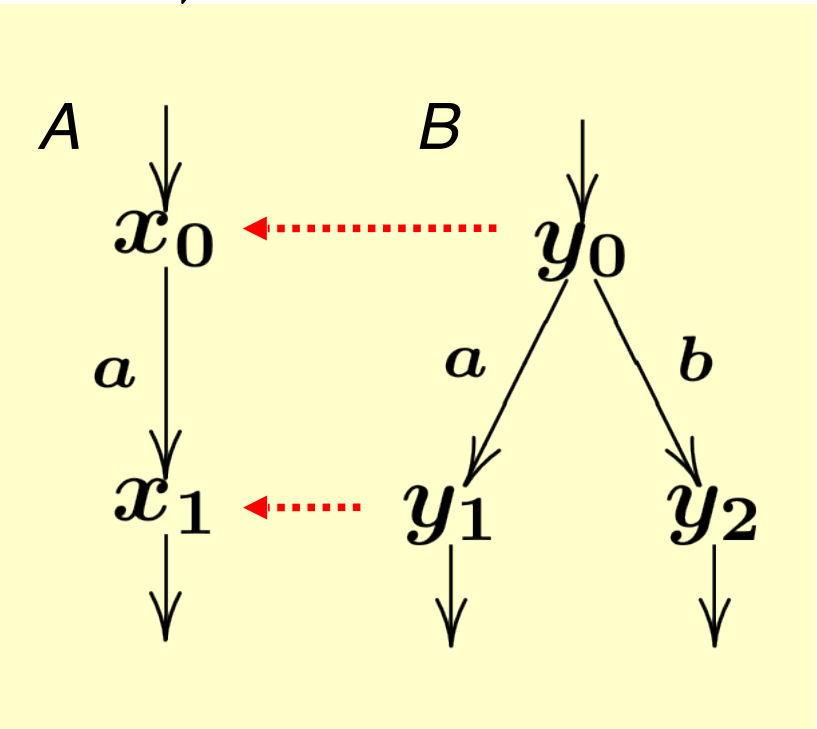
卜部夏木（東大理学部情報科学科）

Simulation = Stepwise Trace Inclusion

- 全体の trace inclusion を調べる代わりに

stepwise な trace inclusion を調べる

⇨ Simulation



$$R \subseteq A \times B \quad \text{s.t.}$$
$$xRy, x \xrightarrow{a} x' \Rightarrow \exists y'.y \xrightarrow{a} y', x'Ry'$$

⇨ AからBへの simulationが存在

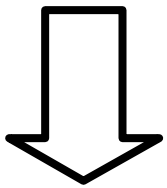
健全性

A から B への simulation が存在

卜部夏木（東大理学部情報科学科）

健全性

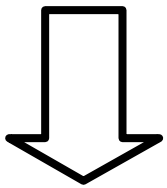
A から *B* への simulation が存在



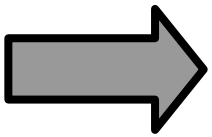
A の trace は *B* の trace に含まれる (trace inclusion)

健全性

A から *B* への simulation が存在



A の trace は *B* の trace に含まれる (trace inclusion)

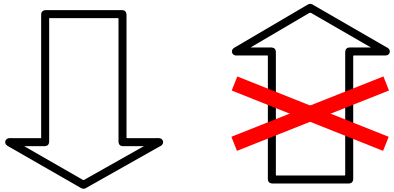


健全性

ト部夏木（東大理学部情報科学科）

健全性

A から *B* への simulation が存在

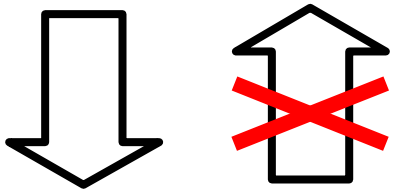


A の trace は *B* の trace に含まれる (trace inclusion)

⇒ 健全性

健全性

A から *B* への simulation が存在



A の trace は *B* の trace に含まれる (trace inclusion)

⇒ 健全性

⇒ 完全性は必ずしも成り立たない

卜部夏木（東大理学部情報科学科）

Overview

- Simulation
- Kleisli Simulation
  1. Definition
  2. Searching Kleisli Simulation for Probabilistic System
- Contribution
  1. Implementation
  2. Increasing the Chance of Simulation
- Experimental Results and Comparison

Kleisli 圏による表現

- **Sets** 上の Kleisli 圏でシステムを表現

$$1 \overset{s}{\nrightarrow} X \overset{c}{\nrightarrow} \overline{F}X \quad \text{in} \quad \mathcal{K}\ell(T)$$

X ・・・状態集合

$s : 1 \nrightarrow X$ ・・・始状態

$c : X \nrightarrow \overline{F}X$ ・・・遷移

$F : \mathbf{Sets} \to \mathbf{Sets}$

$\overline{F} : \mathcal{K}\ell(T) \to \mathcal{K}\ell(T)$

ト部夏木（東大理学部情報科学科）

Kleisli 圏による表現

- モナドTごとに様々なシステム

例：

Powerset monad

$T = P \quad PX = \{A \subseteq X\}$

→ 非決定性オートマトン

Subdistribution monad

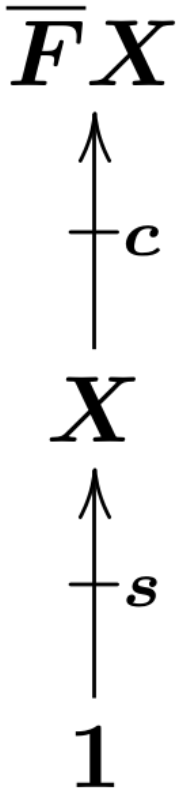
$T = D \quad DX = \{f : X \to [0,1] \mid \sum_x f(x) \leq 1\}$

→ 確率的オートマトン

Trace Semantics via Coinduction

(Hasuo, Jacobs, and Sokolova, 2006)

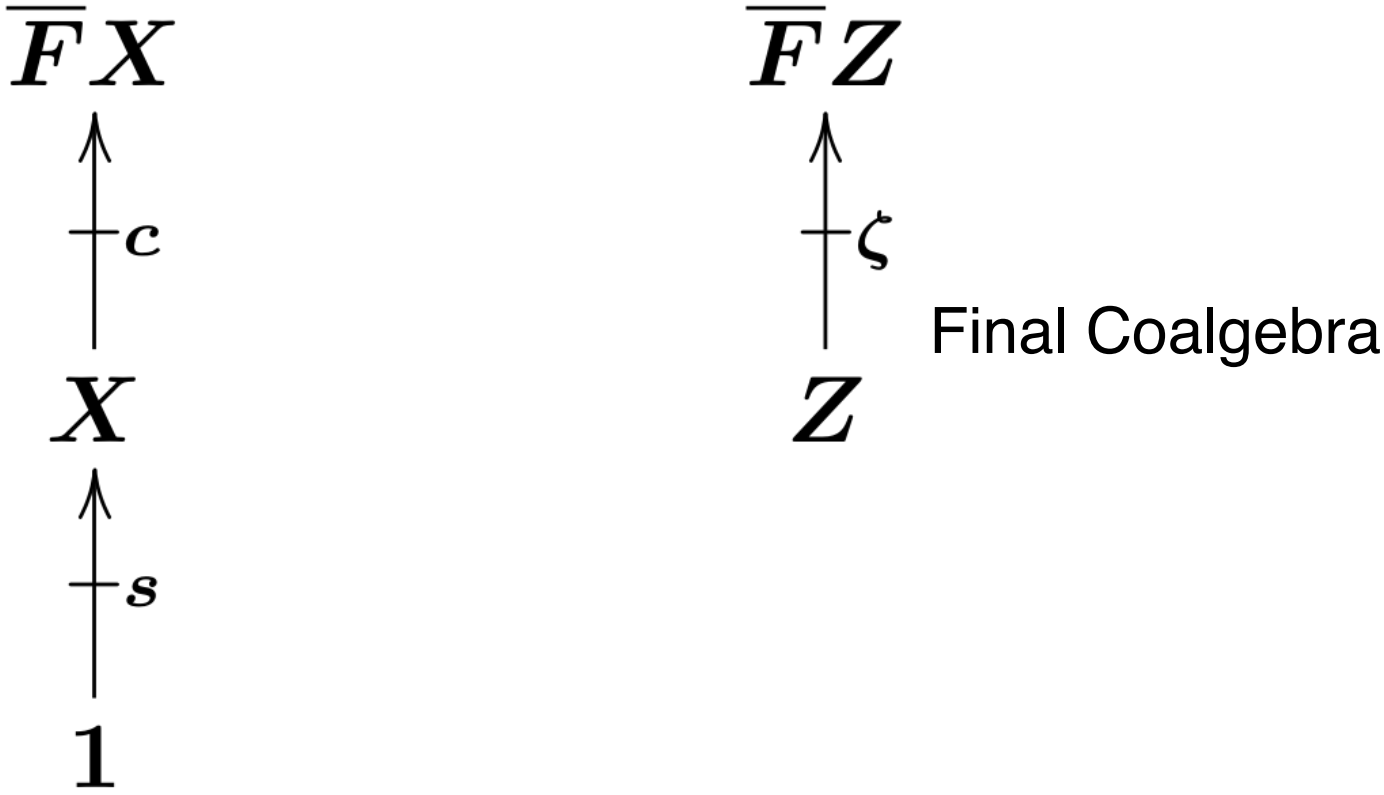
- Kleisli 圏を用いたtrace semantics の一般的な定義
- 様々な遷移系に trace semantics を定義できる



Trace Semantics via Coinduction

(Hasuo, Jacobs, and Sokolova, 2006)

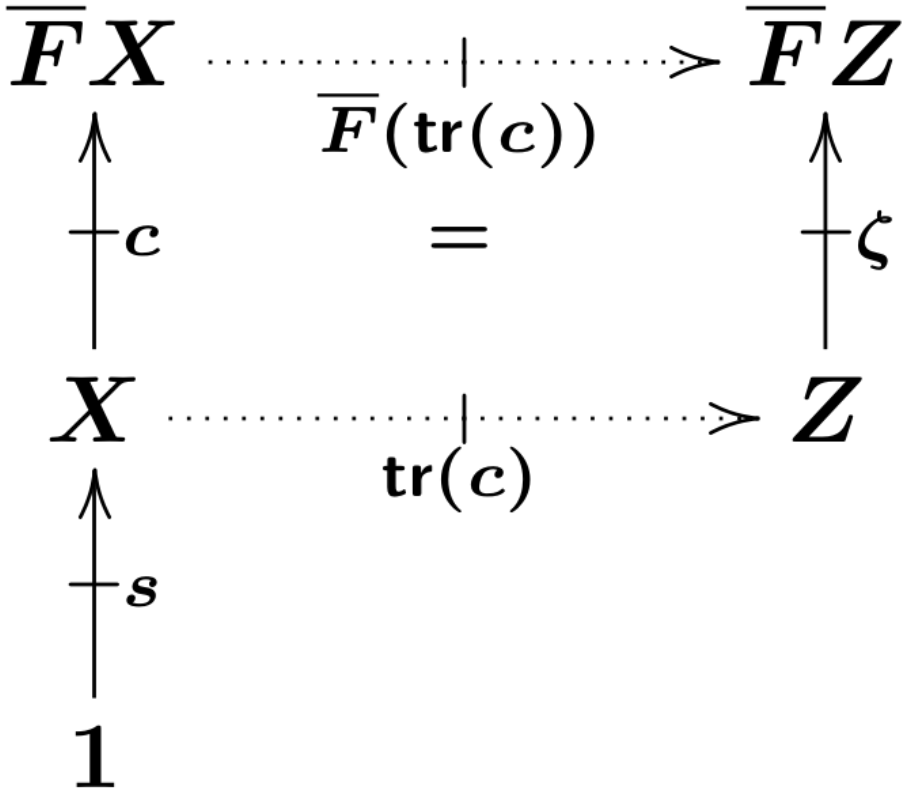
- Kleisli 圏を用いたtrace semantics の一般的な定義
- 様々な遷移系に trace semantics を定義できる



Trace Semantics via Coinduction

(Hasuo, Jacobs, and Sokolova, 2006)

- Kleisli 圏を用いたtrace semantics の一般的な定義
- 様々な遷移系に trace semantics を定義できる

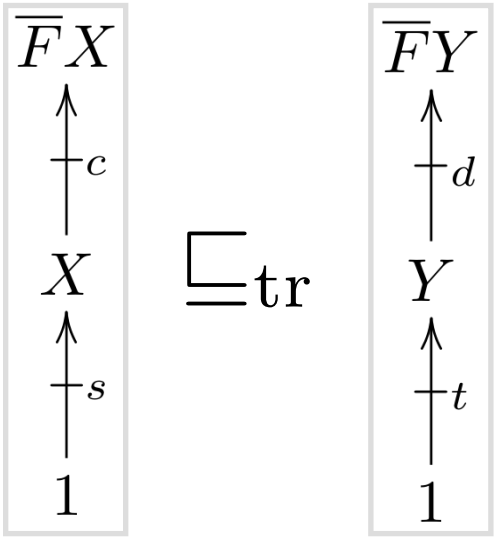


Final Coalgebra

Trace Inclusion via Kleisli Category

- $\mathcal{K}\ell(T)$ の各 homset に順序 (Cppo) を入れる

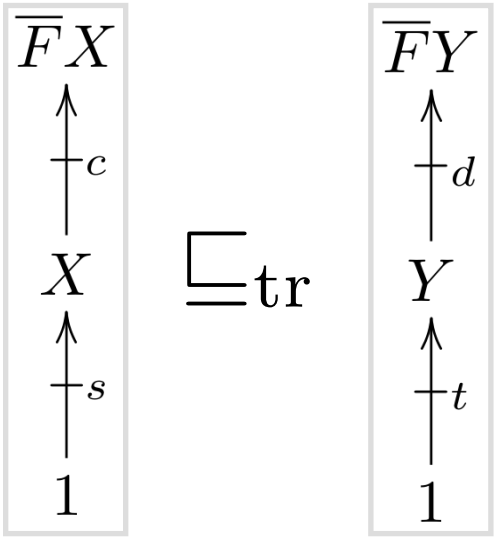
  ⇨ Trace inclusion が定義できる



Trace Inclusion via Kleisli Category

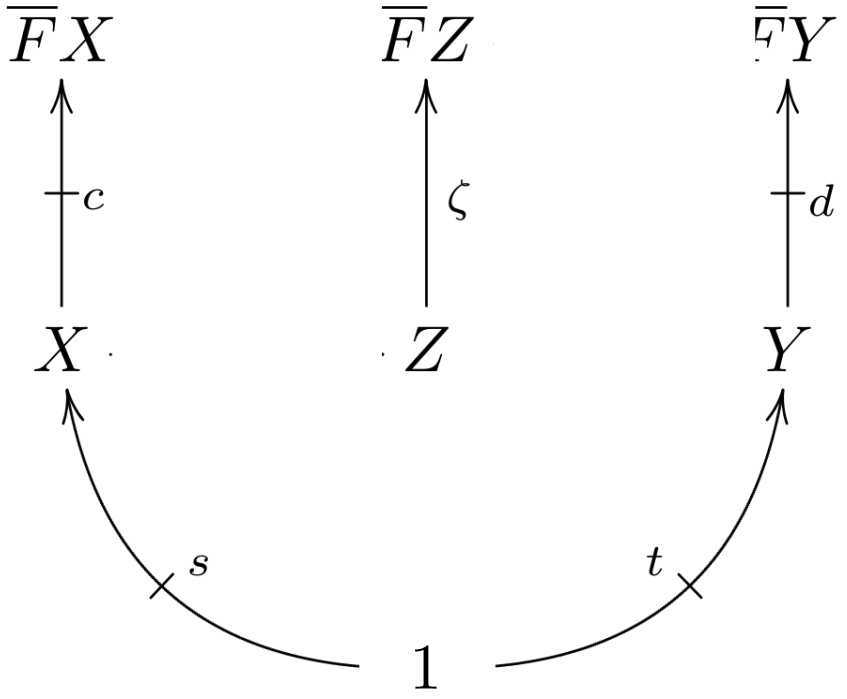
- $\mathcal{K}\ell(T)$ の各 homset に順序 (Cppo) を入れる

  ⇨ Trace inclusion が定義できる



$\sqsubseteq_{\mathrm{tr}}$

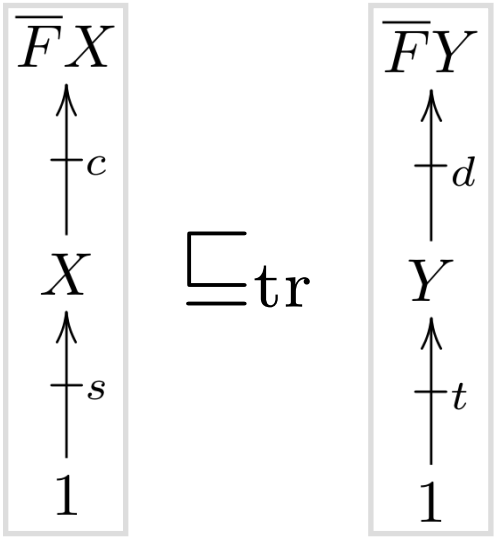
$\overset{\mathrm{def}}{\Leftrightarrow}$



Trace Inclusion via Kleisli Category

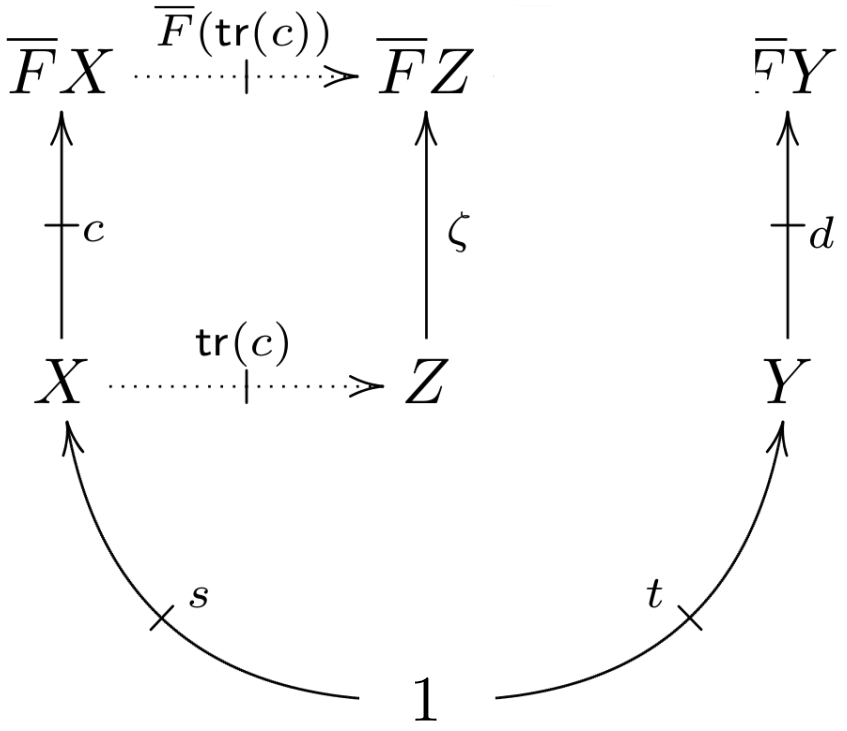
- $\mathcal{K}\ell(T)$ の各 homset に順序 (Cppo) を入れる

  ⇨ Trace inclusion が定義できる



$\sqsubseteq_{\mathrm{tr}}$

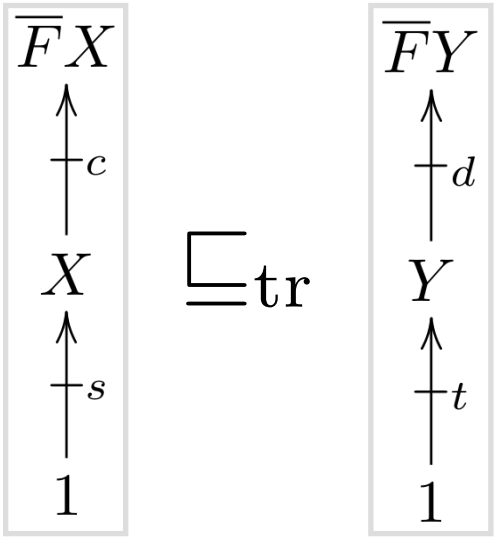
$\overset{\mathrm{def}}{\Leftrightarrow}$



Trace Inclusion via Kleisli Category

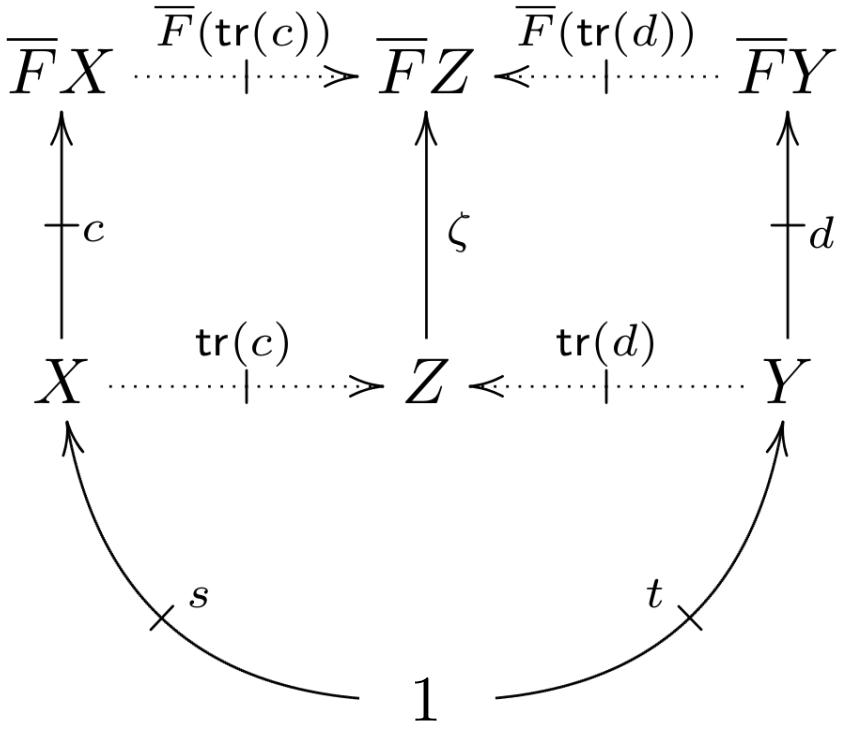
- $\mathcal{K}\ell(T)$ の各 homset に順序 (Cppo) を入れる

  ⇨ Trace inclusion が定義できる



$\sqsubseteq_{\mathrm{tr}}$

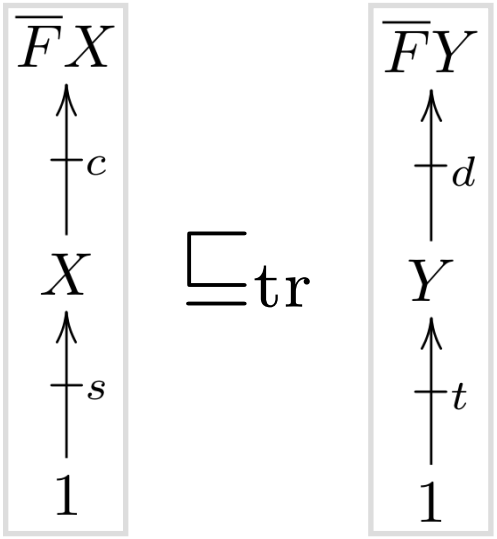
$\overset{\mathrm{def}}{\Leftrightarrow}$



Trace Inclusion via Kleisli Category

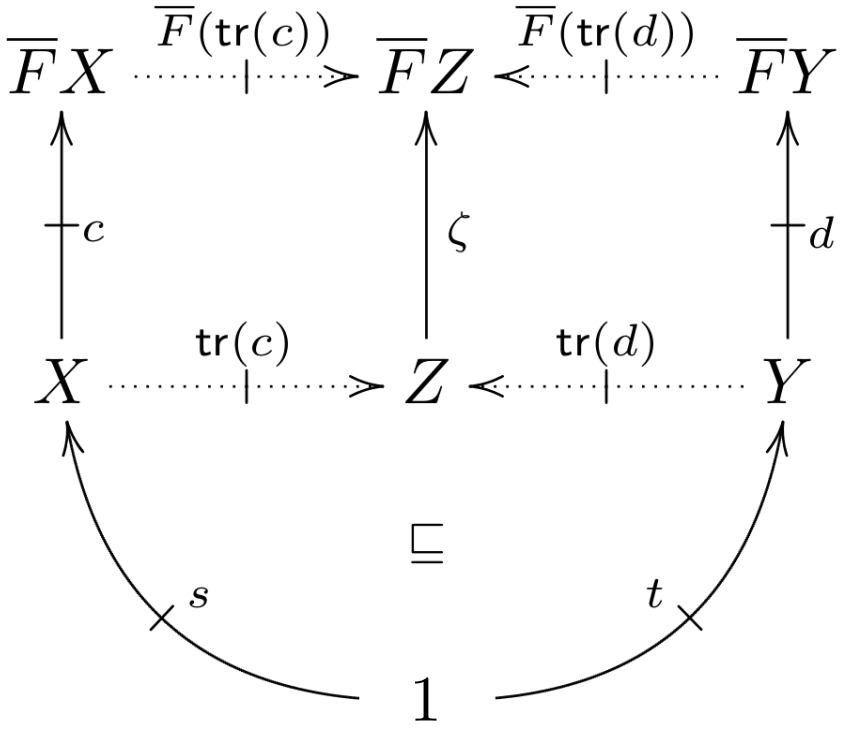
- $\mathcal{K}\ell(T)$ の各 homset に順序 (Cppo) を入れる

  ⇨ Trace inclusion が定義できる



$\sqsubseteq_{\mathrm{tr}}$

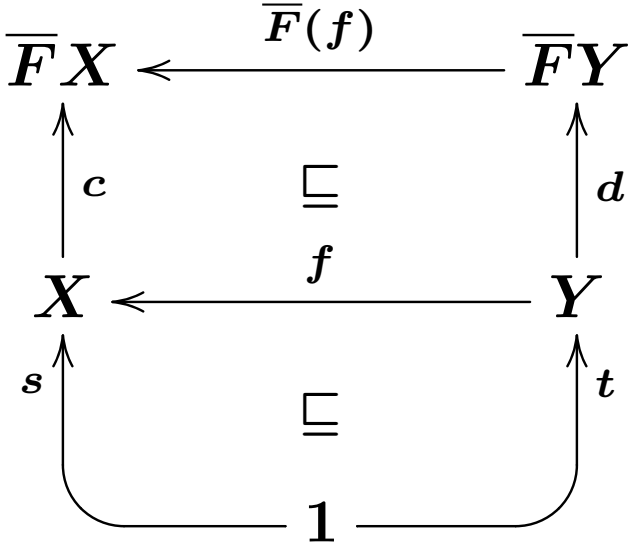
$\overset{\mathrm{def}}{\Leftrightarrow}$



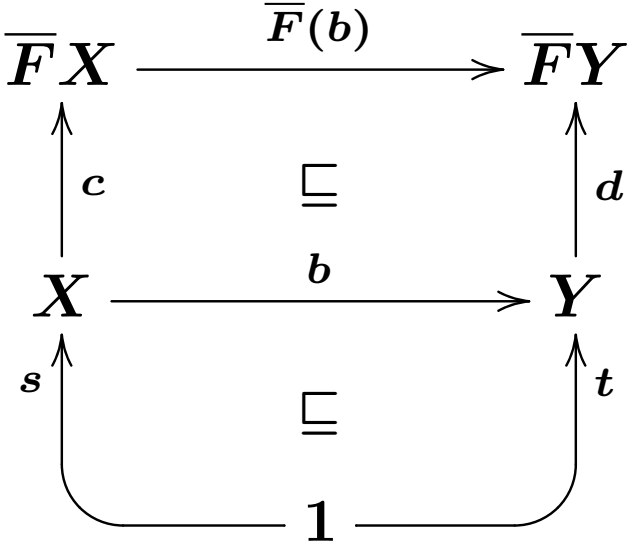
Kleisli Simulation (Hasuo, 2006)

- Kleisli 圏を用いたsimulation の一般的な定義
- 様々な遷移系に simulation (Kleisli simulation)を定義できる

Forward

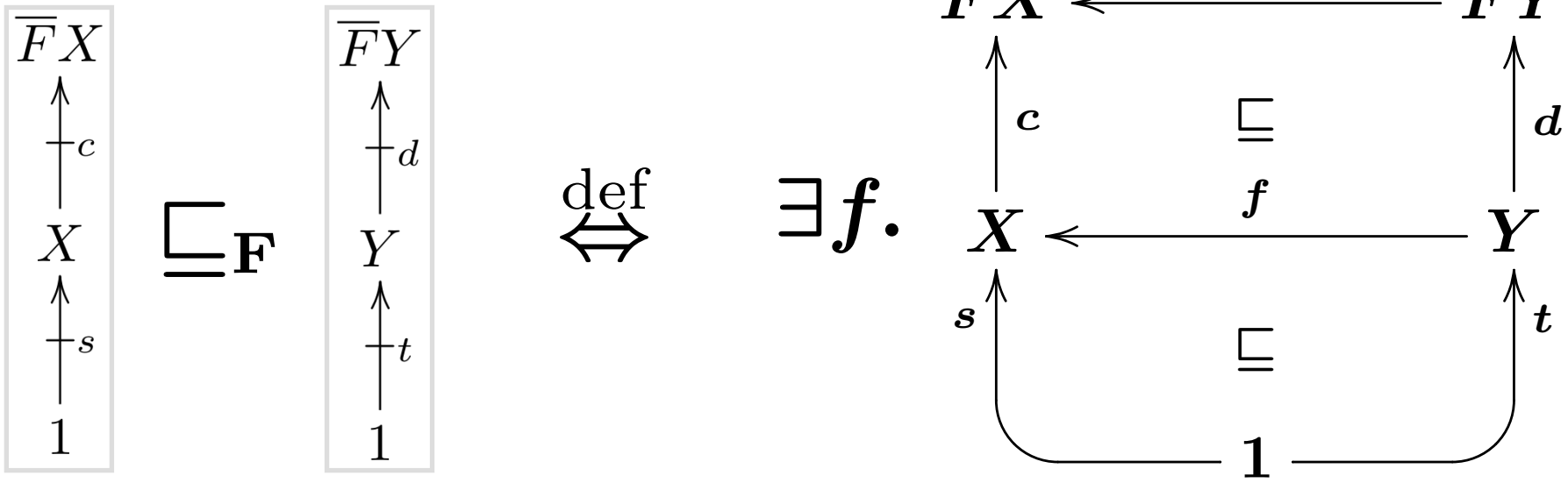


Backward



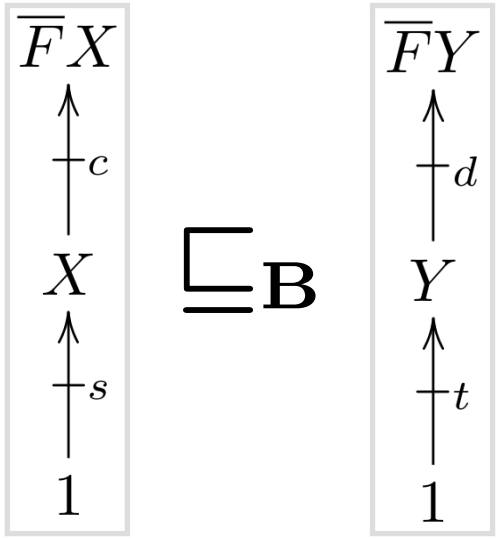
Kleisli Simulation

- Forward Simulation

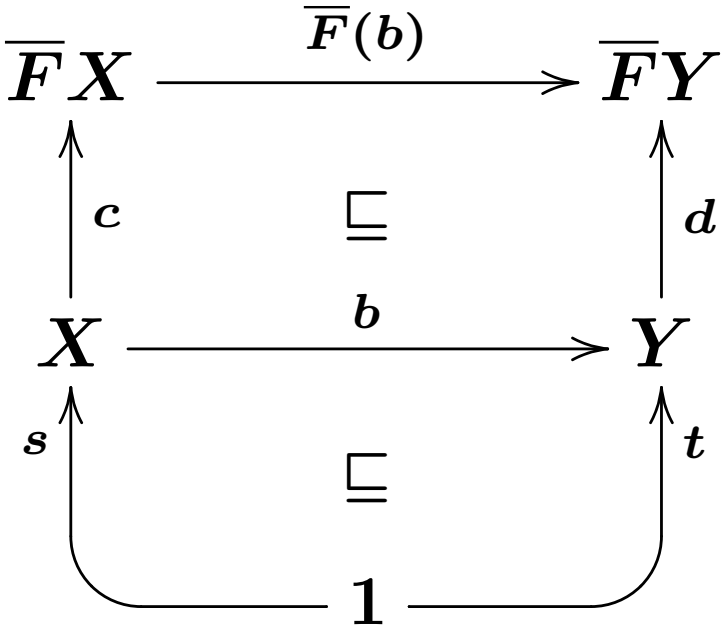


Kleisli Simulation

- Backward Simulation

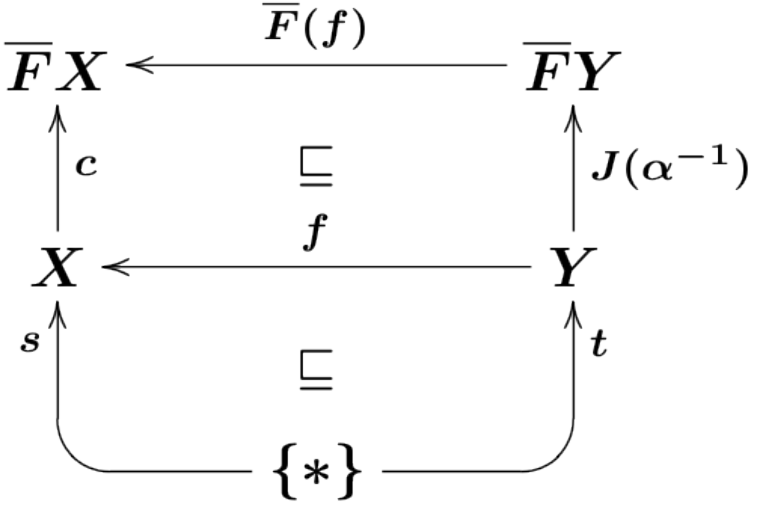


$\overset{\text{def}}{\Leftrightarrow}$ $\exists b.$

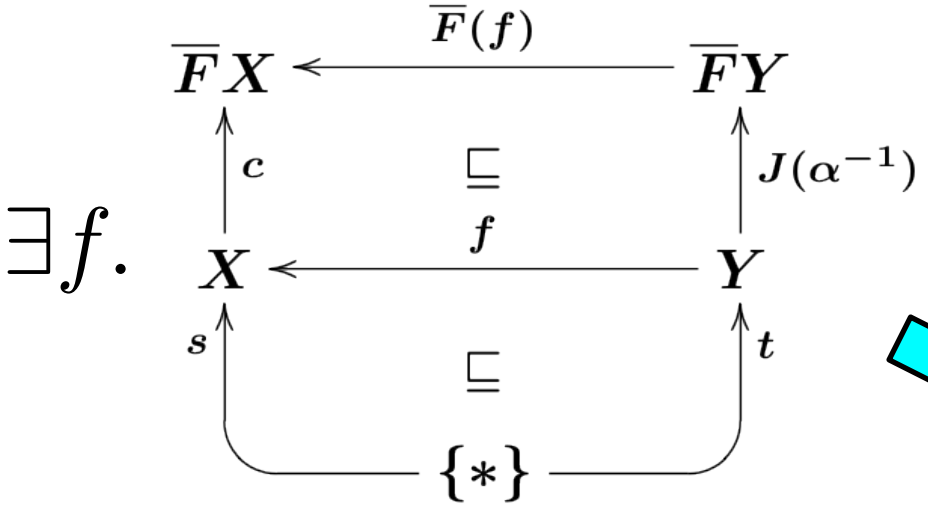


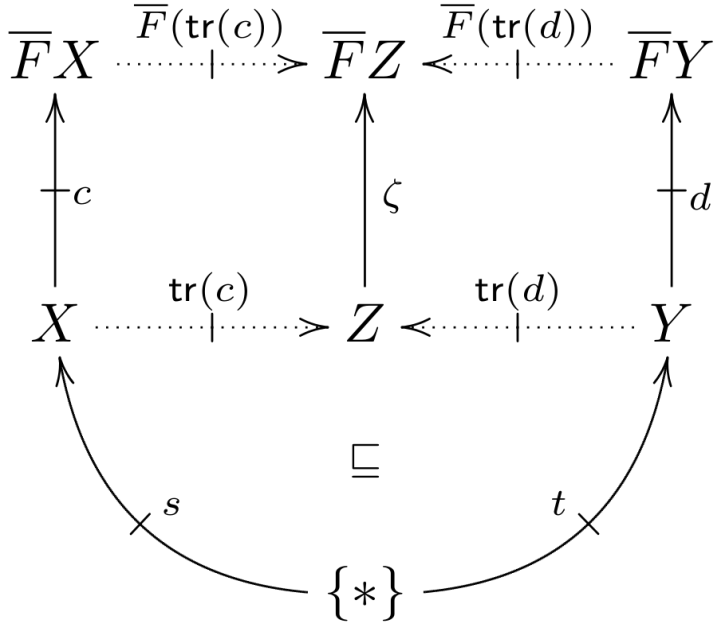
Kleisli Simulation の健全性 (Hasuo, 2006)

$\exists f.$

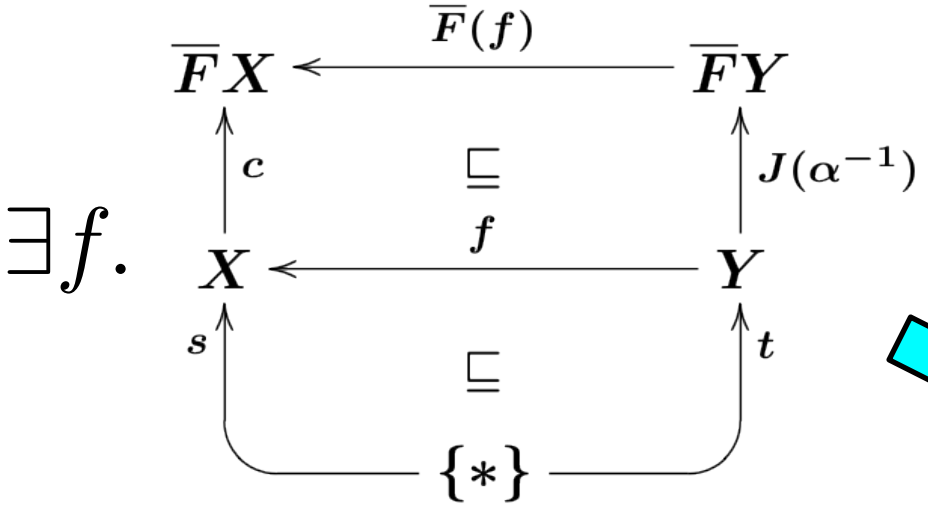


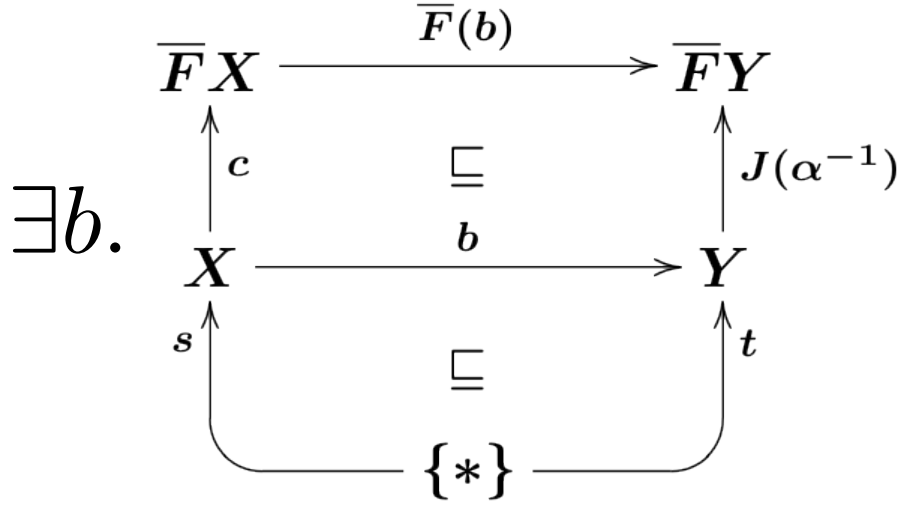
Kleisli Simulation の健全性 (Hasuo, 2006)

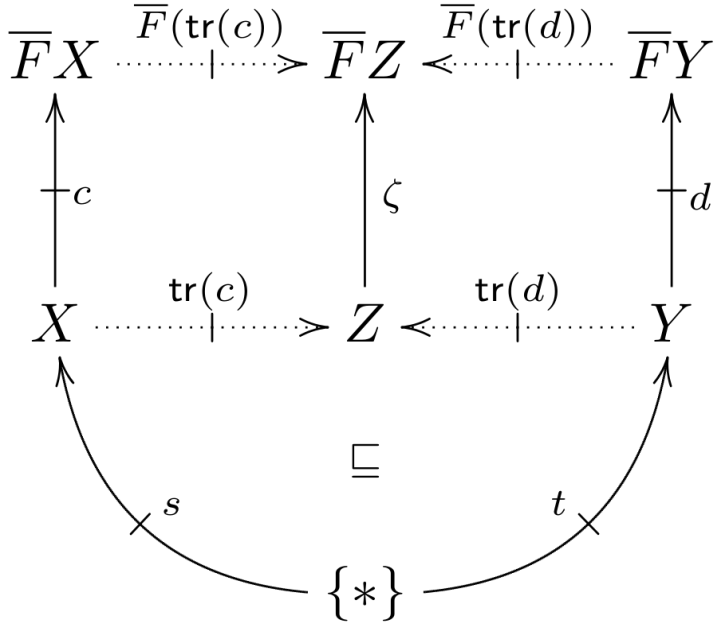




Kleisli Simulation の健全性 (Hasuo, 2006)







Overview

- Simulation
- Kleisli Simulation
  1. Definition
  2. Searching Kleisli Simulation for Probabilistic System
- Contribution
  1. Implementation
  2. Increasing the Chance of Simulation
- Experimental Results and Comparison

卜部夏木（東大理学部情報科学科）

$\mathcal{K}\ell(\mathcal{D})$ での Kleisli Simulation の探索

- $\mathcal{K}\ell(\mathcal{D})$の射は行列で表現できる

$f : X \nrightarrow Y$

卜部夏木（東大理学部情報科学科）

$\mathcal{K}\ell(\mathcal{D})$ での Kleisli Simulation の探索

- $\mathcal{K}\ell(\mathcal{D})$の射は行列で表現できる

$f : X \to \mathcal{D}Y$

卜部夏木（東大理学部情報科学科）

$\mathcal{K}\ell(\mathcal{D})$ での Kleisli Simulation の探索

- $\mathcal{K}\ell(\mathcal{D})$の射は行列で表現できる

$f : X \to \mathcal{D}Y$ ⇨ $M_f = \big(f(x)(y)\big)_{x,y}$

卜部夏木（東大理学部情報科学科）

$\mathcal{K}\ell(D)$ での Kleisli Simulation の探索

- $\mathcal{K}\ell(\mathcal{D})$の射は行列で表現できる

$$f : X \to \mathcal{D}Y \quad \Rightarrow \quad M_f = \big(f(x)(y)\big)_{x,y}$$

- Kleisli Simulation の探索は行列の探索に帰着

卜部夏木（東大理学部情報科学科）

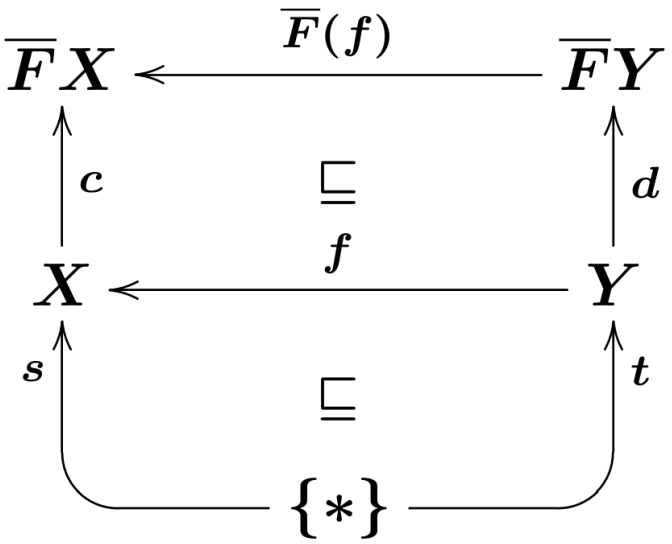
$\mathcal{K}\ell(\mathcal{D})$ での Kleisli Simulation の探索

- $\mathcal{K}\ell(\mathcal{D})$の射は行列で表現できる

$$f : X \to \mathcal{D}Y \quad \Rightarrow \quad M_f = \big(f(x)(y)\big)_{x,y}$$

- Kleisli Simulation の探索は行列の探索に帰着

$\exists f.$



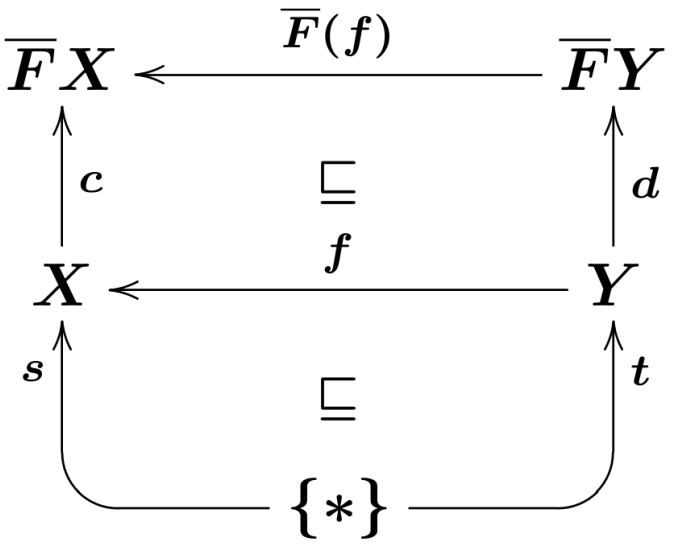
$\mathcal{K}\ell(\mathcal{D})$ での Kleisli Simulation の探索

- $\mathcal{K}\ell(\mathcal{D})$の射は行列で表現できる

$$f : X \to \mathcal{D}Y \quad \Rightarrow \quad M_f = \big(f(x)(y)\big)_{x,y}$$

- Kleisli Simulation の探索は行列の探索に帰着

$\exists f.$



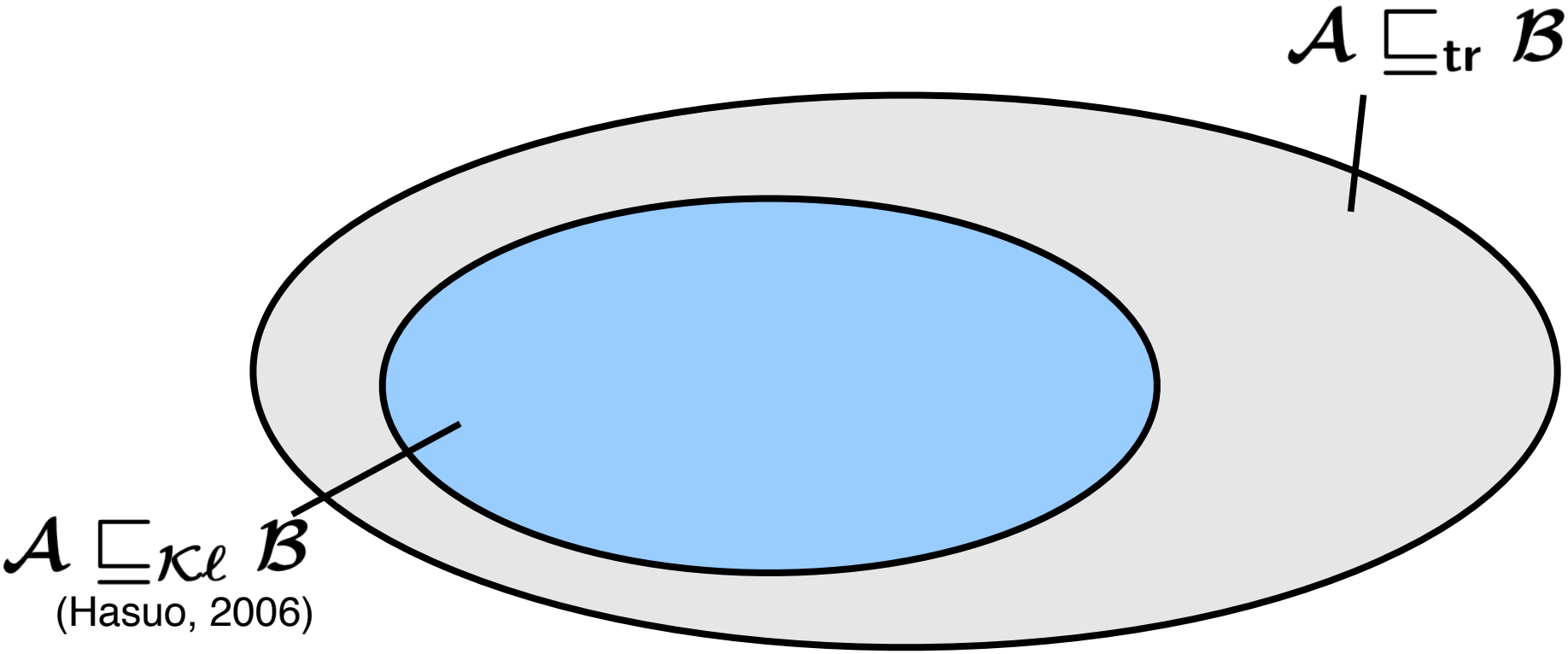
$$\Leftrightarrow \quad \exists M_f.\ M_f M_c \leq M_d M_{\overline{F}f} \wedge M_s \leq M_t M_f$$

ト部夏木（東大理学部情報科学科）

確率的オートマトンのKleisli Simulation

- 確率的オートマトンの trace inclusion は

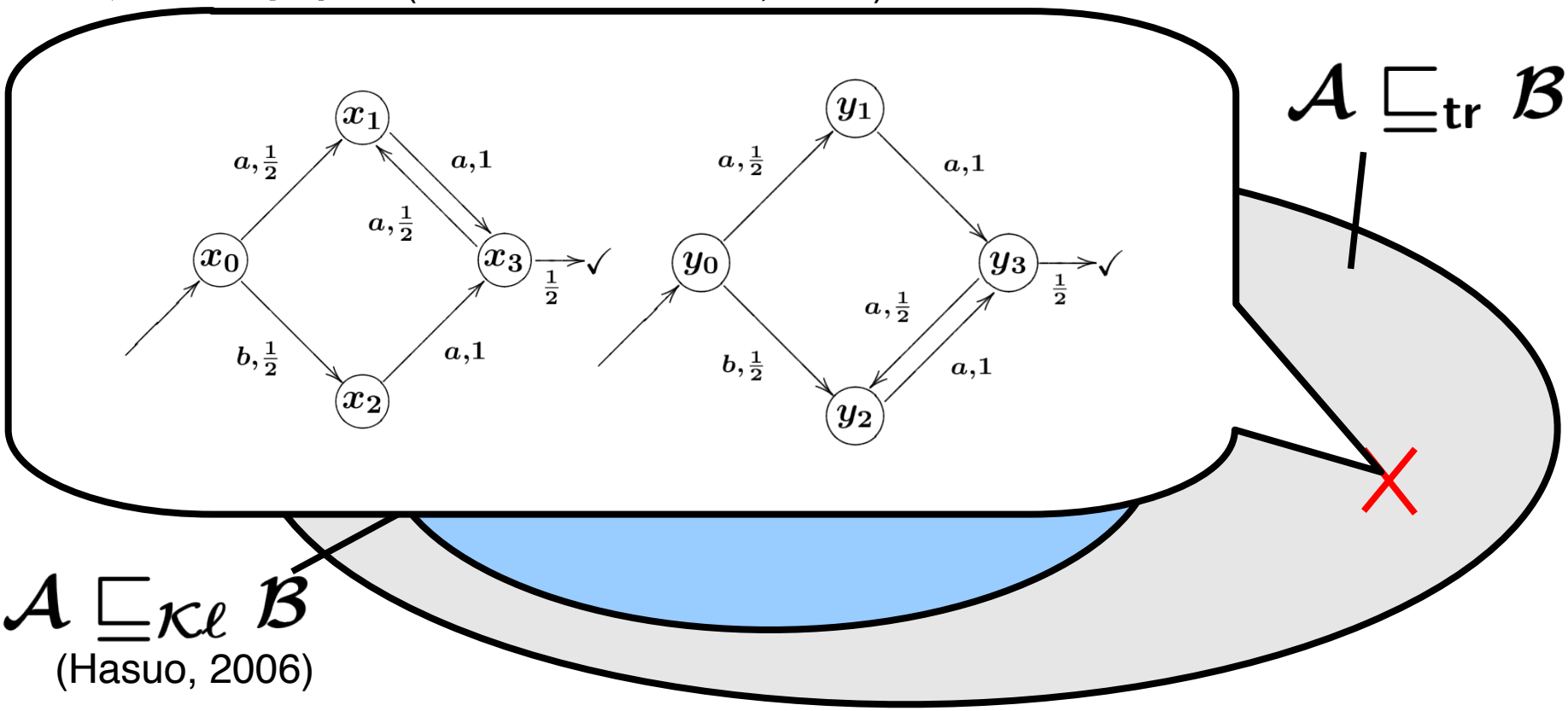
決定不能 (Blondel & Canterni, 2003)



確率的オートマトンのKleisli Simulation

- 確率的オートマトンの trace inclusion は

決定不能 (Blondel & Canterni, 2003)



(Hasuo, 2006)

Overview

- Simulation
- Kleisli Simulation
  1. Definition
  2. Searching Kleisli Simulation for Probabilistic System
- Contribution
  1. Implementation
  2. Increasing the Chance of Simulation
- Experimental Results and Comparison

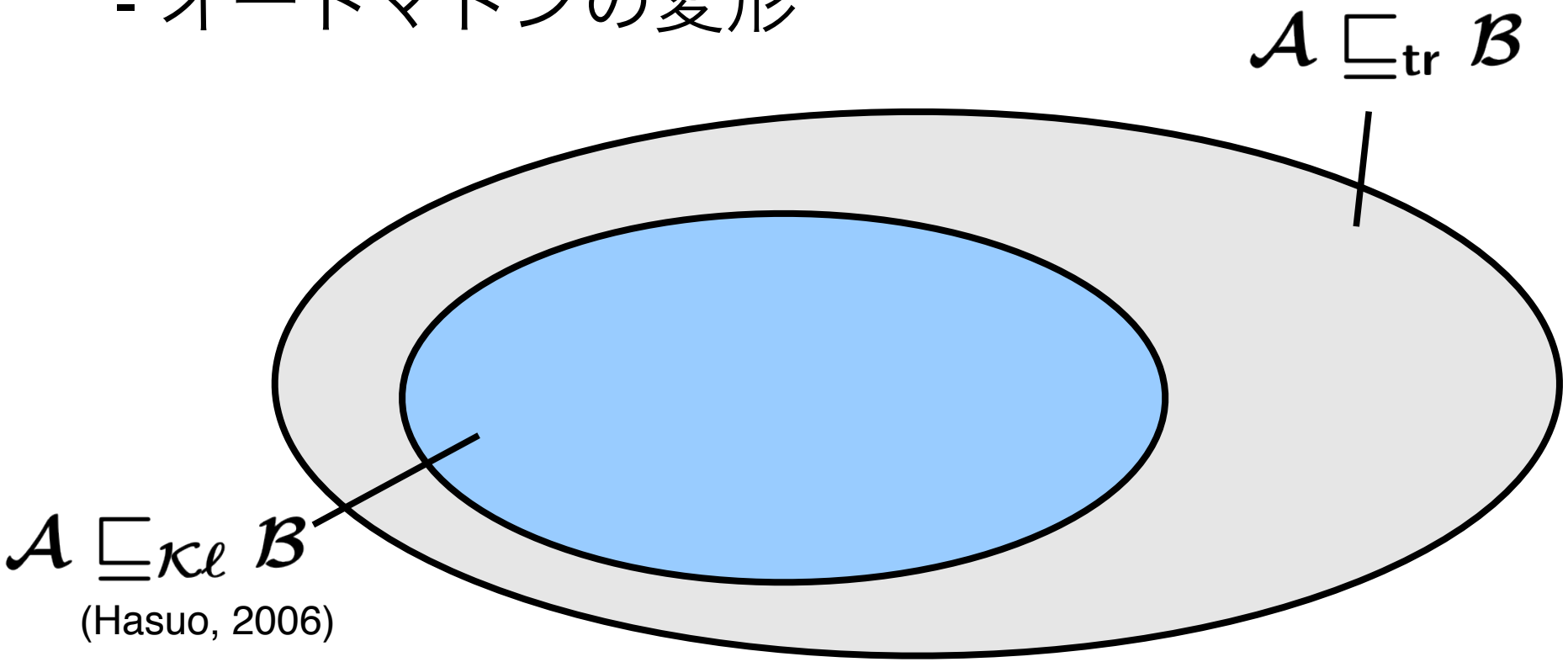
Contributions

1. 線形不等式を解くことで確率的オートマトンの Kleisli simulation を調べるプログラムを実装

Contributions

2. Kleisli simulation を存在しやすくする工夫

- Subdistribution Monad → Multiset Monad
- オートマトンの変形

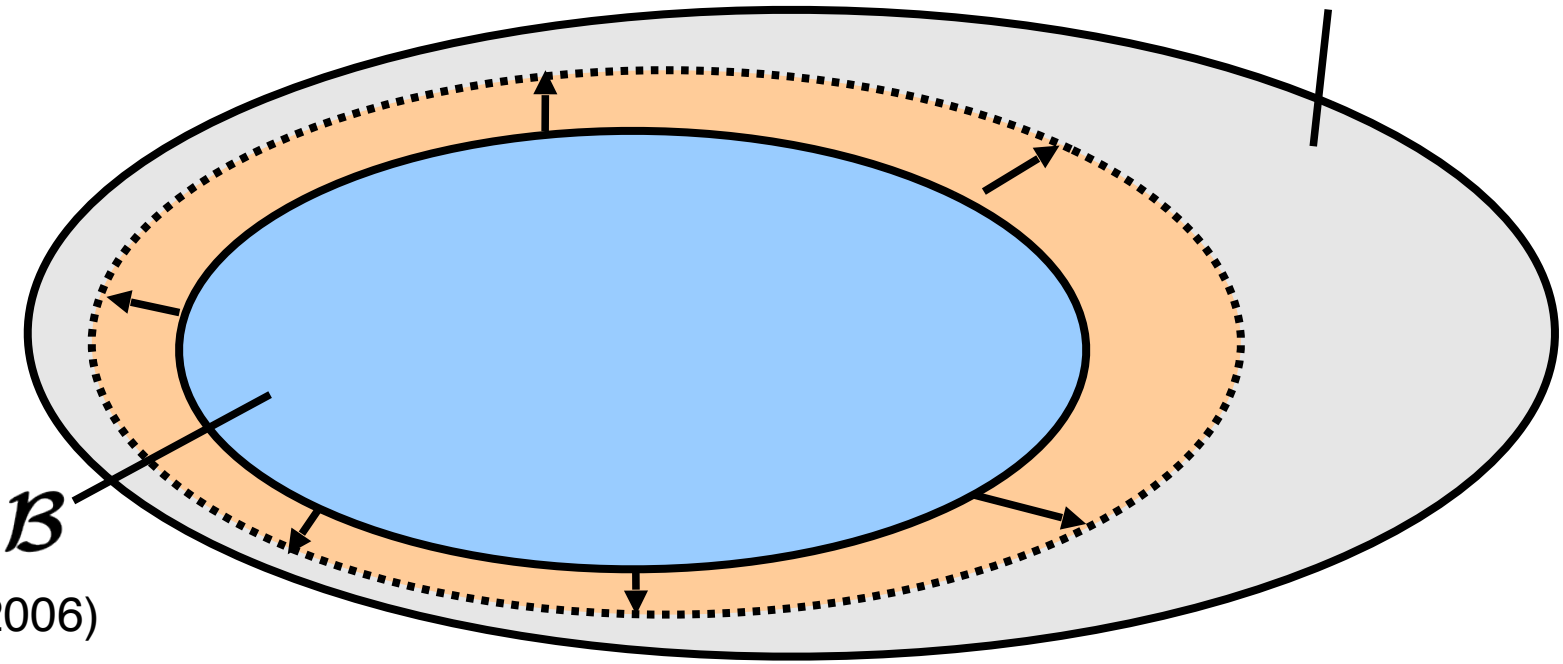


(Hasuo, 2006)

Contributions

2. Kleisli simulation を存在しやすくする工夫

- Subdistribution Monad → Multiset Monad
- オートマトンの変形



(Hasuo, 2006)

Subdistribution Monad → Multiset Monad

$$MX = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

- モナド

  Subdistribution Monad

  $$\mathcal{D}X = \{f : X \to [0, 1] \mid \sum_x f(x) \leq 1\}$$

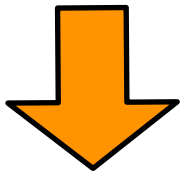
Subdistribution Monad → Multiset Monad

$$MX = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

- モナド

Subdistribution Monad

$$\mathcal{D}X = \{f : X \to [0, 1] \mid \sum_x f(x) \leq 1\}$$



Multiset Monad

$$\mathcal{M}X = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

Subdistribution Monad → Multiset Monad

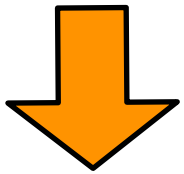
$$MX = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

- モナド

Subdistribution Monad

$$\mathcal{D}X = \{f : X \to [0, 1] \mid \sum_x f(x) \leq 1\}$$

確率的オートマトン



Multiset Monad

$$\mathcal{M}X = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

Subdistribution Monad → Multiset Monad

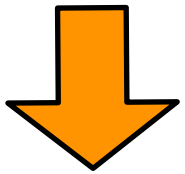
$$MX = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

- モナド

Subdistribution Monad

$$\mathcal{D}X = \{f : X \to [0, 1] \mid \sum_x f(x) \leq 1\}$$

確率的オートマトン



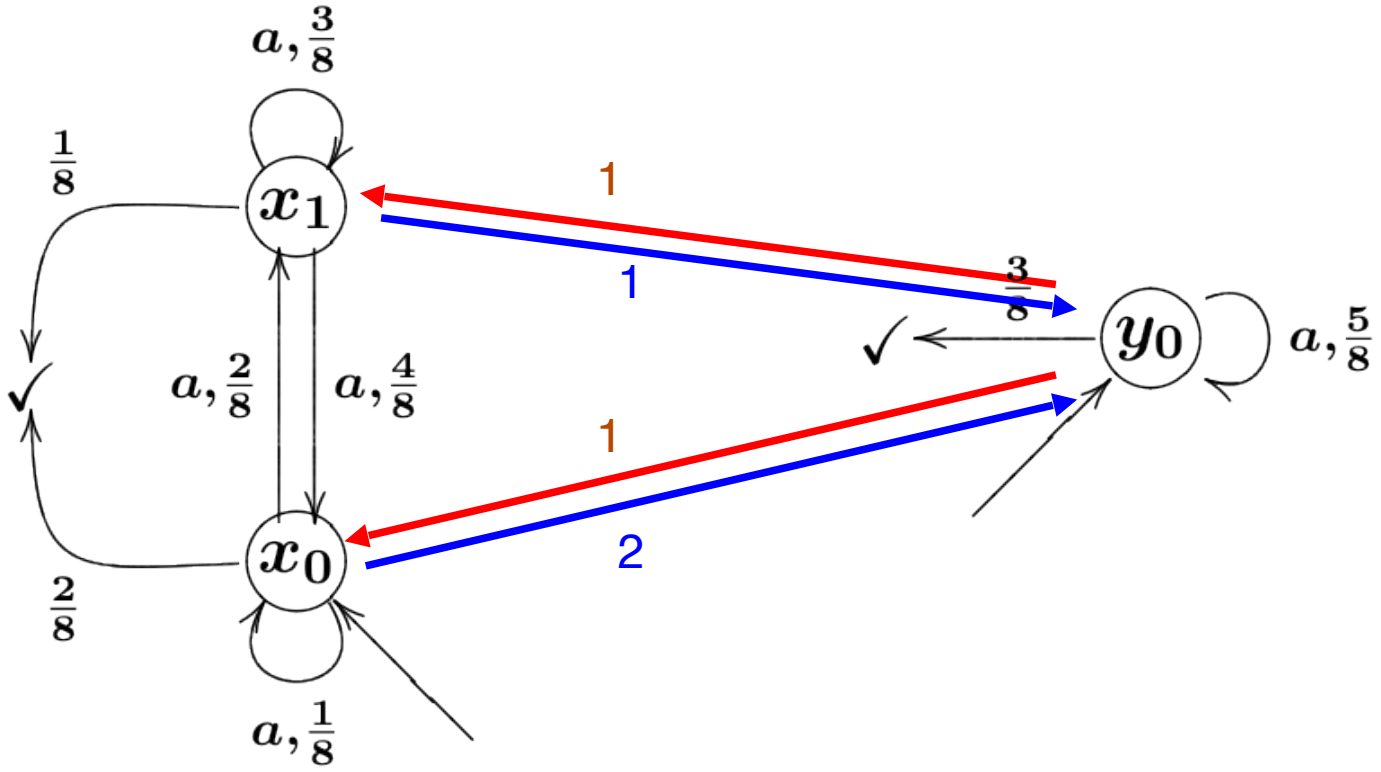
Multiset Monad

$$\mathcal{M}X = \{f : X \to [0, \infty] \mid \mathrm{supp}(f) \text{ is countable}\}$$

実数重み付きオートマトン

確率 → 実数重み

- 新しく Kleisli simulation が見つかることがある



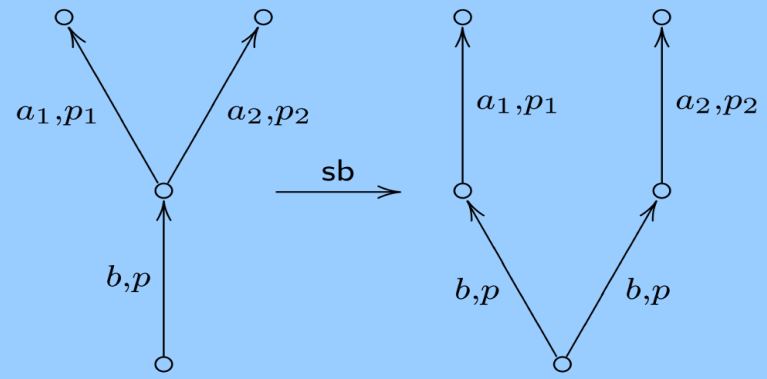
オートマトンの変形

・オートマトンを変形し 、

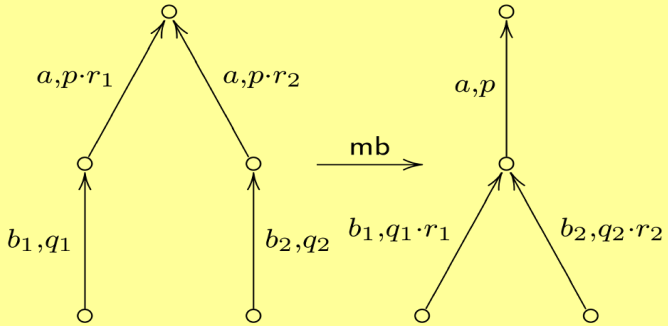
Kleisli simulation を存在しやすく

4種の変形

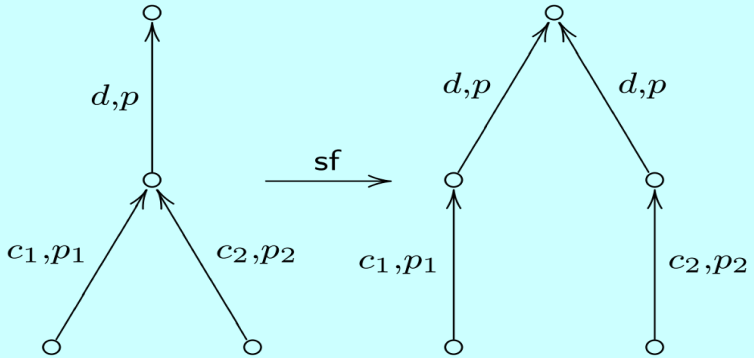
Split Backward



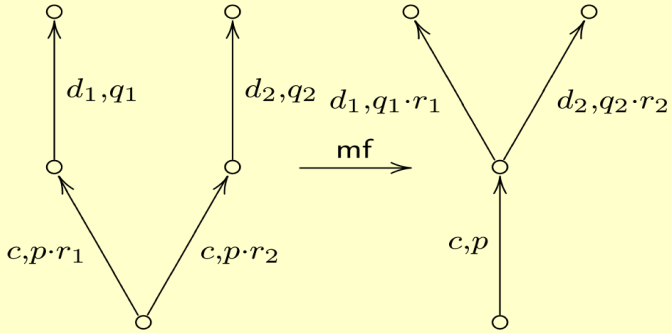
Merge Backward



Split Forward

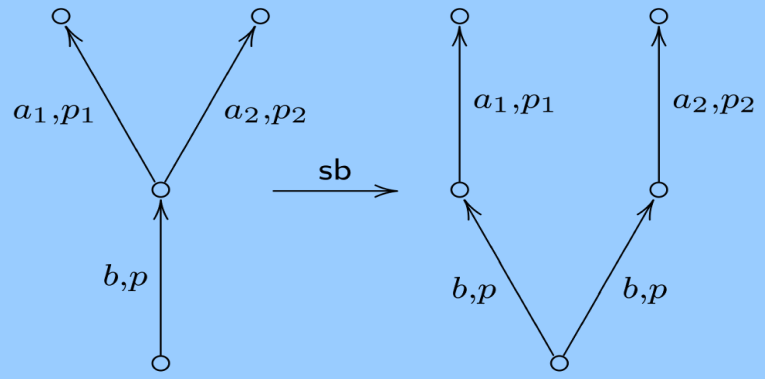


Merge Forward

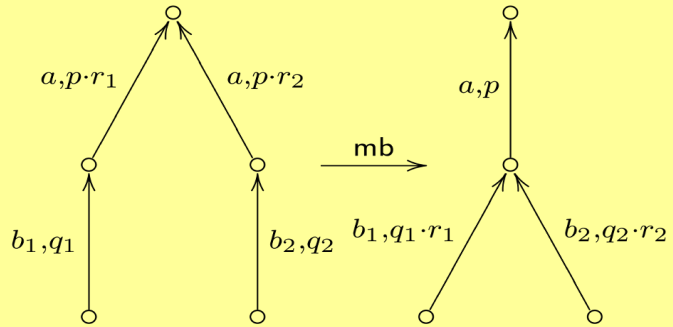


変形の性質

Split Backward

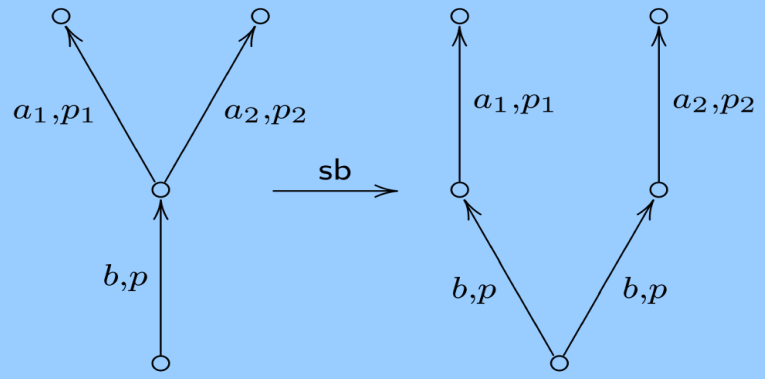


Merge Backward

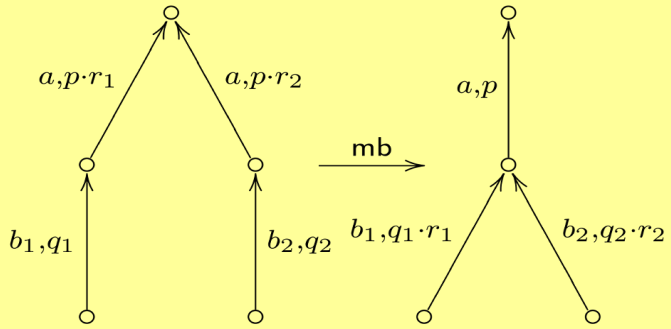


変形の性質

Split Backward



Merge Backward

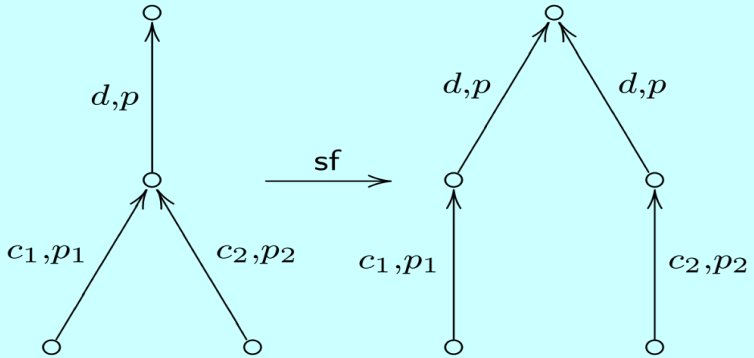


Forward Simulationの左側
Backward Simulationの右側
} に「適切」な変形

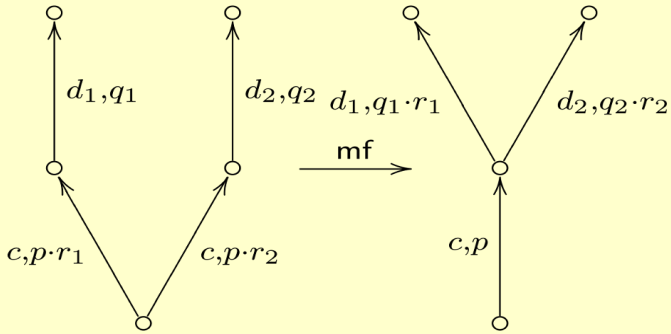
$\mathcal{A} \sqsubseteq_{\mathbf{F}} \mathcal{B}$　　$\mathcal{A} \sqsubseteq_{\mathbf{B}} \mathcal{B}$

変形の性質

Split Forward



Merge Forward

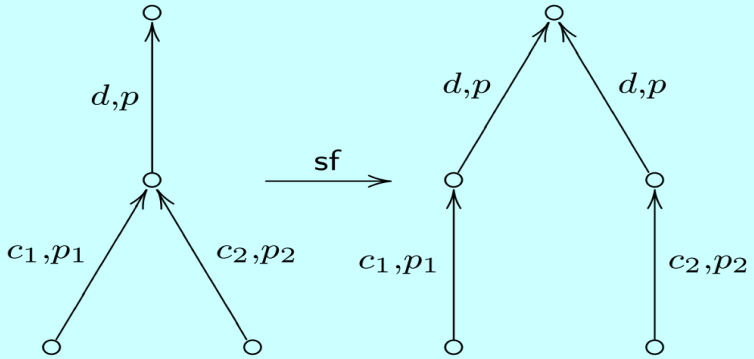


変形の性質

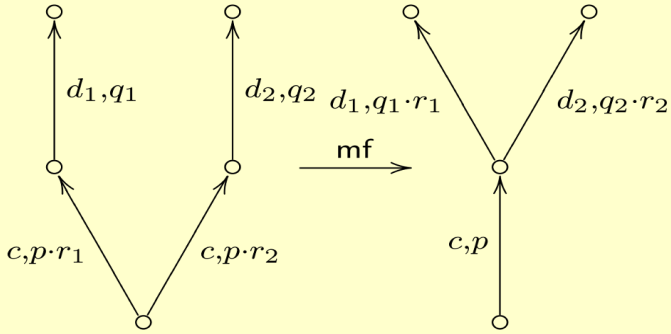
Forward Simulationの右側
Backward Simulationの左側
} に「適切」な変形

$$\mathcal{A} \sqsubseteq_{\mathbf{F}} \mathcal{B} \qquad \mathcal{A} \sqsubseteq_{\mathbf{B}} \mathcal{B}$$

Split Forward



Merge Forward



変形の性質

- 適切な変形は Kleisli simulation の健全性を保つ

$$\left.\begin{array}{l} \mathcal{A}' \sqsubseteq_{\mathbf{F}} \mathcal{B}' \\ \mathcal{A}' \sqsubseteq_{\mathbf{B}} \mathcal{B}' \end{array}\right\rrbracket \Rightarrow \mathcal{A} \sqsubseteq_{\mathbf{tr}} \mathcal{B}$$

- 適切な変形は既存の Kleisli simulation を壊さない

$$\mathcal{A} \sqsubseteq_{\mathbf{F}} \mathcal{B} \quad \Rightarrow \quad \mathcal{A}' \sqsubseteq_{\mathbf{F}} \mathcal{B}'$$

$$\mathcal{A} \sqsubseteq_{\mathbf{B}} \mathcal{B} \quad \Rightarrow \quad \mathcal{A}' \sqsubseteq_{\mathbf{B}} \mathcal{B}'$$

ト部夏木（東大理学部情報科学科）

変形の性質

- これ以上変形できないオートマトンの間の

  Kleisli simulation は**完全**

ト部夏木（東大理学部情報科学科）

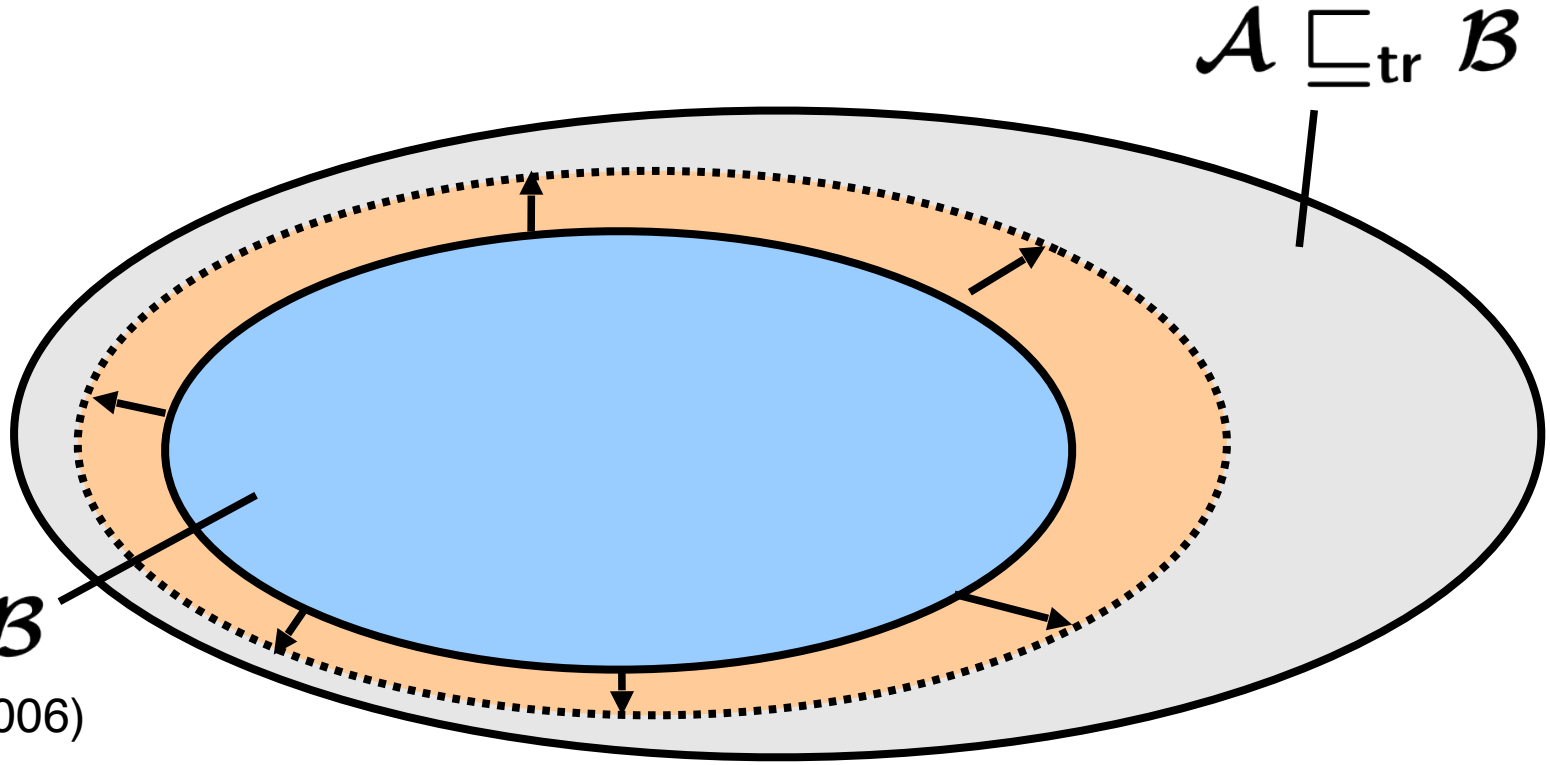
変形の性質

- これ以上変形できないオートマトンの間の Kleisli simulation は**完全**

- 普通は無限回変形できる

変形の性質

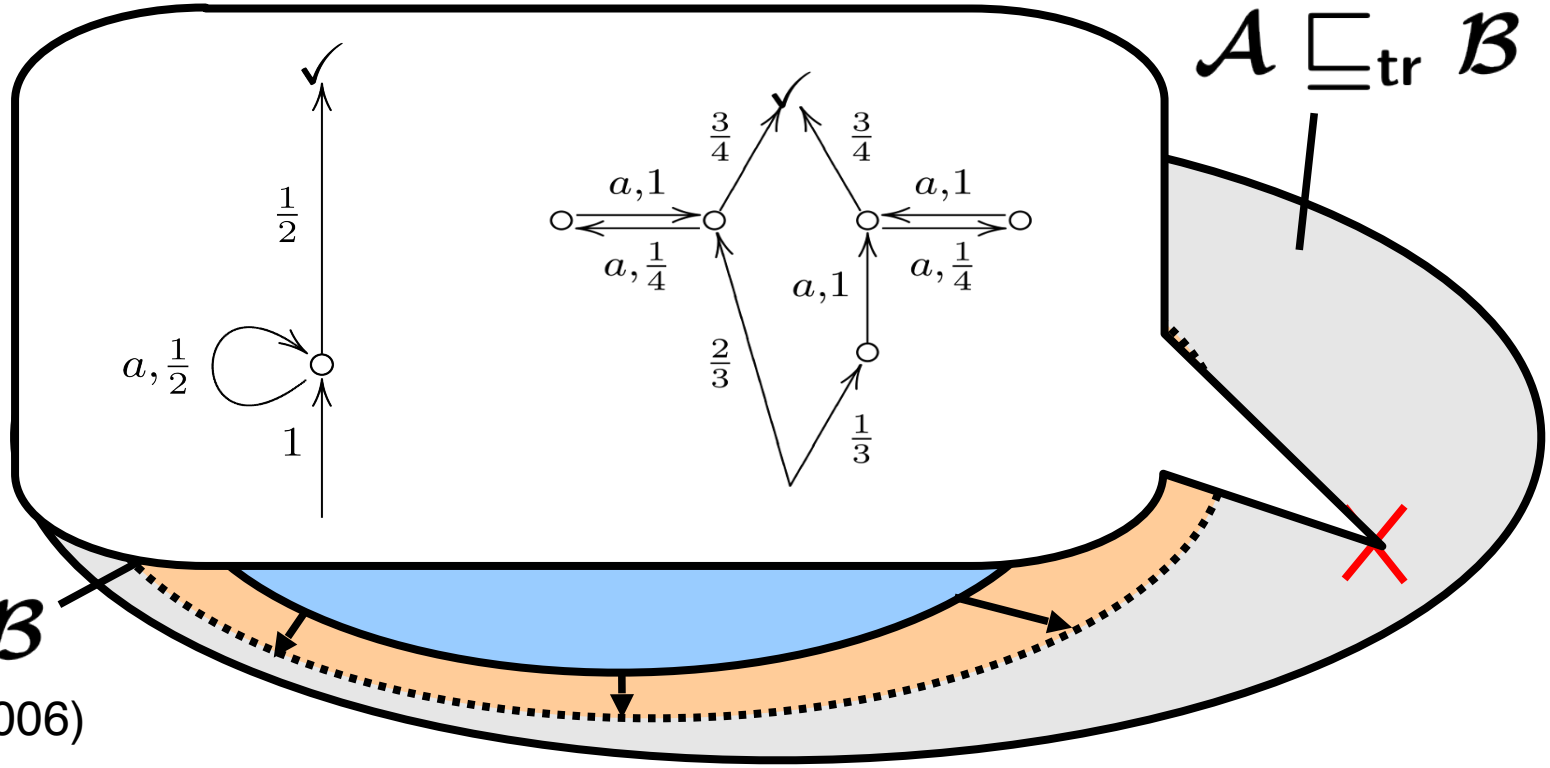
- 確率的オートマトンのtrace inclusion は半決定不能
- 何度変形してもKleisli simulationが
  存在するようにならないことがある



(Hasuo, 2006)

変形の性質

- 確率的オートマトンのtrace inclusion は半決定不能
- 何度変形してもKleisli simulationが
  存在するようにならないことがある



(Hasuo, 2006)

Overview

- Simulation
- Kleisli Simulation
  1. Definition
  2. Searching Kleisli Simulation for Probabilistic System
- Contribution
  1. Implementation
  2. Increasing the Chance of Simulation
- Experimental Results and Comparison

実装

- Kleisli simulation を探すプログラム
  - 線形計画問題ソルバを用いて線形不等式を解く
  - GLPK (GNU Linear Programmin Kit) © Andrew Makhorin
  - 単体法
- オートマトンを変形するプログラム

実験結果

- Grade protocol の正しさの検証
- オートマトンの等価性を調べることで検証できる

(Kiefer et al. 2011)

parm		protocol		specific			type		time(sec)		space	ex
G	S	st	tr	st	tr	al	dir	f/b	user	real	(GB)	
2	8	578	1522	130	642	11	$p \to s$	fwd	1.77	2.00	1.21	E
								bwd	2.03	2.20	1.21	N
							$s \to p$	fwd	1.94	2.08	1.14	N
								bwd	1.72	1.88	1.22	E
2	10	1102	2982	202	1202	13	$p \to s$	fwd	9.42	9.95	4.05	E
								bwd	11.51	12.00	4.08	N
							$s \to p$	fwd	10.99	11.45	3.82	N
								bwd	9.25	9.81	4.09	E
2	12	1874	5162	290	2018	15	$p \to s$	fwd	38.60	40.16	11.51	E
								bwd	49.03	50.38	11.60	N
							$s \to p$	fwd	47.99	49.40	10.83	N
								bwd	38.34	39.78	11.63	E
3	8	1923	7107	243	2163	20	$p \to s$	fwd	44.43	62.04	12.26	E
								bwd	56.92	79.31	12.59	N
							$s \to p$	fwd	55.48	57.12	12.27	N
								bwd	44.11	67.60	12.64	E
4	6	1636	7468	196	1924	23	$p \to s$	fwd	30.28	31.64	10.39	E
								bwd	38.79	39.97	10.49	N
							$s \to p$	fwd	37.86	39.10	9.79	N
								bwd	29.94	31.25	10.49	E

実験結果

- Grade protocol の正しさの検証
- オートマトンの等価性を調べることで検証できる

(Kiefer et al. 2011)

parm G	parm S	protocol st	protocol tr	specific st	specific tr	specific al	type dir	f/b	time(sec) user	time(sec) real	space (GB)	ex
2	8	578	1522	130	642	11	$p \to s$	fwd	1.77	2.00	1.21	E
								bwd	2.03	2.20	1.21	N
							$s \to p$	fwd	1.94	2.08	1.14	N
								bwd	1.72	1.88	1.22	E
2	10	1102	2982	202	1202	13	$p \to s$	fwd	9.42	9.95	4.05	E
								bwd	11.51	12.00	4.08	N
							$s \to p$	fwd	10.99	11.45	3.82	N
								bwd	9.25	9.81	4.09	E
2	12	1874	5162	290	2018	15	$p \to s$	fwd	38.60	40.16	11.51	E
								bwd	49.03	50.38	11.60	N
							$s \to p$	fwd	47.99	49.40	10.83	N
								bwd	38.34	39.78	11.63	E
3	8	1923	7107	243	2163	20	$p \to s$	fwd	44.43	62.04	12.26	E
								bwd	56.92	79.31	12.59	N
							$s \to p$	fwd	55.48	57.12	12.27	N
								bwd	44.11	67.60	12.64	E
4	6	1636	7468	196	1924	23	$p \to s$	fwd	30.28	31.64	10.39	E
								bwd	38.79	39.97	10.49	N
							$s \to p$	fwd	37.86	39.10	9.79	N
								bwd	29.94	31.25	10.49	E

- Space consuming

実験結果

- Grade protocol の正しさの検証
- オートマトンの等価性を調べることで検証できる

(Kiefer et al. 2011)

parm		protocol		specific			type		time(sec)		space	ex
G	S	st	tr	st	tr	al	dir	f/b	user	real	(GB)	
2	8	578	1522	130	642	11	$p \to s$	fwd	1.77	2.00	1.21	E
								bwd	2.03	2.20	1.21	N
							$s \to p$	fwd	1.94	2.08	1.14	N
								bwd	1.72	1.88	1.22	E
2	10	1102	2982	202	1202	13	$p \to s$	fwd	9.42	9.95	4.05	E
								bwd	11.51	12.00	4.08	N
							$s \to p$	fwd	10.99	11.45	3.82	N
								bwd	9.25	9.81	4.09	E
2	12	1874	5162	290	2018	15	$p \to s$	fwd	38.60	40.16	11.51	E
								bwd	49.03	50.38	11.60	N
							$s \to p$	fwd	47.99	49.40	10.83	N
								bwd	38.34	39.78	11.63	E
3	8	1923	7107	243	2163	20	$p \to s$	fwd	44.43	62.04	12.26	E
								bwd	56.92	79.31	12.59	N
							$s \to p$	fwd	55.48	57.12	12.27	N
								bwd	44.11	67.60	12.64	E
4	6	1636	7468	196	1924	23	$p \to s$	fwd	30.28	31.64	10.39	E
								bwd	38.79	39.97	10.49	N
							$s \to p$	fwd	37.86	39.10	9.79	N
								bwd	29.94	31.25	10.49	E

- Space consuming
- Kieferらの実装より遅い
  - Trace inclusionは trace equivalenceより難しい

実験結果

- Crowds protocol の probable innocence の検証
- Trace inclusion を調べることで検証できる (Hasuo, Kawabe, Sakurada, 2010)
- 変形が必要になる場合が発生→変形・探索を繰り返す

parm			orig		inno				time(sec)		space	iter
n	c	p_f	st	tr	st	tr	al	f/b	user	real	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	fwd	52.48	52.56	0.01	2L
								bwd	0.01	0.06	0.01	2L
7	1	$\frac{3}{4}$	9	88	9	118	26	fwd	0.15	0.21	0.03	2L
								bwd	0.02	0.07	0.01	2L
10	2	$\frac{4}{5}$	12	224	12	280	54	fwd	802.47	803.64	0.35	2L
								bwd	0.05	0.12	0.03	2L
20	6	$\frac{4}{5}$	22	1514	22	1696	238	fwd	TO			
								bwd	1.32	1.51	0.78	2L
30	6	$\frac{4}{5}$	32	4732	32	5112	550	fwd	SF			
								bwd	11.84	12.82	5.99	2L

実験結果

- Crowds protocol の probable innocence の検証
- Trace inclusion を調べることで検証できる (Hasuo, Kawabe, Sakurada, 2010)
- 変形が必要になる場合が発生→変形・探索を繰り返す

parm			orig		inno				time(sec)		space	iter
n	c	p_f	st	tr	st	tr	al	f/b	user	real	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	fwd	52.48	52.56	0.01	2L
								bwd	0.01	0.06	0.01	2L
7	1	$\frac{3}{4}$	9	88	9	118	26	fwd	0.15	0.21	0.03	2L
								bwd	0.02	0.07	0.01	2L
10	2	$\frac{4}{5}$	12	224	12	280	54	fwd	802.47	803.64	0.35	2L
								bwd	0.05	0.12	0.03	2L
20	6	$\frac{4}{5}$	22	1514	22	1696	238	fwd	TO			
								bwd	1.32	1.51	0.78	2L
30	6	$\frac{4}{5}$	32	4732	32	5112	550	fwd	SF			
								bwd	11.84	12.82	5.99	2L

実験結果

- Crowds protocol の probable innocence の検証
- Trace inclusion を調べることで検証できる (Hasuo, Kawabe, Sakurada, 2010)
- 変形が必要になる場合が発生→変形・探索を繰り返す

parm			orig		inno				time(sec)		space	iter
n	c	p_f	st	tr	st	tr	al	f/b	user	real	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	fwd	52.48	52.56	0.01	2L
								bwd	0.01	0.06	0.01	2L
7	1	$\frac{3}{4}$	9	88	9	118	26	fwd	0.15	0.21	0.03	2L
								bwd	0.02	0.07	0.01	2L
10	2	$\frac{4}{5}$	12	224	12	280	54	fwd	802.47	803.64	0.35	2L
								bwd	0.05	0.12	0.03	2L
20	6	$\frac{4}{5}$	22	1514	22	1696	238	fwd	TO			
								bwd	1.32	1.51	0.78	2L
30	6	$\frac{4}{5}$	32	4732	32	5112	550	fwd	SF			
								bwd	11.84	12.82	5.99	2L

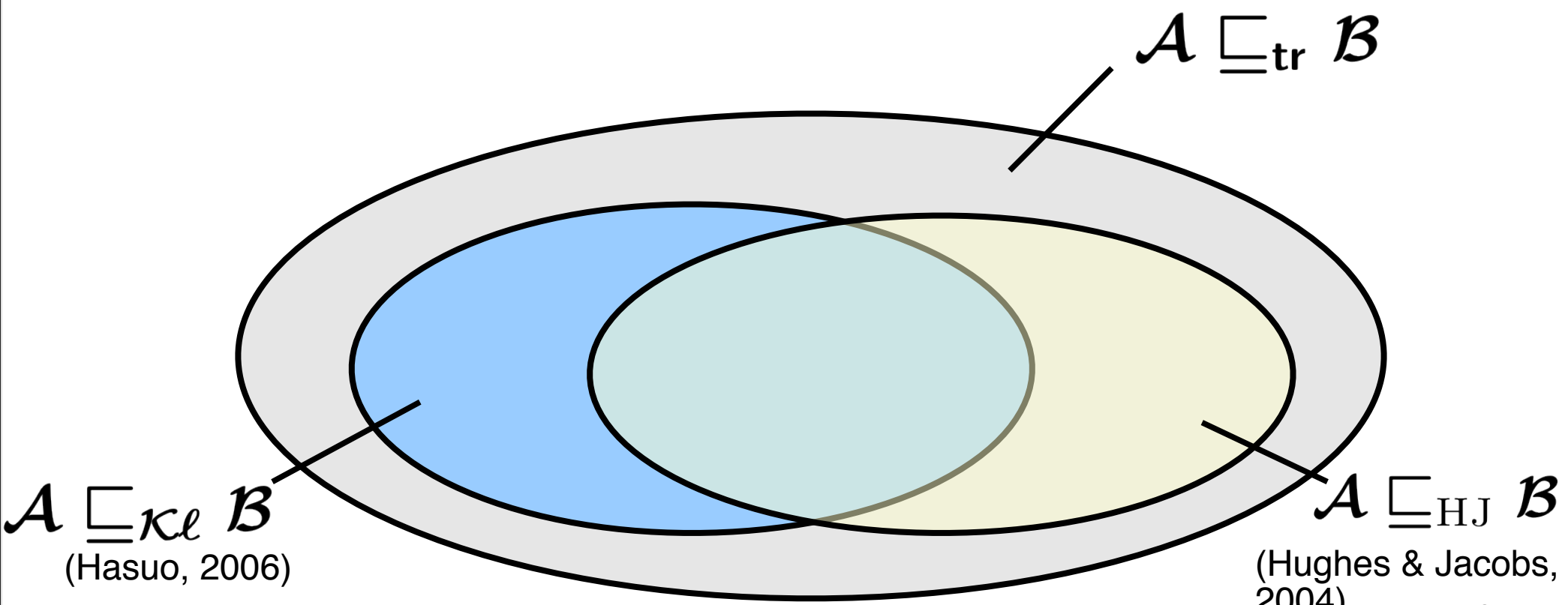
実験結果

- Crowds protocol の probable innocence の検証
- Trace inclusion を調べることで検証できる (Hasuo, Kawabe, Sakurada, 2010)
- 変形が必要になる場合が発生→変形・探索を繰り返す

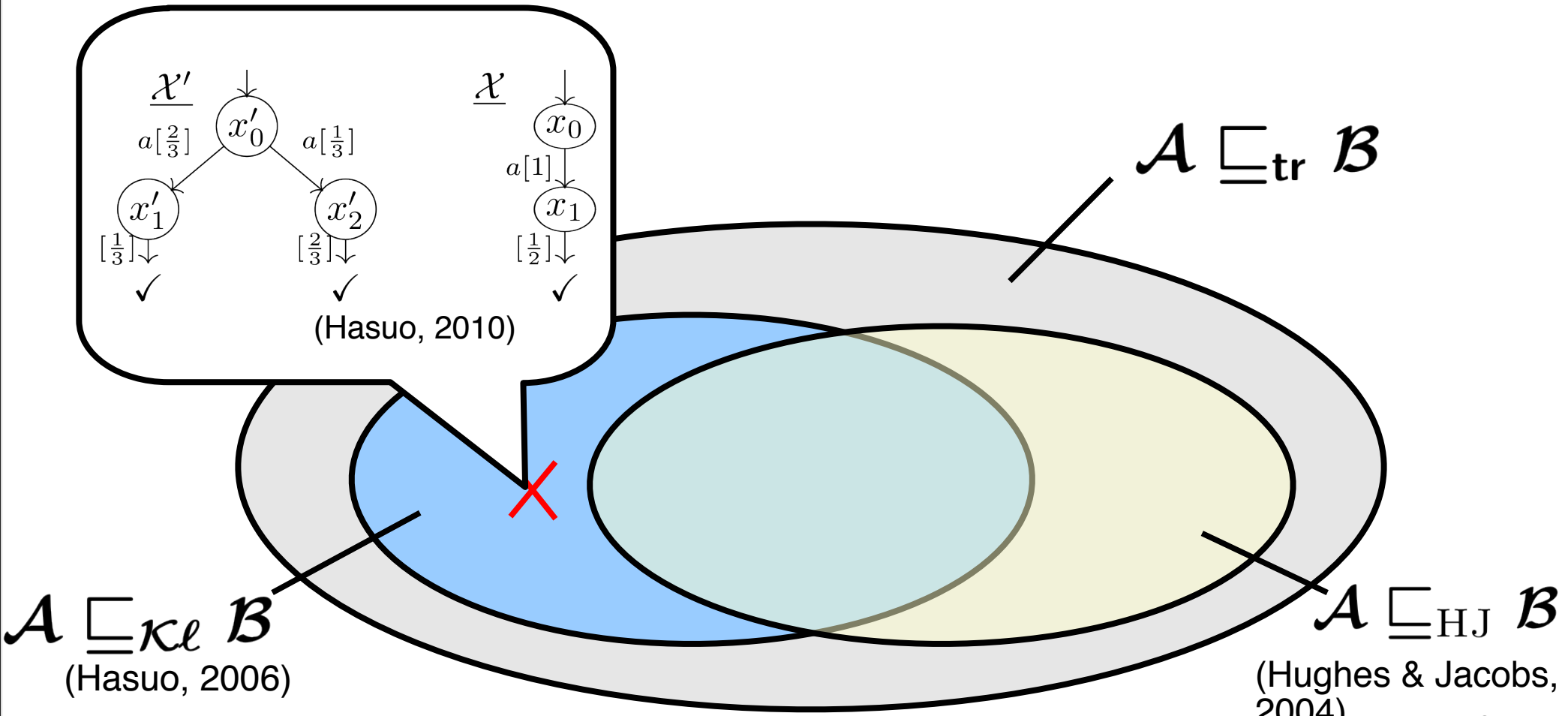
parm			orig		inno				time(sec)		space	iter
n	c	p_f	st	tr	st	tr	al	f/b	user	real	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	fwd	52.48	52.56	0.01	2L
								bwd	0.01	0.06	0.01	2L
7	1	$\frac{3}{4}$	9	88	9	118	26	fwd	0.15	0.21	0.03	2L
								bwd	0.02	0.07	0.01	2L
10	2	$\frac{4}{5}$	12	224	12	280	54	fwd	802.47	803.64	0.35	2L
								bwd	0.05	0.12	0.03	2L
20	6	$\frac{4}{5}$	22	1514	22	1696	238	fwd	TO			
								bwd	1.32	1.51	0.78	2L
30	6	$\frac{4}{5}$	32	4732	32	5112	550	fwd	SF			
								bwd	11.84	12.82	5.99	2L

- Backward simulation is faster

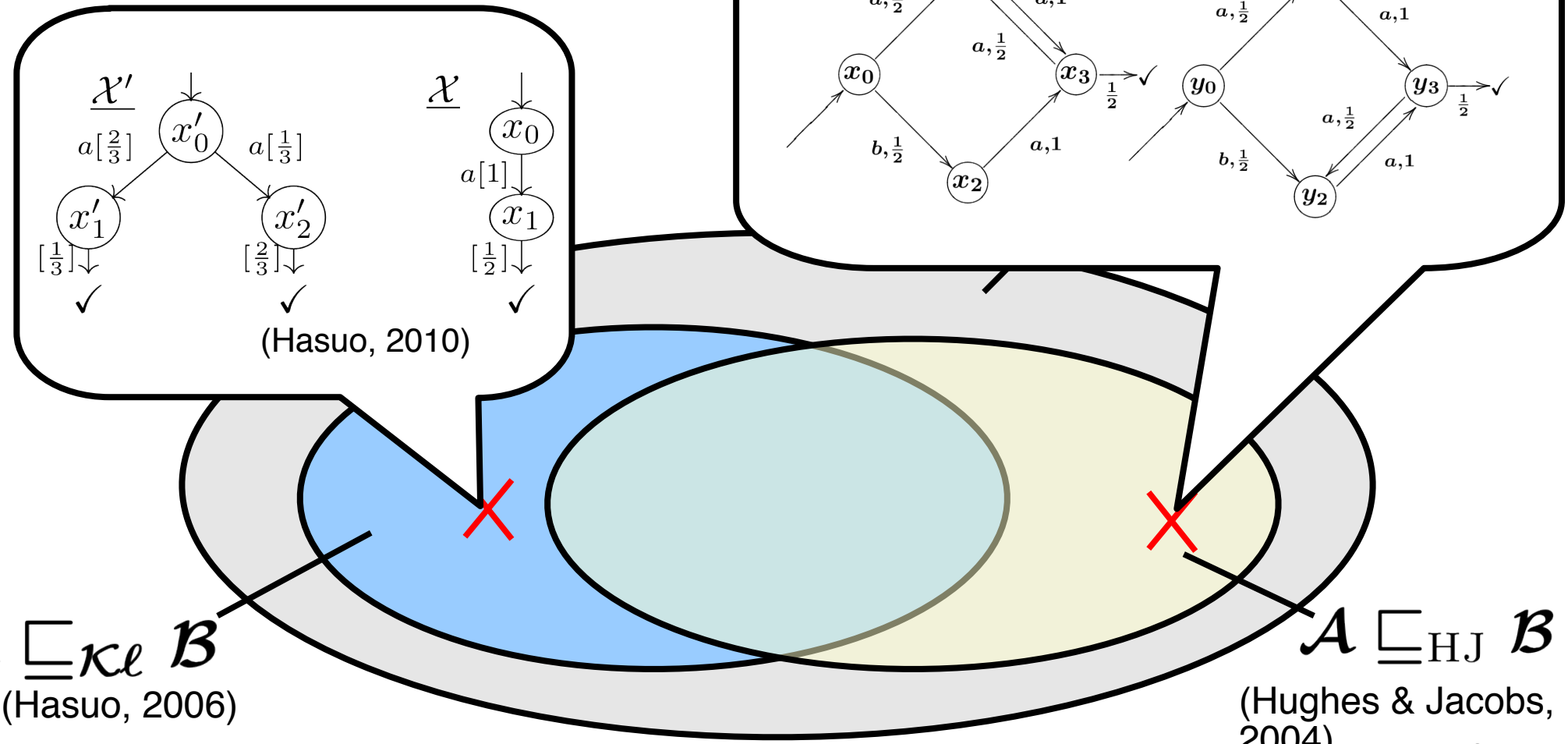
他のSimulationとの比較



他のSimulationとの比較



他のSimulationとの比較



まとめ

- Kleisli simulationを探すプログラムを実装
- Simulation を存在しやすくする工夫

Future Work

- オートマトンの変形を圏論的に
- 他のシステムに対するKleisli simulationの実装